Version 9 Release 1.2
Mai 2016

IBM Interact-Administratorhandbuch

Hinweis

Vor Verwendung dieser Informationen und des darin beschriebenen Produkts sollten die Informationen unter „Bemerkungen" auf Seite 367 gelesen werden.

Diese Ausgabe bezieht sich auf Version 9, Release 1, Modifikation 2 von IBM Interact und alle nachfolgenden Releases und Modifikationen, bis dieser Hinweis in einer Neuausgabe geändert wird.

Diese Veröffentlichung ist eine Übersetzung des Handbuchs
IBM Interact Administrator's Guide, Version 9 Release 1.2,
herausgegeben von International Business Machines Corporation, USA

Informationen, die nur für bestimmte Länder Gültigkeit haben und für Deutschland, Österreich und die Schweiz nicht zutreffen, wurden in dieser Veröffentlichung im Originaltext übernommen.

Möglicherweise sind nicht alle in dieser Übersetzung aufgeführten Produkte in Deutschland angekündigt und verfügbar; vor Entscheidungen empfiehlt sich der Kontakt mit der zuständigen IBM Geschäftsstelle.

Änderung des Textes bleibt vorbehalten.

Herausgegeben von:
TSC Germany
Kst. 2877
Mai 2016

Inhaltsverzeichnis

Anhang C. Interact Designumgebung - Konfigurationseigenschaften. 289

Anhang D. Echtzeit-Personalisierung von Angeboten auf der Clientseite . . 313

Anhang E. Integration von Interact und Digital Recommendations 345

Kapitel 1. Administration von IBM Interact

Im Rahmen Ihrer Administration von Interact konfigurieren und verwalten Sie Benutzer und Rollen, Datenquellen sowie optionale Produktfunktionen. Darüber hinaus überwachen und verwalten Sie die Design- und Laufzeitumgebungen. Hierfür stehen Ihnen produktspezifische Anwendungsprogrammierschnittstellen (APIs) zur Verfügung.

Die Administration von Interact besteht aus mehreren Aufgaben. Zu diesen Aufgaben gehören, ohne darauf beschränkt zu sein, folgende:

- Benutzer und Rollen verwalten
- Datenquellen verwalten
- Optionale Funktionen der Interact-Angebotsunterbreitung konfigurieren
- Laufzeitumgebungsleistung überwachen und warten

Bevor Sie mit der Administration von Interact beginnen, sollten Sie sich mit einigen Schlüsselkonzepten in Bezug auf die Funktionsweise von Interact vertraut machen, damit Sie Ihre Aufgaben einfacher ausführen können. In den folgenden Abschnitten werden die Administrationsaufgaben beschrieben, die mit Interact in Verbindung stehen.

Im zweiten Teil des Administrationshandbuchs finden Sie eine Beschreibung der APIs, die in Interact verfügbar sind:

- Interact-API
- ExternalCallout-API
- Lern-API

Interact-Schlüsselkonzepte

IBM® Interact ist eine interaktive Engine, die personalisierte Marketingangebote an verschiedene Zielgruppen richtet.

Dieser Abschnitt beschreibt die wichtigsten Konzepte, mit denen Sie sich vor dem Arbeiten mit Interact vertraut machen sollten.

Zielgruppenebenen

Eine Zielgruppenebene ist eine Sammlung von IDs, auf die eine Kampagne ausgerichtet werden kann. Sie können Zielgruppenebenen definieren, um Ihre Kampagne auf die richtigen Zielgruppen auszurichten.

Beispielsweise können Kampagnengruppen die Zielgruppenebenen "Haushalt", "Interessent", "Kunde" und "Konto" haben. Jede dieser Ebenen stellt eine bestimmte Ansicht der für eine Kampagne verfügbaren Marketingdaten dar.

Zielgruppenebenen sind gewöhnlich hierarchisch organisiert. Für die obigen Beispiele:

- "Haushalt" steht an der Spitze der Hierarchie und jeder Haushalt kann mehrere Kunden sowie einen oder mehrere Interessenten enthalten.
- Darauf folgt in der Hierarchie "Kunde", und jeder Kunde kann über mehrere Konten verfügen.

- "Konto" ist der niedrigste Hierarchiepunkt.

Weitere, komplexere Beispiele für Zielgruppenhierarchien bestehen in B2B-Umgebungen, wo es möglicherweise Zielgruppenebenen für Unternehmen, Firmen, Abteilungen, Gruppen, Einzelpersonen, Konten usw. gibt.

Diese Zielgruppenebenen können unterschiedliche Beziehungen zueinander haben, beispielsweise "eins-zu-eins", "viele-zu-eins" oder "viele-zu-viele". Durch die Definition von Zielgruppenebenen ermöglichen Sie die Darstellung dieser Konzepte innerhalb von Campaign, sodass Anwender die Beziehungen zwischen diesen verschiedenen Zielgruppen verwalten können, um ihre Kampagnen zielgenauer auszurichten. So möchten Sie vielleicht Mailings auf einen Interessenten pro Haushalt beschränken, obwohl sich in einem Haushalt vielleicht mehrere Interessenten befinden.

Designumgebung

Verwenden Sie die Designumgebung, um verschiedene Interact-Komponenten zu konfigurieren und in der Laufzeitumgebung bereitzustellen.

Den größten Teil der Konfiguration von Interact führen Sie in der Designumgebung aus. In der Designumgebung definieren Sie Ereignisse, Interaktionspunkte, Smart Segments und Verfahrensregeln. Nachdem Sie diese Komponenten konfiguriert haben, stellen Sie sie in der Laufzeitumgebung bereit.

Die Designumgebung wird mit der Campaign-Webanwendung installiert.

Ereignisse

Bei einem Ereignis handelt es sich um eine von einem Besucher ausgeführte Aktion, die eine Aktion in der Laufzeitumgebung auslöst. Ereignisse sind zum Beispiel das Platzieren eines Besuchers in einem Segment, die Darstellung eines Angebots oder das Protokollieren von Daten.

Ereignisse werden zuerst in einem interaktiven Kanal erstellt und dann durch einen Aufruf der Methode `postEvent` an die Interact-API ausgelöst. Ein Ereignis kann zu einer oder mehreren der folgenden Aktionen führen, die in der Interact-Designumgebung definiert sind:

- **Erneute Segmentierung auslösen**. Die Laufzeitumgebung führt erneut alle interaktiven Ablaufdiagramme für die aktuelle Zielgruppenebene aus, die dem interaktiven Kanal zugeordnet ist, und verwendet dazu in der Sitzung des Besuchers die aktuellen Daten.

  Denken Sie beim Entwerfen einer Interaktion daran, dass – sofern Sie kein bestimmtes Ablaufdiagramm angeben – die erneute Segmentierung alle interaktiven Ablaufdiagramme, die dem betreffenden interaktiven Kanal mit der aktuellen Zielgruppenebene zugeordnet sind, erneut ausführt, und dass jede Anforderung von Angeboten wartet, bis alle Ablaufdiagramme beendet sind. Eine übermäßige erneute Segmentierung innerhalb eines einzigen Besuchs kann die Leistung des Touchpoints in einer für den Kunden sichtbaren Art und Weise beeinträchtigen.

  Fügen Sie den Kunden in neue Segmente ein, nachdem signifikante neue Daten zum Laufzeitsitzungsobjekt hinzugefügt wurden. Dies könnten beispielsweise neue Daten von Anforderungen von der Interact-API (wie z. B. Ändern der Zielgruppe) oder von Kundenaktionen (wie z. B. Hinzufügen neuer Artikel zu einer Wunschliste oder einem Warenkorb) sein.

- **Angebotskontakt protokollieren**. Die Laufzeitumgebung kennzeichnet die empfohlenen Angebote für den Datenbankservice, um sie im Kontaktverlauf zu protokollieren.

  Der Touchpoint sollte die Verfahrenscodes für die Angebote, für die Kontakte protokolliert werden sollen, bereitstellen. Wenn es notwendig ist, die Anzahl der Anforderungen zwischen Touchpoint und Laufzeitserver zu begrenzen, ist es möglich, den Angebotskontakt in dem Aufruf, in dem Sie Angebote anfordern, zu protokollieren, ohne einen Verfahrenscode bereitzustellen.

  Wenn der Touchpoint die Verfahrenscodes für die Angebote, die Interact dem Besucher angezeigt hat, nicht zurückgibt, protokolliert die Laufzeitumgebung die letzte Liste von empfohlenen Angeboten.
- **Angebotsannahme protokollieren**. Die Laufzeitumgebung kennzeichnet das ausgewählte Angebot für den Datenbankservice, um es im Antwortverlauf zu protokollieren.
- **Angebotsablehnung protokollieren**. Die Laufzeitumgebung kennzeichnet das ausgewählte Angebot für den Datenbankservice, um es im Antwortverlauf zu protokollieren.
- **Benutzerausdruck auslösen**. Eine *Ausdrucksaktion* ist eine Aktion, die Sie definieren können, indem Sie Interact-Makros, einschließlich Funktionen, Variablen und Operatoren (wie z. B. EXTERNALCALLOUT) verwenden. Sie können den Rückgabewert des Ausdrucks einem beliebigen Profilattribut zuweisen.

  Wenn Sie auf das Symbol "Bearbeiten" neben "Benutzerausdruck auslösen" klicken, wird der Standard-Bearbeitungsdialog "Benutzerausdruck" geöffnet. In diesem Dialog können Sie die Zielgruppenebene, einen optionalen Feldnamen, dem die Ergebnisse zugewiesen werden sollen, und die Definition des Ausdrucks selbst angeben.
- **Ereignisse auslösen**. Über die Aktion "Ereignisse auslösen" können Sie den Namen eines Ereignisses eingeben, das durch diese Aktion ausgelöst werden soll. Wenn Sie ein Ereignis eingeben, das bereits definiert ist, wird bei der Ausführung dieser Aktion dieses Ereignis ausgelöst. Wenn der von Ihnen eingegebene Ereignisname nicht existiert, führt diese Aktion dazu, dass das Ereignis mit der angegebenen Aktion erstellt wird.

Mit Ereignissen können Sie außerdem Aktionen auslösen, die in der Methode `postEvent` definiert sind, z. B. das Protokollieren von Daten in einer Tabelle (einschließlich Daten zum Lernen) oder das Auslösen individueller Ablaufdiagramme.

Ereignisse lassen sich in der Designumgebung bei Bedarf in Kategorien einteilen. Kategorien haben in der Laufzeitumgebung keine bestimmte Funktion.

Interaktive Kanäle

Verwenden Sie die interaktiven Kanäle in Interact, um alle Objekte, Daten und Serverressourcen zu koordinieren, die am interaktiven Marketing beteiligt sind.

Ein interaktiver Kanal ist die Darstellung eines Touchpoints in Campaign, wobei die Benutzeroberflächenmethode ein interaktiver Dialog ist. Diese Softwaredarstellung wird zum Koordinieren aller Objekte, Daten und Serverressourcen verwendet, die mit dem interaktiven Marketing verbunden sind.

Ein interaktiver Kanal ist ein Tool, das Sie zum Definieren von Interaktionspunkten und Ereignissen verwenden. Über die Registerkarte "Analyse" eines interaktiven Kanals können Sie außerdem auf Berichte für diesen interaktiven Kanal zugreifen.

Interaktive Kanäle enthalten zudem Produktionslaufzeit- und Staging-Serverzuordnungen. Sie können mehrere interaktive Kanäle erstellen, um die Ereignisse und Interaktionspunkte zu gliedern, wenn Sie über nur einen Satz von Produktionslaufzeit- und Staging-Servern verfügen, oder um die Ereignisse und Interaktionspunkte nach kundenorientierten Systemen zu unterteilen.

Interaktive Ablaufdiagramme

Verwenden Sie interaktive Ablaufdiagramme, um die Kunden in Segmente zu unterteilen und den Segmenten jeweils ein Profil zuzuweisen.

Ein interaktives Ablaufdiagramm ist einem Campaign-Ablaufdiagramm zur Stapelverarbeitung ähnlich, unterscheidet sich aber ein wenig von diesem. Interaktive Ablaufdiagramme haben im Wesentlichen dieselbe Funktion wie Ablaufdiagramme zur Stapelverarbeitung: Sie teilen die Kunden in Gruppen auf, die als Segmente bezeichnet werden. Im Falle interaktiver Ablaufdiagramme handelt es sich bei diesen Gruppen jedoch um Smart Segments. Interact verwendet diese interaktiven Ablaufdiagramme, um ein Profil einem Segment zuzuordnen, wenn ein verhaltensabhängiges Ereignis oder ein Systemereignis anzeigt, dass eine erneute Segmentierung der Besucher erforderlich ist.

Interaktive Ablaufdiagramme enthalten ein Subset der Prozesse der Ablaufdiagramme zur Stapelverarbeitung sowie einige für interaktive Ablaufdiagramme spezifische Prozesse.

Anmerkung: Interaktive Ablaufdiagramme können nur während einer Campaign-Sitzung erstellt werden.

Interaktionspunkte

Ein Interaktionspunkt ist ein Ort im Touchpoint, an dem Sie ein Angebot anzeigen möchten.

Interaktionspunkte verfügen über Standardinhalt, der angezeigt wird, falls die Laufzeitumgebung keinen anderen passenden Inhalt bereitstellen kann. Interaktionspunkte können in Zonen gegliedert werden.

Angebote

Ein Angebot stellt eine einzelne Marketingnachricht dar, die über unterschiedliche Kanäle übermittelt werden kann.

In Campaign erstellte Angebote können in einer oder mehreren Kampagnen verwendet werden.

Angebote können wie folgt wiederverwendet werden:

- In verschiedenen Kampagnen
- Zu verschiedenen Zeitpunkten
- Für verschiedene Personengruppen (Zellen)
- Als unterschiedliche "Versionen", bei denen sich die Felder mit Parameterangabe des Angebots unterscheiden

Sie weisen Angeboten Interaktionspunkte in den Touchpoints zu, die Besuchern präsentiert werden.

Profile

Als Profil bezeichnet man den Satz Kundendaten, den die Laufzeitumgebung verwendet. Diese Daten können ein Subset der in der Kundendatenbank enthaltenen Kundendaten sein oder in Echtzeit erfasste Daten bzw. eine Kombination aus beidem.

Die Kundendaten werden zu folgenden Zwecken verwendet:

- Um einen Kunden in Echtzeitinteraktionsszenarien einem oder mehreren Smart Segments zuzuordnen.

  Sie benötigen einen Satz Profildaten für jede Zielgruppenebene, nach der Sie segmentieren möchten. Ein Beispiel: Um nach "Ort" zu segmentieren, können Sie aus den gesamten Adressdaten des Kunden nur die Postleitzahl hinzufügen.
- Um Angebote zu personalisieren
- Als Attribute, die zu Lernzwecken verfolgt werden sollen

  Beispiel: Sie können Interact so konfigurieren, dass es den Familienstand eines Besuchers nachverfolgt und erfasst, wie viele Besucher der einzelnen Familienstände bestimmte Angebote annehmen. Die Laufzeitumgebung kann anhand dieser Informationen dann die Angebotsauswahl optimieren.

Diese Daten sind für die Laufzeitumgebung schreibgeschützt.

Laufzeitumgebung

Die Laufzeitumgebung stellt eine Verbindung zu Ihrem Touchpoint her und führt Interaktionen aus. Die Laufzeitumgebung kann aus einem oder mehreren Laufzeitservern mit Verbindung zu einem Touchpoint bestehen.

Die Laufzeitumgebung verwendet die von der Designumgebung bereitgestellten Informationen zusammen mit der Interact-API, um Angebote für den Touchpoint anzuzeigen.

Laufzeitsitzungen

Für jeden Besucher Ihres Touchpoints existiert eine Laufzeitsitzung auf dem Server für die Laufzeitumgebung. Diese Sitzung enthält alle Besucherdaten, die die Laufzeitumgebung verwendet, um die Besucher Segmenten zuzuordnen und Angebote zu empfehlen.

Sie erstellen eine Laufzeitsitzung mit dem Aufruf `startSession`.

Touchpoints

Ein Touchpoint ist eine Anwendung, in der bzw. ein Ort, an dem eine Interaktion mit einem Kunden erfolgt. Ein Touchpoint kann ein Kanal sein, in dem der Kunde den Kontakt einleitet (eine "Inbound"-Interaktion) oder in dem Sie den Kunden kontaktieren (eine "Outbound"-Interaktion).

Häufige Beispiele sind Websites und Call Center-Anwendungen. Mithilfe der Interact-API können Sie Interact in Ihre Touchpoints integrieren, um dem Kunden basierend auf seiner Aktion im Touchpoint Angebote anzuzeigen. Touchpoints werden auch als kundenorientierte Systeme (CFS, Client Facing Systems) bezeichnet.

Verfahrensregeln

Verfahrensregeln ordnen ein Angebot einem Smart Segment zu. Diese Zuordnungen werden durch die benutzerdefinierte Zone, die Sie dem Angebot in der Verfahrensregel zuweisen, weiter eingeschränkt.

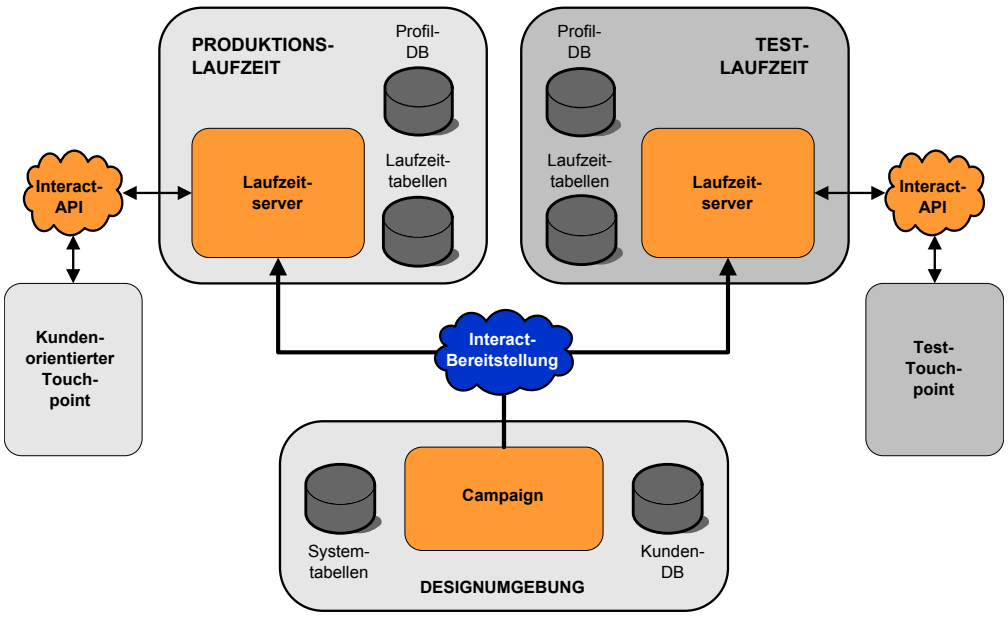
So können Sie zum Beispiel über eine Auswahl von Angeboten verfügen, die Sie einem Smart Segment in der Zone "Anmeldung" zuordnen, sowie über eine weitere Auswahl von Angeboten für dasselbe Segment in der Zone "Nach dem Kauf". Verfahrensregeln werden auf einer Registerkarte "Interaktionsstrategie" einer Kampagne definiert.

Jede Verfahrensregel verfügt außerdem über einen Marketing-Score. Wenn ein Kunde mehreren Segmenten zugeordnet ist und daher mehr als ein Angebot in Frage kommt, wird mithilfe des Marketing-Score ermittelt, welches Angebot von Interact vorgeschlagen wird. Die von der Laufzeitumgebung vorgeschlagenen Angebote können von einem Lernmodul, einer Liste für Angebotsunterdrückung sowie globalen und individuellen Angebotszuweisungen beeinflusst werden.

Interact-Architektur

Die Interact-Umgebung besteht aus mindestens zwei wichtigen Komponenten, der Designumgebung und der Laufzeitumgebung. Möglicherweise setzen sie außerdem optionale Server zum Testen der Laufzeitumgebung ein.

Die folgende Abbildung zeigt einen allgemeinen Überblick über die Architektur.



In der Designumgebung führen Sie den größten Teil der Interact-Konfiguration aus. Die Designumgebung wird mit derCampaign- Webanwendung installiert und verweist auf die Campaign-Systemtabellen und Ihre Kundendatenbanken. Sie verwenden die Designumgebung, um die Interaktionspunkte und Ereignisse zu definieren, die Sie mit der API verwenden.

Nachdem Sie festgelegt und konfiguriert haben, wie Kundeninteraktionen von der Laufzeitumgebung verarbeitet werden sollen, stellen Sie diese Daten entweder ei-

ner Staging-Servergruppe zum Testen oder einer Servergruppe für die Produktionsumgebung für Kundeninteraktionen in Echtzeit bereit.

Die Interact-API stellt die Verbindung zwischen Ihrem Touchpoint und dem Laufzeitserver bereit. Sie referenzieren Objekte (Interaktionspunkte und Ereignisse), die in der Designumgebung erstellt werden, mit der Interact-API und verwenden sie, um Informationen aus dem Laufzeitserver anzufordern.

Überlegungen zum Interact-Netz

Eine Produktionsinstallation von Interact umfasst zumindest zwei Maschinen. In einer hochvolumigen Produktionsumgebung mit mehreren Interact-Laufzeitservern und verteilten Datenbanken kann Ihre Installation Dutzende Maschinen umfassen.

Um die beste Leistung zu erhalten, sind mehrere Netztopologie-Anforderungen zu berücksichtigen.

- Wenn Ihre Implementierung der Interact-API Sitzungen im selben Aufruf startet und beendet, zum Beispiel:

  ```
  executeBatch(startSession, getOffers, postEvent, endSession)
  ```

  müssen Sie keine Sitzungspersistenz (permanente Sitzungen) zwischen der Lastausgleichsfunktion und den Interact-Laufzeitservern aktivieren. Sie können das Interact-Laufzeitserversitzungsmanagement für den lokalen Cachetyp konfigurieren.
- Wenn Ihre Implementierung der Interact-API mehrere Aufrufe verwendet, um Sitzungen zu starten und zu beenden, zum Beispiel:

  ```
  startSession
  . . .
  executeBatch(getOffers, postEvent)
  . . .
  endSession
  ```

  und Sie ein Programm für den Lastausgleich für Ihre Interact-Laufzeitserver verwenden, sollten Sie eine Art von Persistenz für die Lastausgleichsfunktion ermöglichen (auch permanente Sitzungen genannt). Wenn das nicht möglich ist oder wenn Sie keine Lastausgleichsfunktion verwenden, konfigurieren Sie das Interact-Serversitzungsmanagement für einen verteilten `cacheType`. Wenn Sie einen verteilten Cache verwenden, müssen alle Interact-Laufzeitserver in der Lage sein, über Multicasting zu kommunizieren. Sie müssen möglicherweise Ihr Netz optimieren, damit die Kommunikation zwischen Interact-Servern, die dieselbe Multicast-IP-Adresse und Port verwenden, nicht die Systemleistung behindert. Eine Lastausgleichsfunktion mit permanenten Sitzungen bietet eine bessere Leistung als die Verwendung eines verteilten Cache.
- Wenn Sie mehrere Servergruppen haben, die einen verteilten `cacheType` verwenden, sollte jede einen eindeutigen Multicast-Port verwenden. Die Verwendung eines eindeutigen Multicast-Ports und einer eindeutigen Multicast-Adresse für jede Servergruppe ist besser.
- Die beste Leistung erhalten Sie, wenn sich Ihre Laufzeitumgebung (Interact-Server, Marketing Platform-Lastenausgleichsprogramme und Touchpoint) an einem Standort befindet. Die Designumgebung und die Laufzeitumgebung können an verschiedenen Standorten sein, was aber zu einer langsamen Implementierung führen kann.
- Richten Sie eine schnelle Netzverbindung (mindestens 1 Gb) zwischen der Interact-Produktionsservergruppe und ihrem zugehörigen Touchpoint ein.
- Die Designumgebung erfordert http- oder https-Zugriff auf die Laufzeit, um Implementierungsaufgaben durchzuführen. Firewalls oder andere Netzanwendun-

gen müssen so konfiguriert sein, dass sie die Implementierung ermöglichen. Sie müssen möglicherweise die Dauer der HTTP-Zeitlimits zwischen der Designumgebung und den Laufzeitumgebungen verlängern, wenn Sie umfangreiche Implementierungen haben.

- Das Kontakt- und Antwortverlaufsmodul erfordert Zugriff auf die Designzeitdatenbank (Campaign-Systemtabellen) und Zugriff auf die Laufzeitdatenbank (Interact-Laufzeittabellen). Sie müssen Ihre Datenbank und Ihr Netz entsprechend konfigurieren, damit diese Datenübertragung stattfinden kann.

In einer Test- oder Staging-Installation können Sie die Interact-Designumgebung und die Laufzeitumgebung auf derselben Maschine installieren. Dieses Szenario wird nicht für Produktionsumgebungen empfohlen.

Anmelden bei IBM EMM

Verwenden Sie diese Prozedur, um sich bei IBM EMM anzumelden.

Vorbereitende Schritte

Sie benötigen das Folgende.

- Eine Intranet-(Netz-)Verbindung, um auf den IBM EMM-Server zuzugreifen.
- Einen auf dem Computer installierten Browser, der auch unterstützt wird.
- Benutzername und Kennwort, damit Sie sich bei IBM EMM anmelden können.
- Die URL, um im Netz auf IBM EMM zuzugreifen.

Die URL ist:

`http://`*`host.domain.com:port`*`/unica`

wobei

`host` die Maschine ist, auf der Marketing Platform installiert ist.

`domain.com` die Domäne ist, in der sich die Hostmaschine befindet.

`port` die Portnummer ist, auf der die Marketing Platform-Anwendung empfangsbereit ist.

Anmerkung: Die folgende Prozedur setzt voraus, dass Sie sich mit einem Konto anmelden, das für den Zugriff auf Marketing Platform über Administratorrechte verfügt.

Vorgehensweise

Greifen Sie über den Browser auf die IBM EMM-URL zu.

- Falls IBM EMM für die Integration mit Windows Active Directory oder mit einer Plattform zur Webzugriffssteuerung konfiguriert ist und Sie bei diesem System angemeldet sind, wird die Standarddashboardseite angezeigt. Ihre Anmeldung ist abgeschlossen.
- Wenn die Anmeldeanzeige angezeigt wird, melden Sie sich mit den Standardberechtigungsnachweisen für Administratoren an. Verwenden Sie in einer Umgebung mit nur einer Partition `asm_admin` mit `password` als Kennwort. Verwenden Sie in einer Umgebung mit mehreren Partitionen `platform_admin` mit `password` als Kennwort.

Sie werden aufgefordert, das Kennwort zu ändern. Sie können das vorhandene Kennwort verwenden, aber aus Sicherheitsgründen sollten Sie ein neues festlegen.

- Falls IBM EMM für die Verwendung mit SSL konfiguriert ist, werden Sie bei der erstmaligen Anmeldung eventuell aufgefordert, ein digitales Sicherheitszertifikat anzunehmen. Klicken Sie auf **Ja**, um das Zertifikat anzunehmen.

War die Anmeldung erfolgreich, zeigt IBM EMM die Standarddashboardseite an.

Ergebnisse

Mit den Marketing Platform-Administratorkonten zugeordneten Standardberechtigungen können Sie mithilfe der im Menü **Einstellungen** aufgeführten Optionen Benutzerkonten und Sicherheitsaspekte verwalten. Wenn Sie für IBM EMM-Dashboards Administrationsaufgaben auf der höchsten Ebene ausführen möchten, müssen Sie sich als **platform_admin** anmelden.

Kapitel 2. Konfigurieren von IBM Interact-Benutzern

Für Interact müssen zwei Benutzergruppen eingerichtet werden: Laufzeitumgebungsbenutzer und Designzeitumgebungsbenutzer.

- **Laufzeitbenutzer** werden in Marketing Platform erstellt und für das Arbeiten mit den Laufzeitservern konfiguriert.
- **Designzeitbenutzer** sind Campaign-Benutzer. Sie konfigurieren die Sicherheit für die verschiedenen Mitglieder Ihres Designteams wie für Campaign.

Den Laufzeitumgebungsbenutzer konfigurieren

Nach der Installation von Interact müssen Sie zumindest einen Interact-Benutzer, den Laufzeitumgebungsbenutzer, konfigurieren. Laufzeitbenutzer werden in Marketing Platform erstellt.

Informationen zu diesem Vorgang

Der Laufzeitumgebungsbenutzer bietet Zugriff auf die Laufzeittabellen. Als Laufzeitumgebungsbenutzer werden der Benutzername und das Kennwort angegeben, mit denen Sie interaktive Kanäle bereitstellen. Der Laufzeitserver verwendet die Webanwendungsserver-JDBC-Authentifizierung für die Berechtigungsnachweise für die Datenbank. Sie müssen dem Laufzeitumgebungsbenutzer keine Datenquellen der Laufzeitumgebung hinzufügen.

Beachten Sie bei der Erstellung von Laufzeitbenutzern Folgendes:

- Bei eigenen Marketing Platform-Instanzen für jeden Laufzeitserver müssen Sie jeweils denselben Benutzer und dasselbe Kennwort erstellen. Für alle derselben Servergruppe angehörenden Laufzeitserver müssen dieselben Benutzerberechtigungsnachweise angegeben werden.
- Wenn Sie ein Datenbankladedienstprogramm verwenden, müssen Sie die Laufzeittabellen als eine Datenquelle mit Anmeldeberechtigungsnachweisen für den Laufzeitumgebungsbenutzer in Ihren Konfigurationseigenschaften unter `Interact > Allgemein > systemTablesDataSource` definieren.
- Wenn Sie die Sicherheit für JMX-Überwachung mit dem JMXMP-Protokoll einrichten, ist eventuell ein eigener Benutzer für die Sicherheit der JMX-Überwachung erforderlich.

In der Dokumentation zu Marketing Platform finden Sie die Schritte, die zur Erstellung von Laufzeitbenutzern ausgeführt werden müssen.

Designumgebungsbenutzer konfigurieren

Designumgebungsbenutzer sind Campaign-Benutzer. Sie konfigurieren Ihre Designumgebungsbenutzer auf die gleiche Weise, wie Sie Campaign-Rollenberechtigungen konfigurieren.

Informationen zu diesem Vorgang

Für bestimmte Designumgebungsbenutzer sind auch gewisse Campaign-Berechtigungen wie z. B. benutzerdefinierte Makros erforderlich.

Beachten Sie bei der Erstellung von Designumgebungsbenutzern Folgendes:

- Wenn Sie über Campaign-Benutzer verfügen, die zum Bearbeiten von interaktiven Ablaufdiagrammen berechtigt sind, müssen Sie ihnen den Zugriff auf die Testlauftabellendatenquelle erteilen.
- Wenn Interact installiert und konfiguriert ist, sind die unten genannten Zusatzoptionen für die standardmäßige globale Richtlinie und neue Richtlinien verfügbar.
-

Kategorie	Berechtigung
Kampagnen	• Anzeigen von Kampagneninteraktionsstrategien - Der Benutzer kann die Registerkarten des Typs "Interaktionsstrategie" einer Kampagne anzeigen, aber nicht bearbeiten. • Bearbeiten von Kampagneninteraktionsstrategien - Der Benutzer kann die Registerkarten des Typs "Interaktionsstrategie" ändern, einschließlich Verfahrensregeln. • Löschen von Kampagneninteraktionsstrategien - Der Benutzer kann die Registerkarten des Typs "Interaktionsstrategie" aus Kampagnen löschen. Das Löschen einer Registerkarte des Typs "Interaktionsstrategie" ist eingeschränkt, wenn die Interaktionsstrategie in eine Bereitstellung eines interaktiven Kanals aufgenommen wurde. • Hinzufügen von Kampagneninteraktionsstrategien - Der Benutzer kann neue Registerkarten des Typs "Interaktionsstrategie" für eine Kampagne erstellen. • Bereitstellung von Kampagneninteraktionsstrategien zur Initialisierung - Der Benutzer kann eine Registerkarte des Typs "Interaktionsstrategie" zur Bereitstellung oder Zurücknehmen der Bereitstellung markieren.
Interaktive Kanäle	• Interaktive Kanäle bereitstellen - Der Benutzer kann einen interaktiven Kanal für die Interact-Laufzeitumgebungen bereitstellen. • Interaktive Kanäle bearbeiten - Der Benutzer kann die Übersichtsregisterkarte von interaktiven Kanälen ändern. • Interaktive Kanäle löschen - Der Benutzer kann interaktive Kanäle löschen. Das Löschen von interaktiven Kanälen ist eingeschränkt, wenn der interaktive Kanal bereitgestellt worden ist. • Interaktive Kanäle anzeigen - Der Benutzer kann interaktive Kanäle anzeigen, aber nicht bearbeiten. • Interaktive Kanäle hinzufügen - Der Benutzer kann neue interaktive Kanäle hinzufügen. • Berichte zu interaktiven Kanälen anzeigen - Der Benutzer kann die Registerkarte "Analyse" des interaktiven Kanals anzeigen. • Untergeordnete Objekte zum interaktiven Kanal hinzufügen - Der Benutzer kann Interaktionspunkte, Zonen, Ereignisse und Kategorien hinzufügen.

Kategorie	Berechtigung
Sitzungen	• Interaktive Ablaufdiagramme anzeigen - Der Benutzer kann ein interaktives Ablaufdiagramm in einer Sitzung anzeigen. • Interaktive Ablaufdiagramme hinzufügen - Der Benutzer kann neue interaktive Ablaufdiagramme in einer Sitzung erstellen. • Interaktive Ablaufdiagramme bearbeiten - Der Benutzer kann interaktive Ablaufdiagramme ändern. • Interaktive Ablaufdiagramme löschen - Der Benutzer kann interaktive Ablaufdiagramme löschen. Das Löschen von interaktiven Ablaufdiagrammen ist eingeschränkt, wenn der Kanal, dem dieses interaktive Ablaufdiagramm zugeordnet ist, bereitgestellt worden ist. • Interaktive Ablaufdiagramme kopieren - Der Benutzer kann interaktive Ablaufdiagramme kopieren. • Testlauf für interaktive Ablaufdiagramme ausführen - Der Benutzer kann einen Testlauf eines interaktiven Ablaufdiagramms durchführen. • Interaktive Ablaufdiagramme prüfen - Der Benutzer kann ein interaktives Ablaufdiagramm prüfen und Prozesse zur Ansicht von Einstellungen öffnen, aber nicht ändern. • Interaktive Ablaufdiagramme bereitstellen - Der Benutzer kann ein interaktives Ablaufdiagramm zur Bereitstellung oder Zurücknehmen der Bereitstellung markieren.

Beispiel für Designumgebungsberechtigungen

In diesem Beispiel werden die Berechtigungen aufgelistet, die für zwei unterschiedliche Rollen erteilt werden: Hierbei handelt es sich um eine Rolle für die Benutzer, die interaktive Ablaufdiagramme erstellen, und um eine Rolle für die Benutzer, die die Interaktionsstrategien definieren.

Rolle für interaktive Ablaufdiagramme

Diese Tabelle enthält die Berechtigungen, die der Rolle für interaktive Ablaufdiagramme erteilt werden:

Kategorie	Berechtigung
Benutzerdefiniertes Makro	Die Benutzerrolle verfügt über folgende Berechtigungen: • Benutzerdefinierte Makros hinzufügen • Benutzerdefinierte Makros bearbeiten • Benutzerdefinierte Makros verwenden
Abgeleitetes Feld	Die Benutzerrolle verfügt über folgende Berechtigungen: • Abgeleitete Felder hinzufügen • Abgeleitete Felder bearbeiten • Abgeleitete Felder verwenden
Ablaufdiagrammvorlage	Die Benutzerrolle verfügt über folgende Berechtigungen: • Vorlagen einfügen

Kategorie	Berechtigung
Segmentvorlage	Die Benutzerrolle verfügt über folgende Berechtigungen: • Segmente hinzufügen • Segmente bearbeiten
Sitzung	Die Benutzerrolle verfügt über folgende Berechtigungen: • Sitzungsübersicht anzeigen • Interaktive Ablaufdiagramme anzeigen • Interaktive Ablaufdiagramme hinzufügen • Interaktive Ablaufdiagramme bearbeiten • Interaktive Ablaufdiagramme kopieren • Test für interaktive Ablaufdiagramme ausführen • Interaktive Ablaufdiagramme bereitstellen

Rolle für Interaktionsstrategien

Diese Tabelle enthält die Berechtigungen, die der Rolle für Interaktionsstrategien erteilt werden:

Kategorie	Berechtigung
Kampagne	Die Benutzerrolle verfügt über folgende Berechtigungen: • Kampagnenübersicht anzeigen • Zielzellen von Kampagnen verwalten • Kampagneninteraktionsstrategien anzeigen • Kampagneninteraktionsstrategien bearbeiten • Kampagneninteraktionsstrategien hinzufügen • Bereitstellung von Kampagneninteraktionsstrategien initiieren
Angebot	Die Benutzerrolle verfügt über folgende Berechtigungen: • Angebotsübersicht anzeigen
Segmentvorlage	Die Benutzerrolle verfügt über folgende Berechtigungen: • Segmentübersicht anzeigen
Sitzung	Die Benutzerrolle verfügt über folgende Berechtigungen: • Interaktive Ablaufdiagramme prüfen

Kapitel 3. Verwalten von Interact-Datenquellen

Interact erfordert mehrere Datenquellen, um ordnungsgemäß zu funktionieren. Einige Datenquellen enthalten die Informationen, die Interact zum Funktionieren benötigt, und andere Datenquellen enthalten Ihre Daten.

Die folgenden Abschnitte beschreiben die Interact-Datenquellen, einschließlich Informationen, die Sie benötigen, um sie ordnungsgemäß zu konfigurieren, sowie einiger Hinweise zu ihrer Wartung.

Interact-Datenquellen

Interact erfordert mehrere Gruppen von Daten, um zu funktionieren. Die Datengruppen werden gespeichert und von Datenquellen abgerufen. Welche Datenquellen Sie einrichten, hängt davon ab, welche Interact-Funktionen Sie aktivieren.

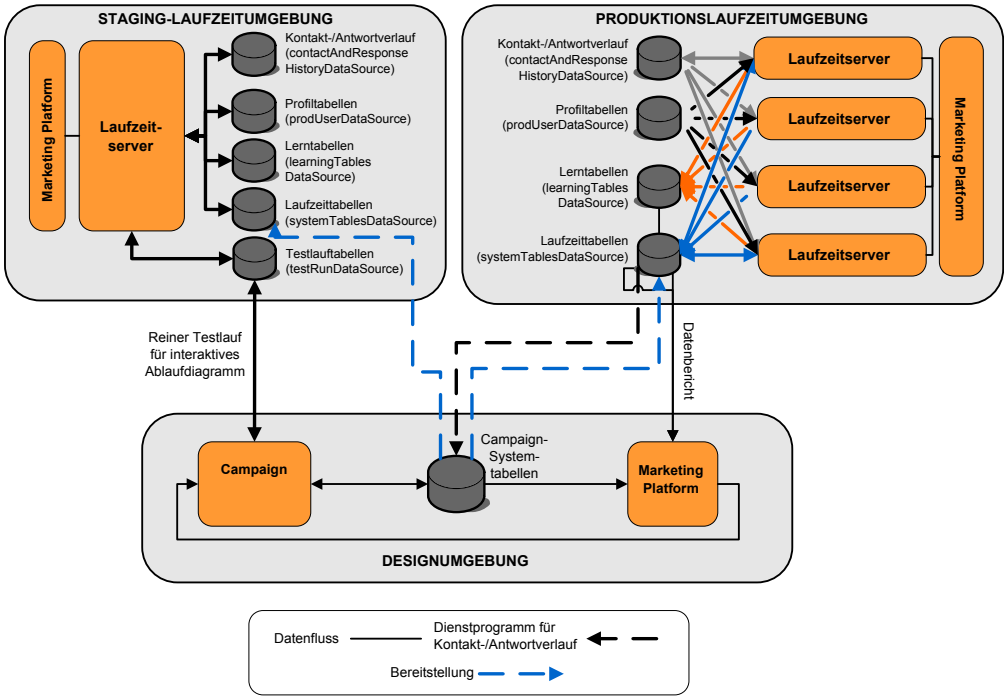
- **Campaign-Systemtabellen**. Neben allen Daten für Campaign enthalten die Campaign-Systemtabellen Daten für Interact-Komponenten, die Sie in der Designumgebung erstellen, wie z. B. Verfahrensregeln und interaktive Kanäle. Die Designumgebung und die Campaign-Systemtabellen nutzen dieselbe physische Datenbank und dasselbe Schema.
- **Laufzeittabellen** (`systemTablesDataSource`). Diese Datenquelle enthält die Bereitstellungsdaten aus der Designumgebung, Staging-Tabellen für Kontakt- und Antwortverlauf sowie Laufzeitstatistikdaten.
- **Profiltabellen** (`prodUserDataSource`). Diese Datenquelle enthält - neben den in Echtzeit erfassten Informationen - alle Kundendaten, die von interaktiven Ablaufdiagrammen benötigt werden, um Besucher ordnungsgemäß in Smart Segments zu platzieren. Wenn Sie sich ausschließlich auf Echtzeitdaten verlassen, benötigen Sie keine Profiltabellen. Wenn Sie Profiltabellen verwenden, müssen Sie mindestens eine Profiltabelle pro vom interaktiven Kanal verwendeter Zielgruppenebene haben.

  Die Profiltabellen können auch die Tabellen zum Erweitern der Angebotsbereitstellung enthalten, einschließlich der Tabellen für Angebotsunterdrückung, Bewertungsüberschreibung sowie globaler und individueller Angebotszuweisung.
- **Testlauftabellen** (`testRunDataSource`). Diese Datenquelle enthält ein Muster aller Daten, die von Ablaufdiagrammen benötigt werden, um Besucher in Smart Segments zu platzieren, einschließlich Daten, die nachahmen, was in Echtzeit während einer Interaktion erfasst wird. Diese Tabellen sind nur für die Servergruppe erforderlich, die als Testlaufservergruppe für die Designumgebung bestimmt ist.
- **Lerntabellen** (`learningTablesDataSource`). Diese Datenquelle enthält alle Daten, die vom integrierten Lerndienstprogramm gesammelt werden. Zu diesen Tabellen kann auch eine Tabelle gehören, die dynamische Attribute definiert. Wenn Sie die Lernfunktion nicht einsetzen oder ein externes Lerndienstprogramm verwenden, benötigen Sie keine Lerntabellen.
- **Kontakt- und Antwortverlauf für sitzungsübergreifende Antwort** (`contactAndResponseHistoryDataSource`). Diese Datenquelle enthält entweder die Campaign-Kontaktverlaufstabellen oder Kopien davon. Wenn Sie die sitzungsübergreifende Antwortfunktion nicht verwenden, müssen Sie diese Kontaktverlaufstabellen nicht konfigurieren.

Datenbanken und die Anwendungen

Die Datenquellen, die Sie erstellen, damit sie von Interact verwendet werden, können auch dazu eingesetzt werden, Daten mit anderen IBM EMM-Anwendungen auszutauschen oder mit ihnen zu teilen.

Das nachfolgende Diagramm zeigt Interact-Datenquellen und ihre Beziehung zu IBM EMM-Anwendungen.



- Sowohl Campaign als auch die Testlaufservergruppe greifen auf die Testlauftabellen zu.
- Die Testlauftabelle wird nur für Testläufe interaktiver Ablaufdiagramme verwendet.
- Wenn Sie einen Laufzeitserver zum Testen einer Implementierung (einschließlich der Interact-API) verwenden, verwendet der Laufzeitserver die Profiltabellen für die Daten.
- Wenn Sie das Kontakt- und Antwortverlaufsmodul konfigurieren, verwendet das Modul einen ETL-Hintergrundprozess (Extrahieren, Transformieren, Laden), um Daten aus den Laufzeit-Staging-Tabellen in die Campaign-Kontakt- und -Antwortverlaufstabellen zu verschieben.
- Die Funktion zur Berichterstellung fragt Daten aus den Lerntabellen, den Laufzeittabellen und den Campaign-Systemtabellen ab, um Berichte in Campaign anzuzeigen.

Sie sollten die Testlaufzeitumgebung so konfigurieren, dass ein anderer Tabellensatz als in Ihren Produktionsumgebungen verwendet wird. Mit separaten Tabellen für das Staging und die Produktion können Sie Ihre Testergebnisse von Ihren tatsächlichen Ergebnissen getrennt halten. Bedenken Sie, dass das Kontakt- und Antwortverlaufsmodul immer Daten in die tatsächlichen Campaign-Kontakt- und -Antwortverlaufstabellen einfügt (Campaign hat keine Testkontakt- und Antwortverlaufstabellen). Wenn Sie separate Lerntabellen für die Testlaufzeitumgebung ha-

ben und die Ergebnisse in Berichten sehen wollen, benötigen Sie eine separate Instanz von IBM Cognos BI, um die Lernberichte für die Testumgebung auszuführen.

Campaign-Systemtabellen

Wenn Sie die Interact-Designumgebung installieren, erstellen Sie auch neue, Interact-spezifische Tabellen in den Campaign-Systemtabellen. Welche Tabellen Sie erstellen, hängt davon ab, welche Interact-Funktionen Sie aktivieren.

Wenn Sie das Kontakt- und Antwortverlaufsmodul aktivieren, kopiert das Modul den Kontakt- und Antwortverlauf aus den Staging-Tabellen in den Laufzeittabellen in die Kontakt- und Antwortverlaufstabellen in den Campaign-Systemtabellen. Die Standardtabellen sind `UA_ContactHistory`, `UA_DtlContactHist` und `UA_ResponseHistory`, aber das Kontakt- und Antwortverlaufsmodul verwendet jeweils Tabellen, die in Campaign für die Kontakt- und Antwortverlaufstabellen zugeordnet sind.

Wenn Sie die globalen Angebotstabellen und Bewertungsüberschreibungstabellen verwenden, um Angebote zuzuweisen, müssen Sie möglicherweise die Tabelle `UACI_ICBatchOffers` in den Campaign-Systemtabellen auffüllen, wenn Sie Angebote verwenden, die nicht in den Verfahrensregeln für den interaktiven Kanal enthalten sind.

Laufzeittabellen

Wenn mehr als eine Zielgruppenebene definiert ist, müssen Sie Staging-Tabellen für die Kontakt- und Antwortverlaufsdaten für jede Zielgruppenebene erstellen.

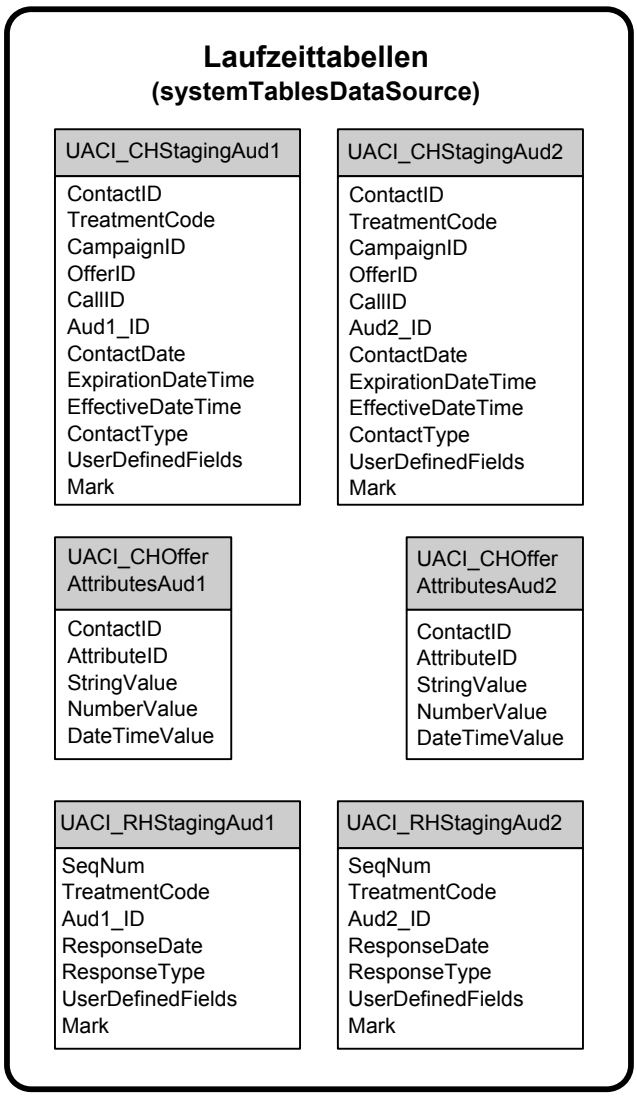
Die SQL-Scripts erstellen die folgenden Standard-Zielgruppenebenen:

- `UACI_CHStaging`
- `UACI_CHOfferAttrib`
- `UACI_RHStaging`

Sie müssen Kopien dieser drei Tabellen für jede Ihrer Zielgruppenebenen in den Laufzeittabellen erstellen.

Wenn Ihre Campaign-Kontakt- und -Antwortverlaufstabellen benutzerdefinierte Felder haben, müssen Sie dieselben Feldnamen und Typen in den Tabellen `UACI_CHStaging` und `UACI_RHStaging` erstellen. Sie können diese Felder während der Laufzeit auffüllen, indem Sie Name/Wert-Paare desselben Namens in den Sitzungsdaten erstellen. Beispiel: Ihre Kontakt- und Antwortverlaufstabellen enthalten das Feld `catalogID`. Sie müssen das Feld `catalogID` beiden Tabellen `UACI_CHStaging` und `UACI_RHStaging` hinzufügen. Später füllt die Interact-API dieses Feld auf, indem ein Ereignisparameter als ein Name/Wert-Paar namens `catalogID` definiert wird. Sitzungsdaten können durch die Profiltabelle, durch zeitbezogene Daten, durch die Lernfunktion oder die Interact-API bereitgestellt werden.

Das folgende Diagramm zeigt Beispieltabellen für die Zielgruppenebenen Aud1 und Aud2. Dieses Diagramm umfasst nicht alle Tabellen in der Laufzeitdatenbank.



Alle Felder in den Tabellen sind erforderlich. Sie können `CustomerID` und `UserDefinedFields` ändern, um Ihren Campaign-Kontakt- und -Antwortverlaufstabellen zu entsprechen.

Testlauftabellen

Die Testlauftabellen werden für Testläufe interaktiver Ablaufdiagramme verwendet. Testläufe interaktiver Ablaufdiagramme sollten Ihre Segmentierungslogik testen. Sie müssen nur eine Testlaufdatenbank für Ihre Interact-Installation konfigurieren. Die Testlauftabellen müssen nicht in einer eigenständigen Datenbank sein. Sie könnten beispielsweise Ihre Kundendatentabellen für Campaign verwenden.

Der den Testlauftabellen zugeordnete Datenbankbenutzer muss CREATE-Berechtigungen haben, um die Testlauf-Ergebnistabellen hinzuzufügen.

Die Testlauf-Datenbank muss alle Tabellen enthalten, die im interaktiven Kanal zugeordnet sind.

Diese Tabellen sollten Daten enthalten, um Szenarien auszuführen, die Sie in Ihren interaktiven Ablaufdiagrammen testen wollen. Beispiel: Wenn Ihre interaktiven Ab-

laufdiagramme Logik haben, um Personen in Segmente basierend auf der getroffenen Auswahl in einem Voicemail-System zu sortieren, sollten Sie zumindest eine Zeile für jede mögliche Auswahl haben. Wenn Sie eine Interaktion erstellen, die mit einem Formular auf Ihrer Website funktioniert, sollten Sie Zeilen aufnehmen, die fehlende oder fehlerhafte Daten darstellen; verwenden Sie beispielsweise `name@domaincom` für den Wert einer E-Mail-Adresse.

Jede Testlauftabelle muss zumindest eine Liste von IDs für die entsprechende Zielgruppenebene und eine Spalte haben, die die Echtzeitdaten darstellt, die Sie zu verwenden gedenken. Da Testläufe keinen Zugriff auf Echtzeitdaten haben, müssen Sie Beispieldaten für jedes Element der erwarteten Echtzeitdaten bereitstellen. Beispiel: Wenn Sie Daten verwenden möchten, die Sie in Echtzeit erfassen können, wie der Name der zuletzt besuchten Webseite (im Attribut `lastPageVisited` gespeichert) oder die Anzahl der Artikel im Warenkorb (im Attribut `shoppingCartItemCount` gespeichert), müssen Sie Spalten mit denselben Namen erstellen und die Spalten mit Beispieldaten auffüllen. Dies ermöglicht es Ihnen, Testläufe auf den Verzweigungen Ihrer Ablaufdiagrammlogik durchzuführen, die auf Verhalten oder Kontext basieren.

Testläufe von interaktiven Ablaufdiagrammen sind nicht für große Datenmengen optimiert. Sie können die Anzahl der für den Testlauf verwendeten Zeilen im Interaktionsprozess begrenzen. Die führt allerdings dazu, dass immer der erste Satz von Zeilen ausgewählt wird. Um sicherzustellen, dass andere Zeilensätze ausgewählt werden, verwenden Sie verschiedene Ansichten der Testlauftabellen.

Um die Durchsatzleistung von interaktiven Ablaufdiagrammen in Laufzeit zu testen, müssen Sie eine Testlaufzeitumgebung erstellen, einschließlich einer Profiltabelle für die Testumgebung.

In der Praxis benötigen Sie möglicherweise drei Testtabellengruppen - eine Testlauftabelle für Testläufe von interaktiven Ablaufdiagrammen, Testprofiltabellen für die Testservergruppe und einen Satz von Produktionsprofiltabellen.

Die Standarddatentypen für dynamisch erstellte Tabellen überschreiben

Die Interact-Laufzeitumgebung erstellt dynamisch Tabellen bei zwei Szenarien: während des Testlaufs eines Ablaufdiagramms und während der Ausführung eines Prozesses "Momentaufnahme", der in eine Tabelle schreibt, die noch nicht existiert. Um diese Tabellen zu erstellen, verwendet Interact fest codierte Datentypen für jeden unterstützten Datenbanktyp.

Sie können die Standarddatentypen überschreiben, indem Sie eine Tabelle von alternativen Datentypen namens `uaci_column_types` in `testRunDataSource` oder `prodUserDataSource` erstellen. Diese zusätzliche Tabelle ermöglicht es Interact, seltene Fälle zu verarbeiten, die nicht durch die fest codierten Datentypen abgedeckt sind.

Wenn die Tabelle `uaci_column_types` definiert ist, verwendet Interact die Metadaten für die Spalten als die Datentypen, die für Tabellengenerierungen verwendet werden. Wenn die Tabelle `uaci_column_types` nicht definiert ist oder wenn es Ausnahmebedingungen gibt, die beim Versuch auftreten, die Tabelle zu lesen, werden die Standarddatentypen verwendet.

Beim Start überprüft das Laufzeitsystem zuerst `testRunDataSource` auf die Tabelle `uaci_column_types`. Wenn die Tabelle `uaci_column_types` nicht in

testRunDataSource vorhanden ist oder wenn prodUserDataSource von einem anderen Datenbanktyp ist, überprüft Interact anschließend prodUserDataSource auf die Tabelle.

Überschreiben der Standarddatentypen

Verwenden Sie dieses Verfahren, um die Standarddatentypen für dynamisch erstellte Tabellen zu überschreiben.

Informationen zu diesem Vorgang

Nach jeder Änderung der Tabelle uaci_column_types müssen Sie den Laufzeitserver erneut starten. Planen Sie Ihre Änderungen so ein, dass die Operationen durch den Neustart des Servers möglichst wenig beeinträchtigt werden.

Vorgehensweise

1. Erstellen Sie eine Tabelle in TestRunDataSource oder ProdUserDataSource mit den folgenden Eigenschaften:

   Tabellenname: uaci_column_types

   Spaltennamen:

   - uaci_float
   - uaci_number
   - uaci_datetime
   - uaci_string

   Definieren Sie jede Spalte unter Verwendung des entsprechenden Datentyps, der von Ihrer Datenbank unterstützt wird.
2. Starten Sie den Laufzeitserver erneut, um es Interact zu ermöglichen, die neue Tabelle zu erkennen.

Standarddatentypen für dynamisch erstellte Tabellen

Für jede unterstützte, vom Interact-Laufzeitsystem verwendete Datenbank gibt es fest codierte Datentypen, die standardmäßig für die Spalten "Gleitkomma", "Zahl", "Datum/Uhrzeit" und "Zeichenfolge" verwendet werden.

Tabelle 1. Standarddatentypen für dynamisch erstellte Tabellen

Datenbank	Standarddatentypen
DB2	• float • bigint • timestamp • varchar
Informix	• float • int8 • DATETIME YEAR TO FRACTION • char2
Oracle	• float • number(19) • timestamp • varchar2

Tabelle 1. Standarddatentypen für dynamisch erstellte Tabellen (Forts.)

Datenbank	Standarddatentypen
SQL Server	• `float` • `bigint` • `datetime` • `nvarchar`

Profildatenbank

Die Inhalte der Profildatenbank hängen gänzlich von den Daten ab, die Sie benötigen, um Ihre interaktiven Ablaufdiagramme und die Interact-API zu konfigurieren. Interact erfordert und empfiehlt, dass jede Datenbank bestimmte Tabellen oder Daten enthält.

Die Profildatenbank muss Folgendes enthalten:

- Alle Tabellen, die im interaktiven Kanal zugeordnet sind.

  Diese Tabellen müssen alle Daten enthalten, die für das Ausführen Ihrer interaktiven Ablaufdiagramme in der Produktion erforderlich sind. Diese Tabellen sollten abgewickelt, optimiert und richtig indiziert sein. Da es Leistungseinbußen beim Zugriff auf Dimensionsdaten gibt, sollten Sie soweit möglich ein denormalisiertes Schema verwenden. Zumindest sollten Sie die Profiltabelle auf der Zielgruppenebene ID-Felder indizieren. Wenn es weitere aus dimensionalen Tabellen abgerufene Felder gibt, sollten diese entsprechend indiziert sein, um die Datenbank-Abrufzeit zu verringern. Die Zielgruppen-IDs für die Profiltabellen müssen mit den Zielgruppen-IDs übereinstimmen, die in Campaign definiert sind.
- Wenn Sie die Konfigurationseigenschaft `enableScoreOverrideLookup` auf "true" festlegen, müssen Sie eine Bewertungsüberschreibungstabelle für zumindest eine Zielgruppenebene aufnehmen. Sie definieren die Namen der Bewertungsüberschreibungstabellen mit der Eigenschaft `scoreOverrideTable`.

  Die Bewertungsüberschreibungstabelle kann einzelne Kunde-zu-Angebot-Paarungen haben. Sie können eine Beispiel-Bewertungsüberschreibungstabelle `UACI_ScoreOverride` erstellen, indem Sie das SQL-Script `aci_usertab` in Ihrer Profildatenbank ausführen. Sie sollten auch diese Tabelle auf der Zielgruppen-ID-Spalte indizieren.

  Wenn Sie die Konfigurationseigenschaft `enableScoreOverrideLookup` auf "false" festlegen, müssen Sie keine Bewertungsüberschreibungstabelle aufnehmen.
- Wenn Sie die Konfigurationseigenschaft `enableDefaultOfferLookup` auf "true" festlegen, müssen Sie die globale Angebotstabelle (`UACI_DefaultOffers`) aufnehmen. Sie können eine globale Angebotstabelle erstellen, indem Sie das SQL-Script `aci_usertab` in Ihrer Profildatenbank ausführen.

  Die globale Angebotstabelle kann Zielgruppe-zu-Angebot-Paarungen enthalten.
- Wenn Sie die Konfigurationseigenschaft `enableOfferSuppressionLookup` auf "true" festlegen, müssen Sie eine Tabelle für Angebotsunterdrückung für zumindest eine Zielgruppenebene aufnehmen. Sie definieren die Namen der Tabelle für Angebotsunterdrückung mit der Eigenschaft `offerSuppressionTable`.

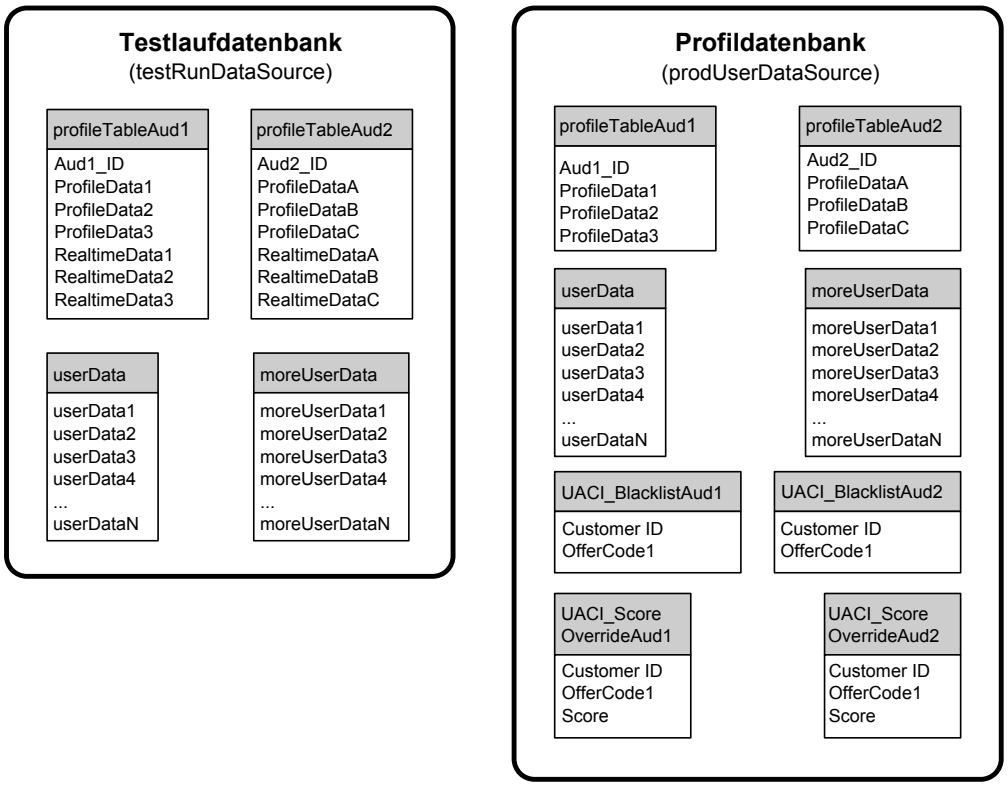
  Die Tabelle für Angebotsunterdrückung kann eine Zeile für jedes für ein Zielgruppenmitglied unterdrücktes Angebot enthalten, auch wenn ein Eintrag nicht für alle Zielgruppenmitglieder erforderlich ist. Sie können eine Beispiel-Tabelle für Angebotsunterdrückung `UACI_BlackList` erstellen, indem Sie das SQL-Script `aci_usertab` in Ihrer Profildatenbank ausführen.

Wenn Sie die Konfigurationseigenschaft `enableOfferSuppressionLookup` auf "false" festlegen, müssen Sie keine Tabelle für Angebotsunterdrückung aufnehmen.

Ein großes Datenvolumen in einer dieser Tabellen kann möglicherweise die Leistung beeinträchtigen. Um beste Ergebnisse zu erzielen, erstellen Sie entsprechende Indizes auf die Zielgruppenebenenspalten bei Tabellen, die zur Laufzeit verwendet werden und große Datenvolumen haben.

Alle oben genannten Konfigurationseigenschaften sind in der Kategorie **Interact > Profil** oder **Interact > Profil > Benutzergruppenebenen > *Benutzergruppenebene***. Das SQL-Script `aci_usertab` befindet sich im `ddl`-Verzeichnis in Ihrem Laufzeitumgebungsinstallationsverzeichnis.

Das folgende Diagramm zeigt Beispieltabellen für den Testlauf und Profildatenbanken für die Zielgruppenebenen Aud1 und Aud2.



Lerntabellen

Falls Sie das integrierte Lernen von Interact verwenden, müssen Sie die Lerntabellen konfigurieren. Diese Tabellen enthalten alle Daten, die die integrierte Lernfunktion benötigt.

Wenn Sie dynamische Lernattribute verwenden, müssen Sie die Tabelle `UACI_AttributeList` auffüllen.

Die Lernfunktion umfasst das Schreiben in temporäre Staging-Tabellen und das Aggregieren von Informationen aus Staging-Tabellen in Lerntabellen. Die Konfigurationseigenschaften `insertRawStatsIntervalInMinutes` und

aggregateStatsIntervalInMinutes in der Kategorie Interact > offerserving > Built-in Learning Config legt fest, wie oft die Lerntabellen aufgefüllt werden.

Das Attribut insertRawStatsIntervalInMinutes legt fest, wie oft Annahme- und Kontaktinformationen für jeden Kunden und jedes Angebot aus dem Speicher in die Staging-Tabellen UACI_OfferStatsTX und UACI_OfferTxAll verschoben werden. Die in den Staging-Tabellen gespeicherten Informationen werden aggregiert und in die Tabellen UACI_OfferStats und UACI_OfferStatsAll in regelmäßigen Intervallen verschoben, die durch die Konfigurationseigenschaft aggregateStatsIntervalInMinutes festgelegt werden.

Das integrierte Lernen von Interact verwendet diese Daten, um endgültige Bewertungen für Angebote zu berechnen.

Kontaktverlauf für sitzungsübergreifende Antwortverfolgung

Wenn Sie die sitzungsübergreifende Antwortfunktion aktivieren, benötigt die Laufzeitumgebung Lesezugriff auf die Campaign-Kontaktverlaufstabellen. Sie können die Laufzeitumgebung konfigurieren, die Campaign-Systemtabellen anzuzeigen, oder Sie können die Campaign-Kontaktverlaufstabellen kopieren. Wenn Sie eine Kopie der Tabellen erstellen, müssen Sie den Prozess steuern, um die Kopie auf dem letzten Stand zu halten. Das Kontakt- und Antwortverlaufsmodul aktualisiert nicht die Kopie der Kontaktverlaufstabellen.

Sie müssen das SQL-Script aci_crhtab auf diesen Kontaktverlaufstabellen ausführen, um Tabellen hinzuzufügen, die für die sitzungsübergreifende Antwortverfolgungsfunktion erforderlich sind.

Datenbankscripts zur Aktivierung von Interact-Funktionen ausführen

Um die in Interact verfügbaren optionalen Funktionen nutzen zu können, führen Sie Datenbankscripts für die Datenbank aus, um Tabellen zu erstellen oder vorhandene Tabellen zu aktualisieren.

Ihre Interact-Installation mit der Designzeit- als auch der Laufzeitumgebung enthält **ddl**-Funktionsscripts. Die **ddl**-Scripts fügen erforderliche Spalten zu Ihren Tabellen hinzu.

Um diese optionalen Funktionen zu aktivieren, führen Sie das entsprechende Script für die angegebene Datenbank oder Tabelle aus.

dbType ist der Datenbanktyp, beispielsweise sqlsvr für Microsoft SQL Server, ora für Oracle oder db2 für IBM DB2.

Verwenden Sie die folgende Tabelle, um Datenbankscripts für die Datenbank auszuführen, um Tabellen zu erstellen oder vorhandene Tabellen zu aktualisieren:

Tabelle 2. Datenbankscripts

Funktionsname	Funktionsscript	Ausführen für	Änderung
Globale Angebote, Angebotsunterdrückung und Bewertungsüberschreibung	**aci_usrtab_***dbType***.sql** in *Interact_Home*\ddl\acifeatures\ (Installationsverzeichnis der Laufzeitumgebung)	Ihre Profildatenbank (userProdDataSource)	Erstellt die Tabellen UACI_DefaultOffers, UACI_BlackList und UACI_ScoreOverride.

Tabelle 2. Datenbankscripts (Forts.)

Funktionsname	Funktionsscript	Ausführen für	Änderung
Bewertung	**aci_scoringfeature**_*dbType*.**sql** in *Interact_Home*\ddl\acifeatures\ (Installationsverzeichnis der Laufzeitumgebung))	Bewertungsüberschreibungstabellen in Ihrer Profildatenbank (userProdDataSource)	Fügt die Spalten LikelihoodScore und AdjExploreScore hinzu.
Lernfunktion	**aci_lrnfeature**_*dbType*.**sql** in *Interact_Home*\interactDT\ddl\acifeatures\ (Installationsverzeichnis der Laufzeitumgebung)	Campaign-Datenbank mit den Kontaktverlaufstabellen	Fügt die Spalten RTSelectionMethod, RTLearningMode und RTLearningModelID zur Tabelle UA_DtlContactHist hinzu. Fügt außerdem die Spalten RTLearningMode und RTLearningModelID zur Tabelle UA_ResponseHistory hinzu. Dieses Script ist außerdem für die Berichtsfunktionen des optional erhältlichen Interact-Berichtspakets erforderlich.

Informationen zur Verfolgung von Kontakt- und Antwortverlauf

Sie können die Laufzeitumgebung so konfigurieren, dass der Kontakt- und Antwortverlauf in den Campaign-Kontakt- und Antwortverlaufstabellen erfasst wird. Die Laufzeitserver speichern den Kontakt- und Antwortverlauf in Staging-Tabellen. Das Kontakt- und Antwortverlaufsmodul kopiert diese Daten aus den Staging-Tabellen in die Campaign-Kontakt- und Antwortverlaufstabellen.

Das Kontakt- und Antwortverlaufsmodul funktioniert nur, wenn Sie die Eigenschaften `interactInstalled` und `contactAndResponseHistTracking > isEnabled` auf der Seite "Konfiguration" für die Designumgebung auf `yes` festlegen.

Wenn Sie das sitzungsübergreifende Antwortüberwachungsmodul verwenden, ist das Kontakt- und Antwortverlaufsmodul eine separate Entität.

Kontakt- und Antworttypen

Sie können einen Kontakttyp und zwei Antworttypen mit Interact erfassen. Sie können auch zusätzliche benutzerdefinierte Antworttypen mit der Methode `postEvent` aufzeichnen.

Tabelle mit contactAndResponseHistTracking-Eigenschaften

In dieser Tabelle werden die Eigenschaften aufgelistet, die in der Kategorie `contactAndResponseHistTracking` enthalten sind:

Ereignis	Kontakt-/Antworttyp	Konfigurationseigenschaft
Angebotskontakt protokollieren	Kontakt	`contacted`
Angebotsannahme protokollieren	Antwort	`accept`
Angebotsablehnung protokollieren	Antwort	`reject`

Tabelle mit UA_UsrResponseType-Eigenschaften

Stellen Sie sicher, dass die Spalte CountsAsResponse der Tabelle UA_UsrResponseType in den Campaign-Systemtabellen ordnungsgemäß konfiguriert ist. Alle diese Antworttypen müssen in der Tabelle UA_UsrResponseType vorhanden sein.

Um ein gültiger Eintrag in der Tabelle UA_UsrResponseType zu sein, müssen Sie einen Wert für alle Spalten in der Tabelle definieren, einschließlich CountsAsResponse. Die folgenden Werte sind für CountsAsResponse gültig:

- 0 - keine Antwort
- 1 - eine Antwort
- 2 - eine Ablehnung
-

Diese Antworten werden für die Berichterstellung verwendet.

Zusätzliche Antworttypen

In Interact können Sie die Methode postEvent in der Interact-API verwenden, um ein Ereignis auszulösen, das eine Aktion "Accept" oder "Reject" für ein Angebot protokolliert. Sie können das System auch erweitern, damit der Aufruf postEvent zusätzliche Antworttypen erfasst, wie z. B. Explore, Consider, Commit oder Fulfill.

Alle diese Antworttypen müssen in der Tabelle UA_UsrResponseType in den Campaign-Systemtabellen vorhanden sein. Wenn Sie bestimmte Ereignisparameter mit der Methode postEvent verwenden, können Sie zusätzliche Antworttypen erfassen und festlegen, ob ein Annehmen in die Lernfunktion aufgenommen werden soll.

Um zusätzliche Antworttypen zu protokollieren, müssen Sie folgende Ereignisparameter hinzufügen:

- **UACIResponseTypeCode** - eine Zeichenfolge, die einen Antworttypcode darstellt. Der Wert muss ein gültiger Eintrag in der Tabelle UA_UsrResponseType sein.

  Um ein gültiger Eintrag in UA_UsrResponseType zu sein, müssen Sie alle Spalten in der Tabelle definieren, einschließlich CountsAsResponse. Gültige Werte für CountsAsResponse sind 0, 1 oder 2. 0 gibt keine Antwort an, 1 gibt eine Antwort an und 2 gibt eine Ablehnung an. Diese Antworten werden für die Berichterstellung verwendet.
- **UACILogToLearning** - eine Zahl mit dem Wert "1" oder "0". "1" gibt an, dass Interact das Ereignis als Annahme für die Lernfunktion protokollieren oder die Angebotsunterdrückung in einer Sitzung aktivieren soll. "0" gibt an, dass Interact das Ereignis für die Lernfunktion nicht protokollieren bzw. die Angebotsunterdrückung nicht in einer Sitzung aktivieren soll. Dieser Parameter ermöglicht es Ihnen, ohne Auswirkung auf die Lernfunktion mehrere postEvent-Methoden zum Protokollieren von verschiedenen Antworttypen zu erstellen. Wenn Sie UACILogToLearning nicht definieren, nimmt Interact den Standardwert von "0" an.

Wenn während der Übergabe eines Annahmeereignisses ein Wert für responseTypeCode angegeben wird, wird das Angebot beim Annehmen nicht unterdrückt. Wenn logToLearningAsAccept auf 0 gesetzt ist, sollte das Angebot unabhängig von dem Wert für ResponseTypeCode (beispielsweise 0, 1, 2) niemals unterdrückt werden. Wenn Sie das Angebot unterdrücken möchten, sollte in Ihrer Instanz von postEvent der Parameter UACIResponseTypeCode nicht angegeben wer-

den. Wenn der Parameter UACIResponseTypeCode angegeben wird, sollte der Wert von UACILogToLearning "1" lauten, falls das Angebot unterdrückt werden soll.

Sie können Folgendes erstellen: mehrere Ereignisse mit der Aktion Angebotsannahme protokollieren, ein Ereignis für jeden Antworttyp, den Sie protokollieren möchten, oder ein einzelnes Ereignis mit der Aktion Angebotsannahme protokollieren, das Sie für jeden postEvent-Aufruf verwenden, mit dem Sie separate Antworttypen protokollieren.

Sie erstellen beispielsweise ein Ereignis mit der Aktion "Angebotsannahme protokollieren" für jeden Antworttyp. Sie definieren die folgenden benutzerdefinierten Antworten in der Tabelle UA_UsrResponseType [als Name (Code)]: Explore (EXP), Consider (CON) und Commit (CMT). Dann erstellen Sie drei Ereignisse und nennen Sie LogAccept_Explore, LogAccept_Consider und LogAccept_Commit. Alle drei Ereignisse sind identisch (weisen die Aktion Angebotsannahme protokollieren auf), haben jedoch unterschiedliche Namen, sodass die Person, die mit der API arbeitet, sie unterscheiden kann.

Sie können auch ein einzelnes Ereignis mit der Aktion "Angebotsannahme protokollieren" erstellen, das Sie für alle benutzerdefinierten Antworttypen verwenden. Nennen Sie es beispielsweise LogCustomResponse.

Beim Arbeiten mit der API besteht kein funktionaler Unterschied zwischen den Ereignissen. Die Namenskonventionen können den Code jedoch verständlicher machen. Wenn Sie jede benutzerdefinierte Antwort anders benennen, zeigt der Bericht Aktivitätsübersicht Kanalereignisse auch genauere Informationen an.

Zuerst definieren Sie alle Name/Wert-Paare.

```
//Define name value pairs for the UACIResponseTypeCode
// Response type Explore
NameValuePair responseTypeEXP = new NameValuePairImpl();
responseTypeEXP.setName("UACIResponseTypeCode");
responseTypeEXP.setValueAsString("EXP");
responseTypeEXP.setValueDataType(NameValuePair.DATA_TYPE_STRING);

// Response type Consider
NameValuePair responseTypeCON = new NameValuePairImpl();
responseTypeCON.setName("UACIResponseTypeCode");
responseTypeCON.setValueAsString("CON");
responseTypeCON.setValueDataType(NameValuePair.DATA_TYPE_STRING);

// Response type Commit
NameValuePair responseTypeCMT = new NameValuePairImpl();
responseTypeCMT.setName("UACIResponseTypeCode");
responseTypeCMT.setValueAsString("CMT");
responseTypeCMT.setValueDataType(NameValuePair.DATA_TYPE_STRING);

//Define name value pairs for UACILOGTOLEARNING
//Does not log to learning
NameValuePair noLogToLearning = new NameValuePairImpl();
noLogToLearning.setName("UACILOGTOLEARNING");
noLogToLearning.setValueAsString("0");
noLogToLearning.setValueDataType(NameValuePair.DATA_TYPE_NUMERIC);

//Logs to learning
NameValuePair LogToLearning = new NameValuePairImpl();
LogToLearning.setName("UACILogToLearning");
LogToLearning.setValueAsString("1");
LogToLearning.setValueDataType(NameValuePair.DATA_TYPE_NUMERIC);
```

Dieses erste Beispiel zeigt die Verwendung von einzelnen Ereignissen.

```
//EXAMPLE 1: This set of postEvent calls use the individually named events
//PostEvent with an Explore response
NameValuePair[] postEventParameters = { responseTypeEXP, noLogToLearning };
response = api.postEvent(sessionId, LogAccept_Explore, postEventParameters);

//PostEvent with a Consider response
NameValuePair[] postEventParameters = { responseTypeCON, noLogToLearning };
response = api.postEvent(sessionId, LogAccept_Consider, postEventParameters);

//PostEvent with a Commit response
NameValuePair[] postEventParameters = { responseTypeCOM, LogToLearning };
response = api.postEvent(sessionId, LogAccept_Commit, postEventParameters);
```

Dieses zweite Beispiel zeigt die Verwendung nur eines Ereignisses.

```
//EXAMPLE 2: This set of postEvent calls use the single event
//PostEvent with an Explore response
NameValuePair[] postEventParameters = { responseTypeEXP, noLogToLearning };
response = api.postEvent(sessionId, LogCustomResponse, postEventParameters);

//PostEvent with a Consider response
NameValuePair[] postEventParameters = { responseTypeCON, noLogToLearning };
response = api.postEvent(sessionId, LogCustomResponse, postEventParameters);

//PostEvent with a Commit response
NameValuePair[] postEventParameters = { responseTypeCOM, LogToLearning };
response = api.postEvent(sessionId, LogCustomResponse, postEventParameters);
```

Beide Beispiele führen genau dieselben Aktionen durch, aber eine Version ist möglicherweise lesbarer als die andere.

Zuordnung der Staging-Tabellen der Laufzeitumgebung zu den Campaign-Verlaufstabellen

Die Interact-Staging-Tabellen für den Kontaktverlauf werden den Campaign-Verlaufstabellen zugeordnet. Sie müssen für jede Zielgruppenebene über eine der Staging-Tabellen der Laufzeitumgebung verfügen.

Zuordnung der Staging-Tabelle "UACI_CHStaging" für den Kontaktverlauf

Diese Tabelle zeigt die Zuordnung der Staging-Tabelle `UACI_CHStaging` der Laufzeitumgebung zu der Campaign-Kontaktverlaufstabelle. Die angezeigten Tabellennamen sind die Beispieltabellen, die für die Standardzielgruppe in den Laufzeittabellen und in den Campaign-Systemtabellen erstellt werden.

Tabelle 3. Kontaktverlauf

UACI_CHStaging **Interact Spaltenname der Staging-Tabelle im Kontaktprotokoll**	**Campaign Kontaktprotokolltabelle**	**Name der Tabellenspalte**
ContactID	k. A.	k. A.
TreatmentCode	UA_Treatment	TreatmentCode
CampaignID	UA_Treatment	CampaignID
OfferID	UA_Treatment	OfferID
CellID	UA_Treatment	CellID
CustomerID	UA_DtlContactHist	CustomerID
ContactDate	UA_DtlContactHist	KontaktDatumUhrzeit

Tabelle 3. Kontaktverlauf (Forts.)

UACI_CHStaging **Interact Spaltenname der Staging-Tabelle im Kontaktprotokoll**	**Campaign Kontaktprotokolltabelle**	**Name der Tabellenspalte**
ExpirationDateTime	UA_Treatment	ExpirationDateTime
EffectiveDateTime	UA_Treatment	EffectiveDateTime
ContactType	UA_DtlContactHist	ContactStatusID
UserDefinedFields	UA_DtlContactHist	UserDefinedFields

`ContactID` ist ein Schlüssel, mit dem die Tabelle `UACI_CHOfferAtrib` mit der Tabelle `UACI_CHStaging` verknüpft wird. Die Spalte `userDefinedFields` kann beliebige Daten Ihrer Wahl enthalten.

Zuordnung der Staging-Tabelle "UACI_CHOfferAttrib" für den Kontaktverlauf

Diese Tabelle zeigt die Zuordnung der Staging-Tabelle `UACI_CHOfferAttrib` der Laufzeitumgebung zu der Campaign-Kontaktverlaufstabelle. Die angezeigten Tabellennamen sind die Beispieltabellen, die für die Standardzielgruppe in den Laufzeittabellen und in den Campaign-Systemtabellen erstellt werden.

Tabelle 4. Angebotsattribute

UACI_CHOfferAttrib **Interact Spaltenname der Staging-Tabelle im Kontaktprotokoll**	**Campaign Kontaktprotokolltabelle**	**Name der Tabellenspalte**
ContactID	k. A.	k. A.
AttributeID	UA_OfferHistAttrib	AttributeID
StringValue	UA_OfferHistAttrib	StringValue
NumberValue	UA_OfferHistAttrib	NumberValue
DateTimeValue	UA_OfferHistAttrib	DateTimeValue

Zuordnung der Staging-Tabelle "UACI_RHStaging" für den Antwortverlauf

Diese Tabelle zeigt die Zuordnung der Staging-Tabelle `UACI_RHStaging` der Laufzeitumgebung zu der Campaign-Antwortverlaufstabelle. Die angezeigten Tabellennamen sind die Beispieltabellen, die für die Standardzielgruppe in den Laufzeittabellen und in den Campaign-Systemtabellen erstellt werden.

Tabelle 5. Antwortverlauf

UACI_RHStaging **Interact Spaltenname der Staging-Tabelle im Antwortverlauf**	**Campaign Antwortverlaufstabelle**	**Name der Tabellenspalte**
SeqNum	k. A.	k. A.
TreatmentCode	UA_ResponseHistory	TreatmentInstID
CustomerID	UA_ResponseHistory	CustomerID
ResponseDate	UA_ResponseHistory	ResponseDateTime

Tabelle 5. Antwortverlauf (Forts.)

UACI_RHStaging **Interact Spaltenname der Staging-Tabelle im Antwortverlauf**	**Campaign Antwortverlaufstabelle**	**Name der Tabellenspalte**
ResponseType	UA_ResponseHistory	ResponseTypeID
UserDefinedFields	UA_ResponseHistory	UserDefinedFields

`SeqNum` ist ein Schlüssel, der im Modul für den Kontakt- und Antwortverlauf zum Identifizieren von Daten verwendet, aber nicht in den Campaign-Antworttabellen aufgezeichnet wird. Die Spalte `userDefinedFields` kann beliebige Daten Ihrer Wahl enthalten.

Zusätzliche Spalten in Staging-Tabellen

Wenn Sie den Staging-Tabellen weitere Spalten hinzufügen, werden diese vom Modul für den Kontakt- und Antwortverlauf in den Tabellen `UA_DtlContactHist` oder `UA_ResponseHistory` in die Spalten mit dem gleichen Namen geschrieben.

Beispiel: Wenn Sie der Tabelle `UACI_CHStaging` die Spalte `linkFrom` hinzufügen, kopiert das Modul für den Kontakt- und Antwortverlauf diese Daten in die Spalte `linkFrom` in der Tabelle `UA_DtlContactHist`.

Zusätzliche Spalten in Kontakt- und Antwortverlaufstabellen von Campaign

Wenn Sie über zusätzliche Spalten in Ihren Kontakt- und Antwortverlaufstabellen von Campaign verfügen, müssen Sie den Staging-Tabellen die entsprechenden Spalten hinzufügen, bevor Sie das Modul für den Kontakt- und Antwortverlauf ausführen.

Um die zusätzlichen Spalten in den Staging-Tabellen auszufüllen, erstellen Sie Spalten mit den gleichen Namen wie Ihre Name/Wert-Paare in den Daten der Laufzeitsitzung.

So können Sie beispielsweise die Name/Wert-Paare `NumberItemsInWishList` und `NumberItemsInShoppingCart` erstellen und diese Ihrer `UACI_RHStaging`-Tabelle hinzufügen. Wenn ein Ereignis zum Protokollieren der Annahme oder der Ablehnung eines Angebots eintritt, füllt die Laufzeitumgebung diese Felder aus. Die Laufzeitumgebung füllt die Tabelle `UACI_CHStaging` aus, wenn ein Ereignis zum Protokollieren eines Angebotskontakts auftritt.

Einbeziehen einer Bewertung für ein Angebot mithilfe von Tabellen

Sie können die benutzerdefinierten Felder verwenden, um die Bewertung einzuschließen, die zur Präsentation eines Angebots verwendet wird. Fügen Sie sowohl der Tabelle `UACI_CHStaging` in den Laufzeittabellen als auch der Tabelle `UA_DtlContactHist` in den Campaign-Systemtabellen jeweils die Spalte `FinalScore` hinzu. Wenn Sie das integrierte Lernmodul verwenden, füllt Interact die Spalte `FinalScore` automatisch mit der endgültigen Bewertung aus, die für das Angebot verwendet wurde.

Wenn Sie ein benutzerdefiniertes Lernmodul aufbauen, können Sie die setActual-ValueUsed-Methode der ITreatment-Benutzeroberfläche und die logEvent-Methode der ILearning-Benutzeroberfläche verwenden.

Wenn Sie kein Lernmodul verwenden, fügen Sie sowohl der Tabelle UACI_CHStaging in den Laufzeittabellen als auch der Tabelle UA_DtlContactHist in den Campaign-Systemtabellen jeweils die Spalte Score hinzu. Interact füllt die Spalte Score automatisch mit der endgültigen Bewertung aus, die für das Angebot verwendet wurde.

Erstellen von neuen Verlaufstabellen in Campaign und Staging-Tabellen in Interact

Wenn Sie statt der Zielgruppenebene Kunde eine andere Zielgruppenebene verwenden, müssen Sie neue Verlaufstabellen in Campaign und neue Staging-Tabellen in Interact erstellen.

Zum Beispiel wird das folgende Beispielscript in der IBM DB2-Designzeitdatenbank verwendet, um Verlaufstabellen in Campaign für eine Zielgruppenebene des Typs Konto zu erstellen.

```
DROP TABLE ACCT_UA_ResponseHistory;
DROP TABLE ACCT_UA_DtlContactHist;
DROP TABLE ACCT_UA_ContactHistory;
CREATE TABLE ACCT_UA_ResponseHistory (
        AccountID           varchar(30) NOT NULL,
        TreatmentInstID      bigint NOT NULL,
        ResponsePackID       bigint NOT NULL,
        ResponseDateTime     timestamp NOT NULL,
        WithinDateRangeFlg   int,
        OrigContactedFlg     int,
        BestAttrib           int,
        FractionalAttrib     float,
        DirectResponse       int,
        CustomAttrib         float,
        ResponseTypeID       bigint,
        DateID               bigint,
        TimeID               bigint,
        UserDefinedFields    char(18),
        CONSTRAINT ACCT_cRespHistory_PK
               PRIMARY KEY (AccountID, TreatmentInstID,
               ResponsePackID )
 );
 CREATE TABLE ACCT_UA_ContactHistory (
        AccountID           varchar(30) NOT NULL,
        CellID               bigint NOT NULL,
        PackageID            bigint NOT NULL,
        ContactDateTime      timestamp,
        UpdateDateTime       timestamp,
        ContactStatusID      bigint,
        DateID               bigint,
        TimeID               bigint,
        UserDefinedFields    char(18),
        CONSTRAINT ACCT_cContactHist_PK
               PRIMARY KEY (AccountID, CellID, PackageID )
 );
 CREATE INDEX ACCT_cContactHist_IX1 ON ACCT_UA_ContactHistory
 (
        CellID );
 CREATE INDEX ACCT_cContactHist_IX2 ON ACCT_UA_ContactHistory
 (
        PackageID                    ,
        CellID );
 CREATE TABLE ACCT_UA_DtlContactHist (
```

```
       AccountID           varchar(30) NOT NULL,
       TreatmentInstID      bigint NOT NULL,
       ContactStatusID      bigint,
       ContactDateTime      timestamp,
       UpdateDateTime       timestamp,
       UserDefinedFields    char(18),
       DateID               bigint NOT NULL,
       TimeID               bigint NOT NULL
);
CREATE INDEX ACCT_cDtlContHist_IX1 ON ACCT_UA_DtlContactHist
(
       AccountID                    ,
       TreatmentInstID );
ALTER TABLE ACCT_UA_ResponseHistory
       ADD CONSTRAINT ACCT_cRespHistory_FK2
              FOREIGN KEY (TimeID)
                           REFERENCES UA_Time  (TimeID);
ALTER TABLE ACCT_UA_ResponseHistory
       ADD CONSTRAINT ACCT_cRespHistory_FK4
              FOREIGN KEY (DateID)
                           REFERENCES UA_Calendar  (DateID);
ALTER TABLE ACCT_UA_ResponseHistory
       ADD CONSTRAINT ACCT_cRespHistory_FK3
              FOREIGN KEY (ResponseTypeID)
                           REFERENCES UA_UsrResponseType  (
              ResponseTypeID);
ALTER TABLE ACCT_UA_ResponseHistory
       ADD CONSTRAINT ACCT_cRespHistory_FK1
              FOREIGN KEY (TreatmentInstID)
                           REFERENCES UA_Treatment  (
              TreatmentInstID);
ALTER TABLE ACCT_UA_ContactHistory
       ADD CONSTRAINT ACCT_cContactHist_FK2
              FOREIGN KEY (DateID)
                           REFERENCES UA_Calendar  (DateID);
ALTER TABLE ACCT_UA_ContactHistory
       ADD CONSTRAINT ACCT_cContactHist_FK3
              FOREIGN KEY (TimeID)
                           REFERENCES UA_Time  (TimeID);
ALTER TABLE ACCT_UA_ContactHistory
       ADD CONSTRAINT ACCT_cContactHist_FK1
              FOREIGN KEY (ContactStatusID)
                           REFERENCES UA_ContactStatus  (
              ContactStatusID);
ALTER TABLE ACCT_UA_DtlContactHist
       ADD CONSTRAINT ACCT_cDtlContactH_FK3
              FOREIGN KEY (TimeID)
                           REFERENCES UA_Time  (TimeID);
ALTER TABLE ACCT_UA_DtlContactHist
       ADD CONSTRAINT ACCT_cDtlContactH_FK2
              FOREIGN KEY (DateID)
                           REFERENCES UA_Calendar  (DateID);
ALTER TABLE ACCT_UA_DtlContactHist
       ADD CONSTRAINT ACCT_cDtlContactH_FK1
              FOREIGN KEY (ContactStatusID)
                           REFERENCES UA_ContactStatus  (
              ContactStatusID);
alter table ACCT_UA_DtlContactHist add RTSelectionMethod int;
alter table ACCT_UA_ResponseHistory add RTSelectionMethod int;
```

Das folgende Beispielscript wird in der IBM DB2-Laufzeitdatenbank verwendet, um Staging-Tabellen für den Verlauf in Interact für eine Zielgruppenebene des Typs Konto zu erstellen.

```
DROP TABLE ACCT_UACI_RHStaging;
DROP TABLE ACCT_UACI_CHOfferAttrib;
DROP TABLE ACCT_UACI_CHStaging;
```

```
DROP TABLE ACCT_UACI_UserEventActivities;
DROP TABLE ACCT_UACI_EventPatternState;
CREATE TABLE ACCT_UACI_RHStaging (
       SeqNum               bigint NOT NULL,
       TreatmentCode        varchar(512),
       AccountID            varchar(30),
       ResponseDate         timestamp,
       ResponseType         int,
       ResponseTypeCode     varchar(64),
       Mark                 bigint NOT NULL
                                          DEFAULT 0,
       UserDefinedFields    char(18),
  RTSelectionMethod    int,
       CONSTRAINT iRHStaging_PK1
              PRIMARY KEY (SeqNum)
);
CREATE TABLE ACCT_UACI_CHOfferAttrib (
        ContactID            bigint NOT NULL,
        AttributeID          bigint NOT NULL,
        StringValue          varchar(512),
        NumberValue          float,
        DateTimeValue        timestamp,
        CONSTRAINT ACCT_iCHOfferAttrib_PK
               PRIMARY KEY (ContactID, AttributeID)
 );
 CREATE TABLE ACCT_UACI_CHStaging (
        ContactID            bigint NOT NULL,
        TreatmentCode        varchar(512),
        CampaignID           bigint,
        OfferID              bigint,
        CellID               bigint,
        AccountID            varchar(30),
        ContactDate          timestamp,
        ExpirationDateTime   timestamp,
        EffectiveDateTime    timestamp,
        ContactType          int,
        UserDefinedFields    char(18),
        Mark                 bigint NOT NULL DEFAULT 0,
  RTSelectionMethod    bigint,
        CONSTRAINT ACCT_iCHStaging_PK
               PRIMARY KEY (ContactID)
 );
CREATE TABLE ACCT_UACI_UserEventActivity
(
       SeqNum               bigint NOT NULL GENERATED ALWAYS AS IDENTITY,
       ICID                 bigint NOT NULL,
       ICName               varchar(64) NOT NULL,
       CategoryID           bigint NOT NULL,
       CategoryName         varchar(64) NOT NULL,
       EventID              bigint NOT NULL,
       EventName            varchar(64) NOT NULL,
       TimeID               bigint,
       DateID               bigint,
       Occurrences          bigint NOT NULL,
       AccountID varchar(30) not null,
       CONSTRAINT iUserEventActivity_PK
              PRIMARY KEY (SeqNum)
);
create table ACCT_UACI_EventPatternState
(
 UpdateTime bigint not null,
 State varchar(1000) for bit data,
 AccountID varchar(30) not null,
        CONSTRAINT iCustomerPatternState_PK
        PRIMARY KEY (AccountID,UpdateTime)
);
 ALTER TABLE ACCT_UACI_CHOfferAttrib
```

```
ADD CONSTRAINT ACCT_iCHOfferAttrib_FK1
      FOREIGN KEY (ContactID)
                  REFERENCES ACCT_UACI_CHStaging  (ContactID);
```

Konfigurieren der JMX-Überwachung für das Kontakt- und Antwortverlaufsmodul

Verwenden Sie dieses Verfahren, um die JMX-Überwachung für das Kontakt- und Antwortverlaufsmodul zu konfigurieren. Die JMXMP- und RMI-Protokolle werden unterstützt. Die Konfiguration der JMX-Überwachung bedeutet nicht, dass die Sicherheit für das Kontakt- und Antwortverlaufsmodul aktiviert wird. Die Konfiguration der JMX-Überwachung erfolgt über Marketing Platform für die Designumgebung.

Informationen zu diesem Vorgang

Zur Verwendung Ihres JMX-Überwachungstools für das Kontakt- und Antwortverlaufsmodul wird folgende Standardadresse verwendet:

- JMXMP-Protokoll: `service:jmx:jmxmp://`*`CampaignServer`*`:`*`Port`*`/campaign`.
- RMI-Protokoll: `service:jmx:rmi:///jndi/rmi://`*`CampaignServer`*`:`*`Port`*`/campaign`.

Wenn Sie die Daten in Ihrem JMX-Überwachungstool anzeigen, werden die Ergebnisattribute zuerst nach Partition und dann nach Zielgruppenebene organisiert.

Vorgehensweise

Bearbeiten Sie in Marketing Platform für die Designumgebung folgende Konfigurationseigenschaften in der Kategorie `Campaign > Überwachung`.

Konfigurationseigenschaft	**Einstellung**
`monitorEnabledForInteract`	**True**
port	Die Portnummer für den JMX-Service
protocol	Das zu verwendende Protokoll: • **JMXMP** • **RMI** Es ist keine Sicherheit für das Kontakt- und Antwortverlaufsmodul aktiviert, selbst wenn Sie das JMXMP-Protokoll auswählen.

Informationen zur sitzungsübergreifenden Antwortverfolgung

Die Besucher schließen möglicherweise nicht bei jedem einzelnen Touchpoint-Besuch immer eine Transaktion ab. Ein Kunde kann zum Beispiel einen Artikel im Warenkorb hinzufügen, aber den Einkauf erst zwei Tage später abschließen. Dennoch ist es nicht sinnvoll, die Laufzeitsitzung für unbestimmte Zeit aktiv zu lassen. Sie können die sitzungsübergreifende Antwortverfolgung aktivieren, um die Präsentation eines Angebots in einer Sitzung zu verfolgen und mit einer Antwort in einer anderen Sitzung abzugleichen.

Die sitzungsübergreifende Antwortverfolgung in Interact kann standardmäßig nach Verfahrenscodes oder Angebotscodes abgleichen. Sie können die sitzungsübergreifende Antwortverfolgung jedoch auch so konfigurieren, dass ein beliebiger Alternativcode abgeglichen wird. Dabei werden alle verfügbaren Daten und Antworten

sitzungsübergreifend abgeglichen. Beispiel: Ihre Website enthält ein Angebot mit einem Werbecode, der eine Woche lang gültig ist und zu dem Zeitpunkt generiert wird, an dem der Rabattartikel angezeigt wird. Ein Benutzer kann den Artikel im Warenkorb hinzufügen, aber den Kauf erst drei Tage später abschließen. Wenn Sie den `postEvent`-Aufruf verwenden, um ein Annahmeereignis zu protokollieren, können Sie nur den Aktionscode einbeziehen. Da die Laufzeit in der aktuellen Sitzung keinen übereinstimmenden Verfahrens- oder Angebotscode finden kann, wird das Annahmeereignis mit allen verfügbaren Informationen in der Staging-Tabelle für die sitzungsübergreifende Antwort (XSessResponse) gespeichert. Der CrossSessionResponse-Service liest die XSessResponse-Tabelle regelmäßig aus und versucht, die Datensätze mit den verfügbaren Kontaktverlaufsdaten abzugleichen. Der CrossSessionResponse-Service gleicht den Aktionscode mit dem Kontaktverlauf ab und erfasst alle erforderlichen Daten, die zum Protokollieren einer ordnungsgemäßen Antwort benötigt werden. Anschließend schreibt der CrossSessionResponse-Service die Antwort in die Staging-Tabellen und (sofern aktiviert) in die Lerntabellen. Das Modul für den Kontakt- und Antwortverlauf schreibt die Antwort dann in die Campaign Kontakt- und Antwortverlaufstabellen. Die erfolgreiche Verarbeitung der sitzungsübergreifenden Antwort hängt von den ursprünglichen Datensätzen zum Kontaktverlauf ab, die durch den Kontaktverlauf-ETL-Prozess zur Campaign-Datenbank migriert wurden.

Konfiguration der Quellendaten für die sitzungsübergreifende Antwortverfolgung

Interact Mit der sitzungsübergreifenden Antwortverfolgung können Sie Sitzungsdaten aus der Laufzeitumgebung mit dem Campaign Kontakt- und Antwortverlauf abgleichen. Standardmäßig verwendet die sitzungsübergreifende Antwortverfolgung den Verfahrenscode oder den Angebotscode zum Abgleich. Sie können die Laufzeitumgebung so konfigurieren, dass ein anderer, benutzerdefinierter Code zum Abgleich verwendet wird.

- Wenn Sie einen alternativen Code abgleichen möchten, müssen Sie den alternativen Code in der `UACI_TrackingType`-Tabelle in den Interact-Laufzeittabellen definieren.
- Die Laufzeitumgebung benötigt Zugriff auf die Campaign Kontaktverlaufstabellen. Um dies sicherzustellen, können Sie entweder die Laufzeitumgebung für den Zugriff auf die Campaign Kontaktverlaufstabellen konfigurieren oder eine Kopie der Kontaktverlaufstabellen in der Laufzeitumgebung erstellen.

  Dieser Zugriff erfolgt schreibgeschützt und unabhängig vom Dienstprogramm für den Kontakt- und Antwortverlauf.

  Wenn Sie eine Kopie der Tabellen erstellen, liegt es in Ihrer Verantwortung, sicherzustellen, dass die Daten in der Kopie des Kontaktverlaufs korrekt sind. Mit der `purgeOrphanResponseThresholdInMinutes`-Eigenschaft können Sie die Dauer konfigurieren, wie lange der CrossSessionResponse-Service Antworten beibehalten soll, die noch nicht abgeglichen wurden, und wie oft die Daten in der Kopie der Kontaktverlaufstabellen aktualisiert werden. Wenn Sie das Modul für den Kontakt- und Antwortverlauf verwenden, sollten Sie die Aktualisierungen des ETL-Prozesses koordinieren, um sicherzustellen, dass die Daten stets auf dem neuesten Stand sind.

Konfigurieren von Kontakt- und Antwortverlaufstabellen für die sitzungsübergreifende Antwortverfolgung

Unabhängig davon, ob Sie eine Kopie der Kontaktverlaufstabellen erstellen oder die tatsächlichen Tabellen in den Campaign-Systemtabellen verwenden, müssen Sie die folgenden Schritte durchführen, um die Kontakt- und Antwortverlaufstabellen zu konfigurieren.

Vorbereitende Schritte

Die Kontakt- und Antwortverlaufstabellen müssen vor Ausführen dieser Schritte in Campaign korrekt zugeordnet werden.

Vorgehensweise

1. Führen Sie das SQL-Script `aci_lrnfeature` im Verzeichnis `interactDT/ddl/acifeatures` im Installationsverzeichnis der Interact-Designumgebung für die Tabellen `UA_DtlContactHist` und `UA_ResponseHistory` in den Campaign-Systemtabellen aus.

   Dadurch wird den Tabellen `UA_DtlContactHist` und `UA_ResponseHistory` die Spalte `RTSelectionMethod` hinzugefügt. Führen Sie das Script `aci_lrnfeature` für diese Tabellen für jede einzelne Zielgruppenebene aus. Bearbeiten Sie das Script, falls erforderlich, um stets mit der korrekten Tabelle für jede Zielgruppenebene zu arbeiten.
2. Wenn Sie die Kontaktverlaufstabellen in die Laufzeitumgebung kopieren möchten, tun Sie das jetzt.

   Wenn Sie eine Kopie der Campaign-Kontaktverlaufstabellen erstellen möchten, die die Laufzeitumgebung aufrufen kann, um die sitzungsübergreifende Antwortverfolgung zu unterstützen, beachten Sie die folgenden Richtlinien:
   - Die sitzungsübergreifende Antwortverfolgung benötigt Lesezugriff auf diese Tabellen.
   - Die sitzungsübergreifende Antwortverfolgung benötigt die folgenden Tabellen aus dem Campaign-Kontaktprotokoll.
     - `UA_DtlContactHist` (für jede Zielgruppenebene)
     - `UA_Treatment`

     Sie müssen die Daten in diesen Tabellen regelmäßig aktualisieren, um die korrekte Antwortverfolgung sicherzustellen.
3. Führen Sie das SQL-Script `aci_crhtab` im Verzeichnis `ddl` im Installationsverzeichnis der Interact-Laufzeitumgebung für die Quellendaten des Kontakt- und Antwortverlaufs aus.

   Dieses Script erstellt die Tabellen `UACI_XsessResponse` und `UACI_CRHTAB_Ver`.
4. Erstellen Sie für jede Zielgruppenebene eine Version der `UACI_XsessResponse`-Tabelle.

Ergebnisse

Um die Performance der sitzungsübergreifenden Antwortverfolgung zu verbessern, können Sie die Menge der Kontaktverlaufsdaten einschränken, indem Sie entweder die Kontaktverlaufsdaten kopieren oder eine entsprechende Ansicht in den Campaign-Kontaktverlaufstabellen konfigurieren. Beispiel: Wenn in Ihren geschäftsrelevanten Prozessen und Verfahren geregelt ist, dass ein Angebot maximal 30 Tage lang gültig sein kann, sollten Sie die Kontaktverlaufsdaten auf die letzten 30 Tage beschränken. Wenn Sie die Anzahl der Tage der zu verwaltenden Kontaktverlaufsdaten ändern wollen, öffnen Sie die Konfigurationseigenschaft **Campaign | partiti-**

ons | partition*n* | Interact | contactAndResponseHistTracking und setzen Sie den Wert auf **daysBackInHistoryToLookupContact**.

Es werden keine Ergebnisse aus der sitzungsübergreifenden Antwortverfolgung angezeigt, bevor das Modul für den Kontakt- und Antwortverlauf aufgerufen wird. Beispiel: Der Standardwert für `processSleepIntervalInMinutes` beträgt 60 Minuten. Es kann daher mindestens eine Stunde dauern, bis sitzungsübergreifende Antworten im Campaign Antwortverlauf angezeigt werden.

UACI_TrackingType-Tabelle

Die `UACI_TrackingType`-Tabelle ist Teil der Laufzeitumgebungstabellen. Diese Tabelle definiert die Verfolgungscodes, die mit der sitzungsübergreifenden Antwortverfolgung verwendet werden. Der Verfolgungscode definiert, welche Methode die Laufzeitumgebung verwendet, um das aktuelle Angebot in einer Laufzeitsitzung mit dem Kontakt- und Antwortverlauf abzugleichen.

Spalte	Typ	Beschreibung
`TrackingCodeType`	int	Eine Zahl, die den Verfolgungscodetyp darstellt. Auf diese Zahl wird von den SQL-Befehlen verwiesen, die verwendet werden, um die Informationen aus den Sitzungsdaten mit den Kontakt- und Antwortverlaufstabellen abzugleichen.
`Name`	varchar(64)	Der Name für den Verfolgungscodetyp. Dieser wird an die Sitzungsdaten übergeben, indem der reservierte Parameter `UACI_TrackingCodeType` mit der Methode `postEvent` verwendet wird.
`Beschreibung`	varchar(512)	Eine Kurzbeschreibung des Verfolgungscodetyps. Dieses Feld ist optional.

Standardmäßig sind für die Laufzeitumgebung zwei Verfolgungscodetypen definiert, wie in der folgenden Tabelle dargestellt. Für jeden anderen Code müssen Sie einen eindeutigen `TrackingCodeType` definieren.

TrackingCodeType	Name	Beschreibung
1	Verfahrenscode	UACI generierter Verfahrenscode
2	Angebotscode	UAC Kampagnenangebotscode

UACI_XSessResponse

Die `UACI_XSessResponse`-Tabelle ist Teil der Laufzeitumgebungstabellen. Diese Tabelle wird für die sitzungsübergreifende Antwortverfolgung verwendet.

Für jede Zielgruppenebene muss eine Instanz dieser Tabelle in der Datenquelle mit dem Kontakt- und Antwortverlauf vorhanden sein, der für die Interact sitzungsübergreifende Antwortverfolgung verfügbar ist.

Spalte	Typ	Beschreibung
`SeqNumber`	bigint	Kennung für die Datenzeile. Der CrossSessionResponse-Service verarbeitet alle Datensätze in der `SeqNumber`-Reihenfolge.
`ICID`	bigint	ID des interaktiven Kanals

Spalte	Typ	Beschreibung
AudienceID	bigint	Die Zielgruppen-ID für diese Zielgruppenebene. Der Name dieser Spalte muss mit der in Campaign definierten Zielgruppen-ID übereinstimmen. Die Beispieltabelle enthält die Spalte CustomerID.
TrackingCode	varchar(64)	Der Wert, der vom UACIOfferTrackingCode-Parameter der postEvent-Methode übergeben wird.
TrackingCodeType	int	Die numerische Darstellung des Verfahrenscodes. Der Wert muss ein gültiger Eintrag in der Tabelle UACI_TrackingType sein.
OfferID	bigint	Die in Campaign definierte Angebots-ID.
ResponseType	int	Der Antworttyp für diesen Datensatz. Der Wert muss ein gültiger Eintrag in der Tabelle UA_UsrResponseType sein.
ResponseTypeCode	varchar(64)	Der Antworttypcode für diesen Datensatz. Der Wert muss ein gültiger Eintrag in der Tabelle UA_UsrResponseType sein.
ResponseDate	Datum/ Uhrzeit	Das Datum der Antwort.
Mark	bigint	Der Wert dieses Feldes identifiziert den Status des Datensatzes. • 1 - In Bearbeitung • 2 - Erfolgreich • NULL - Neuversuch • -1 - Datensatz befindet sich seit mehr als purgeOrphanResponseThresholdInMinutes Minuten in der Datenbank. Im Rahmen der Wartung dieser Tabelle durch den Datenbankadministrator können Sie mit diesem Feld nach Datensätzen suchen, die nicht abgeglichen werden, das sind alle Datensätze mit dem Wert -1. Alle Datensätze mit dem Wert 2 werden automatisch vom CrossSessionResponse-Service entfernt.
UsrDefinedFields	char(18)	Alle benutzerdefinierten Felder, die Sie beim Abgleichen von Angebotsantworten im Kontakt- und Antwortverlauf einschließen möchten. Wenn Sie zum Beispiel einen Aktionscode abgleichen möchten, müssen Sie ein benutzerdefiniertes Feld für den Aktionscode einschließen.

Aktivieren der sitzungsübergreifenden Antwortverfolgung

Mit diesem Verfahren können Sie die sitzungsübergreifende Antwortverfolgung aktivieren.

Vorbereitende Schritte

Um die sitzungsübergreifende Antwortverfolgung optimal nutzen zu können, müssen Sie das Modul für den Kontakt- und Antwortverlauf konfigurieren.

Um die sitzungsübergreifende Antwortverfolgung zu verwenden, müssen Sie die Laufzeitumgebung für den Lesezugriff auf die Kontakt- und Antwortverlaufstabellen in Campaign konfigurieren. Sie können entweder die tatsächlichen Campaign

Kontakt- und Antwortverlaufstabellen in der Designumgebung oder eine Kopie der Tabellen in den Datenquellen der Laufzeitumgebung lesen. Die Konfiguration der Laufzeitumgebung für den Lesezugriff auf die Kontakt- und Antwortverlaufstabelle erfolgt unabhängig von der Konfiguration eines eventuellen Kontakt- und Antwortverlaufsmoduls.

Wenn Sie etwas anderes als den Verfahrens- oder Angebotscode zum Abgleichen verwenden möchten, müssen Sie der Tabelle UACI_TrackingType den entsprechenden Bezug hinzufügen.

Vorgehensweise

1. Erstellen Sie die XSessResponse-Tabellen in den Kontakt- und Antwortverlaufstabellen, die die Laufzeitumgebung aufrufen kann.
2. Definieren Sie die Eigenschaften in der Kategorie contactAndResponseHistoryDataSource für die Laufzeitumgebung.
3. Definieren Sie die Eigenschaft crossSessionResponseTable für jede Zielgruppenebene.
4. Erstellen Sie für jede Zielgruppenebene eine Kategorie OverridePerAudience.

Sitzungsübergreifender Abgleich von Angebot und Antwort

Standardmäßig verwendet die sitzungsübergreifende Antwortverfolgung die Verfahrenscodes oder die Angebotscodes zum Abgleich. Der crossSessionResponse-Service verwendet SQL-Befehle, um Verfahrenscodes, Angebotscodes oder einen benutzerdefinierten Code aus den Sitzungsdaten mit den Campaign Kontakt- und Antwortverlaufstabellen abzugleichen. Sie können diese SQL-Befehle bearbeiten, um alle Anpassungen abzugleichen, die Sie an den Verfolgungscodes, Angebotscodes oder angepassten Codes vorgenommen haben.

Abgleich nach Verfahrenscode

Die SQL zum Abgleich nach Verfahrenscode muss alle Spalten in der XSessResponse-Tabelle für diese Zielgruppenebene plus eine Spalte mit dem Namen OfferIDMatch zurückgeben. Der Wert in der Spalte OfferIDMatch muss mit der offerId-Kennung identisch sein, die dem Verfahrenscode im Datensatz XSessResponse entspricht.

Das folgende Beispiel zeigt den standardmäßig generierten SQL-Befehl zum Abgleichen von Verfahrenscodes. Interact generiert die SQL zum Verwenden der richtigen Tabellennamen für die Zielgruppenebene. Diese SQL wird verwendet, wenn die Eigenschaft Interact > Services > crossSessionResponse > OverridePerAudience > *AudienceLevel* > TrackingCodes > byTreatmentCode > SQL auf **Systemgenerierte SQL verwenden** gesetzt ist.

```
select   distinct treatment.offerId as OFFERIDMATCH,
   tx.*,
            dch.RTSelectionMethod
from     UACI_XSessResponse tx
Left Outer Join UA_Treatment treatment ON tx.trackingCode=treatment.treatmentCode
Left Outer Join UA_DtlContactHist dch ON tx.CustomerID = dch.CustomerID
Left Outer Join UA_ContactHistory ch ON tx.CustomerID = ch.CustomerID
AND treatment.cellID = ch.cellID
AND treatment.packageID=ch.packageID
where   tx.mark=1
and       tx.trackingCodeType=1
```

Die Werte UACI_XsessResponse, UA_DtlContactHist, CustomerID und UA_ContactHistory werden von den Einstellungen in Interact definiert. Beispiel:

UACI_XsessResponse wird von der Konfigurationseigenschaft Interact > Profil > Zielgruppenebenen > [AudienceLevelName] > crossSessionResponseTable definiert.

Wenn Sie die Kontakt- und Antwortverlaufstabellen angepasst haben, müssen Sie diese SQL möglicherweise überarbeiten, um mit den Tabellen arbeiten zu können. SQL-Überschreibungen werden in der Eigenschaft Interact > Services > crossSessionResponse > OverridePerAudience > *(AudienceLevel)* > TrackingCodes > byTreatmentCode > OverrideSQL definiert. Wenn Sie die SQL überschreiben, müssen Sie auch die Eigenschaft SQL in **SQL überschreiben** ändern.

Abgleich nach Angebotscode

Die SQL zum Abgleich nach Angebotscode muss alle Spalten in der XSessResponse-Tabelle für diese Zielgruppenebene plus eine Spalte mit dem Namen TreatmentCodeMatch zurückgeben. Der Wert in der Spalte TreatmentCodeMatch ist der Verfahrenscode, der der Angebots-ID (und dem Angebotscode) im Datensatz XSessResponse entspricht.

Das folgende Beispiel zeigt den standardmäßig generierten SQL-Befehl zum Abgleichen von Angebotscodes. Interact generiert die SQL zum Verwenden der richtigen Tabellennamen für die Zielgruppenebene. Diese SQL wird verwendet, wenn die Eigenschaft Interact > Services > crossSessionResponse > OverridePerAudience > *AudienceLevel* > TrackingCodes > byOfferCode > SQL auf **Systemgenerierte SQL verwenden** gesetzt ist.

```
select   treatment.treatmentCode as TREATMENTCODEMATCH,
   tx.*,
dch.RTSelectionMethod
from     UACI_XSessResponse tx
Left Outer Join UA_DtlContactHist dch ON tx.CustomerID=dch.CustomerID
Left Outer Join UA_Treatment treatment ON tx.offerId = treatment.offerId
Left Outer Join
   (
   select   max(dch.contactDateTime) as maxDate,
      treatment.offerId,
      dch.CustomerID
   from     UA_DtlContactHist dch, UA_Treatment treatment, UACI_XSessResponse tx
   where  tx.CustomerID=dch.CustomerID
   and       tx.offerID = treatment.offerId
   and       dch.treatmentInstId = treatment.treatmentInstId
   group by dch.CustomerID,  treatment.offerId
   ) dch_by_max_date ON tx.CustomerID=dch_by_max_date.CustomerID
    and tx.offerId = dch_by_max_date.offerId
where   tx.mark = 1
and       dch.contactDateTime = dch_by_max_date.maxDate
and       dch.treatmentInstId = treatment.treatmentInstId
and       tx.trackingCodeType=2
union
select   treatment.treatmentCode as TREATMENTCODEMATCH,
   tx.*,
   0
from UACI_XSessResponse tx
Left Outer Join UA_ContactHistory ch ON tx.CustomerID =ch.CustomerID
Left Outer Join UA_Treatment treatment ON tx.offerId = treatment.offerId
Left Outer Join
   (
   select   max(ch.contactDateTime) as maxDate,
      treatment.offerId, ch.CustomerID
   from UA_ContactHistory ch, UA_Treatment treatment, UACI_XSessResponse tx
   where  tx.CustomerID =ch.CustomerID
   and       tx.offerID = treatment.offerId
   and treatment.cellID = ch.cellID
```

```
   and treatment.packageID=ch.packageID
   group by ch.CustomerID,  treatment.offerId
   ) ch_by_max_date ON tx.CustomerID =ch_by_max_date.CustomerID
   and tx.offerId = ch_by_max_date.offerId
   and treatment.cellID = ch.cellID
   and treatment.packageID=ch.packageID
where   tx.mark = 1
   and       ch.contactDateTime = ch_by_max_date.maxDate
   and treatment.cellID = ch.cellID
   and       treatment.packageID=ch.packageID
   and       tx.offerID = treatment.offerId
   and       tx.trackingCodeType=2
```

Die Werte UACI_XsessResponse, UA_DtlContactHist, CustomerID und UA_ContactHistory werden von den Einstellungen in Interact definiert. Beispiel: UACI_XsessResponse wird von der Konfigurationseigenschaft Interact > Profil > Zielgruppenebenen > [AudienceLevelName] > crossSessionResponseTable definiert.

Wenn Sie die Kontakt- und Antwortverlaufstabellen angepasst haben, müssen Sie diese SQL möglicherweise überarbeiten, um mit den Tabellen arbeiten zu können. SQL-Überschreibungen werden in der Eigenschaft Interact > Services > crossSessionResponse > OverridePerAudience > *(AudienceLevel)* > TrackingCodes > byOfferCode > OverrideSQL definiert. Wenn Sie die SQL überschreiben, müssen Sie auch die Eigenschaft SQL in **SQL überschreiben** ändern.

Abgleich nach alternativem Code

Sie können einen SQL-Befehl definieren, um einen Abgleich nach einem beliebigen Alternativcode durchzuführen. So können Sie zum Beispiel unabhängig von den Angebots- und Verfahrenscode auch zusätzliche Werbe- und Produktcodes definieren.

Dieser Alternativcode muss in der Tabelle UACI_TrackingType in den Interact Tabellen der Laufzeitumgebung definiert werden.

Sie müssen eine SQL oder eine gespeicherte Prozedur in der Eigenschaft Interact > Services > crossSessionResponse > OverridePerAudience > *(AudienceLevel)* > TrackingCodes > byAlternateCode > OverrideSQL angeben, die alle Spalten in der Tabelle XSessResponse für diese Zielgruppenebene plus die Spalten TreatmentCodeMatch und OfferIDMatch zurückgibt. Optional können Sie auch offerCode anstelle von OfferIDMatch zurückgeben (in der Form offerCode1, offerCode2, ... offerCode*N* für Angebotscodes mit N Teilen). Die Werte in den Spalten TreatmentCodeMatch und OfferIDMatch (oder in Spalten mit dem Angebotscode) müssen dem TrackingCode im Datensatz XSessResponse entsprechen.

Beispiel: Der folgende SQL-Pseudocode verwendet zum Abgleich die Spalte AlternateCode in der Tabelle XSessResponse.

```
Select m.TreatmentCode as TreatmentCodeMatch, m.OfferID as OfferIDMatch, tx.*
From MyLookup m, UACI_XSessResponse tx
Where m.customerId = tx.customerId
And m.alternateCode = tx.trackingCode
And tx.mark=1
And tx.trackingCodeType = <x>
```

Dabei ist <*x*> der in der Tabelle UACI_TrackingType definierte Verfolgungscode.

Verwenden eines Datenbankladedienstprogramms mit der Laufzeitumgebung

Standardmäßig schreibt die Laufzeitumgebung die Protokolldaten für Kontakte und Antworten aus den Sitzungsdaten in Staging-Tabellen. Auf einem sehr aktiven Produktionssystem kann die benötigte Speichermenge, die zum Zwischenspeichern aller Daten erforderlich ist, bevor diese in die Staging-Tabellen geschrieben werden können, jedoch ein Problem darstellen. Sie können die Laufzeit konfigurieren und ein Datenbankladedienstprogramm verwenden, um die Leistung zu verbessern.

Wenn Sie ein Datenbankladedienstprogramm aktivieren, fixiert die Laufzeit den gesamten Kontakt- und Antwortverlauf nicht im Speicher, sondern schreibt die Daten stattdessen in eine Staging-Datei, bevor in die Staging-Tabellen geschrieben wird. Verwenden Sie die `externalLoaderStagingDirectory`-Eigenschaft, um die Position für das Verzeichnis zu definieren, das die Staging-Dateien enthält. Dieses Verzeichnis enthält mehrere Unterverzeichnisse. Das erste Unterverzeichnis ist das Verzeichnis der Laufzeitinstanz mit den Verzeichnissen `contactHist` und `respHist`. Die Verzeichnisse `contactHist` und `respHist` enthalten eindeutig benannte Unterverzeichnisse im Format *audienceLevelName.uniqueID.currentState*, in denen die Staging-Dateien enthalten sind.

Aktueller Status	Beschreibung
CACHE	Verzeichnisinhalte werden gerade in eine Datei geschrieben.
READY	Verzeichnisinhalte sind zur Verarbeitung bereit.
RUN	Verzeichnisinhalte werden gerade in die Datenbank geschrieben.
PROCESSED	Verzeichnisinhalte wurden in die Datenbank geschrieben.
ERROR	Fehler beim Schreiben der Verzeichnisinhalte in der Datenbank.
ATTN	Verzeichnisinhalte bedürfen Ihrer Aufmerksamkeit. Vermutlich sind manuelle Maßnahmen erforderlich, um den Schreibvorgang in der Datenbank abzuschließen.
RERUN	Verzeichnisinhalte sind zum Schreiben in der Datenbank bereit. Sie sollten die Verzeichnisse `ATTN` oder `ERROR` in `RERUN` umbenennen, nachdem Sie das Problem behoben haben.

Sie können das Verzeichnis der Laufzeitinstanz definieren, indem Sie die JVM-Eigenschaft interact.runtime.instance.name im Startscript des Anwendungsservers definieren. Sie können dem Startscript Ihres Webanwendungsservers zum Beispiel `-Dinteract.runtime.instance.name=instance2` hinzufügen. Sofern nicht anders angegeben, wird der Standardname `DefaultInteractRuntimeInstance` verwendet.

Das Verzeichnis `samples` enthält Beispieldateien, die Ihnen beim Schreiben Ihrer eigenen Steuerdateien für das Datenbankladedienstprogramm behilflich sind.

Aktivieren eines Datenbankladedienstprogramms mit der Laufzeitumgebung

Verwenden Sie dieses Verfahren, um ein Datenbankladedienstprogramm mit der Laufzeitumgebung zu aktivieren.

Vorbereitende Schritte

Sie müssen Befehls- oder Steuerdateien für Ihr Datenbankladedienstprogramm definieren, bevor Sie die Laufzeitumgebung für ihre Verwendung konfigurieren. Diese Dateien müssen an derselben Position auf allen Laufzeitservern in derselben Servergruppe vorhanden sein.

Interact stellt Beispiel-Befehls- und -Steuerdateien im Verzeichnis `loaderService` in Ihrer Interact-Laufzeitserverinstallation bereit.

Vorgehensweise

1. Vergewissern Sie sich, dass der Laufzeitumgebungsbenutzer Anmeldeberechtigungsnachweise für die Laufzeittabellendatenquelle, die in `Interact > general > systemTablesDataSource` in Ihren Konfigurationseigenschaften definiert ist, besitzt.
2. Definieren Sie die Konfigurationseigenschaften `Interact > general > systemTablesDataSource > loaderProperties`.
3. Definieren Sie die Eigenschaft `Interact > services >externalLoaderStaging-Directory`.
4. Überarbeiten Sie ggf. die Konfigurationseigenschaften `Interact > services > responseHist > fileCache`.
5. Überarbeiten Sie ggf. die Konfigurationseigenschaften `Interact > services > contactHist > fileCache`.
6. Starten Sie den Laufzeitserver erneut.

ETL-Prozess für Ereignismuster

Wenn Sie umfangreiche IBM Interact-Ereignismusterdaten verarbeiten müssen und diese Daten für Abfragen und zu Berichtszwecken zur Verfügung stellen möchten, können Sie zur Erzielung einer optimalen Leistung auf jedem unterstützten Server einen eigenständigen ETL-Prozess (Extrahieren, Transformieren, Laden) installieren.

In Interact werden alle Ereignismusterdaten für eine bestimmte Zielgruppen-ID (AudienceID) als einzelne Datensammlung in den Tabellen der Laufzeitdatenbank gespeichert. Die AudienceID und die Informationen zum Musterzustand werden als großes Binärobjekt (BLOB) gespeichert. Damit auf Basis der Ereignismuster SQL-Abfragen durchgeführt oder Berichte erstellt werden können, ist dieser neue ETL-Prozess erforderlich, um das Objekt in Tabellen in einer Zieldatenbank zu unterteilen. Hierfür entnimmt der eigenständige ETL-Prozess die Ereignismusterdaten aus den Tabellen der Interact-Laufzeitdatenbank, verarbeitet diese in Übereinstimmung mit dem von Ihnen angegebenen Zeitplan und speichert sie in der Zieldatenbank, wo sie für SQL-Abfragen oder für eine zusätzliche Berichterstellung zur Verfügung stehen.

Neben der Verschiebung von Ereignismusterdaten in die Zieldatenbank und deren dortiger Transformation synchronisiert der eigenständige ETL-Prozess die Daten in der Zieldatenbank mit den aktuellen Informationen in Ihrer Interact-Laufzeitdatenbank. Wenn Sie beispielsweise ein Ereignismuster in der Interact-Laufzeit löschen, werden die verarbeiteten Daten dieses Ereignismusters bei der nächsten Ausführung des ETL-Prozesses aus der Zieldatenbank entfernt. Die Informationen zum Ereignismusterzustand werden ebenfalls auf dem neuesten Stand gehalten. Daher handelt es sich bei den in der Zieldatenbank gespeicherten Informationen zu Ereignismustern nicht um archivierte Daten, sondern ausschließlich um aktuelle Daten.

Ausführen des eigenständigen ETL-Prozesses

Wenn Sie den eigenständigen ETL-Prozess auf einem Server starten, wird er kontinuierlich im Hintergrund ausgeführt, bis er gestoppt wird. Der Prozess befolgt während seines Betriebs die Anweisungen in den Marketing Platform-Konfigurationseigenschaften, um die Häufigkeit, Datenbankverbindungen und sonstige Details zu bestimmen.

Vorbereitende Schritte

Bevor Sie den eigenständigen ETL-Prozess ausführen, müssen die folgenden Bedingungen erfüllt sein:

- Sie müssen die Berechtigungen der Benutzerrolle als Interact-Administrator besitzen.
- Sie müssen den Prozess auf einem Server installiert und die Dateien sowohl auf dem Server als auch in Marketing Platform für Ihre Konfiguration ordnungsgemäß konfiguriert haben.

Anmerkung:

Wenn Sie den ETL-Prozess unter Microsoft Windows in einer anderen Sprache als in US-amerikanischem Englisch ausführen möchten, legen Sie mithilfe von `chcp` in der Eingabeaufforderung die Codepage für die von Ihnen verwendete Sprache fest. Sie können beispielsweise einen der folgenden Codes verwenden: ja_jp=932, zh_cn=936, ko_kr=949, ru_ru=1251. Für de_de, fr_fr, it_it, es_es und pt_br verwenden Sie 1252. Verwenden Sie vor dem Start des ETL-Prozesses den Befehl `chcp` in der Windows-Befehlseingabeaufforderung, um sicherzustellen, dass die Zeichen ordnungsgemäß angezeigt werden.

Informationen zu diesem Vorgang

Sobald Sie den eigenständigen ETL-Prozess installiert und konfiguriert haben, können Sie den Prozess starten.

Vorgehensweise

1. Öffnen Sie eine Eingabeaufforderung auf dem Server, auf dem der ETL-Prozess installiert ist.
2. Navigieren Sie zu dem Verzeichnis `<Interact_Ausgangsverzeichnis>/PatternStateETL/bin`, das die ausführbaren Dateien für den ETL-Prozess enthält.
3. Führen Sie die Datei `command.bat` (unter Microsoft Windows) bzw. `command.sh` (auf UNIX-ähnlichen Betriebssystemen) mit den folgenden Parametern aus:
   - `-u <Benutzername>`. Für diesen Wert müssen Sie einen gültigen Marketing Platform-Benutzer angeben. Außerdem muss dieser Benutzer mit Zugriff auf die vom ETL-Prozess verwendeten Datenquellen **TargetDS** und **RuntimeDS** konfiguriert worden sein.
   - `-p <Kennwort>`. Ersetzen Sie *<Kennwort>* durch das Kennwort des von Ihnen angegebenen Benutzers. Wenn das Kennwort für diesen Benutzer leer ist, geben Sie zwei Anführungszeichen an (Beispiel: `-p ""`). Das Kennwort ist bei der Ausführung der Befehlsdatei optional; wenn Sie das Kennwort nicht im Befehl angeben, werden Sie bei der Ausführung des Befehls zu dessen Eingabe aufgefordert.

- `-c <Profilname>`. Ersetzen Sie *<Profilname>* durch genau den Namen, den Sie in Marketing Platform über die von Ihnen erstellte Konfiguration **Interact | PatternStateETL** angegeben haben.

  Der von Ihnen hier eingegebene Name muss mit dem Wert übereinstimmen, den Sie bei der Erstellung der Konfiguration im Feld **Neuer Kategoriename** angegeben haben.
- `start`. Der Befehl "start" ist zum Starten des Prozesses erforderlich.

Der vollständige Befehl für den Start des Prozesses hat daher also folgendes Format:

```
command.bat -u <Benutzername> -p <Kennwort> -c <Profilname> start
```

Ergebnisse

Der eigenständige ETL-Prozess wird ausgeführt. Er wird im Hintergrund ausgeführt, bis Sie den Prozess stoppen oder der Server erneut gestartet wird.

Anmerkung:

Wenn Sie den Prozess das erste Mal ausführen, kann die Ausführung der kumulierten Ereignismusterdaten eine beträchtliche Zeit in Anspruch nehmen. Die nachfolgenden Ausführungen des Prozesses verwenden nur die neueste Gruppe von Ereignismusterdaten und dauern daher nicht so lange.

Hinweis: Sie können wie im folgenden Beispiel auch das Argument `help` für die Datei `command.bat` bzw. `command.sh` angeben, um alle verfügbaren Optionen anzuzeigen:

```
command.bat help
```

Stoppen des eigenständigen ETL-Prozesses

Wenn Sie den eigenständigen ETL-Prozess auf einem Server starten, wird er kontinuierlich im Hintergrund ausgeführt, bis er gestoppt wird.

Informationen zu diesem Vorgang

Vorgehensweise

1. Öffnen Sie eine Eingabeaufforderung auf dem Server, auf dem der ETL-Prozess installiert ist.
2. Navigieren Sie zu dem Verzeichnis `<Interact_Ausgangsverzeichnis>/PatternStateETL/bin`, das die ausführbaren Dateien für den ETL-Prozess enthält.
3. Führen Sie die Datei `command.bat` (unter Microsoft Windows) bzw. `command.sh` (auf UNIX-ähnlichen Betriebssystemen) mit den folgenden Parametern aus:
   - `-u <Benutzername>`. Für diesen Wert müssen Sie einen gültigen Marketing Platform-Benutzer angeben. Außerdem muss dieser Benutzer mit Zugriff auf die vom ETL-Prozess verwendeten Datenquellen **TargetDS** und **RuntimeDS** konfiguriert worden sein.
   - `-p <Kennwort>`. Ersetzen Sie *<Kennwort>* durch das Kennwort des von Ihnen angegebenen Benutzers. Wenn das Kennwort für diesen Benutzer leer ist, ge-

ben Sie zwei Anführungszeichen an (Beispiel: `-p ""`). Das Kennwort ist bei der Ausführung der Befehlsdatei optional; wenn Sie das Kennwort nicht im Befehl angeben, werden Sie bei der Ausführung des Befehls zu dessen Eingabe aufgefordert.

- `-c <Profilname>`. Ersetzen Sie *<Profilname>* durch genau den Namen, den Sie in Marketing Platform über die von Ihnen erstellte Konfiguration **Interact | PatternStateETL** angegeben haben.

  Der von Ihnen hier eingegebene Name muss mit dem Wert übereinstimmen, den Sie bei der Erstellung der Konfiguration im Feld **Neuer Kategoriename** angegeben haben.

- `stop`. Der Befehl "stop" ist zum Stoppen des Prozesses erforderlich. Wenn Sie diesen Befehl verwenden, werden vor der Beendigung des Prozesses alle aktuellen Operationen des ETL-Prozesses abgeschlossen.

  Wenn Sie den ETL-Prozess beenden möchten, ohne auf den Abschluss der aktuellen Operationen zu warten, verwenden Sie den Befehl `forcestop` anstelle von `stop`.

Der vollständige Befehl für den Start des Prozesses hat daher also folgendes Format:

```
command.bat -u <username> -p <password> -c <profileName> stop
```

Ergebnisse

Der eigenständige ETL-Prozess wird gestoppt.

Kapitel 4. Services für Angebote

Sie können Interact auf viele Arten konfigurieren, um die Möglichkeiten zu erweitern, wie Angebote zum Präsentieren ausgewählt werden. In den folgenden Abschnitten werden diese optionalen Funktionen im Detail beschrieben.

Angebotsberechtigung

Der Zweck von Interact ist es, auswählbare Angebote zu empfehlen. Einfach gesagt: Interact zeigt die besten Angebote unter den auswählbaren Angeboten an, basierend auf dem Besucher, dem Kanal und der Situation.

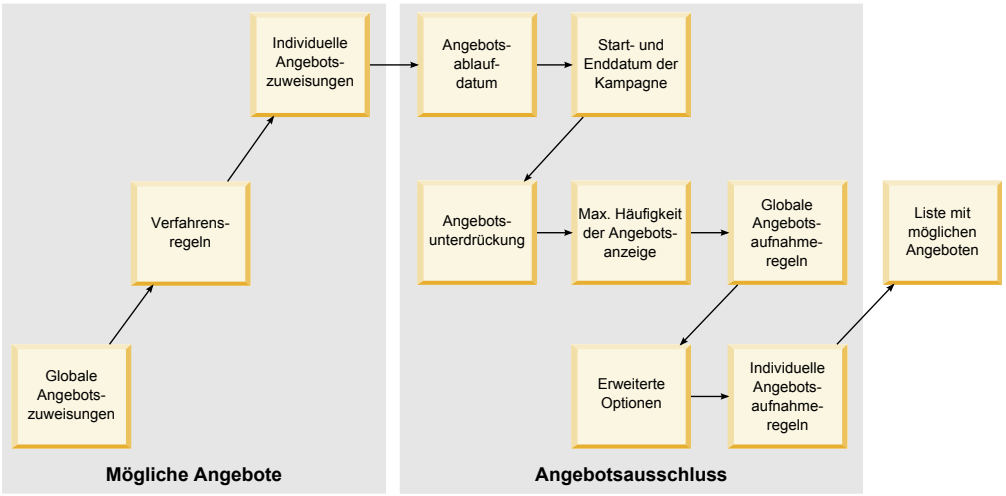
Verfahrensregeln sind nur der erste Schritt, wie Interact ermittelt, welche Angebote für einen Kunden infrage kommen. Interact bietet mehrere optionale Funktionen, die Sie implementieren können, um die Möglichkeiten zu erweitern, wie die Laufzeitumgebung festlegt, welche Angebote zu empfehlen sind. Keine dieser Funktionen garantiert, dass ein Angebot einem Kunden präsentiert wird. Diese Funktionen beeinflussen die Wahrscheinlichkeit, dass ein Angebot auswählbar ist, um einem Kunden präsentiert zu werden. Sie können so viele oder so wenige dieser Funktionen verwenden, die Sie benötigen, um die beste Lösung für Ihre Umgebung zu implementieren.

Es gibt drei wesentliche Bereiche, in denen Sie die Angebotseignung beeinflussen können: Erstellen einer Liste von möglichen Angeboten, Bestimmen des Marketing-Score und die Lernfunktion.

Erstellen einer Liste von möglichen Angeboten

Das Erstellen einer Liste von möglichen Angeboten besteht aus zwei Hauptschritten. Der erste Schritt ist die Generierung einer Liste aller möglichen Angebote, für die der Kunde möglicherweise infrage kommt. Der zweite Schritt ist das Herausfiltern von Angeboten, für die der Kunde nicht mehr infrage kommt. Es gibt mehrere Stellen in beiden Schritten, an denen Sie die Generierung der Liste von möglichen Angeboten beeinflussen können.

Dieses Diagramm zeigt die Schritte der Generierung der Liste von möglichen Angeboten. Die Pfeile zeigen den Ausführungsvorrang an. Beispiel: Wenn ein Angebot den Filter **Max. Anzahl Anzeigewiederholungen eines Angebots** passiert, aber am Filter **Globale Angebotsaufnahmeregeln** scheitert, schließt die Laufzeitumgebung das Angebot aus.

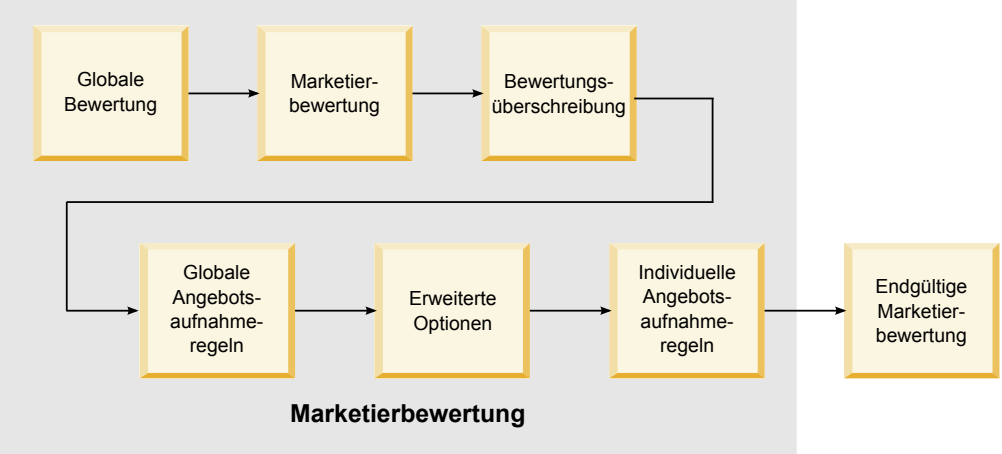


- **Globale Angebotszuweisungen** - Sie können globale Angebote nach Zielgruppenebene mithilfe der globalen Angebotstabelle definieren.
- **Verfahrensregeln** - Die grundlegende Methode, um Angebote nach Segment durch Interaktionspunkt mithilfe der Registerkarte "Interaktionsstrategie" zu definieren.
- **Individuelle Angebotszuweisungen** - Sie können spezielle Angebotszuweisungen nach Kunde mithilfe der Bewertungsüberschreibungstabelle definieren.
- **Angebotsablaufdatum** - Wenn Sie ein Angebot in Campaign erstellen, können Sie ein Ablaufdatum definieren. Wenn das Ablaufdatum für ein Angebot überschritten wurde, schließt die Laufzeitumgebung das Angebot aus.
- **Kampagnenstart- und -enddatum** - Wenn Sie eine Kampagne in Campaign erstellen, können Sie ein Start- und Enddatum für die Kampagne definieren. Wenn das Startdatum für die Kampagne noch nicht eingetreten ist oder das Enddatum für die Kampagne überschritten wurde, schließt die Laufzeitumgebung das Angebot aus.
- **Angebotsunterdrückung** - Sie können eine Angebotsunterdrückung für bestimmte Zielgruppenmitglieder mithilfe der Tabelle für Angebotsunterdrückung definieren.
- **Max. Anzahl Anzeigewiederholungen eines Angebots** - Wenn Sie einen interaktiven Kanal definieren, legen Sie die maximale Häufigkeit fest, mit der ein Angebot einem Kunden pro Sitzung bereitgestellt wird. Wenn ein Angebot bereits mit dieser Häufigkeit bereitgestellt wurde, schließt die Laufzeitumgebung das Angebot aus.
- **Globale Angebotsaufnahmeregeln** - Sie können einen booleschen Ausdruck definieren, um Angebote auf einer Zielgruppenebene mithilfe der globalen Angebotstabelle zu filtern. Wenn das Ergebnis "false" ist, schließt die Laufzeitumgebung das Angebot aus.
- **Erweiterte Optionen** - Sie können die erweiterte Option **Diese Regel als berechtigt betrachten, wenn der folgende Ausdruck wahr ist** in einer Verfahrensregel verwenden, um Angebote auf einer Segmentebene zu filtern. Wenn das Ergebnis "false" ist, schließt die Laufzeitumgebung das Angebot aus.
- **Individuelle Angebotsaufnahmeregeln** - Sie können einen booleschen Ausdruck definieren, um Angebote auf einer Kundenebene mithilfe der Bewertungsüberschreibungstabelle zu filtern. Wenn das Ergebnis "false" ist, schließt die Laufzeitumgebung das Angebot aus.

Berechnen des Marketing-Score

Es gibt viele Möglichkeiten, den Marketing-Score zu beeinflussen (durch Verwendung einer Berechnung) oder sie zu überschreiben.

Dieses Diagramm zeigt die verschiedenen Phasen, in denen Sie den Marketing-Score beeinflussen oder überschreiben können.



Die Pfeile zeigen den Ausführungsvorrang an. Wenn Sie beispielsweise einen Ausdruck definieren, um den Marketing-Score in den erweiterten Optionen für eine Verfahrensregel zu definieren, und einen Ausdruck in der Bewertungsüberschreibungstabelle definieren, hat der Ausdruck in der Bewertungsüberschreibungstabelle Vorrang.

- **Globale Bewertung** - Sie können eine Bewertung pro Zielgruppenebene unter Verwendung der globalen Angebotstabelle definieren.
- **Marketierbewertung** - Sie können eine Bewertung pro Segment unter Verwendung des Schiebereglers in einer Verfahrensregel definieren.
- **Bewertung aus Bewertungsüberschreibung** - Sie können eine Bewertung pro Kunde unter Verwendung der Bewertungsüberschreibungstabelle definieren.
- **Globale Angebotsaufnahmeregeln** - Sie können einen Ausdruck definieren, der eine Bewertung pro Zielgruppenebene unter Verwendung der globalen Angebotstabelle berechnet.
- **Erweiterte Optionen** - Sie können einen Ausdruck definieren, der eine Bewertung pro Segment unter Verwendung der erweiterten Option **Folgenden Ausdruck als Marketing-Score verwenden** in einer Verfahrensregel berechnet.
- **Angebotsaufnahmeregeln für Bewertungsüberschreibung** - Sie können einen Ausdruck definieren, der eine Bewertung pro Kunde unter Verwendung der Bewertungsüberschreibungstabelle berechnet.

Den Lernprozess beeinflussen

Wenn Sie das integrierte Lernmodul von Interact verwenden, können Sie die Lernausgabe über die Standardlernkonfigurationen wie die Liste von Lernattributen oder die Zuverlässigkeitsstufe hinaus beeinflussen. Sie können Komponenten des Lernalgorithmus außer Kraft setzen, während Sie die verbleibenden Komponenten verwenden.

Sie können die Lernfunktion mithilfe der Spalten `LikelihoodScore` und `AdjExploreScore` der Standardangebots- und Bewertungsüberschreibungstabellen überschreiben. Sie können diese Spalten den Standardangebots- und Bewertungsüberschreibungstabellen mithilfe des Funktionsscripts `aci_scoringfeature` hinzufü-

gen. Um diese Überschreibungen ordnungsgemäß verwenden zu können, bedarf es eines umfassenden Verständnisses der integrierten Lernfunktion von Interact.

Das Lernmodul nimmt eine Liste von möglichen Angeboten und die Marketing-Scores pro möglichem Angebot und verwendet sie in den endgültigen Berechnungen. Die Angebotsliste wird mit den Lernattributen verwendet, um die Wahrscheinlichkeit (Annahmewahrscheinlichkeit) zu berechnen, dass der Kunde das Angebot annehmen wird. Unter Verwendung dieser Wahrscheinlichkeiten sowie der Langzeitanzahl von Präsentationen, um zwischen Untersuchung und Nutzung auszugleichen, ermittelt der Lernalgorithmus die Angebotsgewichtung. Schließlich nimmt die integrierte Lernfunktion die Angebotsgewichtung, multipliziert sie mit dem endgültigen Marketing-Score und gibt eine endgültige Bewertung zurück. Die Angebote werden nach dieser endgültigen Bewertung sortiert.

Unterdrücken von Angeboten

Sie können die Laufzeitumgebung so konfigurieren, dass Angebote unterdrückt werden.

Es stehen mehrere Möglichkeiten zur Verfügung, wie die Laufzeitumgebung ein Angebot unterdrückt:

- Mithilfe des Elements **Max. Anzahl Anzeigewiederholungen eines Angebots während eines Besuchs** eines interaktiven Kanals.

  Sie definieren den Wert für **Max. Anzahl Anzeigewiederholungen eines Angebots während eines Besuchs** beim Erstellen oder Bearbeiten eines interaktiven Kanals.
- Mithilfe einer Tabelle für Angebotsunterdrückung.

  Sie erstellen eine Tabelle für Angebotsunterdrückung in Ihrer Profildatenbank.
- Mithilfe von Angeboten, deren Ablaufdatum überschritten wurde.
- Mithilfe von Angeboten aus abgelaufenen Kampagnen.
- Mithilfe von Angeboten, die ausgeschlossen wurden, weil sie einer Angebotsaufnahmeregel nicht entsprechen (erweiterte Option der Verfahrensregel).
- Angebote werden in einer Interact-Sitzung explizit angenommen oder abgelehnt. Wenn ein Kunde ein Angebot explizit annimmt oder ablehnt, wird dieses Angebot für die Dauer der Sitzung unterdrückt.

Aktivieren der Angebotsunterdrückung

Mit diesem Verfahren können Sie die Angebotsunterdrückung aktivieren.

Informationen zu diesem Vorgang

Sie können Interact so konfigurieren, dass es auf eine Liste unterdrückter Angebote verweist.

Vorgehensweise

1. Erstellen Sie eine Tabelle des Typs `offerSuppressionTable`. Dabei handelt es sich um eine neue Tabelle für jede Zielgruppe, in der die Zielgruppen-ID und die Angebots-ID enthalten sind.
2. Legen Sie die Eigenschaft `enableOfferSuppressionLookup` auf **true** fest.
3. Legen Sie für die Eigenschaft `Interact > Profil > offerSuppressionTable` den Namen der Angebotsunterdrückungstabelle für die entsprechende Zielgruppe fest.

Tabelle für Angebotsunterdrückung

Die Tabelle für Angebotsunterdrückung ermöglicht es Ihnen, ein Angebot für eine bestimmte Zielgruppen-ID zu unterdrücken. Beispiel: Wenn Ihre Zielgruppe Kunde ist, können Sie ein Angebot für Kunden John Smith unterdrücken. Eine Version dieser Tabelle für zumindest eine Zielgruppenebene muss in Ihrer Produktionsprofildatenbank vorhanden sein. Sie können eine Beispiel-Tabelle für Angebotsunterdrückung `UACI_BlackList` erstellen, indem Sie das SQL-Script `aci_usertab` in Ihrer Profildatenbank ausführen. Das SQL-Script `aci_usertab` befindet sich im `ddl`-Verzeichnis in Ihrem Laufzeitumgebungsinstallationsverzeichnis.

Sie müssen die Felder `AudienceID` und `OfferCode1` für jede Zeile definieren. Sie können zusätzliche Spalten hinzufügen, wenn Ihre Zielgruppen-ID oder Ihr Angebotscode aus mehreren Spalten besteht. Diese Spalten müssen mit den in Campaign definierten Spaltennamen übereinstimmen. Beispiel: Wenn Sie die Zielgruppe `Customer` anhand der Felder `HHold_ID` und `MemberNum` definieren, müssen Sie `HHold_ID` und `MemberNum` der Tabelle für Angebotsunterdrückung hinzufügen.

Name	Beschreibung
AudienceID	(Erforderlich) Der Name dieser Spalte muss mit dem Namen der Spalte übereinstimmen, die die Zielgruppen-ID in Campaign definiert. Wenn Ihre Zielgruppen-ID mehrere Spalten umfasst, können Sie sie dieser Tabelle hinzufügen. Jede Zeile muss die Zielgruppen-ID enthalten, die Sie dem Standardangebot zuweisen, z. B. `customer1`.
OfferCode1	(Erforderlich) Der Angebotscode für das Angebot, das Sie überschreiben. Wenn Ihre Angebotscodes aus mehreren Feldern bestehen, können Sie die zusätzlichen Spalten hinzufügen, z. B. `OfferCode2` etc.

Globale Angebote und individuelle Zuweisungen

Sie können die Laufzeitumgebung konfigurieren, um bestimmte Angebote außerhalb der Verfahrensregeln zuzuweisen, die auf der Registerkarte "Interaktionsstrategie" konfiguriert sind. Sie können globale Angebote für jedes Mitglied der Zielgruppenebene und individuelle Zuweisungen für bestimmte Zielgruppenmitglieder definieren. Beispiel: Sie können ein globales Angebot für alle Haushalte definieren, die angezeigt werden, wenn keine anderen verfügbar sind, und dann eine individuelle Angebotszuweisung für den bestimmten Smith-Haushalt erstellen.

Sie können sowohl die globalen Angebote als auch individuelle Zuweisungen durch Zone, Zelle und andere Angebotsaufnahmeregeln beschränken. Globale Angebote und individuelle Zuweisungen werden beide konfiguriert, indem Daten bestimmten Tabellen in Ihrer Produktionsprofildatenbank hinzugefügt werden.

Damit globale Angebote und individuelle Zuweisungen ordnungsgemäß funktionieren, müssen alle referenzierten Zell- und Angebotscodes in der Implementierung vorhanden sein. Um sicherzustellen, dass erforderliche Daten verfügbar sind, müssen Sie Standardzellencodes und die Tabelle `UACI_ICBatchOffers` konfigurieren.

Definieren der Standardzellencodes

Wenn Sie die Standardangebots- oder Bewertungsüberschreibungstabellen für globale oder individuelle Angebotszuweisungen verwenden, müssen Sie Standardzellencodes definieren. Der Standardzellencode (`DefaultCellCode`) wird verwendet, wenn kein Zellencode in einer bestimmten Zeile in den Standardangebots- oder Bewertungsüberschreibungstabellen definiert wurde. Die Berichterstellung verwendet diesen Standardzellencode.

Informationen zu diesem Vorgang

Der Wert von `DefaultCellCode` muss dem in Campaign definierten Zellencodeformat entsprechen. Dieser Zellencode wird bei allen Angebotszuweisungen verwendet, die in der Berichterstellung angezeigt werden.

Wenn Sie eindeutige Standardzellencodes definieren, können Sie ohne großen Aufwand Angebote identifizieren, die durch die Standardangebots- oder Bewertungsüberschreibungstabellen zugewiesen wurden.

Vorgehensweise

Definieren Sie die Eigenschaft `DefaultCellCode` für jede Zielgruppenebene und jeden Tabellentyp in der Kategorie `IndividualTreatment`.

Definieren von Angeboten, die nicht in einer Verfahrensregel verwendet werden

Wenn Sie die Standardangebots- oder Bewertungsüberschreibungstabellen verwenden, müssen Sie sicherstellen, dass alle Angebotscodes in der Bereitstellung vorhanden sind. Wenn Sie wissen, dass alle Angebote, die Sie in den Standardangebots- oder Bewertungsüberschreibungstabellen verwenden, in Ihren Verfahrensregeln verwendet werden, sind die Angebote in der Implementierung vorhanden. Alle Angebote, die nicht in einer Verfahrensregel verwendet werden, müssen jedoch in der Tabelle `UACI_ICBatchOffers` definiert werden.

Informationen zu diesem Vorgang

Die Tabelle `UACI_ICBatchOffers` existiert in den Campaign-Systemtabellen.

Vorgehensweise

Füllen Sie die Tabelle `UACI_ICBatchOffers` mit Angebotscodes aus, die in den Standardangebots- oder Bewertungsüberschreibungstabellen verwendet werden. Die Tabelle hat das folgende Format:

Spaltenname	Typ	Beschreibung
ICName	varchar(64)	Der Name des interaktiven Kanals, dem das Angebot zugeordnet ist. Wenn Sie dasselbe Angebot mit zwei verschiedenen interaktiven Kanälen verwenden, müssen Sie eine Zeile für jeden interaktiven Kanal bereitstellen.
OfferCode1	varchar(64)	Der erste Teil des Angebotscodes.
OfferCode2	varchar(64)	Der zweite Teil des Angebotscodes.
OfferCode3	varchar(64)	Der dritte Teil des Angebotscodes.
OfferCode4	varchar(64)	Der vierte Teil des Angebotscodes.
OfferCode5	varchar(64)	Der fünfte Teil des Angebotscodes.

Informationen zur globalen Angebotstabelle

Die globale Angebotstabelle ermöglicht es Ihnen, Verfahren auf der Zielgruppenebene zu definieren. Sie können beispielsweise ein globales Angebot für jedes Mitglied der Zielgruppe "Haushalt" definieren.

Sie können globale Einstellungen für die folgenden Elemente der Interact-Angebotsbereitstellung definieren.

- Globale Angebotszuweisungen
- Globale Marketierbewertung, durch eine Zahl oder durch einen Ausdruck
- Boolescher Ausdruck, um Angebote zu filtern
- Lernwahrscheinlichkeit und Gewichtung, wenn Sie die integrierte Interact-Lernfunktion verwenden
- Globale Lernfunktion-Überschreibung

Zuweisen von globalen Angeboten

Mit diesem Verfahren können Sie die Laufzeitumgebung konfigurieren, um globale Angebote für eine Zielgruppenebene außerhalb der Definitionen in den Verfahrensregeln zuzuweisen.

Vorgehensweise

1. Erstellen Sie eine Tabelle namens `UACI_DefaultOffers` in Ihrer Profildatenbank.

   Verwenden Sie die DDL-Datei `aci_usrtab`, damit die Tabelle `UACI_DefaultOffers` mit den korrekten Spalten erstellt wird.
2. Legen Sie die Eigenschaft `enableDefaultOfferLookup` auf **true** fest.

Globale Angebotstabelle

Die globale Angebotstabelle muss in Ihrer Profildatenbank existieren. Sie können die globale Angebotstabelle `UACI_DefaultOffers` erstellen, indem Sie das SQL-Script `aci_usertab` in Ihrer Profildatenbank ausführen.

Das SQL-Script `aci_usertab` befindet sich im `ddl`-Verzeichnis in Ihrem Laufzeitumgebungsinstallationsverzeichnis.

Sie müssen die Felder `AudienceLevel` und `OfferCode1` für jede Zeile definieren. Die anderen Felder sind optional, um Ihre Angebotszuweisungen weiter zu beschränken oder um die integrierte Lernfunktion auf der Zielgruppenebene zu beeinflussen.

Zur Erzielung einer optimalen Leistung sollten Sie einen Index auf der Zielgruppenebenenspalte dieser Tabelle erstellen.

Name	Typ	Beschreibung
AudienceLevel	varchar(64)	(Erforderlich) Der Name der Zielgruppenebene, der Sie das Standardangebot zuweisen, z. B. `Kunde` oder `Haushalt`. Dieser Name muss mit der in Campaign definierten Zielgruppenebene übereinstimmen.
OfferCode1	varchar(64)	(Erforderlich) Der Angebotscode des Standardangebots. Wenn Ihre Angebotscodes aus mehreren Feldern bestehen, können Sie die zusätzlichen Spalten hinzufügen, z. B. `OfferCode2` etc. Wenn Sie dieses Angebot hinzufügen, um eine globale Angebotszuweisung bereitzustellen, müssen Sie dieses Angebot der Tabelle `UACI_ICBatchOffers` hinzufügen.
Bewertung	float	Eine Zahl, um den Marketing-Score für diese Angebotszuweisung zu definieren.

Name	Typ	Beschreibung
OverrideTypeID	int	Wenn auf 1 gesetzt und das Angebot nicht in der Liste von möglichen Angeboten aufscheint, dieses Angebot der Liste hinzufügen sowie Bewertungsdaten für das Angebot verwenden. Verwenden Sie im Allgemeinen 1, um globale Angebotszuweisungen bereitzustellen. Wenn auf `0`, *null* oder auf eine andere Zahl als 1 festgelegt, Daten für das Angebot nur verwenden, wenn das Angebot in der Liste von möglichen Angeboten aufscheint. In den meisten Fällen wird eine Verfahrensregel oder eine individuelle Zuweisung diese Einstellung überschreiben.
Predicate	varchar(4000)	Sie können einen Ausdruck in diese Spalte als erweiterte Optionen für Verfahrensregeln eingeben. Sie können dieselben Variablen und Makros verwenden, die Ihnen beim Schreiben von erweiterten Optionen für Verfahrensregeln verfügbar sind. Das Verhalten dieser Spalte hängt vom Wert der Spalte `EnableStateID` ab. • Wenn `EnableStateID` `2` ist, funktioniert diese Spalte wie die Option **Diese Regel als berechtigt betrachten, wenn der folgende Ausdruck wahr ist** in den erweiterten Optionen für Verfahrensregeln, um diese Angebotszuweisung zu beschränken. Diese Spalte muss einen booleschen Ausdruck enthalten und zu "true" auflösen, um dieses Angebot aufzunehmen. Wenn Sie versehentlich einen Ausdruck definieren, der in eine Zahl aufgelöst wird, wird jede Zahl ungleich null als "true" und null als "false" angesehen. • Wenn `EnableStateID` `3` ist, funktioniert diese Spalte wie die Option **Folgenden Ausdruck als Marketing-Score verwenden** in den erweiterten Optionen für Verfahrensregeln, um dieses Angebot zu beschränken. Diese Spalte muss einen Ausdruck enthalten, der in eine Zahl aufgelöst wird. • Wenn `EnableStateID` `1` ist, ignoriert Interact jeden Wert in dieser Spalte.
FinalScore	float	Eine Zahl zum Überschreiben der endgültigen Bewertung, die zum Sortieren der endgültigen Liste der zurückgegebenen Angebote verwendet wird. Diese Spalte wird verwendet, wenn Sie das integrierte Lernmodul aktiviert haben. Sie können Ihre eigene Lernfunktion implementieren, um diese Spalte zu verwenden.

Name	Typ	Beschreibung
CellCode	varchar(64)	Der Zellencode für ein interaktives Segment, dem Sie dieses Standardangebot zuweisen wollen. Wenn Ihre Zellencodes aus mehreren Feldern bestehen, können Sie die zusätzlichen Spalten hinzufügen. Sie müssen einen Zellencode bereitstellen, wenn `OverrideTypeID` 0 oder null ist. Wenn Sie keinen Zellencode aufnehmen, ignoriert die Laufzeitumgebung diese Datenzeile. Wenn `OverrideTypeID` 1 ist, müssen Sie in dieser Spalte keinen Zellencode bereitstellen. Wenn Sie keinen Zellencode bereitstellen, verwendet die Laufzeitumgebung den Zellencode, der in der Eigenschaft `DefaultCellCode` definiert ist, für diese Zielgruppenebene und Tabelle für Berichterstellungszwecke.
Zone	varchar(64)	Der Name der Zone, für die diese Angebotszuweisung gelten soll. Wenn NULL gilt dies für alle Zonen.
EnableStateID	int	Der Wert in dieser Spalte definiert das Verhalten der Spalte `Predicate`. • **1** - Die Spalte `Predicate` nicht verwenden. • **2** - `Predicate` als einen booleschen Ausdruck verwenden, um das Angebot zu filtern. Dies befolgt dieselben Regeln wie die erweiterte Option **Diese Regel als berechtigt betrachten, wenn der folgende Ausdruck wahr ist** in einer Verfahrensregel. • **3** - `Predicate` verwenden, um die Marketierbewertung zu definieren. Dies befolgt dieselben Regeln wie die erweiterte Option **Der folgenden Ausdruck als Marketing-Score verwenden** in einer Verfahrensregel. Zeilen, bei denen diese Spalte Null oder ein anderer Wert als 2 oder 3 ist, ignorieren die Spalte `Predicate`.
LikelihoodScore	float	Diese Spalte wird nur verwendet, um die integrierte Lernfunktion zu beeinflussen. Sie können diese Spalte mit der DDL `aci_scoringfeature` hinzufügen.
AdjExploreScore	float	Diese Spalte wird nur verwendet, um die integrierte Lernfunktion zu beeinflussen. Sie können diese Spalte mit der DDL `aci_scoringfeature` hinzufügen.

Informationen zur Bewertungsüberschreibungstabelle

Die Bewertungsüberschreibungstabelle ermöglicht es Ihnen, Verfahren auf einer Zielgruppen-ID- oder einer individuellen Ebene zu definieren. Beispiel: Wenn Ihre Zielgruppenebene Besucher ist, können Sie Überschreibung für bestimmte Besucher erstellen.

Sie können Überschreibungen für die folgenden Elemente der Interact-Angebotsbereitstellung definieren.

- Individuelle Angebotszuweisung
- Individuelle Marketierbewertung, durch eine Zahl oder durch einen Ausdruck
- Boolescher Ausdruck, um Angebote zu filtern
- Lernwahrscheinlichkeit und Gewichtung, wenn Sie die integrierte Lernfunktion verwenden
- Individuelle Lernfunktion-Überschreibung

Konfigurieren von Bewertungsüberschreibungen

Sie können in der Konfiguration von Interact festlegen, dass anstelle des Marketing-Score eine Bewertung verwendet wird, die auf Basis einer Modellierungsanwendung generiert wird.

Vorgehensweise

1. Erstellen Sie eine Bewertungsüberschreibungstabelle für jede Zielgruppenebene, der Sie Überschreibungen bereitstellen möchten.

   Mithilfe der DDL-Datei `aci_usrtab` können Sie eine Beispiel-Bewertungsüberschreibungstabelle mit den korrekten Spalten erstellen.
2. Legen Sie die Eigenschaft `enableScoreOverrideLookup` auf **true** fest.
3. Legen Sie die Eigenschaft `scoreOverrideTable` auf den Namen der Bewertungsüberschreibungstabelle für jede Zielgruppenebene fest, der Sie Überschreibungen bereitstellen möchten.

   Sie müssen nicht eine Bewertungsüberschreibungstabelle für jede Zielgruppenebene bereitstellen.

Bewertungsüberschreibungstabelle

Die Bewertungsüberschreibungstabelle muss in Ihrer Produktionsprofildatenbank existieren. Sie können eine Beispiel-Bewertungsüberschreibungstabelle `UACI_ScoreOverride` erstellen, indem Sie das SQL-Script `aci_usertab` in Ihrer Profildatenbank ausführen.

Das SQL-Script `aci_usertab` befindet sich im `ddl`-Verzeichnis in Ihrem Laufzeitumgebungsinstallationsverzeichnis.

Sie müssen die Felder *`AudienceID`*, `OfferCode1` und `Score` für jede Zeile definieren. Die Werte in den anderen Feldern sind optional, um Ihre einzelnen Angebotszuweisungen weiter zu beschränken oder um Bewertungsüberschreibungsinformationen für die integrierte Lernfunktion bereitzustellen.

Name	Typ	Beschreibung
AudienceID	varchar(64)	(Erforderlich) Der Name dieser Spalte muss mit dem Namen der Spalte übereinstimmen, die die Zielgruppen-ID in Campaign definiert. Die Beispieltabelle, die durch die DLL-Datei `aci_usertab` erstellt wird, erstellt diese Spalte als die Spalte `CustomerID`. Wenn Ihre Zielgruppen-ID mehrere Spalten umfasst, können Sie sie dieser Tabelle hinzufügen. Jede Zeile muss die Zielgruppen-ID enthalten, die Sie dem einzelnen Angebot zuweisen, z. B. `customer1`. Zur Erzielung einer optimalen Leistung sollten Sie einen Index auf dieser Spalte erstellen.

Name	Typ	Beschreibung
OfferCode1	varchar(64)	(Erforderlich) Der Angebotscode für das Angebot. Wenn Ihre Angebotscodes aus mehreren Feldern bestehen, können Sie die zusätzlichen Spalten hinzufügen, z. B. `OfferCode2` etc. Wenn Sie dieses Angebot hinzufügen, um eine einzelne Angebotszuweisung bereitzustellen, müssen Sie dieses Angebot der Tabelle `UACI_ICBatchOffers` hinzufügen.
Score	float	Eine Zahl, um den Marketing-Score für diese Angebotszuweisung zu definieren.
OverrideTypeID	int	Wenn auf `0` oder *null* (oder auf eine andere Zahl als 1) festgelegt, Daten für das Angebot nur verwenden, wenn das Angebot in der Liste von möglichen Angeboten aufscheint. Verwenden Sie im Allgemeinen `0`, um Bewertungsüberschreibungen bereitzustellen. Sie müssen einen Zellknoten bereitstellen. Wenn auf `1` gesetzt und das Angebot nicht in der Liste von möglichen Angeboten aufscheint, dieses Angebot der Liste hinzufügen sowie Bewertungsdaten für das Angebot verwenden. Verwenden Sie im Allgemeinen `1`, um einzelne Angebotszuweisungen bereitzustellen.
Predicate	varchar(4000)	Sie können einen Ausdruck in diese Spalte als erweiterte Optionen für Verfahrensregeln eingeben. Sie können dieselben Variablen und Makros verwenden, die Ihnen beim Schreiben von erweiterten Optionen für Verfahrensregeln verfügbar sind. Das Verhalten dieser Spalte hängt vom Wert der Spalte `EnableStateID` ab. • Wenn `EnableStateID` `2` ist, funktioniert diese Spalte wie die Option **Diese Regel als berechtigt betrachten, wenn der folgende Ausdruck wahr ist** in den erweiterten Optionen für Verfahrensregeln, um diese Angebotszuweisung zu beschränken. Diese Spalte muss einen booleschen Ausdruck enthalten und zu "true" auflösen, um dieses Angebot aufzunehmen. Wenn Sie versehentlich einen Ausdruck definieren, der in eine Zahl aufgelöst wird, wird jede Zahl ungleich null als "true" und null als "false" angesehen. • Wenn `EnableStateID` `3` ist, funktioniert diese Spalte wie die Option **Folgenden Ausdruck als Marketing-Score verwenden** in den erweiterten Optionen für Verfahrensregeln, um dieses Angebot zu beschränken. Diese Spalte muss einen Ausdruck enthalten, der in eine Zahl aufgelöst wird. • Wenn `EnableStateID` `1` ist, ignoriert Interact jeden Wert in dieser Spalte.

Name	Typ	Beschreibung
FinalScore	float	Eine Zahl zum Überschreiben der endgültigen Bewertung, die zum Sortieren der endgültigen Liste der zurückgegebenen Angebote verwendet wird. Diese Spalte wird verwendet, wenn Sie das integrierte Lernmodul aktiviert haben. Sie können Ihre eigene Lernfunktion implementieren, um diese Spalte zu verwenden.
CellCode	varchar(64)	Der Zellencode für ein interaktives Segment, dem Sie dieses Angebot zuweisen wollen. Wenn Ihre Zellencodes aus mehreren Feldern bestehen, können Sie die zusätzlichen Spalten hinzufügen. Sie müssen einen Zellencode bereitstellen, wenn `OverrideTypeID` 0 oder null ist. Wenn Sie keinen Zellencode aufnehmen, ignoriert die Laufzeitumgebung diese Datenzeile. Wenn `OverrideTypeID` 1 ist, müssen Sie in dieser Spalte keinen Zellencode bereitstellen. Wenn Sie keinen Zellencode bereitstellen, verwendet die Laufzeitumgebung den Zellencode, der in der Eigenschaft `DefaultCellCode` definiert ist, für diese Zielgruppenebene und Tabelle für Berichterstellungszwecke.
Zone	varchar(64)	Der Name der Zone, für die diese Angebotszuweisung gelten soll. Wenn NULL gilt dies für alle Zonen.
EnableStateID	int	Der Wert in dieser Spalte definiert das Verhalten der Spalte `Predicate`. • **1** - Die Spalte `Predicate` nicht verwenden. • **2** - `Predicate` als einen booleschen Ausdruck verwenden, um das Angebot zu filtern. Dies befolgt dieselben Regeln wie die erweiterte Option **Diese Regel als berechtigt betrachten, wenn der folgende Ausdruck wahr ist** in einer Verfahrensregel. • **3** - `Predicate` verwenden, um die Marketierbewertung zu definieren. Dies befolgt dieselben Regeln wie die erweiterte Option **Der folgenden Ausdruck als Marketing-Score verwenden** in einer Verfahrensregel. Zeilen, bei denen diese Spalte Null oder ein anderer Wert als 2 oder 3 ist, ignorieren die Spalte `Predicate`.
LikelihoodScore	float	Diese Spalte wird nur verwendet, um die integrierte Lernfunktion zu beeinflussen. Sie können diese Spalte mit der DDL `aci_scoringfeature` hinzufügen.
AdjExploreScore	float	Diese Spalte wird nur verwendet, um die integrierte Lernfunktion zu beeinflussen. Sie können diese Spalte mit der DDL `aci_scoringfeature` hinzufügen.

Übersicht über das integrierte Lernen von Interact

Während Sie alle Anstrengungen unternehmen, um sicherzustellen, dass den richtigen Segmenten die richtigen Angebote vorgeschlagen werden, können Sie jederzeit für Sie Wichtiges aus der tatsächlichen Auswahl Ihrer Besucher lernen. Das tatsächliche Verhalten der Besucher sollte Ihre Strategie beeinflussen. Sie können den Antwortverlauf verwenden und ihn einige Modellierungstools durchlaufen lassen, um eine Bewertung zu erhalten. Diesen können Sie in die interaktiven Ablaufdiagramme einbeziehen.

Dabei handelt es sich jedoch um keine Echtzeitdaten.

Interact bietet Ihnen zwei Optionen, damit Sie aus den Aktionen der Besucher in Echtzeit lernen können.

- Integriertes Lernmodul - Die Laufzeitumgebung verfügt über ein naives bayessches Lernmodul. Dieses Modul überwacht die Kundenattribute Ihrer Wahl und verwendet diese Daten als Hilfe beim Auswählen der Angebote, die angezeigt werden sollen.
- Lern-API - Die Laufzeitumgebung verfügt auch über eine Lern-API, über die Sie Ihr eigenes Lernmodul schreiben können.

Sie müssen die Lernfeatures nicht verwenden. Diese Features sind standardmäßig inaktiviert.

Interact-Lernmodul

Das Interact-Lernmodul überwacht die Antworten des Besuchers auf Angebote und Besucherattribute.

Modi des Lernmoduls

Das Lernmodul weist zwei allgemeine Modi auf:

- Untersuchung - das Lernmodul verarbeitet Angebote, um genügend Antwortdaten zusammenzustellen, damit die im Nutzungsmodus verwendete Einschätzung optimiert werden kann. Die in der Untersuchungsphase angezeigten Angebote spiegeln nicht unbedingt die optimale Auswahl wider.
- Nutzung - Nachdem genügend Daten in der Untersuchungsphase gesammelt wurden, wählt das Lernmodul die anzuzeigenden Angebote auf Basis der Wahrscheinlichkeiten aus.

Das Lernmodul verwendet für den alternativen Einsatz des Untersuchungs- und Nutzungsmodus jeweils zwei Eigenschaften. Dabei handelt es sich um die folgenden beiden Eigenschaften:

- eine Zuverlässigkeitsstufe, die Sie mit der Eigenschaft `confidenceLevel` konfigurieren.
- eine Wahrscheinlichkeit, dass das Lernmodul ein Zufallsangebot anzeigt. Diesen Wert konfigurieren Sie mit der Eigenschaft `percentRandomSelection`.

Eigenschaft für die Zuverlässigkeitsstufe

Für die Eigenschaft `confidenceLevel` legen Sie einen Prozentsatz fest, der angibt, wie sicher sich das Lernmodul sein muss, bevor seine Bewertungen für ein Angebot im Auswahlverfahren verwendet werden. Wenn dem Lernmodul anfangs keine Daten zur Verarbeitung vorliegen, ist es ausschließlich auf den Marketing-Score angewiesen. Nachdem jedes Angebot mit einer durch `minPresentCountThreshold` defi-

nierten Häufigkeit angezeigt wurde, wechselt das Lernmodul in den Untersuchungsmodus. Das Lernmodul benötigt eine große Datenmenge zum Arbeiten, um überzeugt davon zu sein, dass die von ihm berechneten Prozentsätze korrekt sind. Deshalb bleibt es im Untersuchungsmodus.

Das Lernmodul ordnet jedem Angebot Gewichte zu. Zum Berechnen der Gewichtungen verwendet das Lernmodul eine Formel, die auf der konfigurierten Zuverlässigkeitsstufe sowie auf historischen Annahmedaten und den aktuellen Sitzungsdaten basiert. Die Formel gleicht von vornherein zwischen Untersuchung und Nutzung aus und gibt das entsprechende Gewicht zurück.

Eigenschaft für die Zufallsauswahl

Um sicherzustellen, dass das System nicht Angebote bevorzugt, die in den frühen Stadien am besten abschneiden, zeigt Interact ein Zufallsangebot über die mit `percentRandomSelection` festgelegte prozentuale Zeit an. Durch diesen Prozentsatz für Zufallsangebote wird das Lernmodul gezwungen, andere Angebote als die erfolgreichsten zu empfehlen, und kann so ermitteln, ob andere Angebote erfolgreicher wären, wenn sie prominenter präsentiert würden. Wenn Sie `percentRandomSelection` z. B. mit dem Wert 5 konfigurieren, bedeutet dies, dass das Lernmodul während 5 % der Zeit ein Zufallsangebot anzeigt und die Antwortdaten seinen Berechnungen hinzufügt.

Mit dem Wert für **% Zufallswert** können Sie im Fenster "Interaktiver Kanal" in der Registerkarte "Interaktionspunkte" für jede Zone die Wahrscheinlichkeit angeben, mit der das zurückgegebene Angebot zufällig ausgewählt wird.

Vorgehensweise des Lernmoduls beim Bestimmen von Angeboten

Das Lernmodul bestimmt auf folgende Art, welche Angebote präsentiert werden.

1. Es berechnet die Wahrscheinlichkeit, mit der ein Besucher ein Angebot auswählt.
2. Es berechnet die Angebotsgewichtung mithilfe der Wahrscheinlichkeit aus Schritt 1 und ermittelt, ob es sich im Untersuchungs- oder im Nutzungsmodus befinden sollte.
3. Es berechnet die endgültige Bewertung für jedes Angebot unter Verwendung des Marketing-Score und der Angebotsgewichtung aus Schritt 2.
4. Es sortiert die Angebote nach den in Schritt 3 ermittelten Bewertungen und gibt die angeforderte Anzahl von Spitzenangeboten zurück.

Beispielsweise ermittelt das Lernmodul, dass ein Besucher Angebot A wahrscheinlich zu 30 % und Angebot B wahrscheinlich zu 70 % annimmt und dass es diese Informationen nutzen sollte. Anhand der Verfahrensregeln liegt der Marketing-Score für Angebot A bei 75 und für Angebot B bei 55. Da die Bewertung aufgrund der Berechnungen in Schritt 3 jedoch für Angebot B höher als für Angebot A liegt, empfiehlt die Laufzeitumgebung Angebot B.

Eigenschaften für den Gewichtungsfaktor

Der Lernprozess basiert auch auf der Eigenschaft `recencyWeightingFactor` und der Eigenschaft `recencyWeightingPeriod`. Diese Eigenschaften ermöglichen es Ihnen, aktuelleren Daten gegenüber älteren Daten mehr Gewicht zu verleihen. Der Wert von `recencyWeightingFactor` ist der Prozentsatz der Gewichtung, der den aktuellen Daten zugewiesen werden soll. Der Wert von `recencyWeightingPeriod` ist die

aktuelle Dauer. So konfigurieren Sie beispielsweise `recencyWeightingFactor` mit 0,30 und `recencyWeightingPeriod` mit 24. Diese Einstellungen bedeuten, dass es sich bei den Daten der vergangenen 24 Stunden um 30 % aller berücksichtigten Daten handelt. Wenn Ihnen die Daten für eine Woche vorliegen, machen alle über die ersten sechs Tage gemittelten Daten 70 % der Daten und der letzte Tag 30 % aus.

In die Staging-Tabelle geschriebene Daten

Bei jeder Sitzung werden die folgenden Daten in eine Lern-Staging-Tabelle geschrieben:

- Angebotskontakt
- Angebotsannahme
- Lernattribute

Ein Aggregator liest die Daten in einem konfigurierbaren Intervall aus der Staging-Tabelle, kompiliert sie und schreibt sie in eine Tabelle. Das Lernmodul liest diese Aggregatdaten und verwendet sie in Berechnungen.

Aktivieren des Lernmoduls

Alle Laufzeitserver haben ein integriertes Lernmodul. Dieses Modul ist standardmäßig inaktiviert. Sie können das Lernmodul aktivieren, indem Sie eine Konfigurationseigenschaft ändern.

Vorgehensweise

Bearbeiten Sie in Marketing Platform für die Laufzeitumgebung folgende Konfigurationseigenschaften in der Kategorie `Interact > offerserving`.

Konfigurationseigenschaft	Einstellung
`optimizationType`	**BuiltInLearning**

Lernattribute

Das Lernmodul erlernt die Verwendung von Besucherattributen und Angebotsannahmedaten. Sie können auswählen, welche Besucherattribute überwacht werden sollen. Diese Besucherattribute können ein beliebiges Element in einem Kundenprofil sein, einschließlich eines Ereignisparameters, den Sie in Echtzeit erfassen.

Attribute aus dimensionalen Tabellen werden beim Lernen nicht unterstützt.

Auch wenn Sie eine beliebige Anzahl von Attributen zum Überwachen konfigurieren können, so empfiehlt IBM dennoch, dass Sie nicht mehr als zehn Lernattribute zwischen den statischen und dynamischen Lernattributen konfigurieren. Außerdem sollten Sie die folgenden Leitlinien berücksichtigen.

- Wählen Sie unabhängige Attribute aus.

  Wählen Sie keine ähnlichen Attribute aus. Wenn Sie beispielsweise ein Attribut namens HighValue erstellen und dieses Attribut durch eine Berechnung auf der Grundlage des Gehalts definiert wird, wählen Sie weder HighValue noch Gehalt aus. Ähnliche Attribute sind für den Lernalgorithmus hinderlich.
- Wählen Sie Attribute mit eigenständigen Werten aus.

  Wenn ein Attribut Wertspannen aufweist, müssen Sie einen exakten Wert auswählen. Möchten Sie beispielsweise das Gehalt als Attribut verwenden, sollten

Sie jeder Gehaltsspanne einen bestimmten Wert zuweisen: Die Spanne 20.000 – 30.000 sollte A sein; 30.000 – 40.000 sollte B sein usw.

- Begrenzen Sie die Anzahl der Attribute, um die Leistung nicht zu behindern.

  Die Anzahl der Attribute, die Sie verfolgen können, hängt von Ihren Leistungsanforderungen und Ihrer Interact-Installation ab. Sofern möglich verwenden Sie ein anderes Modellierungstool (wie z. B. PredictiveInsight), um die zehn wichtigsten prognostizierbaren Attribute zu bestimmen. Sie können das Lernmodul konfigurieren, um automatisch Attribute zu entfernen, die nicht prognostizierbar sind, aber auch Leistungsbeeinträchtigung bewirken.

Sie können die Leistung verwalten, indem Sie sowohl die Anzahl der zu überwachenden Attribute als auch die Anzahl der Werte pro zu überwachendem Attribut definieren. Mit der Eigenschaft `maxAttributeNames` wird die maximale Anzahl der zu verfolgenden Besucherattribute definiert. Mit der Eigenschaft `maxAttributeValues` wird die maximale Anzahl der pro Attribut zu verfolgenden Werte definiert. Alle anderen Werte werden einer Kategorie zugeordnet, die durch den Eigenschaftswert `otherAttributeValue` definiert wird. Das Lernmodul verfolgt jedoch nur die ersten gefundenen Werte. So möchten Sie beispielsweise das Benutzerattribut Augenfarbe verfolgen. Da Sie nur an den Werten Blau, Braun und Grün interessiert sind, legen Sie den Wert für `maxAttributeValues` auf 3 fest. Die ersten drei Besucher weisen jedoch die Werte Blau, Braun und Haselnussbraun auf. Dies bedeutet, dass allen Besuchern mit grünen Augen das Attribut `otherAttributeValue` zugeordnet wird.

Sie können auch dynamische Lernattribute verwenden, mit denen Sie Ihre Lernkriterien spezifischer definieren können. Dynamische Lernattribute ermöglichen es Ihnen, anhand der Kombination von zwei Attributen wie von einem einzigen Eintrag zu lernen. Betrachten Sie als Beispiel die folgenden Profilinformationen:

Besucher-ID	Kartentyp	Kartensaldo
1	Goldkarte	$1.000
2	Goldkarte	$9.000
3	Bronzekarte	$1.000
4	Bronzekarte	$9.000

Wenn Sie Standardlernattribute verwenden, können Sie nur jeweils anhand des Kartentyps und des Saldos lernen. Besucher 1 und 2 werden, basierend auf demselben Kartentyp, gemeinsam gruppiert, und Besucher 2 und 4 werden basierend auf dem Kartensaldo gruppiert. Dies ist möglicherweise kein korrekter Prädiktor von Angebotsannahmeverhalten. Wenn Goldkarteninhaber dazu tendieren, einen höheren Saldo zu haben, könnte sich das Verhalten von Besucher 2 radikal von Besucher 4 unterscheiden, was die Standardlernattribute verfälschen würde. Wenn Sie allerdings dynamische Lernattribute verwenden, wird individuell anhand der jeweiligen Besucher gelernt, was die Vorhersagen genauer macht.

Wenn Sie dynamische Lernattribute verwenden und der Besucher zwei gültige Werte für ein Attribut hat, wählt das Lernmodul den zuerst gefundenen Wert aus.

Wenn Sie die Eigenschaft `enablePruning` auf `yes` festlegen, ermittelt das Lernmodul algorithmisch, welche Attribute nicht prädiktiv sind, und hört auf, diese Attribute bei der Berechnung von Gewichtungen zu berücksichtigen. Wenn Sie beispielsweise ein Attribut verfolgen, das die Haarfarbe darstellt, und das Lernmodul bestimmt, dass es kein Muster für das Annehmen eines Angebots basierend auf der

Haarfarbe des Besuchers gibt, hört das Lernmodul auf, das Attribut Haarfarbe zu berücksichtigen. Attribute werden jedes Mal neu bewertet, wenn der Lernaggregationsprozess ausgeführt wird (durch die Eigenschaft `aggregateStatsIntervalInMinutes` definiert). Dynamische Lernattribute werden auch entfernt.

Definieren eines Lernattributs

Mit diesem Verfahren können Sie ein Lernattribut definieren.

Informationen zu diesem Vorgang

Sie können die Anzahl der Besucherattribute bis zum maximalen Wert `maxAttributeNames` konfigurieren.

`(learningAttributes)` ist eine Vorlage, um neue Lernattribute zu erstellen. Sie müssen für jedes Attribut einen neuen Namen eingeben. Es ist nicht möglich, zwei gleichnamige Kategorien zu erstellen.

Vorgehensweise

Bearbeiten Sie in Marketing Platform für die Designumgebung folgende Konfigurationseigenschaften in der Kategorie `Campaign > Partitionen > Partition`*`n`* `> Interact > Lernen`.

Konfigurationseigenschaft	Einstellung
`attributeName`	Der Wert für `attributeName` muss mit dem Namen eines Name/Wert-Paars in den Profildaten übereinstimmen. Bei diesem Namen braucht die Groß-/Kleinschreibung nicht berücksichtigt zu werden.

Definieren von dynamischen Lernattributen

Um dynamische Lernattribute zu definieren, müssen Sie die `UACI_AttributeList`-Tabelle in der Lerndatenquelle ausfüllen.

Alle Spalten in dieser Tabelle haben den Typ varchar(64).

Spalte	Beschreibung
`AttributeName`	Der Name des dynamischen Attributs, über das Sie etwas lernen möchten. Dieser Wert muss ein tatsächlich möglicher Wert in `AttributeNameCol` sein.
`AttributeNameCol`	Der vollständig qualifizierte Name der Spalte (hierarchische Struktur, mit der Profiltabelle beginnend), in der `AttributeName` gefunden werden kann. Dieser Spaltenname muss nicht notwendigerweise ein Standardlernattribut sein.
`AttributeValueCol`	Der vollständig qualifizierte Name der Spalte (hierarchische Struktur, mit der Profiltabelle beginnend), in der der zugehörige Wert für `AttributeName` gefunden werden kann.

Betrachten Sie zum Beispiel die folgende Profiltabelle und die zugehörige Dimensionstabelle.

Tabelle 6. MyProfileTable

VisitorID	KeyField
1	Key1
2	Key2
3	Key3
4	Key4

Tabelle 7. MyDimensionTable

KeyField	CardType	CardBalance
Key1	Gold Card	1000
Key2	Gold Card	9000
Key3	Bronze Card	1000
Key4	Bronze Card	9000

Das folgende Beispiel zeigt eine `UACI_AttributeList`-Tabelle zum Abgleich von Kartentyp und Saldo.

Tabelle 8. UACI_AttributeList

AttributeName	AttributeNameCol	AttributeValueCol
Gold Card	MyProfileTable.MyDimensionTable.CardType	MyProfileTable.MyDimensionTable.CardBalance
Bronze Card	MyProfileTable.MyDimensionTable.CardType	MyProfileTable.MyDimensionTable.CardBalance

Konfigurieren der Laufzeitumgebung für die Erkennung von externen Lernmodulen

Sie können die Lern-Java™-API verwenden, um Ihr eigenes Lernmodul zu schreiben. Sie müssen die Laufzeitumgebung konfigurieren, um Ihr Lerndienstprogramm in Marketing Platform zu erkennen.

Informationen zu diesem Vorgang

Sie müssen den Interact-Laufzeitserver erneut starten, damit diese Änderungen wirksam werden.

Vorgehensweise

1. Bearbeiten Sie in Marketing Platform für die Laufzeitumgebung die folgenden Konfigurationseigenschaften in der Kategorie `Interact > offerserving`. Die Konfigurationseigenschaften für die Lernoptimierungsprogramm-API befinden sich in der Kategorie `Interact > offerserving > External Learning Config`.

Konfigurationseigenschaft	Einstellung
`optimizationType`	**ExternalLearning**
`externalLearningClass`	Klassenname für das externe Lernen

Konfigurationseigenschaft	Einstellung
externalLearningClassPath	Der Pfad zu den Klassen- oder JAR-Dateien auf dem Laufzeitserver für das externe Lernen. Wenn Sie eine Servergruppe verwenden und alle Laufzeitserver dieselbe Instanz von Marketing Platform referenzieren, muss jeder Server über eine Kopie der Klassen- oder JAR-Dateien an derselben Position verfügen.

2. Starten Sie den Interact-Laufzeitserver erneut, damit diese Änderungen wirksam werden.

Kapitel 5. Informationen zur Interact-API

Interact stellt Angebote für eine große Vielfalt von Touchpoints bereit. Sie können beispielsweise die Laufzeitumgebung und Ihren Touchpoint konfigurieren, Nachrichten an Ihre Call-Center-Mitarbeiter zu senden, die sie über die besten Up-Selling- oder Cross-Selling-Möglichkeiten bei einem Kunden informieren, der mit einer bestimmten Art von Serviceanfrage anruft. Sie können auch die Laufzeitumgebung und Ihren Touchpoint konfigurieren, maßgeschneiderte Angebote einem Kunden (Besucher) bereitzustellen, der zu einem bestimmten Bereich Ihrer Website navigiert ist.

Die Interact-Anwendungsprogrammierschnittstelle (API) ermöglicht es Ihnen, Ihren Touchpoint und einen Laufzeitserver so zu konfigurieren, dass sie zusammenarbeiten, um die bestmöglichen Angebote bereitzustellen. Mithilfe der API kann der Touchpoint Informationen vom Laufzeitserver anfordern, um den Besucher einer Gruppe (Segment) zuzuweisen und um auf diesem Segment basierende Angebote bereitzustellen. Sie können auch Daten für eine spätere Analyse protokollieren, um Ihre Angebotspräsentationsstrategien zu optimieren.

Die Interact-API ermöglicht außerdem über JavaScript die Kommunikation zwischen dem Endbenutzerclient und dem Server.

Um Ihnen die größtmögliche Flexibilität bei der Integration von Interact in Ihre Umgebungen zu bieten, stellt IBM einen Webservice bereit, der mithilfe der Interact-API zugänglich ist.

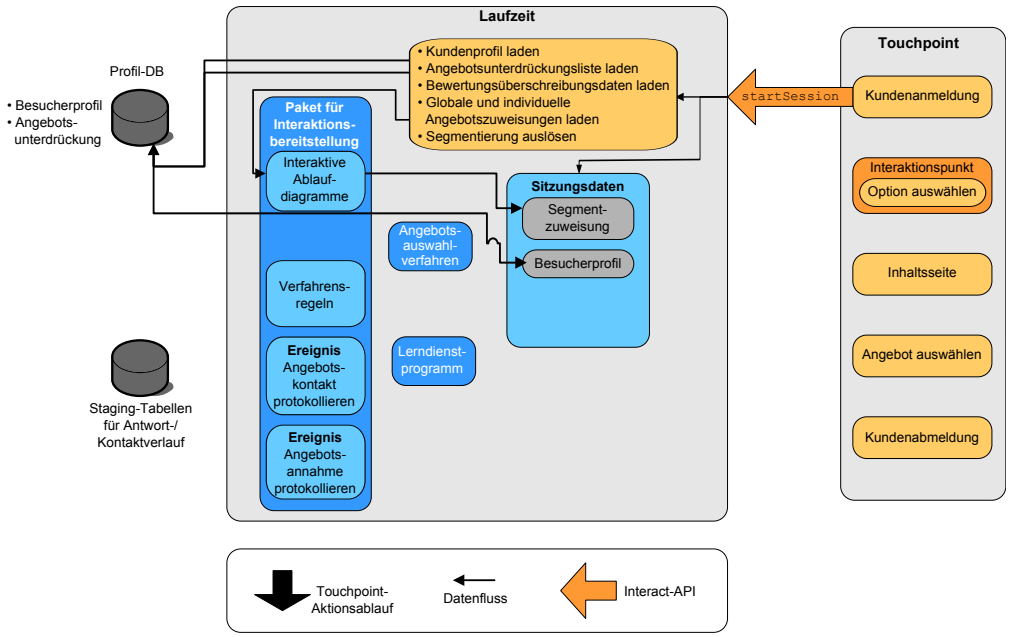
Interact-API-Datenfluss

Dieses Beispiel veranschaulicht, wie die API zwischen Ihrem Touchpoint und der Laufzeitumgebung funktioniert. Der Besucher führt nur vier Aktionen aus - Anmelden, zur Seite mit den Angeboten navigieren, ein Angebot auswählen und Abmelden. Sie können Ihre Integration so kompliziert wie nötig gestalten (innerhalb der Begrenzungen Ihrer Leistungsanforderungen).

Dieses Diagramm zeigt eine einfache Implementierung der Interact-API.

Ein Besucher meldet sich bei einer Website an und navigiert zu einer Seite, auf der Angebote angezeigt werden. Der Besucher wählt ein Angebot aus und meldet sich ab. Auch wenn die Interaktion einfach ist, so gibt es doch mehrere Ereignisse, die im Touchpoint und auf dem Laufzeitserver auftreten:

1. Sitzung starten
2. Zu einer Seite navigieren
3. Angebot auswählen
4. Sitzung schließen



Sitzung starten

Wenn sich ein Besucher anmeldet, löst dies die Methode `startSession` aus.

Die Methode `startSession` führt vier Schritte aus:

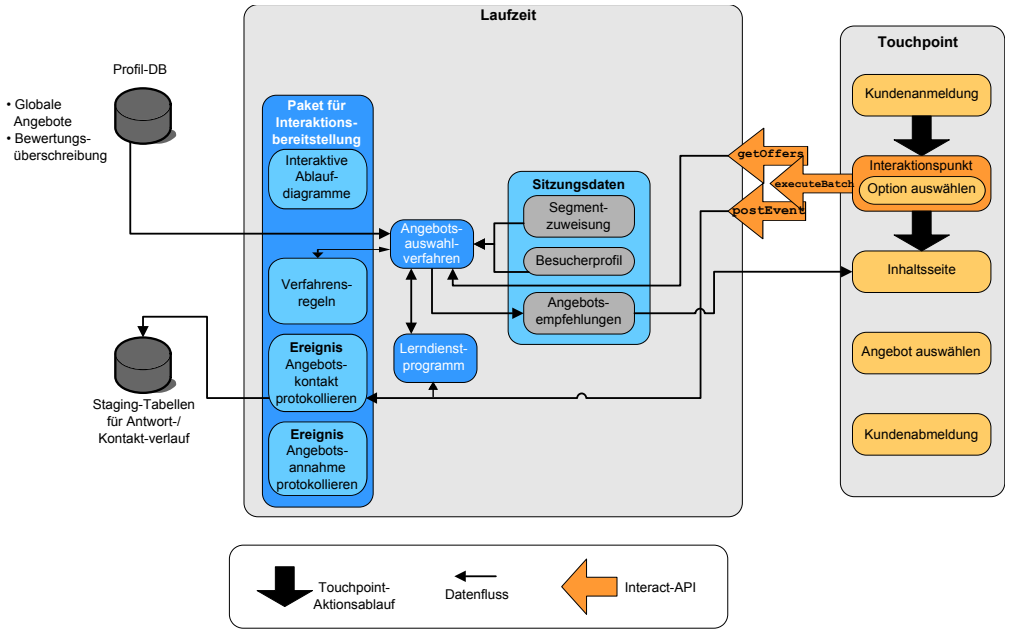
1. Sie erstellt eine neue Laufzeitsitzung.
2. Sie sendet eine Anforderung, die Kundenprofildaten in die Sitzung zu laden.
3. Sie sendet eine Anforderung, die Profildaten zu verwenden und ein interaktives Ablaufdiagramm zu starten, um den Kunden in Segmente zu platzieren. Dieses Ablaufdiagramm wird asynchron ausgeführt.
4. Der Laufzeitserver lädt etwaige Informationen zur Angebotsunterdrückung und globale bzw. individuelle Angebotsverfahrensinformationen in die Sitzung. Die Sitzungsdaten werden für die Dauer der Sitzung im Speicher gehalten.

Zu einer Seite navigieren

Der Besucher navigiert durch die Site, bis er einen vordefinierten Interaktionspunkt erreicht. In der Abbildung ist der zweite Interaktionspunkt (Auswahl der Auswahloption) eine Stelle, an der der Besucher auf einen Link klickt, der eine Gruppe von Angeboten darstellt. Der Touchpoint-Manager hat den Link so konfiguriert, dass die Methode `executeBatch` für die Auswahl eines Angebots ausgelöst wird.

Angebot auswählen

Dieses Diagramm stellt den API-Aufruf dar, der die Methode `executeBatch` auslöst.

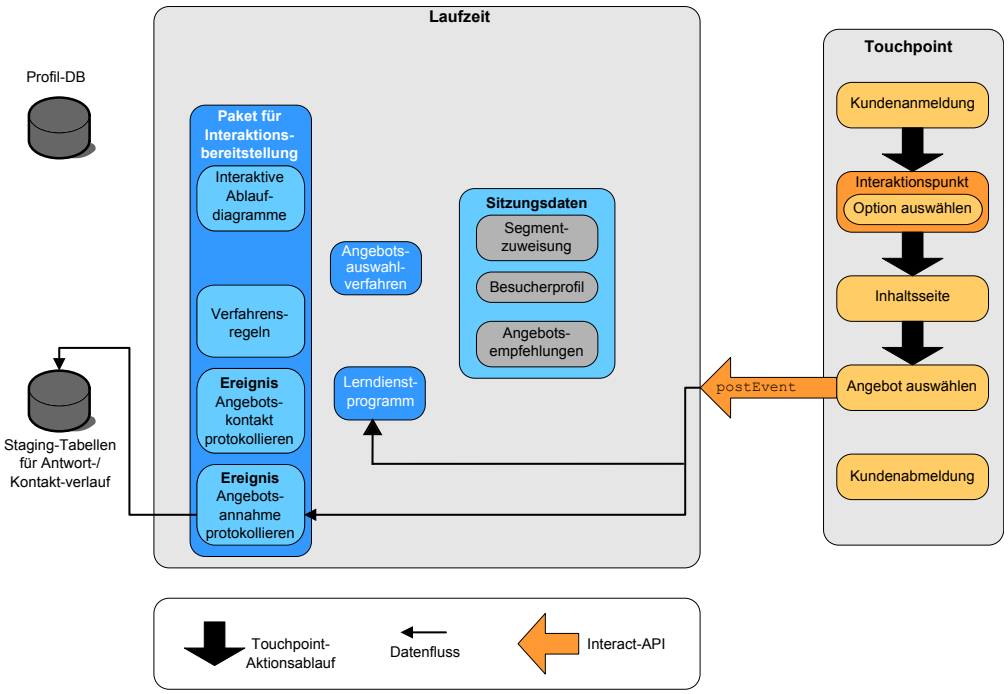


Die Methode `executeBatch` ermöglicht es Ihnen, mehr als eine Methode in einem einzelnen Aufruf an den Laufzeitserver aufzurufen. Diese bestimmte executeBatch-Methode ruft zwei andere Methoden auf, `getOffers` und `postEvent`. Die Methode `getOffers` fordert eine Liste von Angeboten an. Der Laufzeitserver verwendet die Segmentierungsdaten, die Angebotsunterdrückungsliste, die Verfahrensregeln und das Lernmodul, um einen Satz von Angeboten vorzuschlagen. Der Laufzeitserver gibt einen Satz von Angeboten zurück, der auf der Inhaltsseite angezeigt wird.

Die Methode `postEvent` löst eines der Ereignisse aus, die in der Designumgebung definiert wurden. In diesem bestimmten Fall sendet das Ereignis eine Anforderung, die angezeigten Angebote im Kontaktverlauf zu protokollieren.

Der Besucher wählt eines der Angebote aus (Angebot auswählen).

Dieses Diagramm stellt die Methode `postEvent` dar.

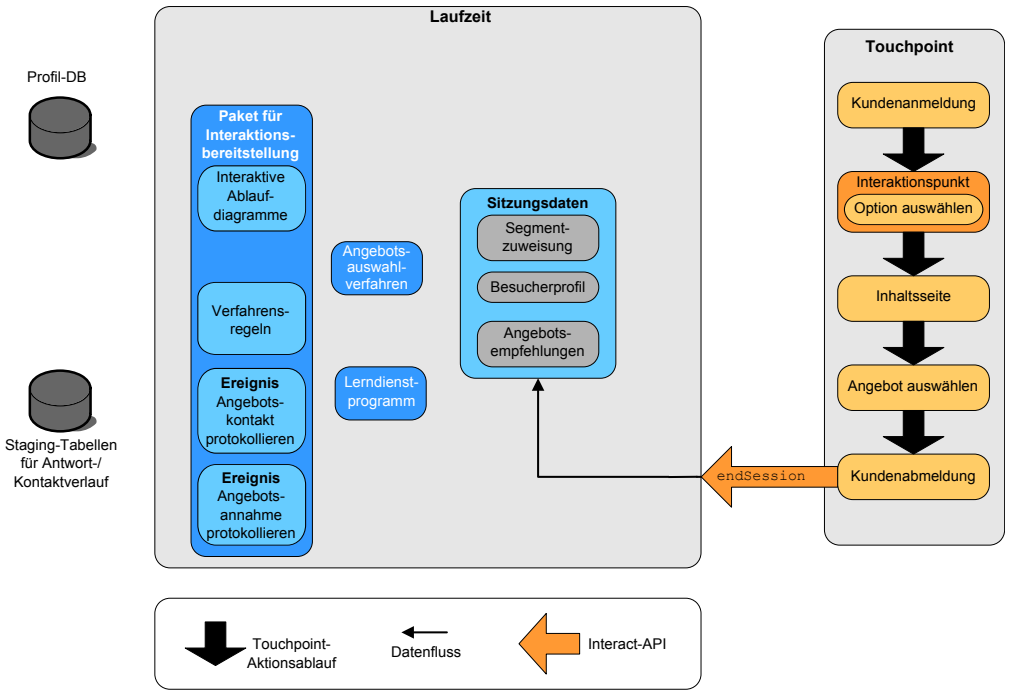


Das der Auswahl des Angebots zugeordnete Steuerelement der Benutzerschnittstelle ist so konfiguriert, dass eine weitere `postEvent`-Methode gesendet wird. Dieses Ereignis sendet eine Anforderung, die Angebotsannahme im Antwortverlauf zu protokollieren.

Sitzung schließen

Nachdem der Besucher das Angebot ausgewählt hat, muss er keine weiteren Aktionen auf der Website ausführen und kann sich abmelden. Der Abmeldebefehl ist mit der Methode `endSession` verknüpft.

Dieses Diagramm stellt die Methode `endSession` dar.



Die Methode `endSession` schließt die Sitzung. Wenn der Besucher vergisst sich abzumelden, gibt es ein konfigurierbares Sitzungszeitlimit um sicherzustellen, dass jede Sitzung einmal endet. Wenn Sie Daten bewahren wollen, die an die Sitzung übergeben werden, wie in Parametern aufgenommene Informationen in der Methode `startSession` oder `setAudience`, wenden Sie sich an die Person, die interaktive Ablaufdiagramme erstellt. Die Person, die ein interaktives Ablaufdiagramm erstellt, kann den Prozess "Momentaufnahme" verwenden, um diese Daten in eine Datenbank zu schreiben, bevor die Sitzung endet und diese Daten verloren gehen. Sie können dann die Methode `postEvent` verwenden, um das interaktive Ablaufdiagramm aufzurufen, das den Prozess "Momentaufnahme" enthält.

Einfaches Beispiel für Interaktionsplanung

In diesem Beispiel entwerfen Sie eine Interaktion für die Website einer Mobiltelefonfirma. Sie erstellen drei unterschiedliche Angebote, richten eine Protokollierung für die Angebote ein, weisen dem Angebot Verfahrenscodes zu und zeigen mehrere Bilder an, die mit den Angeboten verknüpft werden.

Entwurfsprozess

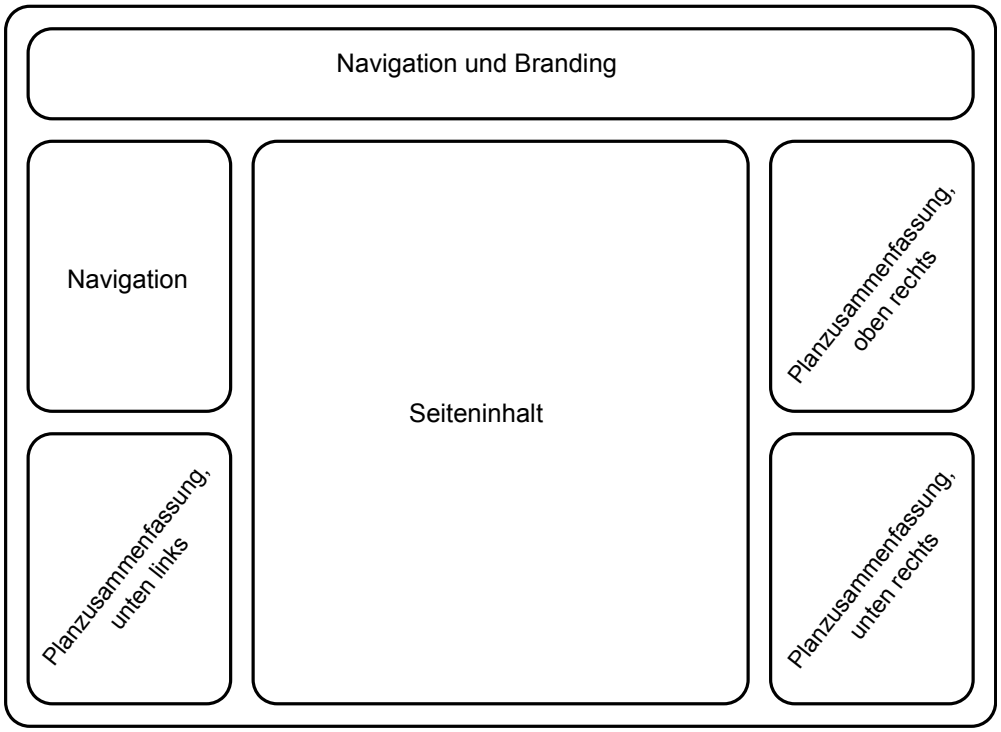
Im Rahmen des Entwurfs einer Interaktion für diesen Kunden können Sie folgende Aktionen ausführen:

1. Identifizieren Sie die Anforderungen für die Übersichtsseite des Kunden.
2. Erstellen Sie Interaktionspunkte für die Angebotsanforderungen.
3. Konfigurieren Sie eine Protokollierung für die Angebote.
4. Erstellen Sie Verfahrenscodes.
5. Verknüpfen Sie eine Reihe von wechselnden Bildern mit den Angeboten.

Dies ist ein grundlegendes Beispiel und zeigt nicht die beste Art, die Integration zu schreiben. Beispielsweise umfasst keines dieser Beispiele eine Fehlerprüfung mithilfe der Antwortklasse.

Identifizieren der Anforderungen für die Übersichtsseite des Mobilfunkvertrags

Das folgende Diagramm zeigt das Layout für die Übersichtsseite des Mobilfunkvertrags.



Sie definieren die folgenden Elemente, damit die Anforderungen der Übersichtsseite des Mobilfunkvertrags erfüllt werden:

Anforderung	Implementierung
Ein Angebot, das in einer Zone für Upgradeangebote angezeigt wird Der Bereich auf der Seite, der das Upgradeangebot anzeigt, muss definiert werden. Auch müssen, nachdem Interact ein Angebot zur Anzeige ausgewählt hat, die Informationen protokolliert werden.	• Interaktionspunkt: `ip_planSummaryBottomRight` • Ereignis: `evt_logOffer`
Zwei Angebote für Telefonupgrades Jeder Bereich auf der Seite, der die Telefonupgrades anzeigt, muss definiert werden.	• Interaktionspunkt: `ip_planSummaryTopRight` • Interaktionspunkt: `ip_planSummaryBottomLeft`
Für Analysezwecke müssen Sie protokollieren, welche Angebote angenommen und welche abgelehnt werden.	• Ereignis: `evt_offerAccept` • Ereignis: `evt_offerReject`
Sie wissen auch, dass Sie den Verfahrenscode eines Angebots übergeben müssen, wenn Sie einen Angebotskontakt, ein Annehmen oder eine Ablehnung protokollieren.	`NameValuePair`

Anforderung	Implementierung
Zeigen Sie drei wechselnde Bilder auf der Seite an. Verknüpfen Sie die Bilder mit den Angeboten.	

Erstellen von Interaktionspunkten

Nun können Sie den Designumgebungsbenutzer bitten, für Sie Interaktionspunkte und Ereignisse zu erstellen, während Sie beginnen, die Integration mit Ihrem Touchpoint zu codieren.

Für jeden Interaktionspunkt, der ein Angebot anzeigen wird, müssen Sie zuerst ein Angebot abrufen, um dann die Informationen zu extrahieren, die für die Anzeige des Angebots erforderlich sind. Beispiel: Fordern Sie ein Angebot für den rechten unteren Bereich Ihrer Webseite (`planSummaryBottomRight`) an.

```
Response response=getOffers(sessionID, ip_planSummaryBottomRight, 1)
```

Dieser Antwortaufruf gibt ein Antwortobjekt zurück, das eine Antwort des Typs `OfferList` enthält. Ihre Webseite kann jedoch kein Objekt des Typs `OfferList` verwenden. Sie brauchen eine Bilddatei für das Angebot, die bekanntlich eines der Angebotsattribute (`offerImg`) ist. Sie müssen das benötigte Angebotsattribut aus `OfferList` extrahieren.

```
OfferList offerList=response.getOfferList();
if(offerList.getRecommendedOffers() != null)
{
 Offer offer = offerList.getRecommendedOffers()[0];
 NameValuePair[] attributes = offer.getAdditionalAttributes();
 for(NameValuePair attribute: attributes)
 {
  if(attribute.getName().equalsIgnoreCase("offerImg"))
  {
   /* Use this value in your code for the page, for
   example: stringHtml = "<img src="+attribute.getValueAsString()+ " >   */
  }
 }
}
```

Konfigurieren der Protokollierung

Jetzt, da Sie das Angebot anzeigen, möchten Sie es als einen Kontakt protokollieren.

```
NameValuePair evtParam_TreatmentCode = new NameValuePairImpl();
evtParam_TreatmentCode.setName("UACIOfferTrackingCode");
evtParam_TreatmentCode.setValueAsString(offer.getTreatmentCode());
evtParam_TreatmentCode.setValueDataType(NameValuePair.DATA_TYPE_STRING);
postEvent(sessionID, evt_logOffer, evtParam_TreatmentCode)
```

Anstatt jede dieser Methoden einzeln aufzurufen, können Sie die Methode `executeBatch` verwenden, wie im folgenden Beispiel für den Abschnitt `planSummaryBottomLeft` der Webseite gezeigt.

```
Command getOffersCommand = new CommandImpl();
getOffersCommand.setMethodIdentifier(Command.COMMAND_GETOFFERS);
getOffersCommand.setInteractionPoint(ip_planSummaryBottomLeft);
getOffersCommand.setNumberRequested(1);

Command postEventCommand = new CommandImpl();
postEventCommand.setMethodIdentifier(Command.COMMAND_POSTEVENT);
postEventCommand.setEvent(evt_logOffer);
```

```
/** Build command array */
Command[] commands =
{
 getOffersCommand,
 postEventCommand
};

/** Make the call */
BatchResponse batchResponse = api.executeBatch(sessionId, commands);
```

Sie müssen in diesem Beispiel UACIOfferTrackingCode nicht definieren. Der Interact-Laufzeitserver protokolliert die letzte empfohlene Liste von Verfahren automatisch als Kontakte, wenn Sie UACIOfferTrackingCode nicht bereitstellen.

Erstellen von Verfahrenscodes

Wo notwendig erstellen Sie ein NameValuePair, um den Verfahrenscode aufzunehmen, wie im folgenden Beispiel gezeigt.

```
NameValuePair evtParam_TreatmentCode = new NameValuePairImpl();
evtParam_TreatmentCode.setName("UACIOfferTrackingCode");
evtParam_TreatmentCode.setValueAsString(offer.getTreatmentCode());
evtParam_TreatmentCode.setValueDataType(NameValuePair.DATA_TYPE_STRING);
```

Verknüpfen von Bildern mit Angeboten

Für den zweiten Bereich auf der Seite, der ein Telefonupgrade anzeigt, haben Sie eine Anweisung geschrieben, durch die das angezeigte Bild alle 30 Sekunden wechselt. Da Sie beschließen, zwischen drei Bildern zu wechseln, verwenden Sie folgenden Text, um den Satz von Angeboten abzurufen und ihn in den Cache zu stellen, damit er in Ihrem Code für den Bildwechsel verwendet werden kann.

```
Response response=getOffers(sessionID, ip_planSummaryBottomLeft, 3)
OfferList offerList=response.getOfferList();
if(offerList.getRecommendedOffers() != null)
{
 for(int x=0;x<3;x++)
 {
  Offer offer = offerList.getRecommendedOffers()[x];
  if(x==0)
  {
   // grab offerimg attribute value and store somewhere;
   // this will be the first image to display
  }
  else if(x==1)
  {
   // grab offerimg attribute value and store somewhere;
   // this will be the second image to display
  }
  else if(x==2)
  {
   // grab offerimg attribute value and store somewhere;
   // this will be the third image to display
  }
 }
}
```

Sie müssen Ihren Kundencode (Client-Code) zum Abrufen aus dem lokalen Cache und zum Protokollieren als Kontakt nur einmal für jedes Angebot schreiben, nachdem das zugehörige Bild angezeigt wird. Um den Kontakt zu protokollieren, muss der Parameter UACITrackingCode wie vorhin bereitgestellt werden. Jedes Angebot hat einen anderen Verfolgungscode.

```
NameValuePair evtParam_TreatmentCodeSTR = new NameValuePairImpl();
NameValuePair evtParam_TreatmentCodeSBR = new NameValuePairImpl();
NameValuePair evtParam_TreatmentCodeSBL = new NameValuePairImpl();

OfferList offerList=response.getOfferList();
if(offerList.getRecommendedOffers() != null)
 {
 for(int x=0;x<3;x++)
  {
  Offer offer = offerList.getRecommendedOffers()[x];
  if(x==0)
  {
   evtParam_TreatmentCodeSTR.setName("UACIOfferTrackingCode");
   evtParam_TreatmentCodeSTR.setValueAsString(offer.getTreatmentCode());
   evtParam_TreatmentCodeSTR.setValueDataType(NameValuePair.DATA_TYPE_STRING);
  }
  else if(x==1)
  {
   evtParam_TreatmentCodeSBR.setName("UACIOfferTrackingCode");
   evtParam_TreatmentCodeSBR.setValueAsString(offer.getTreatmentCode());
   evtParam_TreatmentCodeSBR.setValueDataType(NameValuePair.DATA_TYPE_STRING);
  }
  else if(x==2)
  {
   evtParam_TreatmentCodeSBL.setName("UACIOfferTrackingCode");
   evtParam_TreatmentCodeSBL.setValueAsString(offer.getTreatmentCode());
   evtParam_TreatmentCodeSBL.setValueDataType(NameValuePair.DATA_TYPE_STRING);
  }
 }
}
```

Bei jedem Angebot protokollieren Sie, wenn darauf geklickt wird, das angenommene Angebot und die abgelehnten Angebote. (In diesem Szenario werden Angebote, die nicht explizit ausgewählt werden, als abgelehnt bewertet.) Das Folgende ist ein Beispiel, wenn das Angebot `ip_planSummaryTopRight` ausgewählt wird:

```
postEvent(sessionID, evt_offerAccept, evtParam_TreatmentCodeSTR)
postEvent(sessionID, evt_offerReject, evtParam_TreatmentCodeSBR)
postEvent(sessionID, evt_offerReject, evtParam_TreatmentCodeSBL)
```

In der Praxis wäre es das Beste, diese drei `postEvent`-Aufrufe mit der Methode `executeBatch` zu senden.

Entwerfen der Interact-API-Integration

Der Aufbau Ihrer Interact-API-Integration mit Ihrem Touchpoint erfordert einige Arbeitsschritte, bevor Sie mit der Implementierung beginnen können. Sie müssen mit Ihrem Marketingteam zusammenarbeiten um zu entscheiden, wo in Ihrem Touchpoint die Laufzeitumgebung Angebote bereitstellen soll (Definition der Interaktionspunkte) und welche Art von Protokollierung oder interaktive Funktionalität Sie verwenden wollen (Definition Ihrer Ereignisse).

In der Entwurfsphase können dies einfache Grundelemente sein. Bei einer Telekommunikationswebsite beispielsweise sollte die Planübersichtsseite des Kunden ein Angebot in Bezug auf ein Planupgrade und zwei Angebot für Telefonupgrades anzeigen.

Nachdem Ihr Unternehmen entschieden hat, wann und wie es mit Kunden interagieren möchte, müssen Sie Interact verwenden, um die Details zu definieren. Ein Ablaufdiagrammverfasser muss die interaktiven Ablaufdiagramme entwerfen, die verwendet werden, wenn Neusegmentierungsereignisse auftreten. Sie müssen die Anzahl und die Namen der Interaktionspunkte und Ereignisse festlegen sowie ent-

scheiden, welche Daten für ordnungsgemäße Segmentierung, Ereignisbereitstellung und Angebotsabruf übergeben werden sollen. Der Designumgebungsbenutzer definiert die Interaktionspunkte und Ereignisse für den interaktiven Kanal. Sie verwenden dann diese Namen, wenn Sie die Integration mit Ihrem Touchpoint in der Laufzeitumgebung codieren. Sie sollten auch festlegen, welche Metrikinformationen erforderlich sind um zu bestimmen, wann Sie Angebotskontakte und Antworten protokollieren müssen.

Zu berücksichtigende Punkte

Berücksichtigen Sie beim Entwerfen einer Interaktion die Auswirkungen, die nicht auswählbare Angebote, ein nicht erreichbarer Laufzeitserver und das Prozesstiming auf die Interaktion haben. Seien Sie bei der Definition von Angebotsablehnungen möglichst präzise. Berücksichtigen Sie die optionalen Produktfunktionen, die die Interaktion verbessern können.

Führen Sie beim Entwerfen Ihrer Interaktion folgende Schritte aus:

Erstellen Sie einen Standardfüllinhalt
Erstellen Sie einen Standardfüllinhalt (üblicherweise eine freundliche Brandingnachricht oder einen leeren Inhalt) für jeden Interaktionspunkt, an dem Angebote angezeigt werden können. Dieser Füllinhalt wird verwendet, wenn keine Angebote auswählbar sind, die dem aktuellen Besucher in der aktuellen Situation bereitgestellt werden können. Sie weisen diesen Standardfüllinhalt als Standardzeichenfolge für den Interaktionspunkt zu.

Schließen Sie ein Alternativverfahren für die Darstellung von Inhalten ein
Nehmen Sie eine Methode zur Darstellung von Inhalt auf, falls Ihr Touchpoint die Laufzeitservergruppe aus irgendwelchen Gründen nicht erreichen kann.

Berücksichtigen Sie die Dauer einer Ablaufdiagrammausführung
Beim Auslösen von Ereignissen, die Ihren Besucher neu segmentieren (einschließlich `postEvent` und `setAudience`), müssen Sie beachten, dass das Ausführen von Ablaufdiagrammen etwas Zeit beansprucht. Die Methode `getOffers` wartet mit der Ausführung, bis die Segmentierung abgeschlossen ist. Erst dann wird die Methode `getOffers` ausgeführt. Eine allzu häufige Neusegmentierung könnte die Antwortleistung des Aufrufs `getOffers` beeinträchtigen.

Legen Sie fest, was genau eine "Angebotsablehnung" bedeutet
Mehrere Berichte, wie z. B. der Bericht "Übersicht über Kanäle zum Angebotserfolg", geben die Häufigkeit an, mit der ein Angebot abgelehnt wurde. Dieser Bericht gibt Aufschluss darüber, wie oft die Aktion "Angebotsablehnung protokollieren" von `postEvent` ausgelöst wurde. Sie müssen festlegen, ob die Aktion "Angebotsablehnung protokollieren" für eine tatsächliche Ablehnung gilt. Dies wäre der Fall, wenn beispielsweise auf einen Link mit der Beschriftung **Nein danke** geklickt wurde. Sie kann auch für ein Angebot gelten, das ignoriert wird, z. B. bei einer Seite mit drei Bannerwerbungen, von denen keine ausgewählt wird.

Entscheiden Sie, welche Angebotsauswahlfunktionen verwendet werden sollen
Es gibt mehrere optionale Funktionen, mit denen Sie die Interact-Angebotsauswahl verbessern können. Unter anderem sind folgende Funktionen verfügbar:

- Lernfunktion
- Angebotsunterdrückung
- Individuelle Angebotszuweisungen

- Sonstige Elemente der Angebotsbereitstellung

Sie müssen festlegen, ob eine dieser optionalen Funktionen Ihre Interaktionen bereichern würden.

Kapitel 6. Verwalten der IBM Interact-API

Immer wenn Sie die Methode `startSession` verwenden, erstellen Sie eine Interact-Laufzeitsitzung auf dem Laufzeitserver. Sie können Konfigurationseigenschaften verwenden, um die Sitzungen auf dem Laufzeitserver zu verwalten.

Sie müssen diese Einstellungen möglicherweise konfigurieren, wenn Sie Ihre Interact-Integration mit Ihrem Touchpoint implementieren.

Diese Konfigurationseigenschaften befinden sich in der Kategorie `sessionManagement`.

Ländereinstellungen und die Interact-API

Sie können Interact für nicht-englische Touchpoints verwenden. Der Touchpoint und alle Zeichenfolgen in der API verwenden die Ländereinstellung, die für den Laufzeitumgebungsbenutzer definiert ist.

Sie können nur eine Ländereinstellung pro Servergruppe auswählen.

Beispiel: In der Laufzeitumgebung erstellen Sie zwei Benutzer, asm_admin_en mit der Benutzerländereinstellung Englisch und asm_admin_fr mit der Benutzerländereinstellung Französisch. Wenn Ihr Touchpoint für französisch Sprechende entwickelt wurde, legen Sie die Eigenschaft `asmUserForDefaultLocale` für die Laufzeitumgebung als asm_admin_fr fest.

Informationen zur JMX-Überwachung

Interact stellt den JMX-Überwachungsservice (Java Management Extensions) bereit, auf den Sie mit einer JMX-Überwachungsanwendung zugreifen können. Diese JMX-Überwachung ermöglicht es Ihnen, Ihre Laufzeitserver zu überwachen und zu verwalten.

Die JMX-Attribute stellen eine Menge von Detailinformationen über den Laufzeitserver zur Verfügung. Das JMX-Attribut `ErrorCount` beispielsweise gibt Anzahl von Fehlernachrichten an, die seit der letzten Zurücksetzung oder dem Systemstart protokolliert wurden. Sie können mithilfe dieser Informationen feststellen, wie häufig es Fehler in Ihrem System gibt. Wenn Sie Ihre Website so codiert haben, nur ein End Session aufzurufen, wenn jemand eine Transaktion abschließt, können Sie auch `startSessionCount` mit `endSessionCount` vergleichen um herauszufinden, wie viele Transaktionen unvollständig sind.

Interact unterstützt die RMI- und JMXMP-Protokolle, wie durch JSR 160 definiert. Sie können mit dem JMX-Überwachungsservice über jeden JSR160-kompatiblen JMX-Client verbinden.

Interaktive Ablaufdiagramme können nur mit der JMX-Überwachung überwacht werden. Informationen über interaktive Ablaufdiagramme erscheinen nicht in der Campaign-Überwachung.

Anmerkung: Wenn Sie IBM WebSphere mit einem Knotenmanager verwenden, müssen Sie das generische JVM-Argument definieren, um die JMX-Überwachung zu aktivieren.

Konfigurieren von Interact zur Verwendung der JMX-Überwachung mit dem RMI-Protokoll

Verwenden Sie dieses Verfahren, um Interact zur Verwendung der JMX-Überwachung mit dem RMI-Protokoll zu konfigurieren.

Informationen zu diesem Vorgang

Die Standardadresse für die Überwachung mit dem RMI-Protokoll lautet: `service:jmx:rmi:///jndi/rmi://RuntimeServer:port/interact`.

Vorgehensweise

Bearbeiten Sie in Marketing Platform für die Laufzeitumgebung folgende Konfigurationseigenschaften in der Kategorie `Interact > Überwachung`.

Konfigurationseigenschaft	Einstellung
`protocol`	**RMI**
`port`	Die Portnummer für den JMX-Service
`enableSecurity`	**False** Die Interact-Implementierung des RMI-Protokolls unterstützt keine Sicherheit.

Konfigurieren von Interact zur Verwendung der JMX-Überwachung mit dem JMXMP-Protokoll

Verwenden Sie dieses Verfahren, um Interact zur Verwendung der JMX-Überwachung mit dem JMXMP-Protokoll zu konfigurieren.

Vorbereitende Schritte

Das JMXMP-Protokoll erfordert zwei zusätzliche Bibliotheken in der folgenden Reihenfolge im Klassenpfad: `InteractJMX.jar` und `jmxremote_optional.jar`. Beide Dateien befinden sich im `lib`-Verzeichnis Ihrer Laufzeitumgebungsinstallation.

Informationen zu diesem Vorgang

Wenn Sie Sicherheit aktivieren, muss der Benutzername und das Kennwort mit einem Benutzer in Marketing Platform für die Laufzeitumgebung übereinstimmen. Sie können kein leeres Kennwort verwenden.

Die Standardadresse für die Überwachung mit dem JMXMP-Protokoll lautet: `service:jmx:jmxmp://RuntimeServer:port`.

Vorgehensweise

1. Überprüfen Sie, ob die Bibliotheken `InteractJMX.jar` und `jmxremote_optional.jar` in der richtigen Reihenfolge im Klassenpfad enthalten sind. Falls sie sich nicht im Klassenpfad befinden, fügen Sie diese dort hinzu.
2. Bearbeiten Sie in Marketing Platform für die Laufzeitumgebung folgende Konfigurationseigenschaften in der Kategorie `Interact > Überwachung`.

Konfigurationseigenschaft	Einstellung
`protocol`	**JMXMP**

Konfigurationseigenschaft	Einstellung
port	Die Portnummer für den JMX-Service
enableSecurity	**False**, um die Sicherheit zu inaktivieren, oder **True**, um die Sicherheit zu aktivieren

Konfigurieren von Interact für die Verwendung der jconsole-Scripts zur JMX-Überwachung

Wenn Sie keine separate JMX-Überwachungsanwendung haben, können Sie die mit der JVM installierte jconsole verwenden. Sie können die jconsole mithilfe der Startscripts im Verzeichnis Interact/tools starten.

Informationen zu diesem Vorgang

Das jconsole-Script verwendet standardmäßig das JMXMP-Protokoll für die Überwachung. Die Standardeinstellungen für jconsole.bat lauten wie folgt:

JMXMP-Verbindung

```
%JAVA_HOME%\bin\jconsole.exe -J-Djava.class.path=%JAVA_HOME%
  \lib\jconsole.jar;INTERACT_LIB%\interactJMX.jar; INTERACT_LIB%
  \jmxremote_optional.jar service:jmx:jmxmp://%HOST%:%PORT%
```

RMI-Verbindung

```
%JAVA_HOME%\bin\jconsole.exe -J-Djava.class.path=%JAVA_HOME%
      \lib\jconsole.jar;INTERACT_LIB%\jmxremote_optional.jar
     service:jmx:rmi:///jndi/rmi://%HOST%:%PORT%/interact
```

Vorgehensweise

1. Öffnen Sie Interact\tools\jconsole.bat (Windows) oder Interact/tools/jconsole.sh (UNIX) in einem Texteditor.
2. Legen Sie für INTERACT_LIB den vollständigen Pfad zum Verzeichnis *InteractInstallationDirectory*/lib fest.
3. Legen Sie für HOST den Hostnamen des Laufzeitservers fest, den Sie überwachen möchten.
4. Legen Sie für PORT den Port fest, den Sie für die JMX-Überwachung mit der Eigenschaft Interact > Überwachung > Port konfiguriert haben.
5. Optional: Wenn Sie das RMI-Protokoll für die Überwachung verwenden, setzen Sie ein Kommentarzeichen vor die JMXMP-Verbindung und entfernen Sie das Kommentarzeichen vor der RMI-Verbindung.

JMX-Attribute

Für die JMX-Überwachung sind mehrere Attribute verfügbar. Zu den Attributen der Designumgebung zählt die Überwachung des Kontakt-/Antwortverlaufs im Zusammenhang mit dem ETL-Prozess. Zu den Attributen der Laufzeitumgebung zählen Ausnahmebedingungen, verschiedene Ablaufdiagrammattribute, Ländereinstellung, Protokollprozess und Statistikdaten zum Thread-Pool. Darüber hinaus sind einige Attribute für Servicestatistiken verfügbar. Alle von der JMX-Überwachung bereitgestellten Daten gelten ab der letzten Zurücksetzung oder ab dem Systemstart. Eine Zählung beispielsweise gilt für die Anzahl der Elemente seit der letzten Zurücksetzung oder dem Systemstart, und nicht seit der Installation.

Attribute des ETL-Monitors für den Kontakt-/Antwortverlauf

Die Attribute des ETL-Monitors für den Kontakt-/Antwortverlauf sind Teil der Designumgebung. Alle folgenden Attribute sind Teil der Laufzeitumgebung.

Tabelle 9. ETL-Monitor für den Kontakt-/Antwortverlauf

Attribut	Beschreibung
AvgCHExecutionTime	Die durchschnittliche Zahl von Millisekunden, die das Kontakt- und Antwortverlaufsmodul benötigt, um in die Kontaktverlaufstabelle zu schreiben. Dieser Durchschnitt wird nur für Operationen berechnet, die erfolgreich waren und für die zumindest ein Datensatz in die Kontaktverlaufstabelle geschrieben wurde.
AvgETLExecutionTime	Die durchschnittliche Zahl von Millisekunden, die das Kontakt- und Antwortverlaufsmodul benötigt, um Daten aus der Laufzeitumgebung zu lesen. Der Durchschnitt umfasst die Zeit für sowohl erfolgreiche als auch fehlgeschlagene Operationen.
AvgRHExecutionTime	Die durchschnittliche Zahl von Millisekunden, die das Kontakt- und Antwortverlaufsmodul benötigt, um in die Antwortverlaufstabelle zu schreiben. Dieser Durchschnitt wird nur für Operationen berechnet, die erfolgreich waren und für die zumindest ein Datensatz in die Antwortverlaufstabelle geschrieben wurde.
ErrorCount	Die Anzahl von Fehlernachrichten, die seit der letzten Zurücksetzung oder dem Systemstart protokolliert wurden, falls vorhanden.
HighWaterMarkCHExecutionTime	Die maximale Zahl von Millisekunden, die das Kontakt- und Antwortverlaufsmodul benötigte, um in die Kontaktverlaufstabelle zu schreiben. Dieser Wert wird nur für Operationen berechnet, die erfolgreich waren und für die zumindest ein Datensatz in die Kontaktverlaufstabelle geschrieben wurde.
HighWaterMarkETLExecutionTime	Die maximale Zahl von Millisekunden, die das Kontakt- und Antwortverlaufsmodul benötigte, um Daten aus der Laufzeitumgebung zu lesen. Die Berechnung umfasst sowohl erfolgreiche als auch fehlgeschlagene Operationen.

Tabelle 9. ETL-Monitor für den Kontakt-/Antwortverlauf (Forts.)

Attribut	Beschreibung
HighWaterMarkRHExecutionTime	Die maximale Zahl von Millisekunden, die das Kontakt- und Antwortverlaufsmodul benötigte, um in die Antwortverlaufstabelle zu schreiben. Dieser Wert wird nur für Operationen berechnet, die erfolgreich waren und für die zumindest ein Datensatz in die Antwortverlaufstabelle geschrieben wurde.
LastExecutionDuration	Die Anzahl von Millisekunden, die das Kontakt- und Antwortverlaufsmodul benötigte, um den letzten Kopiervorgang durchzuführen.
NumberOfExecutions	Die Häufigkeit, mit der das Kontakt- und Antwortverlaufsmodul seit der Initialisierung ausgeführt wurde.
LastExecutionStart	Die Uhrzeit, zu der die letzte Ausführung des Kontakt- und Antwortverlaufsmoduls gestartet wurde.
LastExecutionSuccessful	Wenn "true": Die letzte Ausführung des Kontakt- und Antwortverlaufsmoduls war erfolgreich. Bei "false" ist ein Fehler aufgetreten.
NumberOfContactHistoryRecordsMarked	Die Anzahl von Kontaktverlaufsdatensätzen in der Tabelle `UACI_CHStaging`, die während der letzten Ausführung des Kontakt- und Antwortverlaufsmoduls verschoben wurden. Dieser Wert ist nur größer als null, wenn das Kontakt- und Antwortverlaufsmodul aktiv ist.
NumberOfResponseHistoryRecordsMarked	Die Anzahl von Antwortverlaufsdatensätzen in der Tabelle `UACI_RHStaging`, die während der letzten Ausführung des Kontakt- und Antwortverlaufsmoduls verschoben wurden. Dieser Wert ist nur größer als null, wenn das Kontakt- und Antwortverlaufsmodul aktiv ist.

Ausnahmebedingungsattribute

Die Ausnahmebedingungsattribute sind Bestandteil der Laufzeitumgebung.

Tabelle 10. Ausnahmebedingungen

Attribut	Beschreibung
errorCount	Die Anzahl von Fehlernachrichten, die seit der letzten Zurücksetzung oder dem Systemstart protokolliert wurden.

Tabelle 10. Ausnahmebedingungen (Forts.)

Attribut	Beschreibung
warningCount	Die Anzahl von Warnhinweisen, die seit der letzten Zurücksetzung oder dem Systemstart protokolliert wurden.

Attribute für Ablaufdiagramm-Engine-Statistikdaten

Die Attribute für Ablaufdiagramm-Engine-Statistikdaten sind Bestandteil der Laufzeitumgebung.

Tabelle 11. Ablaufdiagramm-Engine-Statistikdaten

Attribut	Beschreibung
activeProcessBoxThreads	Aktive Zählung von Ablaufdiagrammverarbeitungsthreads (von allen Ausführungen gemeinsam genutzt), die derzeit aktiv sind.
activeSchedulerThreads	Aktive Zählung von Ablaufdiagramm-Scheduler-Threads, die derzeit aktiv sind.
avgExecutionTimeMillis	Durchschnittliche Ablaufdiagrammlaufzeit in Millisekunden.
CurrentJobsInProcessBoxQueue	Die Anzahl von Jobs, die auf die Ausführung durch Ablaufdiagrammverarbeitungsthreads warten.
CurrentJobsInSchedulerQueue	Die Anzahl von Jobs, die auf die Ausführung durch Ablaufdiagramm-Scheduler-Threads warten.
maximumProcessBoxThreads	Maximale Anzahl von Ablaufdiagrammverarbeitungsthreads (von allen Ausführungen gemeinsam genutzt), die ausgeführt werden können.
maximumSchedulerThreads	Maximale Anzahl von Ablaufdiagramm-Scheduler-Threads (ein Thread pro Ausführung), die ausgeführt werden können.
numExecutionsCompleted	Die Gesamtzahl der Ablaufdiagrammausführungen, die abgeschlossen wurden.
numExecutionsStarted	Die Gesamtzahl der Ablaufdiagrammausführungen, die gestartet wurden.

Attribute für spezielle Ablaufdiagramme nach interaktivem Kanal

Die Attribute für spezielle Ablaufdiagramme nach interaktivem Kanal sind Bestandteil der Laufzeitumgebung.

Tabelle 12. Spezielle Ablaufdiagramme nach interaktivem Kanal

Attribut	Beschreibung
AvgExecutionTimeMillis	Durchschnittliche Laufzeit in Millisekunden für dieses Ablaufdiagramm in diesem interaktiven Kanal.
HighWaterMarkForExecutionTime	Maximale Laufzeit in Millisekunden für dieses Ablaufdiagramm in diesem interaktiven Kanal.
LastCompletedExecutionTimeMillis	Laufzeit in Millisekunden für die letzte Ausführung dieses Ablaufdiagramms in diesem interaktiven Kanal.
NumExecutionsCompleted	Gesamtzahl von Ausführungen, die für dieses Ablaufdiagramm in diesem interaktiven Kanal abgeschlossen wurden.
NumExecutionsStarted	Gesamtzahl von Ausführungen, die für dieses Ablaufdiagramm in diesem interaktiven Kanal gestartet wurden.

Ländereinstellungsattribute

Die Ländereinstellungsattribute sind Bestandteil der Laufzeitumgebung.

Tabelle 13. Ländereinstellung

Attribut	Beschreibung
locale	Ländereinstellung für den JMX-Client.

Attribute für die Protokollprozesskonfiguration

Die Attribute für die Protokollprozesskonfiguration sind Bestandteil der Laufzeitumgebung.

Tabelle 14. Protokollprozesskonfiguration

Attribut	Beschreibung
Kategorie	Ändern der Protokollkategorie, in der die Protokollebene manipuliert werden kann.

Attribute für Services-Thread-Pool-Statistikdaten

Die Attribute für Services-Thread-Pool-Statistikdaten sind Bestandteil der Laufzeitumgebung.

Tabelle 15. Services-Thread-Pool-Statistikdaten

Attribut	Beschreibung
activeContactHistThreads	Die näherungsweise berechnete Anzahl von Threads, die aktiv Aufgaben für den Kontakt- und Antwortverlauf ausführen.

Tabelle 15. Services-Thread-Pool-Statistikdaten (Forts.)

Attribut	Beschreibung
activeFlushCacheToDBThreads	Die näherungsweise berechnete Anzahl von Threads, die aktiv Aufgaben ausführen, um Cache-Statistikdaten in den Datenspeicher zu schreiben.
activeOtherStatsThreads	Die näherungsweise berechnete Anzahl von Threads, die aktiv Aufgaben für die Services zur Erstellung einer Berechtigungsstatistik (Eligibility Statistics), zur Erstellung einer Ereignisaktivitätsstatistik (Event Activity Statistics) und zur Erstellung einer Statistik über die Verwendung von Standardzeichenfolgen (Default Statistics) ausführen.
CurrentHighWaterMarkInContactHistQueue	Größte Anzahl von Einträgen, die in der Warteschlange eingereiht sind, um von dem Service protokolliert zu werden, der die Kontakt- und Antwortverlaufsdaten erfasst.
CurrentHighWaterMark InFlushCachetoDBQueue	Größte Anzahl von Einträgen, die in der Warteschlange eingereiht sind, um von dem Service protokolliert zu werden, der die Daten im Cache in die Datenbanktabellen schreibt.
CurrentHighWaterMarkInOtherStatsQueue	Größte Anzahl von Einträgen, die in der Warteschlange eingereiht sind, um von dem Service protokolliert zu werden, der die Berechtigungsstatistiken für Angebote, Statistiken zur Verwendung von Standardzeichenfolgen, Ereignisaktivitätsstatistiken und die aus einem benutzerdefinierten Protokoll in eine Tabelle konvertierten Daten erfasst.
currentMsgsInContactHistQueue	Die Anzahl von Jobs in der Warteschlange für den Thread-Pool, der für den Kontakt- und Antwortverlauf verwendet wird.
currentMsgsInFlushCacheToDBQueue	Die Anzahl von Jobs in der Warteschlange für den Thread-Pool, der zum Schreiben von Cache-Statistikdaten in den Datenspeicher verwendet wird.
currentMsgsInOtherStatsQueue	Die Anzahl von Jobs in der Warteschlange für den Thread-Pool, der für die Services zur Erstellung einer Berechtigungsstatistik (Eligibility Statistics), zur Erstellung einer Ereignisaktivitätsstatistik (Event Activity Statistics) und zur Erstellung einer Statistik über die Verwendung von Standardzeichenfolgen (Default Statistics) verwendet wird.

Tabelle 15. Services-Thread-Pool-Statistikdaten (Forts.)

Attribut	Beschreibung
maximumContactHistThreads	Die größte Anzahl von Threads, die sich jemals gleichzeitig in dem Pool befanden, der für den Kontakt- und Antwortverlauf verwendet wird.
maximumFlushCacheToDBThreads	Die größte Anzahl von Threads, die sich jemals gleichzeitig in dem Pool befanden, der für das Schreiben von Cache-Statistikdaten in den Datenspeicher (Flushoperation) verwendet wird.
maximumOtherStatsThreads	Die größte Anzahl von Threads, die sich jemals gleichzeitig in dem Pool befanden, der für die Services zur Erstellung einer Berechtigungsstatistik (Eligibility Statistics), zur Erstellung einer Ereignisaktivitätsstatistik (Event Activity Statistics) und zur Erstellung einer Statistik über die Verwendung von Standardzeichenfolgen (Default Statistics) verwendet wird.

Attribute für Servicestatistiken

Die Servicestatistiken bestehen aus einem Satz von Attributen für jeden Service.

- ContactHistoryMemoryCacheStatistics - Der Service, der Daten für die Kontaktverlaufs-Staging-Tabellen erfasst.
- CustomLoggerStatistics - Der Service, der benutzerdefinierte Daten erfasst, um sie in eine Tabelle zu schreiben (ein Ereignis, das den Ereignisparameter `UACICustomLoggerTableName` verwendet).
- Default Statistics - Der Service, der Statistiken dazu erfasst, wie oft die Standardzeichenfolge für den Interaktionspunkt verwendet wurde.
- Eligibility Statistics - Der Service, der Statistiken über berechtigte Angebote schreibt.
- Event Activity Statistics - Der Service, der die Ereignisstatistiken erfasst, sowohl für Systemereignisse wie `getOffer` oder `startSession` als auch für Benutzerereignisse, die durch `postEvent` ausgelöst werden.
- Response History Memory Cache Statistics - Der Service, der Daten in die Antwortverlaufs-Staging-Tabellen schreibt.
- Cross-session Response Statistics - Der Service, der sitzungsübergreifende Antwortverfolgungsdaten erfasst.

Tabelle 16. Servicestatistiken

Attribut	Beschreibung
Count	Die Anzahl der verarbeiteten Nachrichten.

Tabelle 16. Servicestatistiken (Forts.)

Attribut	Beschreibung
ExecTimeInsideMutex	Die Zeitspanne in Millisekunden, die für die Verarbeitung von Nachrichten für diesen Service benötigt wurde, außer Wartezeiten auf andere Threads. Wenn es zwischen ExecTimeInsidMutex und ExecTimeMillis eine große Differenz gibt, müssen Sie möglicherweise die Thread-Pool-Größe für diesen Service ändern.
ExecTimeMillis	Die Zeitspanne in Millisekunden, die für die Verarbeitung von Nachrichten für diesen Service benötigt wurde, einschließlich Wartezeiten auf andere Threads.
ExecTimeOfDBInsertOnly	Die Zeitspanne in Millisekunden, die nur für die Verarbeitung des Stapeleinfügungsschritts benötigt wurde.
HighWaterMark	Die maximale Anzahl von Nachrichten, die für diesen Service verarbeitet wurden.
NumberOfDBInserts	Die Gesamtzahl der ausgeführten Stapeleinfügungen.
TotalRowsInserted	Die Gesamtzahl von Zeilen, die in die Datenbank eingefügt wurden.

Attribute für Servicestatistiken - Datenbankladedienstprogramm

Die Attribute für "Servicestatistiken - Datenbankladedienstprogramm" sind Bestandteil der Laufzeitumgebung.

Tabelle 17. Servicestatistiken - Datenbankladedienstprogramm

Attribut	Beschreibung
ExecTimeOfWriteToCache	Die Zeitspanne in Millisekunden, die zum Schreiben in den Dateicache benötigt wurde, einschließlich des Schreibens in Dateien und des Abrufens des Primärschlüssels aus der Datenbank, falls erforderlich.
ExecTimeOfLoaderDBAccessOnly	Die Zeitspanne in Millisekunden, die nur für das Ausführen des Datenbankladeprogrammschritts benötigt wurde.
ExecTimeOfLoaderThreads	Die Zeitspanne in Millisekunden, die von den Datenbankladeprogrammthreads benötigt wurde.
ExecTimeOfFlushCacheFiles	Die Zeitspanne in Millisekunden, die zum Leeren des Cache und zur Neuerstellung von neuen Caches benötigt wurde.

Tabelle 17. Servicestatistiken - Datenbankladedienstprogramm (Forts.)

Attribut	Beschreibung
ExecTimeOfRetrievePKDBAccess	Die Zeitspanne in Millisekunden, die für das Abrufen des Primärschlüsseldatenbankzugriffs benötigt wurde.
NumberOfDBLoaderRuns	Die Gesamtzahl der Datenbankladeprogrammausführungen.
NumberOfLoaderStagingDirCreated	Die Gesamtzahl der erstellten Staging-Verzeichnisse.
NumberOfLoaderStagingDirRemoved	Die Gesamtzahl der entfernten Staging-Verzeichnisse.
NumberOfLoaderStagingDirMoved-ToAttention	Die Gesamtzahl der Staging-Verzeichnisse, die in den Warnungszustand versetzt wurden.
NumberOfLoaderStagingDirMovedToError	Die Gesamtzahl der Staging-Verzeichnisse, die in den Fehlerzustand versetzt wurden.
NumberOfLoaderStagingDirRecovered	Die Gesamtzahl von wiederhergestellten Staging-Verzeichnissen, einschließlich zur Startzeit und erneute Ausführung durch Hintergrundthreads.
NumberOfTimesRetrievePKFromDB	Die Gesamtzahl von Vorgängen zum Abrufen des Primärschlüssels aus der Datenbank.
NumberOfLoaderThreadsRuns	Die Gesamtzahl der Datenbankladeprogrammthreadausführungen.
NumberOfFlushCacheFiles	Die Gesamtzahl von Leerungen des Dateicache.

Attribute für API-Statistiken

Die Attribute für API-Statistiken sind Bestandteil der Laufzeitumgebung.

Tabelle 18. API-Statistiken

Attribut	Beschreibung
endSessionCount	Die Anzahl von `endSession`-API-Aufrufen seit der letzten Zurücksetzung oder dem Systemstart.
endSessionDuration	Die beim letzten `endSession`-API-Aufruf verstrichene Zeit.
executeBatchCount	Die Anzahl von `executeBatch`-API-Aufrufen seit der letzten Zurücksetzung oder dem Systemstart.
executeBatchDuration	Die beim letzten `executeBatch`-API-Aufruf verstrichene Zeit.
getOffersCount	Die Anzahl von `getOffers`-API-Aufrufen seit der letzten Zurücksetzung oder dem Systemstart.
getOffersDuration	Die beim letzten `getOffer`-API-Aufruf verstrichene Zeit.

Tabelle 18. API-Statistiken (Forts.)

Attribut	Beschreibung
getProfileCount	Die Anzahl von `getProfile`-API-Aufrufen seit der letzten Zurücksetzung oder dem Systemstart.
getProfileDuration	Die beim letzten `getProfileDuration`-API-Aufruf verstrichene Zeit.
getVersionCount	Die Anzahl von `getVersion`-API-Aufrufen seit der letzten Zurücksetzung oder dem Systemstart.
getVersionDuration	Die beim letzten `getVersion`-API-Aufruf verstrichene Zeit.
loadOfferSuppressionDuration	Die beim letzten `loadOfferSuppression`-API-Aufruf verstrichene Zeit.
LoadOffersBySQLCount	Die Anzahl von `LoadOffersBySQL`-API-Aufrufen seit der letzten Zurücksetzung oder dem Systemstart.
LoadOffersBySQLDuration	Die beim letzten `LoadOffersBySQL`-API-Aufruf verstrichene Zeit.
loadProfileDuration	Die beim letzten `loadProfile`-API-Aufruf verstrichene Zeit.
loadScoreOverrideDuration	Die beim letzten `loadScoreOverride`-API-Aufruf verstrichene Zeit.
postEventCount	Die Anzahl von `postEvent`-API-Aufrufen seit der letzten Zurücksetzung oder dem Systemstart.
postEventDuration	Die beim letzten `postEvent`-API-Aufruf verstrichene Zeit.
runSegmentationDuration	Die beim letzten `runSegmentation`-API-Aufruf verstrichene Zeit.
setAudienceCount	Die Anzahl von `setAudience`-API-Aufrufen seit der letzten Zurücksetzung oder dem Systemstart.
setAudienceDuration	Die beim letzten `setAudience`-API-Aufruf verstrichene Zeit.
setDebugCount	Die Anzahl von `setDebug`-API-Aufrufen seit der letzten Zurücksetzung oder dem Systemstart.
setDebugDuration	Die beim letzten `setDebug`-API-Aufruf verstrichene Zeit.
startSessionCount	Die Anzahl von `startSession`-API-Aufrufen seit der letzten Zurücksetzung oder dem Systemstart.
startSessionAverage	Die beim letzten `startSession`-API-Aufruf verstrichene durchschnittliche Zeit.

Tabelle 18. API-Statistiken (Forts.)

Attribut	Beschreibung
ActiveSessionCount	Die Anzahl der Sitzungen, die in der Instanz der Interact-Laufzeit aktuell aktiv sind. **Anmerkung:** Da ActiveSessionCount in der JMX-MBean com.unicacorp.interact:type=api, group=Statistics Ereignisse, die ihr zulässiges Zeitlimit überschritten haben, nicht berücksichtigt, könnte die angezeigte Anzahl falsch sein.

Attribute für Statistikdaten des Lernoptimierungsprogramms

Die Attribute für Statistikdaten des Lernoptimierungsprogramms sind Bestandteil der Laufzeitumgebung.

Tabelle 19. Statistikdaten des Lernoptimierungsprogramms

Attribut	Beschreibung
LearningOptimizerAcceptCalls	Die Anzahl von an das Lernmodul übergebenen Annahmeereignissen.
LearningOptimizer AcceptTrackingDuration	Die Gesamtzahl von Millisekunden, die zum Protokollieren der Annahmeereignisse in das Lernmodul benötigt wurde.
LearningOptimizerContactCalls	Die Anzahl von an das Lernmodul übergebenen Kontaktereignissen.
LearningOptimizer ContactTrackingDuration	Die Gesamtzahl von Millisekunden, die zum Protokollieren der Kontaktereignisse in das Lernmodul benötigt wurde.
LearningOptimizerLogOtherCalls	Die Anzahl von an das Lernmodul übergebenen Nicht-Kontakt- und Nicht-Annahmeereignissen.
LearningOptimizer LogOtherTrackingDuration	Die Zeitspanne in Millisekunden, die zum Protokollieren von anderen Ereignissen (Nicht-Kontakt und Nicht-Annahme) in das Lernmodul benötigt wurde.
LearningOptimizer NonRandomCalls	Die Häufigkeit, mit der die konfigurierte Learning-Implementierung angewandt wurde.
LearningOptimizer RandomCalls	Die Häufigkeit, mit der die konfigurierte Learning-Implementierung umgangen und eine Zufallsauswahl angewandt wurde.
LearningOptimizer RecommendCalls	Die Anzahl von an das Lernmodul übergebenen Empfehlungsanforderungen.
LearningOptimizer RecommendDuration	Die Gesamtzahl von Millisekunden, die für die Learning-Empfehlungslogik benötigt wurde.

Attribute für Standardangebotsstatistikdaten

Die Attribute für Standardangebotsstatistikdaten sind Bestandteil der Laufzeitumgebung.

Tabelle 20. Standardangebotsstatistikdaten

Attribut	Beschreibung
LoadDefaultOffersDuration	Die beim Laden der Standardangebote verstrichene Zeit.
DefaultOffersCalls	Die Häufigkeit des Standardangebotsladens.

Attribute für Dispatcher für ausgelöste Nachrichten

Attribute für Dispatcher für ausgelöste Nachrichten sind Bestandteil der Laufzeitumgebung.

Tabelle 21. Dispatcher für ausgelöste Nachrichten

Attribut	Beschreibung
NumRequested	Die Gesamtzahl der Angebote, bei denen angefordert wurde, sie mit diesem Dispatcher zu senden.
NumDispatched	Die Gesamtzahl der Angebote, die von diesem Dispatcher erfolgreich gesendet wurden.
AvgExecutionTime	Die durchschnittliche Zeit in Millisekunden, die dieser Dispatcher für das Senden eines Angebots benötigt. In der Berechnung werden nur die Angebote berücksichtigt, die erfolgreich an Gateways gesendet wurden.
CurrentQueueSize	Die Anzahl der Angebote, die derzeit darauf warten, gesendet zu werden.
GatewayInvocation	Die Anzahl der Angebote mit der durchschnittlichen Sendezeit in Millisekunden, die mit diesem Dispatcher an die einzelnen Gateways gesendet werden. Das Format des zugehörigen Wertes lautet {`gateway name=[number of offers, average dispatching time]`}.

Attribute für Gateways für ausgelöste Nachrichten

Attribute für Gateways für ausgelöste Nachrichten sind Bestandteil der Laufzeitumgebung.

Tabelle 22. Gateways für ausgelöste Nachrichten

Attribut	Beschreibung
NumValidationRequested	Die Gesamtzahl der Angebote, die von diesem Gateway zur Validierung angefordert wurden.
NumValidated	Die Gesamtzahl der Angebote, die von diesem Gateway erfolgreich validiert wurden.

Tabelle 22. Gateways für ausgelöste Nachrichten (Forts.)

Attribut	Beschreibung
AvgValidationTime	Die durchschnittliche Zeit in Millisekunden, die dieses Gateway für die Validierung eines Angebots benötigt. In der Berechnung werden nur die Angebote berücksichtigt, die erfolgreich validiert wurden.
NumDeliveryRequested	Die Gesamtzahl der Angebote, die von diesem Gateway für die Bereitstellung angefordert wurden.
NumDelivered	Die Gesamtzahl der Angebote, die von diesem Gateway erfolgreich bereitgestellt wurden.
AvgDeliveryTime	Die durchschnittliche Zeit in Millisekunden, die dieses Gateway für die Bereitstellung eines Angebots benötigt. In der Berechnung werden nur die Angebote berücksichtigt, die erfolgreich bereitgestellt wurden.

Nachrichtenattribute für ausgelöste Nachrichten

Nachrichtenattribute für ausgelöste Nachrichten sind Bestandteil der Laufzeitumgebung.

Tabelle 23. Nachrichten für ausgelöste Nachrichten

Attribut	Beschreibung
ProcessSuccessCount	Die Gesamthäufigkeit, mit der diese ausgelöste Nachricht erfolgreich ausgeführt wurde.
AvgSuccessProcessTime	Die durchschnittliche Zeit in Millisekunden, die diese ausgelöste Nachricht für jede erfolgreiche Ausführung benötigt.
ProcessErrorCount	Die Gesamthäufigkeit, mit der diese ausgelöste Nachricht nicht erfolgreich ausgeführt wurde.
AvgErrorProcessTime	Die durchschnittliche Zeit in Millisekunden, die diese ausgelöste Nachricht für jede nicht erfolgreiche Ausführung benötigt.
SelectBranchCount	Die Gesamthäufigkeit, mit der während der Verarbeitung ausgelöster Nachrichten Zweige ausgewählt wurden.
AvgSelectBranchTime	Die durchschnittliche Zeit in Millisekunden, die während der Verarbeitung ausgelöster Nachrichten für die Zweigauswahl benötigt wird.

Tabelle 23. Nachrichten für ausgelöste Nachrichten (Forts.)

Attribut	Beschreibung
SelectOfferCount	Die Gesamthäufigkeit, mit der während der Verarbeitung ausgelöster Nachrichten Angebote ausgewählt wurden.
AvgSelectOfferTime	Die durchschnittliche Zeit in Millisekunden, die während der Verarbeitung ausgelöster Nachrichten für die Angebotsauswahl benötigt wird.
SelectChannelCount	Die Gesamthäufigkeit, mit der während der Verarbeitung ausgelöster Nachrichten Kanäle ausgewählt wurden.
AvgSelectChannelTime	Die durchschnittliche Zeit in Millisekunden, die während der Verarbeitung ausgelöster Nachrichten für die Kanalauswahl benötigt wird.
FlowchartWaitCount	Die Gesamthäufigkeit, mit der diese ausgelöste Nachricht darauf gewartet hat, dass die Segmentierung abgeschlossen wird.
AvgFlowchartWaitTime	Die durchschnittliche Zeit in Millisekunden, die diese Nachricht darauf gewartet hat, dass die Segmentierung abgeschlossen wird.
WaitFlowchartTimeoutCount	Die Gesamthäufigkeit, mit der das zulässige Zeitlimit dieser Nachricht überschritten wurde, während sie darauf gewartet hat, dass die Segmentierung abgeschlossen wird.

JMX-Operationen

Für die JMX-Überwachung sind mehrere Operationen verfügbar.

In der folgenden Tabelle werden die Operationen, die für die JMX-Überwachung verfügbar sind, beschrieben.

Gruppe	Attribut	Beschreibung
Protokollprozesskonfiguration	activateDebug	Setzt die Protokollebene für die in `Interact/conf/interact_log4j.properties` definierte Protokolldatei auf "debug".
Protokollprozesskonfiguration	activateError	Setzt die Protokollebene für die in `Interact/conf/interact_log4j.properties` definierte Protokolldatei auf "error".
Protokollprozesskonfiguration	activateFatal	Setzt die Protokollebene für die in `Interact/conf/interact_log4j.properties` definierte Protokolldatei auf "fatal".

Gruppe	Attribut	Beschreibung
Protokollprozess-konfiguration	activateInfo	Setzt die Protokollebene für die in `Interact/conf/interact_log4j.properties` definierte Protokolldatei auf "info".
Protokollprozess-konfiguration	activateTrace	Setzt die Protokollebene für die in `Interact/conf/interact_log4j.properties` definierte Protokolldatei auf "trace".
Protokollprozess-konfiguration	activateWarn	Setzt die Protokollebene für die in `Interact/conf/interact_log4j.properties` definierte Protokolldatei auf "warn".
Ländereinstellung	changeLocale	Ändert die Ländereinstellung des JMX-Clients. Von Interact unterstützte Ländereinstellungen sind `de`, `en`, `es` und `fr`.
ContactResponseHistory ETLMonitor	reset	Alle Zähler zurücksetzen.
Standardangebots-statistikdaten	updatePollPeriod	Aktualisiert defaultOfferUpdatePollPeriod. Dieser Wert (in Sekunden) teilt dem System mit, wie lange es warten soll, bevor es Standardangebote im Cache aktualisiert. Bei der Einstellung `-1` liest das System die Anzahl der Standardangebote nur beim Starten.

Kapitel 7. Klassen und Methoden für die Java-, SOAP- und REST-API von IBM Interact

Die folgenden Abschnitte listen Anforderungen und andere Details auf, die Sie kennen sollten, bevor Sie mit der Interact-API zu arbeiten beginnen.

Anmerkung: Dieser Abschnitt setzt voraus, dass Sie mit Ihrem Touchpoint, der Programmiersprache Java und der Arbeit mit einer Java-basierten API vertraut sind.

Die Interact-API hat einen Java-Clientadapter, der Java-Serialisierung über HTTP verwendet. Zusätzlich stellt Interact eine WSDL bereit, um SOAP-Clients zu unterstützen. Die WSDL stellt denselben Satz von Funktionen wie der Java-Clientadapter bereit, daher sind die folgenden Abschnitte, abgesehen von den Beispielen, ebenfalls zutreffend.

Anmerkung: Die mehrmalige Verwendung eines Parameters in einem einzelnen API-Aufruf wird nicht unterstützt.

Interact-API-Klassen

Die Interact-API basiert auf der Klasse `InteractAPI`.

Es gibt 6 unterstützende Benutzeroberflächen.

- `AdvisoryMessage` (nützlicher Hinweis)
- `BatchResponse` (Batch-Antwort)
- `NameValuePair` (Name/Wert-Paar)
- `Offer` (Angebot)
- `OfferList` (Angebotsliste)
- `Response` (Antwort)

Diese Benutzeroberflächen haben 3 unterstützende konkrete Klassen. Die folgenden zwei konkreten Klassen müssen instanziiert und als Argumente in die Interact-API-Methoden übergeben werden.

- NameValuePairImpl
- CommandImpl

Eine dritte konkrete Klasse namens `AdvisoryMessageCode` ist verfügbar, um die Konstanten bereitzustellen, um ggf. die vom Server zurückgegebenen Nachrichtencodes zu unterscheiden.

Der Rest dieses Abschnitts beschreibt die Methoden, aus denen sich die Interact-API zusammensetzt.

Voraussetzungen der Java-Serialisierung über HTTP

Der Java-Clientadapter verwendet Java-Serialisierung über HTTP.

Damit der Java-Clientadapter für die Java-Serialisierung über HTTP verwendet werden kann, wird Folgendes vorausgesetzt:

1. Fügen Sie Ihrem `CLASSPATH` folgende Datei hinzu:

`Interact_Home/lib/interact_client.jar`

2. Alle Objekte, die zwischen dem Client und dem Server übergeben werden, befinden sich im Paket `com.unicacorp.interact.api`. Ausführliche Informationen finden Sie im Interact-API-Javadoc, das auf dem Laufzeitserver unter `Interact_Home/docs/apiJavaDoc` installiert ist. Sie können das Javadoc anzeigen, indem Sie die Datei `index.html` in dieser Position mit einem Web-Browser öffnen.
3. Um eine Instanz der `InteractAPI`-Klasse zu erhalten, rufen Sie die statische Methode `getInstance` mit der URL des Interact-Laufzeitservers auf.

SOAP-Voraussetzungen

Bevor Sie auf den Laufzeitserver mit SOAP zugreifen, müssen Sie zur Konfiguration Ihrer Umgebung einige Aufgaben ausführen, damit die Voraussetzungen erfüllt sind.

Wichtig: Leistungstests zeigen, dass der Java-Serialisierungsadapter mit viel größerer Geschwindigkeit als ein generierter SOAP-Client ausführt. Aus Leistungsgründen sollten Sie wann immer möglich den Java-Serialisierungsadapter verwenden.

Um auf den Laufzeitserver mithilfe von SOAP zuzugreifen, müssen Sie wie folgt vorgehen:

1. Konvertieren Sie die Interact-API-WSDL mit dem SOAP-Toolkit Ihrer Wahl.

   Die Interact-API-WSDL ist mit Interact im Verzeichnis `Interact/conf` installiert.

   Wenn Sie SOAP unter Verwendung der WSDL-XML-Dateien konfigurieren, müssen Sie Ihre URLs in den Hostnamen und Port des Laufzeitservers ändern.

   Sie finden den Text der WSDL am Ende des Interact-Administrationshandbuchs.
2. Installieren und konfigurieren Sie den Laufzeitserver.

   Der Laufzeitserver muss aktiv sein, um Ihre Integration umfassend testen zu können.
3. Prüfen Sie, ob Sie die richtige SOAP-Version verwenden.

   Interact verwendet Axis2 1.3 als die SOAP-Infrastruktur auf den Interact-Laufzeitservern. Informationen darüber, welche Versionen von SOAP Axis2 1.3 unterstützt, finden Sie auf der folgenden Website:

   Apache Axis2

   Interact wurde mit den axis2-, XFire-, JAX-WS-Ri-, DotNet-, SOAPUI- und IBM RAD SOAP-Clients getestet.

Voraussetzungen für REST

Eine Methode des Aufrufs der Interact-API besteht darin, Aufrufe im JSON-Format (JavaScript Object Notation) über HTTP zu verwenden. Dies wird im vorliegenden Dokument als REST-API bezeichnet. Die REST-API bietet gegenüber SOAP eine höhere Leistung, obwohl der Java-Serialisierungsadapter immer noch die schnellste Methode für Interact-API-Aufrufe ist.

Bevor Sie die REST-API verwenden, müssen Sie Folgendes bedenken:

- Die URL, die REST-Aufrufe an die Interact-API unterstützt, lautet:

  `http://Interact-Laufzeitserver:PORT/interact/servlet/RestServlet`, wobei in dem Pfad der tatsächliche Hostname oder die IP-Adresse des Interact-Laufzeitservers und der Port, auf dem Interact bereitgestellt wird, angegeben werden müssen.

- Es gibt zwei Interact-Klassen, die spezifisch für die REST-API sind: `RestClientConnector`, die als Hilfsprogramm dient, um die Verbindung zu einer Interact-Laufzeitinstanz über REST mit dem JSON-Format herzustellen, und `RestFieldConstants`, die das zugrunde liegende Format der JSON-Nachricht beschreibt, die für API-Anforderungen und -Antworten verwendet wird.
- Ein REST-Beispielclient wird unter `Interact _Home/samples/javaApi/InteractRestClient.java` bereitgestellt. Auch wenn es sich nur um einen einfachen Beispielcode handelt, sollte er ein geeigneter Einstieg sein, um die Verwendung der REST-API zu demonstrieren.
- Eine vollständige Beschreibung der REST-API-Klassen zusammen mit allen anderen Informationen zur Interact-API finden Sie im auf dem Laufzeitserver installierten JavaDoc unter `Interact_Home/docs/apiJavaDoc`.
- Die REST-API gibt Sitzungs-IDs und Nachrichten im Format Escaped HTML und nicht im Unicode-Format zurück.

Zusätzlich zu den hier genannten Informationen unterstützt die REST-API alle Methoden, die von den anderen Protokollen zur Verwendung der Interact-API unterstützt werden.

API-JavaDoc

Zusätzlich zu dem Interact-Administratorhandbuch ist das Javadoc für die Interact-API mit dem Laufzeitserver installiert. Das Javadoc ist für Ihre Referenz im Verzeichnis `Interact_Home/docs/apiJavaDoc` installiert.

API-Beispiele

Alle Beispiele in diesem Handbuch wurden mithilfe der Java-Serialisierung über HTTP-Adapter erstellt. Die von der WSDL generierten Klassen können je nach SOAP-Toolkit und den von Ihnen ausgewählten Optionen variieren. Wenn Sie SOAP verwenden, funktionieren diese Beispiele in Ihrer Umgebung möglicherweise nicht auf genau dieselbe Weise.

Arbeiten mit Sitzungsdaten

Wenn Sie eine Sitzung mit der Methode `startSession` initialisieren, werden Sitzungsdaten in den Speicher geladen. Während der Sitzung können Sie die Sitzungsdaten (die eine Obermenge der statischen Profildaten sind) lesen und schreiben.

Die Sitzung enthält die folgenden Daten:

- Statische Profildaten
- Segmentzuordnungen
- Echtzeitdaten
- Angebotsempfehlungen

Alle Sitzungsdaten sind bis zum Aufruf der Methode `endSession` bzw. bis zum Ablauf der `sessionTimeout`-Zeit verfügbar. Mit dem Ende der Sitzung gehen alle Daten verloren, die nicht ausdrücklich in den Kontakt- oder Antwortverlauf oder eine andere Datenbanktabelle gespeichert werden.

Die Daten werden als ein Satz von Name/Wert-Paaren gespeichert. Wenn die Daten aus der Datenbanktabelle gelesen werden, ist der Name die Spalte der Tabelle.

Sie können diese Name/Wert-Paare während der Arbeit mit der Interact-API erstellen. Sie müssen nicht alle Name/Wert-Paare in einem Globalbereich deklarieren. Wenn Sie neue Ereignisparameter als Name/Wert-Paare festlegen, fügt die Laufzeitumgebung die Name/Wert-Paare den Sitzungsdaten hinzu. Wenn Sie beispielsweise Ereignisparameter mit der Methode `postEvent` verwenden, fügt die Laufzeitumgebung die Ereignisparameter den Sitzungsdaten hinzu, selbst wenn die Ereignisparameter nicht in den Profildaten verfügbar waren. Diese Daten existieren nur in den Sitzungsdaten.

Sie können Sitzungsdaten jederzeit überschreiben. Beispiel: Wenn ein Abschnitt des Kundenprofils `creditScore` umfasst, können Sie einen Ereignisparameter mithilfe des benutzerdefinierten Typs `NameValuePair` übergeben. In der Klasse `NameValuePair` können Sie die Methoden `setName` und `setValueAsNumeric` verwenden, um den Wert zu ändern. Der Name muss übereinstimmen. Innerhalb der Sitzungsdaten muss beim Namen die Groß-/Kleinschreibung nicht berücksichtigt zu werden. Daher würden die Namen `creditscore` und `CrEdItScOrE` jeweils `creditScore` überschreiben.

Nur die letzten in die Sitzungsdaten geschriebenen Daten werden aufbewahrt. Beispiel: `startSession` lädt die Profildaten für den Wert `lastOffer`. Eine Methode `postEvent` überschreibt `lastOffer`. Dann überschreibt eine zweite Methode `postEvent` `lastOffer`. Die Laufzeitumgebung bewahrt nur die Daten, die von der zweiten Methode `postEvent` geschrieben wurden, in den Sitzungsdaten.

Wenn die Sitzung endet, gehen die Daten verloren, außer Sie haben besondere Vorkehrungen getroffen, wie z. B. die Verwendung eines Prozesses "Momentaufnahme" in Ihrem interaktiven Ablaufdiagramm, um die Daten in eine Datenbanktabelle zu schreiben. Wenn Sie vorhaben, Prozesse "Momentaufnahme" zu verwenden, achten Sie darauf, dass die Namen den Einschränkungen Ihrer Datenbank entsprechen müssen. Beispiel: Wenn nur 256 Zeichen für den Namen einer Spalte zulässig sind, darf der Name des Name/Wert-Paars nicht 256 Zeichen überschreiten.

Informationen zur Klasse InteractAPI

Die Klasse InteractAPI enthält die Methoden, die Sie verwenden, um Ihren Touchpoint in den Laufzeitserver zu integrieren. Alle anderen Klassen und Methoden in der Interact-API unterstützen die Methoden in dieser Klasse.

Sie müssen Ihre Implementierung anhand von `interact_client.jar` im `lib`-Verzeichnis Ihrer Interact-Laufzeitumgebungsinstallation kompilieren.

endSession

Die `endSession`-Methode markiert das Ende der Laufzeitsitzung. Wenn der Laufzeitserver diese Methode empfängt, wird der Verlauf protokolliert und der Speicher gelöscht.

`endSession(String sessionID)`

- **sessionID** - Eindeutige Zeichenfolge zur Identifizierung der Sitzung.

Zeitlimitüberschreitung der Laufzeitsitzungen, wenn die endSession-Methode nicht aufgerufen wird. Das Zeitlimitintervall ist mit der `sessionTimeout`-Eigenschaft konfigurierbar.

Rückgabewert

Der Laufzeitserver beantwortet die endSession-Methode mit dem Response-Objekt, das die folgenden Attribute enthält:

- SessionID
- ApiVersion
- StatusCode
- AdvisoryMessages

Beispiel

Das folgende Beispiel zeigt, wie Sie die endSession-Methode verwenden und die Antwort parsen können. sessionId ist die gleiche Zeichenfolge zur Identifizierung, die beim Start der Sitzung mit dem startSession-Aufruf verwendet wurde.

```
response = api.endSession(sessionId);
    // check if response is successful or not
    if(response.getStatusCode() == Response.STATUS_SUCCESS)
    {
        System.out.println("endSession call processed with no warnings or errors");
    }
    else if(response.getStatusCode() == Response.STATUS_WARNING)
    {
        System.out.println("endSession call processed with a warning");
    }
    else
    {
        System.out.println("endSession call processed with an error");
    }
    // For any non-successes, there should be advisory messages explaining why
    if(response.getStatusCode() != Response.STATUS_SUCCESS)
        printDetailMessageOfWarningOrError("endSession",
   response.getAdvisoryMessages());
```

executeBatch

Mit der executeBatch-Methode können Sie mehrere Methoden mit einer einzelnen Anfrage an den Laufzeitserver ausführen.

executeBatch(String *sessionID*, *CommandImpl*[] Befehle)

- **sessionID** - Eine Zeichenfolge zur Identifizierung der Sitzungs-ID. Diese Sitzungs-ID wird für alle Befehle verwendet, die dieser Methodenaufruf ausführt.
- **commandImpl[]** - Ein Array aus CommandImpl-Objekten, jeweils eines für jeden Befehl, der ausgeführt werden soll.

Durch den Aufruf dieser Methode wird das gleiche Ergebnis erzielt wie durch den expliziten Aufruf jeder einzelnen Methode im Befehl-Array. Diese Methode minimiert die Anzahl der tatsächlichen Anfragen an den Laufzeitserver. Der Laufzeitserver führt jede Methode seriell aus. Für jeden Aufruf werden alle Fehler oder Warnungen im entsprechenden Response-Objekt für diesen Methodenaufruf aufgezeichnet. Wird ein Fehler gefunden, wird executeBatch mit den verbliebenen Aufrufen im Stapel fortgesetzt. Wenn der Aufruf einer beliebigen Methode in einem Fehler resultiert, wird dieser Fehler im Status auf der höchsten Ebene für das BatchResponse-Objekt angezeigt. Wenn keine Fehler aufgetreten sind, werden im Status auf der höchsten Ebene alle aufgetretenen Warnungen angezeigt. Wenn keine Warnungen aufgetreten sind, wird im Status auf der höchsten Ebene die erfolgreiche Ausführung des Stapels angezeigt.

Rückgabewert

Der Laufzeitserver beantwortet den executeBatch mit einem BatchResponse-Objekt.

Beispiel

Das folgende Beispiel zeigt, wie Sie mit einem einzigen executeBatch-Aufruf alle getOffer- und postEvent-Methoden aufrufen und danach die Antwort bearbeiten können.

```
/** Define all variables for all members of the executeBatch*/
String sessionId="MySessionID-123";
String interactionPoint = "Overview Page Banner 1";
int numberRequested=1;
String eventName = "logOffer";

/** build the getOffers command */
Command getOffersCommand = new CommandImpl();
getOffersCommand.setMethodIdentifier(Command.COMMAND_GETOFFERS);
getOffersCommand.setInteractionPoint(interactionPoint);
getOffersCommand.setNumberRequested(numberRequested);

/** build the postEvent command */
Command postEventCommand = new CommandImpl();
postEventCommand.setMethodIdentifier(Command.COMMAND_POSTEVENT);
postEventCommand.setEventParameters(postEventParameters);
postEventCommand.setEvent(eventName);

/** Build command array */
Command[] commands =
{
  getOffersCommand,
  postEventCommand,
};

/** Make the call */
BatchResponse batchResponse = api.executeBatch(sessionId, commands);

/** Process the response appropriately */
// Top level status code is a short cut to determine if there
// are any non-successes in the array of Response objects
if(batchResponse.getBatchStatusCode() == Response.STATUS_SUCCESS)
{
  System.out.println("ExecuteBatch ran perfectly!");
}
else if(batchResponse.getBatchStatusCode() == Response.STATUS_WARNING)
{
  System.out.println("ExecuteBatch call processed with at least one warning");
}
else
{
  System.out.println("ExecuteBatch call processed with at least one error");
}

// Iterate through the array, and print out the message for any non-successes
for(Response response : batchResponse.getResponses())
{
  if(response.getStatusCode()!=Response.STATUS_SUCCESS)
  {
  printDetailMessageOfWarningOrError("executeBatchCommand",
 response.getAdvisoryMessages());
  }
}
```

executeBatch()-XML-Anforderungen für Interact-SOAP-API schreiben

Anhand der folgenden Schritte können Sie XML-Anforderungen des Typs `executeBatch()` für die Interact-SOAP-API schreiben.

Informationen zu diesem Vorgang

Die Anforderungs-XML für SOAP-API-Aufrufe, die aus einer einzelnen Operation bestehen (`startSession`, `getOffers`, `setAudience`, `endSession` etc.), darf nicht direkt in einen `executeBatch()`-Aufruf kopiert oder eingefügt werden, der aus mehreren Operationen besteht. Die Unterbefehle in Aufrufen des Typs `executeBatch()` weisen leicht abweichende WSDL- und XML-Anforderungsstrukturen im Vergleich zu API-Aufrufen auf, die aus einer einzelnen Operation bestehen. Die strukturellen Unterschiede verursachen Fehlerantworten vom Server, falls die XML-Elemente aus API-Einzeloperationsanforderungen kopiert und in `executeBatch`-Mehrfachoperationsanforderungen eingefügt werden.

Beispielfehlerantworten:

```
** XML Response Element: <ns0:faultstring>org.apache.axis2.databinding.ADBException:
Unexpected subelement audienceID</ns0:faultstring>
** Interact Server Exception: java.lang.Exception: org.apache.axis2.databinding.
ADBException: Unexpected subelement audienceID at
*** ... com.unicacorp.interact.api.soap.service.v1.xsd.CommandImpl$Factory.parse
(CommandImpl.java:1917) at
```

Anhand der folgenden Schritte können Sie eine XML-Anforderung des Typs `executeBatch()` schreiben. Sie können während der Ausführung dieser Schritte auf Parameterwerte von API-Aufruffanforderungen aus einzelnen Operationen verweisen, dürfen aber keine XML-Elemente kopieren und einfügen.

Vorgehensweise

1. Erstellen Sie mithilfe eines WSDL-Verarbeitungstools (zum Beispiel SoapUI) aus der Interact-WSDL-Datei eine korrekt formatierte XML-Anforderung des Typs `executeBatch()`.
2. Fügen Sie nach der WSDL-Definition für untergeordnete `executeBatch()`-Elemente Unterbefehle zur Anforderung hinzu.
3. Vervollständigen Sie nach der WSDL-Definition für untergeordnete `executeBatch()`-Elemente die Argumente der Unterbefehle.

getInstance

Die `getInstance`-Methode erstellt eine Instanz des Interact-APIs, das mit dem angegebenen Laufzeitserver kommuniziert.

```
getInstance(String URL)
```

Wichtig: Jede Anwendung, die Sie mit diesem Interact-API schreiben, muss `getInstance` aufrufen, um ein InteractAPI-Objekt zu instanziieren, das einem Laufzeitserver zugeordnet wird, der im URL-Parameter angegeben ist.

Wenn Sie eine Lastausgleichsfunktion für Servergruppen verwenden, können Sie den Hostnamen und den Port konfigurieren, indem Sie die Lastausgleichsfunktion verwenden. Wenn Sie keine Lastausgleichsfunktion verwenden, müssen Sie eine Logik einschließen, um turnusmäßig zwischen den verfügbaren Laufzeitservern zu wechseln.

Diese Methode eignet sich nur für die Java-Serialisierung über HTTP-Adapter. In der WSDL (Web Services Description Language) für SOAP ist keine entsprechende Methode definiert. Jede SOAP-Clientimplementierung verfügt über eine eigene Methode zum Aufbau der Endpunkt-URL.

- **URL** - Eine Zeichenfolge, die die URL für die Laufzeitinstanz angibt. Beispiel: `http://localhost:7001/Interact/servlet/InteractJSService`.

Rückgabewert

Der Laufzeitserver gibt das InteractAPI zurück.

Beispiel

Das folgende Beispiel zeigt, wie Sie ein InteractAPI-Objekt instanziieren, das auf eine Laufzeitserverinstanz verweist, die auf der gleichen Maschine ausgeführt wird wie der Touchpoint.

```
InteractAPI api=InteractAPI.getInstance("http://localhost:7001/interact/servlet/InteractJSService");
```

getOffers

Mit der `getOffers`-Methode können Sie Angebote vom Laufzeitserver anfordern.

`getOffers(String sessionID, String interactionPoint, int numberOfOffers)`

- **sessionID** - eine Zeichenfolge, die die aktuelle Sitzung angibt.
- **interactionPoint** - eine Zeichenfolge, die den Namen des Interaktionspunkts angibt, auf den diese Methode verweist.

  Anmerkung: Dieser Name muss exakt mit dem Namen des im interaktiven Kanal definierten Interaktionspunkts übereinstimmen.
- **numberOfOffers** - eine Ganzzahl, die die Anzahl der angeforderten Angebote angibt.

Bevor die `getOffers`-Methode ausgeführt wird, wartet sie so viele Millisekunden, wie in der `segmentationMaxWaitTimeInMS`-Eigenschaft angegeben sind, um die erneute Segmentierung abzuschließen. Daher kann es zu einer Verzögerung kommen, wenn Sie eine `postEvent`-Methode aufrufen, die sofort eine erneute Segmentierung oder eine `setAudience`-Methode auslöst, bevor ein `getOffers`-Aufruf erfolgt.

Rückgabewert

Der Laufzeitserver beantwortet `getOffers` mit einem Response-Objekt, das die folgenden Attribute enthält:

- AdvisoryMessages
- ApiVersion
- OfferList
- SessionID
- StatusCode

Beispiel

Dieses Beispiel zeigt, wie Sie ein einzelnes Angebot für den Interaktionspunkt "Overview Page Banner 1" anfordern und danach die Antwort bearbeiten können.

`sessionId` ist die gleiche Zeichenfolge zur Identifizierung, die beim Start der Laufzeitsitzung mit dem `startSession`-Aufruf verwendet wurde.

```
String interactionPoint = "Overview Page Banner 1";
int numberRequested=1;

/** Make the call */
response = api.getOffers(sessionId, interactionPoint, numberRequested);

/** Process the response appropriately */
 // check if response is successful or not
 if(response.getStatusCode() == Response.STATUS_SUCCESS)
 {
  System.out.println("getOffers call processed with no warnings or errors");

        /** Check to see if there are any offers */
        OfferList offerList=response.getOfferList();

        if(offerList.getRecommendedOffers() != null)
        {
            for(Offer offer : offerList.getRecommendedOffers())
            {
                // print offer
                System.out.println("Offer Name:"+offer.getOfferName());
            }
        }
        else // count on the default Offer String
        System.out.println("Default offer:"+offerList.getDefaultString(());
    }
    else if(response.getStatusCode() == Response.STATUS_WARNING)
    {
        System.out.println("getOffers call processed with a warning");
    }
    else
    {
        System.out.println("getOffers call processed with an error");
    }
    // For any non-successes, there should be advisory messages explaining why
    if(response.getStatusCode() != Response.STATUS_SUCCESS)
        printDetailMessageOfWarningOrError("getOffers",
   response.getAdvisoryMessages());
```

getOffersForMultipleInteractionPoints

Mit der `getOffersForMultipleInteractionPoints`-Methode können Sie Angebote vom Laufzeitserver für mehrere IPs mit Deduplizierung anfordern.

`getOffersForMultipleInteractionPoints(String` *`sessionID`*`, String` *`requestStr`*`)`

- **sessionID** - eine Zeichenfolge, die die aktuelle Sitzung angibt.
- **requestStr** - eine Zeichenfolge, die ein Array aus `GetOfferRequest`-Objekten angibt.

  Jedes `GetOfferRequest`-Objekt legt fest:

  – **ipName** - Der Name des Interaktionspunkts (IP), für den das Objekt Angebote anfordert

  – **numberRequested** - Die Anzahl an eindeutigen Angeboten, die für den angegebenen IP erforderlich ist

  – **offerAttributes** - Anforderungen an die Attribute der gelieferten Angebote mit einer Instanz von `OfferAttributeRequirements`

  – **duplicationPolicy** - Duplizierungsrichtlinien-ID für die Angebote, die geliefert werden

    Duplizierungsrichtlinien legen fest, ob doppelte Angebote mit verschiedenen Interaktionspunkten an einen einzelnen Methodenaufruf zurückgegeben werden sollen. (*Innerhalb* eines einzelnen Interaktionspunkts werden doppelte Angebote nie zurückgegeben.) Gegenwärtig werden zwei Duplizierungsrichtlinien unterstützt.

- NO_DUPLICATION (ID-Wert = 1). Keines der Angebote, die in den vorangegangenen GetOfferRequest-Instanzen enthalten waren, wird in diese GetOfferRequest-Instanz einbezogen (das heißt, Interact wendet die Deduplizierung an).
- ALLOW_DUPLICATION (ID-Wert = 2). Alle Angebote, die die Voraussetzungen erfüllen, die in dieser GetOfferRequest-Instanz angegeben sind, werden einbezogen. Es findet kein Abgleich der Angebote statt, die in den vorangegangenen GetOfferRequest-Instanzen enthalten waren.

Die Reihenfolge der Anfragen im Array-Parameter ist auch die Reihenfolge der Priorität, in der die Angebote geliefert werden.

Beispiel: Angenommen, die IPs in der Anfrage heißen IP1 und IP2, duplizierte Angebote sind unzulässig (mit der Duplizierungsrichtlinien-ID = 1) und jeder IP fordert zwei Angebote an. Wenn Interact die Angebote A, B und C für IP1 und die Angebote A und D für IP2 findet, enthält die Antwort die Angebote A und B für IP1 und nur das Angebot D für IP2.

Zusätzlicher Hinweis: Wenn die Duplizierungsrichtlinien-ID 1 lautet, werden Angebote, die über einen IP mit hoher Priorität geliefert wurden, nicht über diesen IP geliefert.

Bevor die getOffersForMultipleInteractionPoints-Methode ausgeführt wird, wartet sie so viele Millisekunden, wie in der segmentationMaxWaitTimeInMS-Eigenschaft angegeben sind, um die erneute Segmentierung abzuschließen. Daher kann es zu einer Verzögerung kommen, wenn Sie eine postEvent-Methode aufrufen, die sofort eine erneute Segmentierung oder eine setAudience-Methode auslöst, bevor ein getOffers-Aufruf erfolgt.

Rückgabewert

Der Laufzeitserver beantwortet getOffersForMultipleInteractionPoints mit einem Response-Objekt, das die folgenden Attribute enthält:

- AdvisoryMessages
- ApiVersion
- Array von OfferList
- SessionID
- StatusCode

Beispiel

```
InteractAPI api = InteractAPI.getInstance("url");
 String sessionId = "123";
 String requestForIP1 = "{IP1,5,1,(5,attr1=1|numeric;attr2=value2|string,
     (3,attr3=value3|string)(3,attr4=4|numeric))}";
 String requestForIP2 = "{IP2,3,2,(3,attr5=value5|string)}";
 String requestForIP3 = "{IP3,2,1}";
 String requestStr = requestForIP1 + requestForIP2 + requestForIP3;
 Response response = api.getOffersForMultipleInteractionPoints(sessionId,
          requestStr);

 if (response.getStatusCode() == Response.STATUS_SUCCESS) {
  // Check to see if there are any offers
  OfferList[] allOfferLists = response.getAllOfferLists();
  if (allOfferLists != null) {
   for (OfferList ol : allOfferLists) {
    System.out.println
("The following offers are delivered for interaction
               point " + ol.getInteractionPointName() + ":");
    for (Offer o : ol.getRecommendedOffers()) {
```

```
     System.out.println(o.getOfferName());
    }
   }
  }
 }
 else {
  System.out.println("getOffersForMultipleInteractionPoints() method calls
          returns an error with code: " + response.getStatusCode());
 }
```

Hinweis: Die Syntax von `requestStr` lautet folgendermaßen:

```
requests_for_IP[<requests_for_IP]
```

wobei

```
<requests_for_IP> = {ip_name,number_requested_for_this_ip,
     dupe_policy[,child_requirements]]}
attribute_requirements = (number_requested_for_these_attribute_requirements
     [,attribute_requirement[;individual_attribute_requirement])
     [,(attribute_requirements))
individual_attribute_requirement = attribute_name=attribute_value | attribute_type
```

Im Beispiel oben bedeutet `requestForIP1 ({IP1,5,1,(5,attr1=1|numeric; attr2=value2|string, (3,attr3=value3|string)(3,attr4=4|numeric))})` für den Interaktionspunkt IP1 eine Lieferung von 5 möglichst verschiedenen Angeboten, die während dieses Methodenaufrufs nicht auch von einem anderen Interaktionspunkt zurückgegeben werden können. Alle 5 Angebote müssen über das numerische Attribut `attr1` mit dem Wert 1 und über das Zeichenfolgeattribut `attr2` mit dem Wert *value2* verfügen. Von diesen 5 Angeboten dürfen maximal 3 über das Zeichenfolgeattribut `attr3` mit dem Wert *value3* und maximal 3 über das numerische Attribut `attr4` mit dem Wert *4* verfügen.

Die zulässigen Attributtypen sind numerisch, Zeichenfolge und Datum/Uhrzeit und der Wert des Datum/Uhrzeit-Attributs muss dem Format `MM/dd/yyyy HH:mm:ss` entsprechen. Zum Abrufen der zurückgegebenen Angebote verwenden Sie die Methode `Response.getAllOfferLists()`. Zum besseren Verständnis der Syntax wird im Beispiel in `setGetOfferRequests` die gleiche Instanz von `GetOfferRequests` bevorzugt mit Java-Objekten erstellt.

getProfile

Mit der `getProfile`-Methode können Sie Profildaten und temporäre Informationen über die Besucher des Touchpoints abrufen.

`getProfile(String sessionID)`

- **sessionID** - eine Zeichenfolge zur Identifizierung der Sitzungs-ID.

Rückgabewert

Der Laufzeitserver beantwortet `getProfile` mit einem Response-Objekt, das die folgenden Attribute enthält:

- AdvisoryMessages
- ApiVersion
- ProfileRecord
- SessionID
- StatusCode

Beispiel

Das folgende Beispiel zeigt, wie Sie `getProfile` verwenden und danach die Antwort bearbeiten können.

`sessionId` ist die gleiche Zeichenfolge zur Identifizierung, die beim Start der Sitzung mit dem `startSession`-Aufruf verwendet wurde.

```
response = api.getProfile(sessionId);
/** Process the response appropriately */
    // check if response is successful or not
    if(response.getStatusCode() == Response.STATUS_SUCCESS)
    {
        System.out.println("getProfile call processed with no warnings or errors");
        // Print the profile - it's just an array of NameValuePair objects
        for(NameValuePair nvp : response.getProfileRecord())
        {
            System.out.println("Name:"+nvp.getName());
            if(nvp.getValueDataType().equals(NameValuePair.DATA_TYPE_DATETIME))
            {
                System.out.println("Value:"+nvp.getValueAsDate());
            }
            else if(nvp.getValueDataType().equals(NameValuePair.DATA_TYPE_NUMERIC))
            {
                System.out.println("Value:"+nvp.getValueAsNumeric());
            }
            else
            {
                System.out.println("Value:"+nvp.getValueAsString());
            }
        }
    }
    else if(response.getStatusCode() == Response.STATUS_WARNING)
    {
        System.out.println("getProfile call processed with a warning");
    }
    else
    {
        System.out.println("getProfile call processed with an error");
    }
    // For any non-successes, there should be advisory messages explaining why
    if(response.getStatusCode() != Response.STATUS_SUCCESS)
        printDetailMessageOfWarningOrError("getProfile",
   response.getAdvisoryMessages());
```

getVersion

Die `getVersion`-Methode gibt die Version der aktuellen Implementierung des Interact Laufzeitservers zurück.

```
getVersion()
```

Es empfiehlt sich, diese Methode zu verwenden, wenn Sie den Touchpoint mit dem Interact API initialisieren.

Rückgabewert

Der Laufzeitserver beantwortet `getVersion` mit einem Response-Objekt, das die folgenden Attribute enthält:

- AdvisoryMessages
- ApiVersion
- StatusCode

Beispiel

Dieses Beispiel zeigt eine einfache Methode, wie Sie `getVersion` aufrufen und die Ergebnisse verarbeiten können.

```
response = api.getVersion();
/** Process the response appropriately */
    // check if response is successful or not
    if(response.getStatusCode() == Response.STATUS_SUCCESS)
    {
        System.out.println("getVersion call processed with no warnings or errors");
        System.out.println("API-Version:" + response.getApiVersion(););
    }
    else if(response.getStatusCode() == Response.STATUS_WARNING)
    {
        System.out.println("getVersion call processed with a warning");
    }
    else
    {
        System.out.println("getVersion call processed with an error");
    }

    // For any non-successes, there should be advisory messages explaining why
    if(response.getStatusCode() != Response.STATUS_SUCCESS)
        printDetailMessageOfWarningOrError("getVersion",
   response.getAdvisoryMessages());
```

postEvent

Mit der `postEvent`-Methode können Sie jedes Ereignis ausführen, das im interaktiven Kanal definiert ist.

`postEvent(String` *`sessionID`*`, String` *`eventName`*`, NameValuePairImpl[]` *`eventParameters`*`)`

- **sessionID**: eine Zeichenfolge zur Identifizierung der Sitzungs-ID.
- **eventName**: eine Zeichenfolge zur Identifizierung des Ereignisnamens.

  Anmerkung: Der Name des Ereignisses muss mit dem im interaktiven Kanal definierten Ereignisnamen übereinstimmen. Bei diesem Namen braucht die Groß-/Kleinschreibung nicht berücksichtigt zu werden.
- **eventParameters**. `NameValuePairImpl`-Objekte geben alle Parameter an, die mit dem Ereignis übergeben werden müssen. Diese Werte werden in den Sitzungsdaten gespeichert.

  Wenn dieses Ereignis eine erneute Segmentierung auslöst, müssen Sie sicherstellen, dass alle vom interaktiven Ablaufdiagramm benötigten Daten in den Sitzungsdaten verfügbar sind. Wenn diese Werte noch nicht durch vorangegangene Aktionen ausgefüllt wurden (zum Beispiel durch `startSession`, durch `setAudience` oder beim Laden der Profiltabelle), müssen Sie für jeden fehlenden Wert einen eventParameter einschließen. Wenn Sie zum Beispiel alle Profiltabellen so konfiguriert haben, dass diese im Hauptspeicher geladen werden, müssen Sie für alle temporären Daten, die für interaktive Ablaufdiagramme erforderlich sind, jeweils ein NameValuePair einschließen.

  Wenn Sie mehrere Zielgruppenebenen verwenden, haben Sie vermutlich verschiedene Sätze an eventParameters für jede Zielgruppenebene. Sie sollten daher eine entsprechende Logik einschließen, die gewährleistet, dass für jede Zielgruppenebene immer der richtige Parametersatz ausgewählt wird.

Wichtig: Wenn dieses Ereignis den Antwortverlauf protokolliert, müssen Sie den Verfahrenscode für das Angebot übergeben. Sie müssen den Namen für das NameValuePair als "UACIOfferTrackingCode" definieren.

Pro Ereignis können Sie immer nur einen Verfahrenscode übergeben. Wenn Sie den Verfahrenscode für einen Angebotskontakt nicht übergeben, protokolliert Interact jeweils einen Angebotskontakt für jedes Angebot in der zuletzt empfohlenen Angebotsliste. Wenn Sie den Verfahrenscode für eine Antwort nicht übergeben, gibt Interact einen Fehler zurück.

- Es gibt eine Reihe weiterer reservierter Parameter, die Sie mit `postEvent` und anderen Methoden verwenden können, die später in diesem Abschnitt erläutert werden.

Wenn Sie eine erneute Segmentierung anfordern oder in den Kontakt- oder Antwortverlauf schreiben, wird nicht auf eine Antwort gewartet.

Im Verlauf einer neuen Segmentierung werden frühere Segmentierungsergebnisse für die aktuelle Zielgruppenebene nicht bereinigt. Sie können den Parameter `UACIExecuteFlowchartByName` zum Definieren bestimmter Ablaufdiagramme verwenden, die ausgeführt werden sollen. Die `getOffers`-Methode wartet, bis die erneute Segmentierung abgeschlossen ist, und wird erst danach ausgeführt. Daher kann es zu einer Verzögerung kommen, wenn Sie eine `postEvent`-Methode aufrufen, die sofort eine erneute Segmentierung auslöst, bevor ein `getOffers`-Aufruf erfolgt.

Rückgabewert

Der Laufzeitserver beantwortet `postEvent` mit einem Response-Objekt, das die folgenden Attribute enthält:

- AdvisoryMessages
- ApiVersion
- SessionID
- StatusCode

Beispiel

Das folgende `postEvent`-Beispiel zeigt, wie Sie neue Parameter für ein Ereignis, das eine erneute Segmentierung auslöst, senden und danach die Antwort bearbeiten können.

`sessionId` ist die gleiche Zeichenfolge zur Identifizierung, die beim Start der Sitzung mit dem `startSession`-Aufruf verwendet wurde.

```
String eventName = "SearchExecution";

NameValuePair parmB1 = new NameValuePairImpl();
parmB1.setName("SearchString");
parmB1.setValueAsString("mortgage");
parmB1.setValueDataType(NameValuePair.DATA_TYPE_STRING);

NameValuePair parmB2 = new NameValuePairImpl();
parmB2.setName("TimeStamp");
parmB2.setValueAsDate(new Date());
parmB2.setValueDataType(NameValuePair.DATA_TYPE_DATETIME);

NameValuePair parmB3 = new NameValuePairImpl();
parmB3.setName("Browser");
parmB3.setValueAsString("IE6");
parmB3.setValueDataType(NameValuePair.DATA_TYPE_STRING);
```

```
NameValuePair parmB4 = new NameValuePairImpl();
parmB4.setName("FlashEnabled");
parmB4.setValueAsNumeric(1.0);
parmB4.setValueDataType(NameValuePair.DATA_TYPE_NUMERIC);

NameValuePair parmB5 = new NameValuePairImpl();
parmB5.setName("TxAcctValueChange");
parmB5.setValueAsNumeric(0.0);
parmB5.setValueDataType(NameValuePair.DATA_TYPE_NUMERIC);

NameValuePair parmB6 = new NameValuePairImpl();
parmB6.setName("PageTopic");
parmB6.setValueAsString("");
parmB6.setValueDataType(NameValuePair.DATA_TYPE_STRING);

NameValuePair[] postEventParameters = { parmB1,
 parmB2,
 parmB3,
 parmB4,
 parmB5,
 parmB6
};

/** Make the call */
response = api.postEvent(sessionId, eventName, postEventParameters);

/** Process the response appropriately */
    // check if response is successful or not
    if(response.getStatusCode() == Response.STATUS_SUCCESS)
    {
        System.out.println("postEvent call processed with no warnings or errors");
    }
    else if(response.getStatusCode() == Response.STATUS_WARNING)
    {
        System.out.println("postEvent call processed with a warning");
    }
    else
    {
        System.out.println("postEvent call processed with an error");
    }

    // For any non-successes, there should be advisory messages explaining why
    if(response.getStatusCode() != Response.STATUS_SUCCESS)
        printDetailMessageOfWarningOrError("postEvent",
   response.getAdvisoryMessages());
```

setAudience

Mit der setAudience-Methode können Sie die Zielgruppen-ID und die Zielgruppenebene für Besucher festlegen.

```
setAudience(String sessionID, NameValuePairImpl[] audienceID,
     String audienceLevel, NameValuePairImpl[] parameters)
```

- **sessionID** - eine Zeichenfolge zur Identifizierung der Sitzungs-ID.
- **audienceID** - ein Array von NameValuePairImpl-Objekten zum Definieren der Zielgruppen-ID.
- **audienceLevel** - eine Zeichenfolge zum Definieren der Zielgruppenebene.
- **parameters** - NameValuePairImpl-Objekte zum Identifizieren aller Parameter, die mit setAudience übergeben werden müssen. Diese Werte werden in den Sitzungsdaten gespeichert und können zur Segmentierung verwendet werden.

  Sie benötigen für jede Spalte in Ihrem Profil einen Wert. Dies ist eine Obermenge aus allen Spalten in den Echtzeitdaten und in allen Tabellen, die für den interak-

tiven Kanal definiert sind. Wenn Sie bereits alle Sitzungsdaten mit `startSession` oder `postEvent` ausgefüllt haben, ist es nicht erforderlich, neue Parameter zu senden.

Die `setAudience`-Methode löst eine erneute Segmentierung aus. Die `getOffers`-Methode wartet, bis die erneute Segmentierung abgeschlossen ist, und wird erst danach ausgeführt. Daher kann es zu einer Verzögerung kommen, wenn Sie eine `setAudience`-Methode aufrufen, bevor ein `getOffers`-Aufruf erfolgt.

Die `setAudience`-Methode lädt auch die Profildaten für die Zielgruppen-ID. Mit der `setAudience`-Methode können Sie erzwingen, dass erneut die gleichen Profildaten geladen werden wie mit der `startSession`-Methode.

Rückgabewert

Der Laufzeitserver beantwortet `setAudience` mit einem Response-Objekt, das die folgenden Attribute enthält:

- AdvisoryMessages
- ApiVersion
- SessionID
- StatusCode

Beispiel

In diesem Beispiel bleibt die Zielgruppenebene gleich, aber die ID ändert sich, wie wenn sich ein anonymer Benutzer anmeldet und dann bekannt wird.

`sessionId` und `audienceLevel` sind die gleichen Zeichenfolgen, die beim Start der Sitzung mit dem `startSession`-Aufruf verwendet wurden, um die Sitzung und die Zielgruppenebene zu identifizieren.

```
NameValuePair custId2 = new NameValuePairImpl();
custId2.setName("CustomerId");
custId2.setValueAsNumeric(123.0);
custId2.setValueDataType(NameValuePair.DATA_TYPE_NUMERIC);

NameValuePair[] newAudienceId = { custId2 };

/** Parameters can be passed in as well.  For this example, there are no parameters
* therefore pass in null */
NameValuePair[] noParameters=null;

/** Make the call */
response = api.setAudience(sessionId, newAudienceId, audienceLevel, noParameters);

 /** Process the response appropriately */
    // check if response is successful or not
    if(response.getStatusCode() == Response.STATUS_SUCCESS)
    {
        System.out.println("setAudience call processed with no warnings or errors");
    }
    else if(response.getStatusCode() == Response.STATUS_WARNING)
    {
        System.out.println("setAudience call processed with a warning");
    }
    else
    {
        System.out.println("setAudience call processed with an error");
    }

    // For any non-successes, there should be advisory messages explaining why
    if(response.getStatusCode() != Response.STATUS_SUCCESS)
        printDetailMessageOfWarningOrError("setAudience",
   response.getAdvisoryMessages());
```

setDebug

Mit der setDebug-Methode können Sie den Detaillierungsgrad der Protokollierung für alle Codepfade für die Sitzung festlegen.

```
setDebug(String sessionID, boolean debug)
```

- **sessionID** - eine Zeichenfolge zur Identifizierung der Sitzungs-ID.
- **debug** - ein boolescher Ausdruck zum Aktivieren oder Inaktivieren von Daten zur Fehlerbehebung. Gültige Werte sind true oder false. Wenn der Wert wahr ist, protokolliert Interact die Daten zur Fehlerbehebung im Protokoll des Laufzeitservers.

Rückgabewert

Der Laufzeitserver beantwortet setDebug mit einem Response-Objekt, das die folgenden Attribute enthält:

- AdvisoryMessages
- ApiVersion
- SessionID
- StatusCode

Beispiel

Im folgenden Beispiel wird die Fehlerbehebungsstufe der Sitzung geändert.

sessionId ist die gleiche Zeichenfolge zur Identifizierung, die beim Start der Sitzung mit dem startSession-Aufruf verwendet wurde.

```
boolean newDebugFlag=false;
/** make the call */
response = api.setDebug(sessionId, newDebugFlag);

/** Process the response appropriately */
    // check if response is successful or not
    if(response.getStatusCode() == Response.STATUS_SUCCESS)
    {
        System.out.println("setDebug call processed with no warnings or errors");
    }
    else if(response.getStatusCode() == Response.STATUS_WARNING)
    {
        System.out.println("setDebug call processed with a warning");
    }
    else
    {
        System.out.println("setDebug call processed with an error");
    }

    // For any non-successes, there should be advisory messages explaining why
    if(response.getStatusCode() != Response.STATUS_SUCCESS)
        printDetailMessageOfWarningOrError("setDebug",
   response.getAdvisoryMessages());
```

startSession

Die startSession-Methode erstellt und definiert eine Laufzeitsitzung.

```
startSession(String sessionID,
   boolean relyOnExistingSession,
   boolean debug,
   String interactiveChannel,
   NameValuePairImpl[] audienceID,
   String audienceLevel,
   NameValuePairImpl[] parameters)
```

startSession kann bis zu fünf Aktionen auslösen:

- Erstellen der Laufzeitsitzung.
- Laden der Besucherprofildaten für die aktuelle Zielgruppenebene in der Laufzeitsitzung inklusive aller Dimensionstabellen, die in der für den interaktiven Kanal definierten Tabellenzuordnung zum Laden markiert sind.
- Auslösen der Segmentierung, indem alle interaktiven Ablaufdiagramme für die aktuelle Zielgruppenebene ausgeführt werden.
- Laden von Angebotsunterdrückungsdaten in der Sitzung, wenn dieenableOfferSuppressionLookup-Eigenschaft auf wahr gesetzt ist.
- Laden von Bewertungsüberschreibungsdaten in der Sitzung, wenn dieenableScoreOverrideLookup-Eigenschaft auf wahr gesetzt ist.

Die startSession-Methode benötigt die folgenden Parameter:

- **sessionID** - eine Zeichenfolge zur Identifizierung der Sitzungs-ID. Sie müssen die Sitzungs-ID definieren. Sie können zum Beispiel eine Kombination aus Kunden-ID und Zeitmarke verwenden.

  Sie müssen eine Sitzungs-ID angeben, um zu definieren, was eine Laufzeitsitzung auszeichnet. Dieser Wert wird vom Client verwaltet. Alle Methodenaufrufe für die gleiche Sitzungs-ID müssen vom Client synchronisiert werden. Das Verhalten für gleichzeitige API-Aufrufe mit der gleichen Sitzungs-ID ist nicht definiert.
- **relyOnExistingSession** - ein boolescher Ausdruck, der definiert, ob diese Sitzung eine neue oder eine vorhandene Sitzung verwendet. Gültige Werte sind true oder false. Wenn der Wert true ist, müssen Sie eine vorhandene Sitzungs-ID mit der startSession-Methode angeben. Wenn der Wert false ist, müssen Sie eine neue Sitzungs-ID angeben.

  Wenn Sie relyOnExistingSession auf true setzen und eine Sitzung vorhanden ist, verwendet die Laufzeitumgebung die vorhandenen Sitzungsdaten. Es werden keine Daten erneut geladen und es wird keine Segmentierung ausgelöst. Wenn die Sitzung nicht vorhanden ist, erstellt die Laufzeitumgebung eine neue Sitzung inklusive Laden der Daten und Auslösen der Segmentierung. Wenn die Sitzungsdauer des Touchpoints länger als die Laufzeitsitzung ist, kann es sinnvoll sein, relyOnExistingSession auf true zu setzen und mit allen startSession-Aufrufen zu verwenden. Beispiel: Die Sitzung einer Website ist 2 Stunden lang aktiv, während die Laufzeitsitzung nur 20 Minuten lang aktiv ist.

  Wenn Sie startSession zweimal mit der gleichen Sitzungs-ID aufrufen, gehen alle Sitzungsdaten des ersten startSession-Aufrufs verloren, wenn relyOnExistingSession auf false gesetzt ist.
- **debug** - ein boolescher Ausdruck zum Aktivieren oder Inaktivieren von Daten zur Fehlerbehebung. Gültige Werte sind true oder false. Wenn der Wert wahr ist, protokolliert Interact die Daten zur Fehlerbehebung in den Protokollen des Laufzeitservers. Das Debugflag wird für jede Sitzung individuell gesetzt. Somit können Sie die Daten zur Fehlerbehebung für einzelne Sitzungen verfolgen.
- **interactiveChannel** - eine Zeichenfolge, die den Namen des interaktiven Kanals definiert, auf den diese Sitzung verweist. Dieser Name muss exakt mit dem in Campaign definierten Namen des interaktiven Kanals übereinstimmen.
- **audienceID** - ein Array aus NameValuePairImpl-Objekten, wobei die Namen mit den physischen Spaltennamen aller Tabellen übereinstimmen müssen, in denen die Zielgruppen-ID enthalten ist.
- **audienceLevel** - eine Zeichenfolge, die die Zielgruppenebene definiert.

- **parameters** - NameValuePairImpl-Objekte zum Identifizieren aller Parameter, die mit `startSession` übergeben werden müssen. Diese Werte werden in den Sitzungsdaten gespeichert und können zur Segmentierung verwendet werden.

  Wenn Sie mehrere interaktive Ablaufdiagramme für die gleiche Zielgruppenebene haben, müssen Sie eine Obermenge mit allen Spalten in allen Tabellen einschließen. Wenn Sie die Laufzeit so konfigurieren, dass die Profiltabelle geladen wird, und die Profiltabelle alle benötigten Spalten enthält, ist es nicht erforderlich, Parameter zu übergeben, es sei denn, Sie möchten die Daten in der Profiltabelle überschreiben. Wenn die Profiltabelle eine Untermenge der benötigten Spalten enthält, müssen Sie die fehlenden Spalten als Parameter einschließen.

Wenn `audienceID` oder `audienceLevel` ungültig und `relyOnExistingSession` false ist, schlägt der `startSession`-Aufruf fehl. Wenn `interactiveChannel` ungültig ist, schlägt `startSession` fehl, unabhängig davon, ob `relyOnExistingSession` true oder false ist.

Wenn `relyOnExistingSession` true ist und Sie einen zweiten `startSession`-Aufruf mit der gleichen `sessionID` durchführen, nachdem die erste Sitzung bereits abgelaufen ist, erstellt Interact eine neue Sitzung.

Wenn `relyOnExistingSession` true ist und Sie einen zweiten `startSession`-Aufruf mit der gleichen `sessionID`, aber mit einer anderen `audienceID` oder `audienceLevel` durchführen, ändert der Laufzeitserver die Zielgruppe für die vorhandene Sitzung.

Wenn `relyOnExistingSession` true ist und Sie einen zweiten `startSession`-Aufruf mit der gleichen `sessionID`, aber mit einem anderen `interactiveChannel` durchführen, erstellt der Laufzeitserver eine neue Sitzung.

Rückgabewert

Der Laufzeitserver beantwortet `startSession` mit einem Response-Objekt, das die folgenden Attribute enthält:

- AdvisoryMessages (wenn StatusCode nicht 0 ist)
- ApiVersion
- SessionID
- StatusCode

Beispiel

Das folgende Beispiel zeigt eine Möglichkeit zum Aufrufen von `startSession`.

```
String sessionId="MySessionID-123";
String audienceLevel="Customer";
NameValuePair custId = new NameValuePairImpl();
custId.setName("CustomerId");
custId.setValueAsNumeric(1.0);
custId.setValueDataType(NameValuePair.DATA_TYPE_NUMERIC);
NameValuePair[] initialAudienceId = { custId };
boolean relyOnExistingSession=false;
boolean initialDebugFlag=true;
String interactiveChannel="Accounts Website";
NameValuePair parm1 = new NameValuePairImpl();
parm1.setName("SearchString");
parm1.setValueAsString("");
parm1.setValueDataType(NameValuePair.DATA_TYPE_STRING);

NameValuePair parm2 = new NameValuePairImpl();
parm2.setName("TimeStamp");
```

```
parm2.setValueAsDate(new Date());
parm2.setValueDataType(NameValuePair.DATA_TYPE_DATETIME);

NameValuePair parm3 = new NameValuePairImpl();
parm3.setName("Browser");
parm3.setValueAsString("IE6");
parm3.setValueDataType(NameValuePair.DATA_TYPE_STRING);

NameValuePair parm4 = new NameValuePairImpl();
parm4.setName("FlashEnabled");
parm4.setValueAsNumeric(1.0);
parm4.setValueDataType(NameValuePair.DATA_TYPE_NUMERIC);

NameValuePair parm5 = new NameValuePairImpl();
parm5.setName("TxAcctValueChange");
parm5.setValueAsNumeric(0.0);
parm5.setValueDataType(NameValuePair.DATA_TYPE_NUMERIC);

NameValuePair parm6 = new NameValuePairImpl();
parm6.setName("PageTopic");
parm6.setValueAsString("");
parm6.setValueDataType(NameValuePair.DATA_TYPE_STRING);

/** Specifying the parameters (optional) */
NameValuePair[] initialParameters = { parm1,
 parm2,
 parm3,
 parm4,
 parm5,
 parm6
 };

/** Make the call */
response = api.startSession(sessionId, relyOnExistingSession, initialDebugFlag,
 interactiveChannel, initialAudienceId, audienceLevel, initialParameters);

/** Process the response appropriately */
processStartSessionResponse(response);
```

processStartSessionResponse ist eine Methode, um das von startSession zurückgegebene Antwortobjekt zu bearbeiten.

```
public static void processStartSessionResponse(Response response)
{
  // check if response is successful or not
  if(response.getStatusCode() == Response.STATUS_SUCCESS)
 {
  System.out.println("startSession call processed with no warnings or errors");
 }
  else if(response.getStatusCode() == Response.STATUS_WARNING)
 {
  System.out.println("startSession call processed with a warning");
 }
  else
 {
  System.out.println("startSession call processed with an error");
 }

 // For any non-successes, there should be advisory messages explaining why
 if(response.getStatusCode() != Response.STATUS_SUCCESS)
  printDetailMessageOfWarningOrError("StartSession",
   response.getAdvisoryMessages());
}
```

Deduplizierung von Angeboten über Angebotsattribute hinweg

Wenn Sie die Anwendungsprogrammierschnittstelle (API) von Interact verwenden, können Sie mit den folgenden beiden API-Aufrufen Angebote erhalten: `getOffers` und `getOffersForMultipleInteractionPoints`.
`getOffersForMultipleInteractionPoints` kann auf der Ebene *OfferID* verhindern, dass doppelte Angebote zurückgegeben werden. Die Deduplizierung von Angeboten über die gesamte Angebotskategorie hinweg ist jedoch nicht möglich. Früher war beispielsweise eine Ausweichmaßnahme erforderlich, damit Interact immer nur jeweils ein Angebot aus jeder Angebotskategorie zurückgab. Der API-Aufruf `startSession` enthält jetzt zwei neue Parameter, die ab sofort eine Deduplizierung von Angeboten über Angebotsattribute hinweg ermöglichen (beispielsweise für Kategorien).

In dieser Liste finden Sie eine Übersicht über die Parameter, die dem API-Aufruf `startSession` hinzugefügt wurden. Sie finden weitere Informationen zu diesen Parametern oder zu sonstigen Aspekten der Interact-API im *IBM Interact-Administratorhandbuch* sowie in den Javadoc-Dateien, die in Ihrer Interact-Installation enthalten sind. Sie finden diese Dateien im Pfad `<Interact_Home>/docs/apiJavaDoc`.

- `UACIOfferDedupeAttribute`. Wenn Sie einen `startSession`-API-Aufruf mit Angebotsdeduplizierung erstellen möchten, damit nachfolgende `getOffer`-Aufrufe immer nur jeweils ein Angebot aus jeder Kategorie zurückgeben, müssen Sie den Parameter `UACIOfferDedupeAttribute` als Bestandteil des API-Aufrufs einschließen. Sie können einen Parameter wie folgt im Format `Name,Wert,Typ` angeben:

  `UACIOfferDedupeAttribute,<Attributname>,string`

  In diesem Beispiel würden Sie *<Attributname>* durch den Namen des Angebotsattributs ersetzen, das Sie als Kriterium für die Deduplizierung verwenden möchten (zum Beispiel "Category" (Kategorie)).

  Anmerkung: Interact überprüft die Angebote, die den von Ihnen angegebenen Attributwert (zum Beispiel "Category") aufweisen. Anschließend wird eine Deduplizierung durchgeführt, damit mit Ausnahme des Angebots, das die höchste Bewertung aufweist, alle Angebote entfernt werden. Wenn die Angebote mit einem Attributduplikat auch identische Bewertungen aufweisen, trifft Interact eine Zufallsauswahl unter den übereinstimmenden Angeboten und gibt diese zurück.

- `UACINoAttributeDedupeIfFewerOf`. Wenn Sie den Parameter `UACIOfferDedupeAttribute` im `startSession`-Aufruf einschließen, können Sie auch den Parameter `UACINoAttributeDedupeIfFewerOf` festlegen. Damit geben Sie an, wie sich das Programm verhalten soll, wenn die Angebotsliste nach der Deduplizierung nicht genügend Angebote enthält, damit die ursprüngliche Anforderung erfüllt werden kann.

  Wenn Sie beispielsweise für `UACIOfferDedupeAttribute` die Verwendung der Angebotskategorie zur Deduplizierung von Angeboten festlegen und Ihr nachfolgender getOffers-Aufruf die Rückgabe von acht Angeboten anfordert, kann es sein, dass aufgrund der Deduplizierung weniger als acht auswählbare Angebote verfügbar sind. Wenn Sie den Parameter `UACINoAttributeDedupeIfFewerOf` auf "true" setzen, werden in diesem Fall einige der deduplizierten Angebote zur Kandidatenliste hinzugefügt, damit die angeforderte Anzahl der Angebote erfüllt wird. Wenn Sie in diesem Beispiel den Parameter auf "false" setzen, wird die angeforderte Anzahl nicht durch die zurückgegebene Anzahl an Angeboten erreicht.

  `UACINoAttributeDedupeIfFewerOf` ist standardmäßig auf "true" gesetzt.

Angenommen, Sie haben wie folgt als Parameter für `startSession` die Angebotskategorie (Category) als Deduplizierungskriterium angegeben:

```
UACIOfferDedupeAttribute, Category,
string;UACINoAttributeDedupeIfFewerOffer, 0, string
```

Diese Parameterkombination bewirkt, dass Interact Angebote auf Basis des Angebotsattributs "Category" dedupliziert und selbst dann nur die deduplizierten Angebote zurückgibt, wenn die resultierende Anzahl der Angebote die angeforderte Anzahl unterschreitet (da der Parameter `UACINoAttributeDedupeIfFewerOffer` auf "false" gesetzt ist).

Wenn Sie einen `getOffers`-API-Aufruf ausgeben, könnte die ursprüngliche Gruppe der auswählbaren Angebote die folgenden Angebote enthalten:

- Category=Electronics: Angebot A1 mit einer Bewertung von 100 und Angebot A2 mit einer Bewertung von 50.
- Category=Smartphones: Angebot B1 mit einer Bewertung von 100, Angebot B2 mit einer Bewertung von 80 und Angebot B3 mit einer Bewertung von 50.
- Category=MP3Players: Angebot C1 mit einer Bewertung von 100, Angebot C2 mit einer Bewertung von 50.

In diesem Fall gab es zwei doppelte Angebote, die mit der ersten Kategorie übereinstimmen, drei doppelte Angebote, die mit der zweiten Kategorie übereinstimmen, und zwei doppelte Angebote, die mit der dritten Kategorie übereinstimmen. Als Angebote werden die Angebote mit der höchsten Bewertung aus jeder Kategorie zurückgegeben, also Angebot A1, Angebot B1 und Angebot C1.

Obwohl der `getOffers`-API-Aufruf sechs Angebote angefordert hat, werden nur drei Angebote zurückgegeben, da der Parameter `UACINoAttributeDedupeIfFewerOffer` in diesem Beispiel auf "false" gesetzt ist.

Wenn der `getOffers`-API-Aufruf sechs Angebote angefordert hat und in diesem Beispiel keine Angabe für den Parameter `UACINoAttributeDedupeIfFewerOffer` gemacht oder der Parameter explizit auf "true" gesetzt wurde, werden einige der doppelten Angebote in das Ergebnis aufgenommen, damit die angeforderte Anzahl erfüllt wird.

Reservierte Parameter

Mit dem Interact API werden mehrere reservierte Parameter verwendet. Einige werden für den Laufzeitserver benötigt und andere können für zusätzliche Funktionen verwendet werden.

postEvent-Funktionen

Funktion	Parameter	Beschreibung
In angepasste Tabelle protokollieren	UACICustomLoggerTableName	Der Name einer Tabelle in der Datenquelle der Laufzeittabellen. Wenn Sie diesen Parameter mit einem gültigen Tabellennamen angeben, schreibt die Laufzeitumgebung alle Sitzungsdaten in die ausgewählte Tabelle. In der Tabelle werden alle Spaltennamen ausgefüllt, die mit den NameValuePair-Sitzungsdaten übereinstimmen. Die Laufzeitumgebung schreibt eine Null in jede Spalte, die nicht mit einem Name/Wert-Paar der Sitzung übereinstimmt. Mit den customLogger-Konfigurationseigenschaften können Sie den Prozess verwalten, der in die Datenbank schreibt.
Mehrere Antworttypen	UACILogToLearning	Eine ganze Zahl mit dem Wert "1" oder "0". "1" gibt an, dass das Ereignis von der Laufzeitumgebung als Annahme für die Lernfunktion protokolliert bzw. die Angebotsunterdrückung in einer Sitzung aktiviert werden soll. "0" gibt an, dass das Ereignis von der Laufzeitumgebung für die Lernfunktion nicht protokolliert bzw. die Angebotsunterdrückung in einer Sitzung nicht aktiviert werden soll. Mit diesem Parameter können Sie mehrere postEvent-Methoden erstellen, die verschiedene Antworttypen protokollieren, ohne das Lernen zu beeinflussen. Es ist nicht erforderlich, diesen Parameter für Ereignisse zu definieren, die einen Kontakt, eine Annahme oder eine Ablehnung protokollieren. Sie müssen diesen Parameter in Verbindung mit UACIResponseTypeCode verwenden. Wenn Sie UACILOGTOLEARNING nicht definieren, verwendet die Laufzeitumgebung den Standardwert 0 (sofern das Ereignis keinen Kontakt, keine Annahme und keine Ablehnung auslöst).
	UACIResponseTypeCode	Ein Wert, der einen Antworttypcode darstellt. Der Wert muss ein gültiger Eintrag in der UA_UsrResponseType-Tabelle sein

Funktion	Parameter	Beschreibung
Antwortverfolgung	UACIOfferTrackingCode	Der Verfahrenscode für das Angebot. Sie müssen diesen Parameter definieren, wenn das Ereignis in den Kontakt- oder Antwortverlauf protokolliert. Pro Ereignis können Sie immer nur einen Verfahrenscode übergeben. Wenn Sie den Verfahrenscode für einen Angebotskontakt nicht übergeben, protokolliert die Laufzeitumgebung jeweils einen Angebotskontakt für jedes Angebot in der zuletzt empfohlenen Angebotsliste. Wenn Sie den Verfahrenscode für eine Antwort nicht übergeben, gibt die Laufzeitumgebung einen Fehler zurück. Wenn Sie die sitzungsübergreifende Antwortverfolgung konfigurieren, können Sie mit dem UACIOfferTrackingcodeType-Parameter definieren, welcher Verfolgungscodetyp anstelle des Verfahrenscodes verwendet werden soll.
Sitzungsübergreifende Antwortverfolgung	UACIOfferTrackingCodeType	Eine Zahl, die den Verfolgungscodetyp definiert. 1 ist der Standardverfahrenscode und 2 ist der Angebotscode. Alle Codes müssen gültige Einträge in der UACI_TrackingType-Tabelle sein. Sie können dieser Tabelle weitere und angepasste Codes hinzufügen.
Ausführung bestimmter Ablaufdiagramme	UACIExecuteFlowchartByName	Wenn Sie diesen Parameter für eine Methode definieren, die eine Segmentierung oder erneute Segmentierung auslöst (startSession, setAudience oder postEvent), führt Interact nicht alle Ablaufdiagramme für die aktuelle Zielgruppenebene, sondern nur die angegebenen Ablaufdiagramme aus. Um eine Liste mit Ablaufdiagrammen anzugeben, trennen Sie diese durch eine vertikale Linie (|).

Reservierte Laufzeitumgebungsparameter

Die folgenden reservierten Parameter werden von der Laufzeitumgebung verwendet. Verwenden Sie diese Namen nicht für Ihre Ereignisparameter.

- UACIEventID
- UACIEventName
- UACIInteractiveChannelID
- UACIInteractiveChannelName
- UACIInteractionPointID
- UACIInteractionPointName
- UACISessionID

Informationen zur Klasse AdvisoryMessage

Die Klasse advisoryMessage enthält Methoden, die das Empfehlungsnachrichtenobjekt definieren. Das Empfehlungsnachrichtenobjekt ist im Antwortobjekt enthalten. Jede Methode in der InteractAPI gibt ein Antwortobjekt zurück. (Ausgenommen die Methode executeBatch, die ein batchResponse-Objekt zurückgibt.)

Wenn es einen Fehler oder eine Warnung gibt, füllt der Interact-Server das Empfehlungsnachrichtenobjekt auf. Das Empfehlungsnachrichtenobjekt enthält die folgenden Attribute:

- **DetailMessage** - eine ausführliche Beschreibung der Empfehlungsnachricht. Dieses Attribut ist möglicherweise nicht für alle Empfehlungsnachrichten verfügbar. Wenn es verfügbar ist, ist die DetailMessage möglicherweise nicht lokalisiert.
- **Message** - eine Kurzbeschreibung der Empfehlungsnachricht.
- **MessageCode** - eine Codenummer für die Empfehlungsnachricht.
- **StatusLevel** - eine Codenummer für die Dringlichkeit der Empfehlungsnachricht.

Sie rufen die advisoryMessage-Objekte mithilfe der Methode `getAdvisoryMessages` ab.

getDetailMessage

Die `getDetailMessage`-Methode gibt die ausführliche und detaillierte Beschreibung eines Advisory Message-Objekts zurück. Nicht alle Nachrichten haben eine detaillierte Nachricht.

```
getDetailMessage()
```

Rückgabewert

Das Advisory Message-Objekt gibt eine Zeichenfolge zurück.

Beispiel

```
// For any non-successes, there should be advisory messages explaining why
if(response.getStatusCode() != Response.STATUS_SUCCESS)
{
 for(AdvisoryMessage msg : response.getAdvisoryMessages())
 {
  System.out.println(msg.getMessage());
  // Some advisory messages may have additional detail:
  System.out.println(msg.getDetailMessage());
 }
}
```

getMessage

Die `getMessage`-Methode gibt die Kurzbeschreibung eines Advisory Message-Objekts zurück.

```
getMessage()
```

Rückgabewert

Das Advisory Message-Objekt gibt eine Zeichenfolge zurück.

Beispiel

Die folgende Methode druckt die Nachricht und die detaillierte Nachricht eines AdvisoryMessage-Objekts aus.

```
// For any non-successes, there should be advisory messages explaining why
if(response.getStatusCode() != Response.STATUS_SUCCESS)
{
 for(AdvisoryMessage msg : response.getAdvisoryMessages())
 {
  System.out.println(msg.getMessage());
```

```
  // Some advisory messages may have additional detail:
  System.out.println(msg.getDetailMessage());
 }
}
```

getMessageCode

Die getMessageCode-Methode gibt den internen Fehlercode zurück, der einem Advisory Message-Objekt zugeordnet ist, wenn die Statusebene 2 ist (STATUS-_LEVEL_ERROR).

```
getMessageCode()
```

Rückgabewert

Das AdvisoryMessage-Objekt gibt eine Ganzzahl zurück.

Beispiel

Die folgende Methode druckt den Nachrichtencode eines AdvisoryMessage-Objekts aus.

```
public static void printMessageCodeOfWarningOrError(String command,AdvisoryMessage[] messages)
    {
        System.out.println("Calling "+command);
        for(AdvisoryMessage msg : messages)
        {
            System.out.println(msg.getMessageCode());
```

getStatusLevel

Die getStatusLevel-Methode gibt die Statusebene eines Advisory Message-Objekts zurück.

```
getStatusLevel()
```

Rückgabewert

Das Advisory Message-Objekt gibt eine Ganzzahl zurück.

- 0 - STATUS_LEVEL_SUCCESS - Die aufgerufene Methode wurde ohne Fehler ausgeführt.
- 1 - STATUS_LEVEL_WARNING - Die aufgerufene Methode wurde mit mindestens einer Warnung (aber ohne Fehler) ausgeführt.
- 2 - STATUS_LEVEL_ERROR - Die aufgerufene Methode wurde nicht erfolgreich ausgeführt und weist mindestens einen Fehler auf.

Beispiel

Die folgende Methode druckt die Statusebene eines AdvisoryMessage-Objekts aus.

```
public static void printMessageCodeOfWarningOrError(String command,AdvisoryMessage[] messages)
    {
        System.out.println("Calling "+command);
        for(AdvisoryMessage msg : messages)
        {
            System.out.println(msg.getStatusLevel());
```

Informationen zur Klasse AdvisoryMessageCode

Die Klasse advisoryMessageCode enthält Methoden, die die Empfehlungsnachrichtencodes definieren. Sie rufen die Empfehlungsnachrichtencodes mit der Methode getMessageCode ab.

Codes von Empfehlungsnachrichten

Sie rufen die Empfehlungsnachrichtencodes mit der Methode `getMessageCode` ab.

In dieser Tabelle werden die Codes von Empfehlungsnachrichten aufgelistet und beschrieben:

Code	Nachrichtentext	Beschreibung
1	INVALID_SESSION_ID	Die Sitzungs-ID verweist nicht auf eine gültige Sitzung.
2	ERROR_TRYING_TO_ABORT_ SEGMENTATION	Beim Versuch, die Segmentierung während eines `endSession`-Aufrufs abzubrechen, ist ein Fehler aufgetreten.
3	INVALID_INTERACTIVE_CHANNEL	Das für den interaktiven Kanal übergebene Argument verweist nicht auf einen gültigen interaktiven Kanal.
4	INVALID_EVENT_NAME	Das für das Ereignis übergebene Argument verweist nicht auf ein gültiges Ereignis für den aktuellen interaktiven Kanal.
5	INVALID_INTERACTION_POINT	Das für den Interaktionspunkt übergebene Argument verweist nicht auf einen gültigen Interaktionspunkt für den aktuellen interaktiven Kanal.
6	ERROR_WHILE_MAKING_INITIAL_ SEGMENTATION_REQUEST	Beim Einreichen einer Segmentierungsanfrage ist ein Fehler aufgetreten.
7	SEGMENTATION_RUN_FAILED	Die Segmentierung wurde nur teilweise ausgeführt und ist dann fehlgeschlagen.
8	PROFILE_LOAD_FAILED	Das Laden der Profil- oder Dimensionstabellen ist fehlgeschlagen.
9	OFFER_SUPPRESSION_LOAD_FAILED	Das Laden der Angebotsunterdrückungstabelle ist fehlgeschlagen.
10	COMMAND_METHOD_ UNRECOGNIZED	Für einen Befehl innerhalb eines `executeBatch`-Aufrufs wurde eine ungültige Befehlsmethode angegeben.
11	ERROR_TRYING_TO_POST_EVENT _PARAMETERS	Bei der Übergabe der Ereignisparameter ist ein Fehler aufgetreten.
12	LOG_SYSTEM_EVENT_EXCEPTION	Ausnahmebedingung bei der Übergabe des Systemereignisses (Sitzung beenden, Angebot erhalten, Profil erhalten, Zielgruppe einstellen, Debuggen einstellen oder Sitzung starten) zur Protokollierung.
13	LOG_USER_EVENT_EXCEPTION	Ausnahmebedingung bei der Übergabe des Benutzerereignisses zur Protokollierung.
14	ERROR_TRYING_TO_LOOK_UP_EVENT	Bei der Suche nach dem Ereignisnamen ist ein Fehler aufgetreten.
15	ERROR_TRYING_TO_LOOK_UP _INTERACTIVE_CHANNEL	Bei der Suche nach dem Namen des interaktiven Kanals ist ein Fehler aufgetreten.

Code	Nachrichtentext	Beschreibung
16	ERROR_TRYING_TO_LOOK_UP _INTERACTION_POINT	Bei der Suche nach dem Namen des Interaktionspunkts ist ein Fehler aufgetreten.
17	RUNTIME_EXCEPTION_ ENCOUNTERED	Es ist eine nicht erwartete Laufzeitausnahmebedingung aufgetreten.
18	ERROR_TRYING_TO_EXECUTE_ ASSOCIATED_ACTION	Fehler beim Ausführen der zugehörigen Aktion (erneute Segmentierung auslösen, Angebotskontakt protokollieren, Angebotsannahme protokollieren oder Angebotsablehnung protokollieren).
19	ERROR_TRYING_RUN_FLOWCHART	Bei der Ausführung des Ablaufdiagramms ist ein Fehler aufgetreten.
20	FLOWCHART_FAILED	Eine Ablaufdiagrammausführung ist fehlgeschlagen.
21	FLOWCHART_ABORTED	Eine Ablaufdiagrammausführung wurde abgebrochen.
22	FLOWCHART_NEVER_RUN	Ein angegebenes Ablaufdiagramm wurde nie ausgeführt.
23	FLOWCHART_STILL_RUNNING	Ein Ablaufdiagramm wird immer noch ausgeführt.
24	ERROR_WHILE_READING_ PARAMETERS	Beim Lesen von Parametern ist ein Fehler aufgetreten.
25	ERROR_WHILE_LOADING_ RECOMMENDED_OFFERS	Fehler beim Laden der empfohlenen Angebote
26	ERROR_WHILE_LOGGING_DEFAULT _TEXT_STATISTICS	Fehler beim Protokollieren von Standardtextstatistiken (Anzahl, wie oft die Standardzeichenfolge für den Interaktionspunkt angezeigt wurde).
27	SCORE_OVERRIDE_LOAD_FAILED	Die Bewertungsüberschreibungstabelle konnte nicht geladen werden.
28	NULL_AUDIENCE_ID	Die Zielgruppen-ID ist leer.
29	UNRECOGNIZED_AUDIENCE_LEVEL	Eine nicht erkannte Zielgruppenebene wurde angegeben.
30	MISSING_AUDIENCE_FIELD	Ein Zielgruppenfeld fehlt.
31	INVALID_AUDIENCE_FIELD_TYPE	Ein ungültiger Zielgruppenfeldtyp wurde angegeben.
32	UNSUPPORTED_AUDIENCE_FIELD _TYPE	Nicht unterstützter Zielgruppenfeldtyp
33	TIMEOUT_REACHED_ON_GET_ OFFERS_CALL	Das Zeitlimit für den Aufruf `getOffers` wurde ohne Rückgabe von Angeboten erreicht.
34	INTERACT_INITIALIZATION_NOT_ COMPLETED_SUCCESSFULLY	Die Initialisierung des Laufzeitservers wurde nicht erfolgreich abgeschlossen.
35	SESSION_ID_UNDEFINED	Die Sitzungs-ID ist nicht definiert.
36	INVALID_NUMBER_OF_OFFERS_ REQUESTED	Eine ungültige Anzahl an Angeboten wurde angefordert.
37	NO_SESSION_EXIST_BUT_WILL_ CREATE_NEW_ONE	Da keine Sitzung vorhanden war, wurde eine Sitzung erstellt.

Code	Nachrichtentext	Beschreibung
38	AUDIENCE_ID_NOT_FOUND_IN_ PROFILE_TABLE	Die angegebene Zielgruppen-ID ist nicht in der Profiltabelle enthalten.
39	LOG_CUSTOM_LOGGER_EVENT_ EXCEPTION	Bei der Übergabe eines Datenereignisses zur benutzerdefinierten Protokollierung ist eine Ausnahmebedingung aufgetreten.
40	SPECIFIED_FLOWCHART_FOR_ EXECUTION_DOES_NOT_EXIST	Das angegebene Ablaufdiagramm kann nicht ausgeführt werden, da es nicht vorhanden ist.
41	AUDIENCE_NOT_DEFINED_IN_ CONFIGURATION	Die angegebene Zielgruppe ist nicht in der Konfiguration definiert.

Informationen zur Klasse BatchResponse

Die Klasse `BatchResponse` enthält Methoden, die die Ergebnisse der Methode `executeBatch` definieren.

Das Batch-Antwortobjekt enthält die folgenden Attribute:

- **BatchStatusCode** - Der höchste Statuscodewert für alle Antworten, die von der Methode `executeBatch` angefordert werden.
- **Responses** - Ein Array der Antwortobjekte, die von der Methode `executeBatch` angefordert werden.

getBatchStatusCode

Die `getBatchStatusCode`-Methode gibt den höchsten Statuscode aus dem Array von Befehlen zurück, die die `executeBatch`-Methode ausgeführt hat.

```
getBatchStatusCode()
```

Rückgabewert

Die `getBatchStatusCode`-Methode gibt eine Ganzzahl zurück.

- 0 - STATUS_SUCCESS - Die aufgerufene Methode wurde ohne Fehler ausgeführt.
- 1 - STATUS_WARNING - Die aufgerufene Methode wurde mit mindestens einer Warnung (aber ohne Fehler) ausgeführt.
- 2 - STATUS_ERROR - Die aufgerufene Methode wurde nicht erfolgreich ausgeführt und weist mindestens einen Fehler auf.

Beispiel

Das folgende Codebeispiel zeigt ein Beispiel zum Abrufen von `BatchStatusCode`.

```
// Top level status code is a short cut to determine if there are any
// non-successes in the array of Response objects
if(batchResponse.getBatchStatusCode() == Response.STATUS_SUCCESS)
{
  System.out.println("ExecuteBatch ran perfectly!");
}
else if(batchResponse.getBatchStatusCode() == Response.STATUS_WARNING)
{
  System.out.println("ExecuteBatch call processed with at least one warning");
}
else
{
  System.out.println("ExecuteBatch call processed with at least one error");
```

```
}

// Iterate through the array, and print out the message for any non-successes
for(Response response : batchResponse.getResponses())
{
  if(response.getStatusCode()!=Response.STATUS_SUCCESS)
  {
      printDetailMessageOfWarningOrError("executeBatchCommand",
   response.getAdvisoryMessages());
  }
}
```

getResponses

Die `getResponses`-Methode gibt das Array von Antwortobjekten zurück, die dem Array von Befehlen entsprechen, die die `executeBatch`-Methode ausgeführt hat.

```
getResponses()
```

Rückgabewert

Die `getResponses`-Methode gibt ein Array von `Response`-Objekten zurück.

Beispiel

Das folgende Beispiel wählt alle Antworten aus und druckt alle Advisory Messages, wenn der Befehl nicht erfolgreich war.

```
for(Response response : batchResponse.getResponses())
{
    if(response.getStatusCode()!=Response.STATUS_SUCCESS)
    {
        printDetailMessageOfWarningOrError("executeBatchCommand",
   response.getAdvisoryMessages());
    }
}
```

Informationen zur Command-Benutzeroberfläche

Die Methode `executeBatch` erfordert, dass Sie ein Array von Objekten übergeben, das die Command-Benutzeroberfläche implementiert. Sie sollten die Standardimplementierung `CommandImpl` verwenden, um die Befehlsobjekte zu übergeben.

Die folgende Tabelle listet den Befehl, die Methode der Interact-API-Klasse, die der Befehl darstellt, und die Command-Benutzeroberflächenmethoden auf, die Sie für jeden Befehl verwenden müssen. Sie müssen keine Sitzungs-ID einbeziehen, weil die Methode `executeBatch` bereits die Sitzungs-ID enthält.

Befehl	Interact-API-Methode	Command-Benutzeroberflächenmethode
COMMAND_ENDSESSION	endSession	Keiner.
COMMAND_GETOFFERS	getOffers	• setInteractionPoint • setNumberRequested
COMMAND_GETPROFILE	getProfile	Keiner.
COMMAND_GETVERSION	getVersion	Keiner.
COMMAND_POSTEVENT	postEvent	• setEvent • setEventParameters

Befehl	Interact-API-Methode	Command-Benutzeroberflächenmethode
COMMAND_SETAUDIENCE	setAudience	• setAudienceID • setAudienceLevel • setEventParameters
COMMAND_SETDEBUG	setDebug	setDebug
COMMAND_STARTSESSION	startSession	• setAudienceID • setAudienceLevel • setDebug • setEventParameters • setInteractiveChannel • setRelyOnExistingSession

setAudienceID

Die setAudienceID-Methode definiert die AudienceID für die Befehle setAudience und startSession.

setAudienceID(*audienceID*)

- **audienceID** - ein Array von NameValuePair-Objekten, die die AudienceID definieren.

Rückgabewert

Keiner.

Beispiel

Das folgende Beispiel ist ein Auszug aus einer executeBatch-Methode, die startSession und setAudience aufruft.

```
NameValuePair custId = new NameValuePairImpl();
custId.setName("CustomerId");
custId.setValueAsNumeric(1.0);
custId.setValueDataType(NameValuePair.DATA_TYPE_NUMERIC);
NameValuePair[] initialAudienceId = { custId };
. . .
Command startSessionCommand = new CommandImpl();
startSessionCommand.setAudienceID(initialAudienceId);
. . .
Command setAudienceCommand = new CommandImpl();
setAudienceCommand.setAudienceID(newAudienceId);
. . .
/** Build command array */
Command[] commands =
    {
        startSessionCommand,
        setAudienceCommand,
    };
/** Make the call */
    BatchResponse batchResponse = api.executeBatch(sessionId, commands);

/** Process the response appropriately */
    processExecuteBatchResponse(batchResponse);
```

setAudienceLevel

Die setAudienceLevel-Methode definiert die Zielgruppenebene für die Befehle setAudience und startSession.

setAudienceLevel(*audienceLevel*)

- *audienceLevel* - eine Zeichenfolge, die die Zielgruppenebene enthält.

  Wichtig: Der *audienceLevel*-Name muss exakt mit dem in Campaign definierten Namen der Zielgruppenebene übereinstimmen. Hierbei muss auf die Groß-/Kleinschreibung geachtet werden.

Rückgabewert

Keiner.

Beispiel

Das folgende Beispiel ist ein Auszug aus einer executeBatch-Methode, die startSession und setAudience aufruft.

```
String audienceLevel="Customer";
. . .
Command startSessionCommand = new CommandImpl();
startSessionCommand.setAudienceID(initialAudienceId);
. . .
Command setAudienceCommand = new CommandImpl();
setAudienceCommand.setAudienceLevel(audienceLevel);
. . .
/** Build command array */
Command[] commands =
    {
        startSessionCommand,
        setAudienceCommand,
    };
/** Make the call */
    BatchResponse batchResponse = api.executeBatch(sessionId, commands);

/** Process the response appropriately */
    processExecuteBatchResponse(batchResponse);
```

setDebug

Die setDebug-Methode definiert die Fehlerbehebungsstufe für den startSession-Befehl.

setDebug(*debug*)

Wenn der Wert wahr ist, protokolliert der Laufzeitserver die Daten zur Fehlerbehebung im Protokoll des Laufzeitservers. Wenn der Wert falsch ist, protokolliert der Laufzeitserver keine Daten zur Fehlerbehebung. Das Debugflag wird für jede Sitzung individuell gesetzt. Somit können Sie die Daten zur Fehlerbehebung für einzelne Laufzeitsitzungen verfolgen.

- **debug** - ein boolescher Wert (true oder false).

Rückgabewert

Keiner.

Beispiel

Das folgende Beispiel ist ein Auszug aus einer executeBatch-Methode, die startSession und setDebug aufruft.

```
boolean initialDebugFlag=true;
boolean newDebugFlag=false;
. . .
/* build the startSession command */
Command startSessionCommand = new CommandImpl();
startSessionCommand.setDebug(initialDebugFlag);
. . .

/* build the setDebug command */
Command setDebugCommand = new CommandImpl();
setDebugCommand.setMethodIdentifier(Command.COMMAND_SETDEBUG);
setDebugCommand.setDebug(newDebugFlag);

/** Build command array */
Command[] commands =
    {
        startSessionCommand,
        setDebugCommand,
    };
/** Make the call */
    BatchResponse batchResponse = api.executeBatch(sessionId, commands);

/** Process the response appropriately */
    processExecuteBatchResponse(batchResponse);
```

setEvent

Die setEvent-Methode definiert den Namen des Ereignisses, das der postEvent-Befehl verwendet.

setEvent(*Ereignis*)

- **event** - Eine Zeichenfolge, die den Ereignisnamen enthält.

Wichtig: Der *event*-Name muss exakt mit dem im interaktiven Kanal definierten Namen übereinstimmen. Hierbei muss auf die Groß-/Kleinschreibung geachtet werden.

Rückgabewert

Keiner.

Beispiel

Das folgende Beispiel ist ein Auszug aus einer executeBatch-Methode, die postEvent aufruft.

```
String eventName = "SearchExecution";

Command postEventCommand = new CommandImpl();
postEventCommand.setMethodIdentifier(Command.COMMAND_POSTEVENT);
postEventCommand.setEventParameters(postEventParameters);
postEventCommand.setEvent(eventName);
```

setEventParameters

Die setEventParameters-Methode definiert die Ereignisparameter, die der postEvent-Befehl verwendet. Diese Werte werden in den Sitzungsdaten gespeichert.

setEventParameters(*eventParameters*)

- **eventParameters** - ein Array von NameValuePair-Objekten, die die Ereignisparameter definieren.

Wenn das Ereignis zum Beispiel ein Angebot im Kontaktprotokoll protokolliert, müssen Sie den Verfahrenscode des Angebots einschließen.

Rückgabewert

Keiner.

Beispiel

Das folgende Beispiel ist ein Auszug aus einer executeBatch-Methode, die postEvent aufruft.

```
NameValuePair parmB1 = new NameValuePairImpl();
parmB1.setName("SearchString");
parmB1.setValueAsString("mortgage");
parmB1.setValueDataType(NameValuePair.DATA_TYPE_STRING);

NameValuePair parmB2 = new NameValuePairImpl();
parmB2.setName("TimeStamp");
parmB2.setValueAsDate(new Date());
parmB2.setValueDataType(NameValuePair.DATA_TYPE_DATETIME);

NameValuePair parmB3 = new NameValuePairImpl();
parmB3.setName("Browser");
parmB3.setValueAsString("IE6");
parmB3.setValueDataType(NameValuePair.DATA_TYPE_STRING);

NameValuePair parmB4 = new NameValuePairImpl();
parmB4.setName("FlashEnabled");
parmB4.setValueAsNumeric(1.0);
parmB4.setValueDataType(NameValuePair.DATA_TYPE_NUMERIC);

NameValuePair parmB5 = new NameValuePairImpl();
parmB5.setName("TxAcctValueChange");
parmB5.setValueAsNumeric(0.0);
parmB5.setValueDataType(NameValuePair.DATA_TYPE_NUMERIC);

NameValuePair parmB6 = new NameValuePairImpl();
parmB6.setName("PageTopic");
parmB6.setValueAsString("");
parmB6.setValueDataType(NameValuePair.DATA_TYPE_STRING);

NameValuePair[] postEventParameters = { parmB1,
 parmB2,
 parmB3,
 parmB4,
 parmB5,
 parmB6
  };
. . .
Command postEventCommand = new CommandImpl();
postEventCommand.setMethodIdentifier(Command.COMMAND_POSTEVENT);
postEventCommand.setEventParameters(postEventParameters);
postEventCommand.setEvent(eventName);
```

setGetOfferRequests

Die **setGetOfferRequests**-Methode legt den Parameter zum Abrufen der Angebote fest, die der getOffersForMultipleInteractionPoints-Befehl verwendet.

```
setGetOfferRequests(numberRequested)
```

- **numberRequested** - ein Array von `GetOfferRequest`-Objekten, die den Parameter zum Abrufen von Angeboten definieren.

Rückgabewert

Keiner.

Beispiel

Das folgende Beispiel ist ein Auszug aus einer `GetOfferRequest`-Methode, die `setGetOfferRequests` aufruft.

```
GetOfferRequest request1 = new GetOfferRequest(5, GetOfferRequest.NO_DUPLICATION);
     request1.setIpName("IP1");
     OfferAttributeRequirements offerAttributes1 = new OfferAttributeRequirements();
     NameValuePairImpl attr1 = new NameValuePairImpl("attr1",
          NameValuePair.DATA_TYPE_NUMERIC, 1);
     NameValuePairImpl attr2 = new NameValuePairImpl("attr2",
          NameValuePair.DATA_TYPE_STRING, "value2");
     NameValuePairImpl attr3 = new NameValuePairImpl("attr3",
          NameValuePair.DATA_TYPE_STRING, "value3");
     NameValuePairImpl attr4 = new NameValuePairImpl("attr4",
          NameValuePair.DATA_TYPE_NUMERIC, 4);
     offerAttributes1.setNumberRequested(5);
     offerAttributes1.setAttributes(new NameValuePairImpl[] {attr1, attr2});
     OfferAttributeRequirements childAttributes1 = new OfferAttributeRequirements();
     childAttributes1.setNumberRequested(3);
     childAttributes1.setAttributes(new NameValuePairImpl[] {attr3});
     OfferAttributeRequirements childAttributes2 = new OfferAttributeRequirements();
     childAttributes2.setNumberRequested(3);
     childAttributes2.setAttributes(new NameValuePairImpl[] {attr4});
     offerAttributes1.setChildRequirements(Arrays.asList(childAttributes1,
          childAttributes2));
     request1.setOfferAttributes(offerAttributes1);

     GetOfferRequest request2 = new GetOfferRequest(3, GetOfferRequest.ALLOW_DUPLICATION);
     request2.setIpName("IP2");
     OfferAttributeRequirements offerAttributes2 = new OfferAttributeRequirements();
     offerAttributes2.setNumberRequested(3);
     offerAttributes2.setAttributes(new NameValuePairImpl[] {new NameValuePairImpl("attr5",
          NameValuePair.DATA_TYPE_STRING, "value5")});
     request2.setOfferAttributes(offerAttributes2);

     GetOfferRequest request3 = new GetOfferRequest(2, GetOfferRequest.NO_DUPLICATION);
     request3.setIpName("IP3");
     request3.setOfferAttributes(null);

     Command getOffersMultiIPCmd = new CommandImpl();
     getOffersMultiIPCmd.setGetOfferRequests(new GetOfferRequest[] {request1,
          request2, request3});
```

setInteractiveChannel

Die `setInteractiveChannel`-Methode definiert den Namen des interaktiven Kanals, den der `startSession`-Befehl verwendet.

`setInteractiveChannel(`*`interactiveChannel`*`)`

- **interactiveChannel** - eine Zeichenfolge, die den Namen des interaktiven Kanals enthält.

Wichtig: Der *interactiveChannel*-Name muss exakt mit dem in Campaign definierten Namen des interaktiven Kanals übereinstimmen. Hierbei muss auf die Groß-/Kleinschreibung geachtet werden.

Rückgabewert

Keiner.

Beispiel

Das folgende Beispiel ist ein Auszug aus einer executeBatch-Methode, die startSession aufruft.

```
String interactiveChannel="Accounts Website";
. . .
Command startSessionCommand = new CommandImpl();
startSessionCommand.setInteractiveChannel(interactiveChannel);
```

setInteractionPoint

Die setInteractionPoint-Methode definiert den Namen des Interaktionspunkts, den die Befehle getOffers und postEvent verwenden.

setInteractionPoint(*interactionPoint*)

- **interactionPoint** - eine Zeichenfolge, die den Namen des Interaktionspunkts enthält.

Wichtig: Der *interactionPoint*-Name muss exakt mit dem im interaktiven Kanal definierten Interaktionspunkt übereinstimmen. Hierbei muss auf die Groß-/Kleinschreibung geachtet werden.

Rückgabewert

Keiner.

Beispiel

Das folgende Beispiel ist ein Auszug aus einer executeBatch-Methode, die getOffers aufruft.

```
String interactionPoint = "Overview Page Banner 1";
int numberRequested=1;

Command getOffersCommand = new CommandImpl();
getOffersCommand.setMethodIdentifier(Command.COMMAND_GETOFFERS);
getOffersCommand.setInteractionPoint(interactionPoint);
getOffersCommand.setNumberRequested(numberRequested);
```

setMethodIdentifier

Die setMethodIdentifier-Methode definiert den Typ des Befehls, der im Befehlsobjekt enthalten ist.

setMethodIdentifier(*methodIdentifier*)

- **methodIdentifier** - eine Zeichenfolge, die den Typ des Befehls enthält.

  Die gültigen Werte sind:
  - **COMMAND_ENDSESSION** - stellt die endSession-Methode dar.
  - **COMMAND_GETOFFERS** - stellt die getOffers-Methode dar.
  - **COMMAND_GETPROFILE** - stellt die getProfile-Methode dar.
  - **COMMAND_GETVERSION** - stellt die getVersion-Methode dar.
  - **COMMAND_POSTEVENT** - stellt die postEvent-Methode dar.
  - **COMMAND_SETAUDIENCE** - stellt die setAudience-Methode dar.
  - **COMMAND_SETDEBUG** - stellt die setDebug-Methode dar.

– **COMMAND_STARTSESSION** - stellt die startSession-Methode dar.

Rückgabewert

Keiner.

Beispiel

Das folgende Beispiel ist ein Auszug aus einer executeBatch-Methode, die getVersion und endSession aufruft.

```
Command getVersionCommand = new CommandImpl();
getVersionCommand.setMethodIdentifier(Command.COMMAND_GETVERSION);

Command endSessionCommand = new CommandImpl();
endSessionCommand.setMethodIdentifier(Command.COMMAND_ENDSESSION);

Command[] commands =
{
    getVersionCommand,
    endSessionCommand
};
```

setNumberRequested

Die setNumberRequested-Methode definiert die Anzahl der Angebote, die der getOffers-Befehl anfordert.

```
setNumberRequested(numberRequested)
```

- **numberRequested** - eine Ganzzahl, die die Anzahl an Angeboten definiert, die der getOffers-Befehl anfordert.

Rückgabewert

Keiner.

Beispiel

Das folgende Beispiel ist ein Auszug aus einer executeBatch-Methode, die getOffers aufruft.

```
String interactionPoint = "Overview Page Banner 1";
int numberRequested=1;

Command getOffersCommand = new CommandImpl();
getOffersCommand.setMethodIdentifier(Command.COMMAND_GETOFFERS);
getOffersCommand.setInteractionPoint(interactionPoint);
getOffersCommand.setNumberRequested(numberRequested);
```

setRelyOnExistingSession

Die setRelyOnExistingSession-Methode definiert eine boolesche Variable, die definiert, ob der startSession-Befehl eine vorhandene Sitzung verwendet oder nicht.

```
setRelyOnExistingSession(relyOnExistingSession)
```

Wenn der Wert true ist, muss die Sitzungs-ID für executeBatch mit einer vorhandenen Sitzungs-ID übereinstimmen. Wenn der Wert false ist, müssen Sie eine neue Sitzungs-ID mit der executeBatch-Methode angeben.

- **relyOnExistingSession** - ein boolescher Wert (true oder false).

Rückgabewert

Keiner.

Beispiel

Das folgende Beispiel ist ein Auszug aus einer executeBatch-Methode, die startSession aufruft.

```
boolean relyOnExistingSession=false;
. . .
Command startSessionCommand = new CommandImpl();
startSessionCommand.setRelyOnExistingSession(relyOnExistingSession);
```

Informationen zur NameValuePair-Benutzeroberfläche

Viele Methoden in der Interact-API geben entweder NameValuePair-Objekte zurück oder erfordern, dass Sie NameValuePair-Objekte als Argumente übergeben. Bei der Übergabe als Argumente in eine Methode sollten Sie die Standardimplementierung NameValuePairImpl verwenden.

getName

Die getName-Methode gibt die Namenskomponente eines NameValuePair-Objekts zurück.

```
getName()
```

Rückgabewert

Die getName-Methode gibt eine Zeichenfolge zurück.

Beispiel

Das folgende Beispiel ist ein Auszug aus einer Methode, die das Antwortobjekt für getProfile verarbeitet.

```
for(NameValuePair nvp : response.getProfileRecord())
{
    System.out.println("Name:"+nvp.getName());
}
```

getValueAsDate

Die getValueAsDate-Methode gibt den Wert eines NameValuePair-Objekts zurück.

```
getValueAsDate()
```

Verwenden Sie getValueDataType, bevor Sie getValueAsDate verwenden, um sicherzustellen, dass auf den richtigen Datentyp verwiesen wird.

Rückgabewert

Die getValueAsDate-Methode gibt ein Datum zurück.

Beispiel

Das folgende Beispiel ist ein Auszug aus einer Methode, die ein NameValuePair verarbeitet und den Wert ausdruckt, wenn es sich um ein Datum handelt.

```
if(nvp.getValueDataType().equals(NameValuePair.DATA_TYPE_DATE))
{
    System.out.println("Value:"+nvp.getValueAsDate());
}
```

getValueAsNumeric

Die getValueAsNumeric-Methode gibt den Wert eines NameValuePair-Objekts zurück.

```
getValueAsNumeric()
```

Verwenden Sie getValueDataType, bevor Sie getValueAsNumeric verwenden, um sicherzustellen, dass auf den richtigen Datentyp verwiesen wird.

Rückgabewert

Die getValueAsNumeric-Methode gibt ein Doppelzeichen zurück. Beispiel: Wenn Sie einen ursprünglich als Ganzzahl in der Profiltabelle gespeicherten Wert abrufen, gibt getValueAsNumeric ein Doppelzeichen zurück.

Beispiel

Das folgende Beispiel ist ein Auszug aus einer Methode, die ein NameValuePair verarbeitet und den Wert ausdruckt, wenn dieser numerisch ist.

```
if(nvp.getValueDataType().equals(NameValuePair.DATA_TYPE_NUMERIC))
{
    System.out.println("Value:"+nvp.getValueAsNumeric());
}
```

getValueAsString

Die getValueAsString-Methode gibt den Wert eines NameValuePair-Objekts zurück.

```
getValueAsString()
```

Verwenden Sie getValueDataType, bevor Sie getValueAsString verwenden, um sicherzustellen, dass auf den richtigen Datentyp verwiesen wird.

Rückgabewert

Die getValueAsString-Methode gibt eine Zeichenfolge zurück. Beispiel: Wenn Sie einen ursprünglich als char, varchar oder char[10] in der Profiltabelle gespeicherten Wert abrufen, gibt getValueAsString eine Zeichenfolge zurück.

Beispiel

Das folgende Beispiel ist ein Auszug aus einer Methode, die ein NameValuePair verarbeitet und den Wert ausdruckt, wenn es sich um eine Zeichenfolge handelt.

```
if(nvp.getValueDataType().equals(NameValuePair.DATA_TYPE_STRING))
{
    System.out.println("Value:"+nvp.getValueAsString());
}
```

getValueDataType

Die getValueDataType-Methode gibt den Datentyp eines NameValuePair-Objekts zurück.

```
getValueDataType()
```

Verwenden Sie getValueDataType, bevor Sie getValueAsDate, getValueAsNumeric oder getValueAsString verwenden, um sicherzustellen, dass auf den richtigen Datentyp verwiesen wird.

Rückgabewert

Die getValueDataType-Methode gibt eine Zeichenfolge zurück, die angibt, ob das NameValuePair ein Datum, eine Zahl oder eine Zeichenfolge enthält.

Die gültigen Werte sind:

- **DATA_TYPE_DATETIME** - ein Datum, das einen Wert mit Datum und Uhrzeit enthält.
- **DATA_TYPE_NUMERIC** - ein Doppelzeichen, das einen Zahlenwert enthält.
- **DATA_TYPE_STRING** - eine Zeichenfolge, die einen Textwert enthält.

Beispiel

Das folgende Beispiel ist ein Auszug aus einer Methode, die das Antwortobjekt aus einer getProfile-Methode verarbeitet.

```
for(NameValuePair nvp : response.getProfileRecord())
{
    System.out.println("Name:"+nvp.getName());
    if(nvp.getValueDataType().equals(NameValuePair.DATA_TYPE_DATETIME))
    {
        System.out.println("Value:"+nvp.getValueAsDate());
    }
    else if(nvp.getValueDataType().equals(NameValuePair.DATA_TYPE_NUMERIC))
    {
        System.out.println("Value:"+nvp.getValueAsNumeric());
    }
    else
    {
        System.out.println("Value:"+nvp.getValueAsString());
    }
}
```

setName

Die setName-Methode definiert die Namenskomponente eines NameValuePair-Objekts.

```
setName(name)
```

- **name** - eine Zeichenfolge, die die Namenskomponente eines NameValuePair-Objekts enthält.

Rückgabewert

Keiner.

Beispiel

Im folgenden Beispiel wird die Namenskomponente in einem NameValuePair definiert.

```
NameValuePair custId = new NameValuePairImpl();
custId.setName("CustomerId");
custId.setValueAsNumeric(1.0);
custId.setValueDataType(NameValuePair.DATA_TYPE_NUMERIC);
NameValuePair[] initialAudienceId = { custId };
```

setValueAsDate

Die setValueAsDate-Methode definiert den Wert eines NameValuePair-Objekts.

setValueAsDate(*valueAsDate*)

- **valueAsDate** - ein Datum, das den Wert mit Datum und Uhrzeit für das NameValuePair-Objekt enthält.

Rückgabewert

Keiner.

Beispiel

Im folgenden Beispiel wird die Wertkomponente in einem NameValuePair definiert, wenn der Wert ein Datum ist.

```
NameValuePair parm2 = new NameValuePairImpl();
parm2.setName("TimeStamp");
parm2.setValueAsDate(new Date());
parm2.setValueDataType(NameValuePair.DATA_TYPE_DATETIME);
```

setValueAsNumeric

Die setValueAsNumeric-Methode definiert den Wert eines NameValuePair-Objekts.

setValueAsNumeric(*valueAsNumeric*)

- **valueAsNumeric** - ein Doppelzeichen, das den numerischen Wert eines NameValuePair-Objekts enthält.

Rückgabewert

Keiner.

Beispiel

Im folgenden Beispiel wird die Wertkomponente in einem NameValuePair definiert, wenn es sich um einen numerischen Wert handelt.

```
NameValuePair parm4 = new NameValuePairImpl();
parm4.setName("FlashEnabled");
parm4.setValueAsNumeric(1.0);
parm4.setValueDataType(NameValuePair.DATA_TYPE_NUMERIC);
```

setValueAsString

Die setValueAsString-Methode definiert den Wert eines NameValuePair-Objekts.

setValueAsString(*valueAsString*)

- **valueAsString** - eine Zeichenfolge, die den Wert eines NameValuePair-Objekts enthält

Rückgabewert

Keiner.

Beispiel

Im folgenden Beispiel wird die Wertkomponente in einem NameValuePair definiert, wenn es sich um einen numerischen Wert handelt.

```
NameValuePair parm3 = new NameValuePairImpl();
parm3.setName("Browser");
parm3.setValueAsString("IE6");
parm3.setValueDataType(NameValuePair.DATA_TYPE_STRING);
```

setValueDataType

Die setValueDataType-Methode definiert den Datentyp eines NameValuePair-Objekts.

```
getValueDataType(valueDataType)
```

Die gültigen Werte sind:

- **DATA_TYPE_DATETIME** - ein Datum, das einen Wert mit Datum und Uhrzeit enthält.
- **DATA_TYPE_NUMERIC** - ein Doppelzeichen, das einen Zahlenwert enthält.
- **DATA_TYPE_STRING** - eine Zeichenfolge, die einen Textwert enthält.

Rückgabewert

Keiner.

Beispiel

Im folgenden Beispiel wird der Datentyp des Werts in einem NameValuePair festgelegt.

```
NameValuePair parm2 = new NameValuePairImpl();
parm2.setName("TimeStamp");
parm2.setValueAsDate(new Date());
parm2.setValueDataType(NameValuePair.DATA_TYPE_DATETIME);

NameValuePair parm3 = new NameValuePairImpl();
parm3.setName("Browser");
parm3.setValueAsString("IE6");
parm3.setValueDataType(NameValuePair.DATA_TYPE_STRING);

NameValuePair parm4 = new NameValuePairImpl();
parm4.setName("FlashEnabled");
parm4.setValueAsNumeric(1.0);
parm4.setValueDataType(NameValuePair.DATA_TYPE_NUMERIC);
```

Informationen zur Klasse Offer

Die Klasse Offer enthält Methoden, die ein Offer-Objekt definieren. Dieses Angebotsobjekt enthält viele der Eigenschaften eines Angebots in Campaign.

Das Angebotsobjekt enthält die folgenden Attribute:

- **AdditionalAttributes** - Name/Wert-Paare, die benutzerdefinierte Angebotsattribute enthalten, die Sie in Campaign definiert haben.
- **Description** - Die Beschreibung des Angebots.
- **EffectiveDate** - Das Wirksamkeitsdatum des Angebots.
- **ExpirationDate** - Das Ablaufdatum des Angebots.
- **OfferCode** - Der Angebotscode des Angebots.
- **OfferName** - Der Name des Angebots.
- **TreatmentCode** - Der Verfahrenscode des Angebots.

- **Bewertung** - Der Marketing-Score des Angebots oder die durch `ScoreOverrideTable` definierte Bewertung, wenn die Eigenschaft `enableScoreOverrideLookup` "true" ist.

getAdditionalAttributes

Die `getAdditionalAttributes`-Methode gibt die benutzerdefinierten Angebotsattribute zurück, die in Campaign definiert sind.

```
getAdditionalAttributes()
```

Rückgabewert

Die `getAdditionalAttributes`-Methode gibt ein Array aus NameValuePair-Objekten zurück.

Beispiel

Das folgende Beispiel zeigt, wie Sie alle zusätzlichen Attribute sortieren, das gültige Datum und das Ablaufdatum überprüfen und die weiteren Attribute ausdrucken können.

```
for(NameValuePair offerAttribute : offer.getAdditionalAttributes())
{
    // check to see if the effective date exists
    if(offerAttribute.getName().equalsIgnoreCase("effectiveDate"))
    {
        System.out.println("Found effective date");
    }
    // check to see if the expiration date exists
    else if(offerAttribute.getName().equalsIgnoreCase("expirationDate"))
    {
        System.out.println("Found expiration date");
    }
    printNameValuePair(offerAttribute);
    }
}
public static void printNameValuePair(NameValuePair nvp)
{
    // print out the name:
    System.out.println("Name:"+nvp.getName());

    // based on the datatype, call the appropriate method to get the value
    if(nvp.getValueDataType()==NameValuePair.DATA_TYPE_DATETIME)
        System.out.println("DateValue:"+nvp.getValueAsDate());
    else if(nvp.getValueDataType()==NameValuePair.DATA_TYPE_NUMERIC)
        System.out.println("NumericValue:"+nvp.getValueAsNumeric());
    else
        System.out.println("StringValue:"+nvp.getValueAsString());
}
```

getDescription

Die `getDescription`-Methode gibt die Beschreibung des in Campaign definierten Angebots zurück.

```
getDescription()
```

Rückgabewert

Die `getDescription`-Methode gibt eine Zeichenfolge zurück.

Beispiel

Im folgenden Beispiel wird die Beschreibung eines Angebots ausgedruckt.

```
for(Offer offer : offerList.getRecommendedOffers())
{
    // print offer
     System.out.println("Offer Description:"+offer.getDescription());
}
```

getOfferCode

Die getOfferCode-Methode gibt den Angebotscode des in Campaign definierten Angebots zurück.

```
getOfferCode()
```

Rückgabewert

Die getOfferCode-Methode gibt ein Array aus Zeichenfolgen zurück, die den Angebotscode des Angebots enthalten.

Beispiel

Im folgenden Beispiel wird der Angebotscode eines Angebots ausgedruckt.

```
for(Offer offer : offerList.getRecommendedOffers())
{
    // print offer
   System.out.println("Offer Code:"+offer.getOfferCode());
}
```

getOfferName

Die getOfferName-Methode gibt den Namen des in Campaign definierten Angebots zurück.

```
getOfferName()
```

Rückgabewert

Die getOfferName-Methode gibt eine Zeichenfolge zurück.

Beispiel

Im folgenden Beispiel wird der Name eines Angebots gedruckt.

```
for(Offer offer : offerList.getRecommendedOffers())
{
// print offer
System.out.println("Offer Name:"+offer.getOfferName());
}
```

getScore

Die Methode getScore gibt eine Bewertung zurück, die auf den von Ihnen konfigurierten Angeboten basiert.

```
getScore()
```

Die getScore-Methode gibt eine der folgenden Optionen zurück:

- Wenn die Standardangebotstabelle, die Tabelle für die Bewertungsüberschreibung oder das integrierte Lernmodul nicht aktiviert ist, gibt diese Methode den auf der Registerkarte "Interaktionsstrategie" definierten Marketing-Score des Angebots zurück.
- Wenn Sie die Tabelle mit den Standardangeboten oder die Tabelle für die Bewertungsüberschreibung, nicht aber das integrierte Lernmodul aktiviert haben, gibt diese Methode die Bewertung des Angebots zurück, wie durch den Vorrang von Bedingungen zwischen der Tabelle mit den Standardangeboten, der Bewertung des Marketiers und der Tabelle für die Bewertungsüberschreibung definiert.
- Wenn Sie das integrierte Lernmodul aktiviert haben, gibt diese Methode die zum Sortieren der Angebote verwendete endgültige Bewertung zurück.

Rückgabewert

Die Methode `getScore` gibt eine Ganzzahl zurück, die die Bewertung des Angebots darstellt.

Beispiel

Im folgenden Beispiel wird die Punktzahl eines Angebots ausgedruckt.

```
for(Offer offer : offerList.getRecommendedOffers())
{
// print offer
System.out.println("Offer Score:"+offer.getOfferScore());
}
```

getTreatmentCode

Die `getTreatmentCode`-Methode gibt den Verfahrenscode des Angebots zurück, wie in Campaign definiert.

```
getTreatmentCode()
```

Da Campaign diesen Verfahrenscode zum Identifizieren der Instanz des erstellten Angebots verwendet, muss dieser Code als ein Ereignisparameter zurückgegeben werden, wenn die `postEvent`-Methode verwendet wird, um ein Kontakt-, Annahme- oder Ablehnungsereignis des Angebots zu protokollieren. Wenn Sie die Annahme oder Ablehnung eines Angebots protokollieren, müssen Sie den Namenswert im NameValuePair, das den Verfahrenscode darstellt, auf `UACIOfferTrackingCode` setzen.

Rückgabewert

Die `getTreatmentCode`-Methode gibt eine Zeichenfolge zurück.

Beispiel

Im folgenden Beispiel wird der Verfahrenscode eines Angebots ausgedruckt.

```
for(Offer offer : offerList.getRecommendedOffers())
{
    // print offer
    System.out.println("Offer Treatment Code:"+offer.getTreatmentCode());
}
```

Informationen zur Klasse OfferList

Die Klasse `OfferList` enthält Methoden, die die Ergebnisse der Methode `getOffers` definieren.

Das OfferList-Objekt enthält die folgenden Attribute:

- **DefaultString** - Die Standardzeichenfolge, die für den Interaktionspunkt im interaktiven Kanal definiert ist.
- **RecommendedOffers** - Ein Array der Angebotsobjekte, die von der Methode `getOffers` angefordert werden.

Die Klasse `OfferList` arbeitet mit Listen von Angeboten. Diese Klasse steht nicht mit Campaign-Angebotslisten in Beziehung.

getDefaultString

Die `getDefaultString`-Methode gibt die Standardzeichenfolge für den Interaktionspunkt zurück, wie in Campaign definiert.

```
getDefaultString()
```

Wenn das RecommendedOffers-Objekt leer ist, sollten Sie den Touchpoint so konfigurieren, dass er diese Zeichenfolge darstellt. Auf diese Weise stellen Sie sicher, dass ein Inhalt vorhanden ist. Interact füllt das DefaultString-Objekt nur aus, wenn das RecommendedOffers-Objekt leer ist.

Rückgabewert

Die `getDefaultString`-Methode gibt eine Zeichenfolge zurück.

Beispiel

Das folgende Beispiel ruft die Standardzeichenfolge ab, wenn das offerList-Objekt keine Angebote enthält.

```
OfferList offerList=response.getOfferList();
if(offerList.getRecommendedOffers() != null)
{
    for(Offer offer : offerList.getRecommendedOffers())
    {
  System.out.println("Offer Name:"+offer.getOfferName());
    }
}
else // count on the default Offer String
    System.out.println("Default offer:"+offerList.getDefaultString());
```

getRecommendedOffers

Die `getRecommendedOffers`-Methode gibt ein Array aus Offer-Objekten zurück, die von der `getOffers`-Methode angefordert wurden.

```
getRecommendedOffers()
```

Wenn die Antwort auf `getRecommendedOffer` leer ist, sollte der Touchpoint das `getDefaultString`-Ergebnis darstellen.

Rückgabewert

Die `getRecommendedOffers`-Methode gibt ein Offer-Objekt zurück.

Beispiel

Das folgende Beispiel verarbeitet das OfferList-Objekt und druckt die Namen aller empfohlenen Angebote aus.

```
OfferList offerList=response.getOfferList();
if(offerList.getRecommendedOffers() != null)
{
    for(Offer offer : offerList.getRecommendedOffers())
    {
        // print offer
        System.out.println("Offer Name:"+offer.getOfferName());
    }
}
else // count on the default Offer String
System.out.println("Default offer:"+offerList.getDefaultString());
```

Informationen zur Klasse Response

Die Klasse `Response` enthält Methoden, die die Ergebnisse einer der InteractAPI-Klassenmethoden definieren.

Das Antwortobjekt enthält die folgenden Attribute:

- **AdvisoryMessages** - ein Array von Empfehlungsnachrichten. Dieses Attribut wird nur aufgefüllt, wenn es während der Ausführung der Methode Warnungen oder Fehler gab.
- **ApiVersion** - eine Zeichenfolge mit der API-Version. Dieses Attribut wird durch die Methode `getVersion` aufgefüllt.
- **OfferList** - das OfferList-Objekt, das die von der Methode `getOffers` angeforderten Angebote enthält.
- **ProfileRecord** - ein Array von Name/Wert-Paaren, das Profildaten enthält. Dieses Attribut wird durch die Methode `getProfile` aufgefüllt.
- **SessionID** - eine Zeichenfolge, die die Sitzungs-ID definiert. Dies wird von allen `InteractAPI`-Klassenmethoden zurückgegeben.
- **StatusCode** - eine Zahl, die angibt, ob die Methode ohne Fehler, mit einer Warnung oder mit Fehlern ausgeführt wurde. Dies wird von allen `InteractAPI`-Klassenmethoden zurückgegeben.

getAdvisoryMessages

Die `getAdvisoryMessages`-Methode gibt ein Array aus Advisory Messages aus dem Response-Objekt zurück.

```
getAdvisoryMessages()
```

Rückgabewert

Die `getAdvisoryMessages`-Methode gibt ein Array aus Advisory Message-Objekten zurück.

Beispiel

Das folgende Beispiel ruft die AdvisoryMessage-Objekte aus einem Response-Objekt ab und durchläuft diese, um die Nachrichten auszudrucken.

```
AdvisoryMessage[] messages = response.getAdvisoryMessages();
    for(AdvisoryMessage msg : messages)
    {
```

```
        System.out.println(msg.getMessage());
        // Some advisory messages may have additional detail:
        System.out.println(msg.getDetailMessage());
    }
```

getApiVersion

Die getApiVersion-Methode gibt die API-Version eines Response-Objekts zurück.

```
getApiVersion()
```

Die getVersion-Methode füllt das ApiVersion-Attribut eines Response-Objekts aus.

Rückgabewert

Das Response-Objekt gibt eine Zeichenfolge zurück.

Beispiel

Das folgende Beispiel ist ein Auszug aus einer Methode, die das Antwortobjekt für getVersion verarbeitet.

```
if(response.getStatusCode() == Response.STATUS_SUCCESS)
{
    System.out.println("getVersion call processed with no warnings or errors");
    System.out.println("API Version:" + response.getApiVersion());
}
```

getOfferList

Die getOfferList-Methode gibt das OfferList-Objekt eines Response-Objekts zurück.

```
getOfferList()
```

Die getOffers-Methode füllt das OfferList-Objekt eines Response-Objekts aus.

Rückgabewert

Das Response-Objekt gibt ein OfferList-Objekt zurück.

Beispiel

Das folgende Beispiel ist ein Auszug aus einer Methode, die das Antwortobjekt für getOffers verarbeitet.

```
OfferList offerList=response.getOfferList();
if(offerList.getRecommendedOffers() != null)
{
    for(Offer offer : offerList.getRecommendedOffers())
    {
    // print offer
    System.out.println("Offer Name:"+offer.getOfferName());
    }
}
```

getAllOfferLists

Die getAllOfferLists-Methode gibt ein Array aus allen OfferLists eines Response-Objekts zurück.

```
getAllOfferLists()
```

Dies wird von der getOffersForMultipleInteractionPoints-Methode verwendet, die das OfferList-Arrayobjekt eines Response-Objekts ausfüllt.

Rückgabewert

Das Response-Objekt gibt ein OfferList-Array zurück.

Beispiel

Das folgende Beispiel ist ein Auszug aus einer Methode, die das Antwortobjekt für getOffers verarbeitet.

```
OfferList[] allOfferLists = response.getAllOfferLists();
 if (allOfferLists != null) {
  for (OfferList ol : allOfferLists) {
   System.out.println("Die folgenden Angebote werden für den Interaktionspunkt
                 geliefert "
     + ol.getInteractionPointName() + ":");
   for (Offer o : ol.getRecommendedOffers()) {
    System.out.println(o.getOfferName());
   }
  }
 }
```

getProfileRecord

Die getProfileRecord-Methode gibt die Profileinträge für die aktuelle Sitzung als ein Array aus NameValuePair-Objekten zurück. Diese Profileinträge beinhalten auch alle eventParameters, die zuvor in der Laufzeitsitzung hinzugefügt wurden.

```
getProfileRecord()
```

Die getProfile-Methode füllt die NameValuePair-Objekte für einen Profildatensatz eines Response-Objekts aus.

Rückgabewert

Das Response-Objekt gibt ein Array aus NameValuePair-Objekten zurück.

Beispiel

Das folgende Beispiel ist ein Auszug aus einer Methode, die das Antwortobjekt für getOffers verarbeitet.

```
for(NameValuePair nvp : response.getProfileRecord())
{
    System.out.println("Name:"+nvp.getName());
    if(nvp.getValueDataType().equals(NameValuePair.DATA_TYPE_DATETIME))
    {
        System.out.println("Value:"+nvp.getValueAsDate());
    }
    else if(nvp.getValueDataType().equals(NameValuePair.DATA_TYPE_NUMERIC))
    {
         System.out.println("Value:"+nvp.getValueAsNumeric());
    }
    else
    {
        System.out.println("Value:"+nvp.getValueAsString());
    }
}
```

getSessionID

Die getSessionID-Methode gibt eine Sitzungs-ID zurück.

```
getSessionID()
```

Rückgabewert

Die getSessionID-Methode gibt eine Zeichenfolge zurück.

Beispiel

Das folgende Beispiel zeigt eine Nachricht, die Sie am Anfang oder am Ende der Fehlerbehandlung anzeigen können, um anzugeben, auf welche Sitzung sich die Fehler beziehen.

```
System.out.println("Diese Antwort betrifft sessionId:"+response.getSessionID());
```

getStatusCode

Die getStatusCode-Methode gibt den Statuscode eines Response-Objekts zurück.

```
getStatusCode()
```

Rückgabewert

Das Response-Objekt gibt eine Ganzzahl zurück.

- 0 - STATUS_SUCCESS - Die aufgerufene Methode wurde ohne Fehler ausgeführt. Möglicherweise sind Advisory Messages vorhanden.
- 1 - STATUS_WARNING - Die aufgerufene Methode wurde mit mindestens einer Warnung (aber ohne Fehler) ausgeführt. Weitere Informationen finden Sie in den Advisory Messages.
- 2 - STATUS_ERROR - Die aufgerufene Methode wurde nicht erfolgreich ausgeführt und hat mindestens eine Fehlernachricht. Weitere Informationen finden Sie in den Advisory Messages.

Beispiel

Das folgende Beispiel zeigt, wie Sie getStatusCode zur Fehlerbehandlung verwenden können.

```
public static void processSetDebugResponse(Response response)
{
    // check if response is successful or not
    if(response.getStatusCode() == Response.STATUS_SUCCESS)
    {
        System.out.println("setDebug call processed with no warnings or errors");
    }
    else if(response.getStatusCode() == Response.STATUS_WARNING)
    {
        System.out.println("setDebug call processed with a warning");
    }
    else
    {
        System.out.println("setDebug call processed with an error");
    }

    // For any non-successes, there should be advisory messages explaining why
    if(response.getStatusCode() != Response.STATUS_SUCCESS)
        printDetailMessageOfWarningOrError("setDebug",
    response.getAdvisoryMessages());
}
```

Kapitel 8. Klassen und Methoden für die IBM Interact-JavaScript-API

In den folgenden Abschnitten werden Anforderungen und andere Details aufgelistet, die Sie kennen sollten, bevor Sie mit der Interact-JavaScript-API zu arbeiten beginnen.

Die Interact-API unterstützt eine JavaScript-Version für die Kommunikation zwischen dem Endbenutzerclient (Browser) und dem Server.

Anmerkung: In diesem Abschnitt wird vorausgesetzt, dass Sie mit APIs vertraut sind, die auf JavaScript basieren.

Anmerkung: Die mehrmalige Verwendung eines Parameters in einem einzelnen API-Aufruf wird nicht unterstützt.

JavaScript-Voraussetzungen

Bevor Sie die Interact-JavaScript-API auf einer Website verwenden, müssen Sie die Datei `interactapi.js` auf den Webseiten einschließen.

Arbeiten mit Sitzungsdaten

Wenn Sie eine Sitzung mit der Methode `startSession` initialisieren, werden Sitzungsdaten in den Speicher geladen. Während der Sitzung können Sie die Sitzungsdaten (die eine Obermenge der statischen Profildaten sind) lesen und schreiben.

Die Sitzung enthält die folgenden Daten:

- Statische Profildaten
- Segmentzuordnungen
- Echtzeitdaten
- Angebotsempfehlungen

Alle Sitzungsdaten sind bis zum Aufruf der Methode `endSession` bzw. bis zum Ablauf der `sessionTimeout`-Zeit verfügbar. Mit dem Ende der Sitzung gehen alle Daten verloren, die nicht ausdrücklich in den Kontakt- oder Antwortverlauf oder eine andere Datenbanktabelle gespeichert werden.

Die Daten werden als ein Satz von Name/Wert-Paaren gespeichert. Wenn die Daten aus der Datenbanktabelle gelesen werden, ist der Name die Spalte der Tabelle.

Sie können diese Name/Wert-Paare während der Arbeit mit der Interact-API erstellen. Sie müssen nicht alle Name/Wert-Paare in einem Globalbereich deklarieren. Wenn Sie neue Ereignisparameter als Name/Wert-Paare festlegen, fügt die Laufzeitumgebung die Name/Wert-Paare den Sitzungsdaten hinzu. Wenn Sie beispielsweise Ereignisparameter mit der Methode `postEvent` verwenden, fügt die Laufzeitumgebung die Ereignisparameter den Sitzungsdaten hinzu, selbst wenn die Ereignisparameter nicht in den Profildaten verfügbar waren. Diese Daten existieren nur in den Sitzungsdaten.

Sie können Sitzungsdaten jederzeit überschreiben. Beispiel: Wenn ein Abschnitt des Kundenprofils `creditScore` umfasst, können Sie einen Ereignisparameter mithilfe des benutzerdefinierten Typs `NameValuePair` übergeben. In der Klasse `NameValuePair` können Sie die Methoden `setName` und `setValueAsNumeric` verwenden, um den Wert zu ändern. Der Name muss übereinstimmen. Innerhalb der Sitzungsdaten muss beim Namen die Groß-/Kleinschreibung nicht berücksichtigt zu werden. Daher würden die Namen `creditscore` und `CrEdItScOrE` jeweils `creditScore` überschreiben.

Nur die letzten in die Sitzungsdaten geschriebenen Daten werden aufbewahrt. Beispiel: `startSession` lädt die Profildaten für den Wert `lastOffer`. Eine Methode `postEvent` überschreibt `lastOffer`. Dann überschreibt eine zweite Methode `postEvent` `lastOffer`. Die Laufzeitumgebung bewahrt nur die Daten, die von der zweiten Methode `postEvent` geschrieben wurden, in den Sitzungsdaten.

Wenn die Sitzung endet, gehen die Daten verloren, außer Sie haben besondere Vorkehrungen getroffen, wie z. B. die Verwendung eines Prozesses "Momentaufnahme" in Ihrem interaktiven Ablaufdiagramm, um die Daten in eine Datenbanktabelle zu schreiben. Wenn Sie vorhaben, Prozesse "Momentaufnahme" zu verwenden, achten Sie darauf, dass die Namen den Einschränkungen Ihrer Datenbank entsprechen müssen. Beispiel: Wenn nur 256 Zeichen für den Namen einer Spalte zulässig sind, darf der Name des Name/Wert-Paars nicht 256 Zeichen überschreiten.

Mit Callback-Parameter arbeiten

Die Callback-Funktion ist ein zusätzlicher Parameter jeder Methode der Interact-JavaScript-API.

Der Hauptbrowserprozess ist eine aus einem Einzelthread bestehende Ereignisschleife. Wenn eine Operation mit langer Laufzeit in einer Ereignisschleife aus einem Einzelthread ausgeführt wird, wird der Prozess blockiert. So wird die Verarbeitung weiterer Ereignisse durch den Prozess blockiert, während er auf den Abschluss der Operation wartet. Zur Vermeidung von Blockierungen bei Operationen mit langer Laufzeit wird von XMLHttpRequest eine asynchrone Schnittstelle bereitgestellt. Dabei wird ein Callback zur Ausführung nach Abschluss der Operation übergeben; während der Verarbeitung wird die Steuerung an die Hauptereignisschleife zurückgegeben und somit nicht der Prozess blockiert.

Wenn die Methode erfolgreich war, wird von der Callback-Funktion `onSuccess` aufgerufen. Wenn die Methode nicht erfolgreich war, wird von der Callback-Funktion `onError` aufgerufen.

Wenn Sie zum Beispiel Angebote auf Ihrer Webseite anzeigen möchten, würden Sie die Methode `getOffers` und den Callback zum Anzeigen auf der Seite verwenden. Die Webseite verhält sich normal und wartet nicht darauf, dass die Angebote von Interact zurückgegeben werden. Wenn die Angebote stattdessen von Interact zurückgegeben werden, wird die Antwort im Callback-Parameter zurückgesendet. Sie können die Callback-Daten auswerten und Angebote auf der Seite anzeigen.

Sie können einen generischen Callback für alle Funktionen oder auch spezifische Callbacks für bestimmte Funktionen verwenden.

Sie können `var callback = InteractAPI.Callback.create(onSuccess, onError);` zum Erstellen einer generischen Callback-Funktion verwenden.

Sie können die folgende Funktion zum Erstellen einer spezifischen Callback-Funktion für die Methode "getOffers" verwenden.

```
var callbackforGetOffer = InteractAPI.Callback.create(onSuccessofGetOffer,
onErrorofGetOffer);
```

Informationen zur Klasse InteractAPI

Die Klasse InteractAPI enthält die Methoden, die Sie verwenden, um Ihren Touchpoint in den Laufzeitserver zu integrieren. Alle anderen Klassen und Methoden in der Interact-API unterstützen die Methoden in dieser Klasse.

Sie müssen Ihre Implementierung anhand von `interact_client.jar` im `lib`-Verzeichnis Ihrer Interact-Laufzeitumgebungsinstallation kompilieren.

startSession

Die `startSession`-Methode erstellt und definiert eine Laufzeitsitzung.

```
function callStartSession(commandsToExecute, callback) {

        //Konfigurierte Startsitzung lesen
        var ssId = document.getElementById('ss_sessionId').value;
        var icName = document.getElementById('ic').value;
        var audId = document.getElementById('audienceId').value;
        var audLevel = document.getElementById('audienceLevel').value;
        var params = document.getElementById('ss_parameters').value;
        var relyOldSs = document.getElementById('relyOnOldSession').value;
        var debug = document.getElementById('ss_isDebug').value;

        InteractAPI.startSession(ssId, icName,
                                 getNameValuePairs(audId), audLevel,
                                 getNameValuePairs(params), relyOldSs,
                                 debug, callback) ;

}
```

`startSession` kann bis zu fünf Aktionen auslösen:

- Erstellen der Laufzeitsitzung.
- Laden der Besucherprofildaten für die aktuelle Zielgruppenebene in der Laufzeitsitzung inklusive aller Dimensionstabellen, die in der für den interaktiven Kanal definierten Tabellenzuordnung zum Laden markiert sind.
- Auslösen der Segmentierung, indem alle interaktiven Ablaufdiagramme für die aktuelle Zielgruppenebene ausgeführt werden.
- Laden von Angebotsunterdrückungsdaten in der Sitzung, wenn die Eigenschaft `enableOfferSuppressionLookup` auf wahr gesetzt ist.
- Laden von Bewertungsüberschreibungsdaten in der Sitzung, wenn die Eigenschaft `enableScoreOverrideLookup` auf wahr gesetzt ist.

Die `startSession`-Methode benötigt die folgenden Parameter:

- **sessionID** - eine Zeichenfolge zur Identifizierung der Sitzungs-ID. Sie müssen die Sitzungs-ID definieren. Sie können zum Beispiel eine Kombination aus Kunden-ID und Zeitmarke verwenden.

  Sie müssen eine Sitzungs-ID angeben, um zu definieren, was eine Laufzeitsitzung auszeichnet. Dieser Wert wird vom Client verwaltet. Alle Methodenaufrufe für die gleiche Sitzungs-ID müssen vom Client synchronisiert werden. Das Verhalten für gleichzeitige API-Aufrufe mit der gleichen Sitzungs-ID ist nicht definiert.

- **relyOnExistingSession** - ein boolescher Ausdruck, der definiert, ob diese Sitzung eine neue oder eine vorhandene Sitzung verwendet. Gültige Werte sind `true` oder `false`. Wenn der Wert `true` ist, müssen Sie eine vorhandene Sitzungs-ID mit der `startSession`-Methode angeben. Wenn der Wert `false` ist, müssen Sie eine neue Sitzungs-ID angeben.

  Wenn Sie `relyOnExistingSession` auf `true` setzen und eine Sitzung vorhanden ist, verwendet die Laufzeitumgebung die vorhandenen Sitzungsdaten. Es werden keine Daten erneut geladen und es wird keine Segmentierung ausgelöst. Wenn die Sitzung nicht vorhanden ist, erstellt die Laufzeitumgebung eine neue Sitzung inklusive Laden der Daten und Auslösen der Segmentierung. Wenn die Sitzungsdauer des Touchpoints länger als die Laufzeitsitzung ist, kann es sinnvoll sein, `relyOnExistingSession` auf true zu setzen und mit allen `startSession`-Aufrufen zu verwenden. Beispiel: Die Sitzung einer Website ist 2 Stunden lang aktiv, während die Laufzeitsitzung nur 20 Minuten lang aktiv ist.

  Wenn Sie `startSession` zweimal mit der gleichen Sitzungs-ID aufrufen, gehen alle Sitzungsdaten des ersten `startSession`-Aufrufs verloren, wenn `relyOnExistingSession` auf false gesetzt ist.
- **debug** - ein boolescher Ausdruck zum Aktivieren oder Inaktivieren von Daten zur Fehlerbehebung. Gültige Werte sind `true` oder `false`. Wenn der Wert wahr ist, protokolliert Interact die Daten zur Fehlerbehebung in den Protokollen des Laufzeitservers. Die Debugmarkierung wird für jede Sitzung individuell gesetzt. Somit können Sie die Daten zur Fehlerbehebung für einzelne Sitzungen verfolgen.
- **interactiveChannel** - eine Zeichenfolge, die den Namen des interaktiven Kanals definiert, auf den diese Sitzung verweist. Dieser Name muss exakt mit dem in Campaign definierten Namen des interaktiven Kanals übereinstimmen.
- **audienceID** - ein Array aus NameValuePairImpl-Objekten, wobei die Namen mit den physischen Spaltennamen aller Tabellen übereinstimmen müssen, in denen die Zielgruppen-ID enthalten ist.
- **audienceLevel** - eine Zeichenfolge, die die Zielgruppenebene definiert.
- **parameters** - NameValuePairImpl-Objekte zum Identifizieren aller Parameter, die mit `startSession` übergeben werden müssen. Diese Werte werden in den Sitzungsdaten gespeichert und können zur Segmentierung verwendet werden.

  Wenn Sie mehrere interaktive Ablaufdiagramme für die gleiche Zielgruppenebene haben, müssen Sie eine Obermenge mit allen Spalten in allen Tabellen einschließen. Wenn Sie die Laufzeit so konfigurieren, dass die Profiltabelle geladen wird, und die Profiltabelle alle benötigten Spalten enthält, ist es nicht erforderlich, Parameter zu übergeben, es sei denn, Sie möchten die Daten in der Profiltabelle überschreiben. Wenn die Profiltabelle eine Untermenge der benötigten Spalten enthält, müssen Sie die fehlenden Spalten als Parameter einschließen.
- **callback** - Wenn die Methode erfolgreich war, wird von der Callback-Funktion `onSuccess` aufgerufen. Wenn die Methode nicht erfolgreich war, wird von der Callback-Funktion `onError` aufgerufen.

Wenn `audienceID` oder `audienceLevel` ungültig und `relyOnExistingSession` false ist, schlägt der `startSession`-Aufruf fehl. Wenn `interactiveChannel` ungültig ist, schlägt `startSession` fehl, unabhängig davon, ob `relyOnExistingSession` true oder false ist.

Wenn `relyOnExistingSession` true ist und Sie einen zweiten `startSession`-Aufruf mit der gleichen `sessionID` durchführen, nachdem die erste Sitzung bereits abgelaufen ist, erstellt Interact eine neue Sitzung.

Wenn relyOnExistingSession true ist und Sie einen zweiten startSession-Aufruf mit der gleichen sessionID, aber mit einer anderen audienceID oder audienceLevel durchführen, ändert der Laufzeitserver die Zielgruppe für die vorhandene Sitzung.

Wenn relyOnExistingSession true ist und Sie einen zweiten startSession-Aufruf mit der gleichen sessionID, aber mit einem anderen interactiveChannel durchführen, erstellt der Laufzeitserver eine neue Sitzung.

Rückgabewert

Der Laufzeitserver beantwortet startSession mit einem Response-Objekt, das die folgenden Attribute enthält:

- AdvisoryMessages (wenn StatusCode nicht 0 ist)
- ApiVersion
- SessionID
- StatusCode

Deduplizierung von Angeboten über Angebotsattribute hinweg

Wenn Sie die Anwendungsprogrammierschnittstelle (API) von Interact verwenden, können Sie mit den folgenden beiden API-Aufrufen Angebote erhalten: getOffers und getOffersForMultipleInteractionPoints.
getOffersForMultipleInteractionPoints kann auf der Ebene *OfferID* verhindern, dass doppelte Angebote zurückgegeben werden. Die Deduplizierung von Angeboten über die gesamte Angebotskategorie hinweg ist jedoch nicht möglich. Früher war beispielsweise eine Ausweichmaßnahme erforderlich, damit Interact immer nur jeweils ein Angebot aus jeder Angebotskategorie zurückgab. Der API-Aufruf startSession enthält jetzt zwei neue Parameter, die ab sofort eine Deduplizierung von Angeboten über Angebotsattribute hinweg ermöglichen (beispielsweise für Kategorien).

In dieser Liste finden Sie eine Übersicht über die Parameter, die dem API-Aufruf startSession hinzugefügt wurden. Sie finden weitere Informationen zu diesen Parametern oder zu sonstigen Aspekten der Interact-API im *IBM Interact-Administratorhandbuch* sowie in den Javadoc-Dateien, die in Ihrer Interact-Installation enthalten sind. Sie finden diese Dateien im Pfad <Interact_Home>/docs/apiJavaDoc.

- UACIOfferDedupeAttribute. Wenn Sie einen startSession-API-Aufruf mit Angebotsdeduplizierung erstellen möchten, damit nachfolgende getOffer-Aufrufe immer nur jeweils ein Angebot aus jeder Kategorie zurückgeben, müssen Sie den Parameter UACIOfferDedupeAttribute als Bestandteil des API-Aufrufs einschließen. Sie können einen Parameter wie folgt im Format Name,Wert,Typ angeben:

  UACIOfferDedupeAttribute,<Attributname>,string

  In diesem Beispiel würden Sie <*Attributname*> durch den Namen des Angebotsattributs ersetzen, das Sie als Kriterium für die Deduplizierung verwenden möchten (zum Beispiel "Category" (Kategorie)).

  Anmerkung: Interact überprüft die Angebote, die den von Ihnen angegebenen Attributwert (zum Beispiel "Category") aufweisen. Anschließend wird eine Deduplizierung durchgeführt, damit mit Ausnahme des Angebots, das die höchste Bewertung aufweist, alle Angebote entfernt werden. Wenn die Angebote mit einem Attributduplikat auch identische Bewertungen aufweisen, trifft Interact eine Zufallsauswahl unter den übereinstimmenden Angeboten und gibt diese zurück.
-

UACINoAttributeDedupeIfFewerOffer. Wenn Sie den Parameter UACIOfferDedupeAttribute im startSession-Aufruf einschließen, können Sie auch den Parameter UACINoAttributeDedupeIfFewerOffer festlegen. Damit geben Sie an, wie sich das Programm verhalten soll, wenn die Angebotsliste nach der Deduplizierung nicht genügend Angebote enthält, damit die ursprüngliche Anforderung erfüllt werden kann.

Wenn Sie beispielsweise für UACIOfferDedupeAttribute die Verwendung der Angebotskategorie zur Deduplizierung von Angeboten festlegen und Ihr nachfolgender getOffers-Aufruf die Rückgabe von acht Angeboten anfordert, kann es sein, dass aufgrund der Deduplizierung weniger als acht auswählbare Angebote verfügbar sind. Wenn Sie den Parameter UACINoAttributeDedupeIfFewerOffer auf "true" setzen, werden in diesem Fall einige der deduplizierten Angebote zur Kandidatenliste hinzugefügt, damit die angeforderte Anzahl der Angebote erfüllt wird. Wenn Sie in diesem Beispiel den Parameter auf "false" setzen, wird die angeforderte Anzahl nicht durch die zurückgegebene Anzahl an Angeboten erreicht.

UACINoAttributeDedupeIfFewerOffer ist standardmäßig auf "true" gesetzt.

Angenommen, Sie haben wie folgt als Parameter für startSession die Angebotskategorie (Category) als Deduplizierungskriterium angegeben:

```
UACIOfferDedupeAttribute,Category,string;

UACINoAttributeDedupeIfFewerOffer,1,string
```

Standardmäßig dedupliziert UACIOfferDedupeAttribute Angebote nicht, wenn weniger als die angeforderte Anzahl zurückgegeben wurde. Um jedoch die Deduplizierung sicherzustellen, wenn weniger als die angeforderte Anzahl Angebote zurückgegeben wurde, muss der Parameter UACINoAttributeDedupeIfFewerOffer angegeben und auf 1 festgelegt werden.

Diese Parameterkombination bewirkt, dass Interact Angebote auf Basis des Angebotsattributs "Category" dedupliziert und selbst dann nur die deduplizierten Angebote zurückgibt, wenn die resultierende Anzahl der Angebote die angeforderte Anzahl unterschreitet (da der Parameter UACINoAttributeDedupeIfFewerOffer auf "false" gesetzt ist).

Wenn Sie einen getOffers-API-Aufruf ausgeben, könnte die ursprüngliche Gruppe der auswählbaren Angebote die folgenden Angebote enthalten:

- Category=Electronics: Angebot A1 mit einer Bewertung von 100 und Angebot A2 mit einer Bewertung von 50.
- Category=Smartphones: Angebot B1 mit einer Bewertung von 100, Angebot B2 mit einer Bewertung von 80 und Angebot B3 mit einer Bewertung von 50.
- Category=MP3Players: Angebot C1 mit einer Bewertung von 100, Angebot C2 mit einer Bewertung von 50.

In diesem Fall gab es zwei doppelte Angebote, die mit der ersten Kategorie übereinstimmen, drei doppelte Angebote, die mit der zweiten Kategorie übereinstimmen, und zwei doppelte Angebote, die mit der dritten Kategorie übereinstimmen. Als Angebote werden die Angebote mit der höchsten Bewertung aus jeder Kategorie zurückgegeben, also Angebot A1, Angebot B1 und Angebot C1.

Obwohl der getOffers-API-Aufruf sechs Angebote angefordert hat, werden nur drei Angebote zurückgegeben, da der Parameter UACINoAttributeDedupeIfFewerOffer in diesem Beispiel auf "false" gesetzt ist.

Wenn der getOffers-API-Aufruf sechs Angebote angefordert hat und in diesem Beispiel keine Angabe für den Parameter UACINoAttributeDedupeIfFewerOffer gemacht oder der Parameter explizit auf "true" gesetzt wurde, werden einige der doppelten Angebote in das Ergebnis aufgenommen, damit die angeforderte Anzahl erfüllt wird.

postEvent

Mit der postEvent-Methode können Sie jedes Ereignis ausführen, das im interaktiven Kanal definiert ist.

```
function callPostEvent(commandsToExecute, callback) {

        var ssId = document.getElementById('pe_sessionId').value;
        var ev = document.getElementById('event').value;
        var params = document.getElementById('parameters').value;

        InteractAPI.postEvent(ssId, ev, getNameValuePairs(params), callback);

}
```

- **sessionID**: eine Zeichenfolge zur Identifizierung der Sitzungs-ID.
- **eventName**: eine Zeichenfolge zur Identifizierung des Ereignisnamens.

  Anmerkung: Der Name des Ereignisses muss mit dem im interaktiven Kanal definierten Ereignisnamen übereinstimmen. Bei diesem Namen braucht die Groß-/Kleinschreibung nicht berücksichtigt zu werden.
- **eventParameters**. NameValuePairImpl-Objekte geben alle Parameter an, die mit dem Ereignis übergeben werden müssen. Diese Werte werden in den Sitzungsdaten gespeichert.

  Wenn dieses Ereignis eine erneute Segmentierung auslöst, müssen Sie sicherstellen, dass alle vom interaktiven Ablaufdiagramm benötigten Daten in den Sitzungsdaten verfügbar sind. Wenn diese Werte noch nicht durch vorangegangene Aktionen ausgefüllt wurden (zum Beispiel durch startSession, durch setAudience oder beim Laden der Profiltabelle), müssen Sie für jeden fehlenden Wert einen eventParameter einschließen. Wenn Sie zum Beispiel alle Profiltabellen so konfiguriert haben, dass diese im Hauptspeicher geladen werden, müssen Sie für alle temporären Daten, die für interaktive Ablaufdiagramme erforderlich sind, jeweils ein NameValuePair einschließen.

  Wenn Sie mehrere Zielgruppenebenen verwenden, haben Sie vermutlich verschiedene Sätze an eventParameters für jede Zielgruppenebene. Sie sollten daher eine entsprechende Logik einschließen, die gewährleistet, dass für jede Zielgruppenebene immer der richtige Parametersatz ausgewählt wird.

  Wichtig: Wenn dieses Ereignis den Antwortverlauf protokolliert, müssen Sie den Verfahrenscode für das Angebot übergeben. Sie müssen den Namen für das NameValuePair als "UACIOfferTrackingCode" definieren.

  Pro Ereignis können Sie immer nur einen Verfahrenscode übergeben. Wenn Sie den Verfahrenscode für einen Angebotskontakt nicht übergeben, protokolliert Interact jeweils einen Angebotskontakt für jedes Angebot in der zuletzt empfohlenen Angebotsliste. Wenn Sie den Verfahrenscode für eine Antwort nicht übergeben, gibt Interact einen Fehler zurück.
- **callback** - Wenn die Methode erfolgreich war, wird von der Callback-Funktion onSuccess aufgerufen. Wenn die Methode nicht erfolgreich war, wird von der Callback-Funktion onError aufgerufen.
- Es gibt eine Reihe weiterer reservierter Parameter, die Sie mit postEvent und anderen Methoden verwenden können, die später in diesem Abschnitt erläutert werden.

Wenn Sie eine erneute Segmentierung anfordern oder in den Kontakt- oder Antwortverlauf schreiben, wird nicht auf eine Antwort gewartet.

Im Verlauf einer neuen Segmentierung werden nicht frühere Segmentierungsergebnisse für die aktuelle Zielgruppenebene bereinigt. Sie können den Parameter `UACIExecuteFlowchartByName` zum Definieren bestimmter Ablaufdiagramme verwenden, die ausgeführt werden sollen. Die `getOffers`-Methode wartet, bis die erneute Segmentierung abgeschlossen ist, und wird erst danach ausgeführt. Daher kann es zu einer Verzögerung kommen, wenn Sie eine `postEvent`-Methode aufrufen, die sofort eine erneute Segmentierung auslöst, bevor ein `getOffers`-Aufruf erfolgt.

Rückgabewert

Der Laufzeitserver beantwortet `postEvent` mit einem Response-Objekt, das die folgenden Attribute enthält:

- AdvisoryMessages
- ApiVersion
- OfferList
- Profil
- SessionID
- StatusCode

getOffers

Mit der `getOffers`-Methode können Sie Angebote vom Laufzeitserver anfordern.

```
function callGetOffers(commandsToExecute, callback) {

        var ssId = document.getElementById('go_sessionId').value;
        var ip = document.getElementById('go_ipoint').value;
        var nofRequested = 5 ;
        var nreqString = document.getElementById('offersRequested').value;

        InteractAPI.getOffers(ssId, ip, nofRequested, callback);

}
```

- **sessionID** - eine Zeichenfolge, die die aktuelle Sitzung angibt.
- **interactionPoint** - eine Zeichenfolge, die den Namen des Interaktionspunkts angibt, auf den diese Methode verweist.

  Anmerkung: Dieser Name muss exakt mit dem Namen des im interaktiven Kanal definierten Interaktionspunkts übereinstimmen.
- **nofRequested** - eine Ganzzahl, die die Anzahl der angeforderten Angebote angibt.
- **callback** - Wenn die Methode erfolgreich war, wird von der Callback-Funktion `onSuccess` aufgerufen. Wenn die Methode nicht erfolgreich war, wird von der Callback-Funktion `onError` aufgerufen.

Bevor die `getOffers`-Methode ausgeführt wird, wartet sie so viele Millisekunden, wie in der `segmentationMaxWaitTimeInMS`-Eigenschaft angegeben sind, um die erneute Segmentierung abzuschließen. Daher kann es zu einer Verzögerung kommen, wenn Sie eine `postEvent`-Methode aufrufen, die sofort eine erneute Segmentierung oder eine `setAudience`-Methode auslöst, bevor ein `getOffers`-Aufruf erfolgt.

Rückgabewert

Der Laufzeitserver beantwortet `getOffers` mit einem Response-Objekt, das die folgenden Attribute enthält:

- AdvisoryMessages
- ApiVersion
- OfferList
- Profil
- SessionID
- StatusCode

getOffersForMultipleInteractionPoints

Mit der `getOffersForMultipleInteractionPoints`-Methode können Sie Angebote vom Laufzeitserver für mehrere IPs mit Deduplizierung anfordern.

```
function callGetOffersForMultipleInteractionPoints(commandsToExecute, callback) {

        var ssId = document.getElementById('gop_sessionId').value;
        var requestDetailsStr = document.getElementById('requestDetail').value;

        //Zeichenfolge abschneiden
        var trimmed = requestDetailsStr.replace(/\{/g, "");
        var parts = trimmed.split("}");

        //Zeichenfolgen bereinigen
        for(i = 0; i < parts.length; i += 1) {
            parts[i] = parts[i].replace(/^\s+|\s+$/g, "");
        }

        //Anforderungen zum Abrufen von Angeboten erstellen
        var getOffReqs = [];
        for(var i = 0; i < parts.length; i += 1) {
            var getofReqObj = parseGetOfferReq(parts[i]);
            if (getofReqObj) {
                getOffReqs.push(getofReqObj);

        InteractAPI.getOffersForMultipleInteractionPoints
        (ssId, getOffReqs, callback);

}
```

- **sessionID** - eine Zeichenfolge, die die aktuelle Sitzung angibt.
- **requestDetailsStr** - eine Zeichenfolge, die ein Array aus `GetOfferRequest`-Objekten angibt.

  Jedes `GetOfferRequest`-Objekt legt fest:

  – **ipName** - Der Name des Interaktionspunkts (IP), für den das Objekt Angebote anfordert

  – **numberRequested** - Die Anzahl an eindeutigen Angeboten, die für den angegebenen IP erforderlich ist

  – **offerAttributes** - Anforderungen an die Attribute der gelieferten Angebote mit einer Instanz von `OfferAttributeRequirements`

  – **duplicationPolicy** - Duplizierungsrichtlinien-ID für die Angebote, die geliefert werden

    Duplizierungsrichtlinien legen fest, ob doppelte Angebote mit verschiedenen Interaktionspunkten an einen einzelnen Methodenaufruf zurückgegeben werden sollen. (*Innerhalb* eines einzelnen Interaktionspunkts werden doppelte Angebote nie zurückgegeben.) Gegenwärtig werden zwei Duplizierungsrichtlinien unterstützt.

- NO_DUPLICATION (ID-Wert = 1). Keines der Angebote, die in den vorangegangenen `GetOfferRequest`-Instanzen enthalten waren, wird in diese `GetOfferRequest`-Instanz einbezogen (das heißt, Interact wendet die Deduplizierung an).
- ALLOW_DUPLICATION (ID-Wert = 2). Alle Angebote, die die Voraussetzungen erfüllen, die in dieser `GetOfferRequest`-Instanz angegeben sind, werden einbezogen. Es findet kein Abgleich der Angebote statt, die in den vorangegangenen `GetOfferRequest`-Instanzen enthalten waren.

– **callback** - Wenn die Methode erfolgreich war, wird von der Callback-Funktion `onSuccess` aufgerufen. Wenn die Methode nicht erfolgreich war, wird von der Callback-Funktion `onError` aufgerufen.

Die Reihenfolge der Anfragen im Array-Parameter ist auch die Reihenfolge der Priorität, in der die Angebote geliefert werden.

Beispiel: Angenommen, die IPs in der Anfrage heißen IP1 und IP2, duplizierte Angebote sind unzulässig (mit der Duplizierungsrichtlinien-ID = 1) und jeder IP fordert zwei Angebote an. Wenn Interact die Angebote A, B und C für IP1 und die Angebote A und D für IP2 findet, enthält die Antwort die Angebote A und B für IP1 und nur das Angebot D für IP2.

Zusätzlicher Hinweis: Wenn die Duplizierungsrichtlinien-ID 1 lautet, werden Angebote, die über einen IP mit hoher Priorität geliefert wurden, nicht über diesen IP geliefert.

Bevor die `getOffersForMultipleInteractionPoints`-Methode ausgeführt wird, wartet sie so viele Millisekunden, wie in der `segmentationMaxWaitTimeInMS`-Eigenschaft angegeben sind, um die erneute Segmentierung abzuschließen. Daher kann es zu einer Verzögerung kommen, wenn Sie eine `postEvent`-Methode aufrufen, die sofort eine erneute Segmentierung oder eine `setAudience`-Methode auslöst, bevor ein `getOffers`-Aufruf erfolgt.

Rückgabewert

Der Laufzeitserver beantwortet `getOffersForMultipleInteractionPoints` mit einem Response-Objekt, das die folgenden Attribute enthält:

- AdvisoryMessages
- ApiVersion
- Array von OfferList
- Profil
- SessionID
- StatusCode

setAudience

Mit der `setAudience`-Methode können Sie die Zielgruppen-ID und die Zielgruppenebene für Besucher festlegen.

```
function callSetAudience(commandsToExecute, callback) {

        var ssId = document.getElementById('sa_sessionId').value;
        var audId = document.getElementById('sa_audienceId').value;
        var audLevel = document.getElementById('sa_audienceLevel').value;
        var params = document.getElementById('sa_parameters').value;

        InteractAPI.setAudience(ssId, getNameValuePairs(audId),audLevel,
                                      getNameValuePairs(params), callback);

}
```

- **sessionID** - eine Zeichenfolge zur Identifizierung der Sitzungs-ID.
- **audienceID** - ein Array von NameValuePairImpl-Objekten zum Definieren der Zielgruppen-ID.
- **audienceLevel** - eine Zeichenfolge zum Definieren der Zielgruppenebene.
- **parameters** - NameValuePairImpl-Objekte zum Identifizieren aller Parameter, die mit setAudience übergeben werden müssen. Diese Werte werden in den Sitzungsdaten gespeichert und können zur Segmentierung verwendet werden.

  Sie benötigen für jede Spalte in Ihrem Profil einen Wert. Dies ist eine Obermenge aus allen Spalten in den Echtzeitdaten und in allen Tabellen, die für den interaktiven Kanal definiert sind. Wenn Sie bereits alle Sitzungsdaten mit startSession oder postEvent ausgefüllt haben, ist es nicht erforderlich, neue Parameter zu senden.
- **callback** - Wenn die Methode erfolgreich war, wird von der Callback-Funktion onSuccess aufgerufen. Wenn die Methode nicht erfolgreich war, wird von der Callback-Funktion onError aufgerufen.

Die setAudience-Methode löst eine erneute Segmentierung aus. Die getOffers-Methode wartet, bis die erneute Segmentierung abgeschlossen ist, und wird erst danach ausgeführt. Daher kann es zu einer Verzögerung kommen, wenn Sie eine setAudience-Methode aufrufen, bevor ein getOffers-Aufruf erfolgt.

Die setAudience-Methode lädt auch die Profildaten für die Zielgruppen-ID. Mit der setAudience-Methode können Sie erzwingen, dass erneut die gleichen Profildaten geladen werden wie mit der startSession-Methode.

Die Methode setAudience lädt die Whitelist- und die Blacklist-Tabelle in einer vorhandenen Sitzung neu. Sie können die Methode setAudience mit den Parametern UACIPurgePriorWhiteListOnLoad und UACIPurgePriorBlackListOnLoad verwenden, um die Whitelist-Tabelle und die Blacklist-Tabelle in einer vorhandenen Sitzung neu zu laden.

Wenn die Methode setAudience aufgerufen wird, wird standardmäßig der gesamte Inhalt der Blacklist entfernt. Die Parameter UACIPurgePriorWhiteListOnLoad und UACIPurgePriorBlackListOnLoad können Sie im Aufruf von setAudience wie folgt festlegen:

- Wenn Sie UACIPurgePriorBlackListOnLoad= 0 festlegen, bleibt der gesamte Inhalt der Whitelist-Tabelle erhalten.
- Wenn Sie UACIPurgePriorWhiteListOnLoad= 1 festlegen, wird der Inhalt der Tabelle entfernt und der Inhalt der Whitelist oder Blacklist für die Zielgruppen-ID wird aus der Datenbank geladen. Im Anschluss daran wird die erneute Segmentierung gestartet.

Rückgabewert

Der Laufzeitserver beantwortet setAudience mit einem Response-Objekt, das die folgenden Attribute enthält:

- AdvisoryMessages
- ApiVersion
- OfferList
- Profil
- SessionID
- StatusCode

getProfile

Mit der getProfile-Methode können Sie Profildaten und temporäre Informationen über die Besucher des Touchpoints abrufen.

```
function callGetProfile(commandsToExecute, callback) {

        var ssId = document.getElementById('gp_sessionId').value;

        InteractAPI.getProfile(ssId, callback);

}
```

- **sessionID** - Eine Zeichenfolge zur Identifizierung der Sitzungs-ID.
- **callback** - Wenn die Methode erfolgreich war, wird von der Callback-Funktion onSuccess aufgerufen. Wenn die Methode nicht erfolgreich war, wird von der Callback-Funktion onError aufgerufen.

Rückgabewert

Der Laufzeitserver beantwortet getProfile mit einem Response-Objekt, das die folgenden Attribute enthält:

- AdvisoryMessages
- ApiVersion
- OfferList
- ProfileRecord
- SessionID
- StatusCode

endSession

Die endSession-Methode markiert das Ende der Laufzeitsitzung. Wenn der Laufzeitserver diese Methode empfängt, wird der Verlauf protokolliert und der Speicher gelöscht.

```
function callEndSession(commandsToExecute, callback) {

        var ssId = document.getElementById('es_sessionId').value;

        InteractAPI.endSession(ssId, callback);

}
```

- **sessionID** - Eindeutige Zeichenfolge zur Identifizierung der Sitzung.
- **callback** - Wenn die Methode erfolgreich war, wird von der Callback-Funktion onSuccess aufgerufen. Wenn die Methode nicht erfolgreich war, wird von der Callback-Funktion onError aufgerufen.

Zeitlimitüberschreitung der Laufzeitsitzungen, wenn die endSession-Methode nicht aufgerufen wird. Das Zeitlimitintervall ist mit der sessionTimeout-Eigenschaft konfigurierbar.

Rückgabewert

Der Laufzeitserver beantwortet die endSession-Methode mit dem Response-Objekt, das die folgenden Attribute enthält:

- SessionID
- ApiVersion
- OfferList

- Profil
- StatusCode
- AdvisoryMessages

setDebug

Mit der setDebug-Methode können Sie den Detaillierungsgrad der Protokollierung für alle Codepfade für die Sitzung festlegen.

```
function callSetDebug(commandsToExecute, callback) {

        var ssId = document.getElementById('sd_sessionId').value;
        var isDebug = document.getElementById('isDebug').value;

        InteractAPI.setDebug(ssId, isDebug, callback);

}
```

- **sessionID** - eine Zeichenfolge zur Identifizierung der Sitzungs-ID.
- **debug** - ein boolescher Ausdruck zum Aktivieren oder Inaktivieren von Daten zur Fehlerbehebung. Gültige Werte sind `true` oder `false`. Wenn der Wert wahr ist, protokolliert Interact die Daten zur Fehlerbehebung im Protokoll des Laufzeitservers.
- **callback** - Wenn die Methode erfolgreich war, wird von der Callback-Funktion `onSuccess` aufgerufen. Wenn die Methode nicht erfolgreich war, wird von der Callback-Funktion `onError` aufgerufen.

Rückgabewert

Der Laufzeitserver beantwortet `setDebug` mit einem Response-Objekt, das die folgenden Attribute enthält:

- AdvisoryMessages
- ApiVersion
- OfferList
- Profil
- SessionID
- StatusCode

getVersion

Die `getVersion`-Methode gibt die Version der aktuellen Implementierung des Interact Laufzeitservers zurück.

```
function callGetVersion(commandsToExecute, callback) {

        InteractAPI.getVersion(callback);


}
```

Es empfiehlt sich, diese Methode zu verwenden, wenn Sie den Touchpoint mit dem Interact API initialisieren.

- **callback** - Wenn die Methode erfolgreich war, wird von der Callback-Funktion `onSuccess` aufgerufen. Wenn die Methode nicht erfolgreich war, wird von der Callback-Funktion `onError` aufgerufen.

Rückgabewert

Der Laufzeitserver beantwortet `getVersion` mit einem Response-Objekt, das die folgenden Attribute enthält:

- AdvisoryMessages
- ApiVersion
- OfferList
- Profil
- SessionID
- StatusCode

executeBatch

Mit der `executeBatch`-Methode können Sie mehrere Methoden mit einer einzelnen Anfrage an den Laufzeitserver ausführen.

```
function callExecuteBatch(commandsToExecute, callback) {

        if (!commandsToExecute)
            return ;

        InteractAPI.executeBatch(commandsToExecute.ssid,
        commandsToExecute.commands, callback);
}
```

- **sessionID** - Eine Zeichenfolge zur Identifizierung der Sitzungs-ID. Diese Sitzungs-ID wird für alle Befehle verwendet, die dieser Methodenaufruf ausführt.
- **commands** - Ein Array aus Befehlsobjekten, jeweils eines für jeden Befehl, der ausgeführt werden soll.
- **callback** - Wenn die Methode erfolgreich war, wird von der Callback-Funktion `onSuccess` aufgerufen. Wenn die Methode nicht erfolgreich war, wird von der Callback-Funktion `onError` aufgerufen.

Durch den Aufruf dieser Methode wird das gleiche Ergebnis erzielt wie durch den expliziten Aufruf jeder einzelnen Methode im Befehl-Array. Diese Methode minimiert die Anzahl der tatsächlichen Anfragen an den Laufzeitserver. Der Laufzeitserver führt jede Methode seriell aus. Für jeden Aufruf werden alle Fehler oder Warnungen im entsprechenden Response-Objekt für diesen Methodenaufruf aufgezeichnet. Wird ein Fehler gefunden, wird executeBatch mit den verbliebenen Aufrufen im Stapel fortgesetzt. Wenn der Aufruf einer beliebigen Methode in einem Fehler resultiert, wird dieser Fehler im Status auf der höchsten Ebene für das BatchResponse-Objekt angezeigt. Wenn keine Fehler aufgetreten sind, werden im Status auf der höchsten Ebene alle aufgetretenen Warnungen angezeigt. Wenn keine Warnungen aufgetreten sind, wird im Status auf der höchsten Ebene die erfolgreiche Ausführung des Stapels angezeigt.

Rückgabewert

Der Laufzeitserver beantwortet den `executeBatch` mit einem BatchResponse-Objekt.

Beispiel für JavaScript-API

```
function isJavaScriptAPISelected() {
    var radios = document.getElementsByName('api');
    for (var i = 0, length = radios.length; i < length; i++) {
        if (radios[i].checked) {
            if (radios[i].value === 'JavaScript')
                return true;
```

```
            else  // only one radio can be logically checked
                break;
        }
    }
    return false;
}

function processFormForJSInvocation(e) {

    if (!isJavaScriptAPISelected())
        return;

    if (e.preventDefault) e.preventDefault();

    var serverurl = document.getElementById('serviceUrl').value ;
    InteractAPI.init( { "url" : serverurl } );

    var commandsToExecute = { "ssid" : null, "commands" : [] };
    var callback = InteractAPI.Callback.create(onSuccess, onError);

    callStartSession(commandsToExecute, callback);
    callGetOffers(commandsToExecute, callback);
    callGetOffersForMultipleInteractionPoints(commandsToExecute, callback);
    callPostEvent(commandsToExecute, callback);
    callSetAudience(commandsToExecute, callback);
    callGetProfile(commandsToExecute, callback);
    callEndSession(commandsToExecute, callback);
    callSetDebug(commandsToExecute, callback);
    callGetVersion(commandsToExecute, callback);

    callExecuteBatch(commandsToExecute, callback);

    // Sie müssen "False" zurückgeben, um das Standardverhalten zu vermeiden
    return false;
}

function callStartSession(commandsToExecute, callback) {

    //Konfigurierte Startsitzung lesen
    var ssId = document.getElementById('ss_sessionId').value;
    var icName = document.getElementById('ic').value;
    var audId = document.getElementById('audienceId').value;
    var audLevel = document.getElementById('audienceLevel').value;
    var params = document.getElementById('ss_parameters').value;
    var relyOldSs = document.getElementById('relyOnOldSession').value;
    var debug = document.getElementById('ss_isDebug').value;

    if (commandsToExecute && !commandsToExecute.ssid) {
        commandsToExecute.ssid = ssId;
    }

    if (commandsToExecute && commandsToExecute.commands) {
        commandsToExecute.commands.push(InteractAPI.CommandUtil.
                                createStartSessionCmd(
                                icName, getNameValuePairs(audId),
                                audLevel, getNameValuePairs(params),
                                relyOldSs, debug));
    }
    else {
        InteractAPI.startSession(ssId, icName,
                                getNameValuePairs(audId), audLevel,
                                getNameValuePairs(params), relyOldSs,
                                debug, callback) ;
    }

}
```

```
function callGetOffers(commandsToExecute, callback) {

    var ssId = document.getElementById('go_sessionId').value;
    var ip = document.getElementById('go_ipoint').value;
    var nofRequested = 5 ;
    var nreqString = document.getElementById('offersRequested').value;
    if (!nreqString && nreqString!== '')
        nofRequested = Number(nreqString);

    if (commandsToExecute && !commandsToExecute.ssid) {
        commandsToExecute.ssid = ssId;
    }

    if (commandsToExecute && commandsToExecute.commands) {
        commandsToExecute.commands.push(InteractAPI.CommandUtil.
                                createGetOffersCmd(ip, nofRequested));
    }
    else {
        InteractAPI.getOffers(ssId, ip, nofRequested, callback);
    }
}

function callPostEvent(commandsToExecute, callback) {

    var ssId = document.getElementById('pe_sessionId').value;
    var ev = document.getElementById('event').value;
    var params = document.getElementById('parameters').value;

    if (commandsToExecute && !commandsToExecute.ssid) {
        commandsToExecute.ssid = ssId;
    }

    if (commandsToExecute && commandsToExecute.commands) {
        commandsToExecute.commands.push(InteractAPI.
                                CommandUtil.createPostEventCmd
                                (ev, getNameValuePairs(params)));
    }
    else {
        InteractAPI.postEvent(ssId, ev, getNameValuePairs(params), callback);
    }
}

function callGetOffersForMultipleInteractionPoints
(commandsToExecute, callback) {

    var ssId = document.getElementById('gop_sessionId').value;
    var requestDetailsStr = document.getElementById('requestDetail').value;

    //Zeichenfolge abschneiden
    var trimmed = requestDetailsStr.replace(/\{/g, "");
    var parts = trimmed.split("}");

    //Zeichenfolgen bereinigen
    for(i = 0; i < parts.length; i += 1) {
        parts[i] = parts[i].replace(/^\s+|\s+$/g, "");
    }

    //Anforderungen zum Abrufen von Angeboten erstellen
    var getOffReqs = [];
    for(var i = 0; i < parts.length; i += 1) {
        var getofReqObj = parseGetOfferReq(parts[i]);
        if (getofReqObj) {
            getOffReqs.push(getofReqObj);
        }
    }

    if (commandsToExecute && !commandsToExecute.ssid) {
```

```
        commandsToExecute.ssid = ssId;
    }

    if (commandsToExecute && commandsToExecute.commands) {
        commandsToExecute.commands.push(InteractAPI.CommandUtil.
                                createGetOffersForMultiple
                                InteractionPointsCmd(getOffReqs));
    }
    else {
        InteractAPI.getOffersForMultipleInteractionPoints
                                (ssId, getOffReqs, callback);
    }
}

function parseGetOfferReq(ofReqStr) {

    if (!ofReqStr || ofReqStr==="")
        return null;

    var posIp = ofReqStr.indexOf(',');
    var ip = ofReqStr.substring(0,posIp);
    var posNmReq = ofReqStr.indexOf(',', posIp+1);
    var numReq = ofReqStr.substring(posIp+1,posNmReq);
    var posDup = ofReqStr.indexOf(',', posNmReq+1);
    var dupPolicy = null;
    var reqAttributes = null;

    if (posDup===-1)
        dupPolicy = ofReqStr.substring(posNmReq+1);
    else
        dupPolicy = ofReqStr.substring(posNmReq+1,posDup);

    //Überprüfen, ob Anforderungszeichenfolge Attribute aufweist
    var reqAttrPos = ofReqStr.search(/\(/g);
    if (reqAttrPos!==-1) {
        var reqAttributesStr = ofReqStr.substring(reqAttrPos);
        reqAttributesStr = trimString(reqAttributesStr);
        reqAttributesStr = removeOpenCloseBrackets(reqAttributesStr);
        reqAttributes = parseReqAttributes(reqAttributesStr);
    }

   return InteractAPI.GetOfferRequest.create(ip, parseInt(numReq),
                                parseInt(dupPolicy), reqAttributes);
}

//Zeichenfolge abschneiden
function trimString(strToTrim) {
    if (strToTrim)
        return strToTrim.replace(/^\s+|\s+$/g, "");
    else
        return null;
}

function trimStrArray(strArray) {
    if (!strArray) return ;
    for(var i = 0; i < strArray.length; i += 1) {
        strArray[i] = trimString(strArray[i]);
    }
}

//Geöffnete und geschlossene Klammern am Ende entfernen
function removeOpenCloseBrackets(strToUpdate) {
    if (strToUpdate)
        return strToUpdate.replace(/^\(+|\)+$/g, "");
    else
        return null;
}
```

```
function parseReqAttributes(ofReqAttrStr) {

    //Zeichenfolge bereinigen
    ofReqAttrStr = trimString(ofReqAttrStr);
    ofReqAttrStr = removeOpenCloseBrackets(ofReqAttrStr);

    if (!ofReqAttrStr || ofReqAttrStr==="")
        return null;

    //Angeforderte Anzahl abrufen
    var pos = ofReqAttrStr.indexOf(",");
    var numRequested = ofReqAttrStr.substring(0,pos);
    ofReqAttrStr = ofReqAttrStr.substring(pos+1);

    //Erster Teil ist Attribut, Rest sind untergeordnete Attribute
    var parts = [];
    pos = ofReqAttrStr.indexOf(",");
    if (pos!==-1) {
        parts.push(ofReqAttrStr.substring(0,pos));
        parts.push(ofReqAttrStr.substring(pos+1));
    }
    else {
        parts.push(ofReqAttrStr);
    }

    for(var i = 0; i < parts.length; i += 1) {
        //Zeichenfolge bereinigen
        parts[i] = trimString(parts[i]);
        parts[i] = removeOpenCloseBrackets(parts[i]);
        parts[i] = trimString(parts[i]);
    }

    //Liste der Attribute erstellen
    var attributes = [];
    var idx = 0;
    if (parts[0]) {
        var attParts = parts[0].split(";");
        for (idx=0; idx<attParts.length; idx++) {
            attParts[idx] = trimString(attParts[idx]);
            attParts[idx] = removeOpenCloseBrackets(attParts[idx]);
            attParts[idx] = trimString(attParts[idx]);

            var atrObj = parseAttribute(attParts[idx]);
            if (atrObj) attributes.push(atrObj);
        }
    }

    //Liste der untergeordneten Attribute erstellen
    var childAttributes = [];
    if (parts[1]) {
        var childAttParts = parts[1].split(")");
        for (idx=0; idx<childAttParts.length; idx++) {

            childAttParts[idx] = trimString(childAttParts[idx]);
            childAttParts[idx] = removeOpenCloseBrackets(childAttParts[idx]);
            childAttParts[idx] = trimString(childAttParts[idx]);


            //Angeforderte Anzahl abrufen
            var noReqPos = childAttParts[idx].indexOf(",");
            var numReqAt = childAttParts[idx].substring(0,noReqPos);
            childAttParts[idx] = childAttParts[idx].substring(noReqPos+1);
            childAttParts[idx] = trimString(childAttParts[idx]);


            var atrObjParsed = parseAttribute(childAttParts[idx]);
```

```
                if (atrObjParsed) {
                    var childReq = InteractAPI.OfferAttributeRequirements.create
                    (parseInt(numReqAt), [atrObjParsed], null);
                    childAttributes.push(childReq);
                }
            }
        }

        return InteractAPI.OfferAttributeRequirements.create(parseInt(numRequested),
        attributes, childAttributes);
    }

    function parseAttribute(attStr) {

        attStr = trimString(attStr);

        if (!attStr || attStr==="")
            return null;

        var pos1 = attStr.indexOf("=");
        var pos2 = attStr.indexOf("|");
        var nvp = InteractAPI.NameValuePair.create
                            ( attStr.substring(0,pos1),
                              attStr.substring(pos1+1, pos2),
                              attStr.substring(pos2+1));
        return nvp;
    }
function callSetAudience(commandsToExecute, callback) {
        if (!document.getElementById('checkSetAudience').checked)
            return ;

        var ssId = document.getElementById('sa_sessionId').value;
        var audId = document.getElementById('sa_audienceId').value;
        var audLevel = document.getElementById('sa_audienceLevel').value;
        var params = document.getElementById('sa_parameters').value;

        if (commandsToExecute && !commandsToExecute.ssid) {
            commandsToExecute.ssid = ssId;
        }

        if (commandsToExecute && commandsToExecute.commands) {
            commandsToExecute.commands.push(InteractAPI.CommandUtil.
                                            createSetAudienceCmd
            (getNameValuePairs(audId), audLevel, getNameValuePairs(params)));
        }
        else {
            InteractAPI.setAudience(ssId, getNameValuePairs(audId),
                                    audLevel, getNameValuePairs(params),
                                    callback);
        }
    }

    function callGetProfile(commandsToExecute, callback) {

        var ssId = document.getElementById('gp_sessionId').value;

        if (commandsToExecute && !commandsToExecute.ssid) {
            commandsToExecute.ssid = ssId;
        }

        if (commandsToExecute && commandsToExecute.commands) {
            commandsToExecute.commands.push(InteractAPI.CommandUtil.
            createGetProfileCmd());
        }
        else {
            InteractAPI.getProfile(ssId, callback);
        }
```

```
}

function callEndSession(commandsToExecute, callback) {

    var ssId = document.getElementById('es_sessionId').value;

    if (commandsToExecute && !commandsToExecute.ssid) {
        commandsToExecute.ssid = ssId;
    }

    if (commandsToExecute && commandsToExecute.commands) {
        commandsToExecute.commands.push(InteractAPI.CommandUtil.
        createEndSessionCmd());
    }
    else {
        InteractAPI.endSession(ssId, callback);
    }
}

function callSetDebug(commandsToExecute, callback) {

    var ssId = document.getElementById('sd_sessionId').value;
    var isDebug = document.getElementById('isDebug').value;

    if (commandsToExecute && !commandsToExecute.ssid) {
        commandsToExecute.ssid = ssId;
    }

    if (commandsToExecute && commandsToExecute.commands) {
        commandsToExecute.commands.push(InteractAPI.CommandUtil.
        createSetDebugCmd(isDebug));
    }
    else {
        InteractAPI.setDebug(ssId, isDebug, callback);
    }
}

function callGetVersion(commandsToExecute, callback) {

    if (commandsToExecute && commandsToExecute.commands) {
        commandsToExecute.commands.push(InteractAPI.CommandUtil.
        createGetVersionCmd());
    }
    else {
        InteractAPI.getVersion(callback);
    }
}

function callExecuteBatch(commandsToExecute, callback) {

    if (!commandsToExecute)
        return ;

    InteractAPI.executeBatch(commandsToExecute.ssid,
    commandsToExecute.commands, callback);
}

function getNameValuePairs(parameters) {

    if (parameters === '')
          return null;

    var parts = parameters.split(';');
    var nvpArray = new Array(parts.length);

    for(i = 0; i < parts.length; i += 1) {
        var nvp = parts[i].split(',') ;
```

```
        var value = null;
        if (nvp[2]===InteractAPI.NameValuePair.prototype.TypeEnum.NUMERIC) {
            if (isNaN(nvp[1])) {
                value = nvp[1];  //a non number was provided as number,
                pass it to API as it is
            }
            else {
                value = Number(nvp[1]);
            }
        }
        else {
            value = nvp[1];
        }
        //Spezielle Behandlung des Werts NULL
        if (value && typeof value === 'string') {
            if (value.toUpperCase() === 'NULL') {
                value = null;
            }
        }
        nvpArray[i] = InteractAPI.NameValuePair.create(nvp[0], value, nvp[2]) ;
    }

    return nvpArray;
}

function showResponse(textDisplay) {
    var newWin = open('','Response','height=300,width=300,titlebar=no,
    scrollbars=yes,toolbar=no,
   resizable=yes,menubar=no,location=no,status=no');

    if (newWin.locationbar !== 'undefined' && newWin.locationbar
    && newWin.locationbar.visible)
        newWin.locationbar.visible = false;

    var displayHTML = '<META HTTP-EQUIV="Content-Type"
    CONTENT="text/html; charset=UTF-8">
    <html><head><style>TD { border-width : thin; border-style : solid }</style.'
                        + "<script language='Javascript'>"
                        + "var desiredDomain = 'unicacorp.com'; "
                        + "if (location.href.indexOf(desiredDomain)>=0) "
                        + "{ document.domain = desiredDomain;} "
                        + "</script></head><body> "
                        + textDisplay
                        + "</body></html>" ;
    newWin.document.body.innerHTML = displayHTML;
    newWin.focus() ;
}

function onSuccess(response) {
    showResponse("********Response********<br> " + JSON.stringify(response)) ;
}

function onError(response) {
    showResponse("********Error********<br> " + response) ;
}

function formatResoponse(response) {

}

function printBatchResponse(batResponse) {

}

function printResponse(response) {

}
```

Beispiel für JavaScript-Antwortobjekt "onSuccesss"

In diesem Beispiel werden drei Variablen für das JavaScript-Antwortobjekt erläutert: "offerLists", "messages" und "profile".

`offerList` gibt eine Liste ohne null zurück, wenn Sie `getOffer` oder `getOffersForMultipleInteractionPoints` als API oder als Bestandteil von Batchbefehlen aufrufen. Sie müssen für dieses Objekt immer null überprüfen, bevor Sie eine Operation für diese Variable ausführen.

Sie müssen immer den Status der JavaScript-Antwort `messages` prüfen.

Für `profile` wird nicht null zurückgegeben, wenn Sie `getProfile` als API oder als Bestandteil von Batchbefehlen verwenden. Wenn Sie `getProfile` nicht verwenden, können Sie diese Variable ignorieren. Sie müssen für dieses Objekt immer null überprüfen, bevor Sie eine Operation für diese Variable ausführen.

```
function onSuccess(response)
InteractAPI.ResponseTransUtil._buildResponse = function(response) {
        'use strict';

        if (!response) return null;

        var offerList = null;
        //offerLists in JS-Objekte umwandeln
        if (response.offerLists) {
            offerList = [];
            for (var ofListCt=0; ofListCt<response.offerLists.length;ofListCt++) {
                var ofListObj = this._buildOfferList(response.offerLists[ofListCt]);
                if (ofListObj) offerList.push(ofListObj);
            }
        }

        var messages = null;
        //messages in JS-Objekte umwandeln
        if (response.messages) {
            messages = [];
            for (var msgCt=0; msgCt<response.messages.length;msgCt++) {
                var msgObj = this._buildAdvisoryMessage(response.messages[msgCt]);
                if (msgObj) messages.push(msgObj);
            }
        }

        var profile = null;
        //profile-nvps in JS-Objekte umwandeln
        if (response.profile) {
            profile = [];
            for (var nvpCt=0; nvpCt<response.profile.length;nvpCt++) {
                var nvpObj = this._buildNameValuePair(response.profile[nvpCt]);
                if (nvpObj) profile.push(nvpObj);
            }
        }

        return InteractAPI.Response.create(response.sessionId,
                                           response.statusCode, offerList,
                                           profile, response.version,
                                           messages) ;
    };
```

Kapitel 9. Informationen zur ExternalCallout-API

Interact stellt ein erweiterbares Makro, EXTERNALCALLOUT, für die Verwendung mit Ihren interaktiven Ablaufdiagrammen bereit. Dieses Makro ermöglicht es Ihnen, angepasste Logik auszuführen, um mit externen Systemen während Ablaufdiagrammausführungen zu kommunizieren. Beispiel: Wenn Sie die Kreditbewertung eines Kunden während einer Ablaufdiagrammausführung berechnen wollen, können Sie eine Java-Klasse (ein Callout) dafür erstellen und dann das Makro EXTERNALCALLOUT in einem SELECT-Prozess in Ihrem interaktiven Ablaufdiagramm verwenden, um die Kreditbewertung von Ihrem Callout abzurufen.

Das Konfigurieren von EXTERNALCALLOUT besteht hauptsächlich aus zwei Schritten. Erstens müssen Sie eine Java-Klasse erstellen, die die ExternalCallout-API implementiert. Zweitens müssen Sie die notwendigen Marketing Platform-Konfigurationseigenschaften auf dem Laufzeitserver in der Kategorie Interact | flowchart | ExternalCallouts konfigurieren.

Zusätzlich zu den Informationen in diesem Abschnitt steht die JavaDoc für die ExternalCallout-API auf jedem Interact-Laufzeitserver im Verzeichnis Interact/docs/externalCalloutJavaDoc zur Verfügung.

IAffiniumExternalCallout-Benutzeroberfläche

Die ExternalCallout-API ist in der Benutzeroberfläche IAffiniumExternalCallout enthalten. Sie müssen die Benutzeroberfläche IAffiniumExternalCallout implementieren, um das Makro EXTERNALCALLOUT zu verwenden.

Die Klasse, die IAffiniumExternalCallout implementiert, sollte einen Konstruktor haben, mit dem sie durch den Laufzeitserver initialisiert werden kann.

- Wenn es keine Konstruktoren in der Klasse gibt, erstellt der Java-Compiler einen Standardkonstruktor, was ausreichend ist.
- Wenn es Konstruktoren mit Argumenten gibt, sollte ein öffentlicher Konstruktor ohne Argument bereitgestellt werden, der vom Laufzeitserver verwendet wird.

Bei der Entwicklung Ihres externen Callouts beachten Sie Folgendes:

- Jede Ausdrucksauswertung mit einem externen Callout erstellt eine neue Instanz der Klasse. Sie müssen Probleme mit der Threadsicherheit für statische Mitglieder in der Klasse steuern.
- Wenn Ihr externes Callout Systemressourcen wie Dateien oder eine Datenbankverbindung verwendet, müssen Sie diese Verbindungen verwalten. Der Laufzeitserver hat keine Funktion, um Verbindungen automatisch zu bereinigen.

Sie müssen Ihre Implementierung anhand von interact_externalcallout.jar im lib-Verzeichnis Ihrer IBM Interact-Laufzeitumgebungsinstallation kompilieren.

IAffiniumExternalCallout ermöglicht es dem Laufzeitserver, Daten aus Ihrer Java-Klasse anzufordern. Die Benutzeroberfläche besteht aus vier Methoden:

- getNumberOfArguments
- getValue
- initialize
- shutdown

Hinzufügen eines Web-Service zur Verwendung mit dem Makro EXTERNALCALLOUT

Mit diesem Verfahren fügen Sie einen Web-Service zur Verwendung mit dem Makro EXTERNALCALLOUT hinzu. Das Makro EXTERNALCALLOUT erkennt Callouts nur, wenn Sie die entsprechenden Konfigurationseigenschaften definiert haben.

Vorgehensweise

Fügen Sie in Marketing Platform für die Laufzeitumgebung folgende Konfigurationseigenschaften in der Kategorie Interact > Ablaufdiagramm > externalCallouts hinzu oder definieren Sie sie.

Konfigurationseigenschaft	Einstellung
Kategorie externalCallouts	Erstellen Sie eine Kategorie für Ihr externes Callout
class	Die Klassennamen für Ihr externes Callout
classpath	Der Klassenpfad zu den Klassendateien Ihres externen Callouts
Kategorie Parameter Data	Wenn Ihr externes Callout Parameter erfordert, erstellen Sie neue Parameterkonfigurationseigenschaften für sie und weisen Sie jeder einen Wert zu

getNumberOfArguments

Die getNumberOfArguments-Methode gibt die Anzahl an Argumenten zurück, die die Java-Klasse erwartet, die Sie zur Integration verwenden.

```
getNumberOfArguments()
```

Rückgabewert

Die getNumberOfArguments-Methode gibt eine Ganzzahl zurück.

Beispiel

Das folgende Beispiel zeigt das Drucken der Anzahl der Argumente.

```
public int getNumberOfArguments()
{
    return 0;
}
```

getValue

Die getValue-Methode führt die zentralen Funktionen des Aufrufs durch und gibt die Ergebnisse zurück.

```
getValue(audienceID, configData, arguments)
```

Die getValue-Methode benötigt die folgenden Parameter:

- **audienceID** - ein Wert, der die Zielgruppen-ID angibt.
- **configData** - eine Zuordnung mit Schlüssel/Wert-Paaren von Konfigurationsdaten, die der Aufruf benötigt.
- **arguments** - die Argumente, die der Aufruf benötigt. Jedes Argument kann eine Zeichenfolge, ein Doppelzeichen, ein Datum oder eine Liste daraus sein. Ein Listenargument kann Nullwerte enthalten, aber eine Liste kann zum Beispiel nicht eine Zeichenfolge und ein Doppelzeichen enthalten.

Sie sollten den Argumenttyp innerhalb der Implementierung überprüfen.

Wenn die getValue-Methode aus einem beliebigen Grund fehlschlägt, wird CalloutException zurückgegeben.

Rückgabewert

Die getValue-Methode gibt eine Liste mit Zeichenfolgen zurück.

Beispiel

```
public List<String> getValue(AudienceId audienceId, Map<String,
  String> configurationData, Object... arguments) throws CalloutException
{
 Long customerId = (Long) audienceId.getComponentValue("Customer");
 // now query scoreQueryUtility for the credit score of customerId
 Double score = scoreQueryUtility.query(customerId);
 String str = Double.toString(score);
 List<String> list = new LinkedList<String>();
 list.add(str);
 return list;
}
```

Initialisieren

Die Methode initialize wird beim Start des Laufzeitservers einmal aufgerufen. Alle eventuell vorhandenen Operationen, die die Leistung während der Laufzeit beeinträchtigen können, sollten durch diese Methode durchgeführt werden, zum Beispiel das Laden einer Datenbanktabelle.

```
initialize(configData)
```

Die initialize-Methode benötigt den folgenden Parameter:

- **configData** - eine Zuordnung mit Schlüssel/Wert-Paaren von Konfigurationsdaten, die der Aufruf benötigt.

  Interact liest diese Werte aus den Parametern, die in der Kategorie Interact > Flowchart > External Callouts > [External Callout] > Parameter Data für das externe Callout definiert sind.

Wenn die initialize-Methode aus einem beliebigen Grund fehlschlägt, wird CalloutException zurückgegeben.

Rückgabewert

Keiner.

Beispiel

```
public void initialize(Map<String, String> configurationData) throws CalloutException
{
    // configurationData has the key-value pairs specific to the environment
    // the server is running in
    // initialize scoreQueryUtility here
}
```

shutdown

Die shutdown-Methode wird beim Beenden des Laufzeitservers einmal aufgerufen. Alle eventuell erforderlichen Bereinigungsaufgaben sollten zu diesem Zeitpunkt ausgeführt werden.

```
shutdown(configData)
```

Die shutdown-Methode benötigt den folgenden Parameter:

- **configData** - eine Zuordnung mit Schlüssel/Wert-Paaren von Konfigurationsdaten, die der Aufruf benötigt.

Wenn die shutdown-Methode aus einem beliebigen Grund fehlschlägt, wird CalloutException zurückgegeben.

Rückgabewert

Keiner.

Beispiel

```
public void shutdown(Map<String, String> configurationData) throws CalloutException
{
    // shutdown scoreQueryUtility here
}
```

Beispiel für die ExternalCallout-API

In diesem Beispiel wird ein externes Callout erstellt, das eine Kreditbewertung abruft.

Erstellen Sie ein externes Callout, das eine Kreditbewertung abruft:

1. Erstellen Sie eine Datei namens GetCreditScore.java mit den folgenden Inhalten. Diese Datei setzt voraus, dass es eine Klasse namens ScoreQueryUtility gibt, die eine Bewertung aus einer Modellanwendung abruft.

```
import java.util.Map;
import com.unicacorp.interact.session.AudienceId;
import com.unicacorp.interact.flowchart.macrolang.storedobjs.IAffiniumExternalCallout;
import com.unicacorp.interact.flowchart.macrolang.storedobjs.CalloutException;
import java.util.Random;

public class GetCreditScore implements IAffiniumExternalCallout
{
 // the class that has the logic to query an external system for a customer's credit score
 private static ScoreQueryUtility scoreQueryUtility;
 public void initialize(Map<String, String> configurationData) throws CalloutException
 {
  // configurationData has the key- value pairs specific to the environment the server is running in
  // initialize scoreQueryUtility here
 }

 public void shutdown(Map<String, String> configurationData) throws CalloutException
 {
 // shutdown scoreQueryUtility here
 }

 public int getNumberOfArguments()
 {
  // do not expect any additional arguments other than the customer's id
  return 0;
 }

 public List<String> getValue(AudienceId audienceId, Map<String, String> configurationData,
  Object... arguments) throws CalloutException
 {
  Long customerId = (Long) audienceId.getComponentValue("Customer");
  // now query scoreQueryUtility for the credit score of customerId
  Double score = scoreQueryUtility.query(customerId);
  String str = Double.toString(score);
  List<String> list = new LinkedList<String>();
```

```
  list.add(str);
  return list;
 }
}
```

2. Kompilieren Sie `GetCreditScore.java` zu `GetCreditScore.class`.
3. Erstellen Sie eine JAR-Datei namens `creditscore.jar`, die `GetCreditScore.class` und die anderen verwendeten Klassendateien enthält.
4. Kopieren Sie die JAR-Datei an eine Position auf dem Laufzeitserver, z. B. `/data/interact/creditscore.jar`.
5. Erstellen Sie ein externes Callout mit dem Namen `GetCreditScore` und dem Klassenpfad`/data/interact/creditscore.jar` in der Kategorie `externalCallouts` auf der Seite **Konfigurationen verwalten**.
6. In einem interaktiven Ablaufdiagramm kann das Callout als `EXTERNALCALLOUT('GetCreditScore')` verwendet werden.

Benutzeroberfläche IInteractProfileDataService

Die Profildatenservices-API ist in der Benutzeroberfläche `iInteractProfileDataService` enthalten. Diese Benutzeroberfläche ermöglicht Ihnen, über eine oder mehrere externe Datenquellen (z. B. eine Flachdatei, einen Web-Service usw.) hierarchische Daten in eine Interact-Sitzung zu importieren, wenn die Interact-Sitzung startet oder wenn sich die Zielgruppen-ID einer Interact-Sitzung ändert.

Um das Importieren hierarchischer Daten mithilfe der Profildatenservices-API zu entwickeln, müssen Sie eine Java-Klasse schreiben, die Informationen von allen Datenquellen abruft und einem `ISessionDataRootNode`-Objekt zuordnet. Danach müssen Sie mithilfe des Makros EXTERNALCALLOUT in einem Prozess "Auswählen" für ein interaktives Ablaufdiagramm auf die zugeordneten Daten verweisen.

Sie müssen Ihre Implementierung anhand von `interact_externalcallout.jar` im `lib`-Verzeichnis Ihrer IBM Interact-Laufzeitumgebungsinstallation kompilieren.

Eine vollständige JavaDoc-Dokumentation zur Verwendung dieser Benutzeroberfläche finden Sie in den Dateien *Interact_home*`/docs/externalCalloutJavaDoc`, auf die Sie in jedem Web-Browser zugreifen können.

Eine Beispielimplementierung zur Verwendung des Profildatenservices einschließlich kommentierter Beschreibungen, wie das Beispiel implementiert wurde, finden Sie unter *Interact_home*`/samples/externalcallout/XMLProfileDataService.java`.

Anmerkung: Die Beispielimplementierung soll nur als Beispiel dienen. Sie sollten dieses Beispiel nicht in Ihrer Implementierung verwenden.

Hinzufügen einer Datenquelle zur Verwendung mit Profildatenservices

Mit diesem Verfahren fügen Sie eine Datenquelle zur Verwendung mit den Profildatenservices hinzu.

Informationen zu diesem Vorgang

Das Makro `EXTERNALCALLOUT` erkennt eine Datenquelle für den Import hierarchischer Daten mithilfe der Profildatenservices nur, wenn Sie die entsprechenden Konfigurationseigenschaften definiert haben.

Vorgehensweise

Fügen Sie in Marketing Platform für die Laufzeitumgebung die folgenden Konfigurationseigenschaften in der Kategorie `Interact > profile > Audience Levels > [AudienceLevelName] > Profile Data Services` hinzu oder definieren Sie sie.

Konfigurationseigenschaft	Einstellung
Kategorie `Neuer Kategoriename`	Der Name der Datenquelle, die Sie definieren. Der Name, den Sie hier eingeben, muss innerhalb der Datenquellen einer Zielgruppenebene eindeutig sein.
`enabled`	Gibt an, ob die Datenquelle für die Zielgruppenebene aktiviert ist, in der sie definiert ist.
`className`	Der vollständig qualifizierte Name der Datenquellenklasse, die `IInteractProfileDataService` implementiert.
`classPath`	Der Klassenpfad zu Ihren Klassendateien in den Profildatenservices. Wenn Sie ihn auslassen, wird standardmäßig der Klassenpfad des übergeordneten Anwendungsservers verwendet.
Kategorie `priority`	Die Priorität dieser Datenquelle in dieser Zielgruppenebene. Der Wert muss für jede Datenquelle in einer Zielgruppenebene eindeutig sein. (Wenn also für eine Datenquelle die Priorität 100 festgelegt ist, kann keine weitere Datenquelle in der Zielgruppenebene eine Priorität von 100 haben.)

Benutzeroberfläche IParameterizableCallout

Die parametrisierbare Aufruf-API ist in der Benutzeroberfläche `IParameterizableCallout` enthalten.

Diese Benutzeroberfläche ist die Basisbenutzeroberfläche der zugänglichen APIs, die Parameter von der Konfiguration über Marketing Platform akzeptieren können. Da dies eine Basisbenutzeroberfläche ist, sollte sie nicht direkt implementiert werden. Die Parameter werden von den untergeordneten Knoten des Knotens `Parameterdaten` in der Kategorie abgerufen, in der auf diese Implementierung verwiesen wird. Im folgenden Beispiel stellt ESB eine angepasste Implementierung des Profildatenservice dar, über den wiederum die Benutzeroberfläche `IParameterizableCallout` implementiert wird. Die Parameter endPoint und login werden zusammen mit den zugehörigen Werten in diese Implementierungsklasse übergeben, wenn die Interact-Engine versucht, diese zu initialisieren und zu beenden.

```
Profile Data Services
...ESB
   ...Parameter Data
      ...endPoint
      ...login
```

Die Benutzeroberfläche umfasst zwei Methoden:

- `initialize`
- `shutdown`

initialize

Mit der Methode `initialize` wird diese Implementierungsklasse initialisiert.

```
void initialize(java.util.Map<java.lang.String,java.lang.String> configurationData)
                throws CalloutException
```

Für die Methode initialize ist der folgende Parameter erforderlich:

- **configurationData** - Eine Zuordnung mit Name/Wert-Paaren der von Benutzern konfigurierten Parameter

Löst aus:

CalloutException

shutdown

Bei der Methode shutdown wird diese Implementierungsklasse beendet.

```
void shutdown(java.util.Map<java.lang.String,java.lang.String> configurationData)
              throws CalloutException
```

Die shutdown-Methode benötigt den folgenden Parameter:

- **configurationData** - Eine Zuordnung mit Name/Wert-Paaren der von Benutzern konfigurierten Parameter

Löst aus:

CalloutException

Benutzeroberfläche ITriggeredMessageAction

Die API der Aktion für ausgelöste Nachrichten ist in der Benutzeroberfläche ITriggeredMessageAction enthalten. Diese Benutzeroberfläche ermöglicht es Ihnen, den Namen dieser Instanz abzurufen und festzulegen.

Die Benutzeroberfläche ITriggeredMessageAction dient als Basisbenutzerschnittstelle für andere Benutzeroberflächen und sollte niemals direkt implementiert werden.

Die Benutzeroberfläche umfasst zwei Methoden:

- getName
- setName

getName

Mit der Methode getName wird der Name der Instanz ITriggeredMessageAction zurückgegeben.

```
java.lang.String getName()
```

setName

Mit der Methode setName wird der Name der Instanz ITriggeredMessageAction festgelegt.

```
void setName(java.lang.String name)
```

Während Sie die Implementierungsklasse dieser Benutzeroberfläche initialisieren, legt Interact den Namen der Benutzeroberfläche mit dem in der Konfigurations-UI angegebenen Namen fest.

Im folgenden Beispiel lautet der Name dieses Gateways InteractLog.

```
triggeredMessage
          ...gateways
             ...InteractLog
```

Für die Methode setName ist der folgende Parameter erforderlich:

- name - Der Name, den Sie für die Instanz ITriggeredMessageAction festlegen möchten.

Benutzeroberfläche IChannelSelector

Die Channel Selector-API ist in der Benutzeroberfläche IChannelSelector enthalten. Diese Benutzeroberfläche ermöglicht es Ihnen, die Kanäle für abgehende Nachrichten basierend auf dem zu sendenden Angebot und den Sitzungsattributen auszuwählen.

Eine Beispielimplementierung zur Verwendung der Aktion für ausgelöste Nachrichten, einschließlich kommentierter Beschreibungen zur Vorgehensweise bei der Implementierung des Beispiels, finden Sie unter *Interact_home*/samples/triggeredmessage/SampleChannelSelector.java.

Anmerkung: Die Beispielimplementierung soll nur als Beispiel dienen. Sie sollten dieses Beispiel nicht in Ihrer Implementierung verwenden.

Sie sollten versuchen, diese Implementierung zu verwenden, statt Ihre eigene Implementierung zu schreiben.

Die Benutzeroberfläche umfasst eine Methode:

- selectChannels

selectChannels

Bei der Methode selectChannels werden die Kanäle für abgehende Nachrichten ausgewählt, an die das übergebene Angebot über die Benutzeroberfläche IChannelSelector gesendet werden sollte.

```
java.util.List<java.lang.String> selectChannels
                    (java.util.Map<java.lang.String,java.util.Map<java.lang.String,
                                   java.lang.Object>> availableChannels,
                                   com.unicacorp.interact.api.Offer offer,
                                   com.unicacorp.interact.treatment.
                                   optimization.IInteractSessionData sessionData)
```

Interact versucht, dieses Angebot an alle diese zurückgegebenen Kanäle zu senden.

Für die Methode selectChannels sind folgende Parameter erforderlich:

- **availableChannels** - Eine Übersicht über die verfügbaren Kanäle für abgehende Nachrichten, die in der UI für ausgelöste Nachrichten in den Einstellungen für die Entwurfszeit von Interact konfiguriert werden. In jedem Eintrag der Übersicht umfasst der Schlüssel den Namen des Kanals und der Wert die konfigurierten Parameter dieses Kanals in der Designzeit von Interact. Die Iterationsreihenfolge in dieser Übersicht entspricht der in dieser UI definierten Reihenfolge. Wenn der vom Profil bevorzugte Kanal in der UI für ausgelöste Nachrichten verwendet wird, wird er durch den tatsächlichen Kanal ersetzt, bevor diese Methode aufgerufen wird. Zusätzlich bleibt nur das Vorkommen mit der höchsten Priorität erhalten und alle Duplikate werden entfernt, wenn in der UI der gleiche Kanal mehrmals auftaucht.
- **offer** - Das Angebot, das bereitgestellt werden soll
- **sessionData** - Die Attribute, die derzeit in der zugehörigen Interact-Sitzung gespeichert sind

Benutzeroberfläche IDispatcher

Die Dispatcher-API ist in der Benutzeroberfläche `IDispatcher` enthalten. Diese Benutzeroberfläche sendet Angebote an zielgruppenspezifische Gateways.

Da es für jeden konfigurierten Dispatcher nur eine Instanz dieser Klasse gibt, muss die Implementierung dieser Benutzeroberfläche statusunabhängig aus der Perspektive von Interact erfolgen.

Eine Beispielimplementierung zur Verwendung der Aktion für ausgelöste Nachrichten, einschließlich kommentierter Beschreibungen zur Vorgehensweise bei der Implementierung des Beispiels, finden Sie unter *Interact_home*`/samples/triggeredmessage/SampleDispatcher.java`.

Anmerkung: Die Beispielimplementierung soll nur als Beispiel dienen. Sie sollten dieses Beispiel nicht in Ihrer Implementierung verwenden.

Sie sollten versuchen, diese Implementierung zu verwenden, statt Ihre eigene Implementierung zu schreiben.

Die Benutzeroberfläche umfasst eine Methode:

- `dispatch`

dispatch

Bei der Methode `dispatch` werden Angebote an die Zielgateways der Benutzeroberfläche `IDispatcher` gesendet.

```
boolean dispatch(java.lang.String channel,
            java.lang.String gatewayName,
            java.util.Collection<com.unicacorp.interact.api.Offer> offers,
            com.unicacorp.interact.api.NameValuePair[] profileData)
              throws com.unicacorp.interact.exceptions.InteractException
```

Sobald Kanäle für abgehende Nachrichten für ein mögliches Angebot ausgewählt wurden, versucht Interact, das mögliche Angebot an die Handler zu senden, die dem Kanal zugeordnet sind. Die Handler werden basierend auf ihren definierten Prioritäten von hoch zu niedrig festgelegt. Interact ruft diese Methode des konfigurierten Dispatchers für jeden Handler auf. Es hängt von der Implementierung dieser Dispatcherinstanz ab, wie das Angebot an das Zielgateway weitergeleitet wird, das bei demselben Handler konfiguriert ist. Wenn demselben Handler aufgrund der gleichen Bewertung ausgelöster Nachrichten mehrere Angebote gesendet werden, versucht Interact, alle diese Angebote in einem Batch zu senden.

Für die Methode `dispatch` sind die folgenden Parameter erforderlich:

- **channel** - Der Kanal für abgehende Nachrichten, an den diese Angebote gesendet werden
- **gatewayName** - Der Name des Zielgateways
- **offers** - Die Angebote, die in einem Stapel an das Gateway gesendet werden sollen
- **profileData** - Von `IGateway.validate` aufgefüllte Profilattribute, die an `IGateway.deliver` übergeben werden

Rückgabewert

Die Methode `dispatch` gibt den Rückgabewert zurück, wenn das Senden erfolgreich war oder fehlgeschlagen ist

Löst aus:

`com.unicacorp.interact.exceptions.InteractException`

Benutzeroberfläche IGateway

Die Gateway-API ist in der Benutzeroberfläche `IGateway` enthalten. Diese Benutzeroberfläche erhält Angebote von Interact und sendet die Angebote an ihr Ziel.

Jede Implementierung dieser Benutzeroberfläche kommuniziert mit einem bestimmten Ziel. Am Ziel müssen die notwendige Datentransformation, Attributauffüllung und ähnliche zielbezogene Aufgaben ausgeführt werden.

Eine Beispielimplementierung zur Verwendung der Aktion für ausgelöste Nachrichten, einschließlich kommentierter Beschreibungen zur Vorgehensweise bei der Implementierung des Beispiels, finden Sie unter *Interact_home*`/samples/triggeredmessage/SampleOutboundGateway.java`.

Anmerkung: Die Beispielimplementierung soll nur als Beispiel dienen. Sie sollten dieses Beispiel nicht in Ihrer Implementierung verwenden.

Die Benutzeroberfläche umfasst zwei Methoden:

- `deliver`
- `validate`

Deliver

Die Methode `deliver` wird aufgerufen, wenn das Angebot bzw. die Angebote an ein Ziel in der Benutzeroberfläche `IGateway` gesendet werden soll(en).

```
void deliver(java.util.Collection<com.unicacorp.interact.api.Offer> offers,
                    com.unicacorp.interact.api.NameValuePair[] profileData,
                    java.lang.String channel)
```

Für die Methode `deliver` sind folgende Parameter erforderlich:

- **offers** - Das Angebot, das gesendet werden soll
- **profileData** - Die mit der Methode "Validate" in `parameterMap` aufgefüllten Profilattribute
- **channel** - Der Kanal für abgehende Nachrichten, an den diese Angebote gesendet werden

Validate

Mit der Methode `validate` werden mögliche Angebote in der Benutzeroberfläche `IGateway` validiert.

```
java.util.Collection<com.unicacorp.interact.api.Offer> validate
    (com.unicacorp.interact.treatment.optimization.
     IInteractSessionData sessionData,
           java.util.Collection<com.unicacorp.interact.api.Offer> candidateOffers,
           java.util.Map<java.lang.String,java.lang.Object> parameterMap,
           java.lang.String channel)
```

Die Interact-Engine ruft diese Methode zur Validierung der möglichen Angebote auf. Bei der Implementierung dieser Methode sollten die Angebote, Angebotsattribute und Sitzungsattribute auf die Voraussetzungen des Ziels geprüft werden, um zu ermitteln, welches Angebot bzw. welche Angebote über dieses Gateway gesen-

det werden kann/können. Zusätzlich könnten notwendige Parameter zur übergebenen Übersicht hinzugefügt werden, die wieder an die Methode zur Bereitstellung übergeben wird.

Für die Methode `validate` sind folgende Parameter erforderlich:

- **sessionData** - die Attribute, die derzeit in der zugehörigen Interact-Sitzung gespeichert sind
- **candidateOffers** - die Angebote, die basierend auf der Methode zur Angebotsauswahl, den zugehörigen Parametern und weiteren Faktoren von Interact ausgewählt wurden. Diese Angebote kommen aus der Perspektive von Interact für die Bereitstellung infrage, allerdings vorbehaltlich des Gateways.
- **parameterMap** - eine Übersicht, die bei der Implementierung dieser Methode zur Übergabe von Parametern an die Methode zur Bereitstellung verwendet werden sollte
- **channel** - der Kanal für abgehende Nachrichten, an den diese Angebote gesendet werden

Kapitel 10. IBM Interact-Dienstprogramme

In diesem Abschnitt werden die Administrationsdienstprogramme beschrieben, die mit Interact verfügbar sind.

Dienstprogramm RunDeployment (runDeployment.sh/.bat)

Mithilfe des Befehlszeilentools `runDeployment` können Sie von der Befehlszeile aus einen interaktiven Kanal für eine bestimmte Servergruppe implementieren. Verwenden Sie dazu die Einstellungen in der Datei `deployment.properties`, die alle möglichen Parameter beschreibt. Sie befindet sich im selben Verzeichnis wie das Tool `runDeployment`. Die Möglichkeit, von der Befehlszeile aus einen interaktiven Kanal zu implementieren, ist besonders nützlich, wenn Sie die Funktion `OffersBySQL` verwenden. Sie können z. B. ein Campaign-Batch-Flowchart konfigurieren, das regelmäßig ausgeführt wird. Wenn die Ausführung des Flowcharts abgeschlossen ist, kann ein Trigger aufgerufen werden, der die Implementierung der Angebote in der `OffersBySQL`-Tabelle mithilfe dieses Befehlszeilentools initialisiert.

Beschreibung

Sie finden das Befehlszeilentool `runDeployment`, das automatisch auf dem Interact-Entwicklungszeitserver installiert wird, im folgenden Verzeichnis:

Interact_home`/interactDT/tools/deployment/runDeployment.sh` (oder `runDeployment.bat` auf einem Windows-Server)

Das einzige Argument, das dem Befehl übergeben wird, ist die Position der Datei `deployment.properties`, die alle möglichen Parameter beschreibt, die zum Implementieren der Kombination von interaktivem Kanal und Laufzeitservergruppe erforderlich sind. Zu Referenzzwecken ist eine Beispieldatei verfügbar.

Anmerkung: Bevor Sie das Dienstprogramm `runDeployment` verwenden, müssen Sie es zunächst in einem beliebigen Texteditor bearbeiten, damit es die Position der Java-Laufzeitumgebung auf dem Server angibt. Sie können z. B. den Pfad *Interact_home*`/jre` oder *Platform_home*`/jre` angeben, wenn eines dieser Verzeichnisse die Java-Laufzeitumgebung enthält, die vom Dienstprogramm verwendet werden soll. Stattdessen können Sie auch den Pfad zu jeder beliebigen Java-Laufzeitumgebung angeben, deren Verwendung mit diesem Release der IBM Produkte unterstützt wird.

Verwenden des Dienstprogramms runDeployment in einer sicheren SSL-Umgebung

Damit Sie das Dienstprogramm „runDeployment“ verwenden können, wenn auf dem Interact-Server Sicherheitsfunktionen aktiviert wurden (und Verbindungen daher über einen SSL-Port hergestellt werden), müssen Sie die folgenden Schritte ausführen, um die Java-TrustStore-Eigenschaft hinzuzufügen:

1. Wenn Sie die Datei `deployment.properties` für Ihre Implementierung des interaktiven Kanals bearbeiten, ändern Sie die Eigenschaft `deploymentURL` so, dass die sichere SSL-URL verwendet wird, so wie im folgenden Beispiel:

   `deploymentURL=https://<`*HOST*`>.<`*DOMAIN*`>:<`*PORT*`>/Campaign/interact/InvokeDeploymentServlet`

2. Bearbeiten Sie das Script runDeployment.sh bzw. runDeployment.bat mithilfe eines beliebigen Texteditors, um das folgende Argument zu der Zeile hinzuzufügen, die mit ${JAVA_HOME} beginnt:

   -Djavax.net.ssl.trustStore=<*TrustStorePath*>

   Die Zeile könnte z. B. so aussehen, nachdem Sie das TrustStore-Argument hinzugefügt haben:

   ```
   ${JAVA_HOME}/bin/java -Djavax.net.ssl.trustStore=<TrustStorePath>
   -cp ${CLASSPATH}com.unicacorp.Campaign.interact.deployment.tools.
   InvokeDeploymentClient $1
   ```

   Ersetzen Sie <*TrustStorePath*> durch den Pfad zum tatsächlichen SSL-geschützten Truststore.

Ausführen des Dienstprogramms

Nachdem Sie das Dienstprogramm so bearbeitet haben, dass es die Java-Laufzeitumgebung angibt, und eine Kopie der Datei deployment.properties an Ihre Umgebung angepasst haben, können Sie das Dienstprogramm mit dem folgenden Befehl ausführen:

Interact_home/interactDT/tools/deployment/runDeployment.sh *deployment.properties*

Ersetzen Sie *Interact_home* durch den tatsächlichen Wert der Interact-Entwicklungszeitinstallation und ersetzen Sie *deployment.properties* durch den tatsächlichen Pfad und den Namen der Eigenschaftendatei, die Sie für diese Implementierung angepasst haben.

Beispieldatei deployment.properties

Die Beispieldatei deployment.properties enthält eine kommentierte Liste aller Parameter, die Sie anpassen müssen, damit sie mit Ihrer Umgebung übereinstimmen. Die Beispieldatei enthält darüber hinaus Anmerkungen, die die einzelnen Parameter erklären und angeben, warum Sie einen bestimmten Wert möglicherweise anpassen müssen.

```
##############################################################################
#
# Die folgenden Eigenschaften werden in das Programm InvokeDeploymentClient
# eingegeben. Das Programm sucht nach der Einstellung für deploymentURL.
# Das Programm sendet eine Anfrage an diese URL; alle weiteren Einstellungen
# werden als Parameter in dieser Anfrage gesendet.
# Das Programm überprüft dann den Status der Implementierung und
# kehrt zurück, wenn die Implementierung abgeschlossen ist (oder der
# angegebene Wert für waitTime erreicht ist).
#
# Die Ausgabe des Programms erfolgt in diesem Format:
# <STATE> : <Misc Detail>
#
# wobei der Wert für STATE einer der folgenden sein kann:
# ERROR
# RUNNING
# SUCCESS
#
# Die Daten in Misc Detail füllen normalerweise den Statusnachrichtenbereich
# in der Implementierungs-GUI auf der Übersichtsseite des interaktiven Kanals
# auf.
# HINWEIS: In Misc Detail können HTML-Tags vorhanden sein
#
##############################################################################
```

```
################################################################################
# deploymentURL: URL zum Servlet InvokeDeployment, das sich in der Interact-
# Entwicklungszeit befindet.  Sollte folgendes Format haben:
# http://dt_host:port/Campaign/interact/InvokeDeploymentServlet
################################################################################
deploymentURL=http://localhost:7001/Campaign/interact/InvokeDeploymentServlet

################################################################################
# dtLogin:  Mit dieser Anmeldung würden Sie sich am Entwicklungszeitserver
# anmelden, wenn Sie den interaktiven Kanal über die Implementierungs-GUI
# auf der Übersichtsseite des interaktiven Kanals implementieren würden.
################################################################################
dtLogin=asm_admin

################################################################################
# dtPW:  Das Kennwort für dtLogin
################################################################################
dtPW=

################################################################################
# icName:  Der Name des interaktiven Kanals, den Sie implementieren möchten
################################################################################
icName=ic1

################################################################################
# partition:  Der Name der Partition
################################################################################
partition=partition1

################################################################################
# request:  Dies ist die Art der Anforderung, die dieses Tool aktuell ausführen
# soll. Dabei sind zwei Verhaltensweisen möglich.  Ist der Wert deploy, so
# wird die Implementierung ausgeführt.  Alle anderen Werte führen dazu, dass
# das Tool nur den Status der letzten Implementierung des angegebenen
# interaktiven Kanals zurückgibt.
################################################################################
request=deploy

################################################################################
# serverGroup:  Der Name der Servergruppe, auf der Sie den interaktiven Kanal
# implementieren möchten.
################################################################################
serverGroup=defaultServerGroup

################################################################################
# serverGroupType:  Gibt an, ob diese Implementierung auf einer
# Produktionsservergruppe oder einer Testservergruppe durchgeführt wird.
# 1 bezeichnet Produktion,
# 2 bezeichnet Test.
################################################################################
serverGroupType=1

################################################################################
# rtLogin:  Das Konto für die Authentifizierung an der Servergruppe,
# auf der Sie die Implementierung durchführen.
################################################################################
rtLogin=asm_admin

################################################################################
# rtPW:  Das zu rtLogin zugehörige Kennwort
################################################################################
rtPW=

################################################################################
# waitTime:  Sobald das Tool die Implementierungsanforderung übergeben hat,
# überprüft es den Status der Implementierung.  Wenn die Implementierung nicht
# abgeschlossen (oder fehlgeschlagen) ist, fragt das Tool den Status weiterhin
```

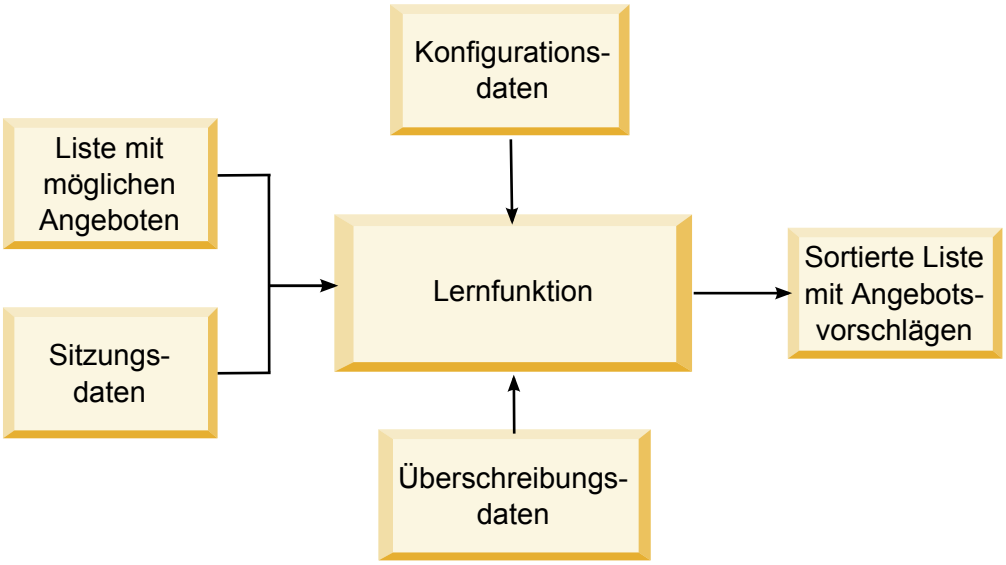
```
# beim System ab, bis der Status abgeschlossen erreicht ist ODER bis der
# angegebene Wert für waitTime (in Sekunden) erreicht ist.
##############################################################################
waitTime=5

##############################################################################
# pollTime: Wenn der Status einer Implementierung noch Laufstatus ist,
# überprüft das Tool den Status weiterhin.  Zwischen den Statusüberprüfungen
# ist das Tool für einige Sekunden inaktiv, basierend auf der Einstellung für
# pollTime.
##############################################################################
pollTime=3

##############################################################################
# global: Die Einstellung false führt dazu, dass das Tool die globalen
# Einstellungen NICHT implementiert. Ist die Eigenschaft nicht verfügbar,
# werden die globalen Einstellungen dennoch implementiert.
##############################################################################
global=true
```

Kapitel 11. Informationen zur Lern-API

Interact stellt ein Lernmodul bereit, das einen naiven bayesschen Algorithmus verwendet, um Besucheraktionen zu überwachen und um optimale Angebote (in Bezug auf Annahme) vorzuschlagen. Sie können dieselbe Java-Benutzeroberfläche mit Ihren eigenen Algorithmen unter Verwendung der Lern-API implementieren, um Ihr eigenes Lernmodul zu erstellen.

Anmerkung: Wenn Sie externes Lernen verwenden, geben die Beispielberichte in Bezug auf die Lernfunktion (Berichte "Lerndetails des interaktiven Angebots" und " Steigerungsanalyse von interaktiven Segmenten") keine gültigen Daten zurück.

Auf der einfachsten Ebene stellt die Lern-API Methoden bereit, um Daten von der Laufzeitumgebung zu erfassen und eine geordnete Liste von empfohlenen Angeboten zurückzugeben.



Sie können die folgenden Daten von Interact erfassen:

- Angebotskontaktdaten
- Angebotsannahmedaten
- Alle Sitzungsdaten
- Campaign-spezifische Angebotsdaten
- Konfigurationseigenschaften, die in der Kategorie `learning` für die Designumgebung und in der Kategorie `offerserving` für die Laufzeitumgebung definiert sind

Sie können diese Daten in Ihren Algorithmen verwenden, um eine Liste von vorgeschlagenen Angeboten zu erstellen. Sie geben dann eine Liste von empfohlenen Angeboten in der Reihenfolge von höchster zu niedrigster Empfehlung zurück.

Sie können auch die Lern-API verwenden, um Daten für Ihre Lernimplementierung zu erfassen (im Diagramm nicht angezeigt). Sie können diese Daten im Speicher ablegen oder sie in einer Datei oder Datenbank für spätere Analyse protokollieren.

Nach der Erstellung Ihrer Java-Klassen können Sie sie in eine JAR-Datei konvertieren. Nachdem Sie JAR-Dateien erstellt haben, müssen Sie die Laufzeitumgebung auch konfigurieren, Ihr externes Lernmodul zu erkennen, indem Sie Konfigurationseigenschaften bearbeiten. Sie müssen Ihre Java-Klassen oder JAR-Dateien auf jeden Laufzeitserver kopieren, der Ihr externes Lernmodul verwendet.

Zusätzlich zu den Informationen in diesem Abschnitt steht die JavaDoc für die Lernoptimierungsprogramm-API auf jedem Laufzeitserver im Verzeichnis `Interact/docs/learningOptimizerJavaDoc` zur Verfügung.

Sie müssen Ihre Implementierung anhand von `interact_learning.jar` im `lib`-Verzeichnis Ihrer Interact-Laufzeitumgebungsinstallation kompilieren.

Wenn Sie Ihre benutzerdefinierte Lernimplementierung schreiben, sollten Sie die folgenden Richtlinien berücksichtigen.

- Die Leistung ist entscheidend.
- Muss mit Multithreading funktionieren und Thread-sicher sein.
- Muss alle externen Ressourcen steuern, unter Berücksichtigung von Fehlermodi und Leistung.
- Verwenden Sie entsprechende Ausnahmebedingungen, entsprechende Protokollierung (log4j) und entsprechenden Speicher.

Konfigurieren der Laufzeitumgebung für die Erkennung von externen Lernmodulen

Sie können die Lern-Java-API verwenden, um Ihr eigenes Lernmodul zu schreiben. Sie müssen die Laufzeitumgebung konfigurieren, um Ihr Lerndienstprogramm in Marketing Platform zu erkennen.

Informationen zu diesem Vorgang

Sie müssen den Interact-Laufzeitserver erneut starten, damit diese Änderungen wirksam werden.

Vorgehensweise

1. Bearbeiten Sie in Marketing Platform für die Laufzeitumgebung die folgenden Konfigurationseigenschaften in der Kategorie `Interact > offerserving`. Die Konfigurationseigenschaften für die Lernoptimierungsprogramm-API befinden sich in der Kategorie `Interact > offerserving > External Learning Config`.

Konfigurationseigenschaft	Einstellung
`optimizationType`	**ExternalLearning**
`externalLearningClass`	Klassenname für das externe Lernen
`externalLearningClassPath`	Der Pfad zu den Klassen- oder JAR-Dateien auf dem Laufzeitserver für das externe Lernen. Wenn Sie eine Servergruppe verwenden und alle Laufzeitserver dieselbe Instanz von Marketing Platform referenzieren, muss jeder Server über eine Kopie der Klassen- oder JAR-Dateien an derselben Position verfügen.

2. Starten Sie den Interact-Laufzeitserver erneut, damit diese Änderungen wirksam werden.

Benutzeroberfläche ILearning

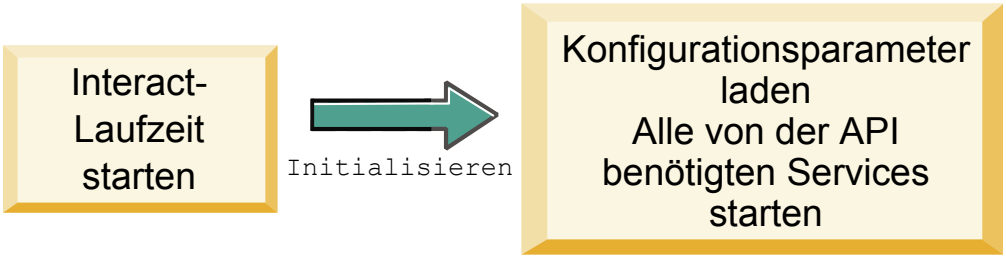
Die Lern-API wird um die Benutzeroberfläche `ILearning` aufgebaut. Sie müssen die Benutzeroberfläche `ILearning` implementieren, um die angepasste Logik Ihres Lernmoduls zu unterstützen.

Unter Anderem ermöglicht es Ihnen die Benutzeroberfläche `ILearning`, Daten aus der Laufzeitumgebung für Ihre Java-Klasse zu erfassen und Angebote zurück an den Laufzeitserver zu senden.

Initialisieren

Die Methode `initialize` wird beim Start des Laufzeitservers einmal aufgerufen. Wenn es Operationen gibt, die nicht wiederholt werden müssen, aber möglicherweise die Leistung während der Laufzeit einschränken, wie z. B. das Laden von statischen Daten aus einer Datenbanktabelle, dann sollten sie durch diese Methode ausgeführt werden.

```
initialize(ILearningConfig config, boolean debug)
```



- **config** - ein `ILearningConfig`-Objekt definiert alle für die Lernfunktion relevanten Konfigurationseigenschaften.
- **debug** - ein boolescher Ausdruck. Wenn `true` gibt er an, dass die Ausführlichkeit der Protokollstufe für das Laufzeitumgebungssystem auf "debug" festgelegt ist. Es wird empfohlen, dass Sie diesen Wert auswählen, bevor in ein Protokoll geschrieben wird.

Wenn die Methode `initialize` aus irgendeinem Grund fehlschlägt, wird eine `LearningException` ausgelöst.

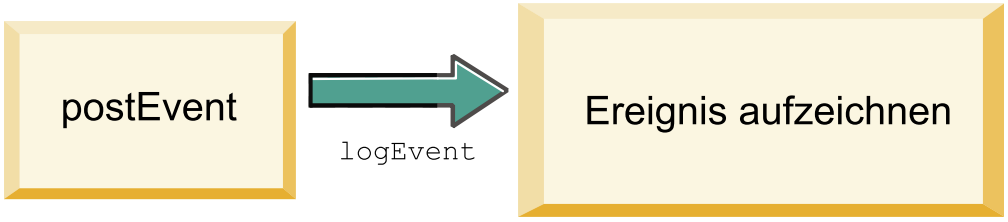
Rückgabewert

Keiner.

logEvent

Die Methode `logEvent` wird vom Laufzeitserver aufgerufen, wenn die Interact-API ein Ereignis bereitstellt, das konfiguriert ist, als ein Kontakt oder als eine Antwort protokolliert zu werden. Verwenden Sie diese Methode, um Kontakt- und Antwortdaten in eine Datenbank oder Datei für Berichterstellungs- oder Lernzwecke zu protokollieren. Beispiel: Wenn Sie algorithmisch die Wahrscheinlichkeit, dass ein Kunde ein Angebot annimmt, basierend auf Kriterien bestimmen wollen, verwenden Sie diese Methode, um die Daten zu protokollieren.

```
logEvent(ILearningContext context,
             IOffer offer,
             IClientArgs clientArgs,
             IInteractSession session,
             boolean debug)
```



- **context** - ein `ILearningContext`-Objekt, das den Lernkontext des Ereignisses definiert, z. B. Kontakt, Annehmen oder Ablehnen.
- **offer** - ein `IOffer`-Objekt, das das Angebot definiert, zu dem dieses Ereignis protokolliert wird.
- **clientArgs** - ein `IClientArgs`-Objekt, das Parameter definiert. Derzeit erfordert `logEvent` keine `clientArgs`, daher ist dieser Parameter möglicherweise leer.
- **session** - ein `IInteractSession`-Objekt, das alle Sitzungsdaten definiert.
- **debug** - ein boolescher Ausdruck. Wenn `true` gibt er an, dass die Ausführlichkeit der Protokollstufe für das Laufzeitumgebungssystem auf "debug" festgelegt ist. Es wird empfohlen, dass Sie diesen Wert auswählen, bevor in ein Protokoll geschrieben wird.

Wenn die Methode `logEvent` fehlschlägt, wird eine `LearningException` ausgelöst.

Rückgabewert

Keiner.

optimizeRecommendList

Die Methode `optimizeRecommendList` sollte eine Liste von empfohlenen Angeboten und die Sitzungsdaten nehmen und eine Liste zurückgeben, die die angeforderte Anzahl von Angeboten enthält. Die Methode `optimizeRecommendList` sollte die Angebote auf eine bestimmte Art mit Ihrem eigenen Lernalgorithmus sortieren. Die Liste der Angebote muss so sortiert sein, dass die Angebote, die Sie zuerst anbieten wollen, sich am Anfang der Liste befinden. Beispiel: Wenn Ihr Lernalgorithmus den besten Angeboten eine niedrige Bewertung gibt, sollten die Angebote 1, 2, 3 sortiert sein. Wenn Ihr Lernalgorithmus den besten Angeboten einee hohe Bewertung gibt, sollten die Angebote 100, 99, 98 sortiert sein.

```
optimizeRecommendList(list(ITreatment) recList,
  IClientArgs clientArg, IInteractSession session,
  boolean debug)
```

Die Methode `optimizeRecommendList` erfordert die folgenden Parameter:

- **recList** - eine Liste der Verfahrensobjekte (Angebote), die von der Laufzeitumgebung empfohlen werden.

- **clientArg** - ein IClientArgs-Objekt, das zumindest die Anzahl der Angebote enthält, die von der Laufzeitumgebung angefordert werden.
- **session** - ein IInteractSession-Objekt, das alle Sitzungsdaten enthält.
- **debug** - ein boolescher Ausdruck. Wenn `true` gibt er an, dass die Ausführlichkeit der Protokollstufe für das Laufzeitumgebungssystem auf "debug" festgelegt ist. Es wird empfohlen, dass Sie diesen Wert auswählen, bevor in ein Protokoll geschrieben wird.

Wenn die Methode `optimizeRecommendList` fehlschlägt, wird eine `LearningException` ausgelöst.

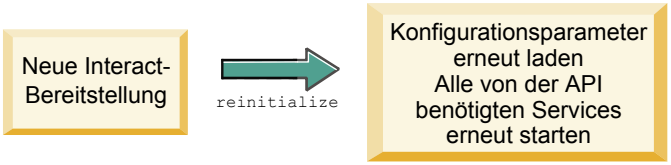
Rückgabewert

Die Methode `optimizeRecommendList` gibt eine Liste von `ITreatment`-Objekten zurück.

reinitialize

Die Laufzeitumgebung ruft die Methode `reinitialize` jedes Mal auf, wenn es eine neue Implementierung gibt. Diese Methode übergibt die gesamten Lernkonfigurationsdaten. Wenn Sie Services haben, die von der Lern-API erfordert werden und Konfigurationseigenschaften lesen, sollte diese Benutzeroberfläche sie erneut starten.

```
reinitialize(ILearningConfig config,
                 boolean debug)
```



- **config** - ein `ILearningConfig`-Objekt, das alle Konfigurationseigenschaften enthält.
- **debug** - ein boolescher Ausdruck. Wenn `true` gibt er an, dass die Ausführlichkeit der Protokollstufe für das Laufzeitumgebungssystem auf "debug" festgelegt ist. Es wird empfohlen, dass Sie diesen Wert auswählen, bevor in ein Protokoll geschrieben wird.

Wenn die Methode `logEvent` fehlschlägt, wird eine `LearningException` ausgelöst.

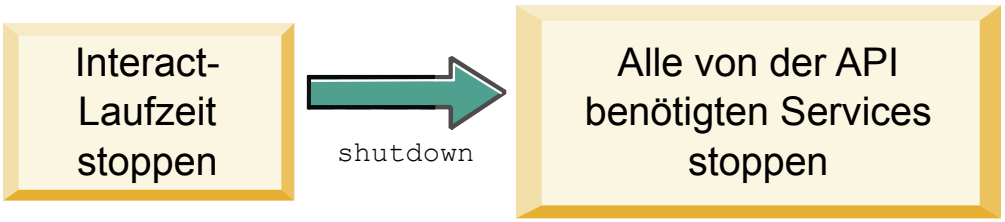
Rückgabewert

Keiner.

Herunterfahren

Die Laufzeitumgebung ruft die Methode `shutdown` auf, wenn der Laufzeitserver herunterfährt. Wenn es Bereinigungsaufgaben gibt, die von Ihrem Lernmodul erfordert werden, sollten sie zu diesem Zeitpunkt ausgeführt werden.

```
shutdown(ILearningConfig config, boolean debug)
```



Die Methode `shutdown` erfordert die folgenden Parameter.

- **config** - ein `ILearningConfig`-Objekt, das alle Konfigurationseigenschaften definiert.
- **debug** - ein boolescher Ausdruck. Wenn `true` gibt er an, dass die Ausführlichkeit der Protokollstufe für das Laufzeitumgebungssystem auf "debug" festgelegt ist. Es wird empfohlen, dass Sie diesen Wert auswählen, bevor in ein Protokoll geschrieben wird.

Wenn die Methode `shutdown` aus irgendeinem Grund fehlschlägt, wird eine `LearningException` ausgelöst.

Rückgabewert

Keiner.

Benutzeroberfläche IAudienceID

Die Benutzeroberfläche `IAudienceID` unterstützt die Benutzeroberfläche `IInteractSession`. Dies ist eine Benutzeroberfläche für die Zielgruppen-ID. Da Ihre Zielgruppen-ID aus mehreren Abschnitten bestehen kann, ermöglicht es Ihnen diese Benutzeroberfläche, auf alle Elemente der Zielgruppen-ID sowie auf den Zielgruppenebenennamen zuzugreifen.

getAudienceLevel

Die `getAudienceLevel`-Methode gibt die Zielgruppenebene zurück.

```
getAudienceLevel()
```

Rückgabewert

Die `getAudienceLevel`-Methode gibt eine Zeichenfolge zurück, die die Zielgruppenebene definiert.

getComponentNames

Die `getComponentNames`-Methode ruft einen Satz mit den Namen der Komponenten ab, aus denen sich die Zielgruppen-ID zusammensetzt. Wenn Ihre Zielgruppen-ID zum Beispiel die Werte `customerName` und `accountID` umfasst, gibt `getComponentNames` einen Satz zurück, der die Zeichenfolgen customerName und accountID enthält.

```
getComponentNames()
```

Rückgabewert

Ein Satz aus Zeichenfolgen, die die Namen der Komponenten der Zielgruppen-ID enthalten.

getComponentValue

Die getComponentValue-Methode gibt den Wert der angegebenen Komponente zurück.

```
getComponentValue(String componentName)
```

- **componentName** - eine Zeichenfolge, die den Namen der Komponente definiert, für die der Wert abgerufen werden soll. Diese Zeichenfolge unterscheidet nicht zwischen Groß- und Kleinschreibung.

Rückgabewert

Die getComponentValue-Methode gibt ein Objekt zurück, das den Wert der Komponente definiert.

IClientArgs

Die Benutzeroberfläche IClientArgs unterstützt die Benutzeroberfläche ILearning. Diese Benutzeroberfläche ist eine Abstraktion, um Daten zu erfassen, die an den Server vom Touchpoint übergeben werden, die noch nicht von den Sitzungsdaten erfasst sind. Beispiel: Die Anzahl der Angebote, die von der Interact-API-Methode getOffers angefordert werden. Diese Daten werden in einer Zuordnung gespeichert.

getValue

Die getValue-Methode gibt den Wert des angeforderten Zuordnungselements zurück.

```
getValue(int clientArgKey)
```

Die folgenden Elemente sind in der Zuordnung erforderlich.

- **1** - NUMBER_OF_OFFERS_REQUESTED. Die Anzahl der Angebote, die die getOffers-Methode des Interact APIs anfordert. Diese Konstante gibt eine Ganzzahl zurück.

Rückgabewert

Die getValue-Methode gibt ein Objekt zurück, das den Wert der angeforderten Zuordnungskonstante definiert.

IInteractSession

Die Benutzeroberfläche IInteractSession unterstützt die Benutzeroberfläche ILearning. Dies ist eine Benutzeroberfläche zur aktuellen Sitzung in der Laufzeitumgebung.

getAudienceId

Die getAudienceId-Methode gibt ein AudienceID-Objekt zurück. Verwenden Sie die IAudienceID-Benutzeroberfläche, um die Werte zu extrahieren.

```
getAudienceId()
```

Rückgabewert

Die getAudienceId-Methode gibt ein AudienceID-Objekt zurück.

getSessionData

Die getSessionData-Methode gibt eine nicht modifizierbare Zuordnung von Sitzungsdaten zurück, wobei der Name der Sitzungsvariablen als Schlüssel verwendet wird. Der Name der Sitzungsvariablen wird immer großgeschrieben. Verwenden Sie die IInteractSessionData-Benutzeroberfläche, um die Werte zu extrahieren.

```
getSessionData()
```

Rückgabewert

Die getSessionData-Methode gibt ein IInteractSessionData-Objekt zurück.

Benutzeroberfläche IInteractSessionData

Die Benutzeroberfläche IInteractSessionData unterstützt die Benutzeroberfläche ILearning. Dies ist eine Benutzeroberfläche zu den Laufzeit-Sitzungsdaten für den aktuellen Besucher. Sitzungsdaten werden in einer Liste von Name/Wert-Paaren gespeichert. Sie können diese Benutzeroberfläche auch verwenden, um den Wert von Daten in der Laufzeitsitzung zu ändern.

getDataType

Die getDataType-Methode gibt den Datentyp für den angegebenen Parameternamen zurück.

```
getDataType(String parameterName)
```

Rückgabewert

Die getDataType-Methode gibt ein InteractDataType-Objekt zurück. InteractDataType ist eine Java-Aufzählung, die als unbekannt, Zeichenfolge, Doppelzeichen, Datum oder Liste dargestellt wird.

getParameterNames

Die getParameterNames-Methode gibt einen Satz mit allen Namen der Daten in der aktuellen Sitzung zurück.

```
getParameterNames()
```

Rückgabewert

Die getParameterNames-Methode gibt einen Satz mit allen Namen zurück, für die Werte festgelegt wurden. Jeder Name im Satz kann in getValue(String) übergeben werden, um einen Wert zurückzugeben.

getValue

Die getValue-Methode gibt den Objektwert zurück, der dem angegebenen parameterName entspricht. Das Objekt kann eine Zeichenfolge, ein Doppelzeichen oder ein Datum sein.

```
getValue(parameterName)
```

Die getValue-Methode benötigt den folgenden Parameter:

- **parameterName** - eine Zeichenfolge, die den Namen des Name/Wert-Paares der Sitzungsdaten definiert.

Rückgabewert

Die getValue-Methode gibt ein Objekt zurück, das den Wert des angegebenen Parameters enthält.

setValue

Mit der setValue-Methode kann ein Wert für den angegebenen parameterName festgelegt werden. Der Wert kann eine Zeichenfolge, ein Doppelzeichen oder ein Datum sein.

```
setValue(String parameterName, Objekt value)
```

Die setValue-Methode benötigt die folgenden Parameter:

- **parameterName** - eine Zeichenfolge, die den Namen des Name/Wert-Paares der Sitzungsdaten definiert.
- **value** - ein Objekt, das den Wert des angegebenen Parameters definiert.

Rückgabewert

Keiner.

ILearningAttribute

Die Benutzeroberfläche ILearningAttribute unterstützt die Benutzeroberfläche ILearningConfig. Dies ist eine Benutzeroberfläche zu den Lernattributen, die in Konfigurationseigenschaften in der Kategorie learningAttributes definiert sind.

getName

Die getName-Methode gibt den Namen des Lernattributs zurück.

```
getName()
```

Rückgabewert

Die getName-Methode gibt eine Zeichenfolge zurück, die den Namen des Lernattributs definiert.

ILearningConfig

Die Benutzeroberfläche ILearningConfig unterstützt die Benutzeroberfläche ILearning. Dies ist eine Benutzeroberfläche zu den Konfigurationseigenschaften der Lernfunktion. Alle diese Methoden geben den Eigenschaftswert zurück.

Die Benutzeroberfläche besteht aus 15 Methoden:

- **getAdditionalParameters** - gibt eine Zuordnung von zusätzlichen Eigenschaften zurück, die in der Kategorie External Learning Config definiert sind
- **getAggregateStatsIntervalInMinutes** - gibt eine Ganzzahl zurück
- **getConfidenceLevel** - gibt eine Ganzzahl zurück
- **getDataSourceName** - gibt eine Zeichenfolge zurück
- **getDataSourceType** - gibt eine Zeichenfolge zurück
- **getInsertRawStatsIntervalInMinutes** - gibt eine Ganzzahl zurück
- **getLearningAttributes** - gibt eine Liste von ILearningAttribute-Objekten zurück
- **getMaxAttributeNames** - gibt eine Ganzzahl zurück
- **getMaxAttributeValues** - gibt eine Ganzzahl zurück

- **getMinPresentCountThreshold** - gibt eine Ganzzahl zurück
- **getOtherAttributeValue** - gibt eine Zeichenfolge zurück
- **getPercentRandomSelection** - gibt eine Ganzzahl zurück
- **getRecencyWeightingFactor** - gibt einen Gleitkommawert zurück
- **getRecencyWeightingPeriod** - gibt eine Ganzzahl zurück
- **isPruningEnabled** - gibt einen booleschen Ausdruck zurück

ILearningContext

Die Benutzeroberfläche `ILearningContext` unterstützt die Benutzeroberfläche `ILearning`.

getLearningContext

Die `getLearningContext`-Methode gibt die Konstante zurück, die festlegt, ob es sich bei diesem Szenario um einen Kontakt, eine Annahme oder eine Ablehnung handelt.

```
getLearningContext()
```

- **1**-LOG_AS_CONTACT
- **2**-LOG_AS_ACCEPT
- **3**-LOG_AS_REJECT

4 und 5 sind für zukünftige Verwendung reserviert.

Rückgabewert

Die `getLearningContext`-Methode gibt eine Ganzzahl zurück.

getResponseCode

Die `getResponseCode`-Methode gibt den Antwortcode zurück, der diesem Angebot zugeordnet ist. Dieser Wert muss in der `UA_UsrResponseType`-Tabelle in den Campaign-Systemtabellen vorhanden sein.

```
getResponseCode()
```

Rückgabewert

Die `getResponseCode`-Methode gibt eine Zeichenfolge zurück, die den Antwortcode definiert.

IOffer

Die Benutzeroberfläche `IOffer` unterstützt die Benutzeroberfläche `ITreatment`. Dies ist eine Benutzeroberfläche zum Angebotsobjekt, das in der Designumgebung definiert ist. Verwenden Sie die Benutzeroberfläche `IOffer`, um die Angebotsdetails aus der Laufzeitumgebung zu erfassen.

getCreateDate

Die `getCreateDate`-Methode gibt das Datum zurück, an dem das Angebot erstellt wurde.

```
getCreateDate()
```

Rückgabewert

Die getCreateDate-Methode gibt ein Datum zurück, das das Datum definiert, an dem das Angebot erstellt wurde.

getEffectiveDateFlag

Die getEffectiveDateFlag-Methode gibt eine Zahl zurück, die das Aktivierungsdatum des Angebots definiert.

```
getEffectiveDateFlag()
```

- **0** - das Aktivierungsdatum ist ein absolutes Datum, z. B. 15. März 2010.
- **1** - das Aktivierungsdatum ist das Datum der Empfehlung.

Rückgabewert

Die getEffectiveDateFlag-Methode gibt eine Ganzzahl zurück, die das gültige Datum des Angebots definiert.

getExpirationDateFlag

Die getExpirationDateFlag-Methode gibt einen Ganzzahlwert zurück, der das Ablaufdatum des Angebots beschreibt.

```
getExpirationDateFlag()
```

- **0** - ein absolutes Datum, z. B. 15. März 2010.
- **1** - eine Anzahl an Tagen nach der Empfehlung, z. B. 14.
- **2** - Monatsende nach Empfehlung. Ein Angebot am 31. März läuft noch am gleichen Tag ab.

Rückgabewert

Die getExpirationDateFlag-Methode gibt eine Ganzzahl zurück, die das Ablaufdatum des Angebots beschreibt.

getOfferAttributes

Die getOfferAttributes-Methode gibt Angebotsattribute zurück, die als ein IOfferAttributes-Objekt für das Angebot definiert sind.

```
getOfferAttributes()
```

Rückgabewert

Die getOfferAttributes-Methode gibt ein IOfferAttributes-Objekt zurück.

getOfferCode

Die getOfferCode-Methode gibt den Angebotscode des in Campaign definierten Angebots zurück.

```
getOfferCode()
```

Rückgabewert

Die getOfferCode-Methode gibt ein IOfferCode-Objekt zurück.

getOfferDescription

Die getOfferDescription-Methode gibt die Beschreibung des in Campaign definierten Angebots zurück.

getOfferDescription()

Rückgabewert

Die getOfferDescription-Methode gibt eine Zeichenfolge zurück.

getOfferID

Die getOfferID-Methode gibt die in Campaign definierte Angebots-ID zurück.

getOfferID()

Rückgabewert

Die getOfferID-Methode gibt einen Langwert zurück, der die Angebots-ID definiert.

getOfferName

Die getOfferName-Methode gibt den in Campaign definierten Namen des Angebots zurück.

getOfferName()

Rückgabewert

Die getOfferName-Methode gibt eine Zeichenfolge zurück.

getUpdateDate

Die getUpdateDate-Methode gibt das Datum der letzten Aktualisierung des Angebots zurück.

getUpdateDate()

Rückgabewert

Die getUpdateDate-Methode gibt ein Datum zurück, das definiert, wann das Angebot zuletzt aktualisiert wurde.

IOfferAttributes

Die Benutzeroberfläche IOfferAttributes unterstützt die Benutzeroberfläche IOffer. Dies ist eine Benutzeroberfläche zu den Angebotsattributen, die für ein Angebot in der Designumgebung definiert sind. Verwenden Sie die Benutzeroberfläche IOfferAttributes, um die Angebotsattribute aus der Laufzeitumgebung zu erfassen.

getParameterNames

Die getParameterNames-Methode gibt eine Liste mit den Parameternamen des Angebots zurück.

getParameterNames()

Rückgabewert

Die getParameterNames-Methode gibt einen Satz zurück, der die Liste mit den Parameternamen des Angebots definiert.

getValue

Die getValue-Methode gibt ein Objekt zurück, das den Wert des Angebotsattributs definiert.

```
getValue(String parameterName)
```

Die getValue-Methode gibt einen Wert für das gegebene Angebotsattribut zurück.

Rückgabewert

Benutzeroberfläche IOfferCode

Die Benutzeroberfläche IOfferCode unterstützt die Benutzeroberfläche ILearning. Dies ist eine Benutzeroberfläche zum Angebotscode, der für ein Angebot in der Designumgebung definiert wurde. Ein Angebotscode kann aus einer oder vielen Zeichenfolgen erstellt werden. Verwenden Sie die Benutzeroberfläche IOfferCode, um den Angebotscode aus der Laufzeitumgebung zu erfassen.

getPartCount

Die getPartCount-Methode gibt die Anzahl der Teile zurück, aus denen der Angebotscode besteht.

```
getPartCount()
```

Rückgabewert

Die getPartCount-Methode gibt eine Ganzzahl zurück, die die Anzahl der Teile des Angebotscodes definiert.

getParts

Die getParts-Methode gibt eine nicht modifizierbare Liste mit den Teilen des Angebotscodes zurück.

```
getParts()
```

Rückgabewert

Die getParts-Methode gibt eine nicht modifizierbare Liste mit den Teilen des Angebotscodes zurück.

LearningException

Die Klasse LearningException unterstützt die Benutzeroberfläche ILearning. Einige Methoden innerhalb der Benutzeroberfläche erfordern Implementierungen, um eine LearningException auszulösen, was eine einfache Unterklasse von java.lang.Exception ist. Es wird dringend für Debugging-Zwecke empfohlen, dass die LearningException mit der verursachenden Ausnahmebedingung erstellt wird, falls eine verursachende Ausnahmebedingung vorhanden ist.

IScoreOverride

Die Benutzeroberfläche IScoreOverride unterstützt die Benutzeroberfläche ITreatment. Diese Benutzeroberfläche ermöglicht es Ihnen, die Daten zu lesen, die in der Bewertungsüberschreibungs- oder Standardangebotstabelle definiert sind.

getOfferCode

Die getOfferCode-Methode gibt den Wert der Spalten mit dem Angebotscode in der Tabelle für die Bewertungsüberschreibung dieses Zielgruppenmitglieds zurück.

```
getOfferCode()
```

Rückgabewert

Die getOfferCode-Methode gibt ein IOfferCode-Objekt zurück, das den Wert der Angebotscodespalten in der Tabelle für die Bewertungsüberschreibung definiert.

getParameterNames

Die getParameterNames-Methode gibt die Liste der Parameter zurück.

```
getParameterNames()
```

Rückgabewert

Die getParameterNames-Methode gibt einen Satz zurück, der die Parameterliste definiert.

Die IScoreOverride-Methode enthält die folgenden Parameter. Sofern nicht anders angegeben, sind diese Parameter mit denen in der Tabelle für die Bewertungsüberschreibung identisch.

- ADJ_EXPLORE_SCORE_COLUMN
- CELL_CODE_COLUMN
- ENABLE_STATE_ID_COLUMN
- ESTIMATED_PRESENT_COUNT - Zum Überschreiben der geschätzten Anzahl der Vorkommen (während die Angebotsgewichtung berechnet wird)
- FINAL_SCORE_COLUMN
- LIKELIHOOD_SCORE_COLUMN
- MARKETER_SCORE
- OVERRIDE_TYPE_ID_COLUMN
- PREDICATE_COLUMN - Zum Erstellen eines booleschen Ausdrucks, um die Eignung des Angebots zu bestimmen
- PREDICATE_SCORE - Zum Erstellen eines Ausdrucks für die Berechnung eines numerischen Werts
- SCORE_COLUMN
- ZONE_COLUMN

Sie können auch auf jede andere Spalte verweisen, indem Sie den Namen einer beliebigen Spalte verwenden, die Sie der Tabelle mit den Standardwerten oder der Tabelle für die Bewertungsüberschreibung hinzugefügt haben.

getValue

Die getValue-Methode gibt den Wert der Spalte für die Zone in der Tabelle für die Bewertungsüberschreibung dieses Zielgruppenmitglieds zurück.

```
getValue(String parameterName)
```

- **parameterName** - eine Zeichenfolge, die den Namen des Parameters definiert, für den der Wert gelten soll.

Rückgabewert

Die getValue-Methode gibt ein Objekt zurück, das den Wert des angeforderten Parameters definiert.

ISelectionMethod

Die Benutzeroberfläche ISelection gibt die Methode an, mit der die Empfehlungsliste bereitgestellt wird. Der Standardwert für das Verfahrensobjekt ist EXTERNAL_LEARNING, daher müssen Sie diesen Wert nicht festlegen. Der Wert wird schließlich in "Detaillierter Kontaktverlauf" für Berichterstellungszwecke gespeichert.

Sie können diese Benutzeroberfläche über die bestehenden Konstanten hinaus erweitern, wenn Sie die Daten für die Analyse später speichern wollen. Sie könnten beispielsweise zwei verschiedene Lernmodule erstellen und sie auf separaten Servergruppen implementieren. Sie könnten die Benutzeroberfläche ISelection erweitern, um SERVER_GROUP_1 und SERVER_GROUP_2 zu umfassen. Sie könnten dann die Ergebnisse Ihrer zwei Lernmodule vergleichen.

Benutzeroberfläche ITreatment

Die Benutzeroberfläche ITreatment unterstützt die Benutzeroberfläche ILearning als eine Benutzeroberfläche zu Verfahrensinformationen. Ein Verfahren stellt ein Angebot dar, das einer bestimmten Zelle zugeordnet ist, wie in der Designumgebung definiert. Von dieser Benutzeroberfläche aus können Sie Zellen- und Angebotsinformationen sowie den zugewiesenen Marketing-Score abrufen.

getCellCode

Die getCellCode-Methode gibt den in Campaign definierten Zellencode zurück. Die Zelle ist die Zelle, die dem mit diesem Angebot verknüpften Smart-Segment zugeordnet ist.

```
getCellCode()
```

Rückgabewert

Die getCellCode-Methode gibt eine Zeichenfolge zurück, die den Zellencode definiert.

getCellId

Die getCellId-Methode gibt die interne ID der in Campaign definierten Zelle zurück. Die Zelle ist die Zelle, die dem mit diesem Angebot verknüpften Smart-Segment zugeordnet ist.

```
getOfferName()
```

Rückgabewert

Die getCellId-Methode gibt einen Langwert zurück, der die Zellen-ID definiert.

getCellName

Die getCellName-Methode gibt den Namen der in Campaign definierten Zelle zurück. Die Zelle ist die Zelle, die dem mit diesem Angebot verknüpften Smart-Segment zugeordnet ist.

```
getCellName()
```

Rückgabewert

Die `getCellName`-Methode gibt eine Zeichenfolge zurück, die den Zellennamen definiert.

getLearningScore

Die `getLearningScore`-Methode gibt die Punktzahl für dieses Verfahren zurück.

```
getLearningScore()
```

Für den Vorrang gilt Folgendes.

1. Rückgabe des neuen Werts, wenn in der von `IScoreoveride.PREDICATE_SCORE_COLUMN` verschlüsselten Zuordnung neue Werte vorhanden sind
2. Rückgabe der berechneten Punktzahl, wenn der Wert nicht null ist
3. Rückgabe der Punktzahl der Marketiers, wenn in der von `IScoreoveride.SCORE` verschlüsselten Zuordnung neue Werte vorhanden sind
4. Rückgabe der Punktzahl der Marketiers

Rückgabewert

Die `getLearningScore`-Methode gibt eine Ganzzahl zurück, die die Punktzahl definiert, die vom Lernalgorithmus ermittelt wurde.

getMarketerScore

Die `getMarketerScore`-Methode gibt die vom Regler auf der Registerkarte "Interaktionsstrategie" für das Angebot definierte Punktzahl des Marketiers zurück.

```
getMarketerScore()
```

Verwenden Sie `getPredicateScore`, um die mit den erweiterten Optionen auf der Registerkarte "Interaktionsstrategie" definierte Punktzahl eines Marketiers abzurufen.

Verwenden Sie `getLearningScore`, um die tatsächlich im Verfahren verwendete Punktzahl eines Marketiers abzurufen.

Rückgabewert

Die `getMarketerScore`-Methode gibt eine Ganzzahl zurück, die die Punktzahl des Marketiers definiert.

getOffer

Die `getOffer`-Methode gibt das Angebot für das Verfahren zurück.

```
getOffer()
```

Rückgabewert

Die `getOffer`-Methode gibt ein `IOffer`-Objekt zurück, das das Angebot für dieses Verfahren definiert.

getOverrideValues

Die getOverrideValues-Methode gibt die in der Tabelle für die Bewertungsüberschreibung oder die in der Tabelle mit den Standardangeboten definierten Überschreibungen zurück.

```
getOverrideValues()
```

Rückgabewert

Die getOverrideValues-Methode gibt ein IScoreOverride-Objekt zurück.

getPredicate

Die getPredicate-Methode gibt das Vergleichselement zurück, das in der Vergleichselementspalte der Tabelle mit den Standardangeboten, in der Tabelle für die Bewertungsüberschreibung oder in den erweiterten Optionen der Verfahrensregeln definiert ist.

```
getPredicate()
```

Rückgabewert

Die getPredicate-Methode gibt eine Zeichenfolge zurück, die das Vergleichselement definiert, das in der Vergleichselementspalte der Tabelle mit den Standardangeboten, in der Tabelle für die Bewertungsüberschreibung oder in den erweiterten Optionen der Verfahrensregeln definiert ist.

getPredicateScore

Die getPredicateScore-Methode gibt die Punktzahl zurück, die in der Vergleichselementspalte der Tabelle mit den Standardangeboten, in der Tabelle für die Bewertungsüberschreibung oder in den erweiterten Optionen der Verfahrensregeln definiert ist.

```
getPredicateScore()
```

Rückgabewert

Die getPredicateScore-Methode gibt ein Doppelzeichen zurück, das die Punktzahl definiert, die in der Vergleichselementspalte der Tabelle mit den Standardangeboten, in der Tabelle für die Bewertungsüberschreibung oder in den erweiterten Optionen der Verfahrensregeln eingestellt ist.

getScore

Die Methode getScore gibt den Marketing-Score zurück, der entweder durch die Interaktionsstrategie in Campaign oder durch die Bewertungsüberschreibungstabelle definiert ist.

```
getScore()
```

Die getScore-Methode gibt eine der folgenden Optionen zurück:

- Der Marketing-Score des Angebots, der auf der Registerkarte "Interaktionsstrategie" in Campaign definiert ist, wenn die enableScoreOverrideLookup-Eigenschaft auf "false" gesetzt ist.
- Die Bewertung des Angebots, die in scoreOverrideTable definiert ist, wenn die enableScoreOverrideLookup-Eigenschaft auf "true" gesetzt ist.

Rückgabewert

Die Methode `getScore` gibt eine Ganzzahl zurück, die die Bewertung des Angebots darstellt.

getTreatmentCode

Die `getTreatmentCode`-Methode gibt den Verfahrenscode zurück.

```
getTreatmentCode()
```

Rückgabewert

Die `getTreatmentCode`-Methode gibt eine Zeichenfolge zurück, die den Verfahrenscode definiert.

setActualValueUsed

Verwenden Sie die `setActualValueUsed`-Methode, um zu definieren, welche Werte in den verschiedenen Stadien bei der Ausführung des Lernalgorithmus verwendet werden.

```
setActualValueUsed(string parmName, object value)
```

Beispiel: Wenn Sie mit dieser Methode in den Kontakt- und Antwortverlaufstabellen schreiben und die vorhandenen Beispielberichte ändern, können Sie Daten aus dem Lernalgorithmus in die Berichte einbinden.

- **parmName** - eine Zeichenfolge, die den Namen des angegebenen Parameters definiert.
- **value** - ein Objekt, das den Wert des angegebenen Parameters definiert.

Rückgabewert

Keiner.

Beispiel für eine Lern-API

Dieser Abschnitt enthält eine Beispielimplementierung von ILearningInterface. Beachten Sie, dass diese Implementierung nur ein Beispiel ist und nicht für die Verwendung in einer Produktionsumgebung geeignet ist.

Dieses Beispiel protokolliert Annahme- und Kontaktzählungen und verwendet das Verhältnis von Annahme zu Kontakten für ein bestimmtes Angebot als die Wahrscheinlichkeitsrate für das Angebot. Nicht präsentierte Angebote erhalten eine höhere Priorität für Empfehlungen. Angebote mit zumindest einem Kontakt werden basierend auf absteigender Annahmewahrscheinlichkeitsrate sortiert.

In diesem Beispiel werden alle Zählungen im Speicher abgelegt. Das ist kein realistisches Szenario, weil der Speicherplatz des Laufzeitservers nicht ausreichen wird. In einem realen Produktionsszenario sollten die Zählungen persistent in einer Datenbank gespeichert werden.

```
package com.unicacorp.interact.samples.learning.v2;

import java.util.ArrayList;
import java.util.Collections;
import java.util.Comparator;
import java.util.HashMap;
import java.util.List;
import java.util.Map;
```

```
import com.unicacorp.interact.samples.learning.SampleOptimizer.MyOfferSorter;
import com.unicacorp.interact.treatment.optimization.IClientArgs;
import com.unicacorp.interact.treatment.optimization.IInteractSession;
import com.unicacorp.interact.treatment.optimization.ILearningConfig;
import com.unicacorp.interact.treatment.optimization.ILearningContext;
import com.unicacorp.interact.treatment.optimization.IOffer;
import com.unicacorp.interact.treatment.optimization.LearningException;
import com.unicacorp.interact.treatment.optimization.v2.ILearning;
import com.unicacorp.interact.treatment.optimization.v2.ITreatment;

/**
 * This is a sample implementation of the learning optimizer.
 * The interface ILearning may be found in the interact.jar library.
 *
 * To actually use this implementation, select ExternalLearning as the optimizationType in the offerServing node
 * of the Interact application within the Platform configuration.  Within the offerserving node there is also
 * an External Learning config category - within there you must set the name of the class to this:
 * com.unicacorp.interact.samples.learning.v2.SampleLearning.  Please note however, this implementation is just a sample
 * and was not designed to be used in a production environment.
 *
 *
 * This example keeps track of accept and contact counts and uses the ratio of accept to contacts
 * for a particular offer as the acceptance probability rate for the offer.
 *
 *
 * Offers not presented will get higher priority for recommending.
 * Offers with at least one contact will be ordered based on descending acceptance probability rate.
 *
 * Note: all counts are kept in memory.  This is not a realistic scenario since you would run out of memory sooner or
 * later.  In a real production scenario, the counts should be persisted into a database.
 *
 */

public class SampleLearning implements ILearning
{

    // A map of offer ids to contact count for the offer id
    private Map<Long,Integer> _offerToContactCount = new HashMap<Long, Integer>();

    // A map of offer ids to contact count for the offer id
    private Map<Long,Integer> _offerToAcceptCount = new HashMap<Long, Integer>();


    /* (non-Javadoc)
     * @see com.unicacorp.interact.treatment.optimization.v2.ILearning#initialize
     *   (com.unicacorp.interact.treatment.optimization.v2.ILearningConfig, boolean)
     */
    public void initialize(ILearningConfig config, boolean debug) throws LearningException
    {
        // If any remote connections are required, this is a good place to initialize those connections as this
        // method is called once at the start of the interact runtime webapp.
        // This example does not have any remote connections and prints for debugging purposes that this method will
        // be called
        System.out.println("Calling initialize for SampleLearning");
    }

    /* (non-Javadoc)
     * @see com.unicacorp.interact.treatment.optimization.v2.ILearning#reinitialize
     *   (com.unicacorp.interact.treatment.optimization.v2.ILearningConfig, boolean)
     */
    public void reinitialize(ILearningConfig config, boolean debug) throws LearningException
    {
        // If an IC is deployed, this reinitialize method is called to allow the implementation to
        // refresh any updated configuration settings
        System.out.println("Calling reinitialize for SampleLearning");
    }


    /* (non-Javadoc)
     * @see com.unicacorp.interact.treatment.optimization.v2.ILearning#logEvent
     *   (com.unicacorp.interact.treatment.optimization.v2.ILearningContext,
     *   com.unicacorp.interact.treatment.optimization.v2.IOffer,
     *   com.unicacorp.interact.treatment.optimization.v2.IClientArgs,
     *   com.unicacorp.interact.treatment.optimization.IInteractSession, boolean)
     */
    public void logEvent(ILearningContext context, IOffer offer, IClientArgs clientArgs,
   IInteractSession session, boolean debug) throws LearningException
    {
        System.out.println("Calling logEvent for SampleLearning");
```

```
    if(context.getLearningContext()==ILearningContext.LOG_AS_CONTACT)
    {
        System.out.println("adding contact");

        // Keep track of all contacts in memory
        synchronized(_offerToAcceptCount)
        {
           Integer count = _offerToAcceptCount.get(offer.getOfferId());
           if(count == null)
               count = new Integer(1);
           else
               count++;
           _offerToAcceptCount.put(offer.getOfferId(), ++count);
        }

    }
    else if(context.getLearningContext()==ILearningContext.LOG_AS_ACCEPT)
    {
        System.out.println("adding accept");
        // Keep track of all accept counts in memory by adding to the map
        synchronized(_offerToAcceptCount)
        {
           Integer count = _offerToAcceptCount.get(offer.getOfferId());
           if(count == null)
               count = new Integer(1);
           else
               count++;
           _offerToAcceptCount.put(offer.getOfferId(), ++count);
        }
        }

}

/* (non-Javadoc)
 * @see com.unicacorp.interact.treatment.optimization.v2.ILearning#optimizeRecommendList
 *   (java.util.List, com.unicacorp.interact.treatment.optimization.v2.IClientArgs,
 *   com.unicacorp.interact.treatment.optimization.IInteractSession, boolean)
 */
public List<ITreatment> optimizeRecommendList(List<ITreatment> recList,
    IClientArgs clientArgs, IInteractSession session, boolean debug)
    throws LearningException
{
    System.out.println("Calling optimizeRecommendList for SampleLearning");

    // Sort the candidate treatments by calling the sorter defined in this class and return the sorted list
    Collections.sort(recList,new MyOfferSorter());

    // now just return what was asked for via "numberRequested" variable
    List<ITreatment> result = new ArrayList<ITreatment>();

    for(int x=0;x<(Integer)clientArgs.getValue(IClientArgs.NUMBER_OF_OFFERS_REQUESTED) && x<recList.size();x++)
    {
        result.add(recList.get(x));
    }
    return result;
}

/* (non-Javadoc)
 * @see com.unicacorp.interact.treatment.optimization.v2.ILearning#shutdown
 *   (com.unicacorp.interact.treatment.optimization.v2.ILearningConfig, boolean)
 */
public void shutdown(ILearningConfig config, boolean debug) throws LearningException
{
    // If any remote connections exist, this would be a good place to gracefully
    // disconnect from them as this method is called at the shutdown of the Interact runtime
    // webapp.   For this example, there is nothing really to do
    // except print out a statement for debugging.
    System.out.println("Calling shutdown for SampleLearning");

}
// Sort by:
// 1. offers with zero contacts  - for ties, order is based on original input
// 2. descending accept probability rate - for ties, order is based on original input

public class MyOfferSorter implements Comparator<ITreatment>
{
    private static final long serialVersionUID = 1L;
```

```
        /* (non-Javadoc)
         * @see java.lang.Comparable#compareTo(java.lang.Object)
         */
        public int compare(ITreatment treatment1, ITreatment treatment2)
        {

            // get contact count for both treatments
            Integer contactCount1 = _offerToContactCount.get(treatment1.getOffer().getOfferId());
            Integer contactCount2 = _offerToContactCount.get(treatment2.getOffer().getOfferId());

            // if treatment hasn't been contacted, then that wins
            if(contactCount1 == null || contactCount1 == 0)
                return -1;

            if(contactCount2 == null || contactCount2 == 0)
                return 1;

            // get accept counts
            Integer acceptCount1 = _offerToAcceptCount.get(treatment1.getOffer().getOfferId());
            Integer acceptCount2 = _offerToAcceptCount.get(treatment2.getOffer().getOfferId());

            float acceptProbability1 = (float) acceptCount1 / (float) contactCount1;
            float acceptProbability2 = (float) acceptCount2 / (float) contactCount2;

            // descending order
            return (int) (acceptProbability2 - acceptProbability1);

        }
    }

}
```

Anhang A. IBM Interact-WSDL

Die Interact-Installation enthält zwei WDSL-XML-Dateien (WSDL = Web Services Description Language), die die verfügbaren Web-Services und die Zugriffsmöglichkeiten darauf beschreiben. Diese Dateien können Sie im Ausgangsverzeichnis von Interact anzeigen. Ein Beispiel wird hier dargestellt.

Nachdem Sie Interact installiert haben, finden Sie die WSDL-Dateien von Interact an den folgenden Positionen:

- <*Interact_home*>/conf/InteractService.wsdl
- <*Interact_home*>/conf/InteractAdminService.wsdl

Die Interact-WDSL kann mit jedem Software-Release und jedem Fixpack geändert werden. Informationen dazu finden Sie in den *Interact-Releaseinformationen* oder in den Readme-Dateien des Release.

Eine Kopie von `InteractService.wsdl` finden Sie hier. Wenn Sie sicherstellen wollen, dass Sie die aktuellsten Informationen verwenden, überprüfen Sie die WDSL-Dateien, die mit Interact installiert werden.

```
<?xml version="1.0" encoding="UTF-8"?>
<wsdl:definitions xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/" xmlns:mime="http://schemas.xmlsoap.org/wsdl/mime/"
xmlns:ns0="http://soap.api.interact.unicacorp.com" xmlns:soap12="http://schemas.xmlsoap.org/wsdl/soap12/"
xmlns:http="http://schemas.xmlsoap.org/wsdl/http/" bloop="http://api.interact.unicacorp.com/xsd"
xmlns:wsaw="http://www.w3.org/2006/05/addressing/wsdl" xmlns:xs="http://www.w3.org/2001/XMLSchema"
xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/" targetNamespace="http://soap.api.interact.unicacorp.com">
 <wsdl:types>
  <xs:schema xmlns:ns="http://soap.api.interact.unicacorp.com" attributeFormDefault="qualified"
   elementFormDefault="qualified" targetNamespace="http://soap.api.interact.unicacorp.com">
   <xs:element name="executeBatch">
    <xs:complexType>
     <xs:sequence>
      <xs:element minOccurs="1" name="sessionID" nillable="false" type="xs:string"/>
      <xs:element maxOccurs="unbounded" minOccurs="1" name="commands" nillable="false" type="ns1:CommandImpl"/>
     </xs:sequence>
    </xs:complexType>
   </xs:element>
   <xs:element name="executeBatchResponse">
    <xs:complexType>
     <xs:sequence>
      <xs:element minOccurs="1" name="return" nillable="false" type="ns1:BatchResponse"/>
     </xs:sequence>
    </xs:complexType>
   </xs:element>
   <xs:element name="endSession">
    <xs:complexType>
     <xs:sequence>
      <xs:element minOccurs="1" name="sessionID" nillable="false" type="xs:string"/>
     </xs:sequence>
    </xs:complexType>
   </xs:element>
   <xs:element name="endSessionResponse">
    <xs:complexType>
     <xs:sequence>
      <xs:element minOccurs="1" name="return" nillable="false" type="ns1:Response"/>
     </xs:sequence>
    </xs:complexType>
   </xs:element>
   <xs:element name="getOffers">
    <xs:complexType>
     <xs:sequence>
      <xs:element minOccurs="1" name="sessionID" nillable="false" type="xs:string"/>
      <xs:element minOccurs="1" name="iPoint" nillable="false" type="xs:string"/>
      <xs:element minOccurs="1" name="numberRequested" type="xs:int"/>
     </xs:sequence>
    </xs:complexType>
```

```
</xs:element>
<xs:element name="getOffersResponse">
 <xs:complexType>
  <xs:sequence>
   <xs:element minOccurs="1" name="return" nillable="false" type="ns1:Response"/>
  </xs:sequence>
 </xs:complexType>
</xs:element>
<xs:element name="getProfile">
 <xs:complexType>
  <xs:sequence>
   <xs:element minOccurs="1" name="sessionID" nillable="false" type="xs:string"/>
  </xs:sequence>
 </xs:complexType>
</xs:element>
<xs:element name="getProfileResponse">
 <xs:complexType>
  <xs:sequence>
   <xs:element minOccurs="1" name="return" nillable="false" type="ns1:Response"/>
  </xs:sequence>
 </xs:complexType>
</xs:element>
<xs:element name="getVersionResponse">
 <xs:complexType>
  <xs:sequence>
   <xs:element minOccurs="1" name="return" nillable="false" type="ns1:Response"/>
  </xs:sequence>
 </xs:complexType>
</xs:element>
<xs:element name="postEvent">
 <xs:complexType>
  <xs:sequence>
   <xs:element minOccurs="1" name="sessionID" nillable="false" type="xs:string"/>
   <xs:element minOccurs="1" name="eventName" nillable="false" type="xs:string"/>
   <xs:element maxOccurs="unbounded" minOccurs="1" name="eventParameters"
    nillable="true" type="ns1:NameValuePairImpl"/>
  </xs:sequence>
 </xs:complexType>
</xs:element>
<xs:element name="postEventResponse">
 <xs:complexType>
  <xs:sequence>
   <xs:element minOccurs="1" name="return" nillable="false" type="ns1:Response"/>
  </xs:sequence>
 </xs:complexType>
</xs:element>
<xs:element name="setAudience">
 <xs:complexType>
  <xs:sequence>
   <xs:element minOccurs="1" name="sessionID" nillable="false" type="xs:string"/>
   <xs:element maxOccurs="unbounded" minOccurs="1" name="audienceID" nillable="false" type="ns1:NameValuePairImpl"/>
   <xs:element minOccurs="1" name="audienceLevel" nillable="false" type="xs:string"/>
   <xs:element maxOccurs="unbounded" minOccurs="1" name="parameters" nillable="true" type="ns1:NameValuePairImpl"/>
  </xs:sequence>
 </xs:complexType>
</xs:element>
<xs:element name="setAudienceResponse">
 <xs:complexType>
  <xs:sequence>
   <xs:element minOccurs="1" name="return" nillable="false" type="ns1:Response"/>
  </xs:sequence>
 </xs:complexType>
</xs:element>
<xs:element name="setDebug">
 <xs:complexType>
  <xs:sequence>
   <xs:element minOccurs="1" name="sessionID" nillable="false" type="xs:string"/>
   <xs:element minOccurs="1" name="debug" type="xs:boolean"/>
  </xs:sequence>
 </xs:complexType>
</xs:element>
<xs:element name="setDebugResponse">
 <xs:complexType>
  <xs:sequence>
   <xs:element minOccurs="1" name="return" nillable="false" type="ns1:Response"/>
  </xs:sequence>
 </xs:complexType>
</xs:element>
<xs:element name="startSession">
```

```
  <xs:complexType>
   <xs:sequence>
    <xs:element minOccurs="1" name="sessionID" nillable="false" type="xs:string"/>
    <xs:element minOccurs="1" name="relyOnExistingSession" type="xs:boolean"/>
    <xs:element minOccurs="1" name="debug" type="xs:boolean"/>
    <xs:element minOccurs="1" name="interactiveChannel" nillable="false" type="xs:string"/>
    <xs:element maxOccurs="unbounded" minOccurs="1" name="audienceID" nillable="false" type="ns1:NameValuePairImpl"/>
    <xs:element minOccurs="1" name="audienceLevel" nillable="false" type="xs:string"/>
    <xs:element maxOccurs="unbounded" minOccurs="1" name="parameters" nillable="true" type="ns1:NameValuePairImpl"/>
   </xs:sequence>
  </xs:complexType>
 </xs:element>
 <xs:element name="startSessionResponse">
  <xs:complexType>
   <xs:sequence>
    <xs:element minOccurs="1" name="return" nillable="false" type="ns1:Response"/>
   </xs:sequence>
  </xs:complexType>
 </xs:element>
</xs:schema>
<xs:schema xmlns:ax21="http://api.interact.unicacorp.com/xsd" attributeFormDefault="qualified"
 elementFormDefault="qualified" targetNamespace="http://api.interact.unicacorp.com/xsd">
 <xs:complexType name="Command">
  <xs:sequence>
   <xs:element maxOccurs="unbounded" minOccurs="1" name="audienceID" nillable="true" type="ax21:NameValuePair"/>
   <xs:element minOccurs="1" name="audienceLevel" nillable="true" type="xs:string"/>
   <xs:element minOccurs="1" name="debug" type="xs:boolean"/>
   <xs:element minOccurs="1" name="event" nillable="true" type="xs:string"/>
   <xs:element maxOccurs="unbounded" minOccurs="1" name="eventParameters" nillable="true" type="ax21:NameValuePair"/>
   <xs:element minOccurs="1" name="interactionPoint" nillable="true" type="xs:string"/>
   <xs:element minOccurs="1" name="interactiveChannel" nillable="true" type="xs:string"/>
   <xs:element minOccurs="1" name="methodIdentifier" nillable="true" type="xs:string"/>
   <xs:element minOccurs="1" name="numberRequested" type="xs:int"/>
   <xs:element minOccurs="1" name="relyOnExistingSession" type="xs:boolean"/>
  </xs:sequence>
 </xs:complexType>
 <xs:complexType name="NameValuePair">
  <xs:sequence>
   <xs:element minOccurs="1" name="name" nillable="true" type="xs:string"/>
   <xs:element minOccurs="1" name="valueAsDate" nillable="true" type="xs:dateTime"/>
   <xs:element minOccurs="1" name="valueAsNumeric" nillable="true" type="xs:double"/>
   <xs:element minOccurs="1" name="valueAsString" nillable="true" type="xs:string"/>
   <xs:element minOccurs="1" name="valueDataType" nillable="true" type="xs:string"/>
  </xs:sequence>
 </xs:complexType>
 <xs:complexType name="CommandImpl">
  <xs:sequence>
   <xs:element maxOccurs="unbounded" minOccurs="1" name="audienceID" nillable="true" type="ax21:NameValuePairImpl"/>
   <xs:element minOccurs="1" name="audienceLevel" nillable="true" type="xs:string"/>
   <xs:element minOccurs="1" name="debug" type="xs:boolean"/>
   <xs:element minOccurs="1" name="event" nillable="true" type="xs:string"/>
   <xs:element maxOccurs="unbounded" minOccurs="1" name="eventParameters" nillable="true" type="ax21:NameValuePairImpl"/>
   <xs:element minOccurs="1" name="interactionPoint" nillable="true" type="xs:string"/>
   <xs:element minOccurs="1" name="interactiveChannel" nillable="true" type="xs:string"/>
   <xs:element minOccurs="1" name="methodIdentifier" nillable="true" type="xs:string"/>
   <xs:element minOccurs="1" name="numberRequested" type="xs:int"/>
   <xs:element minOccurs="1" name="relyOnExistingSession" type="xs:boolean"/>
  </xs:sequence>
 </xs:complexType>
 <xs:complexType name="NameValuePairImpl">
  <xs:sequence>
   <xs:element minOccurs="1" name="name" nillable="true" type="xs:string"/>
   <xs:element minOccurs="1" name="valueAsDate" nillable="true" type="xs:dateTime"/>
   <xs:element minOccurs="1" name="valueAsNumeric" nillable="true" type="xs:double"/>
   <xs:element minOccurs="1" name="valueAsString" nillable="true" type="xs:string"/>
   <xs:element minOccurs="1" name="valueDataType" nillable="true" type="xs:string"/>
  </xs:sequence>
 </xs:complexType>
 <xs:complexType name="BatchResponse">
  <xs:sequence>
   <xs:element minOccurs="0" name="batchStatusCode" type="xs:int"/>
   <xs:element maxOccurs="unbounded" minOccurs="0" name="responses" nillable="false" type="ax21:Response"/>
  </xs:sequence>
 </xs:complexType>
 <xs:complexType name="Response">
  <xs:sequence>
   <xs:element maxOccurs="unbounded" minOccurs="0" name="advisoryMessages" nillable="true" type="ax21:AdvisoryMessage"/>
   <xs:element minOccurs="0" name="apiVersion" nillable="false" type="xs:string"/>
   <xs:element minOccurs="0" name="offerList" nillable="true" type="ax21:OfferList"/>
```

```
    <xs:element maxOccurs="unbounded" minOccurs="0" name="profileRecord" nillable="true" type="ax21:NameValuePair"/>
    <xs:element minOccurs="0" name="sessionID" nillable="true" type="xs:string"/>
    <xs:element minOccurs="0" name="statusCode" type="xs:int"/>
   </xs:sequence>
  </xs:complexType>
  <xs:complexType name="AdvisoryMessage">
   <xs:sequence>
    <xs:element minOccurs="0" name="detailMessage" nillable="true" type="xs:string"/>
    <xs:element minOccurs="0" name="message" nillable="true" type="xs:string"/>
    <xs:element minOccurs="0" name="messageCode" type="xs:int"/>
    <xs:element minOccurs="0" name="statusLevel" type="xs:int"/>
   </xs:sequence>
  </xs:complexType>
  <xs:complexType name="OfferList">
   <xs:sequence>
    <xs:element minOccurs="0" name="defaultString" nillable="true" type="xs:string"/>
    <xs:element maxOccurs="unbounded" minOccurs="0" name="recommendedOffers" nillable="true" type="ax21:Offer"/>
   </xs:sequence>
  </xs:complexType>
  <xs:complexType name="Offer">
   <xs:sequence>
    <xs:element maxOccurs="unbounded" minOccurs="0" name="additionalAttributes" nillable="true" type="ax21:NameValuePair"/>
    <xs:element minOccurs="0" name="description" nillable="true" type="xs:string"/>
    <xs:element maxOccurs="unbounded" minOccurs="0" name="offerCode" nillable="true" type="xs:string"/>
    <xs:element minOccurs="0" name="offerName" nillable="true" type="xs:string"/>
    <xs:element minOccurs="0" name="score" type="xs:int"/>
    <xs:element minOccurs="0" name="treatmentCode" nillable="true" type="xs:string"/>
   </xs:sequence>
  </xs:complexType>
 </xs:schema>
</wsdl:types>
<wsdl:message name="setAudienceRequest">
 <wsdl:part name="parameters" element="ns0:setAudience"/>
</wsdl:message>
<wsdl:message name="setAudienceResponse">
 <wsdl:part name="parameters" element="ns0:setAudienceResponse"/>
</wsdl:message>
<wsdl:message name="postEventRequest">
 <wsdl:part name="parameters" element="ns0:postEvent"/>
</wsdl:message>
<wsdl:message name="postEventResponse">
 <wsdl:part name="parameters" element="ns0:postEventResponse"/>
</wsdl:message>
<wsdl:message name="getOffersRequest">
 <wsdl:part name="parameters" element="ns0:getOffers"/>
</wsdl:message>
<wsdl:message name="getOffersResponse">
 <wsdl:part name="parameters" element="ns0:getOffersResponse"/>
</wsdl:message>
<wsdl:message name="startSessionRequest">
 <wsdl:part name="parameters" element="ns0:startSession"/>
</wsdl:message>
<wsdl:message name="startSessionResponse">
 <wsdl:part name="parameters" element="ns0:startSessionResponse"/>
</wsdl:message>
<wsdl:message name="getVersionRequest"/>
<wsdl:message name="getVersionResponse">
 <wsdl:part name="parameters" element="ns0:getVersionResponse"/>
</wsdl:message>
<wsdl:message name="setDebugRequest">
 <wsdl:part name="parameters" element="ns0:setDebug"/>
</wsdl:message>
<wsdl:message name="setDebugResponse">
 <wsdl:part name="parameters" element="ns0:setDebugResponse"/>
</wsdl:message>
<wsdl:message name="executeBatchRequest">
 <wsdl:part name="parameters" element="ns0:executeBatch"/>
</wsdl:message>
<wsdl:message name="executeBatchResponse">
 <wsdl:part name="parameters" element="ns0:executeBatchResponse"/>
</wsdl:message>
<wsdl:message name="getProfileRequest">
 <wsdl:part name="parameters" element="ns0:getProfile"/>
</wsdl:message>
<wsdl:message name="getProfileResponse">
 <wsdl:part name="parameters" element="ns0:getProfileResponse"/>
</wsdl:message>
<wsdl:message name="endSessionRequest">
 <wsdl:part name="parameters" element="ns0:endSession"/>
```

```
</wsdl:message>
<wsdl:message name="endSessionResponse">
 <wsdl:part name="parameters" element="ns0:endSessionResponse"/>
</wsdl:message>
<wsdl:portType name="InteractServicePortType">
 <wsdl:operation name="setAudience">
  <wsdl:input message="ns0:setAudienceRequest" wsaw:Action="urn:setAudience"/>
  <wsdl:output message="ns0:setAudienceResponse" wsaw:Action="urn:setAudienceResponse"/>
 </wsdl:operation>
 <wsdl:operation name="postEvent">
  <wsdl:input message="ns0:postEventRequest" wsaw:Action="urn:postEvent"/>
  <wsdl:output message="ns0:postEventResponse" wsaw:Action="urn:postEventResponse"/>
 </wsdl:operation>
 <wsdl:operation name="getOffers">
  <wsdl:input message="ns0:getOffersRequest" wsaw:Action="urn:getOffers"/>
  <wsdl:output message="ns0:getOffersResponse" wsaw:Action="urn:getOffersResponse"/>
 </wsdl:operation>
 <wsdl:operation name="startSession">
  <wsdl:input message="ns0:startSessionRequest" wsaw:Action="urn:startSession"/>
  <wsdl:output message="ns0:startSessionResponse" wsaw:Action="urn:startSessionResponse"/>
 </wsdl:operation>
 <wsdl:operation name="getVersion">
  <wsdl:input message="ns0:getVersionRequest" wsaw:Action="urn:getVersion"/>
  <wsdl:output message="ns0:getVersionResponse" wsaw:Action="urn:getVersionResponse"/>
 </wsdl:operation>
 <wsdl:operation name="setDebug">
  <wsdl:input message="ns0:setDebugRequest" wsaw:Action="urn:setDebug"/>
  <wsdl:output message="ns0:setDebugResponse" wsaw:Action="urn:setDebugResponse"/>
 </wsdl:operation>
 <wsdl:operation name="executeBatch">
  <wsdl:input message="ns0:executeBatchRequest" wsaw:Action="urn:executeBatch"/>
  <wsdl:output message="ns0:executeBatchResponse" wsaw:Action="urn:executeBatchResponse"/>
 </wsdl:operation>
 <wsdl:operation name="getProfile">
  <wsdl:input message="ns0:getProfileRequest" wsaw:Action="urn:getProfile"/>
  <wsdl:output message="ns0:getProfileResponse" wsaw:Action="urn:getProfileResponse"/>
 </wsdl:operation>
 <wsdl:operation name="endSession">
  <wsdl:input message="ns0:endSessionRequest" wsaw:Action="urn:endSession"/>
  <wsdl:output message="ns0:endSessionResponse" wsaw:Action="urn:endSessionResponse"/>
 </wsdl:operation>
</wsdl:portType>
<wsdl:binding name="InteractServiceSOAP11Binding" type="ns0:InteractServicePortType">
 <soap:binding style="document" transport="http://schemas.xmlsoap.org/soap/http"/>
 <wsdl:operation name="setAudience">
  <soap:operation soapAction="urn:setAudience" style="document"/>
  <wsdl:input>
   <soap:body use="literal"/>
  </wsdl:input>
  <wsdl:output>
   <soap:body use="literal"/>
  </wsdl:output>
 </wsdl:operation>
 <wsdl:operation name="postEvent">
  <soap:operation soapAction="urn:postEvent" style="document"/>
  <wsdl:input>
   <soap:body use="literal"/>
  </wsdl:input>
  <wsdl:output>
   <soap:body use="literal"/>
  </wsdl:output>
 </wsdl:operation>
 <wsdl:operation name="getOffers">
  <soap:operation soapAction="urn:getOffers" style="document"/>
  <wsdl:input>
   <soap:body use="literal"/>
  </wsdl:input>
  <wsdl:output>
   <soap:body use="literal"/>
  </wsdl:output>
 </wsdl:operation>
 <wsdl:operation name="startSession">
  <soap:operation soapAction="urn:startSession" style="document"/>
  <wsdl:input>
   <soap:body use="literal"/>
  </wsdl:input>
  <wsdl:output>
   <soap:body use="literal"/>
  </wsdl:output>
```

```
 </wsdl:operation>
 <wsdl:operation name="getVersion">
  <soap:operation soapAction="urn:getVersion" style="document"/>
  <wsdl:input>
   <soap:body use="literal"/>
  </wsdl:input>
  <wsdl:output>
   <soap:body use="literal"/>
  </wsdl:output>
 </wsdl:operation>
 <wsdl:operation name="setDebug">
  <soap:operation soapAction="urn:setDebug" style="document"/>
  <wsdl:input>
   <soap:body use="literal"/>
  </wsdl:input>
  <wsdl:output>
   <soap:body use="literal"/>
  </wsdl:output>
 </wsdl:operation>
 <wsdl:operation name="executeBatch">
  <soap:operation soapAction="urn:executeBatch" style="document"/>
  <wsdl:input>
   <soap:body use="literal"/>
  </wsdl:input>
  <wsdl:output>
   <soap:body use="literal"/>
  </wsdl:output>
 </wsdl:operation>
 <wsdl:operation name="getProfile">
  <soap:operation soapAction="urn:getProfile" style="document"/>
  <wsdl:input>
   <soap:body use="literal"/>
  </wsdl:input>
  <wsdl:output>
   <soap:body use="literal"/>
  </wsdl:output>
 </wsdl:operation>
 <wsdl:operation name="endSession">
  <soap:operation soapAction="urn:endSession" style="document"/>
  <wsdl:input>
   <soap:body use="literal"/>
  </wsdl:input>
  <wsdl:output>
   <soap:body use="literal"/>
  </wsdl:output>
 </wsdl:operation>
</wsdl:binding>
<wsdl:binding name="InteractServiceSOAP12Binding" type="ns0:InteractServicePortType">
 <soap12:binding transport="http://schemas.xmlsoap.org/soap/http" style="document"/>
 <wsdl:operation name="setAudience">
  <soap12:operation soapAction="urn:setAudience" style="document"/>
  <wsdl:input>
   <soap12:body use="literal"/>
  </wsdl:input>
  <wsdl:output>
   <soap12:body use="literal"/>
  </wsdl:output>
 </wsdl:operation>
 <wsdl:operation name="postEvent">
  <soap12:operation soapAction="urn:postEvent" style="document"/>
  <wsdl:input>
   <soap12:body use="literal"/>
  </wsdl:input>
  <wsdl:output>
   <soap12:body use="literal"/>
  </wsdl:output>
 </wsdl:operation>
 <wsdl:operation name="getOffers">
  <soap12:operation soapAction="urn:getOffers" style="document"/>
  <wsdl:input>
   <soap12:body use="literal"/>
  </wsdl:input>
  <wsdl:output>
   <soap12:body use="literal"/>
  </wsdl:output>
 </wsdl:operation>
 <wsdl:operation name="startSession">
  <soap12:operation soapAction="urn:startSession" style="document"/>
  <wsdl:input>
```

```
    <soap12:body use="literal"/>
   </wsdl:input>
   <wsdl:output>
    <soap12:body use="literal"/>
   </wsdl:output>
  </wsdl:operation>
  <wsdl:operation name="getVersion">
   <soap12:operation soapAction="urn:getVersion" style="document"/>
   <wsdl:input>
    <soap12:body use="literal"/>
   </wsdl:input>
   <wsdl:output>
    <soap12:body use="literal"/>
   </wsdl:output>
  </wsdl:operation>
  <wsdl:operation name="setDebug">
   <soap12:operation soapAction="urn:setDebug" style="document"/>
   <wsdl:input>
    <soap12:body use="literal"/>
   </wsdl:input>
   <wsdl:output>
    <soap12:body use="literal"/>
   </wsdl:output>
  </wsdl:operation>
  <wsdl:operation name="executeBatch">
   <soap12:operation soapAction="urn:executeBatch" style="document"/>
   <wsdl:input>
    <soap12:body use="literal"/>
   </wsdl:input>
   <wsdl:output>
    <soap12:body use="literal"/>
   </wsdl:output>
  </wsdl:operation>
  <wsdl:operation name="getProfile">
   <soap12:operation soapAction="urn:getProfile" style="document"/>
   <wsdl:input>
    <soap12:body use="literal"/>
   </wsdl:input>
   <wsdl:output>
    <soap12:body use="literal"/>
   </wsdl:output>
  </wsdl:operation>
  <wsdl:operation name="endSession">
   <soap12:operation soapAction="urn:endSession" style="document"/>
   <wsdl:input>
    <soap12:body use="literal"/>
   </wsdl:input>
   <wsdl:output>
    <soap12:body use="literal"/>
   </wsdl:output>
  </wsdl:operation>
 </wsdl:binding>
 <wsdl:binding name="InteractServiceHttpBinding" type="ns0:InteractServicePortType">
  <http:binding verb="POST"/>
  <wsdl:operation name="setAudience">
   <http:operation location="InteractService/setAudience"/>
   <wsdl:input>
    <mime:content part="setAudience" type="text/xml"/>
   </wsdl:input>
   <wsdl:output>
    <mime:content part="setAudience" type="text/xml"/>
   </wsdl:output>
  </wsdl:operation>
  <wsdl:operation name="postEvent">
   <http:operation location="InteractService/postEvent"/>
   <wsdl:input>
    <mime:content part="postEvent" type="text/xml"/>
   </wsdl:input>
   <wsdl:output>
    <mime:content part="postEvent" type="text/xml"/>
   </wsdl:output>
  </wsdl:operation>
  <wsdl:operation name="getOffers">
   <http:operation location="InteractService/getOffers"/>
   <wsdl:input>
    <mime:content part="getOffers" type="text/xml"/>
   </wsdl:input>
   <wsdl:output>
    <mime:content part="getOffers" type="text/xml"/>
```

```
   </wsdl:output>
  </wsdl:operation>
  <wsdl:operation name="startSession">
   <http:operation location="InteractService/startSession"/>
   <wsdl:input>
    <mime:content part="startSession" type="text/xml"/>
   </wsdl:input>
   <wsdl:output>
    <mime:content part="startSession" type="text/xml"/>
   </wsdl:output>
  </wsdl:operation>
  <wsdl:operation name="getVersion">
   <http:operation location="InteractService/getVersion"/>
   <wsdl:input>
    <mime:content part="getVersion" type="text/xml"/>
   </wsdl:input>
   <wsdl:output>
    <mime:content part="getVersion" type="text/xml"/>
   </wsdl:output>
  </wsdl:operation>
  <wsdl:operation name="setDebug">
   <http:operation location="InteractService/setDebug"/>
   <wsdl:input>
    <mime:content part="setDebug" type="text/xml"/>
   </wsdl:input>
   <wsdl:output>
    <mime:content part="setDebug" type="text/xml"/>
   </wsdl:output>
  </wsdl:operation>
  <wsdl:operation name="executeBatch">
   <http:operation location="InteractService/executeBatch"/>
   <wsdl:input>
    <mime:content part="executeBatch" type="text/xml"/>
   </wsdl:input>
   <wsdl:output>
    <mime:content part="executeBatch" type="text/xml"/>
   </wsdl:output>
  </wsdl:operation>
  <wsdl:operation name="getProfile">
   <http:operation location="InteractService/getProfile"/>
   <wsdl:input>
    <mime:content part="getProfile" type="text/xml"/>
   </wsdl:input>
   <wsdl:output>
    <mime:content part="getProfile" type="text/xml"/>
   </wsdl:output>
  </wsdl:operation>
  <wsdl:operation name="endSession">
   <http:operation location="InteractService/endSession"/>
   <wsdl:input>
    <mime:content part="endSession" type="text/xml"/>
   </wsdl:input>
   <wsdl:output>
    <mime:content part="endSession" type="text/xml"/>
   </wsdl:output>
  </wsdl:operation>
 </wsdl:binding>
 <wsdl:service name="InteractService">
  <wsdl:port name="InteractServiceSOAP11port_http" binding="ns0:InteractServiceSOAP11Binding">
   <soap:address location="http://localhost:7001/interact/services/InteractService"/>
  </wsdl:port>
  <wsdl:port name="InteractServiceSOAP12port_http" binding="ns0:InteractServiceSOAP12Binding">
   <soap12:address location="http://localhost:7001/interact/services/InteractService"/>
  </wsdl:port>
  <wsdl:port name="InteractServiceHttpport" binding="ns0:InteractServiceHttpBinding">
   <http:address location="http://localhost:7001/interact/services/InteractService"/>
  </wsdl:port>
 </wsdl:service>
</wsdl:definitions>
```

Anhang B. Interact Laufzeitumgebung - Konfigurationseigenschaften

In diesem Abschnitt werden alle Konfigurationseigenschaften für die Interact-Laufzeitumgebung beschrieben.

Interact | general

Diese Konfigurationseigenschaften definieren allgemeine Einstellungen für Ihre Laufzeitumgebung, einschließlich der Standardprotokollebene und Ländereinstellung.

log4jConfig

Beschreibung

Die Position der Datei, die die log4j-Eigenschaften enthält. Dieser Pfad muss sich auf die `INTERACT_HOME`-Umgebungsvariable beziehen. `INTERACT_HOME` ist die Position des Interact-Installationsverzeichnisses.

Standardwert

`./conf/interact_log4j.properties`

asmUserForDefaultLocale

Beschreibung

Die Eigenschaft `asmUserForDefaultLocale` legt den IBM EMM-Benutzer fest, von dem Interact die Ländereinstellungen ableitet.

Die Ländereinstellungen definieren, welche Sprache in der Designzeit angezeigt wird und in welcher Sprache nützliche Hinweise von der Interact-API erstellt werden. Wenn die Ländereinstellung nicht mit den Einstellungen des Betriebssystems Ihrer Maschine übereinstimmt, funktioniert Interact trotzdem, aber möglicherweise werden nützliche Hinweise in einer anderen Sprache erstellt, als in der Designumgebung verwendet wird.

Standardwert

`asm_admin`

Interact | general | learningTablesDataSource

Diese Konfigurationseigenschaften definieren die Datenquelleneinstellungen für die integrierten Lerntabellen. Sie müssen diese Datenquelle definieren, wenn Sie das integrierte Lernmodul von Interact verwenden.

Wenn Sie mit der Lern-API eine eigene Implementierung des Lernmoduls erstellen, können Sie Ihr benutzerdefiniertes Lernmodul so konfigurieren, dass diese Werte mithilfe der `ILearningConfig`-Benutzeroberfläche gelesen werden.

jndiName

Beschreibung

Verwenden Sie diese `jndiName`-Eigenschaft, um die JNDI-Datenquelle (JNDI = Java Naming and Directory Interface) zu identifizieren, die auf dem An-

wendungsserver (WebSphere oder WebLogic) für die Lerntabellen definiert ist, auf die die Laufzeitserver von Interact zugreifen.

Die Lerntabellen werden von der DLL-Datei `aci_lrntab` erstellt und enthalten (u. a.) die folgenden Tabellen: `UACI_AttributeValue` und `UACI_OfferStats`.

Standardwert

Kein Standardwert definiert.

type

Beschreibung

Der Datenbanktyp für die Datenquelle, die von den Lerntabellen verwendet wird, auf die die Laufzeitserver von Interact zugreifen.

Die Lerntabellen werden von der DLL-Datei `aci_lrntab` erstellt und enthalten (u. a.) die folgenden Tabellen: `UACI_AttributeValue` und `UACI_OfferStats`.

Standardwert

SQLServer

Gültige Werte

SQLServer | DB2 | ORACLE

connectionRetryPeriod

Beschreibung

Die Eigenschaft `ConnectionRetryPeriod` gibt in Sekunden an, wie lange Interact eine fehlgeschlagene Datenbankverbindungsaufforderung für die Lerntabellen automatisch wiederholt. Interact versucht in diesem Zeitraum automatisch, die Verbindung zur Datenbank wiederherzustellen, bevor ein Datenbankfehler gemeldet wird. Wird der Wert auf 0 gesetzt, wiederholt Interact den Verbindungsversuch unbegrenzt. Wenn -1 festgelegt ist, wird kein wiederholter Versuch unternommen.

Die Lerntabellen werden von der DLL-Datei `aci_lrntab` erstellt und enthalten (u. a.) die folgenden Tabellen: `UACI_AttributeValue` und `UACI_OfferStats`.

Standardwert

-1

connectionRetryDelay

Beschreibung

Die Eigenschaft `ConnectionRetryDelay` gibt in Sekunden an, wie lange Interact wartet, bevor versucht wird, die Verbindung zur Datenbank wiederherzustellen, wenn ein Fehler bei den Lerntabellen aufgetreten ist. Wenn der Wert auf -1 gesetzt ist, wird nicht versucht, die Verbindung wiederherzustellen.

Die Lerntabellen werden von der DLL-Datei `aci_lrntab` erstellt und enthalten (u. a.) die folgenden Tabellen: `UACI_AttributeValue` und `UACI_OfferStats`.

Standardwert

-1

Schema

Beschreibung

Der Name des Schemas, das die Tabellen für das integrierte Lernmodul enthält. Interact fügt den Wert für diese Eigenschaft vor allen Tabellennamen ein. Beispiel: `UACI_IntChannel` wird zu `schema.UACI_IntChannel`.

Sie müssen kein Schema definieren. Wenn Sie kein Schema angeben, geht Interact davon aus, dass der Eigner der Tabellen mit dem Schema übereinstimmt. Sie sollten diesen Wert festlegen, um Mehrdeutigkeit zu vermeiden.

Standardwert

Kein Standardwert definiert.

Interact | general | prodUserDataSource

Diese Konfigurationseigenschaften definieren die Datenquelleneinstellungen für die Produktionsprofiltabellen. Sie müssen diese Datenquelle definieren. Auf diese Datenquelle verweist die Laufzeitumgebung beim Ausführen der interaktiven Ablaufdiagramme nach der Bereitstellung.

jndiName

Beschreibung

Verwenden Sie diese `jndiName`-Eigenschaft, um die JNDI-Datenquelle (JNDI = Java Naming and Directory Interface) zu identifizieren, die auf dem Anwendungsserver (WebSphere oder WebLogic) für die Kundentabellen definiert ist, auf die die Laufzeitserver von Interact zugreifen.

Standardwert

Kein Standardwert definiert.

type

Beschreibung

Der Datenbanktyp für die Kundentabellen, auf die Laufzeitserver in Interact zugreifen.

Standardwert

SQLServer

Gültige Werte

SQLServer | DB2 | ORACLE

aliasPrefix

Beschreibung

Die Eigenschaft `AliasPrefix` gibt an, wie Interact den neuen Aliasnamen bildet, der automatisch von Interact erstellt wird, wenn eine Dimensionstabelle verwendet und in eine neue Tabelle in den Kundentabellen geschrieben wird, auf die Laufzeitserver von Interact zugreifen.

Für jede Datenbank gilt eine maximale ID-Länge. Lesen Sie die Dokumentation für die von Ihnen verwendete Datenbank, um sicherzustellen, dass Sie keinen Wert festlegen, der die maximale ID-Länge für Ihre Datenbank überschreitet.

Standardwert

A

connectionRetryPeriod

Beschreibung

Die Eigenschaft `ConnectionRetryPeriod` gibt in Sekunden an, wie lange Interact eine fehlgeschlagene Datenbankverbindungsaufforderung für die Laufzeitkundentabellen automatisch wiederholt. Interact versucht in diesem Zeitraum automatisch, die Verbindung zur Datenbank wiederherzustellen, bevor ein Datenbankfehler gemeldet wird. Wird der Wert auf 0 gesetzt, wiederholt Interact den Verbindungsversuch unbegrenzt. Wenn -1 festgelegt ist, wird kein wiederholter Versuch unternommen.

Standardwert

-1

connectionRetryDelay

Beschreibung

Die Eigenschaft `ConnectionRetryDelay` gibt in Sekunden an, wie lange Interact wartet, bevor versucht wird, die Verbindung zur Datenbank wiederherzustellen, wenn ein Fehler bei den Laufzeitkundentabellen in Interact aufgetreten ist. Wenn der Wert auf -1 gesetzt ist, wird nicht versucht, die Verbindung wiederherzustellen.

Standardwert

-1

Schema

Beschreibung

Der Name des Schemas, das Ihre Profildatentabellen enthält. Interact fügt den Wert für diese Eigenschaft vor allen Tabellennamen ein. Beispiel: `UACI_IntChannel` wird zu `schema.UACI_IntChannel`.

Sie müssen kein Schema definieren. Wenn Sie kein Schema angeben, geht Interact davon aus, dass der Eigner der Tabellen mit dem Schema übereinstimmt. Sie sollten diesen Wert festlegen, um Mehrdeutigkeit zu vermeiden.

Wenn Sie eine DB2-Datenbank verwenden, muss der Schemaname in Großbuchstaben geschrieben sein.

Standardwert

Kein Standardwert definiert.

Interact | general | systemTablesDataSource

Diese Konfigurationseigenschaften definieren die Datenquelleneinstellungen für die Systemtabellen für die Laufzeitumgebung. Sie müssen diese Datenquelle definieren.

jndiName

Beschreibung

Verwenden Sie diese jndiName-Eigenschaft, um die JNDI-Datenquelle (JNDI = Java Naming and Directory Interface) zu identifizieren, die auf dem Anwendungsserver (WebSphere oder WebLogic) für die Laufzeitumgebungstabellen definiert ist.

Die Laufzeitumgebungsdatenbank, die mit den DLL-Scripts aci_runtime und aci_populate_runtime gefüllt ist und beispielsweise (u. a.) die folgenden Tabellen enthält: UACI_CHOfferAttrib und UACI_DefaultedStat.

Standardwert

Kein Standardwert definiert.

type

Beschreibung

Der Typ der Datenbank für die Systemtabellen für die Laufzeitumgebung.

Die Laufzeitumgebungsdatenbank, die mit den DLL-Scripts aci_runtime und aci_populate_runtime gefüllt ist und beispielsweise (u. a.) die folgenden Tabellen enthält: UACI_CHOfferAttrib und UACI_DefaultedStat.

Standardwert

SQLServer

Gültige Werte

SQLServer | DB2 | ORACLE

connectionRetryPeriod

Beschreibung

Die Eigenschaft ConnectionRetryPeriod gibt in Sekunden an, wie lange Interact eine fehlgeschlagene Datenbankverbindungsaufforderung für die Laufzeitsystemtabellen automatisch wiederholt. Interact versucht in diesem Zeitraum automatisch, die Verbindung zur Datenbank wiederherzustellen, bevor ein Datenbankfehler gemeldet wird. Wird der Wert auf 0 gesetzt, wiederholt Interact den Verbindungsversuch unbegrenzt. Wenn -1 festgelegt ist, wird kein wiederholter Versuch unternommen.

Die Laufzeitumgebungsdatenbank, die mit den DLL-Scripts aci_runtime und aci_populate_runtime gefüllt ist und beispielsweise (u. a.) die folgenden Tabellen enthält: UACI_CHOfferAttrib und UACI_DefaultedStat.

Standardwert

-1

connectionRetryDelay

Beschreibung

Die Eigenschaft ConnectionRetryDelay gibt in Sekunden an, wie lange Interact wartet, bevor versucht wird, die Verbindung zur Datenbank wiederherzustellen, wenn ein Fehler bei den Laufzeitsystemtabellen in Interact aufgetreten ist. Wenn der Wert auf -1 gesetzt ist, wird nicht versucht, die Verbindung wiederherzustellen.

Die Laufzeitumgebungsdatenbank, die mit den DLL-Scripts aci_runtime und aci_populate_runtime gefüllt ist und beispielsweise (u. a.) die folgenden Tabellen enthält: UACI_CHOfferAttrib und UACI_DefaultedStat.

Standardwert

-1

Schema

Beschreibung

Der Name des Schemas, das die Tabellen für die Laufzeitumgebung enthält. Interact fügt den Wert für diese Eigenschaft vor allen Tabellennamen ein. Beispiel: UACI_IntChannel wird zu schema.UACI_IntChannel.

Sie müssen kein Schema definieren. Wenn Sie kein Schema angeben, geht Interact davon aus, dass der Eigner der Tabellen mit dem Schema übereinstimmt. Sie sollten diesen Wert festlegen, um Mehrdeutigkeit zu vermeiden.

Standardwert

Kein Standardwert definiert.

Interact | general | systemTablesDataSource | loaderProperties

Diese Konfigurationseigenschaften definieren die Einstellungen des Datenbankladeprogramms für die Systemtabellen für die Laufzeitumgebung. Sie müssen diese Eigenschaften nur definieren, wenn Sie ein Datenbankladeprogramm verwenden.

databaseName

Beschreibung

Der Name der Datenbank, mit der das Datenbankladeprogramm verbunden ist.

Standardwert

Kein Standardwert definiert.

LoaderCommandForAppend

Beschreibung

Der Parameter LoaderCommandForAppend legt einen Befehl fest. Dieser Befehl dient zum Aufrufen des Datenbankladedienstprogramms für das Hinzufügen von Datensätzen zu den Staging-Datenbanktabellen für den Kontakt- und Antwortverlauf in Interact. Sie müssen diesen Parameter festlegen, um das Datenbankladeprogramm für die Kontakt- und Antwortverlaufsdaten zu aktivieren.

Dieser Parameter wird als vollständiger Pfadname zur ausführbaren Datei des Datenbankladedienstprogramms oder zu einem Script, das das Datenbankladeprogramm startet, angegeben. Durch die Verwendung eines Scripts können Sie zusätzliche Einrichtungsvorgänge ausführen, bevor Sie das Ladedienstprogramm starten.

Für den Start der meisten Datenbankladeprogramme sind mehrere Argumente erforderlich. Diese können u. a. die Daten- und Steuerdatei, aus der geladen werden soll, und die Datenbank und Tabelle, in die geladen werden soll, angeben. Die Token werden bei der Ausführung des Befehls durch die festgelegten Elemente ersetzt.

Informieren Sie sich in der Dokumentation zu Ihrem Datenbankladedienstprogramm über die korrekte Syntax, die Sie für den Start des Dienstprogramms verwenden müssen.

Dieser Parameter ist standardmäßig nicht definiert.

In der folgenden Tabelle werden die verfügbaren Token für `LoaderCommandForAppend` beschrieben.

Token	**Beschreibung**
<CONTROLFILE>	Dieses Token wird durch den vollständigen Pfad und Dateinamen der temporären Steuerdatei ersetzt, die von Interact gemäß der im Parameter `LoaderControlFileTemplate` angegebenen Vorlage generiert wird.
<DATABASE>	Dieses Token wird durch den Namen der Datenquelle ersetzt, in die Interact Daten lädt. Dies ist derselbe Datenquellenname, der im Kategorienamen für diese Datenquelle verwendet wird.
<DATAFILE>	Dieses Token wird durch den vollständigen Pfad und Dateinamen der temporären Datendatei ersetzt, die von Interact während des Ladevorgangs erstellt wird. Diese Datei befindet sich im Temp-Verzeichnis von Interact: `UNICA_ACTMPDIR`.
<DBCOLUMNNUMBER>	Dieses Token wird durch die Spaltenordnungszahl in der Datenbank ersetzt.
<FIELDLENGTH>	Dieses Token wird durch die Länge des in die Datenbank geladenen Felds ersetzt.
<FIELDNAME>	Dieses Token wird durch den Namen des in die Datenbank geladenen Felds ersetzt.
<FIELDNUMBER>	Dieses Token wird durch die Nummer des in die Datenbank geladenen Felds ersetzt.
<FIELDTYPE>	Dieses Token wird durch den Literalwert "CHAR( )" ersetzt. Die Länge des Felds wird in den Klammern () angegeben. Wenn der Feldtyp CHAR von der Datenbank nicht verstanden wird, können Sie den entsprechenden Text für den Feldtyp manuell angeben und das Token <FIELDLENGTH> verwenden. Beispiel: Bei SQLSVR und SQL2000 würden Sie "SQLCHAR(<FIELDLENGTH>)" verwenden.
<NATIVETYPE>	Dieses Token wird durch den Typ der Datenbank ersetzt, in die das Feld geladen wird.

Token	Beschreibung
<NUMFIELDS>	Dieses Token wird durch die Anzahl der Felder in der Tabelle ersetzt.
<PASSWORD>	Dieses Token wird mit dem Datenbankkennwort von der aktuellen Ablaufdiagrammverbindung zur Datenquelle ersetzt.
<TABLENAME>	Dieses Token wird durch den Namen der Datenbanktabelle ersetzt, in die Interact Daten lädt.
<USER>	Dieses Token wird mit dem Datenbankbenutzer der aktuellen Ablaufdiagrammverbindung zur Datenquelle ersetzt.

Standardwert

Kein Standardwert definiert.

LoaderControlFileTemplateForAppend

Beschreibung

Die Eigenschaft `LoaderControlFileTemplateForAppend` gibt den vollständigen Pfad und Dateinamen der Steuerdateivorlage an, die zuvor in Interact konfiguriert wurde. Wenn dieser Parameter festgelegt ist, erstellt Interact basierend auf der hier angegebenen Vorlage dynamisch eine temporäre Steuerdatei. Der Pfad und Name dieser temporären Steuerdatei stehen dem Token `<CONTROLFILE>` zur Verfügung, das der Eigenschaft `LoaderCommandForAppend` zur Verfügung steht.

Vor der Verwendung von Interact im Datenbankladeprogrammmodus müssen Sie die Steuerdateivorlage konfigurieren, die durch diesen Parameter festgelegt wird. Die Steuerdateivorlage unterstützt die folgenden Token, die dynamisch ersetzt werden, wenn die temporäre Steuerdatei von Interact erstellt wird.

Informationen über die richtige Syntax für Ihre Steuerdatei finden Sie in der Dokumentation zu Ihrem Datenbankladeprogramm. Die für die Steuerdateivorlage zur Verfügung stehenden Token sind dieselben wie die für die Eigenschaft `LoaderControlFileTemplate`.

Dieser Parameter ist standardmäßig nicht definiert.

Standardwert

Kein Standardwert definiert.

LoaderDelimiterForAppend

Beschreibung

Die Eigenschaft `LoaderDelimiterForAppend` gibt an, ob die temporäre Interact-Datendatei eine Flatfile mit fester Breite oder mit Trennzeichen ist. Bei einer Datei mit Trennzeichen werden außerdem die Zeichen bzw. der Zeichensatz festgelegt, die/der als Trennzeichen verwendet wird.

Ist der Wert nicht definiert, erstellt Interact die temporäre Datendatei als Textdatei mit fester Breite.

Wenn Sie einen Wert angeben, wird dieser verwendet, wenn das Ladeprogramm zum Füllen einer Tabelle aufgerufen wird, von der nicht bekannt ist, dass sie leer ist. Interact erstellt die temporäre Datendatei als eine Flatfile mit Trennzeichen und verwendet den Wert dieser Eigenschaft als Trennzeichen.

Diese Eigenschaft ist standardmäßig nicht definiert.

Standardwert

Gültige Werte

Zeichen, die Sie auf Wunsch in doppelten Anführungszeichen angeben können.

LoaderDelimiterAtEndForAppend

Beschreibung

Einige externe Ladeprogramme erfordern, dass die Datendatei durch Trennzeichen getrennt ist und jede Zeile mit dem Trennzeichen endet. Um diese Anforderung zu erfüllen, setzen Sie den Wert für `LoaderDelimiterAtEndForAppend` auf `TRUE`. Wenn das Ladeprogramm zum Füllen einer Tabelle aufgerufen wird, von der nicht bekannt ist, dass sie leer ist, verwendet Interact Trennzeichen am Ende jeder Zeile.

Standardwert

`FALSE`

Gültige Werte

`TRUE | FALSE`

LoaderUseLocaleDP

Beschreibung

Die Eigenschaft `LoaderUseLocaleDP` legt fest, ob das ländereinstellungsspezifische Symbol als Dezimalzeichen verwendet wird, wenn Interact numerische Werte in Dateien schreibt, die über ein Datenbankladedienstprogramm geladen werden sollen.

Geben Sie `FALSE` an, um festzulegen, dass der Punkt (.) als Dezimalzeichen verwendet werden soll.

Geben Sie `TRUE` an, um festzulegen, dass das ländereinstellungsspezifische Symbol als Dezimalzeichen verwendet werden soll.

Standardwert

`FALSE`

Gültige Werte

`TRUE | FALSE`

Interact | general | testRunDataSource

Diese Konfigurationseigenschaften definieren die Datenquelleneinstellungen für die Testlauftabellen für die Designumgebung in Interact. Sie müssen diese Datenquelle

für mindestens eine der Laufzeitumgebungen definieren. Diese Tabellen werden verwendet, wenn Sie einen Testlauf Ihres interaktiven Ablaufdiagramms durchführen.

jndiName

Beschreibung

Verwenden Sie diese `jndiName`-Eigenschaft, um die JNDI-Datenquelle (JNDI = Java Naming and Directory Interface) zu identifizieren, die auf dem Anwendungsserver (WebSphere oder WebLogic) für die Kundentabellen definiert ist, auf die die Designumgebung beim Durchführen von interaktiven Ablaufdiagrammtestläufen zugreift.

Standardwert

Kein Standardwert definiert.

type

Beschreibung

Der Datenbanktyp für die Kundentabellen, auf die die Designumgebung beim Durchführen von interaktiven Ablaufdiagrammtestläufen zugreift.

Standardwert

SQLServer

Gültige Werte

SQLServer | DB2 | ORACLE

aliasPrefix

Beschreibung

Die Eigenschaft `AliasPrefix` gibt an, wie Interact den neuen Aliasnamen bildet, der automatisch von Interact erstellt wird, wenn eine Dimensionstabelle verwendet wird und in eine neue Tabelle für die Kundentabellen geschrieben wird, auf die die Designumgebung beim Durchführen von interaktiven Ablaufdiagrammtestläufen zugreift.

Für jede Datenbank gilt eine maximale ID-Länge. Lesen Sie die Dokumentation für die von Ihnen verwendete Datenbank, um sicherzustellen, dass Sie keinen Wert festlegen, der die maximale ID-Länge für Ihre Datenbank überschreitet.

Standardwert

A

connectionRetryPeriod

Beschreibung

Die Eigenschaft `ConnectionRetryPeriod` gibt in Sekunden an, wie lange Interact eine fehlgeschlagene Datenbankverbindungsaufforderung für die Testlauftabellen automatisch wiederholt. Interact versucht in diesem Zeitraum automatisch, die Verbindung zur Datenbank wiederherzustellen, bevor ein Datenbankfehler gemeldet wird. Wird der Wert auf 0 gesetzt, wiederholt Interact den Verbindungsversuch unbegrenzt. Wenn -1 festgelegt ist, wird kein wiederholter Versuch unternommen.

Standardwert

-1

connectionRetryDelay

Beschreibung

Die Eigenschaft `ConnectionRetryDelay` gibt in Sekunden an, wie lange Interact wartet, bevor versucht wird, die Verbindung zur Datenbank wiederherzustellen, wenn ein Fehler bei den Testlauftabellen aufgetreten ist. Wenn der Wert auf -1 gesetzt ist, wird nicht versucht, die Verbindung wiederherzustellen.

Standardwert

-1

Schema

Beschreibung

Der Name des Schemas, das die Tabellen für die interaktiven Ablaufdiagrammtestläufe enthält. Interact fügt den Wert für diese Eigenschaft vor allen Tabellennamen ein. Beispiel: `UACI_IntChannel` wird zu `schema.UACI_IntChannel`.

Sie müssen kein Schema definieren. Wenn Sie kein Schema angeben, geht Interact davon aus, dass der Eigner der Tabellen mit dem Schema übereinstimmt. Sie sollten diesen Wert festlegen, um Mehrdeutigkeit zu vermeiden.

Standardwert

Kein Standardwert definiert.

Interact | general | contactAndResponseHistoryDataSource

Diese Konfigurationseigenschaften definieren die Verbindungseinstellungen für die Kontakt- und Antwortverlaufsdatenquelle, die für die sitzungsübergreifende Antwortverfolgung in Interact erforderlich ist. Zwischen diesen Einstellungen und dem Kontakt- und Antwortverlaufsmodul besteht keine Verbindung.

jndiName

Beschreibung

Verwenden Sie diese `jndiName`-Eigenschaft, um die JNDI-Datenquelle (JNDI = Java Naming and Directory Interface) zu identifizieren, die auf dem Anwendungsserver (WebSphere oder WebLogic) für die Kontakt- und Antwortverlaufsdatenquelle definiert ist, die für die sitzungsübergreifende Antwortverfolgung in Interact erforderlich ist.

Standardwert

type

Beschreibung

Der Datenbanktyp für die Datenquelle, die von der Kontakt- und Antwortverlaufsdatenquelle verwendet wird, die für die sitzungsübergreifende Antwortverfolgung in Interact erforderlich ist.

Standardwert

SQLServer

Gültige Werte

SQLServer | DB2 | ORACLE

connectionRetryPeriod

Beschreibung

Die Eigenschaft `ConnectionRetryPeriod` gibt in Sekunden an, wie lange Interact eine fehlgeschlagene Datenbankverbindungsaufforderung für die sitzungsübergreifende Antwortverfolgung in Interact automatisch wiederholt. Interact versucht in diesem Zeitraum automatisch, die Verbindung zur Datenbank wiederherzustellen, bevor ein Datenbankfehler gemeldet wird. Wird der Wert auf 0 gesetzt, wiederholt Interact den Verbindungsversuch unbegrenzt. Wenn -1 festgelegt ist, wird kein wiederholter Versuch unternommen.

Standardwert

-1

connectionRetryDelay

Beschreibung

Die Eigenschaft `ConnectionRetryDelay` gibt in Sekunden an, wie lange Interact wartet, bevor versucht wird, die Verbindung zur Datenbank wiederherzustellen, wenn ein Fehler bei der sitzungsübergreifenden Antwortverfolgung in Interact aufgetreten ist. Wenn der Wert auf -1 gesetzt ist, wird nicht versucht, die Verbindung wiederherzustellen.

Standardwert

-1

Schema

Beschreibung

Der Name des Schemas, das die Tabellen für die sitzungsübergreifende Antwortverfolgung in Interact enthält. Interact fügt den Wert für diese Eigenschaft vor allen Tabellennamen ein. Beispiel: `UACI_IntChannel` wird zu `schema.UACI_IntChannel`.

Sie müssen kein Schema definieren. Wenn Sie kein Schema angeben, geht Interact davon aus, dass der Eigner der Tabellen mit dem Schema übereinstimmt. Sie sollten diesen Wert festlegen, um Mehrdeutigkeit zu vermeiden.

Standardwert

Kein Standardwert definiert.

Interact | general | idsByType

Diese Konfigurationseigenschaften definieren Einstellungen für ID-Nummern, die vom Kontakt- und Antwortverlaufsmodul verwendet werden.

initialValue

Beschreibung

Der ursprüngliche ID-Wert, der bei der Erstellung von IDs mit der UACI_IDsByType-Tabelle verwendet wird.

Standardwert

1

Gültige Werte

Ein beliebiger Wert größer 0.

retries

Beschreibung

Die Anzahl der Wiederholungen, bevor eine Ausnahme ausgelöst wird, wenn IDs mit der UACI_IDsByType-Tabelle erstellt werden.

Standardwert

20

Gültige Werte

Eine beliebige Ganzzahl größer 0.

Interact | flowchart

In diesem Abschnitt werden die Konfigurationseinstellungen für interaktive Ablaufdiagramme definiert.

defaultDateFormat

Beschreibung

Das Standarddatumsformat, das von Interact zum Konvertieren eines Datums in eine Zeichenfolge bzw. einer Zeichenfolge in ein Datum verwendet wird.

Standardwert

MM/TT/JJ

idleFlowchartThreadTimeoutInMinutes

Beschreibung

Die Anzahl von Minuten, die ein Thread, der einem interaktiven Ablaufdiagramm zugewiesen ist, in Interact im Leerlauf sein kann, bevor der Thread freigegeben wird.

Standardwert

5

idleProcessBoxThreadTimeoutInMinutes

Beschreibung

Die Anzahl von Minuten, die ein Thread, der einem interaktiven Ablaufdiagrammprozess zugewiesen ist, in Interact im Leerlauf sein kann, bevor der Thread freigegeben wird.

Standardwert

5

maxSizeOfFlowchartEngineInboundQueue

Beschreibung

Die maximale Anzahl der Aufforderungen zum Ausführen eines Ablaufdiagramms, die in Interact in einer Warteschlange gehalten werden. Wenn diese Anzahl erreicht wird, hört Interact auf, Anfragen anzunehmen.

Standardwert

1000

maxNumberOfFlowchartThreads

Beschreibung

Die maximale Anzahl der Threads, die Aufforderungen für interaktive Ablaufdiagramme zugewiesen sind.

Standardwert

25

maxNumberOfProcessBoxThreads

Beschreibung

Die maximale Anzahl der Threads, die interaktiven Ablaufdiagrammprozessen zugewiesen sind.

Standardwert

50

maxNumberOfProcessBoxThreadsPerFlowchart

Beschreibung

Die maximale Anzahl der Threads, die interaktiven Ablaufdiagrammprozessen pro Ablaufdiagramminstanz zugewiesen sind.

Standardwert

3

minNumberOfFlowchartThreads

Beschreibung

Die minimale Anzahl der Threads, die Aufforderungen für interaktive Ablaufdiagramme zugewiesen sind.

Standardwert

10

minNumberOfProcessBoxThreads

Beschreibung

Die minimale Anzahl der Threads, die interaktiven Ablaufdiagrammprozessen zugewiesen sind.

Standardwert

20

sessionVarPrefix

Beschreibung

Das Präfix für Sitzungsvariablen.

Standardwert

SessionVar

Interact | flowchart | ExternalCallouts | [ExternalCalloutName]

In diesem Abschnitt werden die Klasseneinstellungen für benutzerdefinierte externe Callouts definiert, die Sie mit der externen Callout-API geschrieben haben.

class

Beschreibung

Der Name der Java-Klasse, die diesem externen Callout entspricht.

Dies ist die Java-Klasse, auf die Sie mit dem IBM Makro `EXTERNALCALLOUT` zugreifen können.

Standardwert

Kein Standardwert definiert.

classpath

Beschreibung

Der Klassenpfad für die Java-Klasse, die diesem externen Callout entspricht. Der Klassenpfad muss auf JAR-Dateien auf dem Server für die Laufzeitumgebung verweisen. Wenn Sie eine Servergruppe verwenden und alle Laufzeitserver dieselbe Marketing Platform verwenden, muss jeder Server über eine Kopie der JAR-Datei an derselben Position verfügen. Der Klassenpfad muss absolute Positionen der JAR-Dateien enthalten, die durch das Pfadtrennzeichen des Betriebssystems des Servers für die Laufzeitumgebung getrennt sind, z. B. Semikolon (;) in Windows-Systemen und Doppelpunkt (:) in UNIX-Systemen. Verzeichnisse, die Klassendateien enthalten, sind nicht zulässig. Auf einem UNIX-System beispielsweise: `/path1/file1.jar:/path2/file2.jar`.

Dieser Klassenpfad kann maximal 1024 Zeichen enthalten. Mit der Manifestdatei in einer JAR-Datei können Sie andere JAR-Dateien angeben, sodass im Klassenpfad nur eine JAR-Datei enthalten sein muss.

Dies ist die Java-Klasse, auf die Sie mit dem IBM Makro `EXTERNALCALLOUT` zugreifen können.

Standardwert

Kein Standardwert definiert.

Interact | flowchart | ExternalCallouts | [ExternalCalloutName] | Parameter Data | [parameterName]

In diesem Abschnitt werden die Parametereinstellungen für ein benutzerdefiniertes externes Callout definiert, das Sie mit der externen Callout-API geschrieben haben.

Wert

Beschreibung

Der Wert für jeden Parameter, der für die Klasse des externen Callouts erforderlich ist.

Standardwert

Kein Standardwert definiert.

Beispiel

Wenn das externe Callout den Hostnamen eines externen Servers erfordert, erstellen Sie eine Parameterkatogerie mit der Bezeichnung `host` und definieren Sie die `value`-Eigenschaft als Servernamen.

Interact | monitoring

Dieser Satz Konfigurationseigenschaften ermöglicht das Definieren von JMX-Überwachungseinstellungen. Sie müssen diese Eigenschaften nur konfigurieren, wenn Sie die JMX-Überwachung verwenden. In den Konfigurationseigenschaften für die Designumgebung von Interact müssen für das Kontakt- und Antwortverlaufsmodul separate JMX-Überwachungseigenschaften definiert werden.

protocol

Beschreibung

Definieren Sie das Protokoll für den Interact-Nachrichtenservice.

Bei der Auswahl von JMXMP müssen die folgenden JAR-Dateien im Klassenpfad in der richtigen Reihenfolge enthalten sein:

`Interact/lib/InteractJMX.jar;Interact/lib/jmxremote_optional.jar`

Standardwert

JMXMP

Gültige Werte

JMXMP | RMI

port

Beschreibung

Die Portnummer für den Nachrichtenservice.

Standardwert

9998

enableSecurity

Beschreibung

Ein boolescher Operator, der die Sicherheit für den JMXMP-Nachrichtenservice beim Interact-Laufzeitserver aktiviert oder inaktiviert. Wenn der Wert auf `true` festgelegt ist, müssen Sie einen Benutzernamen und ein Kennwort bereitstellen, um auf den Interact-Laufzeit-JMX-Service zugreifen zu können. Dieser Benutzerberechtigungsnachweis werden von Marketing Platform für den Laufzeitserver authentifiziert. Jconsole erfordert, dass bei der Anmeldung ein Kennwort angegeben werden muss.

Bei einem RMI-Protokoll hat diese Eigenschaft keine Auswirkung. Diese Eigenschaft hat keine Auswirkung auf JMX für Campaign (die Interact-Designumgebung).

Standardwert

True

Gültige Werte

True | False

Interact | profile

Dieser Satz Konfigurationseigenschaften steuert mehrere optionale Funktionen für Angebotservices, einschließlich der Angebotsunterdrückung und Bewertungsüberschreibung.

enableScoreOverrideLookup

Beschreibung

Wenn der Wert auf `True` festgelegt wird, lädt Interact die Bewertungsüberschreibungsdaten aus der `scoreOverrideTable`, wenn eine Sitzung erstellt wird. Wenn `False` festgelegt wird, lädt Interact die Marketing-Score-Überschreibungsdaten nicht, wenn eine Sitzung erstellt wird.

Wenn Sie "true" wählen, müssen Sie auch die Eigenschaft `IBM EMM > Interact > profile > Audience Levels > (Audience Level) > scoreOverrideTable` konfigurieren. Sie müssen nur die `scoreOverrideTable`-Eigenschaft für die erforderlichen Zielgruppenebenen definieren. Wenn `scoreOverrideTable` für eine Zielgruppenebene leer gelassen wird, wird die Tabelle für Bewertungsüberschreibung für die Zielgruppenebene inaktiviert.

Standardwert

`False`

Gültige Werte

True | False

enableOfferSuppressionLookup

Beschreibung

Wenn der Wert auf `True` festgelegt wird, lädt Interact die Angebotsunterdrückungsdaten aus der `offerSuppressionTable`, wenn eine Sitzung erstellt wird. Wenn `False` festgelegt wird, lädt Interact die Marketing Angebotsunterdrückungsdaten nicht, wenn eine Sitzung erstellt wird.

Wenn Sie "true" wählen, müssen Sie auch die Eigenschaft `IBM EMM > Interact > profile > Audience Levels > (Audience Level) > offerSuppressionTable` konfigurieren. Sie müssen nur die `enableOfferSuppressionLookup`-Eigenschaft für die erforderlichen Zielgruppenebenen definieren.

Standardwert

`False`

Gültige Werte

True | False

enableProfileLookup

Beschreibung

In einer Neuinstallation von Interact wird diese Eigenschaft nicht weiter unterstützt. In einer Upgrade-Installation von Interact ist diese Eigenschaft gültig bis zur ersten Bereitstellung.

Das Ladeverhalten für eine Tabelle, die in einem interaktiven Ablaufdiagramm verwendet wird, aber nicht im interaktiven Kanal zugeordnet ist.

Wenn der Wert auf True festgelegt wird, lädt Interact die Profildaten aus der profileTable, wenn eine Sitzung erstellt wird.

Wenn Sie "true" wählen, müssen Sie auch die Eigenschaft IBM EMM > Interact > profile > Audience Levels > (Audience Level) > profileTable konfigurieren.

Die Einstellung **Diese Daten in den Speicher laden, wenn eine Besuchssession startet** im Assistenten für die Zuordnung der interaktiven Kanaltabelle überschreibt diese Konfigurationseigenschaft.

Standardwert

False

Gültige Werte

True | False

defaultOfferUpdatePollPeriod

Beschreibung

Die Anzahl der Sekunden, die das System wartet, bevor es die Standardangebote im Cache mit den Werten aus der Standardangebotstabelle aktualisiert. Wenn der Wert auf -1 gesetzt ist, aktualisiert das System die Standardangebote im Cache nicht, nachdem die ursprüngliche Liste in den Cache geladen wurde, wenn der Laufzeitserver startet.

Standardwert

-1

Interact | profile | Audience Levels | [AudienceLevelName]

Dieser Satz Konfigurationseigenschaften ermöglicht das Definieren der Tabellennamen, die für zusätzliche Funktionen in Interact erforderlich sind. Sie müssen einen Tabellennamen nur definieren, wenn Sie die entsprechende Funktion verwenden.

scoreOverrideTable

Beschreibung

Der Name der Tabelle, die die Informationen zur Bewertungsüberschreibung für diese Zielgruppenebene enthält. Diese Eigenschaft ist anwendbar, wenn Sie enableScoreOverrideLookup auf **true** gesetzt haben. Sie müssen diese Eigenschaft für die Zielgruppenebenen definieren, für die Sie eine Tabelle für die Bewertungsüberschreibung aktivieren möchten. Wenn für diese Zielgruppenebene keine Tabelle für die Bewertungsüberschreibung vorhanden ist, muss diese Eigenschaft nicht definiert werden, selbst wenn enableScoreOverrideLookup auf **true** gesetzt ist.

Interact sucht diese Tabelle in den Kundentabellen, auf die die Laufzeitserver in Interact zugreifen und die durch die prodUserDataSource-Eigenschaften definiert sind.

Wenn Sie die Eigenschaft schema für diese Datenquelle definiert haben, fügt Interact vor diesem Tabellennamen das Schema ein, z. B. schema.UACI_ScoreOverride. Wenn Sie einen vollständig qualifizierten Namen eingeben, z. B. mySchema.UACI_ScoreOverride, fügt Interact den Schemanamen nicht ein.

Standardwert

UACI_ScoreOverride

offerSuppressionTable

Beschreibung

Der Name der Tabelle, die die Informationen zur Angebotsunterdrückung für diese Zielgruppenebene enthält. Sie müssen diese Eigenschaft für die Zielgruppenebenen definieren, für die Sie eine Tabelle für Angebotsunterdrückung aktivieren möchten. Wenn für diese Zielgruppenebene keine Tabelle für Angebotsunterdrückung vorhanden ist, muss diese Eigenschaft nicht definiert werden. Wenn `enableOfferSuppressionLookup` auf "true" gesetzt ist, muss für diese Eigenschaft eine gültige Tabelle festgelegt werden.

Interact sucht diese Tabelle in den Kundentabellen, auf die die Laufzeitserver zugreifen und die durch die `prodUserDataSource`-Eigenschaften definiert sind.

Standardwert

UACI_BlackList

profileTable

Beschreibung

In einer Neuinstallation von Interact wird diese Eigenschaft nicht weiter unterstützt. In einer Upgrade-Installation von Interact ist diese Eigenschaft gültig bis zur ersten Bereitstellung.

Der Name der Tabelle, die die Profildaten für diese Zielgruppenebene enthält.

Interact sucht diese Tabelle in den Kundentabellen, auf die die Laufzeitserver zugreifen und die durch die `prodUserDataSource`-Eigenschaften definiert sind.

Wenn Sie die Eigenschaft `schema` für diese Datenquelle definiert haben, fügt Interact vor diesem Tabellennamen das Schema ein, z. B. `schema.UACI_usrProd`. Wenn Sie einen vollständig qualifizierten Namen eingeben, z. B. `mySchema.UACI_usrProd`, fügt Interact den Schemanamen nicht ein.

Standardwert

Kein Standardwert definiert.

contactHistoryTable

Beschreibung

Der Name der Staging-Tabelle für die Kontaktverlaufsdaten für diese Zielgruppenebene.

Diese Tabelle wird in den Tabellen der Laufzeitumgebung gespeichert (`systemTablesDataSource`).

Wenn Sie die Eigenschaft `schema` für diese Datenquelle definiert haben, fügt Interact vor diesem Tabellennamen das Schema ein, z. B. `schema.UACI_CHStaging`. Wenn Sie einen vollständig qualifizierten Namen eingeben, z. B. `mySchema.UACI_CHStaging`, fügt Interact den Schemanamen nicht ein.

Wenn die Protokollierung des Kontaktverlaufs inaktiviert ist, muss diese Eigenschaft nicht festgelegt werden.

Standardwert

UACI_CHStaging

chOfferAttribTable

Beschreibung

Der Name der Tabelle für die Angebotsattribute des Kontaktverlaufs für diese Zielgruppenebene.

Diese Tabelle wird in den Tabellen der Laufzeitumgebung gespeichert (`systemTablesDataSource`).

Wenn Sie die Eigenschaft `schema` für diese Datenquelle definiert haben, fügt Interact vor diesem Tabellennamen das Schema ein, z. B. `schema.UACI_CHOfferAttrib`. Wenn Sie einen vollständig qualifizierten Namen eingeben, z. B. `mySchema.UACI_CHOfferAttrib`, fügt Interact den Schemanamen nicht ein.

Wenn die Protokollierung des Kontaktverlaufs inaktiviert ist, muss diese Eigenschaft nicht festgelegt werden.

Standardwert

UACI_CHOfferAttrib

responseHistoryTable

Beschreibung

Der Name der Staging-Tabelle für den Antwortverlauf für diese Zielgruppenebene.

Diese Tabelle wird in den Tabellen der Laufzeitumgebung gespeichert (`systemTablesDataSource`).

Wenn Sie die Eigenschaft `schema` für diese Datenquelle definiert haben, fügt Interact vor diesem Tabellennamen das Schema ein, z. B. `schema.UACI_RHStaging`. Wenn Sie einen vollständig qualifizierten Namen eingeben, z. B. `mySchema.UACI_RHStaging`, fügt Interact den Schemanamen nicht ein.

Wenn die Protokollierung des Antwortverlaufs inaktiviert ist, muss diese Eigenschaft nicht festgelegt werden.

Standardwert

UACI_RHStaging

crossSessionResponseTable

Beschreibung

Der Name der Tabelle für diese Zielgruppenebene, die für die sitzungsübergreifende Antwortverfolgung in den Kontakt- und Antwortverlaufstabellen erforderlich ist, auf die die Funktion für die Antwortverfolgung zugreifen kann.

Wenn Sie die Eigenschaft `schema` für diese Datenquelle definiert haben, fügt Interact vor diesem Tabellennamen das Schema ein, z. B. `schema.UACI_XSessResponse`. Wenn Sie einen vollständig qualifizierten Namen eingeben, z. B. `mySchema.UACI_XSessResponse`, fügt Interact den Schemanamen nicht ein.

Wenn die Protokollierung von sitzungsübergreifenden Antworten inaktiviert ist, muss diese Eigenschaft nicht festgelegt werden.

Standardwert

UACI_XSessResponse

userEventLoggingTable

Beschreibung

Dies ist der Name der Datenbanktabelle, die zur Protokollierung von benutzerdefinierten Ereignisaktivitäten verwendet wird. Benutzer definieren Ereignisse auf der Registerkarte "Ereignisse" der Übersichtsseiten des interaktiven Kanals auf der Interact-Benutzeroberfläche. Die Datenbanktabelle, die Sie hier angeben, speichert Informationen wie die Ereignis-ID, den Ereignisnamen, die Häufigkeit des Eintretens dieses Ereignisses für diese Zielgruppenebene seit der letzten Löschung des Ereignisaktivitätscaches usw.

Wenn Sie die Eigenschaft schema für diese Datenquelle definiert haben, fügt Interact vor diesem Tabellennamen das Schema ein, z. B. schema.UACI_UserEventActivity. Wenn Sie einen vollständig qualifizierten Namen eingeben, z. B. mySchema.UACI_UserEventActivity, fügt Interact den Schemanamen nicht ein.

Standardwert

UACI_UserEventActivity

patternStateTable

Beschreibung

Dies ist der Name der Datenbanktabelle, die zur Protokollierung von Ereignismusterzuständen verwendet wird, beispielsweise, ob die Musterbedingung erfüllt wurde oder nicht, ob das Muster abgelaufen oder inaktiviert ist usw.

Wenn Sie die Eigenschaft schema für diese Datenquelle definiert haben, fügt Interact vor diesem Tabellennamen das Schema ein, z. B. schema.UACI_EventPatternState. Wenn Sie einen vollständig qualifizierten Namen eingeben, z. B. mySchema.UACI_EventPatternState, fügt Interact den Schemanamen nicht ein.

Wenn Sie keine Ereignismuster verwenden, ist für jede Zielgruppenebene eine Angabe für "patternStateTable" erforderlich. Der Wert von "patternStateTable" basiert auf dem ddl-Wert der enthaltenen Eigenschaft UACI_EventPatternState. Es folgt ein Beispiel, bei dem die Zielgruppen-ID zwei Komponenten aufweist: ComponentNum und ComponentStr.

```
CREATE TABLE UACI_EventPatternState_Composite
(
   UpdateTime bigint NOT NULL,
   State varbinary(4000),
   ComponentNum bigint NOT NULL,
   ComponentStr nvarchar(50) NOT NULL,
   CONSTRAINT PK_CustomerPatternState_Composite PRIMARY KEY
   (ComponentNum,ComponentStr,UpdateTime)
)
```

Standardwert

UACI_EventPatternState

Interact | profile | Audience Levels | [AudienceLevelName] | Offers by Raw SQL

Dieser Satz Konfigurationseigenschaften ermöglicht das Definieren der Tabellennamen, die für zusätzliche Funktionen in Interact erforderlich sind. Sie müssen einen Tabellennamen nur definieren, wenn Sie die entsprechende Funktion verwenden.

enableOffersByRawSQL

Beschreibung

Wenn auf `True` festgelegt, aktiviert Interact die Funktion offersBySQL für diese Zielgruppenebene, was es Ihnen ermöglicht, SQL-Code zu konfigurieren, um einen gewünschten Satz von möglichen Angeboten zur Laufzeit zu erstellen. Bei `False` verwendet Interact die Funktion offersBySQL nicht.

Wenn Sie diese Eigenschaft auf "true" festlegen, können Sie auch die Eigenschaft `Interact | profile | Audience Levels | (Audience Level) | Offers by Raw SQL | SQL Template` konfigurieren, um eine oder mehrere SQL-Vorlagen zu definieren.

Standardwert

`False`

Gültige Werte

True | False

cacheSize

Beschreibung

Größe des Cache zum Speichern von Ergebnissen der OfferBySQL-Abfragen. Beachten Sie, dass die Verwendung eines Cache negative Leistungsauswirkungen haben kann, wenn die Abfrageergebnisse für die meisten Sitzungen eindeutig sind.

Standardwert

`-1 (aus)`

Gültige Werte

-1 | Wert

cacheLifeInMinutes

Beschreibung

Wenn der Cache aktiviert ist, gibt dies die Anzahl von Minuten an, die gewartet wird, bevor das System den Cache löscht, um die Aktualität zu gewährleisten.

Standardwert

`-1 (aus)`

Gültige Werte

-1 | Wert

defaultSQLTemplate

Beschreibung

Der Name der zu verwendenden SQL-Vorlage, wenn keine über die API-Anrufe angegeben wird.

Standardwert

`Keine`

Gültige Werte

SQL-Vorlagenname

Interact | profile | Audience Levels | [AudienceLevelName] | SQL Template

Mit diesen Konfigurationseigenschaften können Sie eine oder mehrere SQL-Abfragevorlagen für die Verwendung durch die Funktion offersBySQL von Interact definieren.

name

Beschreibung

Der Name, den Sie dieser SQL-Abfragevorlage zuweisen möchten. Geben Sie einen beschreibenden Namen ein, der aussagekräftig ist, wenn Sie diese SQL-Vorlage in API-Anrufen verwenden. Beachten Sie: Wenn Sie hier einen Namen verwenden, der mit einem Namen *identisch* ist, der in einem Interact-Listenprozessfeld für ein offerBySQL-Verfahren definiert ist, wird die SQL im Prozessfeld anstelle der hier eingegebenen SQL verwendet.

Standardwert

Keiner.

SQL

Beschreibung

Enthält die SQL-Abfrage, die von dieser Vorlage aufgerufen wird. Die SQL-Abfrage kann Verweise auf Variablennamen haben, die Teil der Sitzungsdaten des Besuchers (Profil) sind. Beispiel: `select * from MyOffers where category = ${preferredCategory}` würde sich auf die Sitzung beziehen, die eine Variable namens `preferredCategory` enthält.

Sie sollten die SQL so konfigurieren, dass die speziellen Angebotstabellen abgefragt werden, die Sie während der Entwicklung zur Verwendung durch diese Funktion erstellt haben. Beachten Sie, dass hier gespeicherte Prozeduren nicht unterstützt werden.

Standardwert

Keiner.

Interact | profile | Audience Levels | [AudienceLevelName | Profile Data Services | [DataSource]

Dieser Satz Konfigurationseigenschaften ermöglicht das Definieren der Tabellennamen, die für zusätzliche Funktionen in Interact erforderlich sind. Sie müssen einen Tabellennamen nur definieren, wenn Sie die entsprechende Funktion verwenden. In der Kategorie "Profildatenservices" werden Informationen über eine integrierte Datenquelle (Datenbank) angegeben, die für alle Zielgruppenebenen erstellt wird und mit einer Priorität von 100 vorkonfiguriert ist. Sie können jedoch entscheiden, ob sie diese ändern oder inaktivieren möchten. In der Kategorie ist außerdem eine

Vorlage für zusätzliche externe Datenquellen enthalten. Wenn Sie auf die Vorlage mit dem Namen **Externe Datenservices** klicken, können Sie die hier beschriebenen Konfigurationseinstellungen abschließen.

Neuer Kategoriename

Beschreibung

(Nicht für den Standarddatenbankeintrag verfügbar.) Der Name der Datenquelle, die Sie definieren. Der Name, den Sie hier eingeben, muss innerhalb der Datenquellen einer Zielgruppenebene eindeutig sein.

Standardwert

Keiner.

Gültige Werte

Jede Textzeichenfolge ist zulässig.

aktiviert

Beschreibung

Wenn der Wert auf `True` festgelegt ist, wird diese Datenquelle für die Zielgruppenebene aktiviert, der sie zugeordnet ist. Wenn er auf `False` festgelegt ist, verwendet Interact diese Datenquelle nicht für diese Zielgruppenebene.

Standardwert

`True`

Gültige Werte

True | False

className

Beschreibung

(Nicht für den Standarddatenbankeintrag verfügbar.) Der vollständig qualifizierte Name der Datenquellenklasse, die `IInteractProfileDataService` implementiert.

Standardwert

Keiner.

Gültige Werte

Eine Zeichenfolge, die einen vollständig qualifizierten Klassennamen angibt.

classPath

Beschreibung

(Nicht für den Standarddatenbankeintrag verfügbar.) Eine optionale Konfigurationseinstellung, die den Pfad zum Laden dieser Datenquellenimplementierungsklasse angibt. Wenn Sie ihn auslassen, wird standardmäßig der Klassenpfad des übergeordneten Anwendungsservers verwendet.

Standardwert

Wird nicht angezeigt, aber der Klassenpfad des übergeordneten Anwendungsservers wird standardmäßig verwendet, wenn hier kein Wert angegeben ist.

Gültige Werte

Eine Zeichenfolge, die den Klassenpfad angibt.

Priorität

Beschreibung

Die Priorität dieser Datenquelle in dieser Zielgruppenebene. Der Wert muss für jede Datenquelle in einer Zielgruppenebene eindeutig sein. (Das heißt, wenn für eine Datenquelle die Priorität 100 festgelegt ist, kann keine weitere Datenquelle in der Zielgruppenebene eine Priorität von 100 haben.)

Standardwert

`100` für die Standarddatenbank, `200` für eine benutzerdefinierte Datenquelle

Gültige Werte

Jede Ganzzahl, die nicht negativ ist, ist zulässig.

Interact | offerserving

Diese Konfigurationseigenschaften definieren die allgemeinen Lernkonfigurationseigenschaften. Verwenden Sie bei einem integrierten Lernmodul die Konfigurationseigenschaften für die Designumgebung, um Ihre Implementierung des Lernmoduls zu optimieren.

offerTieBreakMethod

Beschreibung

Die Eigenschaft `offerTieBreakMethod` definiert das Verhalten der Angebotsunterbreitung, wenn zwei Angebote gleichwertige (unentschiedene) Bewertungen haben. Wenn Sie diese Eigenschaft auf ihren Standardwert "Random" (Zufällig) festlegen, präsentiert Interact eine zufällige Auswahl aus den gleichwertigen Angeboten. Wenn Sie diese Eigenschaft auf "Newer Offer" (Neueres Angebot) festlegen, präsentiert Interact bei Angeboten, die dieselbe Bewertung haben, das neuere Angebot (mit der höheren Angebots-ID) vor dem älteren Angebot (mit der niedrigeren Angebots-ID).

Anmerkung:

Interact verfügt über eine optionale Funktion, über die der Administrator das System darauf konfigurieren kann, die Angebote in zufälliger Reihenfolge unabhängig von der Bewertung zurückzugeben, indem er die Option `percentRandomSelection (Campaign | partitions | [partition_number] | Interact | learning | percentRandomSelection)` festlegt. Die hier beschriebene Eigenschaft `offerTieBreakMethod` wird nur verwendet, wenn `percentRandomSelection` auf Null festgelegt (inaktiviert) wird.

Standardwert

Wahlfrei

Gültige Werte

Random | Newer Offer

optimizationType

Beschreibung

Die Eigenschaft `optimizationType` legt fest, ob Interact ein Lernmodul zur Unterstützung bei Angebotszuordnungen verwendet. Wenn der Wert auf `NoLearning` festgelegt wird, verwendet Interact kein Lernmodul. Wenn `BuiltInLearning` festgelegt wird, verwendet Interact das Bayes-Lernmodul, das mit Interact erstellt wird. Wenn `ExternalLearning` festgelegt wird, verwendet Interact ein von Ihnen bereitgestelltes Lernmodul. Wenn Sie `ExternalLearning` auswählen, müssen Sie die Eigenschaften `externalLearningClass` und `externalLearningClassPath` definieren.

Standardwert

NoLearning

Gültige Werte

NoLearning | BuiltInLearning | ExternalLearning

segmentationMaxWaitTimeInMS

Beschreibung

Die maximale Dauer in Millisekunden, die der Laufzeitserver wartet, bis ein interaktives Ablaufdiagramm abgeschlossen ist, bevor Angebote angenommen werden.

Standardwert

`5000`

treatmentCodePrefix

Beschreibung

Das Präfix, das in Verfahrenscodes eingefügt wird.

Standardwert

Kein Standardwert definiert.

effectiveDateBehavior

Beschreibung

Legt fest, ob Interact beim Filtern der Angebote, die einem Besucher präsentiert werden, jeweils das Gültigkeitsdatum des Angebots verwenden soll. Zulässige Werte:

- "-1" weist Interact an, das Gültigkeitsdatum des Angebots zu ignorieren.

  "0" weist Interact an, das Gültigkeitsdatum zum Filtern des Angebots zu verwenden, damit das Angebot Besuchern bereitgestellt wird, falls das Gültigkeitsdatum dem aktuellen Datum entspricht oder davor liegt.

  Wenn ein Wert für **effectiveDateGracePeriod** angegeben ist, wird die Karenzzeit auch dazu verwendet, festzulegen, ob das Angebot bereitgestellt werden soll.
- Jede positive ganze Zahl weist Interact an, das aktuelle Datum plus des Werts dieser Eigenschaft zu verwenden, um festzulegen, ob das Angebot Besuchern bereitgestellt werden soll. Dies geschieht, falls das Gültigkeitsdatum des Angebots vor dem aktuellen Datum plus des Werts dieser Eigenschaft liegt.

Wenn ein Wert für **effectiveDateGracePeriod** angegeben ist, wird die Karenzzeit auch dazu verwendet, festzulegen, ob das Angebot bereitgestellt werden soll.

Standardwert

-1

effectiveDateGracePeriodOfferAttr

Beschreibung

Gibt in einer Angebotsdefinition den Namen des benutzerdefinierten Attributs an, das die Karenzzeit des Gültigkeitsdatums angibt. Sie können für diese Eigenschaft beispielsweise den Wert `AltGracePeriod` angeben. Danach können Sie Angebote mit einem benutzerdefinierten Attribut namens `AltGracePeriod` definieren, mit dessen Hilfe die Anzahl von Tagen angegeben wird, die als Karenzzeit für die Eigenschaft **effectiveDateBehavior** dienen soll.

Nehmen wir einmal an, Sie erstellen eine neue Angebotsvorlage, die ein Gültigkeitsdatum hat, das vom aktuellen Datum aus gesehen 10 Tage in der Zukunft liegt, und Sie beziehen dabei ein benutzerdefiniertes Attribut namens `AltGracePeriod` mit ein. Wenn Sie mithilfe dieser Vorlage ein Angebot erstellen und den Wert von `AltGracePeriod` auf 14 Tage setzen, wird das Angebot Besuchern bereitgestellt, da das aktuelle Datum sich innerhalb der Karenzzeit des Gültigkeitsdatums des Angebots befindet.

Standardwert

Leer

alwaysLogLearningAttributes

Beschreibung

Gibt an, ob Interact Informationen über von dem Lernmodul verwendete Besucherattribute in der Protokolldatei erfassen soll. Beachten Sie, dass ein Setzen dieses Wertes auf "True" Auswirkungen auf das Leistungsverhalten beim Lernen und auf die Größe der Protokolldatei hat.

Standardwert

False

Interact | offerserving | Built-in Learning Config

Diese Konfigurationseigenschaften definieren die Schreibeinstellungen der Datenbank für das integrierte Lernmodul. Verwenden Sie die Konfigurationseigenschaften für die Designumgebung, um Ihre Implementierung des Lernmoduls zu optimieren.

version

Beschreibung

Sie können 1 oder 2 auswählen. Version 1 ist die Basiskonfigurationsversion, bei der keine Parameter für die Festlegung von Thread- und Datensatzgrenzwerten verwendet werden. Version 2 ist die erweiterte Konfigurationsversion, bei der Sie zur Leistungssteigerung Thread- und Datensatzparameter festlegen können. Diese Parameter führen eine Aggregation und Löschung durch, sobald diese Parametergrenzwerte erreicht sind.

Standardwert

1

insertRawStatsIntervalInMinutes

Beschreibung

Die Anzahl von Minuten, die Interact wartet, bevor weitere Zeilen in die Lern-Staging-Tabellen eingefügt werden. Abhängig von der Datenmenge, die das Lernmodul in Ihrer Umgebung verarbeitet, muss diese Dauer u. U. geändert werden.

Standardwert

5

Gültige Werte

Eine positive Ganzzahl

aggregateStatsIntervalInMinutes

Beschreibung

Die Anzahl von Minuten, die Interact zwischen dem Aggregieren von Daten in den Lernstatistiktabellen wartet. Abhängig von der Datenmenge, die das Lernmodul in Ihrer Umgebung verarbeitet, muss diese Dauer u. U. geändert werden.

Standardwert

15

Gültige Werte

Eine Ganzzahl größer 0.

autoAdjustPercentage

Beschreibung

Der Wert, der den Prozentsatz der Daten festlegt, dessen Verarbeitung die Aggregationsausführung auf Basis der Metriken der vorherigen Ausführung versucht. Standardmäßig ist dieser Wert auf null gesetzt. Dies bedeutet, dass der Aggregator alle Staging-Datensätze verarbeitet. In diesem Fall ist diese Funktionalität der automatischen Anpassung inaktiviert.

Standardwert

0

Gültige Werte

Eine Zahl zwischen 0 und 100.

enableObservationModeOnly

Beschreibung

Der Wert `True` aktiviert einen Lernmodus, bei dem Interact Daten sammelt, um daraus zu lernen, ohne diese Daten auch für Empfehlungen oder Angebotsarbitrierung zu verwenden. Dadurch können Sie einen Selbstlernmodus als Startmodus ausführen, bis eine ausreichende Datenmenge für Empfehlungen gesammelt wurde.

Standardwert

False

Gültige Werte

True | False

excludeAbnormalAttribute

Beschreibung

Die Einstellung, die bestimmt, ob diese Attribute als ungültig markiert werden. Bei der Einstellung `IncludeAttribute` werden abnormale Attribute eingeschlossen und nicht als ungültig markiert. Bei der Einstellung `ExcludeAttribute` werden abnormale Attribute ausgeschlossen und als ungültig markiert.

Standardwert

IncludeAttribute

Gültige Werte

IncludeAttribute | ExcludeAttribute

Interact | offerserving | Built-in Learning Config | Parameter Data | [parameterName]

Diese Konfigurationseigenschaften definieren alle Parameter für das externe Lernmodul.

numberOfThreads

Beschreibung

Die maximale Anzahl der Threads, die der Lernaggregator für die Verarbeitung der Daten verwendet. Ein gültiger Wert ist eine positive Ganzzahl. Der Wert sollte die maximale Anzahl der Verbindungen, die in der Lerndatenquelle konfiguriert sind, nicht überschreiten. Dieser Parameter wird nur von der Version 2 des Aggregators verwendet.

Standardwert

10

maxLogTimeSpanInMin

Beschreibung

Wenn die Version 1 des Aggregators ausgewählt ist, können Sie die Staging-Datensätze in Iterationen verarbeiten, um extrem umfangreiche Datenbankstapel zu vermeiden. In diesem Fall werden die betreffenden Staging-Datensätze nach Blöcken verarbeitet, und zwar Iteration für Iteration in einem einzelnen Aggregationszyklus. Der Wert dieses Parameters gibt die maximale Zeitspanne für Staging-Datensätze an, deren Verarbeitung der Aggregator in jeder Iteration versucht. Diese Zeitspanne basiert auf dem Feld "LogTime", das jedem Staging-Datensatz zugeordnet ist. Nur die Datensätze, deren LogTime in das früheste Zeitfenster fallen, werden verarbeitet. Ein gültiger Wert ist eine Ganzzahl, die nicht negativ ist. Bei dem Wert 0 ist kein Grenzwert festgelegt, sodass alle Staging-Datensätze in einer einzelnen Iteration verarbeitet werden.

Standardwert

0

maxRecords

Beschreibung

Wenn die Version 2 des Aggregators ausgewählt ist, können Sie die Staging-Datensätze in Iterationen verarbeiten, um extrem umfangreiche Datenbankstapel zu vermeiden. In diesem Fall werden die betreffenden Staging-Datensätze in Blöcken verarbeitet, und zwar Iteration für Iteration in einem einzelnen Aggregationszyklus. Der Wert dieses Parameters gibt die maximale Anzahl der Staging-Datensätze an, deren Verarbeitung der Aggregator in jeder Iteration versucht. Ein gültiger Wert ist eine Ganzzahl, die nicht negativ ist. Bei dem Wert 0 ist kein Grenzwert festgelegt, sodass alle Staging-Datensätze in einer einzelnen Iteration verarbeitet werden.

Standardwert

0

Wert

Beschreibung

Der Wert für jeden Parameter, der von der Klasse für ein integriertes Lernmodul benötigt wird.

Standardwert

Kein Standardwert definiert.

Interact | offerserving | External Learning Config

Diese Konfigurationseigenschaften definieren die Klasseneinstellungen für ein externes Lernmodul, das Sie mit der Lern-API geschrieben haben.

class

Beschreibung

Wenn `optimizationType` auf `ExternalLearning` gesetzt ist, legen Sie `externalLearningClass` auf den Klassennamen für das externe Lernmodul fest.

Standardwert

Kein Standardwert definiert.

Verfügbarkeit

Diese Eigenschaft ist nur anwendbar, wenn `optimizationType` auf `ExternalLearning` festgelegt ist.

classPath

Beschreibung

Wenn `optimizationType` auf `ExternalLearning` gesetzt ist, legen Sie `externalLearningClass` auf den Klassenpfad für das externe Lernmodul fest.

Der Klassenpfad muss auf JAR-Dateien auf dem Server für die Laufzeitumgebung verweisen. Wenn Sie eine Servergruppe verwenden und alle Laufzeitserver dieselbe Marketing Platform verwenden, muss jeder Server über eine Kopie der JAR-Datei an derselben Position verfügen. Der Klassenpfad muss absolute Positionen der JAR-Dateien enthalten, die durch das Pfadtrennzeichen des Betriebssystems des Servers für die Laufzeitumgebung ge-

trennt sind, z. B. Semikolon (;) in Windows-Systemen und Doppelpunkt (:) in UNIX-Systemen. Verzeichnisse, die Klassendateien enthalten, sind nicht zulässig. Auf einem UNIX-System beispielsweise: `/path1/file1.jar:/path2/file2.jar`.

Dieser Klassenpfad kann maximal 1024 Zeichen enthalten. Mit der Manifestdatei in einer JAR-Datei können Sie andere JAR-Dateien angeben, sodass im Klassenpfad nur eine JAR-Datei enthalten sein muss.

Standardwert

Kein Standardwert definiert.

Verfügbarkeit

Diese Eigenschaft ist nur anwendbar, wenn `optimizationType` auf `ExternalLearning` festgelegt ist.

Interact | offerserving | External Learning Config | Parameter Data | [parameterName]

Diese Konfigurationseigenschaften definieren alle Parameter für das externe Lernmodul.

Wert

Beschreibung

Der Wert für jeden Parameter, der für die Klasse eines externen Lernmoduls erforderlich ist.

Standardwert

Kein Standardwert definiert.

Beispiel

Wenn das externe Lernmodul einen Pfad zu einer Algorithmuslösung erfordert, erstellen Sie eine Parameterkategorie mit der Bezeichnung `solverPath` und definieren Sie die Eigenschaft `value` als Pfad zu der Anwendung.

Interact | services

Die Konfigurationseigenschaften in dieser Kategorie definieren Einstellungen für alle Services, die das Sammeln von Kontakt- und Antwortverlaufsdaten sowie Statistiken für die Berichterstellung und Schreibvorgänge in die Systemtabellen der Laufzeitumgebung verwalten.

externalLoaderStagingDirectory

Beschreibung

Diese Eigenschaft definiert die Position für das Staging-Verzeichnis für ein Datenbankladedienstprogramm.

Standardwert

Kein Standardwert definiert.

Gültige Werte

Ein Pfad, der sich auf das Installationsverzeichnis von Interact bezieht oder ein absoluter Pfad zu einem Staging-Verzeichnis.

Wenn Sie ein Datenbankladedienstprogramm aktivieren, müssen Sie die Eigenschaft cacheType in den Kategorien contactHist und responstHist auf External Loader File setzen.

Interact | services | contactHist

Die Konfigurationseigenschaften in dieser Kategorie definieren die Einstellungen für den Service, der Daten für die Staging-Tabellen für den Kontaktverlauf sammelt.

enableLog

Beschreibung

Wenn der Wert auf true festgelegt ist, ist der Service aktiviert, der Daten für die Aufzeichnung der Kontaktverlaufsdaten sammelt. Bei false werden keine Daten gesammelt.

Standardwert

True

Gültige Werte

True | False

cacheType

Beschreibung

Definiert, ob die für den Kontaktverlauf gesammelten Daten im Speicher (Memory Cache) oder in einer Datei (External Loader file) gespeichert werden. Sie können External Loader File nur verwenden, wenn Interact für die Verwendung eines Datenbankladeprogramms konfiguriert ist.

Wenn Sie Memory Cache auswählen, verwenden Sie die Kategorieeinstellungen cache. Wenn Sie External Loader File auswählen, verwenden Sie die Kategorieeinstellungen fileCache.

Standardwert

Memory Cache

Gültige Werte

Memory Cache | External Loader File

Interact | services | contactHist | cache

Die Konfigurationseigenschaften in dieser Kategorie definieren die Cache-Einstellungen für den Service, der Daten für die Staging-Tabelle für den Kontaktverlauf sammelt.

threshold

Beschreibung

Die Anzahl der Datensätze, die angehäuft werden, bevor der flushCacheToDB-Service die gesammelten Kontaktverlaufsdaten in die Datenbank schreibt.

Standardwert

100

insertPeriodInSecs

Beschreibung

Die Anzahl der Sekunden zwischen erzwungenen Schreibvorgängen in die Datenbank.

Standardwert

3600

Interact | services | contactHist | fileCache

Die Konfigurationseigenschaften in dieser Kategorie definieren die Cache-Einstellungen für den Service, der die Kontaktverlaufsdaten sammelt, wenn Sie ein Datenbankladeprogramm verwenden.

threshold

Beschreibung

Die Anzahl der Datensätze, die angehäuft werden, bevor der flushCacheToDB-Service die gesammelten Kontaktverlaufsdaten in die Datenbank schreibt.

Standardwert

100

insertPeriodInSecs

Beschreibung

Die Anzahl der Sekunden zwischen erzwungenen Schreibvorgängen in die Datenbank.

Standardwert

3600

Interact | services | defaultedStats

Die Konfigurationseigenschaften in dieser Kategorie definieren die Einstellungen für den Service, der Statistiken darüber sammelt, wie oft die Standardzeichenfolge für den Interaktionspunkt verwendet wurde.

enableLog

Beschreibung

Wenn der Wert auf `true` festgelegt ist, ist der Service aktiviert, der Statistiken, wie oft die Standardzeichenfolge für den Interaktionspunkt verwendet wurde, in der `UACI_DefaultedStat`-Tabelle sammelt. Bei `false` werden keine Statistiken über die Standardzeichenfolge gesammelt.

Wenn Sie die IBM Berichterstellung nicht verwenden, können Sie diese Eigenschaft auf `false` setzen, da keine Datensammlung erforderlich ist.

Standardwert

True

Gültige Werte

True | False

Interact | services | defaultedStats | cache

Die Konfigurationseigenschaften in dieser Kategorie definieren die Cache-Einstellungen für den Service, der Statistiken darüber sammelt, wie oft die Standardzeichenfolge für den Interaktionspunkt verwendet wurde.

threshold

Beschreibung

Die Anzahl der Datensätze, die angehäuft werden, bevor der flushCacheToDB-Service die gesammelten Statistiken über die Standardzeichenfolge in die Datenbank schreibt.

Standardwert

100

insertPeriodInSecs

Beschreibung

Die Anzahl der Sekunden zwischen erzwungenen Schreibvorgängen in die Datenbank.

Standardwert

3600

Interact | services | eligOpsStats

Die Konfigurationseigenschaften in dieser Kategorie definieren die Einstellungen für den Service, der die Statistiken über berechtigte Angebote schreibt.

enableLog

Beschreibung

Wenn der Wert auf `true` festgelegt ist, ist der Service aktiviert, der Statistiken über berechtigte Angebote sammelt. Bei `false` werden keine Statistiken über berechtigte Angebote gesammelt.

Wenn Sie die IBM Berichterstellung nicht verwenden, können Sie diese Eigenschaft auf `false` setzen, da keine Datensammlung erforderlich ist.

Standardwert

True

Gültige Werte

True | False

Interact | services | eligOpsStats | cache

Die Konfigurationseigenschaften in dieser Kategorie definieren die Cache-Einstellungen für den Service, der Statistiken über berechtigte Angebote sammelt.

threshold

Beschreibung

Die Anzahl der Datensätze, die angehäuft werden, bevor der flushCacheToDB-Service die gesammelten Statistiken über berechtigte Angebote in die Datenbank schreibt.

Standardwert

100

insertPeriodInSecs

Beschreibung

Die Anzahl der Sekunden zwischen erzwungenen Schreibvorgängen in die Datenbank.

Standardwert

3600

Interact | services | eventActivity

Die Konfigurationseigenschaften in dieser Kategorie definieren die Einstellungen für den Service, der Ereignisaktivitätsstatistiken sammelt.

enableLog

Beschreibung

Wenn der Wert auf `true` festgelegt ist, ist der Service aktiviert, der Ereignisaktivitätsstatistiken sammelt. Bei `false` werden keine Ereignisstatistiken gesammelt.

Wenn Sie die IBM Berichterstellung nicht verwenden, können Sie diese Eigenschaft auf `false` setzen, da keine Datensammlung erforderlich ist.

Standardwert

True

Gültige Werte

True | False

Interact | services | eventActivity | cache

Die Konfigurationseigenschaften in dieser Kategorie definieren die Cache-Einstellungen für den Service, der Ereignisaktivitätsstatistiken sammelt.

threshold

Beschreibung

Die Anzahl der Datensätze, die angehäuft werden, bevor der flushCacheToDB-Service die gesammelten Ereignisaktivitätsstatistiken in die Datenbank schreibt.

Standardwert

100

insertPeriodInSecs

Beschreibung

Die Anzahl der Sekunden zwischen erzwungenen Schreibvorgängen in die Datenbank.

Standardwert

3600

Interact | services | eventPattern

Die Konfigurationseigenschaften in der eventPattern-Kategorie definieren die Einstellungen für den Service, der Aktivitätsstatistiken für Ereignismuster sammelt.

persistUnknownUserStates

Beschreibung

Bestimmt, ob die Ereignismusterzustände für eine unbekannte Zielgruppen-ID (Besucher) in der Datenbank beibehalten werden. Standardmäßig werden beim Beenden einer Sitzung die Status aller aktualisierten Ereignismuster, die der Zielgruppen-ID des Besuchers zugeordnet sind, in der Datenbank gespeichert, sofern die Zielgruppen-ID bekannt ist (das heißt, das Profil des Besuchers kann in der Profildatenquelle gefunden werden).

Die Eigenschaft persistUnknownUserStates bestimmt die Vorgehensweise, wenn die Zielgruppen-ID unbekannt ist. Standardmäßig ist diese Eigenschaft auf False festgelegt, und bei unbekannten Zielgruppen-IDs werden die Ereignismusterzustände am Ende der Sitzung verworfen.

Wenn Sie diese Eigenschaft auf True festlegen, werden die Ereignismusterzustände von unbekannten Benutzern (deren Profil im konfigurierten Profildatenservice nicht gefunden werden kann) beibehalten.

Standardwert

False

Gültige Werte

True | False

mergeUnknowUserInSessionStates

Beschreibung

Bestimmt, wie die Ereignismusterzustände für unbekannte Zielgruppen-IDs (Besucher) beibehalten werden. Wenn die Zielgruppen-ID in der Mitte einer Sitzung wechselt, versucht Interact, die gespeicherten Ereignismusterzustände für die neue Zielgruppen-ID aus der Datenbanktabelle zu laden. Wenn die Zielgruppen-ID zuvor unbekannt war und Sie die Eigenschaft mergeUnknowUserInSessionStates auf True festlegen, werden die zu der vorherigen Zielgruppen-ID gehörenden Benutzerereignisaktivitäten mit der neuen Zielgruppen-ID zusammengeführt.

Standardwert

False

Gültige Werte

True | False

enableUserEventLog

Beschreibung

Bestimmt, ob Benutzerereignisaktivitäten in der Datenbank protokolliert werden.

Standardwert

False

Gültige Werte

True | False

Interact | services | eventPattern | userEventCache

Die Konfigurationseigenschaften in der Kategorie `userEventCache` definieren die Einstellungen, die bestimmen, wann eine Ereignisaktivität aus dem Cache zur dauerhaften Speicherung in die Datenbank verschoben wird.

threshold

Beschreibung

Bestimmt die maximale Anzahl von Ereignismusterzuständen, die im Ereignismusterzustandscache gespeichert werden können. Wenn das Limit erreicht ist, werden die am längsten nicht verwendeten Zustände aus dem Cache gelöscht.

Standardwert

100

Gültige Werte

Die gewünschte Anzahl von Ereignismusterzuständen, die im Cache bleiben sollen.

insertPeriodInSecs

Beschreibung

Bestimmt die maximale Zeitdauer in Sekunden, die Benutzerereignisaktivitäten in die Warteschlange des Speichers eingereiht werden. Wenn das durch diese Eigenschaft angegebene Zeitlimit erreicht ist, werden diese Aktivitäten dauerhaft in der Datenbank gespeichert.

Standardwert

3600 (60 Minuten)

Gültige Werte

Die gewünschte Anzahl Sekunden.

Interact | services | eventPattern | advancedPatterns

Die Konfigurationseigenschaften in dieser Kategorie steuern, ob die Integration in Interact Advanced Patterns aktiviert ist. Außerdem definieren sie die Zeitlimitintervalle für Verbindungen mit Interact Advanced Patterns.

enableAdvancedPatterns

Beschreibung

Wenn der Wert auf `true` gesetzt ist, ist die Integration in Interact Advanced Patterns aktiviert. Bei der Einstellung `false` ist die Integration nicht aktiviert. Wenn die Integration zuvor aktiviert war, verwendet Interact die neuesten Musterzustände, die von Interact Advanced Patterns empfangen wurden.

Standardwert

True

Gültige Werte

True | False

connectionTimeoutInMilliseconds

Beschreibung

Gibt an, wie viel Zeit höchstens für die Herstellung einer HTTP-Verbindung von der Interact-Echtzeitumgebung zu Interact Advanced Patterns aufgewendet werden soll. Wenn die Anforderung das Zeitlimit überschreitet, verwendet Interact die Daten aus den Mustern, die zuletzt gespeichert wurden.

Standardwert

30

readTimeoutInMilliseconds

Beschreibung

Gibt die maximale Dauer für den Empfang von Daten an, nachdem eine HTTP-Verbindung zwischen der Interact-Echtzeitumgebung und Interact Advanced Patterns eingerichtet und eine Anforderung bezüglich des Zustandsabrufs eines Ereignismusters an Interact Advanced Patterns gesendet wurde. Wenn die Anforderung das Zeitlimit überschreitet, verwendet Interact die Daten aus den Mustern, die zuletzt gespeichert wurden.

Standardwert

100

connectionPoolSize

Beschreibung

Die Größe des HTTP-Verbindungspools für die Kommunikation zwischen der Interact-Echtzeitumgebung und Interact Advanced Patterns.

Standardwert

10

Interact | services | eventPattern | advancedPatterns | autoReconnect

Die Konfigurationseigenschaften in dieser Kategorie geben Parameter für die Funktion der automatischen Verbindungswiederholung in der Interact Advanced Patterns-Integration an.

aktivieren

Beschreibung

Legt fest, ob das System die Verbindung automatisch wiederherstellt, falls Verbindungsprobleme zwischen der Interact-Echtzeitumgebung und Interact Advanced Patterns auftreten. Mit dem Standardwert **True** wird diese Funktion aktiviert.

Standardwert

True

Gültige Werte

True | False

durationInMinutes

Beschreibung

Diese Eigenschaft gibt in Minuten das Zeitintervall an, in dem das System wiederholt auftretende Verbindungsprobleme zwischen der Interact-Echtzeitumgebung und Interact Advanced Patterns auswerten soll.

Standardwert

10

numberOfFailuresBeforeDisconnect

Beschreibung

Diese Eigenschaft gibt an, wie häufig während des angegebenen Zeitraums Verbindungsfehler auftreten dürfen, bevor das System automatisch die Verbindung zu Interact Advanced Patterns trennt.

Standardwert

3

consecutiveFailuresBeforeDisconnect

Beschreibung

Legt fest, ob die Funktion der automatischen Verbindungswiederholung nur aufeinanderfolgende Verbindungsfehler zwischen der Interact-Echtzeitumgebung und Interact Advanced Patterns auswerten soll. Wenn Sie diesen Wert auf **False** setzen, werden alle Fehler innerhalb des angegebenen Zeitintervalls ausgewertet.

Standardwert

True

sleepBeforeReconnectDurationInMinutes

Beschreibung

Das System wartet die in dieser Eigenschaft angegebene Zahl von Minuten, bevor es eine Verbindung wiederherstellt, nachdem das System die Verbindung trennt, weil es - entsprechend der Angaben in den anderen Eigenschaften dieser Kategorie - wiederholt zu Verbindungsfehlern gekommen ist.

Standardwert

5

sendNotificationAfterDisconnect

Beschreibung

Diese Eigenschaft legt fest, ob das System nach Auftreten eines Verbindungsfehlers eine E-Mail-Benachrichtigung versendet. Die Benachrichtigung enthält den Namen der Interact-Echtzeitinstanz, bei der der Fehler aufgetreten ist, und eine Angabe dazu, wie viel Zeit verstrich, bis die Verbindung wie in der Eigenschaft **sleepBeforeReconnectDurationInMinutes** angegeben wiederhergestellt wurde. Der Standardwert, bei dem Benachrichtigungen versendet werden, ist **True**.

Standardwert

True

Interact | services | customLogger

Die Konfigurationseigenschaften in dieser Kategorie definieren die Einstellungen für den Service, der benutzerdefinierte Daten sammelt, um sie in eine Tabelle zu schreiben (ein Ereignis, das den Ereignisparameter `UACICustomLoggerTableName` verwendet).

enableLog

Beschreibung

Wenn der Wert auf `true` festgelegt ist, ist die Funktion zum Konvertieren des benutzerdefinierten Protokolls in eine Tabelle aktiviert. Bei `false` hat der Ereignisparameter `UACICustomLoggerTableName` keine Auswirkung.

Standardwert

True

Gültige Werte

True | False

Interact | services | customLogger | cache

Die Konfigurationseigenschaften in dieser Kategorie definieren die Cache-Einstellungen für den Service, der benutzerdefinierte Daten in einer Tabelle sammelt (ein Ereignis, das den Ereignisparameter `UACICustomLoggerTableName` verwendet).

threshold

Beschreibung

Die Anzahl der Datensätze, die angehäuft werden, bevor der flushCacheToDB-Service die gesammelten benutzerdefinierten Daten in die Datenbank schreibt.

Standardwert

100

insertPeriodInSecs

Beschreibung

Die Anzahl der Sekunden zwischen erzwungenen Schreibvorgängen in die Datenbank.

Standardwert

3600

Interact | services | responseHist

Die Konfigurationseigenschaften in dieser Kategorie definieren die Einstellungen für den Service, der in die Staging-Tabellen für den Antwortverlauf schreibt.

enableLog

Beschreibung

Wenn der Wert auf `true` festgelegt ist, ist der Service, der in die Staging-Tabellen für den Antwortverlauf schreibt, aktiviert. Bei `false` werden keine Daten in die Staging-Tabellen für den Antwortverlauf geschrieben.

Die Staging-Tabelle für den Antwortverlauf wird durch die Eigenschaft responseHistoryTable für die Zielgruppenebene definiert. Die Standardeinstellung ist UACI_RHStaging.

Standardwert

True

Gültige Werte

True | False

cacheType

Beschreibung

Definiert, ob sich der Cache im Speicher oder in einer Datei befindet. Sie können External Loader File nur verwenden, wenn Interact für die Verwendung eines Datenbankladeprogramms konfiguriert ist.

Wenn Sie Memory Cache auswählen, verwenden Sie die Kategorieeinstellungen cache. Wenn Sie External Loader File auswählen, verwenden Sie die Kategorieeinstellungen fileCache.

Standardwert

Memory Cache

Gültige Werte

Memory Cache | External Loader File

actionOnOrphan

Beschreibung

Diese Einstellung legt fest, wie mit Antwortereignissen verfahren werden soll, die keine entsprechenden Kontaktereignisse aufweisen. Bei der Einstellung NoAction wird das Antwortereignis so verarbeitet, als ob das entsprechende Kontaktereignis übergeben worden wäre. Bei der Einstellung Warning wird das Antwortereignis so verarbeitet, als ob das entsprechende Kontaktereignis übergeben worden wäre, es wird jedoch eine Warnnachricht in das Protokoll interact.log geschrieben. Bei der Einstellung Skip wird das Antwortereignis nicht verarbeitet und es wird eine Fehlernachricht in das Protokoll interact.log geschrieben. Die von Ihnen hier gewählte Einstellung ist unabhängig davon, ob die Protokollierung für den Antwortverlauf aktiviert wurde, wirksam.

Standardwert

NoAction

Gültige Werte

NoAction | Warning | Skip

Interact | services | responseHist | cache

Die Konfigurationseigenschaften in dieser Kategorie definieren die Cache-Einstellungen für den Service, der die Antwortverlaufsdaten sammelt.

threshold

Beschreibung

Die Anzahl der Datensätze, die angehäuft werden, bevor der flushCache-ToDB-Service die gesammelten Antwortverlaufsdaten in die Datenbank schreibt.

Standardwert

100

insertPeriodInSecs

Beschreibung

Die Anzahl der Sekunden zwischen erzwungenen Schreibvorgängen in die Datenbank.

Standardwert

3600

Interact | services | responseHist | fileCache

Die Konfigurationseigenschaften in dieser Kategorie definieren die Cache-Einstellungen für den Service, der die Antwortverlaufsdaten sammelt, wenn Sie ein Datenbankladeprogramm verwenden.

threshold

Beschreibung

Die Anzahl der Datensätze, die angehäuft werden, bevor sie von Interact in die Datenbank geschrieben werden.

`responseHist` – Die Tabelle, die durch die Eigenschaft `responseHistoryTable` für die Zielgruppenebene definiert ist. Die Standardeinstellung ist `UACI_RHStaging`.

Standardwert

100

insertPeriodInSecs

Beschreibung

Die Anzahl der Sekunden zwischen erzwungenen Schreibvorgängen in die Datenbank.

Standardwert

3600

Interact | services | crossSessionResponse

Die Konfigurationseigenschaften in dieser Kategorie definieren allgemeine Einstellungen für den crossSessionResponse-Service und das xsession-Verfahren. Sie müssen diese Einstellungen nur konfigurieren, wenn Sie die sitzungsübergreifende Antwortverfolgung in Interact verwenden.

enableLog

Beschreibung

Wenn der Wert auf `true` festgelegt wird, wird der Service "crossSessionResponse" aktiviert und Interact schreibt Daten in die Staging-Tabellen der sit-

zungsübergreifenden Antwortverfolgung. Wenn der Wert auf `false` festgelegt ist, ist der crossSessionResponse-Service inaktiviert.

Standardwert

False

xsessionProcessIntervalInSecs

Beschreibung

Die Anzahl der Sekunden zwischen Ausführungen des xsession-Verfahrens. Dieses Verfahren verschiebt Daten aus den Staging-Tabellen für die sitzungsübergreifende Antwortverfolgung in die Staging-Tabellen für den Antwortverlauf und das integrierte Lernmodul.

Standardwert

180

Gültige Werte

Eine Ganzzahl größer 0.

purgeOrphanResponseThresholdInMinutes

Beschreibung

Die Anzahl der Minuten, die der crossSessionResponse-Service wartet, bevor Antworten gekennzeichnet werden, die nicht mit den Kontakten in den Kontakt- und Antwortverlaufstabellen übereinstimmen.

Wenn für eine Antwort kein Treffer in den Kontakt- und Antwortverlaufstabellen gefunden wird, wird die Antwort nach `purgeOrphanResponseThresholdInMinutes` Minuten von Interact in der Spalte `Mark` der xSessResponse-Staging-Tabelle mit dem Wert -1 gekennzeichnet. Diese Antworten können dann manuell zugewiesen oder gelöscht werden.

Standardwert

180

Interact | services | crossSessionResponse | cache

Die Konfigurationseigenschaften in dieser Kategorie definieren die Cache-Einstellungen für den Service, der sitzungsübergreifende Antwortdaten sammelt.

threshold

Beschreibung

Die Anzahl der Datensätze, die angehäuft werden, bevor der flushCacheToDB-Service die gesammelten sitzungsübergreifenden Antwortdaten in die Datenbank schreibt.

Standardwert

100

insertPeriodInSecs

Beschreibung

Die Anzahl der Sekunden zwischen erzwungenen Schreibvorgängen in die `XSessResponse`-Tabelle.

Standardwert

3600

Interact | services | crossSessionResponse | OverridePerAudience | [AudienceLevel] | TrackingCodes | byTreatmentCode

Die Eigenschaften in diesem Abschnitt definieren, wie die sitzungsübergreifende Antwortverfolgung Verfahrenscodes dem Kontakt- und Antwortverlauf zuweist.

SQL

Beschreibung

Diese Eigenschaft legt fest, ob Interact die systemgenerierte SQL oder die benutzerdefinierte SQL aus der Eigenschaft `OverrideSQL` verwendet.

Standardwert

Use System Generated SQL

Gültige Werte

Use System Generated SQL | Override SQL

OverrideSQL

Beschreibung

Wenn Sie nicht den Standard-SQL-Befehl verwenden, um den Verfahrenscode dem Kontakt- und Antwortverlauf zuzuweisen, geben Sie hier die SQL oder die gespeicherte Prozedur ein.

Dieser Wert wird ignoriert, wenn `SQL` auf `Use System Generated SQL` festgelegt ist.

Standardwert

useStoredProcedure

Beschreibung

Wenn der Wert auf **true** festgelegt ist, muss `OverrideSQL` eine Referenz auf eine gespeicherte Prozedur enthalten, das den Verfahrenscode dem Kontakt- und Antwortverlauf zuweist.

Wenn der Wert auf **false** festgelegt ist, muss `OverrideSQL`, falls verwendet, eine SQL-Abfrage sein.

Standardwert

false

Gültige Werte

true | false

Typ

Beschreibung

Der zugewiesene TrackingCodeType, der in der `UACI_TrackingType`-Tabelle in den Tabellen der Laufzeitumgebung definiert ist. Wenn Sie die `UACI_TrackingType`-Tabelle nicht überarbeiten, muss `Type` 1 sein.

Standardwert

1

Gültige Werte

Eine Ganzzahl, die in der UACI_TrackingType-Tabelle definiert ist.

Interact | services | crossSessionResponse | OverridePerAudience | [AudienceLevel] | TrackingCodes | byOfferCode

Die Eigenschaften in diesem Abschnitt definieren, wie die sitzungsübergreifende Antwortverfolgung Angebotscodes dem Kontakt- und Antwortverlauf zuweist.

SQL

Beschreibung

Diese Eigenschaft legt fest, ob Interact die systemgenerierte SQL oder die benutzerdefinierte SQL aus der Eigenschaft OverrideSQL verwendet.

Standardwert

Use System Generated SQL

Gültige Werte

Use System Generated SQL | Override SQL

OverrideSQL

Beschreibung

Wenn Sie nicht den Standard-SQL-Befehl verwenden, um den Angebotscode dem Kontakt- und Antwortverlauf zuzuweisen, geben Sie hier die SQL oder die gespeicherte Prozedur ein.

Dieser Wert wird ignoriert, wenn SQL auf Use System Generated SQL festgelegt ist.

Standardwert

useStoredProcedure

Beschreibung

Wenn der Wert auf **true** festgelegt ist, muss OverrideSQL eine Referenz auf die gespeicherte Prozedur enthalten, die den Angebotscode dem Kontakt- und Antwortverlauf zuweist.

Wenn der Wert auf **false** festgelegt ist, muss OverrideSQL, falls verwendet, eine SQL-Abfrage sein.

Standardwert

false

Gültige Werte

true | false

Typ

Beschreibung

Der zugewiesene TrackingCodeType, der in der UACI_TrackingType-Tabelle in den Tabellen der Laufzeitumgebung definiert ist. Wenn Sie die UACI_TrackingType-Tabelle nicht überarbeiten, muss Type 2 sein.

Standardwert

2

Gültige Werte

Eine Ganzzahl, die in der UACI_TrackingType-Tabelle definiert ist.

Interact | services | crossSessionResponse | OverridePerAudience | [AudienceLevel] | TrackingCodes | byAlternateCode

Die Eigenschaften in diesem Abschnitt definieren, wie die sitzungsübergreifende Antwortverfolgung benutzerdefinierten alternativen Code dem Kontakt- und Antwortverlauf zuweist.

Name

Beschreibung

Diese Eigenschaft definiert den Namen für den alternativen Code. Dieser Name muss mit dem Namen in der UACI_TrackingType-Tabelle in den Tabellen der Laufzeitumgebung übereinstimmen.

Standardwert

OverrideSQL

Beschreibung

Der SQL-Befehl oder die gespeicherte Prozedur, die den alternativen Code dem Kontakt- und Antwortverlauf nach Angebotscode oder Verfahrenscode zuweisen soll.

Standardwert

useStoredProcedure

Beschreibung

Wenn der Wert auf **true** festgelegt ist, muss OverrideSQL eine Referenz auf eine gespeicherte Prozedur enthalten, die den alternativen Code dem Kontakt- und Antwortverlauf zuweist.

Wenn der Wert auf **false** festgelegt ist, muss OverrideSQL, falls verwendet, eine SQL-Abfrage sein.

Standardwert

false

Gültige Werte

true | false

Typ

Beschreibung

Der zugewiesene TrackingCodeType, der in der UACI_TrackingType-Tabelle in den Tabellen der Laufzeitumgebung definiert ist.

Standardwert

3

Gültige Werte

Eine Ganzzahl, die in der UACI_TrackingType-Tabelle definiert ist.

Interact | services | threadManagement | contactAndResponseHist

Die Konfigurationseigenschaften in dieser Kategorie definieren die Thread-Administrationseinstellungen für die Services, die Daten für die Staging-Tabellen für den Kontakt- und Antwortverlauf sammeln.

corePoolSize

Beschreibung

Die Anzahl der Threads, die im Pool gespeichert werden, auch wenn sie sich im Leerlauf befinden, um die Daten für den Kontakt- und Antwortverlauf zu sammeln.

Standardwert

5

maxPoolSize

Beschreibung

Die maximale Anzahl der Threads, die im Pool gespeichert werden, um die Daten für den Kontakt- und Antwortverlauf zu sammeln.

Standardwert

5

keepAliveTimeSecs

Beschreibung

Wenn die Anzahl der Threads größer als der Kern ist, gibt dieser Wert die maximale Dauer an, die überzählige Threads im Leerlauf auf neue Aufgaben warten, bevor sie beendet werden, um die Daten für den Kontakt- und Antwortverlauf zu sammeln.

Standardwert

5

queueCapacity

Beschreibung

Die Größe der Warteschlange des Thread-Pools zum Sammeln der Daten für den Kontakt- und Antwortverlauf.

Standardwert

1000

termWaitSecs

Beschreibung

Beim Herunterfahren des Laufzeitservers gibt dieser Wert die Anzahl der Sekunden an, die darauf gewartet wird, dass die Service-Threads das Sammeln der Daten für den Kontakt- und Antwortverlauf abschließen.

Standardwert

5

Interact | services | threadManagement | allOtherServices

Die Konfigurationseigenschaften in dieser Kategorie definieren die Thread-Administrationseinstellungen für die Services, die die Berechtigungsstatistiken für Angebote, Ereignisaktivitätsstatistiken, Statistiken zur Verwendung von Standardzeichenfolgen und die aus einem benutzerdefinierten Protokoll in eine Tabelle konvertierten Daten erfassen.

corePoolSize

Beschreibung

Die Anzahl der Threads, die, auch wenn sie sich im Leerlauf befinden, im Pool für die Services gespeichert werden, die die Berechtigungsstatistiken für Angebote, Ereignisaktivitätsstatistiken, Statistiken zur Verwendung von Standardzeichenfolgen und die aus einem benutzerdefinierten Protokoll in eine Tabelle konvertierten Daten erfassen.

Standardwert

5

maxPoolSize

Beschreibung

Die maximale Anzahl der Threads, die im Pool für die Services gespeichert werden, die die Berechtigungsstatistiken für Angebote, Ereignisaktivitätsstatistiken, Statistiken zur Verwendung von Standardzeichenfolgen und die aus einem benutzerdefinierten Protokoll in eine Tabelle konvertierten Daten erfassen.

Standardwert

5

keepAliveTimeSecs

Beschreibung

Wenn die Anzahl der Threads größer als der Kern ist, gibt dieser Wert die maximale Dauer an, die überzählige Threads im Leerlauf auf neue Aufgaben warten, bevor sie für die Services beendet werden, die die Berechtigungsstatistiken für Angebote, Ereignisaktivitätsstatistiken, Statistiken zur Verwendung von Standardzeichenfolgen und die aus einem benutzerdefinierten Protokoll in eine Tabelle konvertierten Daten erfassen.

Standardwert

5

queueCapacity

Beschreibung

Die Größe der Warteschlange des Thread-Pools für die Services, die die Berechtigungsstatistiken für Angebote, Ereignisaktivitätsstatistiken, Statistiken zur Verwendung von Standardzeichenfolgen und die aus einem benutzerdefinierten Protokoll in eine Tabelle konvertierten Daten erfassen.

Standardwert

1000

termWaitSecs

Beschreibung

Beim Herunterfahren des Laufzeitservers gibt dieser Wert die Anzahl der Sekunden an, die im Fall von Services, die die Berechtigungsstatistiken für Angebote, Ereignisaktivitätsstatistiken, Statistiken zur Verwendung von Standardzeichenfolgen und die aus einem benutzerdefinierten Protokoll in eine Tabelle konvertierten Daten erfassen, darauf gewartet wird, dass die Service-Threads für die Services abgeschlossen werden.

Standardwert

5

Interact | services | threadManagement | flushCacheToDB

Die Konfigurationseigenschaften in dieser Kategorie definieren die Thread-Administrationseinstellungen für die Threads, die gesammelte Daten im Cache in die Datenbanktabellen der Laufzeitumgebung schreiben.

corePoolSize

Beschreibung

Die Anzahl der Threads, die im Pool für geplante Threads gespeichert werden, die Daten im Cache in den Datenspeicher schreiben.

Standardwert

5

maxPoolSize

Beschreibung

Die maximale Anzahl der Threads, die im Pool für geplante Threads gespeichert werden, die Daten im Cache in den Datenspeicher schreiben.

Standardwert

5

keepAliveTimeSecs

Beschreibung

Wenn die Anzahl der Threads größer als der Kern ist, gibt dieser Wert die maximale Dauer an, die überzählige Threads im Leerlauf auf neue Aufgaben warten, bevor sie für geplante Threads beendet werden, die Daten im Cache in den Datenspeicher schreiben.

Standardwert

5

queueCapacity

Beschreibung

Die Größe der Warteschlange des Thread-Pools für geplante Threads, die Daten im Cache in den Datenspeicher schreiben.

Standardwert

1000

termWaitSecs

Beschreibung

Beim Herunterfahren des Laufzeitservers gibt dieser Wert die Anzahl der Sekunden an, die bei geplanten Threads, die Daten im Cache in den Datenspeicher schreiben, darauf gewartet wird, dass die Service-Threads abgeschlossen werden.

Standardwert

5

Interact | services | configurationMonitor

Die Konfigurationseigenschaften in dieser Kategorie ermöglichen Ihnen die Aktivierung oder Inaktivierung der Integration in Interact Advanced Patterns, ohne dass Sie die Interact-Echtzeitinstanz erneut starten müssen. Außerdem definieren sie das Intervall für das Polling des Eigenschaftswerts, der die Integration aktiviert.

enable

Beschreibung

Bei dem Wert `true` wird der Service aktiviert, der den Wert der Eigenschaft **Interact | services | eventPattern | advancedPatterns enableAdvancedPatterns** aktualisiert. Bei dem Wert `false` müssen Sie die Interact-Echtzeitinstanz erneut starten, wenn Sie den Wert der Eigenschaft **Interact | services | eventPattern | advancedPatterns enableAdvancedPatterns** ändern.

Standardwert

False

Gültige Werte

True | False

refreshIntervalInMinutes

Beschreibung

Definiert das Zeitintervall für das Polling des Wertes der Eigenschaft **Interact | services | eventPattern | advancedPatterns enableAdvancedPatterns**.

Standardwert

5

Interact | cacheManagement

Diese Gruppe von Konfigurationseigenschaften definiert die Einstellungen für die Auswahl und Konfiguration der einzelnen unterstützten Cache-Manager, die Sie zur Verbesserung der Leistung von Interact verwenden können. Beispiele für solche Cache-Manager sind Ehcache, der in Ihrer Interact-Installation integriert ist, oder WebSphere eXtreme Scale-Caching, das ein optionales Zusatzprodukt ist. Sie können auch ein anderes externes Caching-System verwenden.

Konfigurieren Sie mithilfe der Konfigurationseigenschaften**Interact | cacheManagement | Cache-Manager** den zu verwendenden Cache-Manager. Geben Sie mit

den Konfigurationseigenschaften **Interact | cacheManagement | Caches** an, welcher Cache-Manager für die Verbesserung der Leistung von Interact verwendet werden soll.

Interact | cacheManagement | Cache Managers

Die Kategorie "Cache-Manager" gibt die Parameter für die Cacheverwaltungslösungen an, die Sie zusammen mit Interact verwenden möchten.

Interact | cacheManagement | Cache Managers | EHCache

Die Kategorie "Ehcache" gibt die Parameter für die Ehcache-Cacheverwaltungslösung an, damit Sie diese zur Verbesserung der Leistung von Interact anpassen können.

Interact | Cache Managers | EHCache | Parameter Data

Die Konfigurationseigenschaften in dieser Kategorie steuern, welche Aktionen das Ehcache-Cacheverwaltungssystem ausführt, um die Leistung von Interact zu verbessern.

cacheType

Beschreibung

Sie können die Interact-Laufzeitserver in einer Servergruppe so konfigurieren, dass sie eine Multicastadresse für die gemeinsame Nutzung von Cachedaten verwenden. Dies wird als *verteilter Cache* bezeichnet. Der Parameter "cacheType" gibt an, ob Sie den integrierten Ehcache-Caching-Mechanismus im **lokalen** (eigenständigen) oder im **verteilten** Modus (wie bei einer Laufzeitservergruppe) verwenden.

Anmerkung:

Wenn Sie für den Parameter "cacheType" **Verteilt** auswählen, müssen alle Server, die den Cache gemeinsam nutzen, derselben einzelnen Servergruppe angehören. Darüber hinaus müssen Sie das Multicasting für alle Mitglieder einer Servergruppe aktivieren.

Standardwert

Local

Gültige Werte

Local | Distributed

multicastIPAddress

Beschreibung

Wenn Sie für den Parameter **cacheType** "distributed" (verteilt) angeben, konfigurieren Sie den Cache so, dass er über Multicasting zwischen allen Mitgliedern einer Interact-Laufzeitservergruppe betrieben wird. Der Wert für "multicastIPAddress" ist die IP-Adresse, die alle Interact-Server für die Servergruppe zum Empfangen verwenden.

Die IP-Adresse muss für alle Servergruppen eindeutig sein.

Standardwert

230.0.0.1

multicastPort

Beschreibung

Wenn Sie für den Parameter **cacheType** "distributed" (verteilt) angeben, gibt der Parameter **multicastPort** den Port an, den alle Interact-Server für die Servergruppe zum Empfangen verwenden.

Standardwert

6363

overflowToDisk

Beschreibung

Der Ehcache-Cache-Manager verwaltet die Sitzungsdaten unter Verwendung des verfügbaren Speichers. Bei Umgebungen mit einer Sitzungsgröße, die aufgrund eines umfangreichen Profils hoch ist, kann es vorkommen, dass die im Speicher zu unterstützende Anzahl der Sitzungen nicht für das Kundenszenario ausreicht. In diesem Fall kann Ehcache eine Zusatzfunktion nutzen, mit der Cachedaten, welche die im Speicher zulässige Menge überschreiten, stattdessen vorübergehend auf ein Festplattenlaufwerk geschrieben werden können.

Wenn Sie die Eigenschaft **overflowToDisk** auf "yes" setzen, kann jede Java Virtual Machine (JVM) mehr gleichzeitige Sitzungen verarbeiten, als dies nur mit dem Speicher möglich wäre.

Standardwert

No

Gültige Werte

No | Yes

diskStore

Beschreibung

Wenn für die Konfigurationseigenschaft **overflowToDisk** der Wert `Yes` (Ja) festgelegt ist, gibt diese Konfigurationseigenschaft das Plattenverzeichnis an, in dem die Cacheeinträge, die zu einem Überlauf im Speicher führen, gespeichert werden sollen. Wenn diese Konfigurationseigenschaft nicht vorhanden oder ihr Wert nicht gültig ist, wird das Plattenverzeichnis automatisch im standardmäßigen temporären Verzeichnis des Betriebssystems erstellt.

Standardwert

Keiner.

Gültige Werte

Ein Verzeichnis, für das die Webanwendung, die die Interact-Laufzeit hostet, Schreibberechtigungen besitzt.

(Parameter)

Beschreibung

Eine Vorlage für die Erstellung eines angepassten Parameters, der in Verbindung mit dem Cache-Manager verwendet werden soll. Sie können jeden beliebigen Parameternamen und seinen gewünschten Wert konfigurieren.

Klicken Sie zur Erstellung eines angepassten Parameters auf ***(Parameter)*** und geben Sie den Namen und den Wert an, die diesem Parameter zugewiesen werden sollen. Wenn Sie auf **Änderungen speichern** klicken, wird der von Ihnen erstellte Parameter der Liste in der Kategorie "Parameterdaten" hinzugefügt.

Standardwert

Keiner.

Interact | cacheManagement | Cache Managers | Extreme Scale

Die Kategorie "Extreme Scale" gibt die Parameter für den Adapter zur Verwendung der WebSphere eXtreme Scale-Cacheverwaltungslösung an, damit Sie diesen zur Verbesserung der Leistung von Interact anpassen können.

ClassName

Beschreibung

Der vollständig qualifizierte Name der Klasse, die Interact mit dem WebSphere eXtreme Scale-Server verbindet. Hierfür muss `com.unicacorp.interact.cache.extremescale.ExtremeScaleCacheManager` angegeben werden.

Standardwert

`com.unicacorp.interact.cache.extremescale.ExtremeScaleCacheManager`

ClassPath

Beschreibung

Der URI für die Position der Datei `interact_wxs_adapter.jar`, beispielsweise `file:///IBM/EMM/Interact/lib/interact_wxs_adapter.jar` oder `file:///C:/IBM/EMM/Interact/lib/interact_wxs_adapter.jar`. Wenn diese JAR-Datei jedoch bereits im Klassenpfad des Hostanwendungsservers enthalten ist, sollte dieses Feld leer gelassen werden.

Standardwert

Leer

Interact | Cache Managers | Extreme Scale | Parameter Data

Die Konfigurationseigenschaften in dieser Kategorie steuern den WebSphere eXtreme Scale-Adapter, der als Zusatzoption in Ihrer Interact-Installation enthalten ist. Diese Einstellungen müssen für jeden Interact-Laufzeitserver konfiguriert werden, der als Client für das eXtreme Scale-Server-Grid agiert.

catalogPropertyFile

Beschreibung

Der URI der Position der Eigenschaftendatei, die für den Start des WebSphere eXtreme Scale-Katalogservers verwendet wird. Wenn für den Start des Katalogservers der eXtreme Scale-Adapter verwendet wird, muss diese Eigenschaft festgelegt werden. Andernfalls wird sie nicht verwendet.

Standardwert

`file:///C:/depot/Interact/dev/main/extremescale/config/catalogServer.props`

containerPropertyFile

Beschreibung

Der URI der Position der Eigenschaftendatei, die für den Start der WebSphere eXtreme Scale-Containerinstanzen verwendet wird. Wenn für den Start der WebSphere eXtreme Scale-Container-Server die enthaltene Serverkomponente verwendet wird, muss diese Eigenschaft festgelegt werden. Andernfalls wird sie nicht verwendet.

Standardwert

```
file:///C:/depot/Interact/dev/main/extremescale/config/
containerServer.props
```

deploymentPolicyFile

Beschreibung

Der URI der Position der Bereitstellungsrichtliniendatei, die für den Start des WebSphere eXtreme Scale-Katalogservers verwendet wird. Wenn für den Start des WebSphere eXtreme Scale-Katalogservers die enthaltene Serverkomponente verwendet wird, muss diese Eigenschaft festgelegt werden. Andernfalls wird sie nicht verwendet.

Standardwert

```
file:///C:/depot/Interact/dev/main/extremescale/config/
deployment.xml
```

objectGridConfigFile

Beschreibung

Der URI der Position der Konfigurationsdatei für das Objektgrid, die für den Start des WebSphere eXtreme Scale-Katalogservers und außerdem der cachenahen Komponente verwendet wird, die gemeinsam mit dem Interact-Laufzeitserver in derselben Java Virtual Machine (JVM) ausgeführt wird.

Standardwert

```
file:///C:/depot/Interact/dev/main/extremescale/config/
objectgrid.xml
```

gridName

Beschreibung

Der Name des WebSphere eXtreme Scale-Grids, das alle Interact-Caches enthält.

Standardwert

InteractGrid

catalogURLs

Beschreibung

Eine URL mit dem Hostnamen oder der IP-Adresse und dem Port, an dem der WebSphere eXtreme Scale-Katalogserver für Verbindungen empfangsbereit ist.

Standardwert

Keiner.

(Parameter)

Beschreibung

Eine Vorlage für die Erstellung eines angepassten Parameters, der in Verbindung mit dem Cache-Manager verwendet werden soll. Sie können jeden beliebigen Parameternamen und seinen gewünschten Wert konfigurieren.

Klicken Sie zur Erstellung eines angepassten Parameters auf ***(Parameter)*** und geben Sie den Namen und den Wert an, die diesem Parameter zugewiesen werden sollen. Wenn Sie auf **Änderungen speichern** klicken, wird der von Ihnen erstellte Parameter der Liste in der Kategorie "Parameterdaten" hinzugefügt.

Standardwert

Keiner.

Interact | caches

Mit dieser Gruppe von Konfigurationseigenschaften können Sie angeben, welchen der unterstützten Cache-Manager Sie verwenden möchten, um die Leistung von Interact zu verbessern (beispielsweise Ehcache oder das WebSphere eXtreme Scale-Caching). Außerdem können Sie bestimmte Cacheeigenschaften für den gerade konfigurierten Laufzeitserver konfigurieren.

Dies umfasst die Caches für das Speichern von Sitzungsdaten, Ereignismusterzuständen und Segmentierungsergebnissen. Durch die Anpassung dieser Einstellungen können Sie angeben, welche Cachelösung für die einzelnen Caching-Typen verwendet werden soll. Darüber hinaus können Sie einzelne Einstellungen angeben, um die Funktionsweise des Cache zu steuern.

Interact | cacheManagement | caches | InteractCache

Die Kategorie "InteractCache" konfiguriert das Caching für alle Sitzungsobjekte. Dazu gehören unter anderem die Profildaten, Segmentierungsergebnisse, die zuletzt bereitgestellten Verfahren, über API-Methoden übergebene Parameter sowie sonstige Objekte, die von der Interact-Laufzeit verwendet werden.

Die Kategorie "InteractCache" ist erforderlich, damit Interact ordnungsgemäß ausgeführt wird.

Bei Einstellungen, die in **Interact | cacheManagement | Caches** nicht unterstützt werden, kann die Kategorie "InteractCache" auch über eine externe Ehcache-Konfiguration konfiguriert werden. Wenn Sie Ehcache verwenden, müssen Sie sicherstellen, dass InteractCache ordnungsgemäß konfiguriert ist.

CacheManagerName

Beschreibung

Der Name des Cache-Managers, der den Interact-Cache verarbeitet. An dieser Stelle muss einer der Cache-Manager eingegeben werden, die in den Konfigurationseigenschaften **Interact | cacheManagement | Cache Managers** definiert sind (beispielsweise `EHCache` oder `Extreme Scale`).

Standardwert

Ehcache

Gültige Werte

Ein beliebiger Cache-Manager, der in der Konfigurationseigenschaft **Interact | cacheManagement | Cache Managers** definiert ist.

maxEntriesInCache

Beschreibung

Die maximale Anzahl der Sitzungsdatenobjekte, die in diesem Cache gespeichert werden können. Sobald die maximale Anzahl der Sitzungsdatenobjekte erreicht ist und Daten für eine weitere Sitzung gespeichert werden müssen, wird das Objekt gelöscht, dessen Verwendung am längsten zurückliegt.

Standardwert

100000

Gültige Werte

Ganzzahl größer 0.

timeoutInSecs

Beschreibung

Die Zeit in Sekunden, die seit der Verwendung oder Aktualisierung eines Sitzungsdatenobjekts verstrichen ist. Anhand dieses Werts wird bestimmt, wann das Objekt aus dem Cache entfernt wird.

Standardwert

300

Gültige Werte

Ganzzahl größer 0.

Interact | Caches | Interact Cache | Parameter Data

Die Konfigurationseigenschaften in dieser Kategorie steuern den Interact-Cache, der automatisch von Ihrer Interact-Installation verwendet wird. Diese Einstellungen müssen für jeden Interact-Laufzeitserver einzeln konfiguriert werden.

asyncIntervalMillis

Beschreibung

Die Zeit in Millisekunden, die der Cache-Manager Ehcache warten soll, bevor er Änderungen in anderen Interact-Laufzeitinstanzen repliziert. Wenn der Wert nicht positiv ist, werden diese Änderungen synchron repliziert.

Diese Konfigurationseigenschaft wird nicht standardmäßig erstellt. Wenn Sie diese Eigenschaft erstellen, wird sie nur verwendet, wenn Ehcache als Cache-Manager festgelegt und die Ehcache-Eigenschaft **cacheType** auf `distributed` gesetzt ist.

Standardwert

Keiner.

(Parameter)

Beschreibung

Eine Vorlage für die Erstellung eines angepassten Parameters, der in Verbindung mit dem Interact-Cache verwendet werden soll. Sie können jeden beliebigen Parameternamen und seinen gewünschten Wert konfigurieren.

Klicken Sie zur Erstellung eines angepassten Parameters auf ***(Parameter)*** und geben Sie den Namen und den Wert an, die diesem Parameter zugewiesen werden sollen. Wenn Sie auf **Änderungen speichern** klicken, wird der von Ihnen erstellte Parameter der Liste in der Kategorie "Parameterdaten" hinzugefügt.

Standardwert

Keiner.

Interact | cacheManagement | caches | PatternStateCache

Die Kategorie "PatternStateCache" hostet die Zustände der Ereignismuster sowie Regeln für die Echtzeitunterdrückung von Angeboten. Da dieser Cache standardmäßig als Lese- und Durchschreibcache konfiguriert ist, versucht Interact zuerst, das Ereignismuster und die Angebotsunterdrückungsdaten im Cache zu verwenden. Falls der angeforderte Eintrag nicht im Cache enthalten ist, lädt die Cacheimplementierung diesen Eintrag aus der Datenquelle. Hierfür wird entweder die JNDI-Konfiguration oder direkt eine JDBC-Verbindung verwendet.

Wenn eine JNDI-Verbindung verwendet wird, verbindet sich Interact mit einem vorhandenen Datenquellenanbieter, der unter Verwendung des JNDI-Namens, der URL usw. durch den angegebenen Server definiert wurde. Bei einer JDBC-Verbindung müssen Sie eine Gruppe von JDBC-Einstellungen angeben, die den Namen der JDBC-Treiberklasse, die Datenbank-URL und Authentifizierungsdaten enthalten.

Beachten Sie hierbei Folgendes: Wenn Sie mehrere JNDI- und JDBC-Quellen definieren, wird die erste aktivierte JNDI-Quelle verwendet. Falls keine JNDI-Quellen aktiviert sind, wird die erste aktivierte JDBC-Quelle verwendet.

Die Kategorie "PatternStateCache" ist erforderlich, damit Interact ordnungsgemäß ausgeführt wird.

Bei Einstellungen, die in **Interact | cacheManagement | Caches** nicht unterstützt werden, kann die Kategorie "PatternStateCache" auch über eine externe Ehcache-Konfiguration konfiguriert werden. Wenn Sie Ehcache verwenden, müssen Sie sicherstellen, dass PatternStateCache ordnungsgemäß konfiguriert ist.

CacheManagerName

Beschreibung

Der Name des Cache-Managers, der den Interact-Musterzustandscache verarbeitet. An dieser Stelle muss einer der Cache-Manager eingegeben werden, die in den Konfigurationseigenschaften **Interact | cacheManagement | Cache Managers** definiert sind (beispielsweise `EHCache` oder `Extreme Scale`).

Standardwert

Ehcache

Gültige Werte

Ein beliebiger Cache-Manager, der in der Konfigurationseigenschaft **Interact | cacheManagement | Cache Managers** definiert ist.

maxEntriesInCache

Beschreibung

Die maximale Anzahl von Ereignismusterzuständen, die in diesem Cache gespeichert werden können. Sobald die maximale Anzahl der Ereignismusterzustände erreicht ist und Daten für einen weiteren Ereignismusterzustand gespeichert werden müssen, wird das Objekt gelöscht, dessen Verwendung am längsten zurückliegt.

Standardwert

100000

Gültige Werte

Ganzzahl größer 0.

timeoutInSecs

Beschreibung

Gibt in Sekunden den Zeitraum an, nach dessen Ablauf ein Ereignismusterzustandsobjekt das Zeitlimit im Cache für Ereignismusterzustände überschritten hat. Wenn ein solches Zustandsobjekt im Cache für die über `timeoutInSecs` angegebene Anzahl an Sekunden inaktiv war, wird es auf Basis der LRU-Regel möglicherweise aus dem Cache entfernt. Beachten Sie, dass der Wert dieser Eigenschaft höher als der in der Eigenschaft `sessionTimeoutInSecs` definierte Wert sein sollte.

Standardwert

300

Gültige Werte

Ganzzahl größer 0.

Interact | Caches | PatternStateCache | Parameter Data:

Die Konfigurationseigenschaften in dieser Kategorie steuern den Musterzustandscache, in dem die Zustände der Ereignismuster sowie Regeln für die Echtzeitunterdrückung von Angeboten gehostet werden.

(Parameter)

Beschreibung

Eine Vorlage für die Erstellung eines angepassten Parameters, der in Verbindung mit dem Musterzustandscache verwendet werden soll. Sie können jeden beliebigen Parameternamen und seinen gewünschten Wert konfigurieren.

Klicken Sie zur Erstellung eines angepassten Parameters auf ***(Parameter)*** und geben Sie den Namen und den Wert an, die diesem Parameter zugewiesen werden sollen. Wenn Sie auf **Änderungen speichern** klicken, wird der von Ihnen erstellte Parameter der Liste in der Kategorie "Parameterdaten" hinzugefügt.

Standardwert

Keiner.

Interact | cacheManagement | caches | PatternStateCache | loaderWriter:

Die Kategorie **loaderWriter** enthält die Konfiguration des Ladeprogramms, das mit externen Repositorys für den Abruf und die Persistenz von Ereignismustern interagiert.

className

Beschreibung

Der vollständig qualifizierte Klassenname für dieses Ladeprogramm. Diese Klasse muss die Anforderung des gewählten Cache-Managers erfüllen.

Standardwert

`com.unicacorp.interact.cache.ehcache.loaderwriter.PatternStateEHCacheLoaderWriter`

Gültige Werte

Ein vollständig qualifizierter Klassenname.

classPath

Beschreibung

Der Pfad der Klassendatei des Ladeprogramms. Wenn Sie diesen Wert leer lassen oder die Eingabe ungültig ist, wird der Klassenpfad verwendet, der für die Ausführung von Interact genutzt wird.

Standardwert

Keiner.

Gültige Werte

Ein gültiger Klassenpfad.

writeMode

Beschreibung

Gibt den Modus für das Ausgabeprogramm an, mit dem die Persistenz der neuen oder aktualisierten Ereignismusterzustände im Cache festgelegt wird. Folgende Optionen sind gültig:

- WRITE_THROUGH. Jedes Mal, wenn ein neuer Eintrag hinzugefügt oder ein bestehender Eintrag aktualisiert wird, wird der betreffende Eintrag sofort in die Repositorys geschrieben.
- WRITE_BEHIND. Der Cache-Manager wartet eine gewisse Zeit, bis er mehrere Änderungen erfasst hat, und definiert diese anschließend im Stapelbetrieb als persistente Änderungen in den Repositorys.

Standardwert

WRITE_THROUGH

Gültige Werte

WRITE_THROUGH oder WRITE_BEHIND.

batchSize

Beschreibung

Die maximale Anzahl von Ereignismusterzustandsobjekten, die das Ausgabeprogramm im Stapelbetrieb als persistent definiert. Diese Eigenschaft wird nur verwendet, wenn **writeMode** auf `WRITE_BEHIND` gesetzt wurde.

Standardwert

100

Gültige Werte

Ganzzahlwert.

maxDelayInSecs

Beschreibung

Die maximale Zeit in Sekunden, die der Cache-Manager wartet, bevor er ein Ereignismusterzustandsobjekt als persistent definiert. Diese Eigenschaft wird nur verwendet, wenn **writeMode** auf `WRITE_BEHIND` gesetzt wurde.

Standardwert

5

Gültige Werte

Ganzzahlwert.

Interact | Caches | PatternStateCache | loaderWriter | Parameter Data:

Die Konfigurationseigenschaften in dieser Kategorie steuern das Ladeprogramm des Musterzustandscache.

(Parameter)

Beschreibung

Eine Vorlage für die Erstellung eines angepassten Parameters, der in Verbindung mit dem Ladeprogramm des Musterzustandscache verwendet werden soll. Sie können jeden beliebigen Parameternamen und seinen gewünschten Wert konfigurieren.

Klicken Sie zur Erstellung eines angepassten Parameters auf ***(Parameter)*** und geben Sie den Namen und den Wert an, die diesem Parameter zugewiesen werden sollen. Wenn Sie auf **Änderungen speichern** klicken, wird der von Ihnen erstellte Parameter der Liste in der Kategorie "Parameterdaten" hinzugefügt.

Standardwert

Keiner.

Interact | cacheManagement | caches | PatternStateCache | loaderWriter | jndiSettings:

Die Kategorie **jndiSettings** enthält die Konfiguration für die JNDI-Datenquelle, die vom Ladeprogramm für die Kommunikation mit der zugrunde liegenden Datenbank verwendet wird. Wenn Sie eine neue Gruppe von JNDI-Einstellungen erstellen möchten, erweitern Sie die Kategorie **jdniSettings** und klicken Sie auf die Eigenschaft ***(jndiSetting)***.

(jndiSettings)

Anmerkung: Wenn WebSphere Application Server verwendet wird, wird "loaderWriter" nicht mit **jndiSettings** verbunden.

Beschreibung

Wenn Sie auf diese Kategorie klicken, wird ein Formular angezeigt. Geben Sie die folgenden Werte an, um eine JNDI-Datenquelle zu definieren:

- **Neuer Kategoriename** ist der Name, den Sie für die Identifizierung dieser JNDI-Verbindung verwenden möchten.

- Mit **enabled** können Sie angeben, ob diese JNDI-Verbindung für die Verwendung verfügbar sein soll oder nicht. Setzen Sie diese Eigenschaft bei neuen Verbindungen auf `True`.
- **jdniName** ist der JNDI-Name, der bei der Einrichtung der Datenquelle bereits in ihr definiert wurde.
- **providerUrl** ist die URL, über die diese JNDI-Datenquelle gefunden werden kann. Wenn Sie dieses Feld leer lassen, wird die URL der Webanwendung verwendet, welche die Interact-Laufzeit hostet.
- **Ausgangskontextfactory** ist der vollständig qualifizierte Klassenname der Ausgangskontextfactory-Klasse für die Verbindung mit dem JNDI-Anbieter. Wenn die Webanwendung, welche die Interact-Laufzeit hostet, als **providerUrl** verwendet wird, lassen Sie dieses Feld leer.

Standardwert

Keiner.

Interact | cacheManagement | caches | PatternStateCache | loaderWriter | jdbcSettings:

Die Kategorie **jdbcSettings** enthält die Konfiguration für die JDBC-Verbindungen, die vom Ladeprogramm für die Kommunikation mit der zugrunde liegenden Datenbank verwendet werden. Wenn Sie eine neue Gruppe von JDBC-Einstellungen erstellen möchten, erweitern Sie die Kategorie **jdbcSettings** und klicken Sie auf die Eigenschaft ***(jdbcSetting)***.

(jdbcSettings)

Beschreibung

Wenn Sie auf diese Kategorie klicken, wird ein Formular angezeigt. Geben Sie die folgenden Werte an, um eine JDBC-Datenquelle zu definieren:

- **Neuer Kategoriename** ist der Name, den Sie für die Identifizierung dieser JDBC-Verbindung verwenden möchten.
- Mit **enabled** können Sie angeben, ob diese JDBC-Verbindung für die Verwendung verfügbar sein soll oder nicht. Setzen Sie diese Eigenschaft bei neuen Verbindungen auf `True`.
- **driverClassName** ist der vollständig qualifizierte Klassenname des JDBC-Treibers. Diese Klasse muss in dem Klassenpfad enthalten sein, der für den Start des hostenden Cache-Servers konfiguriert wurde.
- **databaseUrl** ist die URL, über die diese JDBC-Datenquelle gefunden werden kann.
- **asmUser** ist der Name des IBM EMM-Benutzers, der mit den Berechtigungsnachweisen für die Herstellung einer Verbindung zu der Datenbank in dieser JDBC-Verbindung konfiguriert wurde.
- **asmDataSource** ist der Name der IBM EMM-Datenquelle, die mit den Berechtigungsnachweisen für die Herstellung einer Verbindung zu der Datenbank in dieser JDBC-Verbindung konfiguriert wurde.
- **maxConnection** ist die maximale Anzahl gleichzeitig bestehender Verbindungen, die zu der Datenbank in dieser JDBC-Verbindung hergestellt werden können.

Standardwert

Keiner.

Interact | triggeredMessage

Die Konfigurationseigenschaften in dieser Kategorie definieren Einstellungen für alle ausgelösten Nachrichten und Angebotsbereitstellungen in Kanälen.

backendProcessIntervalMin

Beschreibung

Diese Eigenschaft definiert den Zeitraum in Minuten, in dem der Back-End-Thread verzögerte Angebotsbereitstellungen lädt und verarbeitet. Dieser Wert muss eine Ganzzahl sein. Wenn der Wert null oder negativ ist, wird der Back-End-Prozess inaktiviert.

Gültige Werte

Eine positive Ganzzahl

autoLogContactAfterDelivery

Beschreibung

Wenn diese Eigenschaft auf True festgelegt ist, wird automatisch ein Kontaktereignis übergeben, sobald dieses Angebot gesendet wurde oder für eine verzögerte Bereitstellung in die Warteschlange gestellt wurde. Wenn diese Eigenschaft auf False festgelegt ist, wird für die abgehenden Angebote nicht automatisch ein Kontaktereignis übergeben. Dies ist das Standardverhalten.

Gültige Werte

True | False

waitForFlowchart

Beschreibung

Diese Eigenschaft bestimmt, ob das Ablaufdiagramm warten sollte, bis die derzeit ausgeführte Segmentierung beendet wird, sowie das Verhalten, wenn das Zeitlimit für diese Wartezeit überschritten wird.

DoNotWait: Die Verarbeitung einer ausgelösten Nachricht beginnt unabhängig davon, ob die Segmentierung derzeit ausgeführt wird oder nicht. Wenn jedoch Segmente in der Bedingungsregel verwendet werden und/oder NextBestOffer als Methode zur Angebotsauswahl ausgewählt wurde, wird mit der TM-Ausführung noch gewartet.

OptionalWait: Mit der Verarbeitung einer ausgelösten Nachricht wird gewartet, bis die Segmentierung, die derzeit ausgeführt wird, abgeschlossen wird oder das zulässige Zeitlimit überschritten wird. Wenn die Wartezeit das zulässige Zeitlimit überschreitet, wird eine Warnung protokolliert und die Verarbeitung dieser ausgelösten Nachricht fortgesetzt. Dies ist der Standard.

MandatoryWait: Mit der Verarbeitung einer ausgelösten Nachricht wird gewartet, bis die Segmentierung, die derzeit ausgeführt wird, abgeschlossen wird oder das zulässige Zeitlimit überschritten wird. Wenn die Wartezeit das zulässige Zeitlimit überschreitet, wird ein Fehler protokolliert und die Verarbeitung dieser ausgelösten Nachricht abgebrochen.

Gültige Werte

DoNotWait | OptionalWait | MandatoryWait

Interact | triggeredMessage | offerSelection

Die Konfigurationseigenschaften in dieser Kategorie definieren Einstellungen für die Angebotsauswahl in ausgelösten Nachrichten.

maxCandidateOffers

Beschreibung

Diese Eigenschaft definiert die maximale Anzahl der infrage kommenden Angebote, die von der Engine zurückgegeben werden, um das beste Angebot für die Bereitstellung zu bekommen. Es besteht die Möglichkeit, dass keines dieser zurückgegebenen infrage kommenden Angebote basierend auf dem ausgewählten Kanal gesendet werden kann. Je mehr mögliche Angebote es gibt, desto geringer ist die Wahrscheinlichkeit, dass dies geschieht. Durch mehr mögliche Angebote kann die Verarbeitungszeit jedoch erhöht werden.

Gültige Werte

Eine positive Ganzzahl

defaultCellCode

Beschreibung

Wenn das bereitgestellte Angebot das Ergebnis der Auswertung einer strategischen Regel oder eines tabellengesteuerten Datensatzes ist, ist dem Angebot eine Zielzelle zugeordnet und die Informationen aus dieser Zelle werden bei jeder entsprechenden Protokollierung verwendet. Wenn jedoch eine Liste mit bestimmten Angeboten als Eingabe bei der Angebotsauswahl verwendet wird, ist keine Zielzelle verfügbar. In diesem Fall wird der Wert dieser Konfigurationseinstellung verwendet. Sie müssen sicherstellen, dass diese Zielzelle und die zugehörige Kampagne bei der Bereitstellung eingeschlossen werden. Dies geschieht am einfachsten, indem Sie die Zelle zu einer bereitgestellten Strategie hinzufügen.

Interact | triggeredMessage | dispatchers

Die Konfigurationseigenschaften in dieser Kategorie definieren Einstellungen für alle Dispatcher in ausgelösten Nachrichten.

dispatchingThreads

Beschreibung

Diese Eigenschaft definiert die Anzahl der Threads der Engine, die verwendet werden, um den Dispatcher asynchron aufzurufen. Wenn der Wert 0 oder eine negative Zahl ist, erfolgt das Aufrufen von Dispatchern synchron. Der Standardwert beträgt 0.

Gültige Werte

Eine Ganzzahl

Interact | triggeredMessage | dispatchers | <dispatcherName>

Die Konfigurationseigenschaften in dieser Kategorie definieren Einstellungen für einen bestimmten Dispatcher in ausgelösten Nachrichten.

Kategoriename

Beschreibung

Diese Eigenschaft definiert den Namen dieses Dispatchers. Der Name muss für jeden Dispatcher eindeutig sein.

Typ

Beschreibung

Diese Eigenschaft definiert den Dispatchertyp.

Gültige Werte

InMemoryQueue | JMSQueue | Custom

Anmerkung: Wenn Sie für die Integration von Interact in IBM MQ JMS-Queue oder Custom verwenden, muss die Interact-Laufzeit auf dem App-Server mit JDK 1.7 sein. Für WebSphere und WebLogic wird empfohlen, die aktuell bereitgestellte Version des JDK-Fixpacks zu verwenden.

JMSQueue unterstützt nur WebLogic. Sie können JMSQueue nicht verwenden, wenn Sie WebSphere Application Server verwenden.

className

Beschreibung

Diese Eigenschaft definiert den vollständig qualifizierten Klassennamen der Implementierung dieses Dispatchers. Wenn der Typ "InMemoryQueue" lautet, sollte der Wert leer sein. Wenn der Typ "Custom" lautet, muss diese Eigenschaft den Wert `com.unicacorp.interact.eventhandler.triggeredmessage.dispatchers.IBMMQDispatcher` aufweisen.

classPath

Beschreibung

Diese Eigenschaft definiert die URL der JAR-Datei, die die Implementierung dieses Dispatchers umfasst.

Wenn der Typ "Custom" lautet, muss diese Eigenschaft den Wert `file://<Interact_HOME>/lib/interact_ibmmqdispatcher.jar;file://<Interact_HOME>/lib/com.ibm.mq.allclient.jar;file://<Interact_HOME>/lib/jms.jar` aufweisen

Interact | triggeredMessage | dispatchers | <dispatcherName> | Parameter Data

Die Konfigurationseigenschaften in dieser Kategorie definieren Parameter für einen bestimmten Dispatcher in ausgelösten Nachrichten.

Sie können zwischen drei Typen von Dispatchern auswählen. InMemoryQueue ist der interne Dispatcher für Interact. Custom wird für IBM MQ verwendet. JMS-Queue wird verwendet, um eine Verbindung über JNDI zu einem JMS-Provider herzustellen.

Kategoriename

Beschreibung

Diese Eigenschaft definiert den Namen dieses Parameters. Der Name muss für jeden Parameter für diesen Dispatcher eindeutig sein.

Wert

Beschreibung

Diese Eigenschaft definiert die Parameter, im Format von Name/Wert-Paaren, die von diesem Dispatcher benötigt werden.

Anmerkung: Bei allen Parametern für Auslösenachrichten muss die Groß-/Kleinschreibung beachtet werden und sie sollten wie im Folgenden dargestellt eingegeben werden.

Für den Typ InMemoryQueue wird der folgende Parameter unterstützt.

- queueCapacity: Optional. Die maximale Anzahl der Angebote, die in der Warteschlange warten können, bis sie gesendet werden. Ist diese Eigenschaft angegeben, muss es eine positive Ganzzahl sein. Ist die Eigenschaft nicht angegeben oder ungültig, wird der Standardwert (1000) verwendet.

Für den Typ Custom werden die folgenden Parameter unterstützt.

- providerUrl: <Hostname>:port (Groß-/Kleinschreibung muss beachtet werden)
- queueManager: Der Name des Warteschlangenmanagers, der auf dem IBM MQ-Server erstellt wurde.
- messageQueueName: Der Name der Nachrichtenwarteschlange, die auf dem IBM MQ-Server erstellt wurde.
- enableConsumer: Diese Eigenschaft muss auf True festgelegt sein.
- asmUserforMQAuth: Der Benutzername für die Anmeldung auf dem Server. Dieser ist erforderlich, wenn der Server eine Authentifizierung erzwingt. Andernfalls sollte er nicht angegeben werden.
- authDS: Das Kennwort, das dem Benutzernamen für die Anmeldung auf dem Server zugeordnet ist. Dieses ist erforderlich, wenn der Server eine Authentifizierung erzwingt. Andernfalls sollte es nicht angegeben werden.

Für den Typ JMSQueue wird der folgende Parameter unterstützt.

- providerUrl: Die URL für den JNDI-Provider (Groß-/Kleinschreibung muss beachtet werden).
- connectionFactoryJNDI: Der JNDI-Name der JMS-Verbindungsfactory.
- messageQueueJNDI: Der JNDI-Name der JMS-Warteschlange, an die ausgelöste Nachrichten gesendet und von der ausgelöste Nachrichten abgerufen werden.
- enableConsumer: Gibt an, ob ein Kunde dieser ausgelösten Nachrichten in Interact gestartet werden sollte. Diese Eigenschaft muss auf True festgelegt sein. Ist die Eigenschaft nicht angegeben, wird der Standardwert (False) verwendet.
- initialContextFactory: Der vollständig qualifizierte Name der JNDI-Ausgangskontextfactoryklasse. Wenn Sie WebLogic verwenden, sollte der Wert dieses Parameters `weblogic.jndi.WLInitialContextFactory` lauten.

Interact | triggeredMessage | gateways | <gatewayName>

Die Konfigurationseigenschaften in dieser Kategorie definieren Einstellungen für ein bestimmtes Gateway in ausgelösten Nachrichten.

Interact bietet keine Unterstützung für mehrere Instanzen desselben Gateways. Alle Gateway-Konfigurationsdateien sollten von jedem Interact-Laufzeitknoten aus zu-

gänglich sein. Bei einer verteilten Konfiguration müssen Sie sicherstellen, dass sich die Gateway-Dateien an einer gemeinsam genutzten Position befinden.

Kategoriename

Beschreibung

Diese Eigenschaft definiert den Namen dieses Gateways. Er muss für jedes Gateway eindeutig sein.

className

Beschreibung

Diese Eigenschaft definiert den vollständig qualifizierten Klassennamen der Implementierung dieses Gateways.

classPath

Beschreibung

Diese Eigenschaft definiert den URI der JAR-Datei, die die Implementierung dieses Gateways umfasst. Wird diese Eigenschaft leer gelassen, wird der Klassenpfad der hostenden Interact-Anwendung verwendet.

Wenn zum Beispiel in einem Windows-System die Gateway-JAR-Datei im Verzeichnis `C:\IBM\EMM\EmailGateway\IBM_Interact_OMO_OutboundGateway_Silverpop_1.0\lib\OMO_OutboundGateway_Silverpop.jar` verfügbar ist, sollte classPath auf `file:///C:/IBM/EMM/EmailGateway/IBM_Interact_OMO_OutboundGateway_Silverpop_1.0/lib/OMO_OutboundGateway_Silverpop.jar` festgelegt werden. Wenn in einem UNIX-System die Gateway-JAR-Datei im Verzeichnis `/opt/IBM/EMM/EmailGateway/IBM_Interact_OMO_OutboundGateway_Silverpop_1.0/lib/OMO_OutboundGateway_Silverpop.jar` verfügbar ist, sollte classPath auf `file:///opt/IBM/EMM/EmailGateway/IBM_Interact_OMO_OutboundGateway_Silverpop_1.0/lib/OMO_OutboundGateway_Silverpop.jar` festgelegt werden.

Interact | triggeredMessage | gateways | <gatewayName> | Parameter Data

Die Konfigurationseigenschaften in dieser Kategorie definieren Parameter für ein bestimmtes Gateway in ausgelösten Nachrichten.

Kategoriename

Beschreibung

Diese Eigenschaft definiert den Namen dieses Parameters. Der Name muss für jeden Parameter für dieses Gateway eindeutig sein.

Wert

Beschreibung

Diese Eigenschaft definiert die Parameter, im Format von Name/Wert-Paaren, die von diesem Gateway benötigt werden. Die folgenden Parameter werden für alle Gateways unterstützt.

Anmerkung: Bei allen Parametern für Auslösenachrichten muss die Groß-/Kleinschreibung beachtet werden und sie sollten wie im Folgenden dargestellt eingegeben werden.

- validationTimeoutMillis: Die Dauer in Millisekunden, in der das Zeitlimit der Validierung eines Angebots über dieses Gateway überschritten wird. Der Standardwert beträgt 500.
- deliveryTimeoutMillis: Die Dauer in Millisekunden, in der das Zeitlimit der Bereitstellung eines Angebots über dieses Gateway überschritten wird. Der Standardwert beträgt 1000.

Interact | triggeredMessage | channels

Die Konfigurationseigenschaften in dieser Kategorie definieren Einstellungen für alle Kanäle in ausgelösten Nachrichten.

Typ

Beschreibung

Diese Eigenschaft definiert den Stammknoten für Einstellungen zu einem bestimmten Gateway. Bei "Default" wird die integrierte Kanalauswahl verwendet, die auf der Liste der Kanäle basiert, die in der Benutzerschnittstelle der ausgelösten Nachrichten definiert wurden. Ist "Default" ausgewählt, sollten die Werte className und classPath leer gelassen werden. Bei "Custom" wird die benutzerdefinierte Implementierung von IChannelSelector verwendet.

Gültige Werte

Default | Custom

className

Beschreibung

Diese Eigenschaft definiert den vollständig qualifizierten Klassennamen der Kundenimplementierung der Kanalauswahl. Diese Einstellung ist bei dem Typ "Custom" erforderlich.

classPath

Beschreibung

Diese Eigenschaft definiert die URL der JAR-Datei, die die Implementierung der Kundenimplementierung der Kanalauswahl umfasst. Wird diese Eigenschaft leer gelassen, wird der Klassenpfad der hostenden Interact-Anwendung verwendet.

Interact | triggeredMessage | channels | Parameter Data

Die Konfigurationseigenschaften in dieser Kategorie definieren Parameter für einen bestimmten Kanal in ausgelösten Nachrichten.

Kategoriename

Beschreibung

Diese Eigenschaft definiert den Namen dieses Parameters. Der Name muss für jeden Parameter für diesen Kanal eindeutig sein.

Wert

Beschreibung

Diese Eigenschaft definiert die Parameter, im Format von Name/Wert-Paaren, die für diese Kanalauswahl benötigt werden.

Wenn Sie **Vom Kunden bevorzugte Kanäle** für Ihren Kanal verwenden, müssen Sie erstellen

Interact | triggeredMessage | channels | <channelName>

Die Konfigurationseigenschaften in dieser Kategorie definieren Einstellungen für einen bestimmten Kanal in ausgelösten Nachrichten.

Kategoriename

Beschreibung

Diese Eigenschaft definiert den Namen des Kanals, über den Angebote gesendet werden. Sie sollte mit den in der Designzeit unter **Campaign | partitions | <partition[N]> | Interact | outboundChannels** definierten Namen übereinstimmen.

Interact | triggeredMessage | channels | <channelName> | <handlerName>

Die Konfigurationseigenschaften in dieser Kategorie definieren Einstellungen für einen bestimmten Handler in ausgelösten Nachrichten, der zum Senden von Angeboten verwendet wird.

Kategoriename

Beschreibung

Diese Eigenschaft definiert den Namen des Handlers, der im Kanal zum Senden von Angeboten verwendet wird.

Dispatcher

Beschreibung

Diese Eigenschaft definiert den Namen des Dispatchers, über den dieser Handler Angebote an das Gateway sendet. Es muss einer der unter **interact | triggeredMessage | dispatchers** definierten Dispatcher sein.

Gateway

Beschreibung

Diese Eigenschaft definiert den Namen des Gateways, an das dieser Handler schließlich Angebote sendet. Es muss eines der unter **interact | triggeredMessage | gateways** definierten Gateways sein.

Modus

Beschreibung

Diese Eigenschaft definiert den Verwendungsmodus dieses Handlers. Wenn "Failover" ausgewählt wird, wird dieser Handler nur verwendet, wenn keiner der Handler mit höheren Prioritäten, die in diesem Kanal definiert sind, ein Angebot senden konnte. Wenn "Addon" ausgewählt wird, wird dieser Handler unabhängig davon verwendet, ob andere Handler erfolgreich Angebote gesendet haben.

Priorität

Beschreibung

Diese Eigenschaft definiert die Priorität dieses Handlers. Die Engine versucht zunächst, den Handler mit der höchsten Priorität zum Senden der Angebote zu verwenden.

Gültige Werte

Beliebige Ganzzahl

Standard

100

Interact | ETL | patternStateETL

Die Konfigurationseigenschaften in dieser Kategorie definieren die Einstellungen für den ETL-Prozess.

Neuer Kategoriename

Beschreibung

Geben Sie einen Namen an, der diese Konfiguration eindeutig identifiziert. Beachten Sie, dass Sie bei der Ausführung des eigenständigen ETL-Prozesses genau diesen Namen angeben müssen. Aus praktischen Gründen wird empfohlen, keinen Namen mit Leerzeichen oder Interpunktionen zu wählen, da dieser Name in der Befehlszeile angegeben werden muss. Beispiel: `ETLProfile1`.

runOnceADay

Beschreibung

Legt fest, ob der eigenständige ETL-Prozess in dieser Konfiguration einmal täglich ausgeführt werden soll. Gültige Antworten sind **Yes** (Ja) oder **No** (Nein). Wenn Sie hier die Antwort **No** eingeben, bestimmt **processSleepIntervalInMinutes** den Ausführungszeitplan für den Prozess.

preferredStartTime

Beschreibung

Die bevorzugte Uhrzeit, zu der der eigenständige ETL-Prozess starten soll. Geben Sie die Uhrzeit im Format HH:MM:SS AM/PM an. Beispiel: `01:00:00 AM`.

preferredEndTime

Beschreibung

Die bevorzugte Uhrzeit, zu der der eigenständige ETL-Prozess stoppen soll. Geben Sie die Uhrzeit im Format HH:MM:SS AM/PM an. Beispiel: `08:00:00 AM`.

processSleepIntervalInMinutes

Beschreibung

Wenn Sie in der Konfiguration des eigenständigen ETL-Prozesses nicht festgelegt haben, dass dieser einmal täglich ausgeführt werden soll (Angabe der Eigenschaft **runOnceADay**), gibt diese Eigenschaft das Intervall zwischen den Ausführungen des ETL-Prozesses an. Wenn Sie an dieser

Stelle beispielsweise 15 angeben, wartet der eigenständige ETL-Prozess 15 Minuten nach der Beendigung seiner Ausführung, bis der Prozess wieder gestartet wird.

maxJDBCInsertBatchSize

Beschreibung

Die maximale Anzahl der Datensätze eines JDBC-Batches vor dem Ausführen der Abfrage. Standardmäßig ist diese Eigenschaft auf 5000 gesetzt. Beachten Sie, dass dies nicht die maximale Anzahl der Datensätze ist, die vom ETL-Prozess in einer Iteration verarbeitet werden. Während jeder Iteration verarbeitet der ETL-Prozess alle verfügbaren Datensätze aus der Tabelle UACI_EVENTPATTERNSTATE. Diese Datensätze werden jedoch in **maxJDBCInsertSize**-Datenblöcke unterteilt.

maxJDBCFetchBatchSize

Beschreibung

Die maximale Anzahl der Datensätze eines von der Staging-Datenbank abzurufenden JDBC-Batches.

Sie müssen diesen Wert möglicherweise erhöhen, um die Leistung des ETL-Prozesses zu optimieren.

communicationPort

Beschreibung

Der Netzport, an dem der eigenständige ETL-Prozess für eine Stoppanforderung empfangsbereit ist. Unter normalen Umständen muss dieser Standardwert nicht geändert werden.

queueLength

Beschreibung

Ein Wert, der für die Leistungsoptimierung verwendet wird. Die Erfassungen der Musterzustandsdaten werden abgerufen und in Objekte transformiert, die einer Warteschlange hinzugefügt werden, damit sie verarbeitet und in die Datenbank geschrieben werden können. Diese Eigenschaft steuert die Größe der Warteschlange.

completionNotificationScript

Beschreibung

Gibt den absoluten Pfad zu einem Script an, das ausgeführt wird, wenn der ETL-Prozess abgeschlossen ist. Wenn Sie ein Script angeben, werden drei Argumente an das Abschlussbenachrichtigungsscript übergeben: Startzeit, Endzeit und Gesamtzahl der verarbeiteten Ereignismusterdatensätze. Die Start- und Endzeit sind numerische Werte, die die Anzahl der seit 1970 vergangenen Millisekunden darstellen.

Interact | ETL | patternStateETL | <patternStateETLName> | RuntimeDS

Die Konfigurationseigenschaften in dieser Kategorie definieren die Einstellungen für die Laufzeitdatenquelle des ETL-Prozesses.

type

Beschreibung

Eine Liste der von Ihnen definierten unterstützten Datenbanktypen für die Datenquelle.

dsname

Beschreibung

Der JNDI-Name der Datenquelle. Dieser Name muss auch in der Datenquellenkonfiguration des Benutzers verwendet werden, um sicherzustellen, dass der Benutzer auf die Ziel- und Laufzeitdatenquellen zugreifen kann.

driver

Beschreibung

Der Name des zu verwendenden JDBC-Treibers. Beispiele:

Oracle: `oracle.jdbc.OracleDriver`

Microsoft SQL Server: `com.microsoft.sqlserver.jdbc.SQLServerDriver`

IBM DB2: `com.ibm.db2.jcc.DB2Driver`

serverURL

Beschreibung

Die URL der Datenquelle. Beispiele:

Oracle: `jdbc:oracle:thin:@<Host_Ihrer_Datenbank>:<Port_Ihrer_Datenbank>:<Name_Ihres_Datenbankservice>`

Microsoft SQL Server: `jdbc:sqlserver://<Host_Ihrer_Datenbank>:<Port_Ihrer_Datenbank>;databaseName=<Name_Ihrer_Datenbank>`

IBM DB2: `jdbc:db2://<Host_Ihrer_Datenbank>:<Port_Ihrer_Datenbank>/<Name_Ihrer_Datenbank>`

connectionpoolSize

Beschreibung

Ein Wert, der die Größe des Verbindungspools angibt. Er wird zur Leistungsoptimierung bereitgestellt. Die Musterzustandsdaten werden gelesen und gleichzeitig je nach den verfügbaren Datenbankverbindungen transformiert. Durch eine Erhöhung der Verbindungspoolgröße können mehr gleichzeitige Datenbankverbindungen verwendet werden. Dies hängt jedoch auch von Einschränkungen des Speichers und der Lese-/Schreibfunktionalität für Datenbanken ab. Wenn Sie diesen Wert beispielsweise auf 4 setzen, werden vier Jobs gleichzeitig ausgeführt. Falls Sie ein umfangreiches Datenvolumen haben, müssen Sie diesen Wert unter Umständen auf eine Zahl wie 10 oder 20 erhöhen, sofern der verfügbare Speicher und die Datenbankleistung ausreichen.

Schema

Beschreibung

Der Name des Datenbankschemas, mit dem sich diese Konfiguration verbindet.

connectionRetryPeriod

Beschreibung

Die Eigenschaft `ConnectionRetryPeriod` gibt in Sekunden an, wie lange Interact eine fehlgeschlagene Datenbankverbindungsaufforderung automatisch wiederholt. Interact versucht in diesem Zeitraum automatisch, die Verbindung zur Datenbank wiederherzustellen, bevor ein Datenbankfehler gemeldet wird. Wird der Wert auf 0 gesetzt, wiederholt Interact den Verbindungsversuch unbegrenzt. Wenn -1 festgelegt ist, wird kein wiederholter Versuch unternommen.

connectionRetryDelay

Beschreibung

Die Eigenschaft `ConnectionRetryDelay` gibt in Sekunden an, wie lange Interact wartet, bevor versucht wird, die Verbindung zur Datenbank wiederherzustellen, wenn ein Fehler aufgetreten ist. Wenn der Wert auf -1 gesetzt ist, wird nicht versucht, die Verbindung wiederherzustellen.

Interact | ETL | patternStateETL | <patternStateETLName> | TargetDS

Die Konfigurationseigenschaften in dieser Kategorie definieren die Einstellungen für die Zieldatenquelle des ETL-Prozesses.

type

Beschreibung

Eine Liste der von Ihnen definierten unterstützten Datenbanktypen für die Datenquelle.

dsname

Beschreibung

Der JNDI-Name der Datenquelle. Dieser Name muss auch in der Datenquellenkonfiguration des Benutzers verwendet werden, um sicherzustellen, dass der Benutzer auf die Ziel- und Laufzeitdatenquellen zugreifen kann.

driver

Beschreibung

Der Name des zu verwendenden JDBC-Treibers. Beispiele:

Oracle: `oracle.jdbc.OracleDriver`

Microsoft SQL Server: `com.microsoft.sqlserver.jdbc.SQLServerDriver`

IBM DB2: `com.ibm.db2.jcc.DB2Driver`

serverURL

Beschreibung

Die URL der Datenquelle. Beispiele:

Oracle: `jdbc:oracle:thin:@ <Host_Ihrer_Datenbank>:<Port_Ihrer_Datenbank>:<Name_Ihres_ Datenbankservice>`

Microsoft SQL Server: `jdbc:sqlserver:// <Host_Ihrer_Datenbank>:<Port_Ihrer_Datenbank>;databaseName= <Name_Ihrer_Datenbank>`

IBM DB2: `jdbc:db2:// <Host_Ihrer_Datenbank>:<Port_Ihrer_ Datenbank>/<Name_Ihrer_Datenbank>`

connectionpoolSize

Beschreibung

Ein Wert, der die Größe des Verbindungspools angibt. Er wird zur Leistungsoptimierung bereitgestellt. Die Musterzustandsdaten werden gelesen und gleichzeitig je nach den verfügbaren Datenbankverbindungen transformiert. Durch eine Erhöhung der Verbindungspoolgröße können mehr gleichzeitige Datenbankverbindungen verwendet werden. Dies hängt jedoch auch von Einschränkungen des Speichers und der Lese-/ Schreibfunktionalität für Datenbanken ab. Wenn Sie diesen Wert beispielsweise auf 4 setzen, werden vier Jobs gleichzeitig ausgeführt. Falls Sie ein umfangreiches Datenvolumen haben, müssen Sie diesen Wert unter Umständen auf eine Zahl wie 10 oder 20 erhöhen, sofern der verfügbare Speicher und die Datenbankleistung ausreichen.

Schema

Beschreibung

Der Name des Datenbankschemas, mit dem sich diese Konfiguration verbindet.

connectionRetryPeriod

Beschreibung

Die Eigenschaft `ConnectionRetryPeriod` gibt in Sekunden an, wie lange Interact eine fehlgeschlagene Datenbankverbindungsaufforderung automatisch wiederholt. Interact versucht in diesem Zeitraum automatisch, die Verbindung zur Datenbank wiederherzustellen, bevor ein Datenbankfehler gemeldet wird. Wird der Wert auf 0 gesetzt, wiederholt Interact den Verbindungsversuch unbegrenzt. Wenn -1 festgelegt ist, wird kein wiederholter Versuch unternommen.

connectionRetryDelay

Beschreibung

Die Eigenschaft `ConnectionRetryDelay` gibt in Sekunden an, wie lange Interact wartet, bevor versucht wird, die Verbindung zur Datenbank wiederherzustellen, wenn ein Fehler aufgetreten ist. Wenn der Wert auf -1 gesetzt ist, wird nicht versucht, die Verbindung wiederherzustellen.

Interact | ETL | patternStateETL | <patternStateETLName> | Report

Die Konfigurationseigenschaften in dieser Kategorie definieren die Einstellungen für den Aggregationsprozessbericht des ETL-Prozesses.

enable

Beschreibung

Sie können die Berichtsintegration in den ETL-Prozess aktivieren oder inaktivieren. Diese Eigenschaft ist standardmäßig inaktiviert.

Wenn sie auf `disable` (Inaktivieren) gesetzt ist, inaktiviert diese Eigenschaft Aktualisierungen in der Tabelle UARI_DELTA_PATTERNS. Sie inaktiviert jedoch nicht die gesamte Berichterstellung.

Anmerkung: Wenn Sie die Berichtsintegration in den ETL-Prozess inaktivieren möchten, müssen Sie auch den Auslöser TR_AGGREGATE_DELTA_PATTERNS ändern, um die UACI_ETLPATTERNSTATERUN-Staging-Tabelle zu inaktivieren.

retryAttemptsIfAggregationRunning

Beschreibung

Gibt an, wie oft der ETL-Prozess versucht, zu prüfen, ob die Berichtsaggregation abgeschlossen ist, wenn das Flag "lock" festgelegt ist. Diese Eigenschaft ist standardmäßig auf 3 gesetzt.

sleepBeforeRetryDurationInMinutes

Beschreibung

Die Ruhezeit in Minuten zwischen aufeinanderfolgenden Versuchen. Diese Eigenschaft ist standardmäßig auf 5 Minuten gesetzt.

aggregationRunningCheckSql

Beschreibung

Mit dieser Eigenschaft können Sie angepasstes SQL definieren, mit dessen Ausführung angezeigt werden kann, ob das Flag "lock" für die Berichtsaggregation festgelegt ist. Diese Eigenschaft ist standardmäßig leer.

Wenn diese Eigenschaft nicht festgelegt ist, führt der ETL-Prozess das folgende SQL aus, um das Flag "lock" abzurufen.

```
select count(1) AS ACTIVERUNS from uari_pattern_lock where islock='Y'
=> If ACTIVERUNS is > 0, lock is set
```

aggregationRunningCheck

Beschreibung

Aktivieren oder inaktivieren Sie die Prüfung, ob die Berichtsaggregation aktiv ist, bevor die Ausführung des ETL-Prozesses erfolgt. Diese Eigenschaft ist standardmäßig aktiviert.

Anhang C. Interact Designumgebung - Konfigurationseigenschaften

In diesem Abschnitt werden alle Konfigurationseigenschaften für die Interact-Designumgebung beschrieben.

Campaign | partitions | partition[n] | reports

Die Eigenschaft **Campaign | partitions | partition[n] | reports** definiert die verschiedenen Ordnertypen für Berichte.

offerAnalysisTabCachedFolder

Beschreibung

Die Eigenschaft `offerAnalysisTabCachedFolder` gibt die Ordnerposition des Ordners an, der die Informationen für Bursting-Angebotsberichte (erweiterte Angebotsberichte) enthält, die auf der Registerkarte "Analyse" aufgeführt sind, wenn Sie den Ordner durch Klicken auf den Link "Analyse" im Navigationsbereich öffnen. Der Pfad wird mithilfe der XPath-Schreibweise angegeben.

Standardwert

```
/content/folder[@name='Affinium Campaign - Object Specific
Reports']/folder[@name='offer']/folder[@name='cached']
```

segmentAnalysisTabOnDemandFolder

Beschreibung

Die Eigenschaft `segmentAnalysisTabOnDemandFolder` gibt die Ordnerposition des Ordners an, der die Segmentberichte enthält, die auf der Registerkarte **Analyse** eines Segments aufgeführt sind. Der Pfad wird mithilfe der XPath-Schreibweise angegeben.

Standardwert

```
/content/folder[@name='Affinium Campaign - Object Specific
Reports']/folder[@name='segment']/folder[@name='cached']
```

offerAnalysisTabOnDemandFolder

Beschreibung

Die Eigenschaft `offerAnalysisTabOnDemandFolder` gibt die Ordnerposition des Ordners an, der die Angebotsberichte enthält, die auf der Registerkarte **Analyse** eines Angebots aufgeführt sind. Der Pfad wird mithilfe der XPath-Schreibweise angegeben.

Standardwert

```
/content/folder[@name='Affinium Campaign - Object Specific
Reports']/folder[@name='offer']
```

segmentAnalysisTabCachedFolder

Beschreibung

Die Eigenschaft segmentAnalysisTabCachedFolder gibt die Ordnerposition des Ordners an, der die Informationen für Bursting-Segmentberichte (erweiterte Segmentberichte) enthält, die auf der Registerkarte "Analyse" aufgeführt sind, wenn Sie den Ordner durch Klicken auf den Link "Analyse" im Navigationsbereich öffnen. Der Pfad wird mithilfe der XPath-Schreibweise angegeben.

Standardwert

```
/content/folder[@name='Affinium Campaign - Object Specific
Reports']/folder[@name='segment']
```

analysisSectionFolder

Beschreibung

Die Eigenschaft analysisSectionFolder gibt die Position des Stammordners an, in dem Berichtinformationen gespeichert werden. Der Pfad wird mithilfe der XPath-Schreibweise angegeben.

Standardwert

```
/content/folder[@name='Affinium Campaign']
```

campaignAnalysisTabOnDemandFolder

Beschreibung

Die Eigenschaft campaignAnalysisTabOnDemandFolder gibt die Position des Ordners an, der die Kampagnenberichte enthält, die auf der Registerkarte **Analyse** einer Kampagne aufgeführt sind. Der Pfad wird mithilfe der XPath-Schreibweise angegeben.

Standardwert

```
/content/folder[@name='Affinium Campaign - Object Specific
Reports']/folder[@name='campaign']
```

campaignAnalysisTabCachedFolder

Beschreibung

Die Eigenschaft campaignAnalysisTabCachedFolder gibt die Position des Ordners an, der die Informationen für Bursting-Kampagnenberichte (erweiterte Kampagnenberichte) enthält, die auf der Registerkarte "Analyse" aufgeführt sind, wenn Sie den Ordner durch Klicken auf den Link "Analyse" im Navigationsbereich öffnen. Der Pfad wird mithilfe der XPath-Schreibweise angegeben.

Standardwert

```
/content/folder[@name='Affinium Campaign - Object Specific
Reports']/folder[@name='campaign']/folder[@name='cached']
```

campaignAnalysisTabEmessageOnDemandFolder

Beschreibung

Die Eigenschaft campaignAnalysisTabEmessageOnDemandFolder gibt die Position des Ordners an, der die eMessage-Berichte enthält, die auf der Registerkarte "Analyse" einer Kampagne aufgeführt sind. Der Pfad wird mithilfe der XPath-Schreibweise angegeben.

Standardwert

/content/folder[@name='Affinium Campaign']/folder[@name='eMessage Reports']

campaignAnalysisTabInteractOnDemandFolder

Beschreibung

Zeichenfolge für den Berichtsserverordner für Interact-Berichte.

Standardwert

/content/folder[@name='Affinium Campaign']/folder[@name='Interact Reports']

Verfügbarkeit

Diese Eigenschaft ist nur anwendbar, wenn Sie Interact installieren.

interactiveChannelAnalysisTabOnDemandFolder

Beschreibung

Zeichenfolge für den Berichtsserverordner für Berichte über die Registerkarte "Analyse des interaktiven Kanals".

Standardwert

/content/folder[@name='Affinium Campaign - Object Specific Reports']/folder[@name='interactive channel']

Verfügbarkeit

Diese Eigenschaft ist nur anwendbar, wenn Sie Interact installieren.

Campaign | partitions | partition[n] | Interact | contactAndResponseHistTracking

Diese Konfigurationseigenschaften definieren die Einstellungen für das Interact-Modul für Kontakt- und Antwortverlauf.

isEnabled

Beschreibung

Wenn der Wert auf yes festgelegt ist, wird das Interact-Modul für Kontakt- und Antwortverlauf aktiviert, das die Interact-Kontakt- und Antwortverlaufsdaten aus den Staging-Tabellen in der Laufzeitumgebung von Interact in die Campaign-Kontakt- und Antwortverlaufstabellen kopiert. Die Eigenschaft interactInstalled muss ebenfalls auf yes gesetzt werden.

Standardwert

no

Gültige Werte

yes | no

Verfügbarkeit

Diese Eigenschaft ist nur anwendbar, wenn Interact installiert ist.

runOnceADay

Beschreibung

Gibt an, dass der Kontakt- und Antwortverlauf-ETL-Prozess einmal pro Tag ausgeführt wird. Wenn Sie diese Eigenschaft auf `Yes` festlegen, wird der ETL-Prozess während des geplanten Intervalls ausgeführt, der durch `preferredStartTime` und `preferredEndTime` festgelegt ist.

Wenn der ETL-Prozess mehr als 24 Stunden für die Ausführung benötigt und dadurch die Startzeit am nächsten Tag versäumt, überspringt er diesen Tag und wird zur geplanten Zeit am nächsten Tag ausgeführt. Beispiel: Wenn der ETL-Prozess so konfiguriert ist, dass er zwischen 1:00 und 3:00 ausgeführt wird, und der Prozess um 1:00 am Montag startet und um 2:00 am Dienstag abgeschlossen wird, wird die nächste Ausführung, die ursprünglich für 1:00 am Dienstag geplant war, übersprungen und der nächste ETL-Prozess startet um 1:00 am Mittwoch.

Die ETL-Planung berücksichtigt nicht die Sommerzeit. Wenn die Ausführung des ETL-Prozesses beispielsweise zwischen 1:00 und 3:00 geplant ist, könnte er um 0:00 oder 2:00 ausgeführt werden, wenn die Sommerzeit einsetzt.

Standardwert

No

Verfügbarkeit

Diese Eigenschaft ist nur anwendbar, wenn Interact installiert ist.

processSleepIntervalInMinutes

Beschreibung

Die Anzahl von Minuten, die das Interact-Modul für Kontakt- und Antwortverlauf wartet, bevor es Daten aus den Staging-Tabellen der Laufzeitumgebung von Interact in die Campaign-Kontakt- und Antwortverlaufstabellen kopiert.

Standardwert

60

Gültige Werte

Eine beliebige Ganzzahl größer 0.

Verfügbarkeit

Diese Eigenschaft ist nur anwendbar, wenn Interact installiert ist.

preferredStartTime

Beschreibung

Die bevorzugte Zeit für den Start des täglichen ETL-Prozesses. Wenn diese Eigenschaft zusammen mit der Eigenschaft "preferredEndTime" verwendet wird, legt sie das bevorzugte Zeitintervall für die Ausführung des ETL-Prozesses fest. Der ETL-Prozess startet während des angegebenen Zeitintervalls und verarbeitet maximal die mit `maxJDBCFetchBatchSize` angegebene Anzahl von Datensätzen. Das Format ist HH:mm:ss AM oder PM unter Verwendung des 12-Stunden-Formats.

Standardwert

12:00:00 AM

Verfügbarkeit

Diese Eigenschaft ist nur anwendbar, wenn Interact installiert ist.

preferredEndTime

Beschreibung

Die bevorzugte Zeit für den Abschluss des täglichen ETL-Prozesses. Wenn diese Eigenschaft zusammen mit der Eigenschaft "preferredStartTime" verwendet wird, legt sie das bevorzugte Zeitintervall für die Ausführung des ETL-Prozesses fest. Der ETL-Prozess startet während des angegebenen Zeitintervalls und verarbeitet maximal die mit `maxJDBCFetchBatchSize` angegebene Anzahl von Datensätzen. Das Format ist HH:mm:ss AM oder PM unter Verwendung des 12-Stunden-Formats.

Standardwert

2:00:00 AM

Verfügbarkeit

Diese Eigenschaft ist nur anwendbar, wenn Interact installiert ist.

purgeOrphanResponseThresholdInMinutes

Beschreibung

Die Anzahl von Minuten, die das Interact-Modul für Kontakt- und Antwortverlauf wartet, bevor Antworten ohne entsprechenden Kontakt bereinigt werden. So wird vermieden, dass Antworten ohne Kontakte protokolliert werden.

Standardwert

180

Gültige Werte

Eine beliebige Ganzzahl größer 0.

Verfügbarkeit

Diese Eigenschaft ist nur anwendbar, wenn Interact installiert ist.

maxJDBCInsertBatchSize

Beschreibung

Die maximale Anzahl der Datensätze eines JDBC-Batches vor dem Ausführen der Abfrage. Dies ist nicht die maximale Anzahl von Datensätzen, die das Interact-Modul für Kontakt- und Antwortverlauf in einer einzelnen Iteration verarbeitet. Während jeder Iteration verarbeitet das Interact-Modul für Kontakt- und Antwortverlauf alle verfügbaren Datensätze aus den Staging-Tabellen. Diese Datensätze werden jedoch in `maxJDBCInsertSize`-Datenblöcke unterteilt.

Standardwert

1000

Gültige Werte

Eine beliebige Ganzzahl größer 0.

Verfügbarkeit

Diese Eigenschaft ist nur anwendbar, wenn Interact installiert ist.

maxJDBCFetchBatchSize

Beschreibung

Die maximale Anzahl der Datensätze eines von der Staging-Datenbank abzurufenden JDBC-Batches. Sie müssen diesen Wert möglicherweise erhöhen, um die Leistung des Moduls für Kontakt- und Antwortverlauf zu optimieren.

Beispiel: Um 2,5 Millionen Datensätze zum Kontaktverlauf pro Tag zu verarbeiten, sollten Sie `maxJDBCFetchBatchSize` auf einen höheren Wert als 2,5 M festlegen, damit alle Datensätze für einen Tag verarbeitet werden.

Sie können dann `maxJDBCFetchChunkSize` und `maxJDBCInsertBatchSize` auf kleinere Werte festlegen (in diesem Beispiel vielleicht auf 50.000 bzw. 10.000). Einige Datensätze vom nächsten Tag werden möglicherweise ebenfalls verarbeitet, aber bis zum nächsten Tag beibehalten.

Standardwert

1000

Gültige Werte

Eine beliebige Ganzzahl größer 0

maxJDBCFetchChunkSize

Beschreibung

Die maximale Anzahl einer JDBC-Datenblockgröße von Daten, die während des ETL-Prozesses (Extrahieren, Transformieren, Laden) gelesen werden. In manchen Fällen kann eine Datenblockgröße, die größer als die Einfügegröße ist, die Geschwindigkeit des ETL-Prozesses verbessern.

Standardwert

1000

Gültige Werte

Eine beliebige Ganzzahl größer 0

deleteProcessedRecords

Beschreibung

Legt fest, ob Kontakt- und Antwortverlaufsdatensätze beibehalten werden, nachdem sie verarbeitet wurden.

Standardwert

Yes

completionNotificationScript

Beschreibung

Gibt den absoluten Pfad zu einem Script an, das ausgeführt wird, wenn der ETL-Prozess abgeschlossen ist. Wenn Sie ein Script angeben, werden fünf Argumente an das Abschlussbenachrichtigungsscript übergeben: Startzeit, Endzeit, Gesamtzahl der verarbeiteten Kontakt- und Antwortverlaufsdatensätze. Die Start- und Endzeit sind numerische Werte, die die Anzahl der seit 1970 vergangenen Millisekunden darstellen. Das Statusargument

gibt an, ob der ETL-Job erfolgreich war oder fehlgeschlagen ist. "0" gibt einen erfolgreichen ETL-Job an. "1" gibt an, dass er fehlgeschlagen ist und Fehler im ETL-Job aufgetreten sind.

Standardwert

Keiner.

fetchSize

Beschreibung

Ermöglicht es Ihnen, den JDBC-Abrufumfang beim Abrufen aus Staging-Tabellen festzulegen.

Passen Sie besonders bei Oracle-Datenbanken diese Einstellung an die Anzahl von Datensätzen an, die JDBC bei jedem Netz-Umlauf abrufen soll. Bei umfangreichen Batches von 100 KB oder größer versuchen Sie 10.000. Achten Sie darauf, hier keinen zu großen Wert zu verwenden, weil sich das auf die Speicherbelegung auswirkt und die Leistungszunahme vernachlässigbar, wenn nicht sogar negativ ist.

Standardwert

Keiner.

daysBackInHistoryToLookupContact

Beschreibung

Grenzt die Datensätze, die bei Abfragen des Antwortverlaufs durchsucht werden, ein auf eine angegebene Anzahl vergangener Tage. Bei Datenbanken mit einer hohen Anzahl Antwortverlaufsdatensätze kann die Verarbeitungszeit verkürzt werden, wenn die Suchperiode auf die angegebene Anzahl Tage eingegrenzt wird.

Der Standardwert "0" gibt an, dass alle Datensätze durchsucht werden.

Standardwert

0 (Null)

Campaign | partitions | partition[n] | Interact | contactAndResponseHistTracking | runtimeDataSources | [runtimeDataSource]

Diese Konfigurationseigenschaften definieren die Datenquelle für das Interact-Modul für den Kontakt- und Antwortverlauf.

jndiName

Beschreibung

Verwenden Sie die Eigenschaft `systemTablesDataSource`, um die JNDI-Datenquelle (JNDI = Java Naming and Directory Interface) zu identifizieren, die auf dem Anwendungsserver (WebSphere oder WebLogic) für die Interact-Laufzeittabellen definiert ist.

Die Interact-Laufzeitdatenbank ist die mit den DLL-Scripts `aci_runtime` und `aci_populate_runtime` belegte Datenbank und enthält beispielsweise (u. a.) folgende Tabellen: `UACI_CHOfferAttrib` und `UACI_DefaultedStat`.

Standardwert

Kein Standardwert definiert.

Verfügbarkeit

Diese Eigenschaft ist nur anwendbar, wenn Interact installiert ist.

databaseType

Beschreibung

Datenbanktyp für die Interact-Laufzeitdatenquelle.

Standardwert

SQLServer

Gültige Werte

SQLServer | Oracle | DB2

Verfügbarkeit

Diese Eigenschaft ist nur anwendbar, wenn Interact installiert ist.

schemaName

Beschreibung

Der Name des Schemas, das die Staging-Tabellen des Moduls für den Kontakt- und Antwortverlauf enthält. Dieser Name sollte mit den Tabellen der Laufzeitumgebung übereinstimmen.

Sie müssen kein Schema definieren.

Standardwert

Kein Standardwert definiert.

Campaign | partitions | partition[n] | Interact | contactAndResponseHistTracking | contactTypeMappings

Diese Konfigurationseigenschaften definieren den Kontakttyp von Campaign, der zu Berichts- oder Lernzwecken einem "Kontakt" zugeordnet wird.

contacted

Beschreibung

Der Wert, der der Spalte `ContactStatusID` der Tabelle `UA_DtlContactHist` in den Campaign-Systemtabellen für einen Angebotskontakt zugewiesen wird. Der Wert muss ein gültiger Eintrag in der Tabelle `UA_ContactStatus` sein. Hinweise zum Hinzufügen von Kontakttypen finden Sie im *Campaign-Administratorhandbuch*.

Standardwert

2

Gültige Werte

Eine Ganzzahl größer 0.

Verfügbarkeit

Diese Eigenschaft ist nur anwendbar, wenn Interact installiert ist.

Campaign | partitions | partition[n] | Interact | contactAndResponseHistTracking | responseTypeMappings

Diese Konfigurationseigenschaften definieren die Antworten für das Annehmen oder Ablehnen für die Berichterstellung und das Lernmodul.

accept

Beschreibung

Der Wert, der der Spalte `ResponseTypeID` der Tabelle `UA_ResponseHistory` in den Systemtabellen von Campaign für ein angenommenes Angebot zugewiesen wird. Der Wert muss ein gültiger Eintrag in der Tabelle `UA_UsrResponseType` sein. Der Spalte `CountsAsResponse` sollte der Wert `1`, eine Antwort, zugewiesen werden.

Hinweise zum Hinzufügen von Antworttypen finden Sie im *Campaign-Administratorhandbuch*.

Standardwert

3

Gültige Werte

Eine Ganzzahl größer 0.

Verfügbarkeit

Diese Eigenschaft ist nur anwendbar, wenn Interact installiert ist.

reject

Beschreibung

Der Wert, der der Spalte `ResponseTypeID` der Tabelle `UA_ResponseHistory` in den Systemtabellen von Campaign für ein abgelehntes Angebot zugewiesen wird. Der Wert muss ein gültiger Eintrag in der Tabelle `UA_UsrResponseType` sein. Der Spalte `CountsAsResponse` sollte der Wert `2`, eine Ablehnung, zugewiesen werden. Hinweise zum Hinzufügen von Antworttypen finden Sie im *Campaign-Administratorhandbuch*.

Standardwert

8

Gültige Werte

Eine beliebige Ganzzahl größer 0.

Verfügbarkeit

Diese Eigenschaft ist nur anwendbar, wenn Interact installiert ist.

Campaign | partitions | partition[n] | Interact | report

Diese Konfigurationseigenschaften definieren die Berichtsnamen bei der Integration in Cognos.

interactiveCellPerformanceByOfferReportName

Beschreibung

Name für den Bericht "Erfolg der interaktiven Zellen nach Angebot". Dieser Name muss mit dem Namen dieses Berichts auf dem Cognos-Server übereinstimmen.

Standardwert

Erfolg der interaktiven Zellen nach Angebot

treatmentRuleInventoryReportName

Beschreibung

Name für den Bericht "Inventar der Verfahrensregeln". Dieser Name muss mit dem Namen dieses Berichts auf dem Cognos- Server übereinstimmen.

Standardwert

Bestandsaufnahme Verfahrensregeln des Kanals

deploymentHistoryReportName

Beschreibung

Name für den Bericht "Implementierungsverlaufsbericht". Dieser Name muss mit dem Namen dieses Berichts auf dem Cognos-Server übereinstimmen.

Standardwert

Verlauf der Kanalbereitstellung

Campaign | partitions | partition[n] | Interact | learning

Diese Konfigurationseigenschaften ermöglichen eine Feinabstimmung des integrierten Lernmoduls.

confidenceLevel

Beschreibung

Ein Prozentsatz, der angibt, wie stark das Lerndienstprogramm den gesammelten Daten vertrauen soll, bevor es von der Untersuchung zur Nutzung wechselt. Mit dem Wert 0 wird die Untersuchung effektiv beendet.

Diese Eigenschaft ist nur anwendbar, wenn die Eigenschaft `Interact > offerserving > optimizationType` für die Interact-Laufzeit auf `BuiltInLearning` festgelegt ist.

Standardwert

95

Gültige Werte

Eine Ganzzahl zwischen 0 und 95, teilbar durch 5 oder 99.

validateonDeployment

Beschreibung

Wenn `No` eingestellt ist, wird von Interact nicht das Lernmodul überprüft, wenn Sie es bereitstellen. Wenn `Yes` eingestellt ist, wird von Interact das Lernmodul überprüft, wenn Sie es bereitstellen.

Standardwert

`No`

Gültige Werte

Yes | No

maxAttributeNames

Beschreibung

Die maximale Anzahl von Lernattributen, die das Interact-Lerndienstprogramm überwachen soll.

Diese Eigenschaft ist nur anwendbar, wenn die Eigenschaft `Interact > offerserving > optimizationType` für die Interact-Laufzeit auf `BuiltInLearning` festgelegt ist.

Standardwert

`10`

Gültige Werte

Beliebige Ganzzahl.

maxAttributeValues

Beschreibung

Die maximale Anzahl von Werten, die das Interact-Lernmodul für die einzelnen Lernattribute verfolgen soll.

Diese Eigenschaft ist nur anwendbar, wenn die Eigenschaft `Interact > offerserving > optimizationType` für die Interact-Laufzeit auf `BuiltInLearning` festgelegt ist.

Standardwert

5

otherAttributeValue

Beschreibung

Der Standardname für den Attributwert, der zur Darstellung aller Attributwerte dient, die den Wert von `maxAttributeValues` überschreiten.

Diese Eigenschaft ist nur anwendbar, wenn die Eigenschaft `Interact > offerserving > optimizationType` für die Interact-Laufzeit auf `BuiltInLearning` festgelegt ist.

Standardwert

`Other`

Gültige Werte

Eine Zeichenfolge oder Zahl.

Beispiel

Wenn der Wert von maxAttributeValues auf 3 festgelegt ist und otherAttributeValue auf "Other" festgelegt ist, verfolgt das Lernmodul die ersten drei Werte. Alle anderen Werte werden der anderen Kategorie zugewiesen. Wenn Sie beispielsweise das Benutzerattribut Haarfarbe verfolgen möchten und die ersten fünf Benutzer die Haarfarbe schwarz, braun, blond, rot und grau haben, so verfolgt das Lerndienstprogramm die Haarfarben schwarz, braun und blond. Die Farben rot und grau werden unter `otherAttributeValue` zusammengefasst.

percentRandomSelection

Beschreibung

Der Prozentsatz der Zeit, während der das Lernmodul ein Zufallsangebot anzeigt. Wenn beispielsweise der Wert von percentRandomSelection auf 5 festgelegt wird, zeigt das Lernmodul während 5 % der Zeit (5 aus jeweils 100 Empfehlungen) unabhängig von der Bewertung ein Zufallsangebot an. Durch Aktivieren von `percentRandomSelection` wird die Konfigurationseigenschaft `offerTieBreakMethod` überschrieben. Wenn `percentRandomSelection` aktiviert ist, wird diese Eigenschaft unabhängig davon eingestellt, ob die Lernfunktion aktiv ist und ob die integrierte oder externe Lernfunktion verwendet wird.

Standardwert

5

Gültige Werte

Eine beliebige Ganzzahl von 0 (inaktiviert die Funktion `percentRandomSelection`) bis 100.

recencyWeightingFactor

Beschreibung

Die Dezimaldarstellung eines Prozentsatzes der Datenmenge, die durch den Wert von `recencyWeightingPeriod` definiert wird. Beispielsweise bedeutet der Standardwert 0,15, dass 15 % der vom Lerndienstprogramm verwendeten Daten aus dem Wert von `recencyWeightingPeriod` stammen.

Diese Eigenschaft ist nur anwendbar, wenn die Eigenschaft `Interact > offerserving > optimizationType` für die Interact-Laufzeit auf `BuiltInLearning` festgelegt ist.

Standardwert

0,15

Gültige Werte

Ein Dezimalwert kleiner als 1.

recencyWeightingPeriod

Beschreibung

Die Größe von Daten in Stunden, denen der Prozentsatz des Gewichts `recencyWeightingFactor` vom Lernmodul gewährt wurde. Beispielsweise bedeutet der Standardwert 120, dass der Wert von `recencyWeightingFactor` der vom Lernmodul verwendeten Daten aus den letzten 120 Stunden stammen.

Diese Eigenschaft ist nur anwendbar, wenn `optimizationType` auf `builtInLearning` festgelegt ist.

Standardwert

120

minPresentCountThreshold

Beschreibung

Die minimale Anzahl der Anzeigewiederholungen eines Angebots, bevor seine Daten in Berechnungen verwendet werden und das Lernmodul in den Untersuchungsmodus wechselt.

Standardwert

0

Gültige Werte

Eine Ganzzahl größer oder gleich 0.

enablePruning

Beschreibung

Wenn Sie `Yes` festlegen, bestimmt das Interact-Lerndienstprogramm algorithmisch, wenn ein Lernattribut (Standard oder dynamisch) nicht prognostiziert werden kann. Wenn ein Lernattribut nicht prognostiziert werden kann, wird dieses Attribut bei der Ermittlung des Gewichts für ein Angebot vom Lernmodul nicht berücksichtigt. Dieser Vorgang setzt sich fort, bis das Lernmodul Daten aggregiert.

Wenn dieser Wert auf `No` festgelegt ist, verwendet das Lernmodul immer alle Lernattribute. Dadurch, dass nicht prognostizierbare Attribute nicht gelöscht werden, arbeitet das Lernmodul möglicherweise nicht so präzise wie eigentlich möglich.

Standardwert

Yes

Gültige Werte

Yes | No

Campaign | partitions | partition[n] | Interact | learning | learningAttributes | [learningAttribute]

Diese Konfigurationseigenschaften definieren die Lernattribute.

attributeName

Beschreibung

Jeder Wert von `attributeName` ist der Name eines Benutzerattributs, das vom Lernmodul überwacht werden soll. Dieser Wert muss mit dem Namen eines Name/Wert-Paars in den Sitzungsdaten übereinstimmen.

Diese Eigenschaft ist nur anwendbar, wenn die Eigenschaft `Interact > offerserving > optimizationType` für die Interact-Laufzeit auf `BuiltInLearning` festgelegt ist.

Standardwert

Kein Standardwert definiert.

Campaign | partitions | partition[n] | Interact | deployment

Diese Konfigurationseigenschaften definieren die Implementierungseinstellungen.

chunkSize

Beschreibung

Die maximale Größe der Fragmentierung in KB für jedes Interact-Implementierungspaket.

Standardwert

500

Verfügbarkeit

Diese Eigenschaft ist nur anwendbar, wenn Interact installiert ist.

Campaign | partitions | partition[n] | Interact | serverGroups | [serverGroup]

Diese Konfigurationseigenschaften definieren die Servergruppeneinstellungen.

serverGroupName

Beschreibung

Der Name der Interact-Laufzeitservergruppe. Dies ist der Name, der auf der Registerkarte "Interaktiver Kanal - Übersicht" angezeigt wird.

Standardwert

Kein Standardwert definiert.

Verfügbarkeit

Diese Eigenschaft ist nur anwendbar, wenn Interact installiert ist.

Campaign | partitions | partition[n] | Interact | serverGroups | [serverGroup] | instanceURLs | [instanceURL]

Diese Konfigurationseigenschaften definieren die Interact-Laufzeitserver.

instanceURL

Beschreibung

Die URL des Interact-Laufzeitservers. Eine Servergruppe kann mehrere Interact-Laufzeitserver enthalten, jeder Server muss allerdings unter einer neuen Kategorie erstellt werden.

Standardwert

Kein Standardwert definiert.

Beispiel

`http://server:port/interact`

Verfügbarkeit

Diese Eigenschaft ist nur anwendbar, wenn Interact installiert ist.

Campaign | partitions | partition[n] | Interact | flowchart

Diese Konfigurationseigenschaften definieren die Laufzeitumgebung von Interact, die für Testläufe interaktiver Ablaufdiagramme verwendet wird.

serverGroup

Beschreibung

Der Name der Servergruppe von Interact, die von Campaign zur Ausführung eines Testlaufs verwendet wird. Dieser Name muss mit dem Kategorienamen übereinstimmen, den Sie unter `serverGroups` erstellen.

Standardwert

Kein Standardwert definiert.

Verfügbarkeit

Diese Eigenschaft ist nur anwendbar, wenn Interact installiert ist.

dataSource

Beschreibung

Verwenden Sie die Eigenschaft `dataSource`, um die physische Datenquelle für Campaign zu identifizieren, die beim Ausführen von Testläufen interaktiver Ablaufdiagramme verwendet werden soll. Diese Eigenschaft muss übereinstimmen mit der Datenquelle, die von der Eigenschaft `Campaign > Partitionen > PartitionN > dataSources` für die Testlaufdatenquelle, die für die Designzeit von Interact definiert ist, definiert wurde.

Standardwert

Kein Standardwert definiert.

Verfügbarkeit

Diese Eigenschaft ist nur anwendbar, wenn Interact installiert ist.

eventPatternPrefix

Beschreibung

Bei der Eigenschaft `eventPatternPrefix` handelt es sich um einen Zeichenfolgewert, der den Namen von Ereignismustern vorangestellt wird, damit sie in den Prozessen 'Auswählen' oder 'Entscheidung' von interaktiven Ablaufdiagrammen in Ausdrücken verwendet werden können.

Beachten Sie, dass Sie bei Änderung dieses Werts im interaktiven Kanal globale Änderungen bereitstellen müssen, damit die aktualisierte Konfiguration wirksam wird.

Standardwert

`EventPattern`

Verfügbarkeit

Diese Eigenschaft ist nur anwendbar, wenn Interact installiert ist.

Campaign | partitions | partition[n] | Interact | whiteList | [AudienceLevel] | DefaultOffers

Diese Konfigurationseigenschaften definieren den Standardzellencode für die Standardangebotstabelle. Sie müssen diese Eigenschaften nur konfigurieren, wenn Sie globale Angebotszuweisungen definieren.

DefaultCellCode

Beschreibung

Der Standardzellencode, den Interact verwendet, wenn Sie keinen Zellencode in der Standardangebotstabelle definieren.

Standardwert

Kein Standardwert definiert.

Gültige Werte

Eine Zeichenkette, die mit dem in Campaign definierten Zellencodeformat übereinstimmt.

Verfügbarkeit

Diese Eigenschaft ist nur anwendbar, wenn Interact installiert ist.

Campaign | partitions | partition[n] | Interact | whiteList | [AudienceLevel] | offersBySQL

Diese Konfigurationseigenschaften definieren den Standardzellencode für die offersBySQL-Tabelle. Sie müssen diese Eigenschaften nur konfigurieren, wenn Sie SQL-Abfragen verwenden, um einen gewünschten Satz von möglichen Angeboten zu beziehen.

DefaultCellCode

Beschreibung

Der Standardzellencode, den Interact für ein Angebot in der/den OffersBySQL-Tabelle(n) verwendet, die einen Nullwert in der Zellencodespalte hat/haben (oder wenn der Zellencode fehlt). Dieser Wert muss ein gültiger Zellencode sein.

Standardwert

Kein Standardwert definiert.

Gültige Werte

Eine Zeichenkette, die mit dem in Campaign definierten Zellencodeformat übereinstimmt.

Verfügbarkeit

Diese Eigenschaft ist nur anwendbar, wenn Interact installiert ist.

Campaign | partitions | partition[n] | Interact | whiteList | [AudienceLevel] | ScoreOverride

Diese Konfigurationseigenschaften definieren den Standardzellencode für die Tabelle für die Bewertungsüberschreibung. Sie müssen diese Eigenschaften nur konfigurieren, wenn Sie einzelne Angebotszuweisung definieren.

DefaultCellCode

Beschreibung

Der Standardzellencode, den Interact verwendet, wenn Sie in der Tabelle für die Bewertungsüberschreibung keinen Zellencode definieren.

Standardwert

Kein Standardwert definiert.

Gültige Werte

Eine Zeichenkette, die mit dem in Campaign definierten Zellencodeformat übereinstimmt.

Verfügbarkeit

Diese Eigenschaft ist nur anwendbar, wenn Interact installiert ist.

Campaign | partitions | partition[n] | server | internal

Eigenschaften in dieser Kategorie geben Integrationseinstellungen und die internalID-Grenzwerte für die ausgewählte Campaign-Partition an. Wenn Ihre Campaign-Installation mehrere Partitionen aufweist, legen Sie diese Eigenschaften für jede Partition fest, die beeinflusst werden soll.

internalIdLowerLimit

Konfigurationskategorie

Campaign|partitions|partition[n]|server|internal

Beschreibung

Die Eigenschaften internalIdUpperLimit und internalIdLowerLimit beschränken die internen IDs von Campaign so, dass diese im angegebenen Bereich liegen müssen. Beachten Sie, dass die Werte einschließlich sind: Das heißt, in Campaign kann sowohl die untere als auch die obere Grenze verwendet werden.

Standardwert

0 (Null)

internalIdUpperLimit

Konfigurationskategorie

Campaign|partitions|partition[n]|server|internal

Beschreibung

Die Eigenschaften internalIdUpperLimit und internalIdLowerLimit beschränken die internen IDs von Campaign so, dass diese im angegebenen Bereich liegen. Die Werte sind einschließlich, das heißt, in Campaign kann sowohl der untere als auch der obere Grenzwert verwendet werden. Wenn Distributed Marketing installiert ist, setzen Sie den Wert auf 2147483647.

Standardwert

4294967295

eMessageInstalled

Konfigurationskategorie

Campaign|partitions|partition[n]|server|internal

Beschreibung

Gibt an, dass eMessage installiert ist. Wenn Sie Yes auswählen, sind die eMessage-Funktionen in der Campaign-Benutzeroberfläche verfügbar.

Das IBM Installationsprogramm legt diesen Wert für die Standardpartition Ihrer eMessage-Installation auf Yes fest. Für weitere Partitionen, auf denen eMessage installiert ist, müssen Sie diese Eigenschaft manuell konfigurieren.

Standardwert

No

Gültige Werte

Yes | No

interactInstalled

Konfigurationskategorie

`Campaign|partitions|partition[n]|server|internal`

Beschreibung

Nach der Installation der Designumgebung von Interact sollte diese Konfigurationseigenschaft auf `Yes` festgelegt werden, um die Designumgebung von Interact in Campaign zu aktivieren.

Wenn Interact nicht installiert ist, legen Sie den Wert auf `No` fest. Durch Festlegen dieser Eigenschaft auf `No` werden die Menüs und Optionen von Interact nicht aus der Benutzeroberfläche entfernt. Um Menüs und Optionen zu entfernen, müssen Sie die Registrierung von Interact mithilfe des Dienstprogramms `configTool` manuell aufheben.

Standardwert

No

Gültige Werte

Yes | No

Verfügbarkeit

Diese Eigenschaft ist nur anwendbar, wenn Interact installiert ist.

MO_UC_integration

Konfigurationskategorie

`Campaign|partitions|partition[n]|server|internal`

Beschreibung

Ermöglicht die Integration in Marketing Operations für diese Partition, wenn die Integration in den **Platform**-Konfigurationseinstellungen aktiviert ist. Weitere Informationen hierzu finden Sie im *IBM Marketing Operations- und Campaign-Integrationshandbuch*.

Standardwert

No

Gültige Werte

Yes | No

MO_UC_BottomUpTargetCells

Konfigurationskategorie

`Campaign|partitions|partition[n]|server|internal`

Beschreibung

Ermöglicht auf dieser Partition Bottom-up-Zellen für Arbeitsblätter für Zielzellen, wenn **MO_UC_integration** aktiviert ist. Wenn als Wert `Yes` festgelegt ist, sind sowohl Top-down- als auch Bottom-up-Zielzellen sichtbar, die Bottom-up-Zielzellen sind jedoch schreibgeschützt. Weitere Informationen hierzu finden Sie im *IBM Marketing Operations- und Campaign-Integrationshandbuch*.

Standardwert

No

Gültige Werte

Yes | No

Legacy_campaigns

Konfigurationskategorie

`Campaign|partitions|partition[n]|server|internal`

Beschreibung

Ermöglicht auf dieser Partition den Zugriff auf Kampagnen, die vor der Integration von Marketing Operations und Campaign erstellt wurden. Gilt nur, wenn **MO_UC_integration** auf `Yes` festgelegt ist. Veraltete Kampagnen schließen auch Kampagnen ein, die in Campaign 7.x erstellt und mit Plan 7.x-Projekten verlinkt wurden. Weitere Informationen hierzu finden Sie im *IBM Marketing Operations- und Campaign-Integrationshandbuch*.

Standardwert

No

Gültige Werte

Yes | No

IBM Marketing Operations - Angebotsintegration

Konfigurationskategorie

`Campaign|partitions|partition[n]|server|internal`

Beschreibung

Ermöglicht die Verwendung von Marketing Operations zur Durchführung von Lifecycle-Management-Aufgaben für Angebote auf dieser Partition, wenn **MO_UC_integration** für diese Partition aktiviert ist. Die Angebotsintegration muss in Ihren **Platform**-Konfigurationseinstellungen aktiviert sein. Weitere Informationen hierzu finden Sie im *IBM Marketing Operations- und Campaign-Integrationshandbuch*.

Standardwert

No

Gültige Werte

Yes | No

UC_CM_integration

Konfigurationskategorie

`Campaign|partitions|partition[n]|server|internal`

Beschreibung

Ermöglicht die Digital Analytics-Onlinesegmentintegration für eine Campaign-Partition. Wenn Sie diesen Wert auf `Yes` festlegen, steht im Auswahlprozessfeld die Option zur Verfügung, **Digital Analytics-Segmente** als Eingabe zu verwenden. Um die Digital Analytics-Integration für die einzelnen Partitionen zu konfigurieren, wählen Sie **Einstellungen > Konfiguration> Campaign | partitions | partition[n] | Coremetrics** aus.

Standardwert

No

Gültige Werte

Yes | No

numRowsReadToParseDelimitedFile

Konfigurationskategorie

`Campaign|partitions|partition[n]|server|internal`

Beschreibung

Diese Eigenschaft wird verwendet, wenn eine Datei mit begrenzter Satzlänge als Benutzertabelle zugeordnet wird. Zudem wird sie vom Prozessfeld "Bewertung" verwendet, wenn eine Bewertungsausgabedatei aus IBM SPSS Modeler Advantage Enterprise Marketing Management Edition importiert wird. Um eine Datei mit begrenzter Satzlänge importieren oder zuordnen zu können, muss Campaign die Datei zur Identifizierung der Spalten, Datentypen (Feldtypen) und Spaltenbreiten (Feldlängen) parsen.

Der Standardwert 100 bedeutet, dass Campaign die ersten 50 und die letzten 50 Zeileneinträge in der Datei mit begrenzter Satzlänge überprüft. Campaign ordnet die Feldlänge anschließend basierend auf dem größten innerhalb dieser Einträge gefundenen Wert zu. In den meisten Fällen reicht der Standardwert zur Ermittlung von Feldlängen aus. In sehr großen Dateien mit begrenzter Satzlänge überschreitet ein später hinzugefügtes Feld jedoch möglicherweise die von Campaign berechnete geschätzte Länge. Dies kann einen Fehler während der Laufzeit des Ablaufdiagramms verursachen. Wenn Sie eine sehr große Datei zuordnen, können Sie diesen Wert daher erhöhen, damit Campaign weitere Zeileneinträge überprüfen kann. Bei dem Wert 100 kann Campaign beispielsweise die ersten 100 Zeileneinträge und die letzten 100 Zeileneinträge der Datei überprüfen.

Bei dem Wert 0 wird die gesamte Datei überprüft. Dies ist in der Regel nur dann notwendig, wenn Sie Dateien importieren oder zuordnen, deren Felder eine variable Datenbreite aufweisen, die nicht durch das Lesen der ersten und letzten Zeilen ermittelt werden kann. Bei extrem großen Dateien kann sich die erforderliche Bearbeitungszeit durch das Lesen der gesamten Datei bei der Ausführung einer Tabellenzuordnung und des Prozessfelds "Bewertung" erhöhen.

Standardwert

100

Gültige Werte

0 (alle Zeilen) oder eine beliebige positive Ganzzahl

Campaign | monitoring

Die Eigenschaften in dieser Kategorie geben an, ob die Funktion zur Überwachung von Arbeitsabläufen aktiviert ist, und legen die URL des Servers für die Überwachung von Arbeitsabläufen sowie das Cachingverhalten fest. Die Überwachung von Arbeitsabläufen wird angezeigt und ermöglicht eine Steuerung aktiver Ablaufdiagramme.

cacheCleanupInterval

Beschreibung

Die Eigenschaft cacheCleanupInterval gibt das Intervall zwischen automatischen Bereinigungen des Statuscache für Ablaufdiagramme in Sekunden an.

Diese Eigenschaft ist in früheren Campaign-Versionen als 7.0 nicht verfügbar.

Standardwert

600 (10 Minuten)

cacheRunCompleteTime

Beschreibung

Die Eigenschaft cacheRunCompleteTime gibt die Dauer in Minuten an, über die abgeschlossene Ausführungen zwischengespeichert werden und auf der Überwachungsseite angezeigt werden.

Diese Eigenschaft ist in früheren Campaign-Versionen als 7.0 nicht verfügbar.

Standardwert

4320

monitorEnabled

Beschreibung

Die Eigenschaft monitorEnabled gibt an, ob die Überwachung aktiviert ist.

Diese Eigenschaft ist in früheren Campaign-Versionen als 7.0 nicht verfügbar.

Standardwert

FALSE

Gültige Werte

TRUE | FALSE

serverURL

Beschreibung

Die Eigenschaft Campaign > monitoring > serverURL gibt die URL des Servers für die Überwachung von Arbeitsabläufen an. Dies ist eine obligatorische Einstellung. Ändern Sie den Wert, wenn die Server-URL für die Überwachung von Arbeitsabläufen nicht dem Standardwert entspricht.

Wenn Campaign für die Verwendung von SSL-Verbindungen (SSL = Secure Sockets Layer) konfiguriert ist, legen Sie den Wert dieser Eigenschaft so fest, dass HTTPS verwendet werden muss. Beispiel: serverURL=https://*host:SSL_port*/Campaign/OperationMonitor, wobei gilt:

- *host* ist der Name oder die IP-Adresse der Maschine, auf der die Webanwendung installiert ist.
- *SSL_Port* ist der SSL-Port der Webanwendung.

Beachten Sie das https in der URL.

Standardwert

http://localhost:7001/Campaign/OperationMonitor

monitorEnabledForInteract

Beschreibung

Der Wert TRUE aktiviert den Campaign-JMX-Connector-Server für Interact. Campaign weist keine JMX-Sicherheit auf.

Wenn dieser Wert auf FALSE festgelegt wird, können Sie keine Verbindung zum Campaign-JMX-Connector-Server herstellen.

Diese JMX-Überwachung gilt nur für das Interact-Modul für Kontakt- und Antwortverlauf.

Standardwert

FALSE

Gültige Werte

TRUE | FALSE

Verfügbarkeit

Diese Eigenschaft ist nur anwendbar, wenn Interact installiert ist.

protocol

Beschreibung

Überwachungsprotokoll für den Campaign-JMX-Connector-Server, wenn monitorEnabledForInteract auf "yes" festgelegt ist.

Diese JMX-Überwachung gilt nur für das Interact-Modul für Kontakt- und Antwortverlauf.

Standardwert

JMXMP

Gültige Werte

JMXMP | RMI

Verfügbarkeit

Diese Eigenschaft ist nur anwendbar, wenn Interact installiert ist.

port

Beschreibung

Überwachungsport für den Campaign-JMX-Connector-Server, wenn monitorEnabledForInteract auf "yes" festgelegt ist.

Diese JMX-Überwachung gilt nur für das Interact-Modul für Kontakt- und Antwortverlauf.

Standardwert

2004

Gültige Werte

Eine Ganzzahl zwischen 1025 und 65535.

Verfügbarkeit

Diese Eigenschaft ist nur anwendbar, wenn Interact installiert ist.

Campaign | partitions | partition[n] | Interact | outboundChannels

Diese Konfigurationseigenschaften ermöglichen Ihnen eine Optimierung des Kanals für abgehende Nachrichten für ausgelöste Nachrichten.

Kategoriename

Beschreibung

Diese Eigenschaft definiert den Namen dieses Kanals für abgehende Nachrichten. Der Name muss für jeden Kanal für abgehende Nachrichten eindeutig sein.

Name

Beschreibung

Der Name Ihres Kanals für abgehende Nachrichten.

Anmerkung: Sie müssen Ihren Anwendungsserver erneut starten, damit die Änderungen wirksam werden.

Campaign | partitions | partition[n] | Interact | outboundChannels | Parameter Data

Diese Konfigurationseigenschaften ermöglichen Ihnen eine Optimierung des Kanals für abgehende Nachrichten für ausgelöste Nachrichten.

Kategoriename

Beschreibung

Diese Eigenschaft definiert den Namen dieses Parameters. Der Name muss für jeden Parameter für diesen Kanal für abgehende Nachrichten eindeutig sein.

Wert

Beschreibung

Diese Eigenschaft definiert die Parameter, im Format von Name/Wert-Paaren, die von diesem ausgehenden Gateway benötigt werden.

Anhang D. Echtzeit-Personalisierung von Angeboten auf der Clientseite

In bestimmten Situationen kann es sinnvoll sein, die Echtzeit-Personalisierung von Angeboten ohne Low-Level-Implementierung von SOAP-Aufrufen oder Java Code auf dem Interact Server anzubieten. Dies ist zum Beispiel der Fall, wenn ein Besucher zunächst eine Webseite lädt, auf der nur JavaScript-Inhalte für die erweiterte Programmierung zur Verfügung stehen, oder wenn ein Besucher eine E-Mail-Nachricht öffnet, in der nur HTML-Inhalte möglich sind. IBM Interact stellt mehrere Konnektoren bereit, die eine Echtzeit-Angebotsverwaltung in Situationen ermöglichen, in denen Sie nur die Webinhalte steuern können, die auf der Clientseite geladen werden, oder in denen Sie die Benutzeroberfläche für Interact vereinfachen möchten.

Ihre Interact Installation enthält zwei Konnektoren für die Angebotspersonalisierung, die auf der Clientseite aufgebaut wird:

- „Informationen zum Interact Message Connector". Wenn Sie den Message Connector verwenden, können in Webinhalten in E-Mail-Nachrichten oder anderen elektronischen Medien zum Beispiel Link- und Image-Tags enthalten sein, über die Sie den Interact Server aufrufen können, um Angebotspräsentationen und Landing-Pages zum Durchklicken auf der Seite zu laden.
- „Informationen zum Interact Web Connector" auf Seite 324. Wenn Sie den Web Connector (auch JS Connector genannt) verwenden, können Webseiten auf der Clientseite JavaScript verwenden, um die Prioritäten, die Präsentation und den Kontakt- oder Antwortverlauf von Angeboten über Angebotspräsentationen und Landing-Pages zum Durchklicken auf der Seite zu verwalten.

Informationen zum Interact Message Connector

Mit dem Interact Message Connector können E-Mail-Nachrichten und andere elektronische Medien IBM Interact aufrufen, um während der Öffnungszeit die Präsentation personalisierter Angebote zu ermöglichen. Außerdem kann sich der Kunde jederzeit durch die Nachricht bis zur angegebenen Website klicken. Um dies zu ermöglichen, werden zwei wichtige Tags verwendet: Der Image-Tag (`IMG`), der während der Öffnungszeit die personalisierten Angebote lädt, und der Link-Tag (`A`), der Informationen zum Durchklicken bereitstellt und den Kunden auf eine bestimmte Landing-Page weiterleitet.

Beispiel

Das folgende Beispiel zeigt einen HTML-Code, den Sie in eine Werbefläche einschließen können (z. B. innerhalb einer E-Mail-Nachricht). Der Code enthält sowohl eine URL für den `IMG`-Tag (damit können Informationen übergeben werden, wenn das Dokument auf dem Interact Server geöffnet wird, um im Gegenzug die entsprechende Grafik für das Angebot abzurufen) als auch eine URL für den `A`-Tag (damit wird festgelegt, welche Informationen beim Durchklicken an den Interact Server übergeben werden):

```
<a href="http://www.example.com/MessageConnector/
offerClickthru.jsp?msgId=1234&linkId=1&userid=1&referral=test"><img
src="http:// www.example.com/MessageConnector/offerImage.jsp?msgId=1234
&linkId=1&userid=1&incomeLevel=5&incomeType=numeric"/></a>
```

Im folgenden Beispiel ist ein IMG-Tag in einen A-Tag eingebunden. Dies führt zu folgendem Verhalten:

1. Wenn die E-Mail-Nachricht geöffnet wird, empfängt der Message Connector eine Anfrage, die die codierten Informationen im IMG-Tag enthält: msgID und linkID für diese Nachricht und die Kundenparameter mit Benutzerkennung, Einkommensstufe und Einkommensart.
2. Diese Informationen werden über einen API-Aufruf an den Interact Laufzeitserver übergeben.
3. Der Laufzeitserver gibt ein Angebot an den Message Connector zurück, der die URL der Grafik für das Angebot abruft und diese URL (einschließlich aller zusätzlichen Parameter) bereitstellt, bevor er die Grafikanfrage an die URL für dieses Angebot weiterleitet.
4. Das Angebot wird dem Kunden als Grafik angezeigt.

An diesem Punkt kann der Kunde auf die Grafik klicken, um auf das Angebot zu reagieren. Diese Klickabfolge mit dem A-Tag und dem zugehörigen HREF-Attribut (das die Ziel-URL angibt) sendet eine weitere Anfrage an den Message Connector, um die Landing-Page abzurufen, die mit der URL für dieses Angebot verknüpft ist. Der Browser des Kunden wird dann auf die im Angebot konfigurierte Landing-Page umgeleitet.

Hinweis: Ein A-Tag ist für die Klickabfolge nicht notwendigerweise erforderlich. Das Angebot kann auch nur aus einem Bild bestehen, z. B. ein Coupon zum Ausdrucken durch den Kunden.

Installieren des Message Connectors

Die Dateien, die zum Installieren, Implementieren und Ausführen des Message Connectors erforderlich sind, wurden automatisch in der IBM Interact Installation des Laufzeitservers eingebunden. In diesem Abschnitt werden die Schritte zusammengefasst, die erforderlich sind, um den Message Connector zur Verfügung zu stellen.

Das Installieren und Implementieren des Message Connectors umfasst die folgenden Aufgaben:

- Optionales Konfigurieren der Standardeinstellungen für den Message Connector, wie in „Konfigurieren des Message Connectors" beschrieben.
- Erstellen der Datenbanktabellen, die zum Speichern der Transaktionsdaten im Message Connector erforderlich sind, wie in „Erstellen der Message Connector-Tabellen" auf Seite 319 beschrieben.
- Installieren der Message Connector-Webanwendung, wie in „Bereitstellen und Ausführen des Message Connectors" auf Seite 320 beschrieben.
- Erstellen der IMG- und A-Tags in den Werbeflächen (z. B. E-Mails oder Webseiten), die zum Aufrufen der Message Connector-Angebote beim Öffnen und Durchklicken erforderlich sind, wie in „Erstellen der Message Connector-Links" auf Seite 321 beschrieben.

Konfigurieren des Message Connectors

Bevor Sie den Message Connector bereitstellen können, müssen Sie die Konfigurationsdatei in Ihrer Installation an die jeweilige Umgebung anpassen. Dazu können Sie die XML-Datei MessageConnectorConfig.xml ändern, die sich im Message Connector-Verzeichnis auf dem Interact Laufzeitserver befindet, zum Beispiel <Interact_home>/msgconnector/config/MessageConnectorConfig.xml.

Informationen zu diesem Vorgang

Die Datei `MessageConnectorConfig.xml` enthält sowohl obligatorische als auch optionale Konfigurationseinstellungen. Alle Einstellungen, die Sie verwenden, müssen an Ihre spezifische Installation angepasst werden. Folgen Sie den hier dargestellten Arbeitsschritten, um die Konfiguration zu ändern.

Vorgehensweise

1. Wenn der Message Connector bereits aktiviert wurde und auf dem Webanwendungsserver ausgeführt wird, müssen Sie den Message Connector inaktivieren, bevor Sie den Vorgang fortsetzen.
2. Öffnen Sie auf dem Interact Laufzeitserver die Datei `MessageConnectorConfig.xml` in einem beliebigen Text- oder XML-Editor.
3. Nehmen Sie die erforderlichen Änderungen an den Konfigurationseinstellungen vor und vergewissern Sie sich, dass die folgenden *Pflichteinstellungen* für Ihre Installation korrekt sind.
   - `<interactUrl>`, die URL des Interact Laufzeitservers, auf dem der Message Connector ausgeführt wird und mit dem die Seite eine Verbindung herstellen soll.
   - `<imageErrorLink>`, die URL, auf die der Message Connector umleitet, wenn eine angeforderte Grafik für das Angebot nicht ordnungsgemäß geladen werden kann.
   - `<landingPageErrorLink>`, die URL, auf die der Message Connector umleitet, wenn eine angeforderte Landing-Page für das Angebot nicht ordnungsgemäß geladen werden kann.
   - `<audienceLevels>`, ein Abschnitt der Konfigurationsdatei, der eine oder mehrere Einstellungen für die Zielgruppenebene enthält und die Standardzielgruppenebene festlegt, sofern nicht im Link für den Message Connector angegeben. Mindestens eine Zielgruppenebene muss konfiguriert werden.

   Alle Konfigurationseinstellungen werden ausführlich unter „Konfigurationseinstellungen für den Message Connector" beschrieben.
4. Wenn Sie keine weiteren Konfigurationsänderungen vornehmen möchten, speichern und schließen Sie die Datei `MessageConnectorConfig.xml`.
5. Fahren Sie mit der Einrichtung und Bereitstellung des Message Connectors fort.

Konfigurationseinstellungen für den Message Connector:

Um den Message Connector zu konfigurieren, können Sie die XML-Datei mit dem Namen `MessageConnectorConfig.xml` im Message Connector-Verzeichnis auf dem Interact Laufzeitserver ändern, normalerweise `<Interact_home>/msgconnector/config/MessageConnectorConfig.xml`. Hier werden die einzelnen Konfigurationen in dieser XML-Datei beschrieben. Hinweis: Wenn Sie diese Datei ändern, nachdem der Message Connector implementiert und aktiv ist, müssen Sie die Bereitstellung für den Message Connector erneut ausführen oder den Anwendungsserver erneut starten, um diese Einstellungen erneut zu laden, nachdem Sie die Datei geändert haben.

Allgemeine Einstellungen

Die folgende Tabelle enthält eine Liste mit den optionalen und erforderlichen Einstellungen, die im Abschnitt `generalSettings` der Datei `MessageConnectorConfig.xml` enthalten sind.

Tabelle 24. Allgemeine Einstellungen für den Message Connector

Element	Beschreibung	Standardwert
`<interactURL>`	Die URL des Interact Laufzeitservers zum Bearbeiten der Aufrufe von Message Connector-Seitentags, zum Beispiel für den Laufzeitserver, auf dem der Message Connector ausgeführt wird. Dieses Element ist erforderlich.	`http://localhost:7001/interact`
`<defaultDateTimeFormat>`	Das Standarddatumsformat.	`dd.MM.yyyy`
`<log4jConfigFileLocation>`	Die Position der Eigenschaftendatei Log4j. Die Angabe ist relativ zur *$MESSAGE_CONNECTOR_HOME* Umgebungsvariable, sofern gesetzt. Andernfalls ist dieser Wert relativ zum Rootpfad der Message Connector-Webanwendung.	`config/ MessageConnectorLog4j.properties`

Standardparameterwerte

Die folgende Tabelle enthält eine Liste mit den optionalen und erforderlichen Einstellungen, die im Abschnitt `defaultParameterValues` der Datei `MessageConnectorConfig.xml` enthalten sind.

Tabelle 25. Standardparametereinstellungen für den Message Connector

Element	Beschreibung	Standardwert
`<interactiveChannel>`	Der Name des interaktiven Standardkanals.	
`<interactionPoint>`	Der Name des Standardinteraktionspunkts.	
`<debugFlag>`	Legt fest, ob das Debugging aktiviert ist. Die zulässigen Werte sind `true` und `false`.	`false`
`<contactEventName>`	Der Standardname des übergebenen Kontaktereignisses.	
`<acceptEventName>`	Der Standardname des übergebenen Annahmeereignisses.	
`<imageUrlAttribute>`	Der Standardname des Angebotsattributs mit der URL für die Angebotsgrafik, sofern nicht im Message Connector-Link angegeben.	
`<landingPageUrlAttribute>`	Die Standard-URL für die Klickabfolge zur Landing-Page, sofern nicht im Message Connector-Link angegeben.	

Verhaltenseinstellungen

Die folgende Tabelle enthält eine Liste mit den optionalen und erforderlichen Einstellungen, die im Abschnitt `behaviorSettings` der Datei `MessageConnectorConfig.xml` enthalten sind.

Tabelle 26. Verhaltenseinstellungen für den Message Connector

Element	Beschreibung	Standardwert
`<imageErrorLink>`	Die URL, an die der Connector umleitet, wenn beim Verarbeiten einer Angebotsgrafikanfrage ein Fehler auftritt. Diese Einstellung ist erforderlich.	`/images/default.jpg`
`<landingPageErrorLink>`	Die URL, an die der Connector umleitet, wenn beim Verarbeiten einer Landing-Page-Anfrage zur Klickabfolge ein Fehler auftritt. Diese Einstellung ist erforderlich.	`/jsp/default.jsp`
`<alwaysUseExistingOffer>`	Legt fest, ob das in den Cache gestellte Angebot zurückgegeben werden soll, obwohl es bereits abgelaufen ist. Die zulässigen Werte sind `true` und `false`.	`false`
`<offerExpireAction>`	Die Aktion, die durchgeführt werden soll, wenn das ursprüngliche Angebot gefunden wird, aber bereits abgelaufen ist. Zulässige Werte: • `GetNewOffer` • `RedirectToErrorPage` • `ReturnExpiredOffer`	`RedirectToErrorPage`

Speichereinstellungen

Die folgende Tabelle enthält eine Liste mit den optionalen und erforderlichen Einstellungen, die im Abschnitt `storageSettings` der Datei `MessageConnectorConfig.xml` enthalten sind.

Tabelle 27. Speichereinstellungen für Message Connector

Element	Beschreibung	Standardwert
`<persistenceMode>`	Wenn der Cache neue Einträge in die Datenbank stellt. Die zulässigen Werte sind `WRITE-BEHIND` (dabei werden die Daten zunächst in den Cache geschrieben und zu einem späteren Zeitpunkt in der Datenbank aktualisiert) und `WRITE-THROUGH` (dabei werden die Daten gleichzeitig in den Cache und in die Datenbank geschrieben).	`WRITE-THROUGH`
`<maxCacheSize>`	Die maximale Anzahl an Einträgen im Speichercache.	`5000`
`<maxPersistenceBatchSize>`	Die maximale Stapelgröße, während Einträge in die Datenbank gestellt werden.	`200`

Tabelle 27. Speichereinstellungen für Message Connector (Forts.)

Element	Beschreibung	Standardwert
<macCachePersistInterval>	Die maximale Zeit in Sekunden, wie lange ein Eintrag im Cache bleibt, bevor er in die Datenbank gestellt wird.	3
<maxElementOnDisk>	Die maximale Anzahl an Einträgen im Plattencache.	5000
<cacheEntryTimeToExpireInSeconds>	Die maximale Zeitdauer, wie lange die Einträge im Plattencache bleiben, bevor sie ablaufen.	60000
<jdbcSettings>	Ein Abschnitt der XML-Datei mit spezifischen Informationen, wenn eine JDBC-Verbindung verwendet wird. Dieser Abschnitt und der Abschnitt <dataSourceSettings> sind gegenseitig ausschließend.	Mit der Standardkonfiguration wird eine Verbindung mit der SQL Server-Datenbank hergestellt, die auf dem lokalen Server konfiguriert ist. Wenn Sie jedoch diesen Abschnitt aktivieren, müssen Sie die tatsächlichen JDBC-Einstellungen und die Berechtigungsnachweise für die Anmeldung angeben.
<dataSourceSettings>	Ein Abschnitt der XML-Datei mit spezifischen Informationen, wenn eine Datenquellenverbindung verwendet wird. Dieser Abschnitt und der Abschnitt <jdbcSettings> sind gegenseitig ausschließend.	Mit der Standardkonfiguration wird eine Verbindung mit der InteractDS-Datenquelle hergestellt, die auf dem lokalen Webanwendungsserver definiert ist.

Zielgruppenebenen

Die folgende Tabelle enthält eine Liste mit den optionalen und erforderlichen Einstellungen, die im Abschnitt audienceLevels der Datei MessageConnectorConfig.xml enthalten sind.

Hinweis: Das audienceLevels-Element wird optional verwendet, um die zu verwendende Standardzielgruppenebene anzugeben, sofern nicht im Message Connector-Link angegeben, wie im folgenden Beispiel dargestellt:

```
<audienceLevels default="Customer">
```

In diesem Beispiel stimmt der Wert für das Standardattribut mit dem Namen eines audienceLevel-Elements überein, das in diesem Abschnitt definiert ist. In dieser Konfigurationsdatei muss mindestens eine Zielgruppenebene definiert sein.

Tabelle 28. Zielgruppenebenen-Einstellungen für Message Connector

Element	Element	Beschreibung	Standardwert
<audienceLevel>		Das Element, das die Konfiguration der Zielgruppenebene enthält. Geben Sie ein Namensattribut an, zum Beispiel <audienceLevel name="Customer">	
	<messageLogTable>	Der Name der Protokolltabelle. Dieser Wert ist erforderlich.	UACI_MESSAGE_CONNECTOR_LOG

Tabelle 28. Zielgruppenebenen-Einstellungen für Message Connector (Forts.)

Element	Element	Beschreibung	Standardwert
<fields>	<field>	Die Definition eines oder mehrerer Felder mit der jeweiligen Zielgruppen-ID für audienceLevel.	
	<Name>	Der Name des Feldes mit der Zielgruppen-ID, wie in der Interact-Laufzeit angegeben.	
	<httpParameterName>	Der entsprechende Parametername für dieses Feld mit der Zielgruppen-ID.	
	<dbColumnName>	Der entsprechende Spaltenname in der Datenbank für dieses Feld mit der Zielgruppen-ID.	
	<type>	Der Typ des Feldes mit der Zielgruppen-ID, wie in der Interact-Laufzeit angegeben. Zulässige Werte sind string oder numeric.	

Erstellen der Message Connector-Tabellen

Bevor Sie den IBM Interact Message Connector bereitstellen können, müssen Sie zunächst die Tabellen in der Datenbank erstellen, in der die Interact Laufzeitdaten gespeichert werden. Sie müssen für jede definierte Zielgruppenebene jeweils eine Tabelle erstellen. Interact verwendet die erstellten Tabellen, um für jede Zielgruppenebene die Informationen zu den Transaktionen in Message Connector aufzuzeichnen.

Informationen zu diesem Vorgang

Führen Sie mit Ihrem Datenbankclient im Message Connector das SQL-Script für die entsprechende Datenbank oder das entsprechende Schema aus, um die erforderlichen Tabellen zu erstellen. Die SQL-Scripts für die unterstützte Datenbank werden automatisch installiert, wenn Sie den Interact Laufzeitserver installieren. Weitere Informationen zum Herstellen einer Verbindung mit der Datenbank, die die Interact Laufzeittabellen enthält, finden Sie in den Arbeitsblättern, die Sie im *IBM Interact Installationshandbuch* ausgefüllt haben.

Vorgehensweise

1. Starten Sie den Datenbankclient und stellen Sie eine Verbindung mit der Datenbank her, in der gegenwärtig die Interact Laufzeittabellen gespeichert werden.
2. Führen Sie das entsprechende Script im Verzeichnis <*Interact_home*>/msgconnector/scripts/ddl aus. In der folgenden Tabelle sind die SQL-Beispielscripts aufgeführt, die Sie zum manuellen Erstellen der Message Connector-Tabellen verwenden können:

Tabelle 29. Scripts zum Erstellen von Message Connector-Tabellen

Datenquellentyp	Scriptname
IBM DB2	db_scheme_db2.sql
Microsoft SQL Server	db_scheme_sqlserver.sql
Oracle	db_scheme_oracle.sql

Beachten Sie, dass diese Scripts als Muster bereitgestellt werden. Wenn Sie andere Namenskonventionen oder Strukturen für die Werte der Zielgruppen-ID verwenden, müssen Sie das Script ändern, bevor Sie es ausführen. Im Allgemeinen hat sich das Verfahren bewährt, jeweils eine Tabelle pro Zielgruppenebene zuzuordnen.

Die erstellten Tabellen müssen die folgenden Informationen enthalten:

Tabelle 30. Informationen, die von SQL-Beispielscripts erstellt werden

Spaltenname	Beschreibung
LogId	Der Primärschlüssel dieses Eintrags.
MessageId	Die eindeutige ID jeder Instanz für die Nachrichtenübertragung.
LinkId	Die eindeutige ID für jeden Link in den elektronischen Medien (z. B. E-Mail-Nachricht).
OfferImageUrl	Die URL für die zugehörige Grafik des zurückgegebenen Angebots.
OfferLandingPageUrl	Die URL der zugehörigen Landing-Page des zurückgegebenen Angebots.
TreatmentCode	Der Verfahrenscode des zurückgegebenen Angebots.
OfferExpirationDate	Ablaufdatum und Uhrzeit des zurückgegebenen Angebots.
OfferContactDate	Datum und Uhrzeit, wann das Angebot an den Kunden zurückgegeben wurde.
AudienceId	Die Zielgruppen-ID der elektronischen Medien.

Beachten Sie folgende Hinweise zur dieser Tabelle:

- Abhängig von der Zielgruppenebene gibt es eine AudienceId-Spalte für jede Komponente des Zielgruppenschlüssels.
- Die Kombination aus MessageId, LinkId und AudienceId(s) bildet einen eindeutigen Schlüssel dieser Tabelle.

Wenn das Script erfolgreich ausgeführt wurde, haben Sie die erforderlichen Tabellen für den Message Connector erstellt.

Ergebnisse

Sie können jetzt die Webanwendung für den Message Connector bereitstellen.

Bereitstellen und Ausführen des Message Connectors

Der IBM Interact Message Connector wird als eigenständige Webanwendung auf einem unterstützten Webanwendungsserver implementiert.

Vorbereitende Schritte

Stellen Sie vor der Implementierung des Message Connectors sicher, dass die folgenden Aufgaben abgeschlossen sind:

- Sie müssen den IBM Interact Laufzeitserver installiert haben. Die implementierbare Message Connector-Anwendung wird automatisch zusammen mit dem Laufzeitserver installiert und kann im Interact Ausgangsverzeichnis bereitgestellt werden.
- Sie müssen auch die SQL-Scripts ausgeführt haben, die mit der Installation bereitgestellt wurden, um die erforderlichen Tabellen in der Interact Laufzeitdatenbank zu erstellen, die der Message Connector verwendet, wie in „Erstellen der Message Connector-Tabellen" auf Seite 319 beschrieben

Informationen zu diesem Vorgang

Sie müssen zunächst die Message Connector-Anwendung bereitstellen und verfügbar machen wie alle anderenIBM Anwendungen, die auf einem Webanwendungsserver ausgeführt werden sollen.

Vorgehensweise

1. Stellen Sie mit den erforderlichen Berechtigungen eine Verbindung mit der Managementbenutzeroberfläche für den Webanwendungsserver her, um eine Anwendung bereitzustellen.
2. Folgen Sie den Anweisungen zum Bereitstellen des Webanwendungsservers und führen Sie die Datei *<Interact_home>*/msgconnector/MessageConnector.war aus Ersetzen Sie *<Interact_home>* durch das tatsächliche Verzeichnis, in dem der Interact-Laufzeitserver installiert ist.

Ergebnisse

Der Message Connector kann jetzt verwendet werden. Nachdem Sie die Interact Installation zum Erstellen der erforderlichen Daten konfiguriert haben, die der Message Connector verwendet, um Angebote zu unterbreiten, z. B. interaktive Kanäle, Strategien, Ablaufdiagramme, Angebote und so weiter, können Sie in den elektronischen Medien die Links erstellen, die der Message Connector annimmt.

Erstellen der Message Connector-Links

Sie müssen entsprechende Links erstellen und in Ihre Nachricht einbinden, wenn Sie den Message Connector verwenden möchten, um benutzerdefinierte Angebotsgrafiken und Landing-Pages bereitzustellen, wenn ein Endbenutzer mit den elektronischen Medien interagiert (z. B. durch Öffnen einer E-Mail-Nachricht) und sich durch das Angebot klickt. Dieser Abschnitt enthält eine Übersicht über die HTML-Tags, die Sie für diese Links benötigen.

Informationen zu diesem Vorgang

Unabhängig davon, welches System Sie zum Generieren von abgehenden Nachrichten an die Endbenutzer verwenden, müssen Sie das HTML-Tagging generieren, das die entsprechenden Felder enthält (die in den HTML-Tags als Attribute angegeben werden), in denen die Informationen enthalten sind, die an den Interact-Laufzeitserver übergeben werden sollen. Folgen Sie den Anweisungen unten, um die Minimalanforderungen zu konfigurieren, die für eine Nachricht im Message Connector erforderlich sind.

Hinwies: Obwohl sich die Anweisungen hier speziell auf Nachrichten beziehen, die Message Connector-Links enthalten, können Sie dieses Verfahren auch zur Konfiguration verwenden, um Webseiten oder beliebigen anderen elektronischen Medien Links hinzuzufügen.

Vorgehensweise

1. Sie müssen mindestens die folgenden Parameter verwenden, um den IMG-Link zu erstellen, der in der Nachricht angezeigt werden soll:
   - msgID, um die eindeutige ID für diese Nachricht anzugeben.
   - linkID, um die eindeutige ID für den Link in der Nachricht anzugeben.
   - audienceID, um die Kennung der Zielgruppe anzugeben, zu der der Empfänger der Nachricht gehört.

Hinweis: Wenn sich die Zielgruppen-ID aus mehreren ID-Komponenten zusammensetzt, müssen alle Komponenten im Link enthalten sein.

Sie können auch optionale Parameter einbeziehen, um die Zielgruppenebene, den Namen des interaktiven Kanals, den Namen des Interaktionspunkts, die URL mit der Position der Grafik und eigene benutzerdefinierte Parameter anzugeben, die nicht speziell vom Message Connector verwendet werden.

2. Optional können Sie auch einen A-Link erstellen, der den IMG-Link einschließt, damit der Benutzer auf das Bild klicken kann, um die entsprechende Seite mit dem Angebot im Browser zu laden. Der A-Link muss auch die drei oben genannten Parameter (msgID, linkID und audienceID) und alle optionalen Parameter (Zielgruppenebene, Name des interaktiven Kanals und Name des Interaktionspunkts) und benutzerdefinierten Parameter enthalten, die nicht speziell vom Message Connector verwendet werden. Hinweis: Der A-Link kann bei Bedarf auch eigenständig auf der Seite verwendet werden, obwohl er im Message Connector in den meisten Fällen auch einen IMG-Link enthält. Wenn der Link einen IMG-Link enthält, sollte der IMG-Link den gleichen Parametersatz enthalten wie der umschließende A-Link (einschließlich aller optionalen oder benutzerdefinierten Parameter).
3. Wenn alle Links korrekt definiert sind, können Sie die E-Mail-Nachrichten generieren und senden.

Ergebnisse

Beispiel-Links und weitere Informationen zu den verfügbaren Parametern finden Sie unter „HTTP-Anforderungsparameter für die Tags "IMG" und "A"”

HTTP-Anforderungsparameter für die Tags "IMG" und "A"

Wenn der Message Connector eine Anfrage empfängt, weil ein Endbenutzer auf ein A-Tag geklickt oder eine E-Mail geöffnet hat, die einen Message Connector-codierten IMG-Tag enthält, werden die in der Anfrage enthaltenen Parameter geparst, um die entsprechenden Angebotsdaten zurückzugeben. Dieser Abschnitt enthält eine Liste der Parameter, die in der anfordernden URL enthalten sein können (entweder im IMG-Tag, das automatisch geladen wird, wenn ein in Tags eingeschlossenes Bild aufgerufen oder die E-Mail geöffnet wird, oder im A-Tag, das geladen wird, wenn sich der Benutzer durch die E-Mail-Nachricht zur angegebenen Website klickt).

Parameter

Wenn der Message Connector eine Anfrage empfängt, werden die in der Anfrage enthaltenen Parameter geparst. Diese Parameter können vollständig oder teilweise Folgendes enthalten:

Parametername	Beschreibung	Erforderlich?	Standardwert
msgId	Die eindeutige ID der E-Mail-Instanz oder Webseite.	Ja	Ohne. Dies wird von dem System bereitgestellt, das die eindeutige Instanz der E-Mail-Nachricht oder der Webseite erstellt, die den Tag enthält.
linkId	Die eindeutige ID des Links in dieser E-Mail oder Webseite.	Ja	Ohne. Dies wird von dem System bereitgestellt, das die eindeutige Instanz der E-Mail-Nachricht oder der Webseite erstellt, die den Tag enthält.
audienceLevel	Die Zielgruppenebene, zu der der Empfänger dieser Kommunikation gehört.	Nein	Der audienceLevel-Standardwert, der im audienceLevels-Element der Datei MessageConnectorConfig.xml angegeben ist.
ic	Der Name des interaktiven Zielkanals (IC)	Nein	Der Wert des interactiveChannel-Elements im Abschnitt defaultParameterValues der Datei MessageConnectorConfig.xml. Der Standardwert lautet "interactiveChannel".

Parametername	Beschreibung	Erforderlich?	Standardwert
ip	Der Name des Interaktionspunkts (IP)	Nein	Der Wert des interactionPoint-Elements im Abschnitt defaultParameterValues der Datei MessageConnectorConfig.xml. Der Standardwert lautet "headBanner".
offerImageUrl	Die URL der Zielangebotsgrafik für die IMG-URL in der Nachricht.	Nein	Keiner.
offerImageUrlAttr	Der Name des Angebotsattributs mit der URL der Zielangebotsgrafik	Nein	Der Wert des imageUrlAttribute-Elements im Abschnitt defaultParameterValues der Datei MessageConnectorConfig.xml.
offerLandingPageUrl	Die URL der Landing-Page für das Zielangebot.	Nein	Keiner.
offerLandingPageUrlAttr	Der Name des Zielattributs mit der URL der Landing-Page für das Zielangebot.	Nein	Der Wert des landingPageUrlAttribute-Elements im Abschnitt defaultParameterValues der Datei MessageConnectorConfig.xml.
contactEvent	Der Name des Kontaktereignisses.	Nein	Der Wert des contactEventName-Elements im Abschnitt defaultParameterValues der Datei MessageConnectorConfig.xml. Der Standardwert lautet "contact".
responseEvent	Der Name des Annahmeereignisses.	Nein	Der Wert des acceptEventName-Elements im Abschnitt defaultParameterValues der Datei MessageConnectorConfig.xml. Der Standardwert lautet "accept".
debug	Das Debugflag. Setzen Sie diesen Parameter nur zur Fehlerbehebung und auf Anweisung durch den IBM technischen Support auf "true".	Nein	Der Wert des debugFlag-Elements im Abschnitt defaultParameterValues der Datei MessageConnectorConfig.xml. Der Standardwert lautet "false".
<audience id>	Die Zielgruppen-ID dieses Benutzers. Der Name dieses Parameters ist in der Konfigurationsdatei definiert.	Ja	Ohne.

Wenn der Message Connector einen unbekannten Parameter empfängt (das heißt einen Parameter, der nicht in obiger Liste enthalten ist), gibt es zwei Möglichkeiten:

- Wenn ein unbekannter Parameter angegeben ist (z. B. "attribute", wie in attribute="attrValue") und ein übereinstimmender Parameter mit dem gleichen Namen und dem Wortzusatz "Type" vorhanden ist (z. B. "attributeType", wie in attributeType="string"), erstellt der Message Connector einen übereinstimmenden Interact Parameter und übergibt diesen der Interact Laufzeit.

  Für den Typparameter sind folgende Werte zulässig:

  – Zeichenfolge
  – Zahl
  – Datum/Uhrzeit

    Für einen Typparameter "datetime" sucht der Message Connector auch nach einem Parameter mit dem gleichen Namen und dem Wortzusatz "Pattern" (z. B. "attributePattern"), falls ein gültiger Wert im Format für Datum/Uhrzeit vorhanden ist. Sie können zum Beispiel den Parameter attributePattern="MM/dd/yyyy" angeben.

    Hinweis: Wenn Sie den Parametertyp "datetime" angeben, ohne ein übereinstimmendes Datumsmuster anzugeben, wird der Wert verwendet, der in der Message Connector-Konfigurationsdatei (in <installation_directory>/msgconnector/config/MessageConnectorConfig.xml) auf dem Interact Server angegeben ist.

- Wenn ein unbekannter Parameter angegeben wird und kein übereinstimmender Typwert vorhanden ist, übergibt der Message Connector diesen Parameter an die URL für die Zielweiterleitung.

Der Message Connector übergibt alle unbekannten Parameter an den Interact Laufzeitserver, ohne die Parameter zu verarbeiten oder zu speichern.

Beispielcode für den Message Connector

Der folgende A-Tag enthält ein Beispiel mit einem Satz von Message Connector-Links, die in einer E-Mail-Nachricht enthalten sein können:

```
<a href="http://www.example.com/MessageConnector/offerClickthru.jsp?msgId=234
   &linkId=1&userid=1&referral=xyz">
 <img src="http://www.example.com/MessageConnector/offerImage.jsp?msgId=234&linkId=1
      &userid=1&incomeCode=3&incomeType=numeric"/>
</a>
```

In diesem Beispiel wird der IMG-Tag beim Öffnen der E-Mail-Nachricht automatisch geladen. Indem die Grafik aus der angegebenen Seite abgerufen wird, übergibt die Nachricht die Parameter für die eindeutige Nachrichten-ID (msgID), die eindeutige Link-ID (linkID) und die eindeutige Benutzer-ID (userid) zusammen mit den beiden zusätzlichen Parametern (incomeCode und incomeType) an die Interact Laufzeit.

Der A-Tag stellt das HREF-Attribut (Hypertext Reference) bereit, das die Angebotsgrafik in einen Link zum Anklicken in der E-Mail-Nachricht umwandelt. Wenn der Endbenutzer die Grafik in der Nachricht anzeigt und sich dann zur Landing-Page durchklickt, werden die eindeutige Nachrichten-ID (`msgId`), die eindeutige Link-ID (`linkId`) und die eindeutige Benutzer-ID (`userid`) jeweils zusammen mit allen zusätzlichen Parametern (`referral`) der URL für die Zielweiterleitung an den Server übergeben.

Informationen zum Interact Web Connector

Der Interact WebConnector (wird auch als JavaScript Connector oder JSConnector bezeichnet) bietet einen Service auf dem Interact Laufzeitserver, mit dem JavaScript-Code die Interact Java-API aufrufen kann. Auf diese Weise können Webseiten die Interact Echtzeit-Personalisierung von Angeboten aufrufen, indem nur der eingebettete JavaScript-Code verwendet wird, ohne sich auf Web-Entwicklungssprachen (wie z. B. Java, PHP, JSP und so weiter) verlassen zu müssen. So können Sie zum Beispiel ein kleines Snippet mit JavaScript-Code auf jeder Seite Ihrer Website integrieren, um die von Interact empfohlenen Angebote bereitzustellen. Dadurch wird bei jedem Seitenaufruf die Interact API aufgerufen, um sicherzustellen, dass auf der geladenen Seite stets die besten Angebote für den Besucher der Website angezeigt werden.

Verwenden Sie den Interact Web Connector in Situationen, in denen Sie den Besuchern auf einer Seite Angebote anzeigen möchten, ohne die serverseitige Anzeige der Seite programmatisch steuern zu können (wie das zum Beispiel mit PHP oder einem anderen serverbasierten Scripting der Fall wäre). Dazu können Sie im Seiteninhalt JavaScript-Code integrieren, der vom Web-Browser des Besuchers ausgeführt wird.

Tipp: Die Dateien für den Interact Web Connector werden auf dem Interact Laufzeitserver automatisch im Verzeichnis `<Interact_home>/jsconnector` installiert. Das Verzeichnis `<Interact_home>/jsconnector` enthält die Datei ReadMe.txt mit wichti-

gen Informationen und Hinweisen zu den Funktionen des Web Connectors. Hier finden Sie auch Beispieldateien und den Quellcode des Web Connectors als Basis zur Entwicklung eigener Lösungen. Weitere Informationen finden Sie außerdem auch im Verzeichnis jsconnector.

Installieren des Web Connectors auf dem Laufzeitserver

Eine Instanz des Web Connectors wird automatisch mit dem IBM Interact Laufzeitserver installiert und standardmäßig aktiviert. Sie müssen jedoch einige Einstellungen ändern, bevor Sie den Web Connector konfigurieren und verwenden können.

Informationen zu diesem Vorgang

Die Einstellungen, die geändert werden müssen, bevor Sie den auf dem Laufzeitserver installierten Web Connector verwenden können, werden der Konfiguration des Webanwendungsservers hinzugefügt. Aus diesem Grund muss der Webanwendungsserver erneut gestartet werden, nachdem Sie diese Schritte abgeschlossen haben.

Vorgehensweise

1. Legen Sie für den Webanwendungsserver, auf dem der Interact Laufzeitserver installiert ist, die folgenden Java-Eigenschaften fest:

   `-DUI_JSCONNECTOR_ENABLE_INPROCESS=true`

   `-DUI_JSCONNECTOR_HOME=<`*`jsconnectorHome`*`>`

   Ersetzen Sie <*jsconnectorHome*> durch den Pfad zum Verzeichnis jsconnector auf dem Laufzeitserver. Dieser lautet <Interact_Home>/jsconnector.

   Die Art und Weise, wie die Java-Eigenschaften festgelegt werden, hängt von Ihrem Webanwendungsserver ab. Beispiel: In WebLogic bearbeiten Sie die Datei startWebLogic.sh oder startWebLogic.cmd, um die Einstellung JAVA_OPTIONS zu aktualisieren, wie nachfolgend dargestellt:

   ```
   JAVA_OPTIONS="${SAVE_JAVA_OPTIONS} -DUI_JSCONNECTOR_HOME=/UnicaFiles/
   jsconnector"
   ```

   Im WebSphere Application Server legen Sie diese Eigenschaft über die Administrationskonsole im Fenster der Java Virtual Machine fest.

   Weitere Informationen zum Einstellen spezifischer Java-Eigenschaften finden Sie in der Dokumentation des Webanwendungsservers.

2. Der Webanwendungsserver muss jetzt neu gestartet werden, damit die neuen Java-Eigenschaften übernommen werden.

Ergebnisse

Mit dem erfolgreichen Neustart des Webanwendungsservers ist die Installation des Web Connectors auf dem Laufzeitserver abgeschlossen. Als nächstes muss eine Verbindung mit der Webseite zur Konfiguration unter http://<*host*>:<*port*>/interact/jsp/WebConnector.jsp hergestellt werden, wobei <*host*> den Interact Namen des Laufzeitservers und <*port*> den Port angibt, den der Web Connector für das Listening verwendet, wie im Webanwendungsserver angegeben.

Installieren des Web Connectors als separate Webanwendung

Eine Instanz des Web Connectors wird automatisch mit dem IBM Interact Laufzeitserver installiert und standardmäßig aktiviert. Sie können den Web Connector jedoch auch als eigenständige Webanwendung implementieren (zum Beispiel in einem Webanwendungsserver auf einem separaten System) und für die Kommunikation mit dem fernen Interact Laufzeitserver konfigurieren.

Informationen zu diesem Vorgang

Diese Anleitung beschreibt, wie Sie den Web Connector als separate Webanwendung mit Zugriff auf einen fernen Interact Laufzeitserver implementieren.

Bevor Sie den Web Connector implementieren können, müssen Sie den IBM Interact Laufzeitserver installiert haben. Außerdem benötigen Sie einen Webanwendungsserver auf einem anderen System mit Netzzugang (ohne Sperre durch eine Firewall) zum Interact Laufzeitserver.

Vorgehensweise

1. Kopieren Sie das Verzeichnis `jsconnector` mit den Web Connector-Dateien vom Interact Laufzeitserver in das System, auf dem bereits der konfigurierte Webanwendungsserver (z. B. WebSphere Application Server) ausgeführt wird. Das Verzeichnis `jsconnector` befindet sich in Ihrem Interact-Installationsverzeichnis.
2. Konfigurieren Sie auf dem System, auf dem Sie die Web Connector-Instanz implementieren, die Datei `jsconnector/jsconnector.xml` in einem beliebigen Text- oder XML-Editor, um das Attribut `interactURL` zu ändern.

   Der Standardwert lautet `http://localhost:7001/interact`. Sie müssen den Wert ändern, damit er mit der URL des fernen Interact Laufzeitservers übereinstimmt, z. B. `http://runtime.example.com:7011/interact`.

   Nachdem Sie den Web Connector implementiert haben, können Sie eine Webbenutzeroberfläche verwenden, um die weiteren Einstellungen in der Datei `jsconnector.xml` anzupassen. Weitere Informationen finden Sie unter „Konfigurieren des Web Connectors" auf Seite 327.
3. Legen Sie für den Webanwendungsserver, auf dem Sie den Web Connector implementieren, die folgende Java-Eigenschaft fest:

   `-DUI_JSCONNECTOR_HOME=<`*`jsconnectorHome`*`>`

   Ersetzen Sie <*jsconnectorHome*> durch den tatsächlichen Pfad, in den Sie das Verzeichnis `jsconnector` auf dem Webanwendungsserver kopiert haben.

   Die Art und Weise, wie die Java-Eigenschaften festgelegt werden, hängt von Ihrem Webanwendungsserver ab. Beispiel: In WebLogic bearbeiten Sie die Datei `startWebLogic.sh` oder `startWebLogic.cmd`, um die Einstellung `JAVA_OPTIONS` zu aktualisieren, wie nachfolgend dargestellt:

   ```
   JAVA_OPTIONS="${SAVE_JAVA_OPTIONS} -DUI_JSCONNECTOR_HOME=/InteractFiles/
   jsconnector"
   ```

   Im WebSphere Application Server legen Sie diese Eigenschaft über die Administrationskonsole im Fenster der Java Virtual Machine fest.

   Weitere Informationen zum Einstellen spezifischer Java-Eigenschaften finden Sie in der Dokumentation des Webanwendungsservers.
4. Der Webanwendungsserver muss jetzt neu gestartet werden, damit die neue Java-Eigenschaft übernommen wird.

   Warten Sie, bis der Startvorgang des Webanwendungsservers abgeschlossen ist, bevor Sie den Vorgang fortsetzen.
5. Stellen Sie mit den erforderlichen Berechtigungen eine Verbindung mit der Managementbenutzeroberfläche für den Webanwendungsserver her, um eine Anwendung bereitzustellen.
6. Folgen Sie den Anweisungen zum Bereitstellen des Webanwendungsservers und rufen Sie die folgende Datei auf: `jsConnector/jsConnector.war`

Ergebnisse

Der Web Connector wird jetzt in der Webanwendung implementiert. Wenn der vollständig konfigurierte und erneut gestartete Interact Server ausgeführt wird, müssen Sie als Nächstes eine Verbindung mit der Webseite zur Konfiguration des Web Connectors unter `http://` *<host>*`:` *<port>*`/interact/jsp/WebConnector.jsp` herstellen. Dabei bezeichnet *<host>* das System, auf dem der Webanwendungsserver ausgeführt wird, auf dem Sie den Web Connector gerade implementiert haben, und *<port>* bezeichnet den Port, den der Web Connector für das Listening verwendet, wie im Webanwendungsserver angegeben.

Konfigurieren des Web Connectors

Die Konfigurationseinstellungen für den Interact Web Connector werden in der Datei `jsconnector.xml` auf dem System gespeichert, auf dem der Web Connector bereitgestellt wird (dies kann der Interact Laufzeitserver selbst oder ein separates System sein, auf dem ein Webanwendungsserver ausgeführt wird). Sie können die Datei `jsconnector.xml` direkt bearbeiten, indem Sie einen beliebigen Texteditor oder XML-Editor verwenden. Die meisten Konfigurationseinstellungen können jedoch auch erheblich leichter konfiguriert werden, indem Sie einfach die Seite zur Konfiguration des Web Connectors im Web-Browser aufrufen.

Vorbereitende Schritte

Bevor Sie die Webbenutzeroberfläche zum Konfigurieren des Web Connectors verwenden können, müssen Sie zunächst die Webanwendung installieren und implementieren, die den Web Connector bereitstellt. Auf dem Interact Laufzeitserver wird automatisch eine Instanz des Web Connectors installiert, wenn Sie Interact installieren und implementieren. Auf jedem anderen Webanwendungsserver müssen Sie die Web Connector-Webanwendung installieren und implementieren, wie in „Installieren des Web Connectors als separate Webanwendung" auf Seite 325 beschrieben.

Vorgehensweise

1. Öffnen Sie Ihren unterstützten Web-Browser und öffnen Sie eine URL, die der folgenden URL ähnlich ist:

   `http://`*<host>*`:`*<port>*`/interact/jsp/WebConnector.jsp`

   - Ersetzen Sie *<host>* durch den Server, auf dem der Web Connector ausgeführt wird, z. B. durch den Hostnamen des Laufzeitservers oder den Namen des Servers, auf dem Sie eine separate Instanz des Web Connectors bereitgestellt haben.
   - Ersetzen Sie *<port>* durch die Portnummer, die der Web Connector für das Listening verwendet, um Verbindungen mit der Webanwendung herzustellen. Normalerweise kann der voreingestellte Standardport der Webanwendung übernommen werden.
2. Füllen Sie auf der anschließend angezeigten Seite "Konfiguration" die folgenden Abschnitte aus:

Tabelle 31. Zusammenfassung der Konfigurationseinstellungen für den Web Connector.

Abschnitt	Einstellungen
Grundlegende Einstellungen	Auf der Seite mit den Basiseinstellungen können Sie das gesamte Verhalten des Web Connectors für die Website konfigurieren, auf der die in Tags eingeschlossenen Seiten aufgerufen werden sollen. Diese Einstellungen beinhalten die Basis-URL für die Website und Informationen über die Besucher, die Interact verwenden, und ähnliche Einstellungen, die sich auf alle Seiten beziehen, die Sie mit dem Web Connector-Code in Tags einschließen möchten. Weitere Informationen finden Sie unter „Basisoptionen zur Konfiguration des WebConnectors" auf Seite 329.
HTML-Anzeigetypen	Verwenden Sie die Seite HTML-Anzeigetypen, um den HTML-Code zu bestimmen, der für jeden Interaktionspunkt auf der Seite zur Verfügung gestellt wird. Sie können aus der Liste mit Standardvorlagen (FLT-Dateien) auswählen, die eine Kombination aus CSS-Code für Cascading Style Sheets, HTML-Code und JavaScript-Code für jeden Interaktionspunkt enthalten. Sie können die zur Verfügung gestellten Vorlagen verwenden und bei Bedarf anpassen oder eigene Vorlagen erstellen. Die Konfigurationseinstellungen auf dieser Seite beziehen sich auf den Abschnitt `interactionPoints` der Konfigurationsdatei `jsconnector.xml`. Weitere Informationen finden Sie unter „HTML-Anzeigetypen zur Konfiguration von WebConnector" auf Seite 331.
Erweiterte Seiten	Auf der Seite 'Erweiterte Seiten' können Sie einem URL-Muster seitenspezifische Einstellungen zuweisen. Sie können zum Beispiel eine Seitenzuordnung einrichten, um für jede URL, die den Text "index.htm" enthält, die allgemeine Begrüßungsseite anzuzeigen, die spezielle Ereignisse zum Laden der Seite und definierte Interaktionspunkte für diese Zuordnung enthält. Die Konfigurationseinstellungen auf dieser Seite beziehen sich auf den Abschnitt `pageMapping` der Konfigurationsdatei `jsconnector.xml`. Weitere Informationen finden Sie unter „Erweiterte Seiten zur Konfiguration von WebConnector" auf Seite 334.

3. Stellen Sie auf der Seite Basiseinstellungen sicher, dass die Website-weiten Einstellungen für die Installation gültig sind, und legen Sie optional den Debugmodus fest (nur empfohlen, wenn Sie die Fehlersuche zu einem Problem durchführen). Überprüfen Sie auch die Digital Analytics for On Premises Integration der Seiten-Tags und geben Sie die Standardeinstellungen für die Interaktionspunkte an, bevor Sie unter Konfigurationen auf den Link HTML-Anzeigetypen klicken.
4. Befolgen Sie diese Schritte auf der Seite HTML-Anzeigetypen, um für die Anzeige Vorlagen hinzuzufügen oder zu ändern, die die Interaktionspunkte auf der Webseite des Kunden definieren.

   Vorlagen für die Anzeige (FLT-Dateien) werden standardmäßig in `<jsconnector_home>/conf/html` gespeichert.

   a. Wählen Sie in der Liste die FLT-Datei aus, die Sie überprüfen oder als Ausgangspunkt verwenden möchten, oder klicken Sie auf 'Typ hinzufügen', um eine neue, leere Vorlage für den Interaktionspunkt zu erstellen und zu verwenden.

      Neben der Vorlagenliste werden weitere Informationen zum Inhalt der Vorlage angezeigt, falls vorhanden.

b. Optional können Sie den Namen der Vorlage im Feld **Dateiname für diesen Anzeigetyp** ändern. Für eine neue Vorlage ändern Sie `CHANGE_ME.flt`, indem Sie einen beschreibenden Namen eingeben.

   Wenn Sie den Namen der Vorlage hier umbenennen, erstellt der Web Connector eine neue Datei mit diesem Namen, sobald Sie die Vorlage speichern. Vorlagen werden gespeichert, wenn Sie den Hauptteil des Textes ändern und dann zu einem anderen Feld navigieren.

c. Ändern oder vervollständigen Sie bei Bedarf die Informationen im HTML-Snippet, indem Sie CSS-Code für Style-Sheets, JavaScript und HTML-Code eingeben. Hinweis: Sie können auch Variablen eingeben, die während der Laufzeit durch Interact Parameter ersetzt werden. Beispiel: `${offer.HighlightTitle}` wird an der angegebenen Position automatisch durch den Angebotstitel des Interaktionspunkts ersetzt.

   Verwenden Sie die Beispiele, die unter dem Feld HTML-Snippet angezeigt werden, um weitere Informationen zum Formatieren von CSS, JavaScript oder HTML-Codeblöcken zu erhalten.

5. Verwenden Sie bei Bedarf die Seite 'Erweiterte Seiten', um die Seitenzuordnungen einzurichten und festzulegen, wie mit bestimmten URL-Mustern auf den Seiten umgegangen werden soll.
6. Wenn Sie keine weiteren Konfigurationseigenschaften einrichten möchten, klicken Sie auf **Rollout der Änderungen durchführen**. Wenn Sie auf **Rollout der Änderungen durchführen** klicken, werden die folgenden Aktionen ausgeführt:
   - Zeigt den IBM Interact Web Connector-Seitentag an, der den JavaScript-Code enthält, den Sie auf der Web Connector-Seite kopieren und in den Webseiten einfügen können.
   - Sichert die vorhandene Web Connector-Konfigurationsdatei auf dem Interact Server (die Datei `jsconnector.xml` auf dem Server, auf dem der Web Connector installiert ist) und erstellt eine neue Konfigurationsdatei mit den von Ihnen definierten Einstellungen.

     Sicherungskonfigurationsdateien werden in `<jsconnector_home>/conf/archive/jsconnector.xml.<date>.<time>` als `jsconnector.xml.20111113.214933.750-0500` gespeichert (dabei entspricht 20111113 der Zeichenfolge für das Datum und 214933.750-0500 der Zeichenfolge für die Uhrzeit einschließlich der Zeitzone)

Ergebnisse

Sie haben jetzt die Konfiguration des Web Connectors abgeschlossen.

Um die Konfiguration zu ändern, können Sie entweder alle oben genannten Schritte erneut ausführen, um neue Werte einzugeben, oder die vorhandene Konfigurationsdatei(`<Interact_home>/jsconnector/conf/jsconnector.xml`) in einem beliebigen Text- oder XML-Editor öffnen, um die vorhandenen Werte zu ändern.

Basisoptionen zur Konfiguration des WebConnectors

Verwenden Sie die Seite Basiseinstellungen der Seite zur Konfiguration des Web Connectors, um das Web Connector-Verhalten für die gesamte Website zu konfigurieren, auf der die in Tags eingeschlossenen Seiten aufgerufen werden sollen. Diese Einstellungen beinhalten die Basis-URL für die Website und Informationen über die Besucher, die Interact verwenden, und ähnliche Einstellungen, die sich auf alle Seiten beziehen, die Sie mit dem Web Connector-Code in Tags einschließen möchten.

Website-weite Einstellungen

Die Website-weiten Einstellungen enthalten Konfigurationsoptionen für die globalen Einstellungen, die das Verhalten der gesamten Installation des konfigurierten Web Connectors beeinflussen. Sie können die folgenden Werte angeben:

Tabelle 32. Website-weite Einstellungen für die Web Connector-Installation

Einstellung	Beschreibung	Funktional entsprechende Einstellung in jsconnector.xml
Interact API-URL	Die Basis-URL des Interact Laufzeitservers. **Anmerkung:** Diese Einstellung wird nur verwendet, wenn der Web Connector nicht innerhalb des Interact-Laufzeitservers ausgeführt wird (somit separat implementiert wurde).	*<interactURL>*
Web Connector-URL	Die Basis-URL, die zum Generieren der URL für die Klickabfolge verwendet wird.	*<jsConnectorURL>*
Name des interaktiven Kanals für die Ziel-Website	Der Name des interaktiven Kanals, den Sie auf dem Interact Server definiert haben, der diese Seitenzuordnung darstellt.	*<interactiveChannel>*
Zielgruppenebene der Besucher	Die Campaign Zielgruppenebene für die ankommenden Besucher; wird im API-Aufruf für die Interact Laufzeit verwendet.	*<audienceLevel>*
Feldname der Zielgruppen-ID in der Profiltabelle	Name des audienceId-Feldes, das im API-Aufruf für Interact verwendet wird. Hinweis: Zielgruppen-IDs für mehrere Felder werden gegenwärtig nicht unterstützt.	*<audienceIdField>*
Datentyp des Feldes Zielgruppen-ID	Der Datentyp des Feldes Zielgruppen-ID (entweder "numeric" oder "string"), der im API-Aufruf für Interact verwendet wird.	*<audienceIdFieldType>*
Cookiename, der die Sitzungs-ID darstellt	Der Name des Cookies, das die Sitzungs-ID enthält.	*<sessionIdCookie>*
Cookiename, der die Besucher-ID darstellt	Der Name des Cookies, das die Besucher-ID enthält.	*<visitorIdCookie>*

Optionale Funktionen

Die Zusatzfunktionen sind optionale Konfigurationsoptionen und optionale globale Einstellungen für die Installation des konfigurierten Web Connectors. Sie können die folgenden Werte angeben:

Tabelle 33. Optionale Website-weite Einstellungen für die Web Connector-Installation

Einstellung	Beschreibung	Funktional entsprechende Einstellung in jsconnector.xml
Debugmodus aktivieren	Zeigt (mit `ja` oder `nein`), ob der spezielle Debugmodus verwendet wird. Wenn Sie diese Funktion aktivieren, enthält der vom Web Connector zurückgegebene Inhalt einen JavaScript-Aufruf mit einem Warnhinweis, der den Client über die gerade erfolgte Seitenzuordnung informiert. Der Client muss einen Eintrag in der unter *<authorizedDebugClients>* angegebenen Datei haben, um den Warnhinweis abzurufen.	*<enableDebugMode>*
Hostdatei für autorisierte Debug-Clients	Der Pfad zu einer Datei, die eine Liste mit qualifizierenden Hosts oder IP-Adressen (Internet Protocol) für den Debugmodus enthält. Der Hostname oder die IP-Adresse des Clients muss in der angegebenen Datei enthalten sein, damit die Debuginformationen erfasst werden.	*<authorizedDebugClients>*
Digital Analytics for On Premises Seitentag-Integration aktivieren	Zeigt (mit `ja` oder `nein`), ob der Web Connector den angegebenen IBM Digital Analytics for On Premises-Tag am Ende des Seiteninhalts anhängen soll.	`<enableNetInsightTagging>`
Digital Analytics for On Premises# HTML-Vorlagendatei für den Tag	Die HTML/JavaScript-Vorlage, die verwendet wird, um einen Aufruf für den Digital Analytics for On Premises-Tag zu integrieren. Im Allgemeinen sollten Sie die Standardeinstellung übernehmen, solange Sie nicht aufgefordert werden, eine andere Vorlage bereitzustellen.	`<netInsightTag>`

HTML-Anzeigetypen zur Konfiguration von WebConnector

Verwenden Sie die Seite HTML-Anzeigetypen, um den HTML-Code zu bestimmen, der für jeden Interaktionspunkt auf der Seite zur Verfügung gestellt wird. Sie können aus der Liste mit Standardvorlagen (FLT-Dateien) auswählen, die eine Kombination aus CSS-Code für Cascading Style Sheets, HTML-Code und JavaScript-Code für jeden Interaktionspunkt enthalten. Sie können die zur Verfügung gestellten Vorlagen verwenden und bei Bedarf anpassen oder eigene Vorlagen erstellen.

Anmerkung: Die Konfigurationseinstellungen auf dieser Seite beziehen sich auf den Abschnitt `interactionPoints` der Konfigurationsdatei `jsconnector.xml`.

Der Interaktionspunkt kann auch Platzhalter (Zonen) enthalten, in denen die Angebotsattribute automatisch abgelegt werden können. Beispiel: Wenn Sie `${offer.TREATMENT_CODE}` einschließen, wird dies während der Interaktion durch den Verfahrenscode ersetzt, der diesem Angebot zugeordnet ist.

Die Vorlagen auf dieser Seite werden automatisch aus den Dateien geladen, die im `<Interact_home>/jsconnector/conf/html` Verzeichnis des Web Connector-Servers gespeichert sind. Alle neu erstellten Vorlagen werden ebenfalls in diesem Verzeichnis gespeichert.

Wenn Sie auf der Seite HTML-Anzeigetypen eine vorhandene Vorlage anzeigen oder ändern möchten, klicken Sie in der Liste auf die entsprechende `.flt`-Datei.

Um auf der Seite HTML-Anzeigetypen eine neue Vorlage zu erstellen, klicken Sie auf **Typ hinzufügen**.

Unabhängig von der Methode, die Sie zum Erstellen oder Ändern einer Vorlage verwenden, werden neben der Vorlagenliste die folgenden Informationen angezeigt:

Einstellung	**Beschreibung**	**Funktional entsprechende Einstellung in jsconnector.xml**
Dateiname für diesen Anzeigetyp	Der zugewiesene Name der Vorlage, die Sie bearbeiten. Der Name muss für das Betriebssystem zulässig sein, unter dem der Web Connector ausgeführt wird. Der Name darf zum Beispiel keinen Schrägstrich (/) enthalten, wenn Sie das Betriebssystem Microsoft Windows verwenden. Wenn Sie eine neue Vorlage erstellen, ist in diesem Feld `CHANGE_ME.flt` voreingestellt. Ändern Sie diese Voreinstellung, indem Sie einen aussagekräftigen Wert eingeben, bevor Sie den Vorgang fortsetzen.	*<htmlSnippet>*

Einstellung	Beschreibung	Funktional entsprechende Einstellung in jsconnector.xml
HTML-Snippet	Der spezifische Inhalt, den der Web Connector in den Interaktionspunkt auf der Webseite einfügen soll. Dieses Snippet kann HTML-Code, CSS-Formatierungsinformationen oder JavaScript zur Ausführung auf der Seite enthalten. Alle drei dieser Inhaltstypen müssen jeweils im Code BEGIN und END eingeschlossen werden, wie in den folgenden Beispielen dargestellt: • ${BEGIN_HTML} <*Ihr HTML-Inhalt*> ${END_HTML} • ${BEGIN_CSS} <*Ihre spezifischen Style-Sheet-Informationen für den Interaktionspunkt*> ${END_CSS} • ${BEGIN_JAVASCRIPT} <*Ihr spezifischer JavaScript-Code für den Interaktionspunkt*> ${END_JAVASCRIPT} Sie können auch mehrere vordefinierte Spezialcodes eingeben, die beim Laden der Seite automatisch ersetzt werden, zum Beispiel: • `${logAsAccept}` : Ein Makro übernimmt zwei Parameter (eine Ziel-URL und den TreatmentCode, der verwendet wird, um die Annahme des Angebots zu signalisieren) und ersetzt sie durch die URL für die Klickabfolge. • `${offer.AbsoluteLandingPageURL}` • `${offer.OFFER_CODE}` • `${offer.TREATMENT_CODE}` • `${offer.TextVersion}` • `$offer.AbsoluteBannerURL}` Alle hier aufgelisteten Angebotscodes stellen Angebotsattribute dar, die in IBM Campaign in der Angebotsvorlage definiert sind, die der Marketier zum Erstellen der Angebote verwendet hat, die Interact zurückgibt. Hinweis: Der Web Connector verwendet eine Vorlagenengine mit dem Namen FreeMarker. Diese bietet viele Zusatzoptionen, die beim Einrichten der Codes in den Seitenvorlagen hilfreich sein können. Weitere Informationen finden Sie unter http://freemarker.org/docs/index.html.	Keine funktional entsprechende Einstellung, weil das HTML-Snippet als eigenständige Datei außerhalb von jsconnector.xml bereitgestellt wird.

Einstellung	Beschreibung	Funktional entsprechende Einstellung in jsconnector.xml
Beispiele für Spezialcodes	Enthält Beispiele für Typen von Spezialcodes, mit denen Sie zum Beispiel HTML-, CSS- und JAVASCRIPT-Blöcke oder Ablagezonen einfügen und kennzeichnen können, um auf spezifische Metadaten eines Angebots zu verweisen.	Keine funktional entsprechende Einstellung.

Die Änderungen an dieser Seite werden automatisch gespeichert, wenn Sie zu einer anderen Web Connector-Konfigurationsseite navigieren.

Erweiterte Seiten zur Konfiguration von WebConnector

Auf der Seite 'Erweiterte Seiten' können Sie einem URL-Muster seitenspezifische Einstellungen zuweisen. Sie können zum Beispiel eine Seitenzuordnung einrichten, um für jede eingehende URL, die den Text "index.htm" enthält, die allgemeine Begrüßungsseite anzuzeigen, die spezielle Ereignisse zum Laden der Seite und definierte Interaktionspunkte für diese Zuordnung enthält.

Anmerkung: Die Konfigurationseinstellungen auf dieser Seite beziehen sich auf den Abschnitt `pageMapping` der Konfigurationsdatei `jsconnector.xml`.

Wenn Sie die Seite 'Erweiterte Seiten' verwenden möchten, um eine neue Seitenzuordnung zu erstellen, klicken Sie auf den Link **Seite hinzufügen** und geben Sie die erforderlichen Informationen für die Zuordnung an.

Seiteninfo

Die Seiteninfo-Konfigurationsoptionen für die Seitenzuordnung definieren das URL-Muster, das als Trigger für diese Zuordnung verwendet wird, und einige zusätzliche Einstellungen für die Art und Weise, wie diese Seitenzuordnung von Interact verarbeitet wird.

Einstellung	Beschreibung	Funktional entsprechende Einstellung in jsconnector.xml
URL enthält	Dies ist das URL-Muster, das der Web Connector in den eingehenden Seitenanforderungen suchen soll. Beispiel: Wenn die anfordernde URL "mortgage.htm" enthält, können Sie das mit der entsprechenden Informationsseite abgleichen.	*<urlPattern>*
Anzeigename für diese Seite oder diesen Seitensatz	Ein beschreibender Name mit einer Beschreibung dieser Seitenzuordnung zur eigenen Verwendung, z. B. "Seite mit Informationen zur Hypothek".	*<friendlyName>*
Gibt Angebote auch als JSON-Daten zur JavaScript-Verwendung zurück	Eine Dropdown-Liste, mit der Sie angeben können, ob der Web Connector die Rohdaten des Angebots im Format JavaScript Object Notation (http://www.json.org/) am Ende des Seiteninhalts einbeziehen soll.	*<enableRawDataReturn>*

Ereignisse, die bei einem Besuch (onload) dieser Seite oder dieses Seitensatzes ausgelöst werden

Diese Konfigurationsoptionen für die Seitenzuordnung definieren das URL-Muster, das als Trigger für diese Zuordnung verwendet wird, und einige zusätzliche Einstellungen für die Art und Weise, wie diese Seitenzuordnung von Interact verarbeitet wird.

Anmerkung: Die Konfigurationseinstellungen in diesem Abschnitt entsprechen dem Abschnitt `<pageLoadEvents>` der Datei `jsconnector.xml`.

Einstellung	Beschreibung	Funktional entsprechende Einstellung in jsconnector.xml
Einzelne Ereignisse	Eine Liste mit Ereignissen, die für diese Seite oder diesen Seitensatz verfügbar sind. Die Ereignisse in dieser Liste sind in Interact definiert. Wählen Sie mindestens ein Ereignis, das beim Laden der Seite ausgelöst werden soll. Für die Interact API-Aufrufe gilt folgende Reihenfolge: 1. `startSession` 2. `postEvent` für jedes einzelne Ereignis, das beim Laden der Seite ausgelöst wird (sofern Sie die einzelnen Ereignisse in Interact definiert haben) 3. Für jeden Interaktionspunkt: • `getOffers` • `postEvent(ContactEvent)`	*<event>*

Interaktionspunkte (Positionen zum Anzeigen der Angebote) für diese Seite oder diesen Seitensatz

Mit diesen Konfigurationsoptionen für die Seitenzuordnung können Sie auswählen, welche Interaktionspunkte auf der Seite angezeigt werdenInteract.

Anmerkung: Die Konfigurationseinstellungen in diesem Abschnitt entsprechen dem Abschnitt `<pageMapping>` | `<page>` | `<interactionPoints>` der Datei `jsconnector.xml`.

Einstellung	Beschreibung	Funktional entsprechende Einstellung in jsconnector.xml
Kontrollkästchen mit dem Namen des Interaktionspunkts	In diesem Abschnitt der Seite werden alle Interaktionspunkte angezeigt, die in der Konfigurationsdatei definiert sind. Wenn Sie das Kontrollkästchen neben dem Namen des Interaktionspunkts aktivieren, werden die verfügbaren Optionen angezeigt.	*<interactionPoint>*

Einstellung	Beschreibung	Funktional entsprechende Einstellung in jsconnector.xml
HTML-Element-ID (Interact nimmt die innerHTML-Einstellung vor)	Der Name des HTML-Elements, das den Inhalt für diesen Interaktionspunkt empfangen soll. Beispiel: Wenn Sie `<div id="welcomebanner">` auf der Seite angegeben haben, geben Sie `welcomebanner` (den ID-Wert) in dieses Feld ein.	*<htmlElementId>*
HTML-Anzeigetyp	Eine Dropdown-Liste, in der Sie den HTML-Anzeigetyp (die HTML-Snippets oder FLT-Dateien, die Sie zuvor auf einer anderen Web Connector-Konfigurationsseite definiert haben) für diesen Interaktionspunkt auswählen können.	*<htmlSnippet>*
Maximale Anzahl an Angeboten zur Präsentation (in einem Karussell oder Daumenkino)	Die maximale Anzahl an Angeboten, die der Web Connector für diesen Interaktionspunkt vom Interact Server abrufen soll. Dieses Feld ist optional und gilt nur für einen Interaktionspunkt, der die präsentierten Angebote regelmäßig aktualisiert, ohne die Seite erneut zu laden, z. B. im Karussellszenario, das mehrere Angebote abruft und jeweils immer nur ein Angebot präsentiert.	*<maxNumberOfOffers>*
Auszulösendes Ereignis, wenn das Angebot präsentiert wird	Der Name des Kontaktereignisses, das für diesen Interaktionspunkt übergeben werden soll.	*<contactEvent>*
Auszulösendes Ereignis, wenn das Angebot angenommen wird	Der Name des Annahmeereignisses, das in dem Moment, in dem auf den Angebotslink geklickt wird, für diesen Interaktionspunkt übergeben soll.	*<acceptEvent>*
Auszulösendes Ereignis, wenn das Angebot abgelehnt wird	Der Name des Ablehnungsereignisses, das für diesen Interaktionspunkt übergeben werden soll. **Anmerkung:** Diese Funktion wird derzeit noch nicht unterstützt	*<rejectEvent>*

Konfigurationsoptionen für den Web Connector

Im Allgemeinen können Sie die grafische Web Connector-Benutzeroberfläche verwenden, um die Web Connector-Einstellungen zu konfigurieren. Alle Einstellungen werden jedoch auch in der Datei jsconnector.xml im Verzeichnis `jsconnector/conf` gespeichert. Hier werden die einzelnen Parameter beschrieben, die in der Konfigurationsdatei jsconnector.xml gespeichert sind.

Parameter und Beschreibungen

Die folgenden Parameter werden in der Datei `jsconnector.xml` gespeichert und für die Interaktionen im Web Connector verwendet. Es gibt zwei Möglichkeiten zum Ändern dieser Einstellungen:

- Verwenden der Webseite zur Konfiguration des Web Connectors. Die Seite ist automatisch verfügbar, nachdem Sie die Web Connector-Anwendung implementiert und gestartet haben. Um die Webseite zur Konfiguration zu verwenden, öffnen Sie in Ihrem Web-Browser eine URL, die der folgenden URL ähnlich ist: `http://<host>:<port>/interact/jsp/WebConnector.jsp`.

  Die Änderungen, die Sie in der Webseite zur Administration vornehmen, werden in der Datei `jsconnector.xml` auf dem Server gespeichert, auf dem Sie den Web Connector implementiert haben.
- Bearbeiten Sie die Datei `jsconnector.xml` direkt in einem beliebigen Text- oder XML-Editor. Stellen Sie sicher, dass Ihnen der Umgang mit XML-Tags und Werten vertraut ist, bevor Sie diese Methode verwenden.

  Anmerkung: Jedes Mal, wenn Sie die Datei `jsconnector.xml` manuell bearbeiten, können Sie diese Einstellungen erneut laden, indem Sie die Seite zur Administration des Web Connectors laden (unter `http://<host>:<port>/interact/jsp/jsconnector.jsp`) und dann auf **Konfiguration erneut laden** klicken.

Die folgende Tabelle beschreibt die Konfigurationsoptionen, wie Sie sie in der Datei `jsconnector.xml` einstellen können.

Tabelle 34. Konfigurationsoptionen für den Web Connector

Parametergruppe	Parameter	Beschreibung
`defaultPageBehavior`		
	`friendlyName`	Eine leicht lesbare Kennung, die anstelle des URL-Musters auf der Webseite zur Konfiguration des Web Connectors angezeigt wird.
	`interactURL`	Die Basis-URL des Interact-Laufzeitservers. Hinweis: Dieser Parameter muss nur gesetzt werden, wenn Sie den Web Connector-Service (jsconnector) als implementierte Webanwendung ausführen. Es ist nicht erforderlich, diesen Parameter zu setzen, wenn der WebConnector automatisch als Teil des Interact-Laufzeitservers ausgeführt wird.
	`jsConnectorURL`	Die Basis-URL, die zum Generieren der URL für die Klickabfolge verwendet wird, z. B. `http://host:port/jsconnector/clickThru`
	`interactiveChannel`	Name des interaktiven Kanals, der diese Seitenzuordnung darstellt.
	`sessionIdCookie`	Name des Cookies, das die Sitzungs-ID enthält, die in den API-Aufrufen für Interact verwendet wird.
	`visitorIdCookie`	Name des Cookies, das die Zielgruppen-ID enthält.
	`audienceLevel`	Die Zielgruppenebene der Kampagne, die für ankommende Besucher im API-Aufruf für die Interact Laufzeit verwendet wird.
	`audienceIdField`	Name des `audienceId`-Feldes, das im API-Aufruf für die Interact Laufzeit verwendet wird. **Anmerkung:** Hinweis: Zielgruppen-IDs für mehrere Felder werden gegenwärtig nicht unterstützt.

Tabelle 34. Konfigurationsoptionen für den Web Connector (Forts.)

Parametergruppe	Parameter	Beschreibung
	audienceIdFieldType	Der Datentyp [numeric | string], der für das Feld mit der Zielgruppen-ID im API-Aufruf für die Interact Laufzeit verwendet wird
	audienceLevelCookie	Name des Cookies, das die Zielgruppenebene enthält. Dies ist optional. Wenn Sie diesen Parameter nicht setzen, verwendet das System die für audienceLevel definierte Einstellung.
	relyOnExistingSession	Wird im API-Aufruf für die Interact Laufzeit verwendet. Im Allgemeinen ist dieser Parameter auf "true" gesetzt.
	enableInteractAPIDebug	Wird im API-Aufruf für die Interact Laufzeit verwendet, um die Debuggerausgabe in den Protokolldateien zu aktivieren.
	pageLoadEvents	Das Ereignis, das übergeben wird, sobald diese bestimmte Seite geladen wird. Geben Sie innerhalb dieses Tags mindestens ein Ereignis ein. Verwenden Sie dazu das Format <event>event1</event>.
	interactionPointValues	Alle Elemente in dieser Kategorie werden als Standardwerte für fehlende Werte in den IP-spezifischen Kategorien verwendet.
	interactionPointValuescontactEvent	Standardname des Kontaktereignisses, das für diesen bestimmten Interaktionspunkt übergeben werden soll.
	interactionPointValuesacceptEvent	Standardname des Annahmeereignisses, das für diesen bestimmten Interaktionspunkt übergeben werden soll.
	interactionPointValuesrejectEvent	Standardname des Ablehnungsereignisses, das für diesen bestimmten Interaktionspunkt übergeben werden soll. (Hinweis: Diese Funktion wird derzeit nicht verwendet.)
	interactionPointValueshtmlSnippet	Der Name der HTML-Vorlage, die für diesen Interaktionspunkt bereitgestellt wird.
	interactionPointValuesmaxNumberOfOffers	Standardwert für die maximale Anzahl an Angeboten, die Interact für diesen Interaktionspunkt abruft.
	interactionPointValueshtmlElementId	Standardname des HTML-Elements, das den Inhalt für diesen Interaktionspunkt empfängt.
	interactionPoints	Diese Kategorie enthält die Konfiguration für jeden Interaktionspunkt. Für alle fehlenden Eigenschaften verwendet das System die Konfigurationen unter der Kategorie interactionPointValues.
	interactionPointname	Name des Interaktionspunkts (IP).
	interactionPointcontactEvent	Name des Kontaktereignisses, das für diesen bestimmten IP übergeben werden soll.

Tabelle 34. Konfigurationsoptionen für den Web Connector (Forts.)

Parametergruppe	Parameter	Beschreibung
	interactionPointacceptEvent	Name des Annahmeereignisses, das für diesen bestimmten IP übergeben werden soll.
	interactionPointrejectEvent	Name des Ablehnungsereignisses, das für diesen bestimmten IP übergeben werden soll. (Hinweis: Diese Funktion wird derzeit noch nicht unterstützt.)
	interactionPointhtmlSnippet	Name der HTML-Vorlage, die für diesen IP bereitgestellt wird.
	interactionPointmaxNumberOfOffers	Maximale Anzahl an Angeboten, die Interact für diesen IP abruft
	interactionPointhtmlElementId	Name des HTML-Elements, das den Inhalt für diesen Interaktionspunkt empfängt.
	enableDebugMode	Boolesches Flag (zulässige Werte: true oder false) zum Aktivieren des speziellen Debugmodus. Wenn dieser Wert auf true gesetzt ist, enthält der vom Web Connector zurückgegebene Inhalt einen JavaScript-Aufruf mit einem Warnhinweis, der den Client über die gerade erfolgte Seitenzuordnung informiert. Der Client muss einen Eintrag in der Datei authorizedDebugClients haben, um den Warnhinweis zu generieren.
	authorizedDebugClients	Eine vom speziellen Debugmodus verwendete Datei, die eine Liste mit qualifizierenden Hostnamen und IP-Adressen (Internet Protocol) für den Debugmodus enthält.
	enableRawDataReturn	Ein boolesches Flag (zulässige Werte: true oder false), um festzulegen, ob der Web Connector die Rohdaten des Angebots im JSON-Format am Ende des Inhalts anhängt.
	enableNetInsightTagging	Ein boolesches Flag (zulässige Werte: true oder false), um festzulegen, ob der Web Connector einen Digital Analytics for On Premises-Tag am Ende des Inhalts anhängt.
	apiSequence	Stellt eine Implementierung der APISequence-Benutzeroberfläche dar, die die Reihenfolge der API-Aufrufe im Web Connector festlegt, wenn ein pageTag aufgerufen wird. Standardmäßig verwendet die Implementierung die Reihenfolge StartSession, pageLoadEvents, getOffers und logContact, wobei die letzten beiden spezifisch für jeden Interaktionspunkt sind.
	clickThruApiSequence	Stellt eine Implementierung der APISequence-Benutzeroberfläche dar, die die Reihenfolge der API-Aufrufe im Web Connector festlegt, wenn ein clickThru aufgerufen wird. Standardmäßig verwendet die Implementierung die Reihenfolge StartSession und logAccept.

Tabelle 34. Konfigurationsoptionen für den Web Connector (Forts.)

Parametergruppe	Parameter	Beschreibung
	`netInsightTag`	Stellt die HTML- und JavaScript-Vorlage dar, die verwendet wird, um einen Aufruf des Digital Analytics for On Premises Tags zu integrieren. Im Allgemeinen ist es nicht erforderlich, diese Option zu ändern.

Verwenden der Administratorseite in Web Connector

Der Web Connector enthält eine Administrationsseite, die einige Tools bereitstellt, die beim Verwalten und Testen der Konfiguration behilflich sind, die mit spezifischen URL-Mustern verwendet werden kann. Sie können die Administratorseite auch verwenden, um eine geänderte Konfiguration erneut zu laden.

Informationen zur Administratorseite

Sie können `http://host:port/interact/jsp/jsconnector.jsp` mit jedem unterstützten Web-Browser öffnen. Dabei bezeichnet *host:port* den Namen des Hosts, auf dem der Web Connector ausgeführt wird, und den Port, der für das Listening von Verbindungen verwendet wird, z. B. `runtime.example.com:7001`

Es gibt mehrere Möglichkeiten, wie Sie die Administratorseite verwenden können:

Tabelle 35. Optionen für die Administratorseite im Web Connector

Option	Zweck
Konfiguration erneut laden	Klicken Sie auf den Link **Konfiguration erneut laden**, um alle auf dem Datenträger gespeicherten Konfigurationsänderungen erneut in den Speicher zu laden. Dies ist erforderlich, wenn Sie Änderungen direkt in der Web Connector-Konfigurationsdatei jsconnector.xml vorgenommen haben anstatt die Webseiten zur Konfiguration zu verwenden.
Konfiguration anzeigen	WebConnector-Konfiguration anzeigen, die dem URL-Muster im Feld **Konfiguration anzeigen** entspricht. Wenn Sie die URL einer Seite eingeben und auf **Konfiguration anzeigen** klicken, gibt der Web Connector die Konfiguration zurück, die das System aufgrund des zugeordneten Musters verwendet. Wird keine Übereinstimmung gefunden, wird die Standardkonfiguration zurückgegeben. Dies ist hilfreich, um zu testen, ob die richtige Konfiguration für eine bestimmte Seite verwendet wird.
Seitentag ausführen	Wenn Sie die Felder auf dieser Seite ausfüllen und dann auf **Seitentag ausführen** klicken, gibt der Web Connector das `pageTag`-Ergebnis zurück, das diesem URL-Muster entspricht. Dadurch wird der Aufruf eines Seitentags simuliert. Der Unterschied zwischen dem pageTag-Aufruf mit diesem Tool und der Verwendung einer echten Website liegt darin, dass alle Fehler oder Ausnahmen angezeigt werden, wenn Sie diese Administratorseite verwenden. Bei einer echten Website werden die Ausnahmen nicht zurückgegeben, sondern nur in der Web Connector-Protokolldatei angezeigt.

Web Connector-Beispielseite

Zusammen mit dem Interact Web Connector wird eine Beispieldatei mit dem Namen `WebConnectorTestPageSA.html` geliefert (im Verzeichnis `<Interact_Home/`

jsconnector/webapp/html), die demonstriert, wie viele Funktionen des Web Connectors in einer Seite markiert würden. Diese Seite wird auch im folgenden Beispiel dargestellt.

HTML-Beispielseite im Web Connector

```
<?xml version="1.0" encoding="us-ascii"?>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
    "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
   <head>
   <meta http-equiv="Content-Type" content="text/html; charset=us-ascii" />
   <meta http-equiv="CACHE-CONTROL" content="NO-CACHE" />
<script language="javascript" type="text/javascript">
//<![CDATA[
/* ####################################################################
This is a test page that  contains the WebConnector pageTag.  Because the
name of this file has TestPage embedded, the WebConnector will detect a URL
pattern match to the url pattern "testpage" in the default version of the
jsconnector.xml - the configuration definition mapped to that "testpage"
URL pattern will apply here. That means there should this page the
corresponding html element ids that correspond to the IPs for this URL
pattern (ie. 'welcomebanner', 'crosssellcarousel', and 'textservicemessage')
#################################################################### */

/* ####################################################################
This section sets the cookies for sessionId and visitorId.
Note that in a real production website, this is done most likely by the login
component. For the sake of testing, it's done here...  the name of the cookie
has to match what's configured in the jsconnector xml.
#################################################################### */
     function setCookie(c_name,value,expiredays)
     {
       var exdate=new Date();
       exdate.setDate(exdate.getDate()+expiredays);
       document.cookie=c_name+ "=" +escape(value)+
       ((expiredays==null) ? "" : ";expires="+exdate.toGMTString());
     }
     setCookie("SessionID","123");
     setCookie("CustomerID","1");

   /* ####################################################
   Now set up the html element IDs that correspond to the IPs
   #################################################### */
     document.writeln("<div id='welcomebanner'> This should change, "
  + "otherwise something is wrong <\/div>");
     document.writeln("<div id='crosssellcarousel'> This should change, "
  + "otherwise something is wrong <\/div>");
     document.writeln("<div id='textservicemessage'> This should change, "
  + "otherwise something is wrong <\/div>");
   //]]&gt;
</script><!--
################################################################################
this is what is pasted from the pageTag.txt file in the conf directory of
the WebConnector installation... the var unicaWebConnectorBaseURL needs to be
tweaked to conform to your local WebConnector environment
################################################################################
-->
<!-- BEGIN: IBM Interact Web Connector Page Tag -->
<!--
# ******************************************************************************
#  Licensed Materials - Property of IBM
#  IBM Interact
#  (c) Copyright IBM Corporation 2001, 2012.
#  US Government Users Restricted Rights - Use, duplication or disclosure
#  restricted by GSA ADP Schedule Contract with IBM Corp.
# ******************************************************************************
```

```
-->
<script language="javascript" type="text/javascript">
//<![CDATA[
   var unicaWebConnectorBaseURL=
        "[CHANGE ME - http://host:port/<jsconnector>/pageTag]";
   var unicaURLData = "ok=Y";
   try {
    unicaURLData += "&url=" + escape(location.href)
   } catch (err) {}
   try {
    unicaURLData += "&title=" + escape(document.title)
   } catch (err) {}
   try {
    unicaURLData += "&referrer=" + escape(document.referrer)
   } catch (err) {}
   try {
    unicaURLData += "&cookie=" + escape(document.cookie)
   } catch (err) {}
   try {
    unicaURLData += "&browser=" + escape(navigator.userAgent)
   } catch (err) {}
   try {
    unicaURLData += "&screensize=" +
    escape(screen.width + "x" + screen.height)
   } catch (err) {}
   try {
    if (affiliateSitesForUnicaTag) {
        var unica_asv = "";
        document.write("<style id=\"unica_asht1\" type=\"text/css\"> "
     + "p#unica_ashtp a {border:1px #000000 solid; height:100px "
     + "!important;width:100px "
     + "!important; display:block !important; overflow:hidden "
     + "!important;} p#unica_ashtp a:visited {height:999px !important;"
     + "width:999px !important;} <\/style>");
        var unica_ase = document.getElementById("unica_asht1");
        for (var unica_as in affiliateSitesForUnicaTag) {
         var unica_asArr = affiliateSitesForUnicaTag[unica_as];
         var unica_ashbv = false;
         for (var unica_asIndex = 0; unica_asIndex <
     unica_asArr.length && unica_ashbv == false;
     unica_asIndex++)
   {
          var unica_asURL = unica_asArr[unica_asIndex];
          document.write("<p id=\"unica_ashtp\" style=\"position:absolute; "
     + "top:0;left:-10000px;height:20px;width:20px;overflow:hidden; \
   margin:0;padding:0;visibility:visible;\"> \
   <a href=\"" + unica_asURL + "\">" + unica_as + " <\/a><\/p>");
          var unica_ae = document.getElementById("unica_ashtp").childNodes[0];
          if (unica_ae.currentStyle) {
           if (parseFloat(unica_ae.currentStyle["width"]) > 900)
       unica_ashbv = true
          } else if (window.getComputedStyle) {
           if (parseFloat(document.defaultView.getComputedStyle
       (unica_ae, null).getPropertyValue("width")) > 900)
    unica_ashbv = true
         }
          unica_ae.parentNode.parentNode.removeChild(unica_ae.parentNode)
        }
        if (unica_ashbv == true) {
         unica_asv += (unica_asv == "" ? "" : ";") + unica_as
        }
       }
       unica_ase.parentNode.removeChild(unica_ase);
       unicaURLData += "&affiliates=" + escape(unica_asv)
    }
   } catch (err) {}
   document.write("<script language='javascript' "
```

```
     + " type='text/javascript' src='" + unicaWebConnectorBaseURL + "?"
  + unicaURLData + "'><\/script>");
  //]]&gt;
</script>
<style type="text/css">
/*<![CDATA[*/
   .unicainteractoffer {display:none !important;}
   /*]]&gt;*/
</style>
   <title>Beispielseite in Interact Web Connector</title>
   </head>
   <body>
   <!-- END: IBM Interact Web Connector Page Tag -->
   <!--
################################################################################
end of pageTag paste
################################################################################
-->
   </body>
</html>
```

Anhang E. Integration von Interact und Digital Recommendations

IBM Interact kann mit IBM Digital Recommendations integriert werden, um Interact-basierte Produktempfehlungen bereitzustellen. Mit beiden Produkten können Produktempfehlungen für Angebote bereitgestellt werden, jedoch mithilfe unterschiedlicher Methoden. Digital Recommendations verwendet das Webverhalten eines Besuchers (Collaborative Filtering), um Korrelationen zwischen Besuchern und empfohlenen Angeboten herzustellen. Interact basiert auf dem bisherigen Verhalten, den Attributen und dem Protokoll eines Kunden, weniger auf Angeboten auf der Ansichtsebene, um so zu lernen, welche Angebote am besten mit dem Verhaltensprofil eines Kunden übereinstimmen (basierend auf Demografie und weiteren Informationen über den Kunden). Angebotsannahmeraten helfen dabei, mithilfe von Selbstlernfunktionen ein Vorhersagemodell zu erstellen. Interact nutzt die Vorteile beider Produkte, um so mithilfe eines persönlichen Profils Angebote zu definieren, die eine Kategorie-ID an Digital Recommendations übergeben. Anschließend werden Produktempfehlungen anhand der Beliebtheit (die „Weisheit der Vielen“) abgerufen, die dem Besucher als Teil des ausgewählten Angebots angezeigt werden. Auf diese Weise erhalten Kunden bessere Empfehlungen, die zu mehr Klickabfolgen und besseren Ergebnissen führen als die Verwendung nur eines der Produkte.

In den folgenden Abschnitten wird beschrieben, wie diese Integration funktioniert und wie Sie die bereitgestellte Beispielanwendung verwenden können, um eine benutzerdefinierte Angebotsintegration zu erstellen.

Übersicht über die Integration von Interact mit Digital Recommendations

In diesem Abschnitt wird beschrieben, wie IBM Interact mit IBM Digital Recommendations integriert werden kann, um Interact-basierte Produktempfehlungen bereitzustellen. Dies umfasst auch eine Beschreibung des Integrationsprozesses und der Mechanismen der Integration.

Die Integration von IBM Interact mit IBM Digital Recommendations erfolgt über eine REST-API (Representational State Transfer), die über die Digital Recommendations-Installation verfügbar ist. Mithilfe der REST-API-Aufrufe mit der entsprechenden Kategorie-ID kann Interact empfohlene Produkte abrufen und in die Angebotsinformationen auf der benutzerdefinierten Seite einbeziehen, die der Besucher anzeigt.

Wenn ein Besucher die URL der Webseite anzeigt (z. B. die Beispiel-JSP-Seite, die Bestandteil Ihrer Interact-Installation ist), ruft die Seite Interact auf, um ein Angebot abzurufen. Vorausgesetzt, dass das Angebot in Interact mit den richtigen Parametern konfiguriert wurde, so treten im einfachsten Fall die folgenden Schritte ein:

1. Die Seitenlogik ermittelt die Kunden-ID des Besuchers.
2. Ein API-Aufruf an Interact wird durchgeführt, der die erforderlichen Informationen übergibt, um ein Angebot für diesen Kunden zu erstellen.

3. Das zurückgegebene Angebot stellt der Webseite mindestens drei Attribute zur Verfügung: die URL für das Bild des Angebots, die URL der Landing-Page, wenn der Kunde sich durchklickt, und die Kategorie-ID, um zu bestimmen, welche Produkte empfohlen werden.
4. Die Kategorie-ID wird dann für einen Digital Recommendations-Aufruf verwendet, um die empfohlenen Produkte abzurufen. Diese Produktmenge ist JSON-formatiert (JavaScript Object Notation) und nach bestverkauften Produkten in dieser Kategorie sortiert.
5. Das Angebot und die Produkte werden dann im Browser des Besuchers angezeigt.

Diese Integration ist nützlich, um Angebotsempfehlungen und Produktempfehlungen zu kombinieren. Sie können z. B. auf einer Webseite zwei Interaktionspunkte einschließen: einen für ein Angebot und einen für Empfehlungen, die mit dem Angebot übereinstimmen. Dazu führt die Webseite einen Aufruf an Interact durch, um mithilfe einer Echtzeitsegmentierung das beste Angebot zu ermitteln (z. B. 10 % Rabatt auf alle Kleingeräte). Wenn die Seite das Angebot von Interact erhält, enthält das Angebot die Kategorie-ID (in diesem Beispiel für Kleingeräte). Die Seite übergibt dann mithilfe eines API-Aufrufs die Kategorie-ID für Kleingeräte an Digital Recommendations und erhält als Antwort die besten Produktempfehlungen für diese Kategorie anhand der Beliebtheit.

In einem einfacheren Beispiel führt eine Webseite einen Aufruf an Interact durch, nur um eine Kategorie herauszufinden (z. B. hochwertiges Besteck), die mit dem Kundenprofil übereinstimmt. Anschließend übergibt die Seite die erhaltene Kategorie-ID an Digital Recommendations und erhält Produktempfehlungen für Besteck.

Voraussetzungen für die Integration

Bevor Sie die Integration von Digital Recommendations mit Interact verwenden können, müssen Sie zunächst sicherstellen, dass die in diesem Abschnitt beschriebenen Voraussetzungen erfüllt sind.

Stellen Sie sicher, dass die folgenden Voraussetzungen erfüllt sind:

- Sie sind mit der Verwendung der Interact-API vertraut, die an einer anderen Stelle im *Administratorhandbuch* und in der Onlinehilfe dokumentiert ist.
- Sie sind mit der Digital Recommendations-REST-API vertraut, die in Ihrer Digital Recommendations-Dokumentation für Entwickler beschrieben ist.
- Sie verfügen über grundlegende Kenntnisse in HTML, JavaScript, CSS und JSON (JavaScript Object Notation).

  JSON ist wichtig, weil die Digital Recommendations-REST-API die von Ihnen angeforderten Produktinformationen als Daten im JSON-Format zurückgibt.
- Sie sind mit serverseitiger Webseitencodierung vertraut, weil die mit Interact bereitgestellte Demonstrationsanwendung JSP verwendet (JSP ist jedoch nicht erforderlich).
- Sie verfügen über ein gültiges Digital Recommendations-Konto und die Liste der Kategorie-IDs, die Interact zum Abrufen von Produktempfehlungen verwenden soll (die am besten verkauften oder beliebtesten Produkte in der von Ihnen angegebenen Kategorie).
- Sie verfügen über den Link zur Digital Recommendations-REST-API (eine URL für Ihre Digital Recommendations-Umgebung).

  In der Beispielanwendung, die Bestandteil Ihrer Interact-Installation ist, finden Sie ein Beispiel. Weitere Informationen finden Sie im Beispielcode unter „Verwenden des Integrationsbeispielprojekts" auf Seite 348.

Konfigurieren eines Angebots mit Digital Recommendations-Integration

Bevor Ihre Webseite Digital Analytics Digital Recommendations aufrufen kann, um ein empfohlenes Produkt abzurufen, müssen Sie zunächst das IBM Interact-Angebot mit den erforderlichen Informationen konfigurieren, die an Digital Recommendations übermittelt werden sollen.

Informationen zu diesem Vorgang

Um ein Angebot einzurichten, das mit Digital Recommendations verlinkt ist, müssen Sie zunächst darauf achten, dass die folgenden Bedingungen erfüllt sind:

- Stellen Sie sicher, dass Ihr Interact-Laufzeitserver ordnungsgemäß eingerichtet ist und ausgeführt wird.
- Stellen Sie sicher, dass der Laufzeitserver eine Verbindung zum Digital Recommendations-Server aufbauen kann. Achten Sie auch darauf, dass Ihre Firewall ausgehende Standardwebverbindungen (Port 80) nicht verhindert.

Führen Sie die folgenden Schritte aus, um ein mit Digital Recommendations integriertes Angebot einzurichten.

Vorgehensweise

1. Erstellen oder bearbeiten Sie ein Angebot für Interact.

   Informationen zum Erstellen und Ändern von Angeboten finden Sie im Benutzerhandbuch *IBM Interact User's Guide* und in der IBM Campaign-Dokumentation.
2. Achten Sie darauf, dass das Angebot neben den anderen enthaltenen Einstellungen die folgenden Angebotsattribute enthält:
   - Die URL (Uniform Resource Locator) mit dem Link zum Bild des Angebots.
   - Die URL mit dem Link zur Landing-Page für das Angebot.
   - Eine diesem Angebot zugeordnete Digital Recommendations-Kategorie-ID.

     Sie können die Kategorie-ID manuell aus Ihrer Digital Recommendations-Konfiguration abrufen. Interact kann Kategorie-ID-Werte nicht direkt abrufen.

   In der Demonstrationswebanwendung, die Bestandteil Ihrer Interact-Installation ist, heißen diese Angebotsattribute `ImageURL`, `ClickThruURL` und `CategoryID`. Sie können beliebige beschreibende Namen verwenden, solange Ihre Webanwendung mit den Werten übereinstimmt, die das Angebot erwartet.

   Sie können z. B. ein Angebot namens „10PercentOff" definieren, das diese Attribute enthält und in dem die Kategorie-ID (die Sie aus Ihrer Digital Recommendations-Konfiguration abgerufen haben) `PROD1161127` ist, die URL der Angebotsklickabfolge `http://www.example.com/success` ist und die URL des für das Angebot angezeigten Bildes `http://localhost:7001/sampleIO/img/10PercentOffer.jpg` ist (in diesem Fall befindet sich die URL lokal auf dem Interact-Laufzeitserver).
3. Definieren Sie die Verfahrensregeln für einen interaktiven Kanal, der dieses Angebot enthalten soll, und implementieren Sie den interaktiven Kanal wie gewohnt.

Ergebnisse

Das Angebot ist jetzt mit den Informationen definiert, die für die Integration mit Digital Recommendations erforderlich sind. Damit Digital Recommendations Inter-

act Produktempfehlungen bereitstellen kann, müssen Sie jetzt nur noch Ihre Webseiten so konfigurieren, dass sie die erforderlichen API-Aufrufe durchführen.

Wenn Sie Ihre Webanwendung so konfigurieren, dass sie Besuchern die integrierte Seite bereitstellt, müssen Sie darauf achten, dass die folgenden Dateien im Verzeichnis `WEB-INF/lib` enthalten sind:

- *Interact_Home*`/lib/interact_client.jar`, die zum Bearbeiten von Aufrufen von Ihrer Webseite an die Interact-API erforderlich ist.
- *Interact_Home*`/lib/JSON4J_Apache.jar`, die zum Bearbeiten der Daten erforderlich ist, die vom Aufruf an die Digital Recommendations-REST-API zurückgegeben werden. Diese API gibt Daten im JSON-Format zurück.

Weitere Informationen zum Bereitstellen der Angebote für Ihre Kunden finden Sie unter „Verwenden des Integrationsbeispielprojekts".

Verwenden des Integrationsbeispielprojekts

Jede Interact-Laufzeitinstallation beinhaltet ein Beispielprojekt, das den Prozess der Integration von Digital Recommendations mit Interact demonstriert. Das Beispielprojekt stellt eine vollständige End-to-End-Demonstration zur Verfügung, wie Sie eine Webseite erstellen, die ein Angebot aufruft, das eine Kategorie-ID enthält. Diese Kategorie-ID wird dann an Digital Recommendations übergeben, um eine Liste mit empfohlenen Produkten abzurufen, die an den Interaktionspunkten der Seite dargestellt wird.

Übersicht

Sie können das enthaltene Beispielprojekt ohne Änderungen verwenden, wenn Sie den Integrationsprozess testen möchten. Sie können es auch als Ausgangspunkt verwenden, um eigene benutzerdefinierte Seiten zu entwickeln. Sie finden das Beispielprojekt in der folgenden Datei:

Interact_home`/samples/IntelligentOfferIntegration/MySampleStore.jsp`

Diese Datei enthält neben einem vollständigen, funktionsfähigen Beispiel des Integrationsprozesses auch umfassende Anmerkungen, die erklären, was Sie in Interact einrichten müssen, was Sie in der JSP-Datei anpassen müssen und wie Sie die Seite ordnungsgemäß implementieren, damit sie mit Ihrer Installation ausgeführt wird.

MySampleStore.jsp

Zur Vereinfachung ist die Datei „MySampleStore.jsp" hier dargestellt. Dieses Beispiel wird in nachfolgenden Releases von Interact u. U. aktualisiert. Verwenden Sie daher als Ausgangspunkt für alle Beispiele, die Sie möglicherweise benötigen, die Datei, die Bestandteil Ihrer Installation ist.

```
<!--
# ********************************************************************************
#  Licensed Materials - Property of IBM
#  IBM Interact
#  (c) Copyright IBM Corporation 2001, 2011.
#  US Government Users Restricted Rights - Use, duplication or disclosure
#  restricted by GSA ADP Schedule Contract with IBM Corp.
# ********************************************************************************

-->


<%@ page contentType="text/html; charset=UTF-8" language="java" %>
<%@ page import="java.net.URL,
```

```
    java.net.URLConnection,
    java.io.InputStreamReader,
    java.io.BufferedReader,
    com.unicacorp.interact.api.*,
    com.unicacorp.interact.api.jsoverhttp.*,
    org.apache.commons.json.JSONObject,
    org.apache.commons.json.JSONArray" %>

<%

 /*****************************************************************************
 * This sample jsp program demonstrates integration of Interact and Digital Recommendations.
 *
 * When the URL for this jsp is accessed via a browser the logic will call Interact
 * to fetch an Offer.  Based on the categoryID associated to the offer, the logic
 * will call Digital Recommendations to fetch recommended products.  The offer and products
 * will be displayed.
 * To toggle the customerId in order to demonstrate different offers, one can simply
 * append cid=<id> to the URL of this JSP.
 *
 * Prerequisites to understand this demo:
 * 1) familiarity of Interact and its java API
 * 2) familiarity of IntelligentOffer and its RestAPI
 * 3) some basic web background ( html, css, javascript) to mark up a web page
 * 4) Technology used to generate a web page (for this demo, we use JSP executed on the server side)
 *
 *
 * Steps to get this demo to work:
 * 1) set up an Interact runtime environment that can serve up offers with the following
 * offer attributes:
 *  ImageURL : url that links to the image of the offer
 *  ClickThruURL : url that links to the landing page of the offer
 * CategoryID: Digital Recommendations category id associated to the offer
 * NOTE: alternate names for the attributes may be used as long as the references to those
 * attributes in this jsp are modified to match.
 * 2) Obtain a valid REST API URL to the Intelligent Offer environment
 * 3) Embed this JSP within a Java web application
 * 4) Make sure interact_client.jar is in the WEB-INF/lib directory (communication with Interact)
 * 5) Make sure JSON4J_Apache.jar (from interact install) is in the
 *    WEB-INF/lib directory (communication with IO)
 * 6) set the environment specific properties in the next two sections
 *****************************************************************************/

 /*****************************************************************************
 *   *****************CHANGE THESE SETTINGS TO REFLECT YOUR ENV*******************
 * Set your Interact environment specific properties here ...
 *****************************************************************************/

 final String sessionId="123";
 final String interactiveChannel = "SampleIO";
 final String audienceLevel = "Customer";
 final String audienceColumnName="CustomerID";
 final String ip="ip1";
 int customerId=1;
 final String interactURL="http://localhost:7011/interact/servlet/InteractJSService";
 final boolean debug=true;
 final boolean relyOnExistingSession=true;

 /*****************************************************************************
   *****************CHANGE THESE SETTINGS TO REFLECT YOUR ENV*******************
 * Set your Digital Recommendations environment specific properties here...
 *****************************************************************************/

 final String ioURL="http://recs.coremetrics.com/iorequest/restapi";
 final String zoneID="ProdRZ1";
 final String cID="90007517";



 /*****************************************************************************
   *****************************************************************************/

 StringBuilder interactErrorMsg = new StringBuilder();
 StringBuilder intelligentOfferErrorMsg = new StringBuilder();

 // get the customerID if passed in as a parameter
 String cid = request.getParameter("cid");
 if(cid != null)
 {
```

```
 customerId = Integer.parseInt(cid);
}

// call Interact to get offer
Offer offer=getInteractOffer(interactURL,sessionId,interactiveChannel,audienceLevel,
       audienceColumnName,ip,customerId,debug,relyOnExistingSession,interactErrorMsg);

// get specific attributes from the offer (img url, clickthru url, & category id)
String offerImgURL=null;
String offerClickThru=null;
String categoryId="";

if(offer != null)
{
 for(NameValuePair offerAttribute : offer.getAdditionalAttributes())
    {
        if(offerAttribute.getName().equalsIgnoreCase("ImageURL"))
        {
         offerImgURL=offerAttribute.getValueAsString();
        }
        else if(offerAttribute.getName().equalsIgnoreCase("ClickThruURL"))
        {
         offerClickThru=offerAttribute.getValueAsString();
        }
        else if(offerAttribute.getName().equalsIgnoreCase("CategoryID"))
        {
         categoryId=offerAttribute.getValueAsString();
        }
    }
}

// call Digital Recommendations to get products
JSONObject products=getProductsFromIntelligentOffer(ioURL, cID, zoneID, categoryId,
       intelligentOfferErrorMsg);

%>

<html>
 <head>
  <title>My Favorite Store</title>

  <script language="javascript" type="text/javascript">
   var unicacarousel=(function(){var g=false;var h;var j=0;var k=0;var l=0;var m=40;
         var n=new Array(0,2,6,20,40,60,80,88,94,97,99,100);var o=function(a){var b=a.parentNode;
         h=b.getElementsByTagName("UL")[0];var c=h.getElementsByTagName("LI");j=c[0].offsetWidth;
         k=c.length;l=Math.round((b.offsetWidth/j));unicacarousel.recenter()};var p=function(a)
         {var b=parseFloat(h.style.left);if(isNaN(b))b=0;for(var i=0;i<n.length;i++)
         {setTimeout("unicacarousel.updateposition("+(b+(a*(n[i]/100)))+");",((i*m)+50))}
         setTimeout("unicacarousel.recenter();",((i*m)+50))};return{gotonext:function(a,b)
         {if(!g){o(a);g=true;p((-1*b*j))}},gotoprev:function(a,b){if(!g){o(a);g=true;p((b*j))}},
         updateposition:function(a){h.style.left=a+"px"},recenter:function(){var a=parseFloat(h.style.left);
         if(isNaN(a))a=0;var b=j*Math.round(((l-k)/2));var c=Math.abs(Math.round((b-a)/j));
         if(a<b){var d=h.getElementsByTagName("LI");var e=new Array();
         for(var i=0;i<c;i++){e[e.length]=d[i]}for(var i=0;i<e.length;i++)
         {h.insertBefore(e[i],null)}unicacarousel.updateposition(b)}else
         if(a>b){var d=h.getElementsByTagName("LI");var e=new Array();
         for(var i=0;i<c;i++){e[e.length]=d[d.length-c+i]}var f=d[0];
         for(var i=0;i<e.length;i++){h.insertBefore(e[i],f)}unicacarousel.updateposition(b)}g=false}})();
  </script>

  <style type="text/css">
 .unicaofferblock_container {width:250px; position:relative; display:block;
                             text-decoration:none; color:#000000; cursor: pointer;}
 .unicaofferblock_container .unicateaserimage {margin:0px 0.5em 0.25em 0px; float:left;}
 .unicaofferblock_container .unicabackgroundimage {position:absolute; top:0px; left:0px;}
 .unicaofferblock_container .unicabackgroundimagecontent {width:360px; height:108px;
                     padding:58px 4px 4px 20px; position:relative; top:0px;}
 .unicaofferblock_container h4 {margin:0px; padding:0px; font-size:14px;}

 .unicacarousel {width:588px; position:relative; top:0px;}
 .unicacarousel_sizer {width:522px; height:349px; margin:0px 33px; padding:0;
                       overflow:hidden; position:relative;}
 .unicacarousel_rotater {height:348px; width:1000px; margin:0 !important;
                         padding:0; list-style:none; position:absolute; top:0px;
                         left:0px;}
 .unicacarousel li {width:167px; height:349px; float:left; padding:0 4px;
                    margin:0px !important; list-style:none !important;
                    text-indent:0px !important;}
 .unicacarousel_gotoprev, .unicacarousel_gotonext {width:18px; height:61px;
```

```
                    top:43px; background:url(../img/carouselarrows.png) no-repeat;
                   position:absolute; z-index:2; text-align:center; cursor:pointer;
                   display:block; overflow:hidden; text-indent:-9999px;
                   font-size:0px; margin:0px !important;}
 .unicacarousel_gotoprev {background-position:0px 0; left:0;}
 .unicacarousel_gotonext {background-position:-18px 0; right:0;}

  </style>


 </head>

 <body>

  <b>Welcome To My Store</b> Mr/Mrs. <%=customerId %>
  <br><br>
<% if(offer != null) { %>
 <!-- Interact Offer HTML -->

 <div onclick="location.href='<%=offerClickThru %>'" class="unicaofferblock_container">
  <div class="unicabackgroundimage">
   <a href="<%=offerClickThru %>"><img src="<%=offerImgURL %>" height="170"
          width="695" border="0"></a>
  </div>
 </div>


<% } else { %>
 No offer available.. <br> <br>
 <%=interactErrorMsg.toString() %>
<% } %>

<% if(products != null) { %>
 <!-- IntelligentOffer Products HTML -->
 <br><br><br> <br><br><br> <br><br><br> <br><br><br> <br>
 <div class="unicacarousel">
 <div class="unicacarousel_sizer">
  <ul class="unicacarousel_rotater">


 <% JSONArray recs = products.getJSONObject("io").getJSONArray("recs");
  if(recs != null)
  {
   for(int x=0;x< recs.length();x++)
   {
    JSONObject rec = recs.getJSONObject(x);
    if(rec.getString("Product Page") != null &&
              rec.getString("Product Page").trim().length()>0) {
    %>

    <li>
     <a href="<%=rec.getString("Product Page") %>" title="<%=rec.getString("Product Name") %>">
             <img src="<%=rec.getString("Image") %>"  width="166" height="148" border="0" />
     <%=rec.getString("Product Name") %>
     </a>
    </li>

    <% }
   }
  }
 %>
 </ul>
 </div>
 <p class="unicacarousel_gotoprev" onclick="unicacarousel.gotoprev(this,1);"></p>
 <p class="unicacarousel_gotonext" onclick="unicacarousel.gotonext(this,1);"></p>
</div>
<% } else { %>
 <div>
 <br><br> <br><br><br> <br><br><br> <br><br><br> <br>
 No products available...<br> <br>
 <%=intelligentOfferErrorMsg.toString() %>
 </div>
<% } %>

 </body>
</html>
```

```
<%!
 /*****************************************************************************
 * The following are convenience functions that will fetch from Interact and
 *   Digital Recommendations
 *****************************************************************************/

 /*****************************************************************************
 * Call Digital Recommendations to retrieve recommended products
 *****************************************************************************/
 private JSONObject getProductsFromIntelligentOffer(String ioURL, String cID,
     String zoneID, String categoryID, StringBuilder intelligentOfferErrorMsg)
 {

  try {

   ioURL += "?cm_cid="+cID+"&cm_zoneid="+zoneID+"&cm_targetid="+categoryID;
   System.out.println("CoreMetrics URL:"+ioURL);
   URL url = new java.net.URL(ioURL);

    URLConnection conn = url.openConnection();

    InputStreamReader inReader = new InputStreamReader(conn.getInputStream());
    BufferedReader in = new BufferedReader(inReader);

    StringBuilder response = new StringBuilder();

    while(in.ready())
    {
     response.append(in.readLine());
    }

    in.close();

    intelligentOfferErrorMsg.append(response.toString());

    System.out.println("CoreMetrics:"+response.toString());

    if(response.length()==0)
     return null;

    return new JSONObject(response.toString());
  }
  catch(Exception e)
  {
   intelligentOfferErrorMsg.append(e.getMessage());
   e.printStackTrace();
  }

  return null;

 }

 /*****************************************************************************
 * Call Interact to retrieve offer
 *****************************************************************************/
 private Offer getInteractOffer(String interactURL,String sessionId,String interactiveChannel,
     String audienceLevel,
        String audienceColumnName,String ip, int customerId,boolean debug,
                        boolean relyOnExistingSession, StringBuilder interactErrorMsg)
 {
  try {
   InteractAPI api = InteractAPI.getInstance(interactURL);
   NameValuePairImpl custId = new NameValuePairImpl();
      custId.setName(audienceColumnName);
       custId.setValueAsNumeric(Double.valueOf(customerId));
     custId.setValueDataType(NameValuePair.DATA_TYPE_NUMERIC);
       NameValuePairImpl[] audienceId = { custId };

       // call startSession
      Response response = api.startSession(sessionId, relyOnExistingSession,
          debug, interactiveChannel, audienceId, audienceLevel, null);

      if(response.getStatusCode() == Response.STATUS_ERROR)
      {
       printDetailMessageOfWarningOrError("startSession",response, interactErrorMsg);
   }

      // call getOffers
      response = api.getOffers(sessionId, ip, 1);
```

```
      if(response == null  || response.getStatusCode() == Response.STATUS_ERROR)
      {
       printDetailMessageOfWarningOrError("getOffers",response, interactErrorMsg);
   }

            OfferList offerList=response.getOfferList();

            if(offerList != null && offerList.getRecommendedOffers() != null)
            {
             return offerList.getRecommendedOffers()[0];
            }
  }
  catch(Exception e)
  {
   interactErrorMsg.append(e.getMessage());
   e.printStackTrace();
  }
  return null;
 }

    private void printDetailMessageOfWarningOrError(String command, Response response,
        StringBuilder interactErrorMsg)
    {
     StringBuilder sb = new StringBuilder();
        sb.append("Calling "+command).append("<br>");
        AdvisoryMessage[] messages = response.getAdvisoryMessages();

        for(AdvisoryMessage msg : messages)
        {
         sb.append(msg.getMessage()).append(":");
         sb.append(msg.getDetailMessage());
         sb.append("<br>");
        }
        interactErrorMsg.append(sb.toString());
    }
%>
```

Anhang F. Integration von Interact und Digital Data Exchange

Mit Digital Data Exchange kann Ihre Website eine Verbindung zu Interact herstellen, um eine leistungsfähige Omni-Channel-Ausführungsengine bereitzustellen, die die besten Angebote an die optimalen Kanäle übergibt und aus dem Angebotsfeedback lernt, um die Effektivität des Marketings kontinuierlich zu erhöhen.

Sie können dieses Tool verwenden, wenn Ihr Marketing-Team Interact zur Verwaltung von Omni-Channel-Angeboten verwendet und diese individuell gestalteten Intelligent Offers auf Ihre Websites erweitern möchte.

IBM Digital Data Exchange integriert Marketinglösungen von IBM und anderen Anbietern mit digitalen Kundeninformationen über eine API für die Syndikation von Echtzeitdaten und eine unternehmensweite Tagmanagementlösung.

Ohne IBM Digital Data Exchange sind die Anbieter beim Verbinden von Interact mit ihrer Webseite und beim Aufrufen der Interact-API von verschiedenen Webseiten von der IT abhängig. Mit IBM Digital Data Exchange können Anbieter die IT umgehen und eine direkt Verbindung zu IBM Digital Data Exchange herstellen, um IBM Digital Data Exchange-Tags auf verschiedenen Webseiten einzuschließen.

Voraussetzungen

Bevor Sie die Integration von Interact mit Digital Data Exchange verwenden können, müssen Sie zunächst sicherstellen, dass die in diesem Abschnitt beschriebenen Voraussetzungen erfüllt sind.

Stellen Sie sicher, dass die folgenden Voraussetzungen erfüllt sind.

- Sie sind mit der Interact-JavaScript-API vertraut, die an einer anderen Stelle im Administratorhandbuch und in der Onlinehilfe dokumentiert ist.
- Sie sind mit Digital Data Exchange-Tagging und Seitengruppen vertraut.
- Sie verfügen über ein gültiges Digital Data Exchange-Konto.
- Die Datei `interactapi.js` wird öffentlich per Hosting bereitgestellt, damit auf sie in den Einstellungen **Anbieter** zugegriffen werden kann.

IBM Interact mit IBM Digital Data Exchange in Website integrieren

Anhand der folgenden Schritte können Sie Interact unter Verwendung von Digital Data Exchange in Ihre Website integrieren.

Vorgehensweise

1. Geben Sie die Position der Datei `Interactapi.js` an.
   a. Navigieren Sie in Digital Data Exchange zu **Anbieter** > **Anbietereinstellungen**.
   b. Wählen Sie IBM Interact in der Dropdown-Liste **Anbieter** aus.
   c. Geben Sie unter **Bibliothekspfad** die URL ein, unter der Sie die Datei `Interactapi.js` per Hosting bereitstellen. Schließen Sie in diese URL nicht das Protokoll (http oder https) ein.
   d. Fügen Sie unter **Pfad zu öffentlichem REST-Servlet** den Pfad zum REST-Servlet hinzu.

2. Navigieren Sie in Digital Data Exchange zu **Verwalten** > **Globale Einstellungen**, um den Objektnamen anzugeben, der als Seitenkennung unter **Eindeutige Seitenkennung** verwendet werden soll. Als Objektnamen können Sie zum Beispiel `digitalData.pageInstanceID` festlegen.
3. Schließen Sie die Datei `eluminate.js` und eine Kennung auf der Webseite ein, auf der die Tags von Digital Data Exchange eingefügt werden sollen. Sie müssen für jede Webseite eine eindeutige Kennung festlegen, damit die verschiedenen Seiten von Digital Data Exchange unterschieden werden können.

   Sie können zum Beispiel das folgende Script zur Homepage hinzufügen.

```
<!-- Festlegen der Seitenkennung -->
          <script>
                  digitalData={pageInstanceID:"INTERACT_HomePage"};
          </script>

<!-- Einschließen des eluminate-Scripts -->
         <script type="text/javascript" src="http://libs.
                  coremetrics.com/eluminate.js">
         </script>
         <script type="text/javascript">
                  cmSetClientID("51310000|INTERACTTEST",false,"data.
                  coremetrics.com",document.domain);
         </script>
```

4. Erstellen Sie in Digital Data Exchange Tags, Codesegmente, Funktionen und weitere Elemente, die Sie auf der Webseite hinzufügen möchten.
5. Erstellen Sie Seitengruppen, um zu definieren, was auf jeder Seite abgelegt werden soll.

   Weitere Informationen finden Sie im IBM Digital Data Exchange-Benutzerhandbuch.

Interact-Tags in Digital Data Exchange

Mit den Digital Data Exchange-Standardtags können Sie Variationen der Tags definieren, die für Webseiten geeignet sind, auf denen Daten von anderen Standorten dargestellt werden. Sobald sie definiert sind, werden diese Tags zur Interact-Tagliste hinzugefügt. Tags müssen nicht Felder aufweisen, die definiert werden müssen, müssen nicht über erforderliche Tagfelder verfügen und können direkt verwendet werden.

Die folgenden Interact-Tags sind in Digital Data Exchange unter **Tags** verfügbar.

- Sitzung beenden
- Angebot erhalten
- Bibliothek laden
- Ereignis senden
- Zielgruppe festlegen
- Sitzung starten

Wenn Sie Interact-Tags verwenden möchten, bearbeiten Sie die Tags, um Tagfeld, Methode, Objektname, Datentyp und Änderungswert für jeden Interact-Tag zu definieren.

Für die Tags "Ereignis senden", "Zielgruppe festlegen" und "Sitzung starten" sind benutzerdefinierte Tagfelder zulässig. Verwenden Sie das Symbol zum Hinzufügen von Tagfeldern und klicken Sie auf das Symbol "Bearbeiten", um den angepassten Parameter zu definieren. Der Prozess ist zwar derselbe wie bei jeder Parameterdefinition, der Name des Parameters kann jedoch bearbeitet werden und muss aus

dem Parameternamen, einem Doppelpunkt und dem Datentyp des Parameters bestehen. Die Reihenfolge der angepassten Parameter im Tag kann mithilfe der Abwärts- und Aufwärtspfeile geändert werden.

Die Tags können auch an JavaScript-Funktionen oder HTML-Objekte gebunden werden, sodass sie nach dem Auslösen der Funktion oder einem HTML-Objektereignis ausgelöst werden.

Weitere Informationen zum Definieren und Binden von Tags sowie zum Arbeiten mit Tags finden Sie im IBM Digital Data Exchange-Benutzerhandbuch.

Detaillierte Anwendungsfälle der Interact- und Digital Data Exchange-Integration finden Sie unter https://www.ibm.com/developerworks/community/wikis/home?lang=en#!/wiki/W214f7731a379_4712_a1ce_5d7a833d4cca/page/IBM%20Interact%20and%20IBM%20Digital%20Data%20Exchange%20Integration.

Sitzung beenden

Der Tag "Sitzung beenden" markiert das Ende einer Websitzung.

Für den Tag "Sitzung beenden" stehen die folgenden Tagfelder zur Verfügung.

Tabelle 36. Tags "Sitzung beenden"

Tagfeld	Beschreibung
*Sitzungs-ID	Gibt die Sitzungs-ID an.
Name der Callback-Funktion bei Erfolg	Definiert den Namen der Funktion, die aufgerufen werden soll, wenn die Methode zum Beenden der Sitzung erfolgreich ist.
Name der Callback-Funktion bei Fehler	Definiert den Namen der Funktion, die aufgerufen werden soll, wenn die Methode zum Beenden der Sitzung nicht erfolgreich ist.

Alle **Tagfelder** mit der Markierung ***** sind erforderlich.

Angebot erhalten

Mit dem Tag "Angebot erhalten" können Sie Angebote vom Laufzeitserver anfordern.

Für den Tag "Angebot erhalten" stehen die folgenden Tagfelder zur Verfügung.

Tabelle 37. Tags "Angebot erhalten"

Tagfeld	Beschreibung
*Sitzungs-ID	Gibt die Sitzungs-ID an.
*Name des Interaktionspunkts	Gibt den Namen des Interaktionspunkts an, auf den von dieser Methode verwiesen wird. Dieser Name muss exakt mit dem Namen des im interaktiven Kanal definierten Interaktionspunkts übereinstimmen.
*Erforderliche Anzahl	Gibt die Anzahl der erforderlichen Angebote an.
Name der Callback-Funktion bei Erfolg	Definiert den Namen der Funktion, die aufgerufen werden soll, wenn die Methode zum Abrufen der Angebote erfolgreich ist.

Tabelle 37. Tags "Angebot erhalten" (Forts.)

Tagfeld	Beschreibung
Name der Callback-Funktion bei Fehler	Definiert den Namen der Funktion, die aufgerufen werden soll, wenn die Methode zum Abrufen der Angebote nicht erfolgreich ist.

Alle **Tagfelder** mit der Markierung ***** sind erforderlich.

Der Tag "Angebot erhalten" muss einer Seitengruppe zugeordnet sein, für deren Container `Default` eingestellt ist.

Bibliothek laden

Bei Verwendung des Tags "Bibliothek laden" wird die Interact-JavaScript-Bibliothek in den Anfangsabschnitt der Seite geladen.

Für den Tag "Bibliothek laden" werden keine Parameter bereitgestellt. Die Position der Bibliothek wird aus `Bibliothekspfad` unter **Anbietereinstellungen** abgerufen. Sie muss in einer Seitengruppe eingeschlossen sein, in der ein Container verwendet wird, für den `Head` eingestellt ist; außerdem muss sie auf jeder Seite ausgeführt werden, auf der das Interact-Tagging angewendet wird.

Wichtig: Keiner der anderen Tags funktioniert, wenn der Tag "Bibliothek laden" nicht eingeschlossen ist. Die Datei "interact.js" wird nicht geladen, wenn dieser Tag nicht eingeschlossen ist.

Ereignis senden

Mit dem Tag "Ereignis senden" kann jedes Ereignis ausgeführt werden, das im interaktiven Kanal definiert ist.

Für den Tag "Ereignis senden" stehen die folgenden Tagfelder zur Verfügung.

Tabelle 38. Tags "Ereignis senden"

Tagfeld	Beschreibung
*Sitzungs-ID	Gibt die Sitzungs-ID an.
*Ereignisname	Gibt den Namen des Ereignisses an. Der Name des Ereignisses muss mit dem im interaktiven Kanal definierten Ereignisnamen übereinstimmen. Bei diesem Namen braucht die Groß-/Kleinschreibung nicht berücksichtigt zu werden.
Name der Callback-Funktion bei Erfolg	Definiert den Namen der Funktion, die aufgerufen werden soll, wenn die Methode zum Senden des Ereignisses erfolgreich ist.
Name der Callback-Funktion bei Fehler	Definiert den Namen der Funktion, die aufgerufen werden soll, wenn die Methode zum Senden des Ereignisses nicht erfolgreich ist.

Alle **Tagfelder** mit der Markierung ***** sind erforderlich.

Optionale Parameter können mit der Funktion für benutzerdefinierte Tagfelder hinzugefügt werden. Benutzerdefinierte Tagnamen müssen aus dem Parameternamen, einem Doppelpunkt und dem Datentyp bestehen.

Zielgruppe festlegen

Mit dem Tag "Zielgruppe festlegen" können die Zielgruppen-ID und die Ebene für einen Besucher festgelegt werden.

Für den Tag "Zielgruppe festlegen" stehen die folgenden Tagfelder zur Verfügung.

Tabelle 39. Tags "Zielgruppe festlegen"

Tagfeld	Beschreibung
*Sitzungs-ID	Gibt die Sitzungs-ID an.
*Zielgruppen-ID	Gibt die Zielgruppen-ID an. Die Namen müssen mit den physischen Spaltennamen aller Tabellen übereinstimmen, in denen die Zielgruppen-ID enthalten ist. Die Zielgruppen-ID darf nicht mehr als 17 signifikante Ziffern enthalten. Wenn eine Zielgruppen-ID mehr als 17 signifikante Ziffern enthält, muss sie partitioniert oder durch eine Zeichenfolge ersetzt werden.
*Zielgruppenebene	Definiert die Zielgruppenebene.
Name der Callback-Funktion bei Erfolg	Definiert den Namen der Funktion, die aufgerufen werden soll, wenn die Methode Festlegen der Zielgruppe erfolgreich ist.
Name der Callback-Funktion bei Fehler	Definiert den Namen der Funktion, die aufgerufen werden soll, wenn die Methode Festlegen der Zielgruppe nicht erfolgreich ist.

Alle **Tagfelder** mit der Markierung * sind erforderlich.

Optionale Parameter können mit der Funktion für benutzerdefinierte Tagfelder hinzugefügt werden. Benutzerdefinierte Tagnamen müssen aus dem Parameternamen, einem Doppelpunkt und dem Datentyp bestehen.

Sitzung starten

Mit dem Tag "Sitzung starten" wird eine Websitzung erstellt und definiert.

Für den Tag "Sitzung starten" stehen die folgenden Tagfelder zur Verfügung.

Tabelle 40. Tags "Sitzung starten"

Tagfeld	Beschreibung
*Sitzungs-ID	Gibt die Sitzungs-ID an.
*Interact-Kanal	Definiert den Namen des interaktiven Kanals, auf den diese Sitzung verweist. Dieser Name muss exakt mit dem in Campaign definierten Namen des interaktiven Kanals übereinstimmen.
*Zielgruppen-ID	Gibt die Zielgruppen-ID an. Die Namen müssen mit den physischen Spaltennamen aller Tabellen übereinstimmen, in denen die Zielgruppen-ID enthalten ist.
*Zielgruppenebene	Definiert die Zielgruppenebene.
*Vorhandene Sitzung als Grundlage	Definiert, ob diese Sitzung eine neue oder eine vorhandene Sitzung verwendet.
*Debug	Aktiviert oder inaktiviert die Daten zur Fehlerbehebung.

Tabelle 40. Tags "Sitzung starten" (Forts.)

Tagfeld	Beschreibung
Name der Callback-Funktion bei Erfolg	Definiert den Namen der Funktion, die aufgerufen werden soll, wenn die Methode zum Starten der Sitzung erfolgreich ist.
Name der Callback-Funktion bei Fehler	Definiert den Namen der Funktion, die aufgerufen werden soll, wenn die Methode zum Starten der Sitzung nicht erfolgreich ist.

Alle **Tagfelder** mit der Markierung * sind erforderlich.

Optionale Parameter können mit der Funktion für benutzerdefinierte Tagfelder hinzugefügt werden. Benutzerdefinierte Tagnamen müssen aus dem Parameternamen, einem Doppelpunkt und dem Datentyp bestehen.

Der Tag "Sitzung starten" muss einer Seitengruppe zugeordnet sein, für deren Container `Default` eingestellt ist.

Beispiel für Tageinstellungen

An diesem Beispiel wird eine einfache Konfiguration der Einstellungen der Tags "Sitzung starten", "Ereignis senden", "Angebot erhalten" und "Sitzung beenden" veranschaulicht.

Für jeden Tag können Sie die Werte der Tagfelder über das Cookie mit der Cookiemethode oder über das JavaScript-Objekt mit der Methode "javascriptobject" abrufen.

Diese Tags unterstützen weitere Parameter, die in diesem Beispiel nicht dargestellt werden. Weitere Informationen zu zusätzlichen Parametern finden Sie im IBM Digital Data Exchange-Benutzerhandbuch.

Detaillierte Anwendungsfälle zur Interact- und Digital Data Exchange-Integration finden Sie unter https://www.ibm.com/developerworks/community/wikis/home?lang=en#!/wiki/W214f7731a379_4712_a1ce_5d7a833d4cca/page/IBM%20Interact%20and%20IBM%20Digital%20Data%20Exchange%20Integration.

Beispieltageinstellungen für "Sitzung starten"

Klicken Sie auf **Tags** > **IBM Tags** > **IBM Interact** > **Typ: Sitzung starten**, um einen Tag des Typs "Sitzung starten" zu erstellen. Bearbeiten Sie den Tag mit den folgenden Einstellungen.

Einstellungen für Sitzungs-ID

- **Methode**: `Constant`
- **Konstante**: `5555`
- **Datentyp**: `String`
- **Änderungswert**: `<null>`

Einstellungen für "Interaktiver Kanal"

- **Methode**: `Constant`
- **Konstante**: `WSCDemo`
- **Datentyp**: `String`

- **Änderungswert**: <null>

Einstellungen für "Zielgruppen-ID"
- **Methode**: Constant
- **Konstante**: USERS_ID,2002,numeric
- **Datentyp**: String
- **Änderungswert**: <null>

Einstellungen für "Zielgruppenebene"
- **Methode**: Constant
- **Konstante**: WSCUserId
- **Datentyp**: String
- **Änderungswert**: <null>

Einstellungen für "Vorhandene Sitzung als Grundlage"
- **Methode**: Constant
- **Konstante**: False
- **Datentyp**: Boolean
- **Änderungswert**: <null>

Debug
- **Methode**: Constant
- **Konstante**: True
- **Datentyp**: Boolean
- **Änderungswert**: <null>

Einstellungen für "Name der Callback-Funktion bei Erfolg"
- **Methode**: Unassigned
- **Wert**: <null>

Einstellungen für "Name der Callback-Funktion bei Fehler"
- **Methode**: Unassigned
- **Wert**: <null>

Beispieltageinstellungen für "Angebot erhalten"

Klicken Sie auf **Tags** > **IBM Tags** > **IBM Interact** > **Typ: Angebot erhalten**, um einen Tag des Typs "Angebot erhalten" zu erstellen. Bearbeiten Sie den Tag mit den folgenden Einstellungen.

Einstellungen für Sitzungs-ID
- **Methode**: Constant
- **Konstante**: 5555
- **Datentyp**: String
- **Änderungswert**: <null>

Einstellungen für "Name des Interaktionspunkts"
- **Methode**: Constant
- **Konstante**: AuroraHomepageHeaderBannerLeft

- **Datentyp**: String
- **Änderungswert**: <null>

Einstellungen für "Erforderliche Anzahl"
- **Methode**: Constant
- **Konstante**: 1
- **Datentyp**: integer
- **Änderungswert**: <null>

Einstellungen für "Name der Callback-Funktion bei Erfolg"
- **Methode**: Constant
- **Konstante**: onOfferReturnSuccess
- **Datentyp**: string
- **Änderungswert**: <null>

Einstellungen für "Name der Callback-Funktion bei Fehler"
- **Methode**: Constant
- **Konstante**: onOfferReturnError
- **Datentyp**: string
- **Änderungswert**: <null>

Beispieltageinstellungen für "Ereignis senden"

Klicken Sie auf **Tags** > **IBM Tags** > **IBM Interact** > **Typ: Ereignis senden**, um einen Tag des Typs "Ereignis senden" zu erstellen. Bearbeiten Sie den Tag mit den folgenden Einstellungen.

Einstellungen für Sitzungs-ID
- **Methode**: Constant
- **Konstante**: 5555
- **Datentyp**: String
- **Änderungswert**: <null>

Einstellungen für "Ereignisname"
- **Methode**: Constant
- **Konstante**: ACCEPTOFFER
- **Datentyp**: String
- **Änderungswert**: <null>

Einstellungen für "Name der Callback-Funktion bei Erfolg"
- **Methode**: Constant
- **Konstante**: onSuccessTestFunction
- **Datentyp**: String
- **Änderungswert**: <null>

Einstellungen für "Name der Callback-Funktion bei Fehler"
- **Methode**: Constant
- **Konstante**: onErrorTestFunction
- **Datentyp**: String

- **Änderungswert**: <null>

Einstellungen für zusätzliches Parameterfeld

- **Tagfeld**: UACIOfferTrackingCode:string
- **Methode**: JavaScriptObject
- **Objektname**: oa.treatmentCode
- **Datentyp**: String
- **Änderungswert**: <null>

Beispieltageinstellungen für "Sitzung beenden"

Klicken Sie auf **Tags** > **IBM Tags** > **IBM Interact** > **Typ: Sitzung beenden**, um einen Tag des Typs "Sitzung beenden" zu erstellen. Bearbeiten Sie den Tag mit den folgenden Einstellungen.

Einstellungen für Sitzungs-ID

- **Methode**: Constant
- **Konstante**: 5555
- **Datentyp**: String
- **Änderungswert**: <null>

Einstellungen für "Name der Callback-Funktion bei Erfolg"

- **Methode**: Unassigned
- **Wert**: <null>

Einstellungen für "Name der Callback-Funktion bei Fehler"

- **Methode**: Unassigned
- **Wert**: <null>

Beispielfunktionen

Für die Funktionen, die für die Einstellungen von "Name der Callback-Funktion bei Erfolg" und "Name der Callback-Funktion bei Fehler" verwendet werden, müssen Sie nur den Funktionsnamen angeben, wenn Sie einen neuen Tag erstellen, falls die Funktion bereits auf der Webseite vorhanden ist.

Sie können auch die Digital Data Exchange-Dienstprogramme verwenden, um Funktionen zu erstellen und zu den Webseiten hinzuzufügen.

Im folgenden Beispiel wird dargestellt, wie ein Angebot angezeigt werden soll, das von Interact auf der Webseite zurückgegeben wird. Sie müssen dieses Script auf der Seite einschließen oder den Codeausschnitt aus Digital Data Exchange verwenden, um es einzufügen.

```
<script>
oa = {treatmentCode: ""};
function acceptOffer(treatmentCode) {
oa.treatmentCode = treatmentCode;
}
function onOfferReturnSuccess(response) {
var offer = response.offerList[0].offers[0];
var attributes = offer.attributes;
var offerText = "";
var offerLinkURL = "#";
for(var i = 0; i<attributes.length; i++)
```

```
{
if(attributes[i].n == "OfferTerms")
{
offerText = attributes[i].v;
}
else if(attributes[i].n == "OfferLinkURL")
{
offerLinkURL = attributes[i].v;
}
}

var link = "<a href=\"'+offerLinkURL+"\" onclick=\"acceptOffer
('"+offer.treatmentCode+"')\">"+offerText+"</a>";
document.getElementById("offerContainer").innerHTML="
<div style=\"text-align:center;padding:
10px 0;background-color:#f5f5f5;\">"+link+"</div>";
}
function onOfferReturnError(response) {
(JSON.stringify(response));
}
</script>
```

Integrationskonfiguration überprüfen

Mit dem Digital Data Exchange-Testtool und der Datei `Interact.log` können Konfigurationsproblem behoben werden.

Mit dem Digital Data Exchange-Testtool können Sie die Enzyklopädie überprüfen, um festzustellen, ob die Konfiguration wie erwartet funktioniert. Klicken Sie zum Öffnen des Testtools in Digital Data Exchange auf **Bereitstellung** > **Testtool**.

Weitere Informationen zum Testtool finden Sie im IBM Digital Data Exchange-Benutzerhandbuch.

Sie können die Datei `Interact.log` anzeigen, um die Details der verschiedenen Interact-API-Aufrufe zu überprüfen, die ausgeführt wurden. Fügen Sie die Callback-Funktion bei Erfolg und die Callback-Funktion bei Fehler zu jedem Tag hinzu, um ein Debugging für die unterschiedlichen Aufrufe auszuführen.

Vor der Kontaktaufnahme zum Technical Support von IBM

Sollte sich ein Problem nicht mithilfe der Dokumentation beheben lassen, können sich die für den Support zuständigen Kontaktpersonen Ihres Unternehmens telefonisch an den technischen Support von IBM wenden. Damit wir Ihnen möglichst schnell und erfolgreich helfen können, beachten Sie dabei bitte die Anleitungen in diesem Abschnitt.

Wenn Sie wissen möchten, wer die für den Support zuständige Kontaktperson Ihres Unternehmens ist, wenden Sie sich an Ihren IBM Administrator.

Anmerkung: Der technische Support schreibt bzw. erstellt keine API-Scripts. Wenden Sie sich zur Unterstützung bei der Implementierung unserer API-Angebote an IBM Professional Services.

Zusammenzustellende Informationen

Halten Sie folgende Informationen bereit, wenn Sie sich an den technischen Support von IBM wenden:

- Kurze Beschreibung der Art Ihres Problems
- Detaillierte Fehlernachrichten, die beim Auftreten des Problems angezeigt werden.
- Schritte zum Reproduzieren des Problems
- Entsprechende Protokolldateien, Sitzungsdateien, Konfigurationsdateien und Daten
- Informationen zu Ihrer Produkt- und Systemumgebung von , die Sie entsprechend der Beschreibung unter „Systeminformationen“ abrufen können.

Systeminformationen

Bei Ihrem Anruf beim technischen Support von IBM werden Sie um verschiedene Informationen gebeten.

Sofern das Problem Sie nicht an der Anmeldung hindert, finden Sie einen Großteil der benötigten Daten auf der Info-Seite. Dort erhalten Sie Informationen zu Ihren installierten IBM Anwendungen.

Sie können über **Hilfe > Info** (Help > About) auf die Info-Seite zugreifen. Wenn Sie nicht auf die Info-Seite zugreifen können, überprüfen Sie, ob sich die Datei `version.txt` im Installationsverzeichnis Ihrer Anwendung befindet.

Kontaktinformationen für den technischen Support von IBM

Wenn Sie sich an den Technical Support von IBM wenden möchten, finden Sie weitere Informationen auf der Website des Technical Support für IBM Produkte (http://www.ibm.com/support/entry/portal/open_service_request).

Anmerkung: Um eine Supportanforderung einzugeben, müssen Sie sich mit einem IBM Account anmelden. Dieser Account muss mit Ihrer IBM Kundennummer verknüpft sein. Weitere Informationen zum Zuordnen Ihres Accounts zu Ihrer IBM Kundennummer erhalten Sie unter **Unterstützungsressourcen > Gültige Softwareunterstützung** im Support-Portal.

Bemerkungen

Die vorliegenden Informationen wurden für Produkte und Services entwickelt, die auf dem deutschen Markt angeboten werden.

Möglicherweise bietet IBM die in dieser Dokumentation beschriebenen Produkte, Services oder Funktionen in anderen Ländern nicht an. Informationen über die gegenwärtig im jeweiligen Land verfügbaren Produkte und Services sind beim zuständigen IBM Ansprechpartner erhältlich. Hinweise auf IBM Lizenzprogramme oder andere IBM Produkte bedeuten nicht, dass nur Programme, Produkte oder Services von IBM verwendet werden können. Anstelle der IBM Produkte, Programme oder Services können auch andere, ihnen äquivalente Produkte, Programme oder Services verwendet werden, solange diese keine gewerblichen oder anderen Schutzrechte von IBM verletzen. Die Verantwortung für den Betrieb von Produkten, Programmen und Services anderer Anbieter liegt beim Kunden.

Für die in diesem Handbuch beschriebenen Erzeugnisse und Verfahren kann es IBM Patente oder Patentanmeldungen geben. Mit der Auslieferung dieses Handbuchs ist keine Lizenzierung dieser Patente verbunden. Lizenzanforderungen sind schriftlich an folgende Adresse zu richten (Anfragen an diese Adresse müssen auf Englisch formuliert werden):

IBM Director of Licensing
IBM Europe, Middle East & Africa
Tour Descartes
2, avenue Gambetta
92066 Paris La Defense
France

For license inquiries regarding double-byte (DBCS) information, contact the IBM Intellectual Property Department in your country or send inquiries, in writing, to:

Trotz sorgfältiger Bearbeitung können technische Ungenauigkeiten oder Druckfehler in dieser Veröffentlichung nicht ausgeschlossen werden. Die hier enthaltenen Informationen werden in regelmäßigen Zeitabständen aktualisiert und als Neuausgabe veröffentlicht. IBM kann ohne weitere Mitteilung jederzeit Verbesserungen und/oder Änderungen an den in dieser Veröffentlichung beschriebenen Produkten und/oder Programmen vornehmen.

Verweise in diesen Informationen auf Websites anderer Anbieter werden lediglich als Service für den Kunden bereitgestellt und stellen keinerlei Billigung des Inhalts dieser Websites dar. Das über diese Websites verfügbare Material ist nicht Bestandteil des Materials für dieses IBM Produkt. Die Verwendung dieser Websites geschieht auf eigene Verantwortung.

Werden an IBM Informationen eingesandt, können diese beliebig verwendet werden, ohne dass eine Verpflichtung gegenüber dem Einsender entsteht.

Lizenznehmer des Programms, die Informationen zu diesem Produkt wünschen mit der Zielsetzung: (i) den Austausch von Informationen zwischen unabhängig voneinander erstellten Programmen und anderen Programmen (einschließlich des vorliegenden Programms) sowie (ii) die gemeinsame Nutzung der ausgetauschten Informationen zu ermöglichen, wenden sich an folgende Adresse:

IBM Corporation
B1WA LKG1
550 King Street
Littleton, MA 01460-1250
U.S.A.

Die Bereitstellung dieser Informationen kann unter Umständen von bestimmten Bedingungen - in einigen Fällen auch von der Zahlung einer Gebühr - abhängig sein.

Die Lieferung des im Dokument aufgeführten Lizenzprogramms sowie des zugehörigen Lizenzmaterials erfolgt auf der Basis der IBM Rahmenvereinbarung bzw. der Allgemeinen Geschäftsbedingungen von IBM, der IBM Internationalen Nutzungsbedingungen für Programmpakete oder einer äquivalenten Vereinbarung.

Alle in diesem Dokument enthaltenen Leistungsdaten stammen aus einer kontrollierten Umgebung. Die Ergebnisse, die in anderen Betriebsumgebungen erzielt werden, können daher erheblich von den hier erzielten Ergebnissen abweichen. Einige Daten stammen möglicherweise von Systemen, deren Entwicklung noch nicht abgeschlossen ist. Eine Gewährleistung, dass diese Daten auch in allgemein verfügbaren Systemen erzielt werden, kann nicht gegeben werden. Darüber hinaus wurden einige Daten unter Umständen durch Extrapolation berechnet. Die tatsächlichen Ergebnisse können davon abweichen. Benutzer dieses Dokuments sollten die entsprechenden Daten in ihrer spezifischen Umgebung prüfen.

Alle Informationen zu Produkten anderer Anbieter stammen von den Anbietern der aufgeführten Produkte, deren veröffentlichten Ankündigungen oder anderen allgemein verfügbaren Quellen. IBM hat diese Produkte nicht getestet und kann daher keine Aussagen zu Leistung, Kompatibilität oder anderen Merkmalen machen. Fragen zu den Leistungsmerkmalen von Produkten anderer Anbieter sind an den jeweiligen Anbieter zu richten.

Aussagen über Pläne und Absichten von IBM unterliegen Änderungen oder können zurückgenommen werden und repräsentieren nur die Ziele von IBM.

Alle von IBM angegebenen Preise sind empfohlene Richtpreise und können jederzeit ohne weitere Mitteilung geändert werden. Händlerpreise können unter Umständen von den hier genannten Preisen abweichen.

Diese Veröffentlichung enthält Beispiele für Daten und Berichte des alltäglichen Geschäftsablaufs. Sie sollen nur die Funktionen des Lizenzprogramms illustrieren und können Namen von Personen, Firmen, Marken oder Produkten enthalten. Alle diese Namen sind frei erfunden; Ähnlichkeiten mit tatsächlichen Namen und Adressen sind rein zufällig.

COPYRIGHTLIZENZ:

Diese Veröffentlichung enthält Beispielanwendungsprogramme, die in Quellensprache geschrieben sind und Programmiertechniken in verschiedenen Betriebsumgebungen veranschaulichen. Sie dürfen diese Beispielprogramme kostenlos kopieren, ändern und verteilen, wenn dies zu dem Zweck geschieht, Anwendungsprogramme zu entwickeln, zu verwenden, zu vermarkten oder zu verteilen, die mit der Anwendungsprogrammierschnittstelle für die Betriebsumgebung konform sind, für die diese Beispielprogramme geschrieben werden. Diese Beispiele wurden nicht unter allen denkbaren Bedingungen getestet. Daher kann IBM die Zuverlässigkeit, Wartungsfreundlichkeit oder Funktion dieser Programme weder zusagen noch

gewährleisten. Die Beispielprogramme werden ohne Wartung (auf "as-is"-Basis) und ohne jegliche Gewährleistung zur Verfügung gestellt. IBM übernimmt keine Haftung für Schäden, die durch die Verwendung der Beispielprogramme entstehen.

Marken

IBM, das IBM Logo und ibm.com sind Marken oder eingetragene Marken der IBM Corporation in den USA und/oder anderen Ländern. Weitere Produkt- und Servicenamen können Marken von IBM oder anderen Unternehmen sein. Eine aktuelle Liste der IBM Marken finden Sie auf der Webseite "Copyright and trademark information" unter www.ibm.com/legal/copytrade.shtml.

Hinweise zu Datenschutzrichtlinien und Nutzungsbedingungen

IBM Softwareprodukte, einschließlich Software as a Service-Lösungen ("Softwareangebote"), können Cookies oder andere Technologien verwenden, um Informationen zur Produktnutzung zu erfassen, die Endbenutzererfahrung zu verbessern und Interaktionen mit dem Endbenutzer anzupassen oder zu anderen Zwecken. Ein Cookie ist ein Datenelement, das von einer Website an Ihren Browser gesendet wird und dann als Tag auf Ihrem Computer gespeichert werden kann, mit dem Ihr Computer identifiziert wird. Häufig werden von diesen Cookies keine personenbezogenen Daten erfasst. Für den Fall, dass Sie mit einem von Ihnen genutzten Softwareangebot mit Cookies und ähnlichen Technologien personenbezogene Daten erfassen können, informieren wir Sie nachstehend über die entsprechenden Spezifikationen.

Abhängig von den implementierten Konfigurationen kann dieses Softwareangebot Sitzungscookies und permanente Cookies verwenden, mit denen der Benutzername des Benutzers und andere personenbezogene Daten zum Zwecke des Sitzungsmanagements, zur Verbesserung der Benutzerfreundlichkeit und zu anderen funktionsbezogenen Zwecken sowie zur Nutzungsüberwachung erfasst werden. Diese Cookies können inaktiviert werden, wodurch dann aber die von ihnen unterstützte Funktionalität nicht mehr zur Verfügung steht.

In verschiedenen Rechtsordnungen ist die Erfassung personenbezogener Daten durch Cookies und ähnliche Technologien gesetzlich geregelt. Falls die für dieses Softwareangebot implementierte Konfiguration Ihnen als Kunden die Möglichkeit zur Erfassung personenbezogener Daten mit Cookies und anderen Technologien bietet, sollten Sie sich über eventuell geltende Gesetze zu einer solchen Datenerfassung beraten lassen. Diese Beratung sollte gegebenenfalls auch Anforderungen hinsichtlich erforderlicher Hinweise oder Zustimmungen berücksichtigen.

IBM setzt voraus, dass Kunden folgende Bedingungen erfüllen: (1) Sie stellen einen klar erkennbaren und auffälligen Link zu den Nutzungsbedingungen der Kundenwebsite (z. B. Datenschutzerklärung) bereit. Dieser Link muss wiederum einen Link zu der Vorgehensweise von IBM und des Kunden bei der Datenerhebung und Datennutzung umfassen. (2) Sie weisen darauf hin, dass Cookies und Clear GIFs/Web-Beacons von IBM im Auftrag des Kunden auf dem Computer des Besuchers platziert werden. Dieser Hinweis muss eine Erläuterung hinsichtlich des Zwecks dieser Technologie umfassen. (3) Sie müssen in dem gesetzlich vorgeschriebenen Umfang die Einwilligung von Websitebesuchern einholen, bevor Cookies und Clear GIFs/Web-Beacons vom Kunden oder von IBM im Auftrag des Kunden auf den Geräten der Websitebesucher platziert werden.

Weitere Informationen zur Verwendung verschiedener Technologien einschließlich der Verwendung von Cookies zu diesen Zwecken finden Sie im IBM Online Privacy Statement unter der Webadresse http://www.ibm.com/privacy/details/us/en im Abschnitt mit dem Titel "Cookies, Web Beacons and Other Technologies".