IBM Unica Interact
Versión 8 Release 6
25 de mayo de 2012

Guía del administrador

Nota

Antes de utilizar esta información y el producto al que da soporte, lea la información del apartado "Avisos" en la página 263.

Esta edición se aplica a la versión 8, release 5, modificación 0 de IBM Unica Interact (número de producto 5725-D22) y a todos los releases y modificaciones posteriores hasta que se indique lo contrario en nuevas ediciones.

Contenido

Capítulo 1. Administración de IBM Unica Interact

La administración de Interact consta de varias tareas. Estas tareas son, entre otras:

- Mantenimiento de usuarios y roles
- Mantenimiento de orígenes de datos
- Configuración de las características de presentación de ofertas opcionales de Interact
- Supervisión y mantenimiento del rendimiento de entorno de ejecución

Antes de empezar a administrar Interact, debe conocer algunos conceptos clave sobre cómo funciona Interact que le facilitarán las tareas. En las siguientes secciones se describen las tareas administrativas asociadas con Interact.

En la segunda parte de esta guía se describen las interfaces de programación de aplicaciones (API) disponibles con Interact: la API de Interact, la API de ExternalCallout y la API de aprendizaje.

Conceptos clave de Interact

En esta sección se describen algunos de los conceptos clave que debe conocer antes de empezar a trabajar con Interact.

Niveles de audiencia

Un nivel de audiencia es una colección de identificadores que pueden ser el objetivo de una campaña. Por ejemplo, un conjunto de campañas puede utilizar los niveles de audiencia "Unidad familiar", "Posible cliente", "Cliente" y "Cuenta." Cada uno de estos niveles representa una determinada vista de los datos de marketing disponibles para una campaña.

Los niveles de audiencia normalmente se organizan jerárquicamente. Utilizando los ejemplos anteriores:

- La unidad familiar se encuentra en la parte superior de la jerarquía, y cada unidad familiar puede contener varios clientes, así como uno o varios posibles clientes.
- Cliente es el siguiente nivel en la jerarquía, y cada cliente puede tener varias cuentas.
- Cuenta está en la parte inferior de la jerarquía.

En los entornos interempresariales existen otros ejemplos más complejos de jerarquías de audiencia, donde debe haber niveles de audiencia para las empresas, las compañías, las divisiones, los grupos, las personas, las cuentas, etc.

Estos niveles de audiencia puede tener diferentes relaciones entre ellos, por ejemplo de uno a uno, de varios a uno, o de varios a varios. Al definir los niveles de audiencia, permite que estos conceptos se representen en Campaign, para que los usuarios puedan gestionar las relaciones entre las distintas audiencias para poder definir objetivos. Por ejemplo, aunque puede haber varios posibles clientes por unidad familiar, puede limitar el envío de correos a un posible cliente por unidad familiar.

Entorno de diseño

El entorno de diseño es donde se realiza la mayor parte de la configuración de Interact. En el entorno de diseño se definen eventos, puntos de interacción, segmentos inteligentes y reglas de tratamiento. Después de configurar estos componentes, los despliega en el entorno de ejecución.

El entorno de diseño se instala con la aplicación web de Campaign.

Eventos

Un evento es una acción, realizada por un visitante, que desencadena una acción en el entorno de ejecución, como colocar un visitante en un segmento, presentar una oferta, o registrar datos.

En primer lugar, los eventos se crean en un canal interactivo y a continuación los desencadena una llamada a la API de Interact utilizando el método `postEvent`. Un evento puede producir una o más de las acciones siguientes definidas en el entorno de diseño de Interact:

- Activar re-segmentación
- Registrar contacto de oferta
- Registrar aceptación de oferta
- Registrar rechazo de oferta

También puede utilizar eventos para desencadenar acciones definidas por el método `postEvent`, incluyendo registrar datos en una tabla, incluir datos para aprendizaje o desencadenar diagramas de flujo individuales.

Para su comodidad, los eventos se pueden organizar en categorías en el entorno de diseño. Las categorías no tienen ninguna finalidad funcional en el entorno de ejecución.

Canales interactivos

Un canal interactivo es una representación en Campaign de un punto de encuentro donde el método de la interfaz es un diálogo interactivo. Esta representación de software se utiliza para coordinar todos los objetos, datos y recursos de servidor implicados en el marketing interactivo.

Un canal interactivo es una herramienta que se utiliza para definir eventos y puntos de interacción. También puede acceder a los informes de un canal interactivo desde la pestaña Análisis de ese canal interactivo.

Los canales interactivos también contienen asignaciones de servidor de preparación y tiempo de ejecución de producción. Puede crear varios canales interactivos para organizar los eventos y puntos de interacción si tiene solo un conjunto de servidores de preparación y tiempo de ejecución de producción, o para dividir los eventos y puntos de interacción por sistema orientado al cliente.

Diagramas de flujo interactivos

Un diagrama de flujo interactivo está relacionado con un diagrama de flujo por lotes de Campaign, pero no es exactamente igual. Los diagramas de flujo interactivos realizan la misma función principal que los diagramas de flujo de proceso por lotes, es decir, dividir los clientes en grupos conocidos como segmentos. Sin embargo, en el caso de los diagramas interactivos, los grupos son segmentos inteligentes. Interact utiliza estos diagramas de flujo interactivos para

asignar un perfil a un segmento con un evento de comportamiento o del sistema que indica que se requiere volver a segmentar un visitante.

Los diagramas de flujo interactivos contienen un subconjunto de los procesos de los diagramas de flujo de proceso por lotes, así como procesos específicos de los diagramas de flujo interactivos.

Nota: Los diagramas de flujo interactivos se pueden crear solo en una sesión de Campaign.

Puntos de interacción

Un punto de interacción es un lugar del punto de encuentro donde desea presentar una oferta. Los puntos de interacción contienen contenido de completado predeterminado en ubicaciones en los que el entorno de ejecución no tiene otro contenido apto para presentar.

Los puntos de interacción se pueden organizar en zonas.

Ofertas

Una oferta representa un mensaje de marketing individual, que puede entregarse de varias formas.

En Campaign, puede crear ofertas que se utilicen en una o varias campañas.

Las ofertas son reutilizables:

- en distintas campañas;
- en distintos puntos en el tiempo;
- para distintos grupos de personas (celdas);
- como “versiones” diferentes variando los campos parametrizados de la oferta.

Puede asignar ofertas a las celdas de destino en los diagramas de flujo utilizando uno de los procesos de contacto, y realizar un seguimiento de los resultados de la campaña mediante la captura de datos sobre los clientes que han recibido la oferta y los clientes que han respondido.

Perfiles

Un perfil es el conjunto de datos de cliente utilizados por el entorno de ejecución. Estos datos pueden ser un subconjunto de los datos de cliente disponibles en la base de datos de cliente, los datos recopilados en tiempo real o una combinación de ambos. Estos datos se utilizan con los fines siguientes:

- Para asignar un cliente a uno o más segmentos inteligentes en escenarios de interacción en tiempo real.

  Necesita un conjunto de datos de perfil para cada nivel de audiencia por el que desee segmentar. Por ejemplo, si está segmentando por ubicación, podría incluir sólo el código postal del cliente de toda la información de dirección que tenga.
- Para personalizar ofertas
- Como atributos para realizar seguimiento para el aprendizaje

  Por ejemplo, puede configurar Interact para supervisar el estado civil de un visitante y cuántos visitantes de cada estado aceptan una oferta específica. A continuación, el entorno de ejecución puede utilizar esa información para acotar la selección de oferta.

Estos datos son de solo lectura para el entorno de ejecución.

Entorno de ejecución

El entorno de ejecución conecta al punto de encuentro y realiza interacciones. El entorno de ejecución puede consistir en uno o varios servidores de ejecución conectados al punto de encuentro.

El entorno de ejecución utiliza la información desplegada desde el entorno de diseño en combinación con la API de Interact para presentar ofertas para su punto de encuentro.

Sesiones de ejecución

Existe una sesión de ejecución en el servidor de ejecución para cada visitante a su punto de contacto. Esta sesión contiene todos los datos del visitante que utiliza el entorno de ejecución para asignar visitantes a segmentos y recomendar ofertas.

Puede crear una sesión de ejecución al utilizar la llamada `startSession`.

Puntos de encuentro

Un punto de encuentro es una aplicación o ubicación donde puede interactuar con un cliente. Un punto de encuentro puede ser un canal donde el cliente inicia el contacto (una interacción "entrante") o donde se contacta con el cliente (una interacción "saliente"). Ejemplos comunes son los sitios web y las aplicaciones de centro de atención al cliente. Mediante la API de Interact, puede integrar Interact con los puntos de encuentro para presentar ofertas a los clientes en función de la acción que realicen en el punto de encuentro. Los puntos de encuentro también se denominan sistemas orientados al cliente.

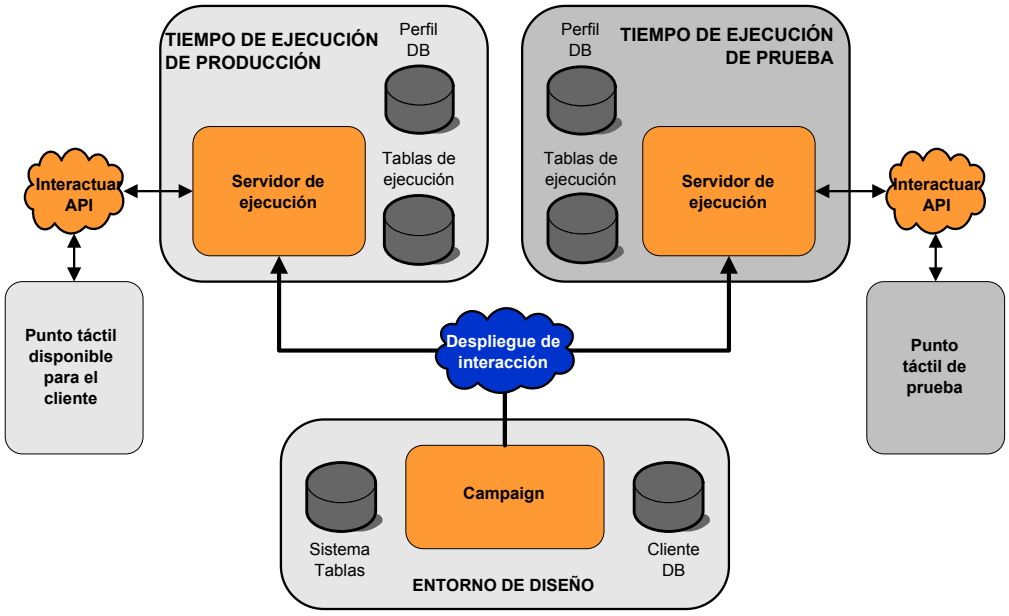
Reglas de tratamiento

Las reglas de tratamiento asignan una oferta a un segmento inteligente. Estas asignaciones las restringe adicionalmente la zona definida de forma personalizada que se asocia a la oferta en la regla de tratamiento. Por ejemplo, puede tener un conjunto de ofertas que asigna a un segmento inteligente en la zona "inicio de sesión", pero un conjunto de ofertas distintas para el mismo segmento en la zona "después de compra". Las reglas de tratamiento se definen en una pestaña de estrategia de interacción de una campaña.

Cada regla de tratamiento también tiene una puntuación de marketing. Si se asigna un cliente a más de un segmento, y por lo tanto más de una oferta es aplicable, las puntuaciones de marketing ayudan a definir qué oferta sugiere Interact. Las ofertas que sugiere el entorno de ejecución pueden resultar influenciadas por un módulo de aprendizaje, una lista de supresión de ofertas y asignaciones de ofertas globales e individuales.

Arquitectura de Interact

El entorno de Interact está formado por dos componentes principales, como mínimo: el entorno de diseño y el entorno de ejecución. También puede tener servidores de ejecución de pruebas opcionales. En la siguiente figura se muestra la visión general de la arquitectura de alto nivel.



El entorno de diseño es donde realiza la mayor parte de la configuración de Interact. El entorno de diseño se instala con la aplicación web de Campaign, y hace referencia a las tablas del sistema de Campaign y las bases de datos del cliente. Puede utilizar el entorno de diseño para definir los puntos de interacción y los eventos que utiliza con la API.

Después de diseñar y configurar cómo desea que el entorno de ejecución maneje las interacciones con el cliente, puede desplegar los datos en un grupo de servidores de preparación para realizar pruebas o en un grupo de servidores de ejecución de producción para la interacción con el cliente en tiempo real.

La API de Interact proporciona la conexión entre el punto de encuentro y el servidor de ejecución. Puede hacer referencia a los objetos (los puntos de interacción y los eventos) creados en el entorno de diseño con la API de Interact y utilizarlos para solicitar información al servidor de ejecución.

Consideraciones de red de Interact

Una instalación de producción de Interact abarca al menos dos máquinas. En un entorno de producción de gran volumen, con varios servidores de ejecución de Interact y bases de datos distribuidas, la instalación puede llegar a abarcar docenas de máquinas. Para garantizar el mejor rendimiento, se deben tener en cuenta varios requisitos de topología de red.

- Si la implementación de la API de Interact inicia y finaliza sesiones en la misma llamada, por ejemplo:

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

  no es necesario que habilite la persistencia de sesiones (sesiones permanentes) entre el equilibrador de carga y los servidores de ejecución de Interact. Puede configurar la gestión de sesiones de los servidores de ejecución de Interact para la memoria caché local.
- Si la implementación de la API de Interact utiliza varias llamadas para iniciar y finalizar sesiones, por ejemplo:

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

y utiliza un equilibrador de carga para los servidores de ejecución de Interact, debe habilitar algún tipo de persistencia para el equilibrador de carga (también conocidas como sesiones permanentes). Si no es posible, o si no utiliza un equilibrador de carga, configure la gestión de sesiones de los servidores de Interact para un `cacheType` distribuido. Si está utilizando una memoria caché distribuida, todos los servidores de ejecución de Interact deben poder comunicarse mediante multidifusión. Es posible que deba ajustar la red para que la comunicación entre los servidores de Interact que utilizan la misma dirección IP y el mismo puerto no afecte al rendimiento del sistema. Un equilibrador de carga con sesiones permanentes proporciona un mejor rendimiento que el uso de la memoria caché distribuida.

- Si tiene varios grupos de servidores que utilizan un `cacheType` distribuido, cada uno debe utilizar un puerto de multidifusión exclusivo. Se recomienda utilizar un puerto de multidifusión exclusivo y una dirección para cada grupo de servidores.
- Mantenga los servidores de Interact del entorno de ejecución, la Marketing Platform, los equilibradores de carga y el punto de encuentro en la misma ubicación geográfica para garantizar el máximo rendimiento. El tiempo de diseño y el tiempo de ejecución pueden estar en distintas ubicaciones geográficas, pero el despliegue será más lento.
- Utilice una conexión de red rápida (al menos 1 Gb) entre el grupo de servidores de Interact y su punto de encuentro asociado.
- El tiempo de diseño requiere acceso http o https al tiempo de ejecución para completar las tareas de despliegue. Los cortafuegos u otras aplicaciones de red deben estar configurados para permitir el despliegue. Es posible que deba ampliar las longitudes de tiempo de espera HTTP entre el entorno de diseño y los entornos de ejecución si tiene grandes despliegues.
- El módulo de historial de respuestas y contactos requiere acceso a la base de datos de tiempo de diseño (tablas del sistema de Campaign) y acceso a la base de datos de ejecución (las tablas de ejecución de Interact). Debe configurar las bases de datos y la red correctamente para que se produzca esta transferencia de datos.

En una instalación de prueba o preparación, puede instalar el tiempo de diseño y el tiempo de ejecución de Interact en la misma máquina. Este escenario no se recomienda para un entorno de producción.

Capítulo 2. Configuración de usuarios de IBM Unica Interact

Interact requiere que configure dos conjuntos de usuarios: los usuarios del entorno de ejecución y los usuarios del entorno de diseño.

- Los **usuarios de tiempo de ejecución** se crean en la Marketing Platform configurada para trabajar con los servidores de ejecución.
- Los **usuarios de tiempo de diseño** son los usuarios de Campaign. Configure la seguridad de los distintos miembros de su equipo de diseño como para Campaign.

Configuración del usuario del entorno de ejecución

Después de instalar Interact, debe configurar al menos un usuario de Interact, el usuario del entorno de ejecución.

El usuario del entorno de ejecución proporciona acceso a las tablas de ejecución. Este es el nombre de usuario y la contraseña que utiliza al desplegar los canales interactivos. El servidor de ejecución utiliza la autenticación JDBC del servidor de aplicaciones web para las credenciales de la base de datos. Por lo tanto, no tendrá que añadir orígenes de datos de entorno de ejecución al usuario de entorno de ejecución.

Importante: Todos los servidores de ejecución que pertenecen al mismo grupo de servidores deben compartir las mismas credenciales de usuario. Si tiene instancias de Marketing Platform diferentes para cada servidor de ejecución, debe crear el mismo usuario y la misma contraseña en cada una de ellas.

Si está utilizando una utilidad de carga de base de datos, debe definir las tablas de ejecución como un origen de datos con credenciales de inicio de sesión para el usuario del entorno de ejecución. El nombre de este origen de datos debe ser `systemTablesDataSource`.

Si habilita la seguridad para la supervisión JMX con el protocolo JMXMP, necesitará un usuario aparte para la seguridad de supervisión JMX.

Configuración de usuarios del entorno de diseño

Los usuarios del entorno de diseño son usuarios de Campaign. Puede configurar los usuarios del entorno de diseño de la forma en que configura los permisos de rol de Campaign.

Debe otorgar a los usuarios de Campaign que tengan permiso para editar los diagramas de flujo interactivos acceso al origen de datos de tablas de ejecución de prueba.

Si tiene Interact instalado y configurado, las siguientes opciones adicionales están disponibles para la Política global predeterminada y las nuevas políticas. Recuerde que algunos usuarios del entorno de diseño también requieren algunos permisos de Campaign como Macros personalizadas.

Categoría	Permisos
Campañas	• Ver estrategias de interacción de campaña: capacidad de ver pero no de editar las pestañas de estrategia de interacción de una campaña. • Editar estrategias de interacción de campaña: capacidad de realizar cambios en las pestañas de estrategia de interacción, incluidas las reglas de tratamiento. • Suprimir estrategias de interacción de campaña: capacidad de eliminar pestañas de estrategia de interacción de las campañas. La supresión de una pestaña de estrategia de interacción está restringida si la estrategia de interacción se ha incluido en un despliegue de canal interactivo. • Añadir estrategias de interacción de campaña: capacidad de crear nuevas pestañas de estrategia de interacción en una campaña. • Inicializar despliegues de estrategia de interaccción de campaña: capacidad de marcar una pestaña de estrategia de interacción para un despliegue o una anulación de despliegue.
Canales interactivos	• Desplegar canales interactivos: capacidad de desplegar un canal interactivo en los entornos de ejecución de Interact. • Editar canales interactivos: capacidad de realizar cambios en la pestaña de resumen de canales interactivos. • Suprimir canales interactivos: capacidad de eliminar canales interactivos. La supresión de canales interactivos está restringida si el canal interactivo se ha desplegado. • Ver canales interactivos: capacidad de ver pero no de editar canales interactivos. • Añadir canales interactivos: capacidad de crear nuevos canales interactivos. • Ver informes de canales interactivos: capacidad de ver la pestaña de análisis del canal interactivo. • Añadir objetos hijo de canal interactivo: capacidad de añadir puntos de interacción, zonas, eventos y categorías.

Categoría	Permisos
Sesiones	• Ver diagramas de flujo interactivos: capacidad de ver un diagrama de flujo interactivo en una sesión. • Añadir diagramas de flujo interactivos: capacidad de crear nuevos diagramas de flujo interactivos en una sesión. • Editar diagramas de flujo interactivos: capacidad de realizar cambios en diagramas de flujo interactivos. • Suprimir diagramas de flujo interactivos: capacidad de eliminar diagramas de flujo interactivos. La supresión de diagramas de flujo interactivos está restringida si el canal interactivo al que está asignado este diagrama de flujo interactivo se ha desplegado. • Copiar diagramas de flujo interactivos: capacidad de copiar diagramas de flujo interactivos. • Realizar ejecución de prueba de diagramas de flujo interactivos: capacidad de iniciar un ejecución de prueba de un diagrama de flujo interactivo. • Revisar diagramas de flujo interactivos: capacidad de ver un diagrama de flujo interactivo y abrir procesos para ver valores, pero sin poder realizar cambios. • Desplegar diagramas de flujo interactivos: capacidad de marcar un diagrama de flujo interactivo para su despliegue o anulación de despliegue.

Permisos del entorno de diseño de ejemplo

Por ejemplo, puede crear dos roles, uno para personas que crean diagramas de flujo interactivos, y uno para las personas que definen las estrategias de interacción. Cada sección lista los permisos otorgados al rol.

Rol de diagrama de flujo interactivo

Macro personalizada

- Añadir macros personalizadas
- Editar macros personalizadas
- Utilizar macros personalizadas

Campos derivados

- Añadir campos derivados
- Editar campos derivados
- Utilizar campos derivados

Plantilla de diagrama de flujo

- Pegar plantillas

Plantilla de segmento

- Añadir segmentos
- Editar segmentos

Sesión

- Ver resumen de sesión

- Ver diagramas de flujo interactivos
- Añadir diagramas de flujo interactivos
- Editar diagramas de flujo interactivos
- Copiar diagramas de flujo interactivo
- Realizar ejecución de prueba de diagramas de flujo interactivos
- Desplegar diagramas de flujo interactivos

Rol de estrategia de interacción

Campaña
- Ver resumen de campaña
- Gestionar celdas objetivo de campaña
- Ver estrategias de interacción de campaña
- Editar estrategias de interacción de campaña
- Añadir estrategias de interacción de campaña
- Iniciar despliegues de estrategia de interacción de campaña

Oferta
- Ver resumen de oferta

Plantilla de segmento
- Ver resumen de segmento

Sesión
- Revisar diagramas de flujo interactivos

Capítulo 3. Gestión de orígenes de datos de Interact

Interact requiere varios orígenes de datos para funcionar correctamente. Algunos orígenes de datos contienen la información que Interact necesita para funcionar, mientras que otros orígenes de datos contienen los datos.

En las siguientes secciones se describen los orígenes de datos de Interact, incluida la información que necesita para configurarlos correctamente y algunas sugerencias para su mantenimiento.

Cómo trabajar con los orígenes de datos de Interact

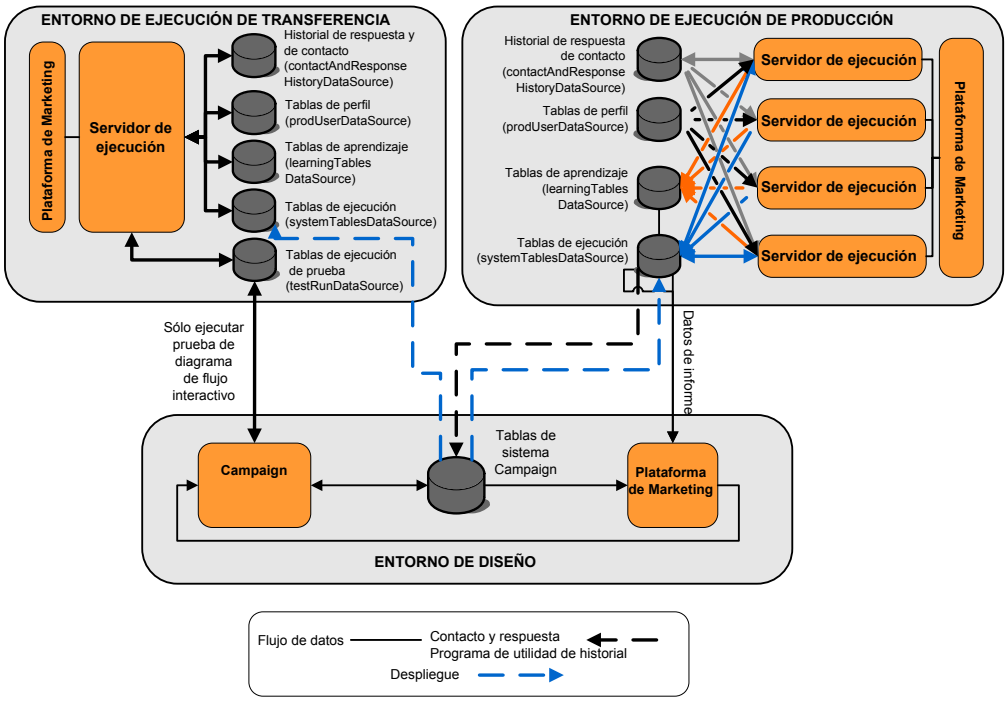
Interact requiere varios conjuntos de datos para funcionar.

- **Tablas del sistema de Campaign** — aparte de todos los datos de Campaign, las tablas del sistema de Campaign contienen datos para los componentes de Interact que crea en el entorno de diseño como, por ejemplo, las reglas de tratamiento y los canales interactivos. El entorno de diseño y las tablas del sistema de Campaign comparten el mismo esquema y la misma base de datos física.
- **Tablas de ejecución** -(`systemTablesDataSource`) contiene los datos de despliegue del entorno de diseño, las tablas de preparación del historial de contactos y respuestas, y las estadísticas de tiempo de ejecución.
- **Tablas de perfil** -(`prodUserDataSource`) contiene los datos de cliente, además de la información recopilada en tiempo real, que necesitan los diagramas de flujo interactivos para colocar correctamente los visitantes en segmentos inteligentes. Si se basa solamente en los datos en tiempo real, no necesita las tablas de perfil. Si utiliza las tablas de perfil, debe tener al menos una tabla de perfil por nivel de audiencia utilizado por el canal interactivo.

  Las tablas de perfil también pueden contener las tablas utilizadas para aumentar la presentación de ofertas, incluidas las tablas de supresión de oferta, alteración temporal de puntuaciones y asignación de ofertas globales e individuales.
- **Tablas de ejecución de prueba** -(`testRunDataSource`) contiene una muestra de todos los datos que necesitan los diagramas de flujo interactivos para colocar los visitantes en segmentos inteligentes, incluidos los datos que imitan qué se recopila en tiempo real durante una interacción. Estas tablas sólo son necesarias para el grupo de servidores designado como grupo de servidores de ejecución de prueba para el entorno de diseño.
- **Tablas de aprendizaje** -(`learningTablesDataSource`) contiene todos los datos recopilados por la utilidad de aprendizaje incorporada. Estas tablas pueden incluir una tabla que define atributos dinámicos. Si no utiliza el aprendizaje o utiliza una utilidad de aprendizaje externa que ha creado, no necesita las tablas de aprendizaje.
- **Historial de respuestas y contactos para la respuesta de sesiones cruzadas** -(`contactAndResponseHistoryDataSource`) son las tablas del historial de contactos de Campaign o una copia de ellas. Si no utiliza la característica de respuesta de sesiones cruzadas, no es necesario configurar estas tablas del historial de contactos.

Bases de datos y las aplicaciones

El siguiente diagrama muestra los posibles orígenes de datos de Interact y cómo se relacionan con las aplicaciones de IBM® Unica .



- El grupo de servidores de ejecución de prueba y Campaign acceden a las tablas de ejecución de prueba.
- Las tablas de ejecución de prueba sólo se utilizan para las ejecuciones de prueba de diagrama de flujo interactivo.
- Cuando utiliza un servidor de ejecución para probar un despliegue, incluida la API de Interact, el servidor de ejecución utiliza las tablas de perfil para los datos.
- Si configura el módulo de historial de contactos y respuestas, el módulo utiliza un proceso ETL (Extracción, Transformación y Carga) de fondo para mover los datos de las tablas de preparación de ejecución a las tablas de historial de contactos y respuestas de Campaign.
- La función de creación de informes consulta los datos de las tablas de aprendizaje, las tablas de ejecución y las tablas del sistema de Campaign para visualizar informes en Campaign.

Debe configurar los entornos de ejecución de prueba para que utilicen un conjunto de tablas diferente al de los entornos de ejecución de producción. Al utilizar tablas diferentes para la preparación y la producción, puede mantener separados los resultados de las pruebas y los resultados reales. Tenga en cuenta que el módulo de historial de contactos y respuestas siempre inserta datos en las tablas de historial de contactos y respuestas de Campaign (Campaign no tiene tablas de historial de contactos y respuestas de prueba). Si utiliza tablas de aprendizaje aparte para el entorno de ejecución de prueba y desea ver los resultados en los informes, necesitará una instancia independiente de IBM Cognos BI para ejecutar los informes de aprendizaje para el entorno de prueba.

Tablas del sistema de Campaign

Al instalar el entorno de diseño, también crea nuevas tablas específicas de Interact en las tablas del sistema de Campaign.

Si habilita el módulo de historial de contactos y respuestas, el módulo copia el historial de contactos y respuestas de las tablas de preparación de las tablas de ejecución en las tablas del sistema de Campaign. Las tablas predeterminadas son `UA_ContactHistory`, `UA_DtlContactHist` y `UA_ResponseHistory`, pero el módulo del historial de contactos y respuestas utilizará las tablas correlacionadas en Campaign para las tablas de historial de contactos y respuestas.

Si utiliza las tablas de ofertas globales y las tablas de sustituciones globales para asignar ofertas, es posible que necesite completar la tabla `UACI_ICBatchOffers` de las tablas del sistema de Campaign si utiliza ofertas que no están incluidas en las reglas de tratamiento para el canal interactivo.

Tablas de ejecución

Si tiene más de un nivel de audiencia, debe crear tablas de preparación para los datos del historial de contactos y respuestas para cada nivel de audiencia.

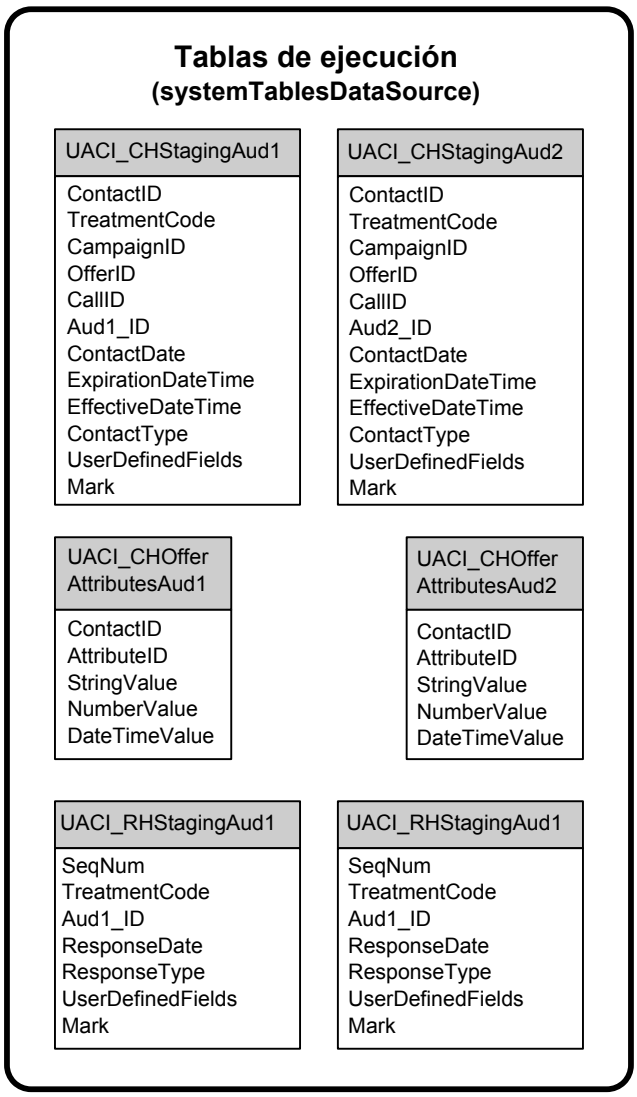
Los scripts SQL crean las siguientes tablas para el nivel de audiencia predeterminado:

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

Debe crear copias de estas tres tablas para cada uno de los niveles de audiencia en las tablas de ejecución.

Si las tablas del historial de contactos y respuestas de Campaign tienen campos definidos por el usuario, debe crear los mismos nombres y tipos de campo en las tablas `UACI_CHStaging` y `UACI_RHStaging`. Puede completar estos campos durante el tiempo de ejecución mediante la creación de pares nombre-valor del mismo nombre en los datos de sesión. Por ejemplo, las tablas del historial de contactos y respuestas contienen el campo `catalogID`. Debe añadir el campo `catalogID` a las tablas `UACI_CHStaging` y `UACI_RHStaging`. Posteriormente, la API de Interact completa este campo mediante la definición de un parámetro de evento como un par nombre-valor denominado `catalogID`. Los datos de sesión pueden proporcionarlos la tabla de perfil, los datos temporales, el aprendizaje o la API de Interact.

En el siguiente diagrama se incluyen las tablas de muestra para las audiencias Aud1 y Aud2. Este diagrama no incluye todas las tablas de la base de datos de tiempo de ejecución.



Todos los campos de las tablas son necesarios. Puede modificar `CustomerID` y `UserDefinedFields` para que coincidan con sus tablas del historial de contactos y respuestas de Campaign.

Tablas de ejecución de prueba

Las tablas de ejecución de prueba sólo se utilizan para las ejecuciones de prueba de diagramas de flujo interactivos. Las ejecuciones de prueba de diagramas de flujo interactivos deben probar su lógica de segmentación. Sólo necesita configurar una base de datos de ejecución de prueba para la instalación de Interact. Las tablas de ejecución de prueba no necesitan estar en una base de datos autónoma. Por ejemplo, puede utilizar tablas de datos de cliente para Campaign.

El usuario de base de datos asociado con las tablas de ejecución de prueba debe tener privilegios CREATE para añadir las tablas de resultados de ejecución de prueba.

La base de datos de ejecución de prueba debe contener todas las tablas correlacionadas en el canal interactivo.

Estas tablas deben contener datos para ejecutar los escenarios que desea probar en los diagramas de flujo interactivos. Por ejemplo, si los diagramas de flujo interactivos tienen lógica para clasificar a las personas en segmentos basándose en la opción seleccionada en un sistema de correo de voz, debe tener al menos una fila para cada selección posible. Si está creando una interacción que funciona con un formulario en el sitio web, debe incluir filas que representen los datos que faltan o están dañados, por ejemplo, utilice `name@domaincom` para el valor de una dirección de correo electrónico.

Cada tabla de ejecución de prueba debe contener como mínimo una lista de ID para el nivel de audiencia adecuado, y una columna que represente los datos en tiempo real que espera utilizar. Como las ejecuciones de prueba no tienen acceso a los datos en tiempo real, debe proporcionar datos de muestra para cada dato en tiempo real esperado. Por ejemplo, si desea utilizar datos recopilados en tiempo real como el nombre de la última página web visitada, que se almacena en el atributo `lastPageVisited`, o el número de artículos de un carro de la compra, que se almacena en el atributo `shoppingCartItemCount`, debe crear columnas con los mismos nombres y completarlas con datos de muestra. Esto permite probar la ejecución de las ramas de la lógica del diagrama de flujo que tienen una naturaleza de comportamiento o contextual.

Las ejecuciones de prueba de los diagramas de flujo interactivos no están optimizadas para trabajar con grandes conjuntos de datos. Puede limitar el número de filas que se utilizan para la ejecución de prueba en el proceso Interacción. No obstante, siempre se seleccionará el primero conjunto de filas. Para garantizar que se seleccionen otros conjuntos de filas, utilice distintas vistas de las tablas de ejecución de prueba.

Para probar el rendimiento de los diagramas de flujo interactivos en el tiempo de ejecución, debe crear un entorno de ejecución de prueba, incluida una tabla de perfil para el entorno de prueba.

En la práctica, necesitará tres conjuntos de tablas para la prueba: una tabla de ejecución de prueba para las ejecuciones de prueba de los diagramas de flujo interactivos, tablas de perfil de prueba para el grupo de servidores de prueba, y un conjunto de tablas de perfil de producción.

Sustitución de los tipos de datos predeterminados para tablas creadas dinámicamente

El entorno de ejecución de Interact crea dinámicamente tablas en dos escenarios: durante una ejecución de prueba de un diagrama de flujo y durante la ejecución de un proceso Instantánea que graba en una tabla que no existe ya. Para crear estas tablas, Interact se basa en tipos de datos codificados para cada tipo de base de datos soportado.

Puede sustituir los tipos de datos predeterminados creando una tabla de tipos de datos alternativos, denominada `uaci_column_types`, en `testRunDataSource` o `prodUserDataSource.` Esta tabla adicional permite a Interact contener casos no habituales que no se incluyen en los tipos de datos codificados.

Cuando la tabla `uaci_column_types` está definida, Interact utiliza los metadatos para las columnas como los tipos de datos que se utilizarán para cualquier generación de tabla. Si no se define la tabla `uaci_column_types`, o si se han encontrado excepciones al intentar leer la tabla, se utilizarán los tipos de datos predeterminados.

Durante el inicio, en primer lugar el sistema de ejecución comprueba el testRunDataSource para la tabla uaci_column_types. Si la tabla uaci_column_types no existe en testDataSource, o si el prodUserDataSource es de un tipo de base de datos distinto, a continuación Interact comprueba el prodUserDataSource para la tabla.

Sustituir los tipos de datos predeterminados

Siga estos pasos para sustituir los tipos de datos predeterminados para tablas creadas dinámicamente.

1. Cree una tabla en TestRunDataSource o ProdUserDataSource con las propiedades siguientes:

   Nombre de tabla: uaci_column_types

   Nombres de columna:

   - uaci_float
   - uaci_number
   - uaci_datetime
   - uaci_string

   Defina cada columna utilizando el tipo de datos adecuado al que da soporte la base de datos.
2. Reinicie el servidor de ejecución para permitir que Interact reconozca la nueva tabla.

   Importante: El servidor de ejecución se reiniciará cada vez que se realicen cambios en la tabla uaci_column_types.

Tipos de datos predeterminados para tablas creadas dinámicamente

La tabla siguiente lista los tipos de datos codificados que utiliza de forma predeterminada el sistema de ejecución de Interact para cada base de datos soportada para columnas de valores flotantes, numéricos, de fecha/hora y de serie.

Tabla 1. Tipos de datos predeterminados para tablas creadas dinámicamente

Base de datos	Tipos de datos predeterminados
DB2	• float • bigint • timestamp • varchar
Informix	• float • int8 • DATETIME YEAR TO FRACTION • char2
Oracle	• float • number(19) • timestamp • varchar2

Tabla 1. Tipos de datos predeterminados para tablas creadas dinámicamente (continuación)

Base de datos	Tipos de datos predeterminados
SQL Server	• `float` • `bigint` • `datetime` • `nvarchar`

Base de datos de perfil

El contenido de la base de datos de perfil depende totalmente de los datos que necesita para configurar los diagramas de flujo interactivos y la API de Interact. Interact requiere o recomienda que cada base de datos contenga determinadas tablas o datos.

La base de datos de perfil debe contener lo siguiente:

- Todas las tablas correlacionadas en el canal interactivo.

  Estas tablas deben contener todos los datos necesarios para ejecutar los diagramas de flujo interactivos en producción. Estas tablas deben acoplarse, dinamizarse e indexarse correctamente. Como existe un coste de rendimiento al acceder a los datos dimensionales, debe utilizar un esquema desnormalizado siempre que sea posible. Como mínimo, debe indexar la tabla de perfil en los campos ID de nivel de audiencia. Si se recuperan otros campos de las tablas dimensionales, deben indexarse correctamente para reducir el tiempo de recuperación de la base de datos. Los ID de audiencia de las tablas de perfil deben coincidir con los ID de audiencia definidos en Campaign.
- Si establece la propiedad de configuración `enableScoreOverrideLookup` en true, debe incluir una tabla de alteración temporal de puntuaciones para al menos un nivel de audiencia. Los nombres de tabla de alteración temporal de puntuaciones se definen con la propiedad `scoreOverrideTable`.

  La tabla de alteración temporal de puntuaciones puede contener pares individuales de cliente y oferta. Puede crear una tabla de alteración temporal de puntuaciones de muestra, `UACI_ScoreOverride`, ejecutando el script SQL `aci_usertab` en la base de datos de perfil. También debe indexar esta tabla en la columna ID de audiencia.

  Si establece la propiedad `enableScoreOverrideLookup` en false, no necesita incluir una tabla de alteración temporal de puntuaciones.
- Si establece la propiedad de configuración `enableDefaultOfferLookup` en true, debe incluir la tabla de ofertas globales (`UACI_DefaultOffers`). Puede crear la tabla de ofertas globales ejecutando el script SQL `aci_usertab` en la base de datos de perfil.

  La tabla de ofertas globales puede contener pares de audiencia y oferta.
- Si establece la propiedad `enableOfferSuppressionLookup` en true, debe incluir una tabla de supresión de ofertas para al menos un nivel de audiencia. Los nombres de tabla de supresión de ofertas se definen con la propiedad `offerSuppressionTable`.

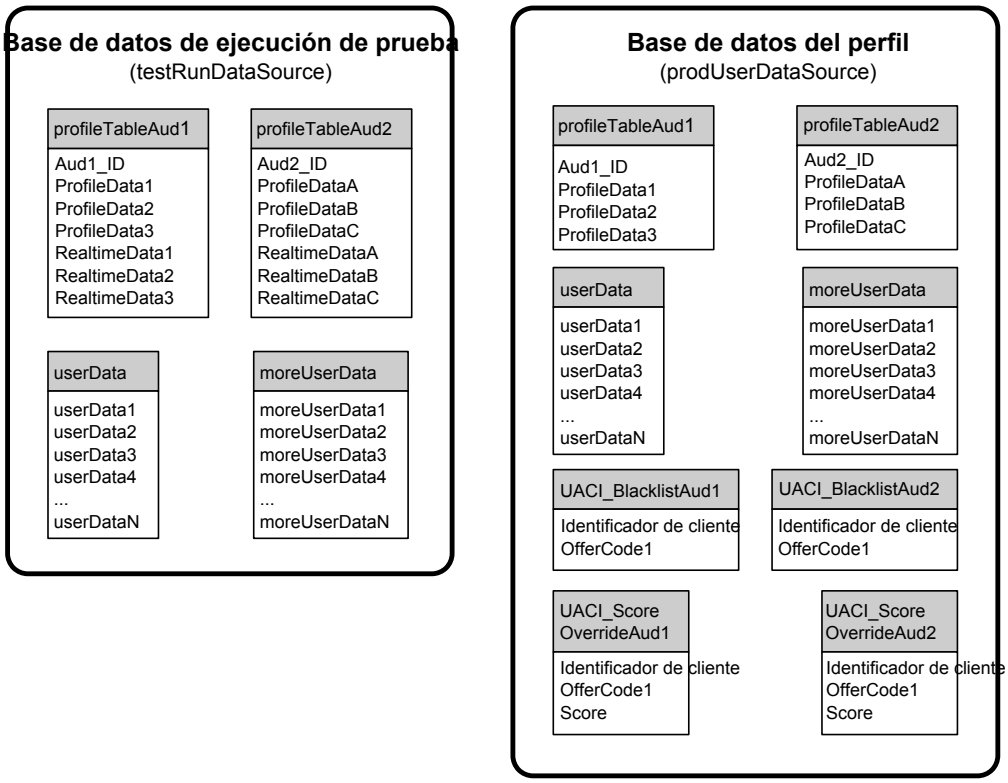
  La tabla de supresión de ofertas puede contener una fila para cada oferta suprimida para un miembro de la audiencia, aunque no se necesita una entrada para todos los miembros de audiencia. Puede crear una tabla de supresión de ofertas de muestra, `UACI_BlackList`, ejecutando el script SQL `aci_usertab` en la base de datos de perfil.

Si establece la propiedad `enableOfferSuppressionLookup` en false, no necesita incluir una tabla de supresión de ofertas.

Una gran cantidad de datos en cualquiera de estas tablas puede disminuir el rendimiento. Para obtener los mejores resultados, coloque los índices adecuados en las columnas de nivel de audiencia de las tablas utilizadas en el tiempo de ejecución que tengan grandes cantidades de datos.

Todas las propiedades de configuración referencias anteriormente están en la categoría **Interact > perfil** o **Interact > perfil > Niveles de audiencia > *AudienceLevel***. El script SQL `aci_usertab` se encuentra en el directorio `ddl` del directorio de instalación del entorno de ejecución.

El siguiente diagrama incluye las tablas de muestra de las bases de datos de perfil y ejecución de prueba para los niveles de audiencia Aud1 y Aud2.



Tablas de aprendizaje

Si utiliza el aprendizaje incorporado de Interact, debe configurar las tablas de aprendizaje. Estas tablas contienen todos los datos de los que aprende la característica de aprendizaje incorporado.

Si utiliza atributos de aprendizaje dinámicos, debe completar la tabla `UACI_AttributeList`.

El aprendizaje implica escribir en las tablas de preparación intermedias y agregar información de las tablas de preparación en las tablas de aprendizaje. Las propiedades de configuración `insertRawStatsIntervalInMinutes` y

aggregateStatsIntervalInMinutes en la categoría Interact > offerserving > Configuración de aprendizaje incorporado determinan con qué frecuencia se completan las tablas de aprendizaje.

El atributo insertRawStatsIntervalInMinutes determina con qué frecuencia se mueve la información de contacto y aceptación de cada cliente desde la memoria a las tablas de preparación, UACI_OfferStatsTX y UACI_OfferAllTx. La información almacenada en las tablas de preparación se agrega y traslada a las tablas UACI_OfferStats y UACI_OfferStatsAll a intervalos regulares determinados por la propiedad de configuración aggregateStatsIntervalInMinutes.

El aprendizaje incorporado de Interact utiliza estos datos para calcular las puntuaciones finales de las ofertas.

Historial de contactos para el seguimiento de respuestas de sesiones cruzadas

Si habilita la característica de respuestas de sesiones cruzadas, el entorno de ejecución necesita acceso de sólo lectura a las tablas del historial de contactos de Campaign. Puede configurar el entorno de ejecución para ver las tablas del sistema de Campaign, o puede crear una copia de las tablas del historial de contactos de Campaign. Si crea una copia de las tablas, debe gestionar el proceso de mantener la copia actualizada. El módulo del historial de contactos y respuestas no actualizará la copia de las tablas del historial de contactos.

Debe ejecutar el script SQL aci_crhtab en estas tablas de historial de contactos para añadir las tablas necesarias para la característica de seguimiento de respuestas de sesiones cruzadas.

Utilización de los scripts de características de Interact

Varias de las características opcionales disponibles con Interact requieren cambios en tablas específicas en las bases de datos de perfil. La instalación de Interact, para el entorno de diseño y el entorno de ejecución, incluye scripts ddl de características. Estos scripts añaden las columnas específicas que necesita la tabla.

Para habilitar cualquiera de estas características, ejecute el script correspondiente en la tabla o base de datos adecuada.

dbType es el tipo de base de datos, por ejemplo, sqlsvr para Microsoft SQL Server.

Nombre de característica	**Script de características**	**Ejecutar en**	**Cambio**
Ofertas globales, supresión de oferta y alteración temporal de puntuaciones	Directorio de instalación de entorno de ejecución\ddl\ aci_usrtab_*dbType*.sql	La base de datos de perfil (userProdDataSource)	Crea las tablas DefaultOffers, UACI_BlackList y UACI_ScoreOverride
Puntuación	Directorio de instalación de entorno de ejecución\ddl\ acifeatures\ aci_scoringfeature_*dbType*.sql	Tablas de alteración temporal de puntuaciones en la base de datos de perfil (userProdDataSource)	Añade las columnas LikelihoodScore y AdjExploreScore.

Nombre de característica	Script de características	Ejecutar en	Cambio
Aprendizaje	Directorio de instalación de entorno de ejecución\ddl\ acifeatures\ aci_lrnfeature_*dbType*.sql	Base de datos de Campaign que contiene las tablas del historial de contactos	Añada la columna RTSelectionMethod a la tabla UA_DtlContactHist.

Acerca del seguimiento del historial de contactos y respuestas

Puede configurar el entorno de ejecución para registrar el historial de contactos y respuestas en las tablas de historial de contactos y respuestas de Campaign. Los servidores de ejecución almacenan el historial de contactos y respuestas en las tablas de preparación. El módulo de historial de respuestas y contactos copia estos datos desde las tablas de preparación en las tablas de historial de contactos y respuestas de Campaign.

El módulo del historial de contactos y respuestas sólo funciona si establece las propiedades interactInstalled y contactAndResponseHistTracking > isEnabled en la página de configuración del entorno de diseño en yes.

Si utiliza el módulo de seguimiento de respuestas de sesiones cruzadas, el módulo del historial de contactos y respuestas es una entidad aparte.

Configuración de los tipos de contactos y respuestas

Puede registrar un tipo de contacto y dos tipos de respuesta en Interact tal como se muestra en la tabla siguiente. Todas estas propiedades se encuentran en la categoría contactAndResponseHistTracking.

Evento	Tipo de contacto/respuesta	Propiedad de configuración
Registrar contacto de oferta	Contacto	contactado
Registrar aceptación de oferta	Respuesta	aceptar
Registrar rechazo de oferta	Respuesta	rechazar

Puede registrar tipos de respuesta personalizados adicionales utilizando el método postEvent.

Debe asegurarse también de que la columna CountsAsResponse de la tabla UA_UsrResponseType de las tablas del sistema de Campaign esté configurada correctamente. Todos estos tipos de respuesta deben existir en la tabla UA_UsrResponseType.

Para que sea una entrada válida en UA_UsrResponseType, debe definir un valor para todas las columnas de la tabla, incluido CountsAsResponse. Los valores válidos de CountsAsResponse son 0, 1 ó 2. 0 indica que no ha habido ninguna respuesta, 1 indica una respuesta y 2 indica rechazo. Estas respuestas se utilizan para la creación de informes.

Tipos de respuesta adicionales

En Interact, puede utilizar el método postEvent en la API de Interact para desencadenar un evento que registra una acción "aceptar" o "rechazar" para una oferta. También puede aumentar el sistema para que la llamada postEvent pueda

registrar tipos de respuesta adicionales como, por ejemplo, Explorar, Considerar, Confirmar o Cumplir. Todos estos tipos de respuesta deben existir en la tabla UA_UsrResponseType en las tablas del sistema de Campaign. Utilizando parámetros de evento específicos en el método postEvent, puede registrar tipos de respuesta adicionales y definir si debe incluirse una aceptación en el aprendizaje.

Para registrar tipos de respuesta adicionales, debe añadir los siguientes parámetros de evento:

- **UACIRESPONSETYPECODE** — una cadena que representa un código de tipo de respuesta. El valor debe ser una entrada válida en la tabla UA_UsrResponseType.

  Para que sea una entrada válida en UA_UsrResponseType, debe definir todas las columnas de la tabla, incluida CountsAsResponse. Los valores válidos para CountsAsResponse son 0, 1, o 2. 0 indica que no hay respuesta, 1 indica una respuesta y 2 indica un rechazo. Estas respuestas se utilizan para la creación de informes.
- **UACILOGTOLEARNING** — Un número con el valor 1 ó 0. El valor 1 indica que Interact debe registrar el evento como una aceptación de aprendizaje. El valor 0 indica que Interact no debe registrar el evento de aprendizaje. Este parámetro permite crear varios métodos postEvent que registran distintos tipos de respuesta sin influir en el aprendizaje. Si no define UACILOGTOLEARNING, Interact utiliza el valor predeterminado 0.

Si lo desea, puede crear varios eventos con la acción Registrar aceptación de oferta, uno para cada tipo de respuesta que desee registrar, o un solo evento con la acción Registrar aceptación de oferta que utiliza para cada llamada postEvent utilizada para registrar tipos de respuesta diferentes.

Por ejemplo, cree un evento con la acción Registrar aceptación de oferta para cada tipo de respuesta. Defina las siguientes propiedades personalizadas en la tabla UA_UsrResponseType [como Nombre (código)]: Explorar (EXP), Considerar (CON) y Confirmar (CMT). A continuación, cree tres eventos y denomínelos LogAccept_Explore, LogAccept_Consider y LogAccept_Commit. Los tres eventos son exactamente el mismo (tienen la acción Registrar aceptación de oferta), pero los nombres son diferentes, para que la persona que trabaja con la API pueda distinguirlos.

O bien, puede crear un solo evento con la acción Registrar aceptación de oferta que utiliza para todos los tipos de respuesta personalizados. Por ejemplo, denomínelo LogCustomResponse.

Cuando trabaja con la API, no hay ninguna diferencia funcional entre los eventos, pero los convenios de denominación pueden clarificar el código. Asimismo, si asigna a cada respuesta personalizada un nombre diferente, el informe Resumen de actividad de eventos de canal muestra información más precisa.

En primer lugar, configure todos los pares nombre-valor

```
//Definir pares de nombre y valor para UACIRESPONSETYPECODE
// Tipo de respuesta Explore
NameValuePair responseTypeEXP = new NameValuePairImpl();
responseTypeEXP.setName("UACIRESPONSETYPECODE");
responseTypeEXP.setValueAsString("EXP");
responseTypeEXP.setValueDataType(NameValuePair.DATA_TYPE_STRING);

// Tipo de respuesta Consider
NameValuePair responseTypeCON = new NameValuePairImpl();
responseTypeCON.setName("UACIRESPONSETYPECODE");
```

```
responseTypeCON.setValueAsString("CON");
responseTypeCON.setValueDataType(NameValuePair.DATA_TYPE_STRING);

// Tipo de respuesta Commit
NameValuePair responseTypeCMT = new NameValuePairImpl();
responseTypeCMT.setName("UACIRESPONSETYPECODE");
responseTypeCMT.setValueAsString("CMT");
responseTypeCMT.setValueDataType(NameValuePair.DATA_TYPE_STRING);

//Definir pares de nombre y valor para UACILOGTOLEARNING
//No se registra en el aprendizaje
NameValuePair noLogToLearning = new NameValuePairImpl();
noLogToLearning.setName("UACILOGTOLEARNING");
noLogToLearning.setValueAsString("0");
noLogToLearning.setValueDataType(NameValuePair.DATA_TYPE_NUMERIC);

//Se registra en el aprendizaje
NameValuePair LogToLearning = new NameValuePairImpl();
LogToLearning.setName("UACILOGTOLEARNING");
LogToLearning.setValueAsString("1");
LogToLearning.setValueDataType(NameValuePair.DATA_TYPE_NUMERIC);
```

Este primer ejemplo muestra cómo utilizar los eventos individuales.

```
//EJEMPLO 1: Este conjunto de llamadas postEvent utilizan los eventos denominados individualmente
//PostEvent con una respuesta Explore
NameValuePair[] postEventParameters = { responseTypeEXP, noLogToLearning };
response = api.postEvent(sessionId, LogAccept_Explore, postEventParameters);

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

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

Este segundo ejemplo muestra cómo utilizar un solo evento.

```
//EJEMPLO 2: Este conjunto de llamadas postEvent utilizan un solo evento
//PostEvent con una respuesta Explore
NameValuePair[] postEventParameters = { responseTypeEXP, noLogToLearning };
response = api.postEvent(sessionId, LogCustomResponse, postEventParameters);

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

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

Ambos ejemplos realizan exactamente las mismas acciones, sin embargo, una versión puede ser más fácil de leer que la otra.

Correlación de tablas de preparación del entorno de ejecución con tablas de historial de Campaign

Las tablas siguientes muestran cómo se correlacionan las tablas de preparación del entorno de ejecución con las tablas de historial de Campaign. Recuerde que debe tener una de estas tablas en cada nivel de audiencia. Los nombres de tabla que se muestran son las tablas de muestra creadas para la audiencia predeterminada en las tablas de ejecución y las tablas del sistema de Campaign.

Tabla 2. Historial de contactos

UACI_CHStaging **Nombre de columna de la tabla de preparación de historial de contactos de Interact**	**Tabla de historial de contactos de Campaign**	**Nombre de columna de la tabla**
ContactID	N/D	N/D
TreatmentCode	UA_Treatment	TreatmentCode
CampaignID	UA_Treatment	CampaignID
OfferID	UA_Treatment	OfferID
CellID	UA_Treatment	CellID
CustomerID	UA_DtlContactHist	CustomerID
ContactDate	UA_DtlContactHist	ContactDateTime
ExpirationDateTime	UA_Treatment	ExpirationDateTime
EffectiveDateTime	UA_Treatment	EffectiveDateTime
ContactType	UA_DtlContactHist	ContactStatusID
UserDefinedFields	UA_DtlContactHist	UserDefinedFields

`ContactID` es una clave para unir `UACI_CHOfferAtrib` a `UACI_CHStaging`.

Tabla 3. Atributos de oferta

UACI_CHOfferAttrib **Nombre de columna de la tabla de preparación de historial de contactos de Interact**	**Tabla de historial de contactos de Campaign**	**Nombre de columna de la tabla**
ContactID	N/D	N/D
AttributeID	UA_OfferHistAttrib	AttributeID
StringValue	UA_OfferHistAttrib	StringValue
NumberValue	UA_OfferHistAttrib	NumberValue
DateTimeValue	UA_OfferHistAttrib	DateTimeValue

`ContactID` es una clave para unir `UACI_CHOfferAtrib` a `UACI_CHStaging`.

Tabla 4. Historial de respuestas

UACI_RHStaging **Nombre de columna de la tabla de preparación del historial de respuestas de Interact**	**Tabla del historial de respuestas de Campaign**	**Nombre de columna de la tabla**
SeqNum	N/D	N/D
TreatmentCode	UA_ResponseHistory	TreatmentInstID
CustomerID	UA_ResponseHistory	CustomerID
ResponseDate	UA_ResponseHistory	ResponseDateTime
ResponseType	UA_ResponseHistory	ResponseTypeID
UserDefinedFields	UA_ResponseHistory	UserDefinedFields

SeqNum es una clave que utiliza el módulo de historial de contactos y respuestas para identificar datos, pero no se registra en las tablas de respuesta de Campaign.

La columna userDefinedFields puede contener los datos que elija. Si añade columnas a las tablas de preparación, el módulo de historial de contactos y respuestas los graba en las tablas UA_DtlContactHist o UA_ResponseHistory en columnas de la misma tabla. Por ejemplo, si añade la columna linkFrom a la tabla UACI_CHStaging, el módulo de historial de contactos y respuestas copiará esos datos en la columna linkFrom de la tabla UA_DtlContactHist.

Importante: Si tiene columnas adicionales en las tablas de historial de contactos y respuestas de Campaign, debe añadir columnas coincidentes a las tablas de preparación antes de ejecutar el módulo de historial de contactos y respuestas.

Puede completar columnas adicionales de las tablas de preparación creando columnas con el mismo nombre que los pares nombre-valor de los datos de sesión de ejecución. Por ejemplo, si crea pares nombre-valor NumberItemsInWishList y NumberItemsInShoppingCart, cuando se produzca un evento Registrar aceptación de oferta o Registrar rechazo de oferta, si existen las columnas NumberItemsInWishList y NumberItemsInShoppingCart en la tabla UACI_RHStaging el entorno de ejecución completará estos campos. El entorno de ejecución completa la tabla UACI_CHStaging cuando se produce un evento Registrar contacto de oferta.

Puede utilizar estos campos definidos por el usuario para incluir la puntuación utilizada para presentar una oferta. Añada una columna denominada PuntuaciónFinal a la tabla UACI_CHStaging de las tablas de ejecución y a la tabla UA_DtlContactHist de las tablas del sistema de Campaign. Interact completa automáticamente la columna PuntuaciónFinal con la puntuación final utilizada para la oferta si está utilizando aprendizaje incorporado.

Si está creando un módulo de aprendizaje personalizado, puede utilizar el método setActualValueUsed de la interfaz de ITreatment y el método logEvent de la interfaz de ILearning.

Si no está utilizando aprendizaje, añada una columna denominada Puntuación a la tabla UACI_CHStaging de las tablas de ejecución y a la tabla UA_DtlContactHist de las tablas del sistema de Campaign. Interact completa automáticamente la columna Puntuación con la puntuación utilizada para la oferta.

Configuración de la supervisión JMX para el módulo de historial de contactos y respuestas

En la Marketing Platform del entorno de diseño, edite las siguientes propiedades de configuración en la categoría Campaign > monitoring.

Propiedad de configuración	Valor
monitorEnabledForInteract	**True**
port	El número de puerto del servicio JMX
protocol	**JMXMP** o **RMI** La seguridad no está habilitada para el módulo de historial de contactos y respuestas, aunque seleccione el protocolo JMXMP.

Cuando visualiza los datos del historial de contactos y respuestas en la herramienta de supervisión JMX, los atributos se organizan primero por partición y, a continuación, por nivel de audiencia.

La dirección predeterminada de supervisión para el módulo de historial de contactos y respuestas con el protocolo JMXMP es `service:jmx:jmxmp://`*`CampaignServer`*`:`*`port`*`/campaign`.

La dirección predeterminada de supervisión para el módulo de historial de contactos y respuestas con el protocolo RMI es `service:jmx:rmi:///jndi/rmi://`*`CampaignServer`*`:`*`port`*`/campaign`.

Acerca del seguimiento de respuestas de sesiones cruzadas

Los visitantes no siempre completan una transacción en una única visita al punto de encuentro. Un cliente puede añadir un elemento a su carro de compra en el sitio web y no completar la compra hasta dos días después. No es factible mantener la sesión de ejecución activa de forma indefinida. Puede habilitar el seguimiento de respuestas de sesiones cruzadas para realizar seguimiento de una presentación de oferta en una sesión y correlacionarla con una respuesta en otra sesión.

De forma predeterminada, el seguimiento de respuestas de sesiones cruzadas de Interact puede realizar la correlación según los códigos de tratamiento o los códigos de oferta. También puede configurarlo para correlacionar cualquier código personalizado que elija. La respuesta de sesiones cruzadas se correlaciona según los datos disponibles. Por ejemplo, el sitio web incluye una oferta con un código promocional generado en el momento de la visualización de un descuento válido durante una semana. Un usuario puede añadir artículos a su carro de compra, pero no completar la compra hasta tres días después. Cuando se utiliza la llamada a `postEvent` para registrar un evento de aceptación, puede incluir solo el código promocional. Dado que el tiempo de ejecución no puede encontrar un código de tratamiento u oferta que correlacionar en la sesión actual, el tiempo de ejecución coloca el evento de aceptación con la información disponible en una tabla de preparación de respuesta de sesiones cruzadas (XSessResponse). El servicio CrossSessionResponse lee periódicamente la tabla XSessResponse e intenta correlacionar los registros con los datos de historial de contactos disponibles. El servicio CrossSessionResponse correlaciona el código promocional con el historial de contactos y recopila todos los datos necesarios para registrar la respuesta adecuada. A continuación, el servicio CrossSessionResponse graba la respuesta en las tablas de preparación de respuestas y, si el aprendizaje está habilitado, las tablas de aprendizaje. A continuación, el módulo del historial de contactos y respuestas graba la respuesta en las tablas del historial de contactos y respuestas de Campaign.

Configuración del origen de datos de seguimiento de respuestas de sesiones cruzadas

El seguimiento de respuestas de sesiones cruzadas de Interact correlaciona los datos de sesión del entorno de ejecución con el historial de contactos y respuestas de Campaign. De forma predeterminada, el seguimiento de respuestas de sesiones cruzadas realiza la correlación según el código de tratamiento o el código de oferta. Puede configurar el entorno de ejecución para que realice la correlación según un código alternativo personalizado.

- Si elige correlacionar según un código alternativo, debe definir el código alternativo en la tabla `UACI_TrackingType` de las tablas de ejecución de Interact.

- El entorno de ejecución debe tener acceso a las tablas de historial de contactos de Campaign. Puede ser configurando el entorno de ejecución para tener acceso a las tablas de historial de contactos de Campaign o creando una copia de las tablas del historial de contactos en el entorno de ejecución.

  Este acceso es de solo lectura y es independiente de la utilidad de historial de contactos y respuestas.

  Si crea una copia de las tablas, es su responsabilidad asegurarse de que los datos de la copia del historial de contactos sean precisos. Puede configurar el periodo de tiempo que el servicio CrossSessionResponse mantiene las respuestas no correlacionadas para correlacionar la frecuencia con la que se renuevan los datos en la copia de las tablas del historial de contactos utilizando la propiedad `purgeOrphanResponseThresholdInMinutes`. Si está utilizando el módulo de historial de contactos y respuestas, debe coordinar las actualizaciones de ETL para asegurarse de que tienen los datos más actuales.

Configuración de las tablas de historial de contactos y respuestas para seguimiento de respuestas de sesiones cruzadas

Tanto si crea una copia de las tablas de historial de contactos como si utiliza las propias tablas en las tablas del sistema de Campaign, debe realizar los pasos siguientes.

1. Las tablas de historial de contactos y respuestas se deben correlacionar correctamente en Campaign.
2. Debe ejecutar el script SQL `aci_lrnfeature` en el directorio `interactDT/ddl/acifeatures` del directorio de instalación del entorno de diseño de Interact en las tablas `UA_DtlContactHist` y `UA_ResponseHistory` de las tablas del sistema de Campaign.

   Esto añade la columna `RTSelectionMethod` a las tablas `UA_DtlContactHist` y `UA_ResponseHistory`. Ejecute el script `aci_lrnfeature` en estas tablas para cada uno de sus niveles de audiencia. Edite el script según sea necesario con la tabla correcta para cada nivel de audiencia.
3. Si va a copiar las tablas de historial de contactos en el entorno de ejecución, hágalo ahora.
4. Ejecute el script SQL `aci_crhtab` en el directorio `ddl` del directorio de instalación del entorno de ejecución de Interact en el origen de datos de historial de contactos y respuestas.

   Este script crea las tablas `UACI_XsessResponse` y `UACI_CRHTAB_Ver`.
5. Cree una versión de la tabla `UACI_XsessResponse` para cada nivel de audiencia.

Si está creando una copia de las tablas de historial de contactos de Campaign accesible por el entorno de ejecución para el soporte de seguimiento de respuestas de sesiones cruzadas, utilice las directrices siguientes:

- El seguimiento de respuestas de sesiones cruzadas requiere acceso de solo lectura a estas tablas.
- El seguimiento de respuestas de sesiones cruzadas requiere las tablas siguientes del historial de contactos de Campaign.
  - `UA_DtlContactHist` (para cada nivel de audiencia)
  - `UA_Treatment`

  Debe actualizar regularmente los datos de estas tablas para asegurarse de que se realiza un seguimiento preciso de las respuestas.

Para mejorar el rendimiento del seguimiento de respuestas de sesiones cruzadas, es posible que desee limitar la cantidad de datos de historial de contactos, ya sea mediante la forma en la que copia los datos de historial de contactos o configurando una vista a las tablas de historial de contactos de Campaign. Por ejemplo, si tiene una práctica de empresa por la que ninguna oferta es válida más de 30 días, debe limitar los datos del historial de contactos a los últimos 30 días.

No verá resultados del seguimiento de respuestas de sesiones cruzadas hasta que se ejecute el módulo de historial de contactos y respuestas. Por ejemplo, el valor predeterminado `processSleepIntervalInMinutes` es de 60 minutos. Por lo tanto, puede pasar al menos una hora hasta que las respuestas de sesiones cruzadas aparezcan en el historial de respuestas de Campaign.

Tabla UACI_TrackingType

La tabla `UACI_TrackingType` forma parte de las tablas del entorno de ejecución. Esta tabla define los códigos de seguimiento utilizados con el seguimiento de respuestas de sesiones cruzadas. El código de seguimiento define qué método utiliza el entorno de ejecución para correlacionar la oferta actual de una sesión de ejecución con el historial de contactos y respuestas.

Columna	Tipo	Descripción
`TrackingCodeType`	int	Número que representa el tipo de código de seguimiento. Los comandos SQL utilizados para correlacionar información de los datos de sesión con las tablas de historial de contactos y respuestas hacen referencia a este número.
`Nombre`	varchar(64)	Nombre del tipo de código de seguimiento. Se pasa a los datos de sesión mediante el parámetro reservado `UACI_TrackingCodeType` con el método `postEvent`.
`Descripción`	varchar(512)	Breve descripción del tipo de código de seguimiento. Este campo es opcional.

De forma predeterminada, el entorno de ejecución tiene dos tipos de código de seguimiento definidos, tal como se muestra en la tabla siguiente. Para cualquier código alternativo, debe definir un `TrackingCodeType` exclusivo.

TrackingCodeType	Nombre	Descripción
1	Código de tratamiento	Código de tratamiento generado por UACI
2	Código de oferta	Código de oferta de campaña UAC

UACI_XSessResponse

Debe existir una instancia de esta tabla para cada nivel de audiencia en el origen de datos de historial de contactos y respuestas disponible para el seguimiento de respuestas de sesiones cruzadas de Interact.

Columna	Tipo	Descripción
`SeqNumber`	bigint	Identificador de la fila de datos. El servicio CrossSessionResponse procesa todos los registros en el orden `SeqNumber`.

Columna	Tipo	Descripción
ICID	bigint	ID de canal interactivo
IDAudiencia	bigint	ID de audiencia de este nivel de audiencia. El nombre de esta columna debe coincidir con el ID de audiencia definido en Campaign. La tabla de muestra contiene la columna CustomerID.
TrackingCode	varchar(64)	Valor pasado por el parámetro UACIOfferTrackingCode del método postEvent.
TrackingCodeType	int	Representación numérica del código de seguimiento. El valor debe ser una entrada válida de la tabla UACI_TrackingType.
OfferID	bigint	ID de oferta tal como se define en Campaign.
ResponseType	int	Tipo de respuesta para este registro. El valor debe ser una entrada válida de la tabla UA_UsrResponseType.
ResponseTypeCode	varchar(64)	Código de tipo de respuesta para este registro. El valor debe ser una entrada válida de la tabla UA_UsrResponseType.
ResponseDate	datetime	Fecha de la respuesta.
Mark	bigint	El valor de este campo identifica el estado del registro. • 1: en curso • 2: satisfactorio • NULL: reintentar • -1: el registro ha estado en la base de datos durante más de purgeOrphanResponseThresholdInMinutes minutos. Como parte del mantenimiento de esta tabla que realiza el administrador de bases de datos, puede seleccionar este campo para registros que no se correlacionan, es decir, todos los registros con valor de -1. Todos los registros con valor 2 los elimina automáticamente el servicio CrossSessionResponse.
UsrDefinedFields	char(18)	Los campos personalizados que desee incluir al correlacionar respuestas de oferta con el historial de contactos y respuestas. Por ejemplo, si desea correlacionar según un código promocional, incluya un campo definido por el usuario de código promocional.

Habilitar seguimiento de respuestas de sesiones cruzadas

Debe configurar el módulo del historial de contactos y respuestas para beneficiarse completamente del seguimiento de respuestas de sesiones cruzadas.

Para utilizar seguimiento de respuestas de sesiones cruzadas, debe configurar el entorno de ejecución para tener acceso de lectura a las tablas de historial de contactos y respuestas de Campaign. Puede leer las propias tablas de historial de contactos y respuestas de Campaign en el entorno de diseño, o una copia de las tablas en los orígenes de datos del entorno de diseño. Este es independiente de la configuración del módulo de historial de contactos y respuestas.

Si está realizando la correlación con algún elemento que no es el código de tratamiento o el código de oferta, debe añadirlo a la tabla UACI_TrackingType.

1. Cree las tablas XSessResponse en las tablas de historial de contactos y respuestas accesibles al entorno de ejecución.
2. Defina las propiedades en la categoría contactAndResponseHistoryDataSource para el entorno de ejecución.
3. Defina la propiedad crossSessionResponseTable para cada nivel de audiencia.
4. Cree una categoría OverridePerAudience para cada nivel de audiencia.

Correlación de ofertas de respuesta entre sesiones

De forma predeterminada, la correlación del seguimiento de respuestas de sesiones cruzadas se realiza según los códigos de tratamiento o los códigos de oferta. El servicio crossSessionResponse utiliza comandos SQL para correlacionar códigos de tratamiento, códigos de oferta o un código personalizado de datos de sesión con tablas de historial de contactos y respuestas de Campaign. Puede editar estos comandos SQL para correlacionar las personalizaciones que se hayan realizado en los códigos de seguimiento, códigos de oferta o códigos personalizados.

Correlación por código de tratamiento

El SQL para correlacionar por código de tratamiento debe devolver todas las columnas de la tabla XSessResponse para este nivel de audiencia además de una columna denominada CorrelaciónIDOferta. El valor de la columna CorrelaciónIDOferta debe ser el offerId que va con el código de tratamiento del registro XSessResponse.

A continuación se muestra un ejemplo del comando SQL generado predeterminado que coincide con los códigos de tratamiento. Interact genera el SQL para utilizar los nombres de tabla correctos para el nivel de audiencia. Este SQL se utiliza si la propiedad Interact > services > crossSessionResponse > OverridePerAudience > *nivelAudiencia* > TrackingCodes > byTreatmentCode > SQL está establecida en **Utilizar SQL generado por el sistema**.

```
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
```

Los valores UACI_XsessResponse, UA_DtlContactHist, CustomerID y UA_ContactHistory los definen los valores de Interact. Por ejemplo, UACI_XsessResponse lo define la propiedad de configuración Interact > profile > Audience Levels > [NombreNivelAudiencia] > crossSessionResponseTable.

Si ha personalizado las tablas de historial de contactos y respuestas, es posible que necesite revisar este SQL para trabajar con las tablas. Puede definir sustituciones de SQL en la propiedad Interact > services > crossSessionResponse > OverridePerAudience > *(NivelAudiencia)* > TrackingCodes > byTreatmentCode > OverrideSQL. Si proporciona algún SQL de sustitución, debe cambiar también la propiedad SQL a **SQL de sustitución**.

Correlación por código de oferta

El SQL para correlacionar por código de oferta debe devolver todas las columnas de la tabla XSessResponse para este nivel de audiencia además de una columna denominada `CorrelaciónCódigoTratamiento`. El valor de la columna `CorrelaciónCódigoTratamiento` es el código de tratamiento que va con el ID de oferta (y código de oferta) en el registro XSessResponse.

A continuación se muestra un ejemplo del comando SQL generado predeterminado que correlaciona los códigos de oferta. Interact genera el SQL para utilizar los nombres de tabla correctos para el nivel de audiencia. Este SQL se utiliza si la propiedad `Interact > services > crossSessionResponse > OverridePerAudience > NivelAudiencia > TrackingCodes > byOfferCode > SQL` está establecida en **Utilizar SQL generado por el sistema**.

```
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
```

Los valores UACI_XsessResponse, UA_DtlContactHist, CustomerID y UA_ContactHistory los definen los valores de Interact. Por ejemplo, UACI_XsessResponse lo define la propiedad de configuración Interact > profile > Audience Levels > [NombreNivelAudiencia] > crossSessionResponseTable.

Si ha personalizado las tablas de historial de contactos y respuestas, es posible que necesite revisar este SQL para trabajar con las tablas. Puede definir sustituciones de SQL en la propiedad Interact > services > crossSessionResponse > OverridePerAudience > *(NivelAudiencia)* > TrackingCodes > byOfferCode > OverrideSQL. Si proporciona algún SQL de sustitución, debe cambiar también la propiedad SQL a **SQL de sustitución**.

Correlación por código alternativo

Puede definir un comando SQL para correlacionar por algún código alternativo que elija. Por ejemplo, podría tener códigos promocionales o códigos de producto además de los códigos de oferta o tratamiento.

Debe definir este código alternativo en la tabla UACI_TrackingType en las tablas de entorno de ejecución de Interact.

Debe proporcionar SQL o un procedimiento almacenado en la propiedad Interact > services > crossSessionResponse > OverridePerAudience > *(nivelAudiencia)* > TrackingCodes > byAlternateCode > OverrideSQL que devuelve todas las columnas de la tabla XSessResponse para este nivel de audiencia además de las columnas CorrelaciónIDTratamiento y CorrelaciónIDOferta. Opcionalmente, puede devolver el códigoOferta en lugar de CorrelaciónIDOferta (con el formato códigoOferta1, códigoOferta2, ... códigoOferta*N* para códigos de oferta de parte N). Los valores de la columna CorrelaciónCódigoTratamiento y CorrelaciónIDOferta (o columnas de código de oferta) deben corresponderse con el CódigoSeguimiento en el registro XSessResponse.

Por ejemplo, el siguiente pseudocódigo SQL de la columna CódigoAlternativo de la tabla 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>
```

Donde <*x*> es el código de seguimiento definido en la tabla UACI_TrackingType.

Utilización de una utilidad de carga de base de datos con el entorno de ejecución

De forma predeterminada, el entorno de ejecución graba los datos de historial de contactos y respuestas de los datos de sesión en las tablas de preparación. Sin embargo, en un sistema de producción muy activo, la cantidad de memoria necesaria para guardar en caché todos los datos antes de que el tiempo de ejecución pueda grabarlos en las tablas de preparación puede ser prohibitiva. Puede configurar el tiempo de ejecución para utilizar una utilidad de carga de base de datos para mejorar el rendimiento.

Cuando habilita una utilidad de carga de base de datos, en lugar de alojar todo el historial de contactos y respuestas en memoria antes de grabar en las tablas de

preparación, el tiempo de ejecución graba los datos en un archivo de preparación. Puede definir la ubicación del directorio que contiene los archivos de preparación con la propiedad `externalLoaderStagingDirectory`. Este directorio contiene varios subdirectorios. El primer subdirectorio es el directorio de instancia de tiempo de ejecución, que contiene los directorios `contactHist` y `respHist`. Los directorios `contactHist` y `respHist` contienen subdirectorios con nombre exclusivo y el formato *nombreNivelAudiencia.ID_exclusivo.estadoActual*, que contienen los archivos de preparación.

Estado Actual	Descripción
CACHE	Contenido del directorio que actualmente se está grabando en un archivo.
READY	Contenido de un directorio listo para su proceso.
RUN	Contenido del directorio que se está grabando actualmente en la base de datos.
PROCESSED	El contenido del directorio se ha grabado en la base de datos.
ERROR	Se ha producido un error al grabar el contenido del directorio en la base de datos.
ATTN	El contenido del directorio requiere su atención. Es decir, es posible que necesite realizar pasos manuales para completar la grabación del contenido de este directorio en la base de datos.
RERUN	El contenido del directorio está listo para grabarse en la base de datos. Debe renombrar un directorio de `ATTN` o `ERROR` a `RERUN` después de haber corregido el problema.

Puede definir el directorio de la instancia de tiempo de ejecución definiendo la propiedad de JVM interact.runtime.instance.name en el script de inicio del servidor de aplicaciones. Por ejemplo, puede añadir `-Dinteract.runtime.instance.name=instance2` al script de inicio del servidor de aplicaciones web. Si no está establecida, el nombre predeterminado es `DefaultInteractRuntimeInstance`.

El directorio `samples` contiene archivos de muestra para ayudarle a escribir sus propios archivos de control de utilidad de carga de base de datos.

Habilitación de una utilidad de carga de base de datos con el entorno de ejecución

Debe definir los archivos de control o comando de la utilidad de carga de base de datos antes de configurar el entorno de ejecución para utilizarlos. Estos archivos deben existir en la misma ubicación en todos los servidores de ejecución en el mismo grupo de servidores.

Interact proporciona archivos de control y comando de muestra en el directorio `loaderService` en la instalación del servidor de ejecución de Interact.

1. Confirme que el usuario del entorno de ejecución tenga credenciales de inicio de sesión para el origen de datos de tablas de ejecución definido en la Marketing Platform.

   El nombre del origen de datos en la Marketing Platform debe ser `systemTablesDataSource`.
2. Defina las propiedades de configuración `Interact > general > systemTablesDataSource > loaderProperties`.

3. Defina la propiedad `Interact > services >externalLoaderStagingDirectory`.
4. Revise las propiedades de configuración `Interact > services > responseHist > fileCache`, si es necesario.
5. Revise las propiedades de configuración `Interact > services > contactHist > fileCache`, si es necesario.
6. Reinicie el servidor de ejecución.

Capítulo 4. Presentación de ofertas

Puede configurar Interact de varias formas para mejorar cómo selecciona las ofertas que se presentan. En las secciones siguientes se describen estas características opcionales con más detalle.

Elegibilidad de la oferta

El objetivo de Interact es presentar las ofertas elegibles. Simplemente, Interact presenta la oferta más óptima entre las elegibles, según el visitante, el canal y la situación.

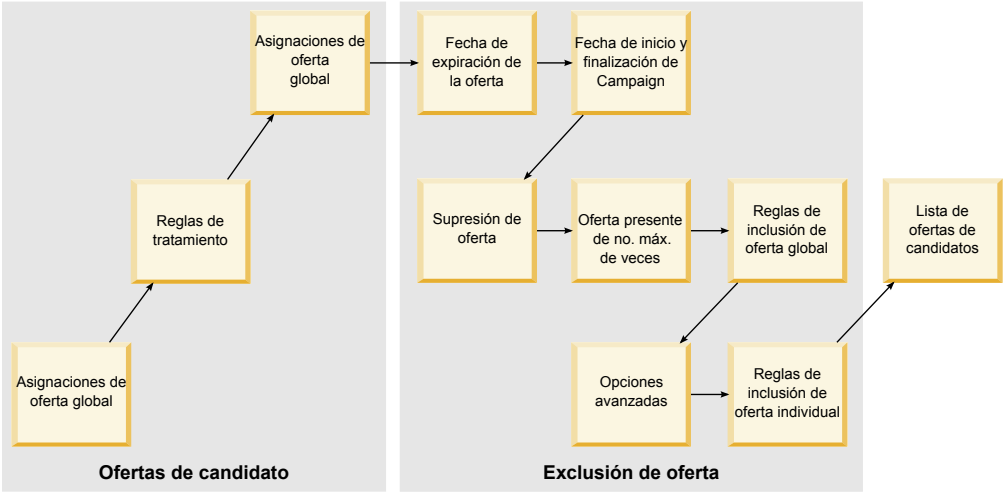
Las reglas de tratamiento son sólo el principio de cómo Interact determina qué ofertas son elegibles para un cliente. Interact tiene varias características opcionales que puede implementar para mejorar cómo determina el entorno de ejecución qué ofertas debe presentar. Ninguna de estas características garantiza que una oferta se presente a un cliente. Estas características influyen en la probabilidad de que una oferta sea elegible para su presentación a un cliente. Puede utilizar tantas o tan pocas características como necesite para implementar la mejor solución para su entorno.

Hay tres áreas principales donde puede influir en la elegibilidad de las ofertas: la generación de la lista de ofertas candidatos, la determinación de la puntuación de marketing y el aprendizaje.

Generación de una lista de ofertas candidatas

La generación de una lista de ofertas candidatas tiene dos etapas principales. La primera etapa es la generación de una lista de todas las posibles ofertas para las que se puede elegir el cliente. La segunda etapa filtra las ofertas para las que ya no se puede elegir el cliente. Hay varios lugares en las dos etapas donde puede influir en la generación de la lista de ofertas candidatas.

Este diagrama muestra las etapas de la generación de una lista de ofertas candidatas. Las flechas indican el orden de preferencia. Por ejemplo, si una oferta pasa el filtro **Número máximo de veces que se presenta una oferta**, pero no pasa el filtro **Reglas de inclusión de ofertas globales**, el entorno de ejecución excluye la oferta.

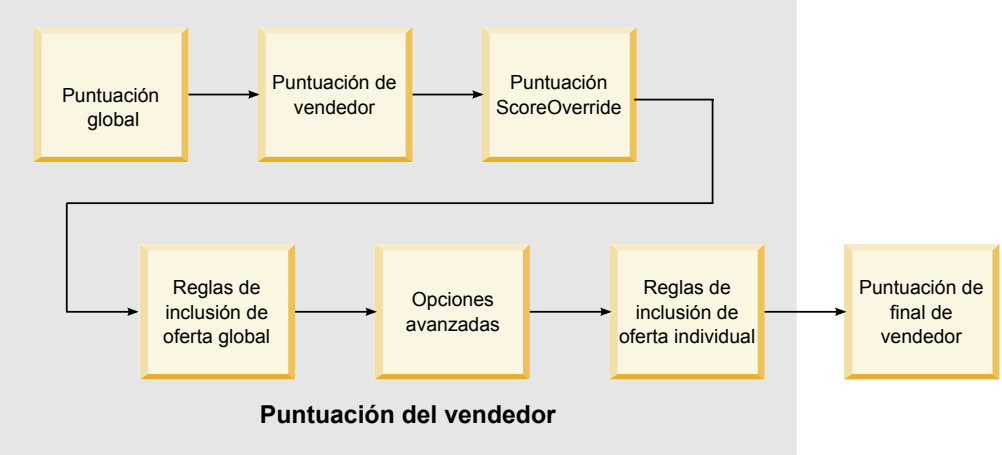


- **Asignaciones de ofertas globales** — Puede definir ofertas globales por nivel de audiencia utilizando la tabla de ofertas globales.
- **Reglas de tratamiento** — El método básico para definir ofertas por segmento y por punto de interacción utilizando la pestaña de estrategia de interacción.
- **Asignaciones de ofertas individuales** — Puede definir asignaciones de ofertas específicas por cliente utilizando la tabla de alteración temporal de puntuaciones.
- **Fecha de caducidad de la oferta** — Cuando crea una oferta en Campaign, puede definir una fecha de caducidad. Si pasa la fecha de caducidad de una oferta, el entorno de ejecución excluye la oferta.
- **Fecha de inicio y finalización de campaña** — Cuando crea una campaña en Campaign, puede definir una fecha de inicio y de finalización para la campaña. Si no llega la fecha de inicio de la campaña o si pasa la fecha de finalización, el entorno de ejecución excluye la oferta.
- **Supresión de ofertas** — Puede definir la supresión de ofertas para miembros específicos de la audiencia utilizando la tabla de supresión de ofertas.
- **Número máximo de veces que se presenta una oferta** — Cuando se define un canal interactivo, debe definir el número máximo de veces que se presenta una oferta a un cliente por sesión. Si la oferta ya se ha presentado este número de veces, el entorno de ejecución excluye la oferta.
- **Reglas de inclusión de ofertas globales** — Puede definir una expresión booleana para filtrar las ofertas en un nivel de audiencia utilizando la tabla de ofertas globales. Si el resultado es false, el entorno de ejecución excluye la oferta.
- **Opciones avanzadas** — Puede utilizar la opción avanzada **Considerar que esta regla se puede elegir si la siguiente expresión es verdadera** en una regla de tratamiento para filtrar las ofertas en un nivel de segmento. Si el resultado es false, el entorno de ejecución excluye la oferta.
- **Reglas de inclusión de ofertas individuales** — Puede definir una expresión booleana para filtrar las ofertas en un nivel de cliente utilizando la tabla de alteración temporal de puntuaciones. Si el resultado es false, el entorno de ejecución excluye la oferta.

Cálculo de la puntuación de marketing

Hay muchas formas de influir (mediante un cálculo) o alterar temporalmente la puntuación de marketing. El siguiente diagrama muestra las distintas etapas donde puede influir o alterar temporalmente la puntuación de marketing. Las flechas

indican el orden de preferencia. Por ejemplo, si define una expresión para determinar la puntuación de marketing en las opciones avanzadas de una regla de tratamiento y define una expresión en la tabla de alteración temporal de puntuaciones, la expresión en la tabla de alteración temporal de puntuaciones tiene prioridad.



- **Puntuación global** — Puede definir una puntuación por nivel de audiencia utilizando la tabla de ofertas globales.
- **Puntuación del vendedor** — Puede definir una puntuación por segmento utilizando la barra de desplazamiento en una regla de tratamiento.
- **Puntuación de alteración temporal de puntuaciones** — Puede definir una puntuación por cliente utilizando la tabla de alteración temporal de puntuaciones.
- **Reglas de inclusión de oferta global** — Puede definir una expresión que calcule una puntuación por nivel de audiencia utilizando la tabla de ofertas globales.
- **Opciones avanzadas** — Puede definir una expresión que calcule una puntuación por segmento utilizando la opción avanzada **Usar la siguiente expresión como la puntuación de marketing** en una regla de tratamiento.
- **Reglas de inclusión de oferta de alteración temporal de puntuaciones** — Puede definir una expresión que calcule una puntuación por cliente utilizando la tabla de alteración temporal de puntuaciones.

Influencia en el aprendizaje

Si está utilizando el módulo de aprendizaje incorporado de Interact, puede influir en la salida de aprendizaje más allá de las configuraciones de aprendizaje estándar como, por ejemplo, la lista de atributos de aprendizaje o el nivel de confianza. Puede alterar temporalmente los componentes del algoritmo de aprendizaje mientras utiliza el resto de componentes.

Puede alterar temporalmente de aprendizaje utilizando las columnas `LikelihoodScore` y `AdjExploreScore` de las tablas de ofertas predeterminadas y de alteración temporal de puntuaciones. Puede añadir estas columnas a las tablas de ofertas predeterminadas y de alteración temporal de puntuaciones utilizando el script de característica `aci_scoringfeature`. Para utilizar correctamente estas alteraciones temporales, necesita un conocimiento avanzado del aprendizaje incorporado de Interact.

El módulo de aprendizaje utiliza en los cálculos finales la lista de ofertas candidatas y la puntuación de marketing por oferta candidata. La lista de ofertas se utiliza con los atributos de aprendizaje para calcular la probabilidad

(probabilidad de aceptación) de que el cliente acepte la oferta. Utilizando estas probabilidades y el número histórico de presentaciones para conseguir un equilibrio entre la exploración y la explotación, el algoritmo de aprendizaje determina la ponderación de la oferta. Por último, el aprendizaje incorporado multiplica la ponderación de la oferta por la puntuación de marketing final y devuelve una puntuación final. Las ofertas se clasifican por esta puntuación final.

Acerca de la supresión de ofertas

Existen varias formas de que el entorno de ejecución suprima una oferta:

- El elemento **Núm. máximo de veces para mostrar ofertas durante una única visita** de un canal interactivo.

  El **Núm. máximo de veces para mostrar ofertas durante una única visita** se define cuando crea o edita un canal interactivo.
- El uso de una tabla de supresión de ofertas.

  La tabla de supresión de ofertas se crea en la base de datos de perfil.
- Ofertas cuya fecha de caducidad ha pasado.
- Ofertas de campañas caducadas.
- Ofertas excluidas porque no pasan una regla de inclusión de oferta (opción avanzada de regla de tratamiento).
- Ofertas aceptadas o rechazadas explícitamente en una sesión de Interact. Si un cliente acepta o rechaza explícitamente una oferta, la oferta se suprime el tiempo que dura la sesión.

Habilitación de la tabla de supresión de ofertas

Puede configurar Interact para que haga referencia a una lista de ofertas suprimidas.

1. Cree una `offerSuppressionTable`, una nueva tabla para cada audiencia que contiene el ID de audiencia y el ID de oferta.
2. Establezca la propiedad `enableOfferSuppressionLookup` en **true**.
3. Establezca la propiedad `offerSuppressionTable` en el nombre de la tabla de supresión de ofertas para la audiencia correspondiente.

Tabla de supresión de ofertas

La tabla de supresión de ofertas permite suprimir una oferta para un ID de audiencia específico. Por ejemplo, si su audiencia es Customer, puede suprimir una oferta para el cliente John Smith. Debe existir una versión de esta tabla para al menos un nivel de audiencia en la base de datos de perfil. Puede crear una tabla de supresión de ofertas de muestra, `UACI_Blacklist`, ejecutando el script SQL `aci_usertab` en la base de datos de perfil. El script SQL `aci_usertab` se encuentra en el directorio `ddl` del directorio de instalación del entorno de ejecución.

Debe definir los campos `AudienceID` y `OfferCode1` para cada fila. Puede añadir columnas adicionales si el ID de oferta o el código de oferta consta de varias columnas. Estas columnas deben coincidir con los nombres de columna definidos en Campaign. Por ejemplo, si define la audiencia `Customer` por los campos `HHold_ID` y `MemberNum`, debe añadir `HHold_ID` y `MemberNum` a la tabla de supresión de ofertas.

Nombre	Descripción
AudienceID	(Necesario) El nombre de esta columna debe coincidir con el nombre de la columna que define el ID de audiencia en Campaign. Si el ID de audiencia está formado por varias columnas, puede añadirlas a esta tabla. Cada fila debe contener el ID de audiencia al que asigna la oferta predeterminada, por ejemplo, `customer1`.
OfferCode1	(Necesario) El código de oferta para la oferta que está alterando temporalmente. Si los códigos de oferta están formados por varios campos, puede añadir las columnas adicionales, por ejemplo, `OfferCode2`, etc.

Ofertas globales y asignaciones individuales

Puede configurar el entorno de ejecución para asignar ofertas específicas más allá de las reglas de tratamiento configuradas en la pestaña de estrategia de interacción. Puede definir ofertas globales para cualquier miembro de un nivel de audiencia y asignaciones individuales para miembros de audiencia específicos. Por ejemplo, puede definir una oferta global para que la vean todas las unidades familiares cuando no haya otras disponibles y, a continuación, crear una asignación de oferta individual para la unidad familiar específica Smith.

Puede restringir las ofertas globales y las asignaciones individuales por zona, celda y reglas de inclusión de ofertas. Las ofertas globales y las asignaciones individuales se configuran mediante la adición de datos a tablas específicas en la base de datos de perfil de producción.

Para que las ofertas globales y las asignaciones individuales funcionen correctamente, deben existir todos los códigos de oferta y celda referenciados en el despliegue. Para garantizar que los datos necesarios estén disponibles, debe configurar códigos de celda predeterminados y la tabla `UACI_ICBatchOffers`.

Definición de los códigos de celda predeterminados

Si utiliza las tablas de ofertas predeterminadas o de alteración temporal de puntuaciones para las asignaciones de ofertas globales o individuales, debe definir los códigos de celda predeterminados mediante la definición de la propiedad `DefaultCellCode` para cada nivel de audiencia y tipo de tabla en la categoría `IndividualTreatment`.

El `DefaultCellCode` se utiliza si no ha definido un código de celda en una determinada fila en las tablas de ofertas predeterminadas o de alteración temporal de puntuaciones. Los informes utilizan este código de celda predeterminado.

El `DefaultCellCode` debe coincidir con el formato de código de celda definido en Campaign. Este código de celda se utiliza para todas las asignaciones de oferta que aparecen en los informes. Si define códigos de celda predeterminados exclusivos, puede identificar fácilmente las ofertas asignadas por las tablas de ofertas predeterminadas o de alteración temporal de puntuaciones.

Definición de la tabla UACI_ICBatchOffers

Si utiliza las tablas de ofertas predeterminadas o de alteración temporal de puntuaciones, asegúrese de que todos los códigos de oferta existan en el despliegue. Si sabe que todas las ofertas que se utilizan en las tablas de ofertas predeterminadas o de alteración temporal de puntuaciones se utilizan en las reglas

de tratamiento, las ofertas existen en el despliegue. No obstante, una oferta que no se utilice en una regla de tratamiento debe definirse en la tabla UACI_ICBatchOffers.

La tabla UACI_ICBatchOffers existe en las tablas del sistema de Campaign.

Debe completar la tabla UACI_ICBatchOffers con los códigos de oferta utilizados en las tablas de ofertas predeterminadas o de alteración temporal de puntuaciones. La tabla tiene el siguiente formato.

Nombre de columna	Tipo	Descripción
ICName	varchar(64)	El nombre del canal interactivo con el que está asociada la oferta. Si utiliza la misma oferta con dos canales interactivos diferentes, debe proporcionar una fila para cada canal interactivo.
OfferCode1	varchar(64)	La primera parte del código de oferta.
OfferCode2	varchar(64)	La segunda parte del código de oferta, si es necesario.
OfferCode3	varchar(64)	La tercera parte del código de oferta, si es necesario.
OfferCode4	varchar(64)	La cuarta parte del código de oferta, si es necesario.
OfferCode5	varchar(64)	La quinta parte del código de oferta, si es necesario.

Acerca de la tabla de ofertas globales

La tabla de ofertas globales permite definir tratamientos en el nivel de audiencia. Por ejemplo, puede definir una oferta global para cada miembro de la audiencia Unidad familiar.

Puede definir valores globales para los siguientes elementos de la presentación de ofertas de Interact.

- Asignación de ofertas globales
- Puntuación del usuario de marketing global, mediante un número o una expresión
- Expresión booleana para filtrar las ofertas
- Probabilidad y ponderación de aprendizaje, si utiliza el aprendizaje incorporado de Interact
- Alteración temporal de aprendizaje global

Habilitación de la tabla de ofertas globales

Puede configurar el entorno de ejecución para asignar ofertas globales para un nivel de audiencia, aparte de lo definido en las reglas de tratamiento.

1. Cree una tabla llamada UACI_DefaultOffers en la base de datos de perfil.

   Puede crear la tabla UACI_DefaultOffers con las columnas correctas utilizando el archivo ddl aci_usrtab.
2. Establezca la propiedad enableDefaultOfferLookup en **true**.

Tabla de ofertas globales

La tabla de ofertas globales debe existir en la base de datos de perfil. Puede crear la tabla de ofertas globales, `UACI_DefaultOffers` ejecutando el script SQL `aci_usertab` en la base de datos de perfil. El script SQL `aci_usertab` se encuentra en el directorio `ddl` del directorio de instalación del entorno de ejecución.

Debe definir los campos `NivelAudiencia` y `CódigoOferta1` para cada fila. Los otros campos son opcionales para restringir adicionalmente las asignaciones de oferta o influenciar el aprendizaje incorporado en el nivel de audiencia.

Para obtener el mejor rendimiento, debe crear un índice en esta tabla en la columna de nivel de audiencia.

Nombre	Tipo	Descripción
NivelAudiencia	varchar(64)	(Necesario) Nombre del nivel de audiencia al que se asigna la oferta predeterminada, por ejemplo, `cliente` o `unidad familiar`. Este nombre debe coincidir con el nivel de audiencia tal como está definido en Campaign.
CódigoOferta1	varchar(64)	(Necesario) Código de oferta de la oferta predeterminada. Si los códigos de oferta se componen de varios campos, puede añadir columnas adicionales, por ejemplo, `CódigoOferta2` y así sucesivamente Si está añadiendo esta oferta para proporcionar una asignación de oferta global, debe añadir esta oferta a la tabla `UACI_ICBatchOffers`.
Puntuación	float	Número para definir la puntuación de marketing de esta asignación de oferta.
IdTipoAnulación	int	Si se establece en `1`, si la oferta no existe en la lista candidata de ofertas, añada esta oferta a la lista y utilice los datos de puntuación para la oferta. Normalmente, utilice 1 para proporcionar asignaciones de oferta global. Si se establece en `0`, *nulo* o cualquier otro número que no sea 1, utilice los datos de la oferta solo si la oferta existe en la lista candidata de ofertas. En la mayoría de los casos, una regla de tratamiento o asignación individual sustituirá este valor.

Nombre	Tipo	Descripción
Predicado	varchar(4000)	Puede especificar expresiones en esta columna de la misma forma que para las opciones avanzadas de reglas de tratamiento. Puede utilizar las mismas variables y macros que están disponibles al escribir las opciones avanzadas de las reglas de tratamiento. El comportamiento de esta columna depende del valor de la columna `HabilitarIDestado`. • Si `HabilitarIDestado` es 2, esta columna funciona de la misma forma que la opción **Considerar esta regla eligible si la siguiente expresión es true** en las opciones avanzadas de las reglas de tratamiento que restringen esta asignación de oferta. Esta columna debe contener una expresión booleana y resolverse en true para incluir esta oferta. Si define accidentalmente una expresión que se resuelve en un número, un número distinto de cero se considera true y cero se considera false. • Si `HabilitarIDestado` es 3, esta columna funciona de la misma forma que la opción **Utilizar la siguiente expresión como puntuación de marketing** de las opciones avanzadas de las reglas de tratamiento para restringir esta oferta. Esta columna debe contener una expresión que se resuelva en un número. • Si `HabilitarIDestado` es 1, Interact ignora cualquier valor de esta columna.
PuntuaciónFinal	float	Número para sustituir la puntuación final utilizada para ordenar la lista final de ofertas devueltas. Esta columna se utiliza si se habilitado el módulo de aprendizaje incorporado. Puede implementar su propio aprendizaje para utilizar esta columna.
Códigodecelda	varchar(64)	Código de celda para un segmento interactivo al que desea asignar esta oferta predeterminada. Si los códigos de celda se componen de varios campos, puede añadir las columnas adicionales. Debe proporcionar un código de celda si `IdTipoAnulación` es 0 o nulo. Si no incluye un código de celda, el entorno de ejecución ignora esta fila de datos. Si `IdTipoAnulación` es 1, no necesita proporcionar un código de celda en esta columna. Si no proporciona un código de celda, el entorno de ejecución utiliza el código de celda definido en la propiedad `DefaultCellCode` para este nivel de audiencia y tabla para fines de creación de informes.
Zona	varchar(64)	Nombre de la zona a la que desea que se aplique esta asignación de oferta. Si es NULL, se aplicará a todas las zonas.

Nombre	Tipo	Descripción
HabilitarIDestado	int	El valor de esta columna define el comportamiento de la columna `Predicado`. • **1**: no utilizar la columna `Predicado`. • **2**: utilizar `Predicado` como booleano para filtrar la oferta. Permite las mismas reglas que la opción avanzada **Considerar esta regla elegible si la siguiente expresión es true** de una regla de tratamiento. • **3**: utilizar `Predicado` para definir la puntuación del usuario de marketing. Sigue las mismas reglas que la opción avanzada **Utilizar la siguiente expresión como puntuación de marketing** de una regla de tratamiento. Cualquier fila donde esta columna sea nulo o cualquier valor que no sea 2 ó 3 ignora la columna `Predicado`.
PuntuaciónProbabilidad	float	Esta columna se utiliza solo para influenciar al aprendizaje incorporado. Puede añadir esta columna con la DDL `aci_scoringfeature`.
PuntuaciónExplorar	float	Esta columna se utiliza solo para influenciar al aprendizaje incorporado. Puede añadir esta columna con la DDL `aci_scoringfeature`.

Acerca de la tabla de alteración temporal de puntuaciones

La tabla de alteración temporal de puntuaciones permite definir los tratamientos en un ID de audiencia o un nivel individual. Por ejemplo, si el nivel de audiencia es Visitante, puede crear alteraciones temporales para visitantes específicos.

Puede definir alteraciones temporal para los siguientes elementos de la presentación de ofertas de Interact.

- Asignación de ofertas individuales
- Puntuación del usuario de marketing individual, mediante un número o una expresión
- Expresión booleana para filtrar las ofertas
- Probabilidad y ponderación de aprendizaje, si utiliza el aprendizaje incorporado
- Alteración temporal de aprendizaje individual

Habilitación de la tabla de alteración temporal de puntuaciones

Puede configurar Interact para utilizar una puntuación generada a partir de una aplicación de modelado en lugar de la puntuación de marketing.

1. Cree una tabla de alteración temporal de puntuaciones para cada nivel de audiencia para el que desea proporcionar alteraciones temporales.

   Puede crear una tabla de alteración temporal de puntuaciones de muestra con las columnas correctas utilizando el archivo ddl `aci_usrtab`.
2. Establezca la propiedad `enableScoreOverrideLookup` en **true**.

3. Establezca la propiedad `scoreOverrideTable` en el nombre de la tabla de alteración temporal de puntuaciones para cada nivel de audiencia para el que desea proporcionar alteraciones temporales.

   No es necesario proporcionar una tabla de alteración temporal de puntuaciones para cada nivel de audiencia.

Tabla de sustituciones de puntuación

La tabla de sustituciones de puntuación debe existir en la base de datos de perfil de producción. Puede crear una tabla de sustituciones de puntuación de muestra, `UACI_ScoreOverride`, ejecutando el script SQL `aci_usertab` en la base de datos de perfil. El script SQL `aci_usertab` se encuentra en el directorio `ddl` del directorio de instalación del entorno de ejecución.

Debe definir los campos *`IDAudiencia`*, `CódigoOferta1` y `Puntuación` para cada fila. Los valores de los demás campos son opcionales para restringir adicionalmente asignaciones de oferta individuales o proporcionar información de sustitución de puntuación para aprendizaje incorporado.

Nombre	Tipo	Descripción
IDAudiencia	varchar(64)	(Necesario) El nombre de esta columna debe coincidir con el nombre de la columna que define el ID de audiencia en Campaign. La tabla de muestra creada por el archivo DLL `aci_usertab` crea esta columna como una columna `IDCliente`. Si el ID de audiencia consta de varias columnas, puede añadirlas a esta tabla. Cada fila debe contener el ID de audiencia al que asigna la oferta individual, por ejemplo, `cliente1`. Para obtener un mejor rendimiento, debe crear un índice en esta columna.
CódigoOferta1	varchar(64)	(Necesario) Código de oferta para la oferta. Si los códigos de oferta se componen de varios campos, puede añadir columnas adicionales, por ejemplo, `CódigoOferta2` y así sucesivamente Si está añadiendo esta oferta para proporcionar una asignación de oferta individual, debe añadir esta oferta a la tabla `UACI_ICBatchOffers`.
Puntuación	float	Número para definir la puntuación de marketing de esta asignación de oferta.
IdTipoAnulación	int	Si se establece `0` o *nulo* (o cualquier número distinto de 1), utilice datos de la oferta solo si la oferta existe en la lista candidata de ofertas. Normalmente, utilice `0` para proporcionar sustituciones de puntuación. Debe proporcionar un código de celda. Si se establece en `1`, si la oferta no existe en la lista candidata de ofertas, añada esta oferta a la lista y utilice los datos de puntuación para la oferta. Normalmente, utilice 1 para proporcionar asignaciones de oferta individuales.

Nombre	Tipo	Descripción
Predicado	varchar(4000)	Puede especificar expresiones en esta columna de la misma forma que para las opciones avanzadas de reglas de tratamiento. Puede utilizar las mismas variables y macros que están disponibles al escribir las opciones avanzadas de las reglas de tratamiento. El comportamiento de esta columna depende del valor de la columna `HabilitarIDestado`. • Si `HabilitarIDestado` es `2`, esta columna funciona de la misma forma que la opción **Considerar esta regla eligible si la siguiente expresión es true** en las opciones avanzadas de las reglas de tratamiento que restringen esta asignación de oferta. Esta columna debe contener una expresión booleana y resolverse en true para incluir esta oferta. Si define accidentalmente una expresión que se resuelve en un número, un número distinto de cero se considera true y cero se considera false. • Si `HabilitarIDestado` es `3`, esta columna funciona de la misma forma que la opción **Utilizar la siguiente expresión como puntuación de marketing** de las opciones avanzadas de las reglas de tratamiento para restringir esta oferta. Esta columna debe contener una expresión que se resuelva en un número. • Si `HabilitarIDestado` es `1`, Interact ignora cualquier valor de esta columna.
PuntuaciónFinal	float	Número para sustituir la puntuación final utilizada para ordenar la lista final de ofertas devueltas. Esta columna se utiliza si se habilitado el módulo de aprendizaje incorporado. Puede implementar su propio aprendizaje para utilizar esta columna.
Códigodecelda	varchar(64)	Código de celda de un segmento interactivo al que desea asignar esta oferta. Si los códigos de celda se componen de varios campos, puede añadir las columnas adicionales. Debe proporcionar un código de celda si `IdTipoAnulación` es `0` o nulo. Si no incluye un código de celda, el entorno de ejecución ignora esta fila de datos. Si `IdTipoAnulación` es `1`, no necesita proporcionar un código de celda en esta columna. Si no proporciona un código de celda, el entorno de ejecución utiliza el código de celda definido en la propiedad `DefaultCellCode` para este nivel de audiencia y tabla para fines de creación de informes.
Zona	varchar(64)	Nombre de la zona a la que desea que se aplique esta asignación de oferta. Si es NULL, se aplicará a todas las zonas.

Nombre	Tipo	Descripción
HabilitarIDestado	int	El valor de esta columna define el comportamiento de la columna `Predicado`. • **1**: no utilizar la columna `Predicado`. • **2**: utilizar `Predicado` como booleano para filtrar la oferta. Permite las mismas reglas que la opción avanzada **Considerar esta regla elegible si la siguiente expresión es true** de una regla de tratamiento. • **3**: utilizar `Predicado` para definir la puntuación del usuario de marketing. Sigue las mismas reglas que la opción avanzada **Utilizar la siguiente expresión como puntuación de marketing** de una regla de tratamiento. Cualquier fila donde esta columna sea nulo o cualquier valor que no sea 2 ó 3 ignora la columna `Predicado`.
PuntuaciónProbabilidad	float	Esta columna se utiliza solo para influenciar al aprendizaje incorporado. Puede añadir esta columna con la DDL `aci_scoringfeature`.
PuntuaciónExplorar	float	Esta columna se utiliza solo para influenciar al aprendizaje incorporado. Puede añadir esta columna con la DDL `aci_scoringfeature`.

Visión general del aprendizaje incorporado de Interact

Aunque haga todo lo posible para asegurarse de que propone las ofertas correctas a los segmentos correctos, siempre puede aprender algo de las propias selecciones de los visitantes. El comportamiento real de los visitantes debe influir en su estrategia. Puede ejecutar el historial de respuestas utilizando herramientas de modelado para obtener una puntuación e incluirla en los diagramas de flujo interactivos. No obstante, estos datos no están tiempo real.

Interact ofrece dos opciones para aprender de las acciones del visitante en tiempo real:

- Módulo de aprendizaje incorporado — El entorno de ejecución tiene un módulo de aprendizaje basado en Naive Bayesian. Este módulo supervisa los atributos de los clientes que elija y utiliza esos datos para ayudarle a seleccionar qué ofertas debe presentar.
- API de aprendizaje — El entorno de ejecución también tiene una API de aprendizaje para que escriba su propio módulo de aprendizaje.

No es obligatorio utilizar el aprendizaje. De forma predeterminada, el aprendizaje está inhabilitado.

Aprendizaje de Interact

El módulo de aprendizaje de Interact supervisa las respuestas del visitante a las ofertas y los atributos de visitante. El módulo de aprendizaje tiene dos modos generales:

- **Exploración** - el módulo de aprendizaje sirve las ofertas para recopilar datos de respuesta suficientes para optimizar el cálculo que se utiliza posteriormente durante la explotación. Las ofertas servidas durante la exploración no reflejan necesariamente la opción óptima.

- **Explotación** - una vez recopilados suficientes datos en la fase de exploración, el módulo de aprendizaje utiliza las probabilidades para ayudarle a seleccionar qué ofertas se deben presentar.

El módulo de aprendizaje alterna entre la exploración y la explotación basándose en dos propiedades: un nivel de confianza que configura con la propiedad `confidenceLevel` y una probabilidad de que el módulo de aprendizaje presente una oferta aleatoria que configura con la propiedad `percentRandomSelection`.

Debe establecer `confidenceLevel` en un porcentaje que representa el grado de seguridad (o confianza) que debe tener el módulo de aprendizaje para que se utilicen sus puntuaciones de una oferta en el arbitraje. Al principio, cuando el módulo de aprendizaje no tiene datos a partir de los que trabajar, depende totalmente de la puntuación de marketing. Cuando cada oferta se ha presentado tantas veces como se define en `minPresentCountThreshold`, el módulo de aprendizaje entra en el modo de exploración. Sin una gran cantidad de datos con los que trabajar, el módulo de aprendizaje no está seguro de que los porcentajes que calcula son correctos. Por lo tanto, permanece en el modo de exploración.

El módulo de aprendizaje asigna ponderaciones a cada oferta. Para calcular las ponderaciones, el módulo de aprendizaje utiliza una fórmula que toma como entrada el nivel de confianza configurado, así como los datos de aceptación históricos y los datos de la sesión actual. La fórmula establece un equilibrio intrínseco entre la exploración y explotación, y devuelve la ponderación adecuada.

Para garantizar que el sistema favorece las ofertas con un mayor rendimiento durante las primeras fases, Interact presenta una oferta aleatoria el `percentRandomSelection` por ciento del tiempo. Esto obliga al módulo de aprendizaje a recomendar ofertas que no sean las más satisfactorias para determinar si las otras ofertas tendrían un mayor rendimiento si estuvieran más expuestas. Por ejemplo, si configura `percentRandomSelection` en 5, esto significa que el módulo de aprendizaje presenta una oferta aleatoria el 5% del tiempo y añade los datos de respuesta a sus cálculos.

El módulo de aprendizaje determina qué ofertas se presentan de la siguiente forma.

1. Calcula la probabilidad de que un visitante seleccione una oferta.
2. Calcula la ponderación de la oferta basándose en la probabilidad del paso 1 y determina si debe estar en modo de exploración o explotación.
3. Calcula una puntuación final para cada oferta utilizando la puntuación de marketing y la ponderación de oferta del paso 2.
4. Clasifica las ofertas según las puntuaciones determinadas en el paso 3 y devuelve el número solicitado de ofertas principales.

Por ejemplo, el módulo de aprendizaje determina que un visitante tiene un 30% de probabilidades de aceptar una oferta A y un 70% de probabilidades de aceptar una oferta B, y que debe explotar esta información. A partir de las reglas de tratamiento, la puntuación de marketing de la oferta A es 75 y 55 para la oferta B. Sin embargo, los cálculos del paso 3 convierten la puntuación final de la oferta B en mayor que la de la oferta A. Por lo tanto, el entorno de ejecución recomienda la oferta B.

El aprendizaje también se basa en la propiedad `recencyWeightingFactor` y en la propiedad `recencyWeightingPeriod`. Estas propiedades permiten aumentar la ponderación de los datos más recientes. `recencyWeightingFactor` es el porcentaje

de ponderación que deben tener los datos recientes. recencyWeightingPeriod es el período de tiempo que se considera reciente. Por ejemplo, puede configurar recencyWeightingFactor en 0,30 y recencyWeightingPeriod en 24. Esto significa que las últimas 24 horas de datos, se pondera el 30% de todos los datos. Si tiene el equivalente a una semana de datos, todos los datos promediados en los primeros seis días constituyen el 70% de los datos y en el último día constituyen el 30% de los datos.

Cada sesión escribe los siguientes datos en una tabla de preparación de aprendizaje:

- Contacto de oferta
- Aceptación de oferta
- Atributos de aprendizaje

En un intervalo configurable, un agregador lee los datos de la tabla de preparación, los compila y los escribe en una tabla. El módulo de aprendizaje lee estos datos agregados y los utiliza en los cálculos.

Habilitación del módulo de aprendizaje

Todos los servidores de ejecución tienen un módulo de aprendizaje incorporado. De forma predeterminada, este módulo de aprendizaje está inhabilitado. Puede habilitar el aprendizaje cambiando una propiedad de configuración.

En la Marketing Platform del entorno de ejecución, edite las siguientes propiedades de configuración en la categoría Interact > offerserving.

Propiedad de configuración	Valor
optimizationType	**BuiltInLearning**

Atributos de aprendizaje

El módulo de aprendizaje aprende utilizando los atributos de visitante y los datos de aceptación de ofertas. Puede seleccionar qué atributos de visitante desea supervisar. Estos atributos de visitante pueden ser cualquier elemento en un perfil de cliente, por ejemplo, un atributo almacenado en una tabla de dimensiones referenciada en un diagrama de flujo interactivo o algún parámetro de evento que se recopila en tiempo real.

Aunque puede configurar el número de atributos que desee para supervisarlos, IBM recomienda no configurar más de diez atributos de aprendizaje entre los atributos de aprendizaje estáticos y dinámicos, y seguir estas directrices.

- Seleccione atributos independientes.

  No seleccione atributos que sean similares. Por ejemplo, si crea un atributo denominado HighValue y este atributo se define mediante un cálculo basado en el salario, no seleccione HighValue y Salary. Los atributos similares no ayudan al algoritmo de aprendizaje.
- Seleccione atributos con valores discretos.

  Si un atributo tiene rangos de valores, debe seleccionar un valor exacto. Por ejemplo, si desea utilizar el salario como un atributo, debe dar a cada rango de salarios un valor específico, por ejemplo, el rango 20.000-30.000 debe ser A, 30.001-40.000 debe ser B, etc.

- Limite el número de atributos a los que realiza un seguimiento para no disminuir el rendimiento.

  El número de atributos que puede rastrear depende de los requisitos de rendimiento y de la instalación de Interact. Si puede, utilice otra herramienta de modelado (por ejemplo, PredictiveInsight) para determinar los diez primeros atributos predictivos. Puede configurar el módulo de aprendizaje para podar automáticamente los atributos que no sean predictivos, pero esto también afecta al rendimiento.

Puede gestionar el rendimiento definiendo el número de atributos que desea supervisar y el número de valores por atributo que se van a supervisar. La propiedad `maxAttributeNames` define el número máximo de atributos de visitante a los que puede realizar un seguimiento. La propiedad `maxAttributeValues` define el número máximo de valores que puede rastrear por atributo. Los demás valores se asignan a una categoría definida por el valor de la propiedad `otherAttributeValue`. Sin embargo, el motor de aprendizaje sólo realiza un seguimiento de los primeros valores que encuentra. Por ejemplo, supongamos que realiza un seguimiento del atributo de visitante de color de ojos. Sólo está interesado en los valores azul, marrón y verde, por lo que establece `maxAttributeValues` en 3. Sin embargo, los tres primeros visitantes tiene los valores azul, marrón y avellana. Esto significa que a todos los visitantes con los ojos verdes se les asigna `otherAttributeValue`.

También puede utilizar atributos de aprendizaje dinámicos que permiten definir los criterios de aprendizaje más específicamente. Los atributos de aprendizaje dinámicos permiten aprender de la combinación de dos atributos como una sola entrada. Por ejemplo, supongamos la siguiente información de perfil.

ID del visitante	Tipo de tarjeta	Saldo de tarjeta
1	Tarjeta Oro	$1.000
2	Tarjeta Oro	$9.000
3	Tarjeta Bronce	$1.000
4	Tarjeta Bronce	$9.000

Si utiliza los atributos de aprendizaje estándar, sólo puede aprender del tipo de tarjeta y el saldo individualmente. Los visitantes 1 y 2 se agruparán basándose en el Tipo de tarjeta, y los visitantes 2 y 4 se agruparán basándose en el Saldo de tarjeta. Este puede que no sea un indicador preciso de un comportamiento de aceptación de oferta. Si los titulares de la tarjeta Oro tienden a tener saldos mayores, el comportamiento del visitante 2 puede ser radicalmente distinto al del visitante 4, lo que puede sesgar los atributos de aprendizaje estándar. Sin embargo, si se utilizan atributos de aprendizaje dinámicos, se aprende sobre cada uno de estos visitantes individualmente y las predicciones serán más precisas.

Si utiliza atributos de aprendizaje dinámicos y el visitante tiene dos valores válidos para un atributo, el módulo de aprendizaje selecciona el primer valor que encuentra.

Si establece la propiedad `enablePruning` en `yes`, el módulo de aprendizaje determina mediante algoritmos qué atributos no son predictivos y deja de tener en cuenta esos atributos cuando calcula las ponderaciones. Por ejemplo, si realiza un seguimiento de un atributo que representa el color de cabello y el módulo de aprendizaje determina que no existe ningún patrón para aceptar una oferta según el color de cabello del visitante, el módulo de aprendizaje deja de tener en cuenta

el atributo de color de cabello. Los atributos se vuelven a evaluar cada vez que se ejecuta el proceso de agregación de aprendizaje (definido por la propiedad aggregateStatsIntervalInMinutes). Los atributos de aprendizaje dinámicos también se podan.

Definición de un atributo de aprendizaje

Puede configurar atributos de visitantes hasta el número especificado en maxAttributeNames.

En la Marketing Platform del entorno de diseño, edite las siguientes propiedades de configuración en la categoría Campaign > partitions > partition*n* > Interact > learning.

(learningAttributes) es una plantilla para crear nuevos atributos de aprendizaje. Debe especificar un nuevo nombre para cada atributo. No puede crear dos categorías con el mismo nombre.

Propiedad de configuración	Valor
attributeName	attributeName debe coincidir con el nombre de un par nombre-valor en los datos del perfil. Este nombre es sensible a mayúsculas y minúsculas.

Definir atributos de aprendizaje dinámicos

Para definir los atributos de aprendizaje dinámicos, debe completar la tabla UACI_AttributeList del origen de datos Aprendizaje.

Todas las columnas de esta tabla tienen el tipo de varchar(64).

Columna	Descripción
AttributeName	Nombre del atributo sobre el que desea obtener información. Debe ser un valor real posible en AttributeNameCol.
AttributeNameCol	Nombre de columna completo (estructura jerárquica, empezando desde la tabla de perfil) donde se puede encontrar AttributeName. No es necesario que este nombre de columna sea el atributo de aprendizaje estándar.
AttributeValueCol	Nombre de columna completo (estructura jerárquica, empezando desde la tabla de perfil) donde se puede encontrar el valor asociado para AttributeName.

Por ejemplo, considere la siguiente tabla de perfil y su tabla de dimensiones asociada.

Tabla 5. MyProfileTable

VisitorID	KeyField
1	Key1
2	Key2
3	Key3
4	Key4

Tabla 6. MyDimensionTable

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

A continuación se muestra una tabla `UACI_AttributeList` de muestra que se correlaciona según el tipo de tarjeta y el saldo.

Tabla 7. UACI_AttributeList

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

Habilitación del aprendizaje externo

Puede utilizar la API Java de aprendizaje para escribir su propio módulo de aprendizaje. Debe configurar el entorno de ejecución para que reconozca su utilidad de aprendizaje en la Marketing Platform.

En la Marketing Platform del entorno de ejecución, edite las siguientes propiedades de configuración en la categoría `Interact > offerserving`. Las propiedades de configuración de la API del optimizador de aprendizaje se encuentran en la categoría `Interact > offerserving > External Learning Config`.

Propiedad de configuración	Valor
`optimizationType`	**ExternalLearning**
`externalLearningClass`	Nombre de clase del aprendizaje externo
`externalLearningClassPath`	La ruta de la clase o los archivos jar en el servidor de ejecución para el aprendizaje externo. Si utiliza un grupo de servidores y todos los servidores de ejecución hacen referencia a la misma instancia de Marketing Platform, cada servidor debe tener una copia de la clase o los archivos jar en la misma ubicación.

Debe reiniciar el servidor de ejecución de Interact para que estos cambios entren en vigor.

Capítulo 5. Entender la API de Interact

Interact proporciona ofertas dinámicamente a una gran variedad de puntos de encuentro. Por ejemplo, puede configurar el entorno de ejecución y el punto de encuentro para enviar mensajes a los empleados del centro de atención al cliente informándoles sobre los mejores posibilidades de aumento de venta o venta cruzada para un cliente que ha llamado con un tipo específico de consulta de servicio. También puede configurar el entorno de ejecución y el punto de encuentro para proporcionar ofertas personalizadas a un cliente (visitante) que ha entrado en un área determinada del sitio web.

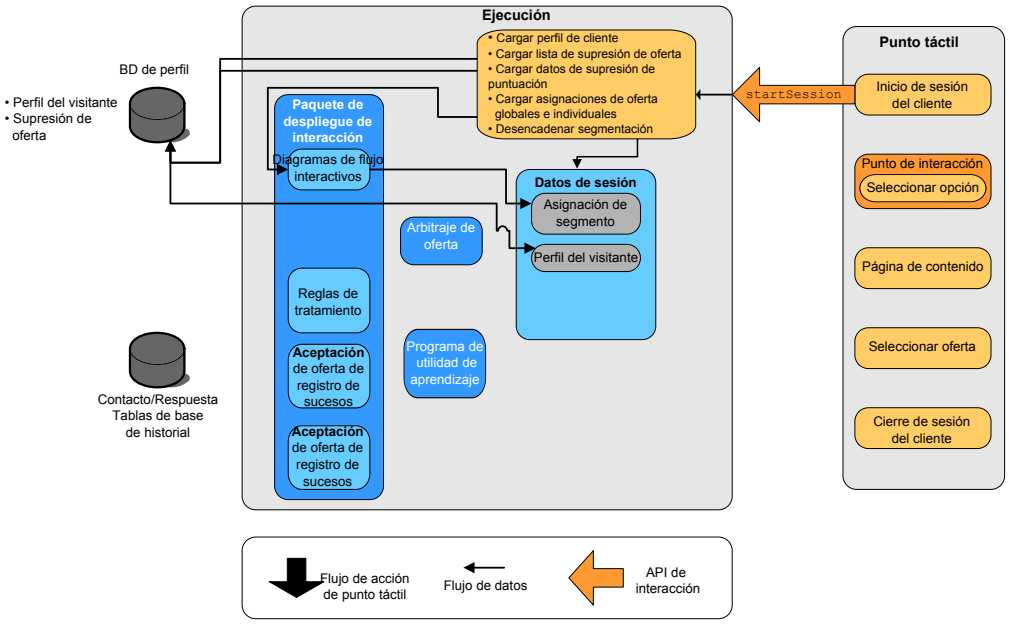
La interfaz de programación de aplicaciones (API) de Interact le permite configurar el punto de encuentro y un servidor de ejecución para que funcionen juntos para proporcionar las mejores ofertas posibles. Mediante la API, el punto de encuentro puede solicitar información del servidor de ejecución para asignar el visitante a un grupo (o segmento) y presentar ofertas basadas en ese segmento. También puede registrar datos para su posterior análisis para acotar las estrategias de presentación.

Para proporcionarle la mayor flexibilidad posible en la integración de Interact con sus entornos, IBM proporciona un servicio web accesible utilizando la API de Interact.

Flujo de datos de la API de Interact

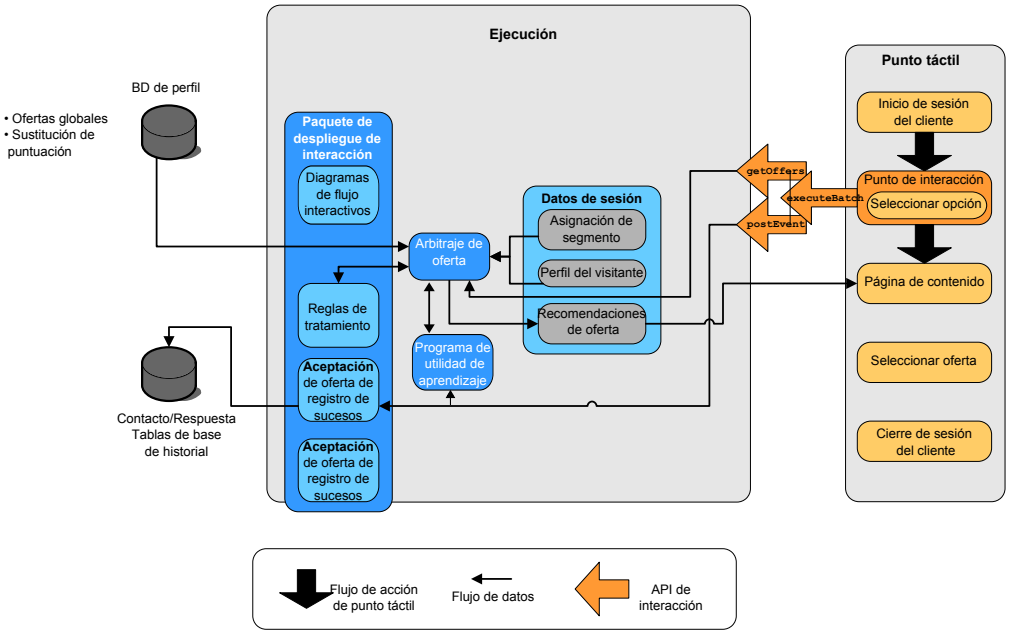
La siguiente figura se muestra una implementación simple de la API de Interact. Un visitante inicia la sesión en un sitio web y navega a una página que muestra las ofertas. El visitante selecciona una oferta y cierra la sesión. Mientras la interacción es simple, se producen varios eventos en el punto de encuentro y el servidor de ejecución.

Cuando un visitante inicia la sesión, se desencadena una `startSession`.



En este ejemplo, el método `startSession` realiza cuatro acciones. En primer lugar, crea una nueva sesión de ejecución. En segundo lugar, envía una solicitud para los datos de perfil del cliente en la sesión. En tercer lugar, envía una solicitud para utilizar los datos del perfil e iniciar un diagrama de flujo interactivo para incluir el cliente en segmentos. Esta ejecución de diagrama de flujo es asíncrona. En cuarto lugar, el entorno de ejecución carga la supresión de ofertas y la información de tratamiento de ofertas individuales y globales en la sesión. Los datos de sesión se mantienen en la memoria el tiempo que dura la sesión.

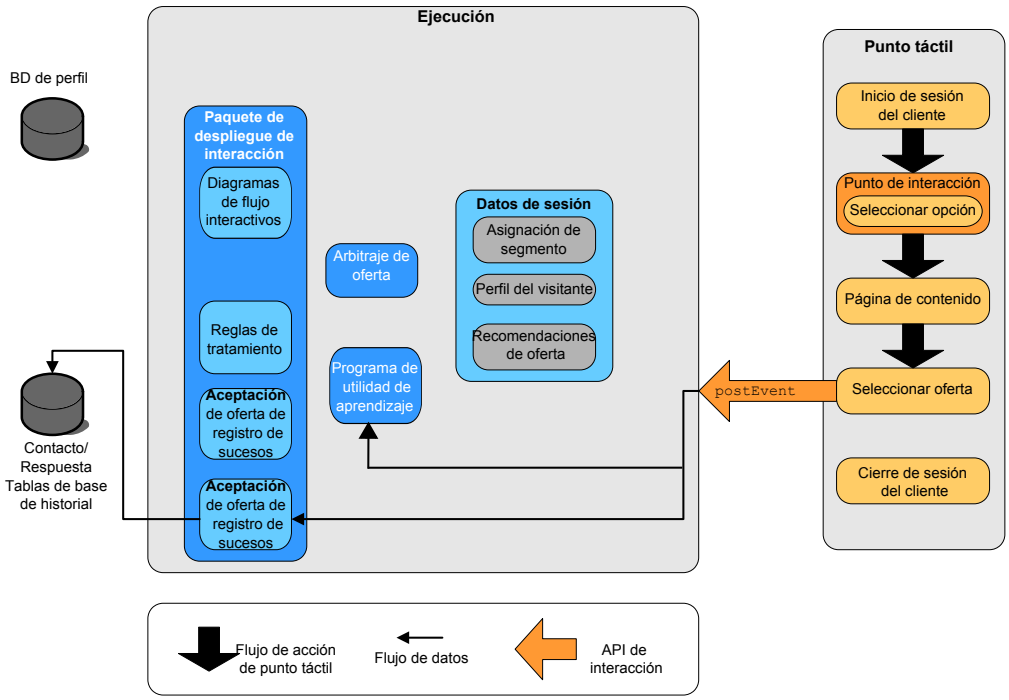
El visitante navega por el sitio hasta que llega a un punto de interacción predefinido. En la figura, el segundo punto de interacción (Seleccionar opción) es el lugar donde el visitante pulsa un enlace que presenta un conjunto de ofertas. El gestor de puntos de encuentro ha configurado el enlace para desencadenar un método `executeBatch`.



El método `executeBatch` permite llamar a más de un método en una sola llamada al servidor de ejecución. Estas llamadas a executeBatch llaman a dos otros métodos, `getOffers` y `postEvent`. El método `getOffers` solicita una lista de ofertas. El tiempo de ejecución utiliza los datos de segmentación, la lista de supresión de ofertas, las reglas de tratamiento y el módulo de aprendizaje para proponer un conjunto de ofertas. El tiempo de ejecución devuelve un conjunto de ofertas que se muestran en la página de contenidos.

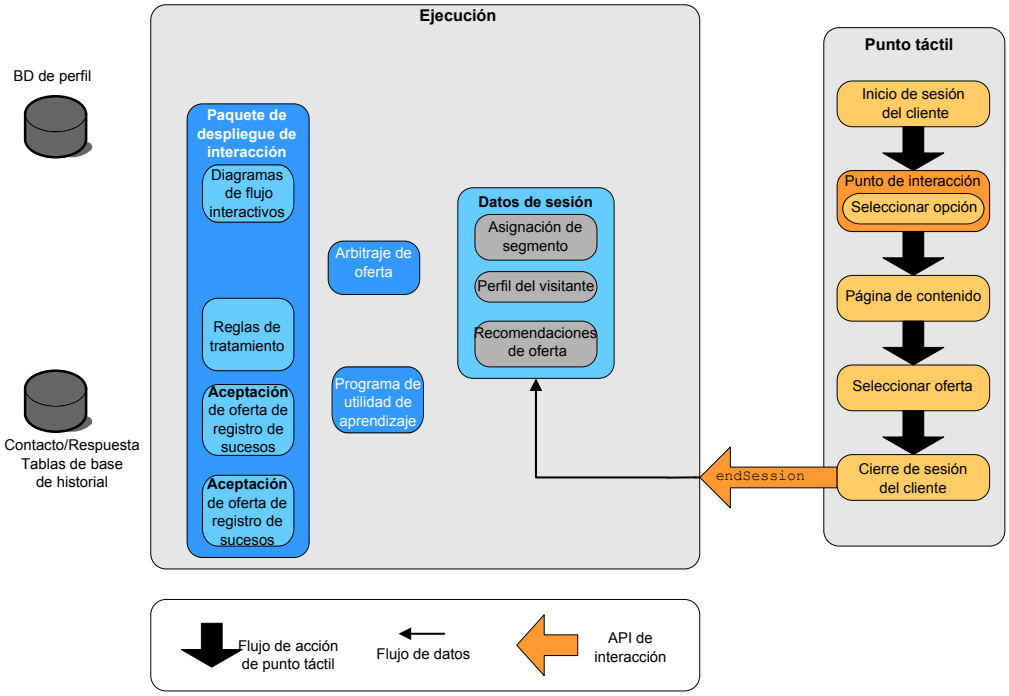
El método `postEvent` activa uno de los eventos definidos en el entorno de diseño. En este caso concreto, el evento envía una solicitud para registrar las ofertas presentadas al historial de contactos.

El visitante selecciona una de las ofertas (Seleccionar oferta).



El botón asociado con la selección de la oferta está configurado para enviar otro método `postEvent`. Este evento envía una solicitud para registrar la aceptación de la oferta en el historial de respuestas.

El visitante, después de seleccionar la oferta, termina con el sitio web y cierra la sesión. El comando de cerrar sesión está enlazado con el método `endSession`.



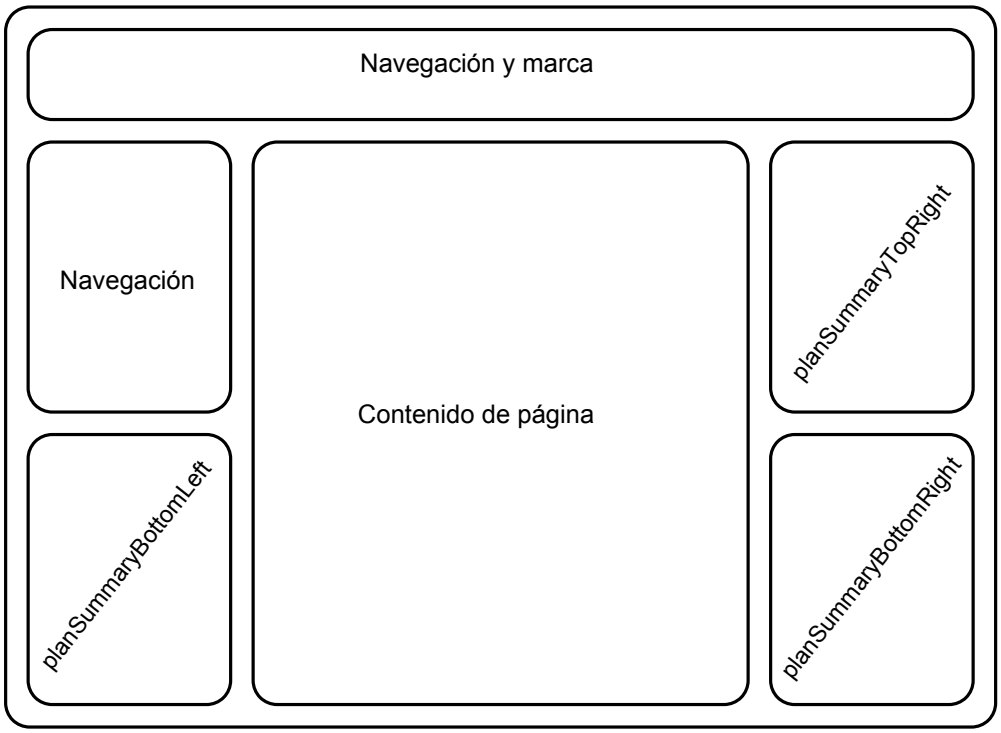
El método `endSession` cierra la sesión. Si el visitante olvida cerrar la sesión, existe un tiempo de espera de sesión configurable para garantizar que finalicen todas las sesiones al final. Si desea mantener algunos de los datos pasados a la sesión como, por ejemplo, la información incluida en los parámetros en los métodos

`startSession` o `setAudience`, trabaje con la persona que crea los diagramas de flujo interactivos. La persona que crea un diagrama de flujo interactivo puede utilizar el proceso Instantánea para escribir esos datos en una base de datos antes de la sesión finalice y se pierdan los datos. A continuación, puede utilizar el método `postEvent` para llamar al diagrama de flujo interactivo que contiene el proceso Instantánea.

Este ejemplo es muy simple (el visitante sólo realiza cuatro acciones —iniciar la sesión, ir a la página que muestra las ofertas, seleccionar una oferta y cerrar la sesión—, que es una interacción simple) para mostrar los conceptos básicos sobre cómo funciona la API entre su punto de encuentro y el entorno de ejecución. Puede diseñar la integración para que sea lo complicada que desee (dentro de los límites de sus requisitos de rendimiento).

Ejemplo de planificación de interacción simple

Supongamos que está diseñando una interacción para el sitio web de una empresa de teléfonos móviles. El siguiente diagrama muestra el diseño de la página de resumen del plan de teléfonos móviles.



Defina los siguientes elementos para cumplir los requisitos de la página de resumen del plan de teléfonos móviles.

Una oferta que se va a mostrar en una zona dedicada a ofertas acerca de las actualizaciones

- Debe definirse el área de la página que muestra la oferta de actualizaciones. Asimismo, una vez que Interact selecciona una oferta para visualizarla, la información debe registrarse.

  Punto de interacción: `ip_planSummaryBottomRight`

  Evento: `evt_logOffer`

Dos ofertas para actualizaciones de teléfono

- Debe definirse cada área de la página que muestra las actualizaciones de teléfono.

  Punto de interacción: `ip_planSummaryTopRight`

  Punto de interacción: `ip_planSummaryBottomLeft`

Para el análisis, debe registrar qué ofertas se aceptan y cuáles se rechazan.

Evento: `evt_offerAccept`

Evento: `evt_offerReject`

También sabe que debe pasar el código de tratamiento de una oferta siempre que registre un contacto de oferta, aceptación o rechazo. Siempre que sea necesario, debe crear un `NameValuePair` que contenga el código de tratamiento, como en el siguiente ejemplo.

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

Ahora puede pedir al usuario del entorno de diseño que cree los puntos de interacción y los eventos mientras empieza a codificar la integración con su punto de encuentro.

Para cada punto de interacción que va a mostrar una oferta, debe obtener primero una oferta y, a continuación, extraer la información que necesita para mostrar la oferta. Por ejemplo, solicite una oferta para el área inferior derecha de la página web (`planSummaryBottomRight`)

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

Se devolverá un objeto de respuesta que incluye una respuesta de `OfferList`. No obstante, la página web no puede utilizar un objeto `OfferList`. Necesita un archivo de imagen de la oferta, que sabe que es uno de los atributos de oferta (`offerImg`). Debe extraer el atributo de oferta que necesita de `OfferList`.

```
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"))
  {
  /* Utilizar este valor en el código de la página, por
   ejemplo: stringHtml = "<img src="+attribute.getValueAsString()+ " >   */
  }
 }
}
```

Ahora que está mostrando la oferta, desea registrarla como un contacto.

```
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)
```

En lugar de llamar a cada uno de estos métodos de forma individual, puede utilizar el método executeBatch, como se muestra en el siguiente ejemplo para la parte planSummaryBottomLeft de la página web.

```
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);

/** Crear la matriz de comandos */
Command[] commands =
{
 getOffersCommand,
 postEventCommand
};

/** Realizar la llamada */
BatchResponse batchResponse = api.executeBatch(sessionId, commands);
```

No es necesario definir el UACIOfferTrackingCode en este ejemplo porque el servidor de ejecución de Interact registra automáticamente la última lista de tratamientos recomendada como contactos si no proporciona el UACIOfferTrackingCode.

También ha escrito algo para cambiar la imagen que se visualiza cada 30 segundos para la segunda área en la página que muestra una oferta de actualización de teléfono. Ha decidido rotar entre tres imágenes, por lo que debe utilizar lo siguiente para recuperar el conjunto de ofertas que se deben guardar en caché para su uso en el código para rotar las imágenes.

```
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)
  {
   // obtener el valor de atributo offerimg y almacenarlo en algún lugar;
   // será la primera imagen que se muestre
  }
  else if(x==1)
  {
   // obtener el valor de atributo offerimg y almacenarlo en algún lugar;
   // será la segunda imagen que se muestre
  }
  else if(x==2)
  {
   // obtener el valor de atributo offerimg y almacenarlo en algún lugar;
   // será la tercera imagen que se muestre
  }
 }
}
```

Debe escribir el código de cliente captado en la memoria caché local y registrarlo en el contacto sólo una vez para cada oferta una vez visualizada su imagen. Para registrar el contacto, el parámetro UACITrackingCode debe publicarse como antes. Cada oferta tendrá un código de seguimiento diferente.

```
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);
  }
 }
}
```

Para cada oferta, si se pulsa la oferta, debe registrar la oferta aceptada y las ofertas rechazadas. (En este escenario, las ofertas no seleccionadas explícitamente se consideran rechazadas). A continuación, se muestra un ejemplo si se selecciona la oferta `ip_planSummaryTopRight`:

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

En la práctica, será mejor para enviar las tres llamadas `postEvent` con el método `executeBatch`.

Se trata de un ejemplo básico que no muestra la mejor forma de escribir la integración. Por ejemplo, ninguno de estos ejemplos incluye la comprobación de error utilizando la clase de respuesta.

Diseño de la integración de la API de Interact

Creación de la integración de la API de Interact con su punto de encuentro requiere algún tipo de diseño antes de iniciar la implementación. Debe trabajar con el equipo de marketing para decidir dónde desea que el entorno de ejecución presente las ofertas en el punto de encuentro (definir los puntos de interacción), y qué otro tipo de seguimiento o funcionalidad interactiva desea utilizar (definir los eventos). En la fase de diseño, pueden ser meros esquemas. Por ejemplo, para un sitio web de telecomunicaciones, la página de resumen del plan del cliente debe mostrar una oferta sobre la actualización de planes y dos ofertas de actualizaciones de teléfono.

Una vez la empresa ha decidido dónde y cómo desea interactuar con los clientes, debe utilizar Interact para definir los detalles. Un autor de diagrama de flujo debe diseñar los diagramas de flujo interactivos que se utilizarán cuando se produzcan

los eventos de resegmentación. Debe decidir el número y los nombres de los puntos de interacción y los eventos, así como qué datos deben pasarse para garantizar una segmentación adecuada, la publicación de eventos y la recuperación de ofertas. El usuario del entorno de diseño define los puntos de interacción y los eventos para el canal interactivo. A continuación, utilice esos nombres cuando codifique la integración con el punto de encuentro en el entorno de ejecución. También debe definir qué información de métricas es necesaria, para definir cuándo tiene que registrar los contactos y las respuestas de las ofertas.

Puntos a tener en cuenta

Recuerde los siguientes consejos cuando escriba una integración.

- Cuando diseñe el punto de encuentro, cree un contenido de relleno predeterminado(normalmente un mensaje de marca positivo o un contenido vacío) para cada punto de interacción donde se puedan presentar ofertas. Se aplica en el caso de que no se pueda elegir ninguna oferta para presentársela al visitante actual en la situación actual. Debe asignar este contenido de relleno predeterminado como la cadena predeterminada para el punto de interacción.
- Cuando diseñe el punto de encuentro, incluya algún método para presentar el contenido en el caso de que el punto de encuentro no pueda alcanzar el grupo de servidores de ejecución por algún motivo imprevisto.
- Cuando se desencadenan eventos que resegmentan el visitante, incluidos `postEvent` y `setAudience`, recuerde que la ejecución de diagramas de flujo necesita un tiempo adicional. El método `getOffers` espera a que la resegmentación finalice antes de ejecutarse. Una resegmentación demasiado frecuente puede dificultar el rendimiento de las respuestas a las llamada de `getOffers`.
- Deberá decidir qué significa un "rechazo de oferta". Varios informes como, por ejemplo, el informe Resumen de rendimiento de canal de oferta, muestran el número de veces que se ha rechazado una oferta. Es un recuento del número de veces que un `postEvent` ha desencadenado la acción Registrar rechazo de oferta. Debe determinar si Registrar rechazo de oferta es para un rechazo real (por ejemplo, la pulsación de un enlace con la etiqueta "No, gracias.") o para una oferta que se ignora (por ejemplo, una página que muestra tres anuncios distintos, ninguno de los cuales se selecciona).
- Existen varias características opcionales que puede habilitar para mejorar la selección de ofertas de Interact como, por ejemplo, el aprendizaje, de supresión de ofertas, las asignaciones de ofertas individuales y otros elementos de presentación de ofertas. Debe determinar cuántas, si existen, de estas características opcionales mejorarán sus interacciones.

Capítulo 6. Gestión de la API de IBM Unica Interact

Siempre que utiliza el método `startSession`, crea una sesión de ejecución de Interact en el servidor de ejecución. Puede utilizar las propiedades de configuración para gestionar las sesiones en un servidor de ejecución. Deberá configurar estos valores a medida que implemente la integración de Interact con el punto de encuentro.

Estas propiedades de configuración están en la categoría `sessionManagement`.

Configuración regional y la API de Interact

Puede utilizar Interact para los puntos de encuentro distintos del inglés. El punto de encuentro y todas las cadenas en la API utilizan la configuración regional definida para el usuario del entorno de ejecución.

Sólo puede seleccionar una configuración regional por grupo de servidores.

Por ejemplo, en el entorno de ejecución, supongamos que crea dos usuarios: asm_admin_en con la configuración regional del usuario establecida en inglés y asm_admin_fr con la configuración regional del usuario establecida en francés. Si el punto de encuentro está diseñado para francófonos, defina la propiedad `asmUserForDefaultLocale` para el entorno de ejecución como asm_admin_fr.

Acerca de la supervisión JMX

Interact proporciona el servicio de supervisión JMX (Java Management Extensions) al que puede acceder con cualquier aplicación de supervisión JMX. Esta supervisión JMX permite supervisar y gestionar los servidores de ejecución. Los atributos JMX proporcionan una gran cantidad de información detallada sobre el servidor de ejecución. Por ejemplo, el atributo JMX `ErrorCount` proporciona el número de mensajes de error registrados desde el último restablecimiento o el último inicio del sistema. Puede utilizar esta información para ver con qué frecuencia se producen errores en el sistema. Si ha codificado el sitio web para sólo invocar la finalización de la sesión si alguien completa una transacción, también puede comparar `startSessionCount` con `endSessionCount` para ver cuántas transacciones están incompletas.

Interact da soporte a los protocolos RMI y JMXMP, tal como se define en JSR 160. Puede conectarse al servicio de supervisión JMX con un cliente JMX compatible con JSR160.

Las diagramas de flujo interactivos sólo pueden supervisarse con la supervisión JMX. La información sobre los diagramas de flujo interactivos no aparecen en Campaign Monitoring.

Nota: Si utiliza IBM WebSphere con un gestor de nodos, debe definir el argumento JVM genérico para habilitar la supervisión JMX.

Configuración de Interact para utilizar la supervisión JMX con el protocolo RMI

En la Marketing Platform del entorno de ejecución, edite las siguientes propiedades de configuración en la categoría `Interact > monitoring`.

Propiedad de configuración	Valor
`protocol`	**RMI**
`port`	El número de puerto del servicio JMX
`enableSecurity`	**False** La implementación de Interact del protocolo RMI no da soporte a la seguridad.

La dirección predeterminada de supervisión para el protocolo RMI es `service:jmx:rmi:///jndi/rmi://`*`RuntimeServer`*`:`*`port`*`/interact`.

Configuración de Interact para utilizar la supervisión JMX con el protocolo JMXMP

El protocolo JMXMP requiere dos bibliotecas adicionales en el siguiente orden en la ruta de clase: `InteractJMX.jar` y `jmxremote_optional.jar`. Ambos archivos se pueden encontrar en el directorio `lib` de la instalación del entorno de ejecución.

Nota: Si habilita la seguridad, el nombre de usuario y la contraseña deben coincidir con un usuario en la Marketing Platform para el entorno de ejecución. No puede utilizar una contraseña vacía.

En la Marketing Platform del entorno de ejecución, edite las siguientes propiedades de configuración en la categoría `Interact > monitoring`.

Propiedad de configuración	Valor
`protocol`	**JMXMP**
`port`	El número de puerto del servicio JMX
`enableSecurity`	**False** para inhabilitar la seguridad, o **True** para habilitar la seguridad

La dirección predeterminada de supervisión para el protocolo JMXMP es `service:jmx:jmxmp://`*`RuntimeServer`*`:`*`port`*.

Utilización de los scripts de jconsole

Si no tiene una aplicación de supervisión JMX aparte, puede utilizar la jconsole que se instala con la JVM. Puede iniciar jconsole utilizando los scripts de inicio en el directorio `Interact/tools`.

1. Abra `Interact\tools\jconsole.bat` (Windows) o `Interact/tools/jconsole.sh` (Unix) en un editor de texto.
2. Establezca `INTERACT_LIB` en la ruta completa del directorio *`InteractInstallationDirectory`*`/lib`.
3. Establezca `HOST` en el nombre de host del servidor de ejecución que desea supervisar.

4. Establezca `PORT` en el puerto que ha configurado para que escuche JMX con la propiedad `Interact > monitoring > port`.
5. Si realiza la supervisión mediante el protocolo RMI, añada un comentario antes de la conexión JMXMP y elimínelo antes de la conexión RMI.

   El script realiza la supervisión a través del protocolo JMXMP de forma predeterminada.

   Por ejemplo, consulte los valores predeterminados para `jconsole.bat`.

   La conexión JMXMP

   ```
   %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%
   ```

   La conexión RMI

   ```
   %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
   ```

Atributos JMX

En las tablas siguientes se describen los atributos disponibles con la supervisión JMX.

Todos los datos proporcionados por JMX son desde el último restablecimiento o el último inicio del sistema. Por ejemplo, un recuento del número de elementos desde el último restablecimiento o el último inicio del sistema, no desde la instalación.

Tabla 8. Supervisor de ETL del historial de contactos y respuestas

Atributo	Descripción
AvgCHExecutionTime	El número promedio de milisegundos que el módulo del historial de contactos y respuestas necesita para escribir en la tabla del historial de contactos. Este promedio sólo se calcula para las operaciones que han sido satisfactorias y para las que se ha escrito un registro como mínimo en la tabla del historial de contactos.
AvgETLExecutionTime	El número promedio de milisegundos que el módulo del historial de contactos y respuestas necesita para leer datos del entorno de ejecución. El promedio incluye el tiempo de las operaciones satisfactorias y anómalas.
AvgRHExecutionTime	El número promedio de milisegundos que el módulo del historial de contactos y respuestas necesita para escribir en la tabla del historial de respuestas. Este promedio sólo se calcula para las operaciones que han sido satisfactorias y para las que se ha escrito un registro como mínimo en la tabla del historial de respuestas.
ErrorCount	El número de mensajes de error registrados desde el último restablecimiento o el último inicio del sistema, si existen.

Tabla 8. Supervisor de ETL del historial de contactos y respuestas (continuación)

Atributo	Descripción
HighWaterMarkCHExecutionTime	El número máximo de milisegundos que el módulo del historial de contactos y respuestas ha necesitado para escribir en la tabla del historial de contactos. Este valor sólo se calcula para las operaciones que han sido satisfactorias y para las que se ha escrito un registro como mínimo en la tabla del historial de contactos.
HighWaterMarkETLExecutionTime	El número máximo de milisegundos que el módulo del historial de contactos y respuestas ha necesitado para leer datos del entorno de ejecución. El cálculo incluye las operaciones satisfactorias y anómalas.
HighWaterMarkRHExecutionTime	El número máximo de milisegundos que el módulo del historial de contactos y respuestas ha necesitado para escribir en la tabla del historial de respuestas. Este valor sólo se calcula para las operaciones que han sido satisfactorias y para las que se ha escrito un registro como mínimo en la tabla del historial de respuestas.
LastExecutionDuration	El número de milisegundos que el módulo del historial de contactos y respuestas ha necesitado para realizar la última copia.
NumberOfExecutions	El número de veces que el módulo del historial de contactos y respuestas se ha ejecutado desde la última inicialización.
LastExecutionStart	La hora a la que se ha iniciado la última ejecución del módulo del historial de contactos y respuestas.
LastExecutionSuccessful	Si es true, la última ejecución del módulo del historial de contactos y respuestas ha sido satisfactoria. Si es false, se ha producido un error.
NumberOfContactHistoryRecordsMarked	El número de registros del historial de contactos en la tabla `UACI_CHStaging` que se mueven durante la ejecución actual del módulo del historial de contactos y respuestas. Este valor sólo es mayor que cero si se está ejecutando actualmente el módulo del historial de contactos y respuestas.

Tabla 8. Supervisor de ETL del historial de contactos y respuestas (continuación)

Atributo	Descripción
NumberOfResponseHistoryRecordsMarked	El número de registros del historial de respuestas en la tabla `UACI_RHStaging` que se mueven durante la ejecución actual del módulo del historial de contactos y respuestas. Este valor sólo es mayor que cero si se está ejecutando actualmente el módulo del historial de contactos y respuestas.

Los atributos del Supervisor de ETL del historial de contactos y respuestas forman parte del entorno de diseño. Todos los siguientes atributos forman parte del entorno de ejecución.

Tabla 9. Excepciones

Atributo	Descripción
errorCount	El número de mensajes de error registrados desde el último restablecimiento o el último inicio del sistema.
warningCount	El número de mensajes de aviso registrados desde el último restablecimiento o el último inicio del sistema.

Tabla 10. Estadísticas de motor de diagrama de flujo

Atributo	Descripción
activeProcessBoxThreads	Recuento activo de subprocesos del proceso de diagrama de flujo (compartidos entre todas las ejecuciones) que están actualmente en ejecución.
activeSchedulerThreads	Recuento activo de subprocesos del planificador de diagrama de flujo que están actualmente en ejecución.
avgExecutionTimeMillis	Promedio de tiempo de ejecución de diagrama de flujo en milisegundos.
CurrentJobsInProcessBoxQueue	El número de trabajos que están a la espera de ser ejecutados por los subprocesos del proceso de diagrama de flujo.
CurrentJobsInSchedulerQueue	El número de trabajos que están a la espera de ser ejecutados por los subprocesos del planificador de diagrama de flujo.
maximumProcessBoxThreads	Número máximo de subprocesos del proceso de diagrama de flujo (compartidos entre todas las ejecuciones) que pueden ejecutarse.
maximumSchedulerThreads	Número máximo de subprocesos del planificador de diagrama de flujo (un subproceso por ejecución) que pueden ejecutarse.

Tabla 10. Estadísticas de motor de diagrama de flujo (continuación)

Atributo	Descripción
numExecutionsCompleted	Número total de ejecuciones de diagrama de flujo que se han completado.
numExecutionsStarted	Número total de ejecuciones de diagrama de flujo que se han iniciado.

Tabla 11. Diagramas de flujo específicos por canal interactivo

Atributo	Descripción
AvgExecutionTimeMillis	Promedio de tiempo de ejecución en milisegundos para este diagrama de flujo en este canal interactivo.
HighWaterMarkForExecutionTime	Tiempo de ejecución máximo en milisegundos para este diagrama de flujo en este canal interactivo.
LastCompletedExecutionTimeMillis	Tiempo de ejecución en milisegundos para la última finalización de este diagrama de flujo en este canal interactivo.
NumExecutionsCompleted	Número total de ejecuciones que se han completado para este diagrama de flujo en este canal interactivo.
NumExecutionsStarted	Número total de ejecuciones que se han iniciado para este diagrama de flujo en este canal interactivo.

Tabla 12. Configuración regional

Atributo	Descripción
locale	Valor de configuración regional del cliente JMX.

Tabla 13. Configuración del registrador

Atributo	Descripción
category	Cambia la categoría de registro en la que puede manipularse el nivel de registro.

Tabla 14. Estadísticas de agrupación de subprocesos de servicios

Atributo	Descripción
activeContactHistThreads	El número aproximado de subprocesos que están ejecutando activamente tareas para el historial de contactos y el historial de respuestas.
activeFlushCacheToDBThreads	El número aproximado de subprocesos que están ejecutando activamente tareas para vaciar las estadísticas guardadas en caché en el almacén de datos.

Tabla 14. Estadísticas de agrupación de subprocesos de servicios (continuación)

Atributo	Descripción
activeOtherStatsThreads	El número aproximado de subprocesos que están ejecutando activamente tareas para Estadísticas elegibles, Actividades de eventos y Estadísticas predeterminadas.
CurrentHighWaterMarkInContactHistQueue	Número máximo de entradas en cola que va a registrar el servicio que recopila los datos del historial de contactos y respuestas.
CurrentHighWaterMark InFlushCachetoDBQueue	Número máximo de entradas en cola que va a registrar el servicio que escribe los datos de la memoria caché en las tablas de base de datos.
CurrentHighWaterMarkInOtherStatsQueue	Número máximo de entradas en cola que va a registrar el servicio que recopila las estadísticas de elegibilidad de ofertas, las estadísticas de uso de cadenas predeterminadas, las estadísticas de actividades de eventos y el registro personalizado en los datos de la tabla.
currentMsgsInContactHistQueue	El número de trabajos en la cola de la agrupación de subprocesos utilizada para el historial de contactos y el historial de respuestas.
currentMsgsInFlushCacheToDBQueue	El número de trabajos en la cola de la agrupación de subprocesos utilizada para vaciar las estadísticas guardadas en caché en el almacén de datos.
currentMsgsInOtherStatsQueue	El número de trabajos en la cola de la agrupación de subprocesos utilizada para Estadísticas elegibles, Actividades de eventos y Estadísticas predeterminadas.
maximumContactHistThreads	El número máximo de subprocesos que han estado simultáneamente en la agrupación utilizada para el historial de contactos y el historial de respuestas.
maximumFlushCacheToDBThreads	El número máximo de subprocesos que han estado simultáneamente en la agrupación utilizada para vaciar las estadísticas guardadas en caché en el almacén de datos.
maximumOtherStatsThreads	El número máximo de subprocesos que han estado simultáneamente en la agrupación utilizada para Estadísticas elegibles, Actividades de eventos y Estadísticas predeterminadas.

Las estadísticas de servicio están formadas por un conjunto de atributos para cada servicio.

- ContactHistoryMemoryCacheStatistics — El servicio que recopila datos para las tablas de preparación del historial de contactos.
- CustomLoggerStatistics — El servicio que recopila datos personalizados para escribir en una tabla (un evento que utiliza el parámetro de evento `UACICustomLoggerTableName`).
- Estadísticas predeterminadas — El servicio que recopila las estadísticas sobre el número de veces que se ha utilizado la cadena predeterminada para el punto de interacción.
- Estadísticas de elegibilidad — El servicio que escribe estadísticas para las ofertas elegibles.
- Estadísticas de actividades de eventos — El servicio que recopila las estadísticas de eventos, de eventos del sistema como `getOffer` o `startSession` y eventos de usuario desencadenados por `postEvent`.
- Estadísticas de memoria caché del historial de respuestas — El servicio que escribe en las tablas de preparación del historial de respuestas.
- Estadísticas de respuestas de sesiones cruzadas — El servicio que recopila los datos de seguimiento de respuestas de sesiones cruzadas.

Tabla 15. Estadísticas de servicios

Atributo	Descripción
Count	El número de mensajes procesados.
ExecTimeInsideMutex	La cantidad de tiempo necesaria para procesar mensajes para este servicio, excepto el tiempo a la espera de otros subprocesos, en milisegundos. Si hay una gran diferencia entre ExecTimeInsidMutex y ExecTimeMillis, deberá cambiar el tamaño de la agrupación de subprocesos para el servicio.
ExecTimeMillis	La cantidad de tiempo necesaria para procesar mensajes para este servicio, incluido el tiempo a la espera de otros subprocesos, en milisegundos.
ExecTimeOfDBInsertOnly	La cantidad de tiempo en milisegundos necesaria para procesar sólo la parte de inserción por lotes.
HighWaterMark	El número máximo de mensajes procesados para este servicio.
NumberOfDBInserts	El número total de inserciones por lotes ejecutadas.
TotalRowsInserted	El número total de filas insertadas en la base de datos.

Tabla 16. Estadísticas de servicios — Utilidad de carga de base de datos

Atributo	Descripción
ExecTimeOfWriteToCache	La cantidad de tiempo en milisegundos necesaria para escribir en la memoria caché de archivos, incluida la escritura en los archivos y la obtención de la clave primaria de la base de datos cuando sea necesario.

Tabla 16. Estadísticas de servicios — Utilidad de carga de base de datos (continuación)

Atributo	Descripción
ExecTimeOfLoaderDBAccessOnly	La cantidad de tiempo en milisegundos necesaria para ejecutar sólo la parte de cargador de base de datos.
ExecTimeOfLoaderThreads	La cantidad de tiempo en milisegundos necesaria para los subprocesos del cargador de base de datos.
ExecTimeOfFlushCacheFiles	La cantidad de tiempo en milisegundos necesaria para vaciar la memoria caché y volver a crear las nuevas.
ExecTimeOfRetrievePKDBAccess	La cantidad de tiempo en milisegundos necesaria para recuperar el acceso de base de datos de clave primaria.
NumberOfDBLoaderRuns	El número total de ejecuciones del cargador de base de datos.
NumberOfLoaderStagingDirCreated	El número total de directorios de preparación creados.
NumberOfLoaderStagingDirRemoved	El número total de directorios de preparación eliminados.
NumberOfLoaderStagingDirMovedToAttention	El número total de directorios de preparación renombrados como de atención.
NumberOfLoaderStagingDirMovedToError	El número total de directorios de preparación renombrados como de error.
NumberOfLoaderStagingDirRecovered	El número total de directorios de preparación recuperados, incluido en el inicio y los que se han vuelto a ejecutar en los subprocesos de fondo.
NumberOfTimesRetrievePKFromDB	El número total de veces que se recupera la clave primaria de la base de datos.
NumberOfLoaderThreadsRuns	El número total de ejecuciones de subprocesos del cargador de base de datos.
NumberOfFlushCacheFiles	El número total de veces que se vacía la memoria caché de archivos.

Tabla 17. Estadísticas de API

Atributo	Descripción
endSessionCount	El número de llamadas de API `endSession` desde el último restablecimiento o el último inicio del sistema.
endSessionDuration	Tiempo transcurrido para la última llamada de API `endSession`.
executeBatchCount	El número de llamadas de API `executeBatch` desde el último restablecimiento o el último inicio del sistema.

Tabla 17. Estadísticas de API (continuación)

Atributo	Descripción
executeBatchDuration	Tiempo transcurrido para la última llamada de API `executeBatch`.
getOffersCount	El número de llamadas de API `getOffers` desde el último restablecimiento o el último inicio del sistema.
getOffersDuration	Tiempo transcurrido para la última llamada de API `getOffer`.
getProfileCount	El número de llamadas de API `getProfile` desde el último restablecimiento o el último inicio del sistema.
getProfileDuration	Tiempo transcurrido para la última llamada de API `getProfileDuration`.
getVersionCount	El número de llamadas de API `getVersion` desde el último restablecimiento o el último inicio del sistema.
getVersionDuration	Tiempo transcurrido para la última llamada de API `getVersion`.
loadOfferSuppressionDuration	Tiempo transcurrido para la última llamada de API `loadOfferSuppression`.
LoadOffersBySQLCount	El número de llamadas de API `LoadOffersBySQL` desde el último restablecimiento o el último inicio del sistema.
LoadOffersBySQLDuration	Tiempo transcurrido para la última llamada de API `LoadOffersBySQL`.
loadProfileDuration	Tiempo transcurrido para la última llamada de API `loadProfile`.
loadScoreOverrideDuration	Tiempo transcurrido para la última llamada de API `loadScoreOverride`.
postEventCount	El número de llamadas de API `postEvent` desde el último restablecimiento o el último inicio del sistema.
postEventDuration	Tiempo transcurrido para la última llamada de API `postEvent`.
runSegmentationDuration	Tiempo transcurrido para la última llamada de API `runSegmentation`.
setAudienceCount	El número de llamadas de API `setAudience` desde el último restablecimiento o el último inicio del sistema.
setAudienceDuration	Tiempo transcurrido para la última llamada de API `setAudience`.
setDebugCount	El número de llamadas de API `setDebug` desde el último restablecimiento o el último inicio del sistema.

Tabla 17. Estadísticas de API (continuación)

Atributo	Descripción
setDebugDuration	Tiempo transcurrido para la última llamada de API `setDebug`.
startSessionCount	El número de llamadas de API `startSession` desde el último restablecimiento o el último inicio del sistema.
startSessionDuration	Tiempo transcurrido para la última llamada de API `startSession`.

Tabla 18. Estadísticas de optimizador de aprendizaje

Atributo	Descripción
LearningOptimizerAcceptCalls	El número de eventos de aceptación pasados al módulo de aprendizaje.
LearningOptimizerAcceptTrackingDuration	El número total de milisegundos necesarios para registrar los eventos de aceptación en el módulo de aprendizaje.
LearningOptimizerContactCalls	El número de eventos de contacto pasados al módulo de aprendizaje.
LearningOptimizerContactTrackingDuration	El número total de milisegundos necesarios para registrar los eventos de contacto en el módulo de aprendizaje.
LearningOptimizerLogOtherCalls	El número de eventos de no aceptación y de no contacto pasados al módulo de aprendizaje.
LearningOptimizerLogOtherTrackingDuration	La duración en milisegundos necesaria para registrar otros eventos (de no contacto y no aceptación) en el módulo de aprendizaje.
LearningOptimizerNonRandomCalls	El número de veces que se ha aplicado la implementación de aprendizaje configurada.
LearningOptimizerRandomCalls	El número de veces que se ha omitido la implementación de aprendizaje configurada y se ha aplicado la selección aleatoria.
LearningOptimizerRecommendCalls	El número de solicitudes de recomendación pasadas al módulo de aprendizaje.
LearningOptimizerRecommendDuration	El número total de milisegundos invertidos en la lógica de recomendación de aprendizaje.

Tabla 19. Estadísticas de ofertas predeterminadas

Atributo	Descripción
LoadDefaultOffersDuration	Tiempo transcurrido en la carga de ofertas predeterminadas.
DefaultOffersCalls	Número de veces que se realiza la carga de ofertas predeterminadas.

Operaciones de JMX

En la siguiente tabla se describen las operaciones disponibles para la supervisión JMX.

Grupo	Atributo	Descripción
Configuración del registrador	activateDebug	Establece el nivel de registro del archivo de registro definido en `Interact/conf/interact_log4j.properties` en Depurar.
Configuración del registrador	activateError	Establece el nivel de registro del archivo de registro definido en `Interact/conf/interact_log4j.properties` en Error.
Configuración del registrador	activateFatal	Establece el nivel de registro del archivo de registro definido en `Interact/conf/interact_log4j.properties` en Muy grave.
Configuración del registrador	activateInfo	Establece el nivel de registro del archivo de registro definido en `Interact/conf/interact_log4j.properties` en Información.
Configuración del registrador	activateTrace	Establece el nivel de registro del archivo de registro definido en `Interact/conf/interact_log4j.properties` en Rastreo.
Configuración del registrador	activateWarn	Establece el nivel de registro del archivo de registro definido en `Interact/conf/interact_log4j.properties` en Aviso.
Configuración regional	changeLocale	Cambia la configuración regional del cliente JMX. Las configuraciones regionales soportadas de Interact son `de`, `en`, `es` y `fr`.
ContactResponseHistory ETLMonitor	reset	Restablece todos los contadores.
Estadísticas de ofertas predeterminadas	updatePollPeriod	Actualiza defaultOfferUpdatePollPeriod. Este valor, en segundos, indica al sistema cuánto tiempo debe esperar antes de actualizar las ofertas predeterminadas en la memoria caché. Si se establece en `-1`, el sistema sólo lee el número de ofertas predeterminadas en el inicio.

Capítulo 7. Clases y métodos para la API de IBM Unica Interact

En las siguientes secciones se muestran de requisitos y otros detalles que debe conocer antes de empezar a trabajar con la API de Interact.

Nota: En esta sección se supone que está familiarizado con el punto de encuentro, el lenguaje de programación Java y el trabajo con una API basada en Java.

La API de Interact tiene un adaptador de cliente Java que utiliza la serialización Java a través de HTTP. Asimismo, Interact proporciona un WSDL para dar soporte a los clientes SOAP. El WSDL muestra el mismo conjunto de funciones que el adaptador de cliente Java, por lo que continúan aplicándose las siguientes secciones, excepto los ejemplos.

Clases de API de Interact

La API de Interact se basa en la clase `InteractAPI`. Existen 6 interfaces de soporte.

- `AdvisoryMessage`
- `BatchResponse`
- `NameValuePair`
- `Offer`
- `OfferList`
- `Response`

Estas interfaces tienen 3 clases de soporte concretas. Se debe crear una instancia de las siguientes dos clases concretas y se deben pasar como argumentos a los métodos de API de Interact:

- NameValuePairImpl
- CommandImpl

Una tercera clase concreta, denominada `AdvisoryMessageCode`, está disponible para proporcionar las constantes utilizadas para distinguir los códigos de mensaje devueltos del servidor siempre que sea aplicable.

El resto de esta sección describe los métodos que componen la API de Interact.

Serialización Java a través de los requisitos previos HTTP

1. Antes de empezar a trabajar con el adaptador de serialización Java, debe añadir el siguiente archivo a la `CLASSPATH`:

   `Interact_Runtime_Environment_Installation_Directory/lib/interact_client.jar`
2. Todos los objetos que se pasan entre el cliente y el servidor pueden encontrarse en el paquete `com.unicacorp.interact.api`. Consulte el JavaDoc de la API de Interact para obtener más información.
3. Para obtener una instancia de la clase InteractAPI, llame al método estático `getInstance` con el URL del servidor de ejecución de Interact.

Requisitos previos de SOAP

Importante: La prueba de rendimiento muestra que el adaptador de serialización Java tiene un rendimiento con una tasa mucho mayor que un cliente SOAP generado. Para garantizar el máximo rendimiento, utilice el adaptador de serialización Java siempre que sea posible.

Para acceder al servidor de ejecución utilizando SOAP, debe hacer lo siguiente:

1. Convierta el WSDL de la API de Interact utilizando el kit de herramientas SOAP que prefiera.

   El WSDL de la API de Interact se instala con Interact en el directorio `Interact/conf`.

   El texto del WSDL está disponible al final de esta guía.
2. Instale y configure el servidor de ejecución.

   El servidor de ejecución debe estar ejecutándose para probar completamente la integración.

Versiones de SOAP

Interact utiliza axis2 1.3 como la infraestructura SOAP en los servidores de ejecución de Interact. Para obtener información detallada sobre qué versiones de SOAP admite axis2 1.3, consulte el siguiente sitio web:

Apache Axis2

Interact se ha probado con los clientes SOAP de axis2, XFire, JAX-WS-Ri, DotNet, SOAPUI e IBM RAD.

JavaDoc de la API

Además de esta guía, el JavaDoc de la API de Interact se instala con el servidor de ejecución. El JavaDoc se instala para realizar consultas en el directorio `Interact/docs/apiJavaDoc`.

Acerca de los ejemplos de API

Todos los ejemplos de esta guía se han creado utilizando la serialización Java a través del adaptador HTTP. Si utiliza SOAP, como las clases generadas a partir de WSDL pueden variar según el kit de herramientas SOAP y las opciones que seleccione, estos ejemplos no funcionarán exactamente igual en su entorno.

Cómo trabajar con datos de sesión

Cuando inicia una sesión con el método `startSession`, los datos de sesión se cargan en la memoria. Durante la sesión, puede leer y escribir en los datos de sesión (que son un superconjunto de los datos de perfil estáticos). La sesión contiene los siguientes datos:

- Datos de perfil estáticos
- Asignaciones de segmento
- Datos en tiempo real
- Recomendaciones de ofertas

Todos los datos de sesión están disponibles hasta que invoca el método `endSession` o transcurre el tiempo de `sessionTimeout`. Una vez finalizada la sesión, todos los

datos que no se han guardado de forma explícita en el historial de contactos o respuestas o en otra tabla de base de datos se pierden.

Los datos se almacenan como un conjunto de pares nombre-valor. Si los datos se leen de una tabla de base de datos, el nombre es la columna de la tabla.

Puede crear estos pares nombre-valor cuando trabaje con la API de Interact. No es necesario declarar todos los pares nombre-valor en un área global. Si establece nuevos parámetros como pares nombre-valor, el entorno de ejecución añade los pares nombre-valor a los datos de sesión. Por ejemplo, si utiliza los parámetros de evento con el método `postEvent`, el entorno de ejecución añade los parámetros de evento a los datos de sesión, aunque los parámetros de evento no estén disponibles en los datos del perfil. Estos datos sólo existen en los datos de sesión.

Puede sobrescribir datos de sesión en cualquier momento. Por ejemplo, si parte del perfil de cliente incluye `creditScore`, puede pasar en un parámetro de evento utilizando el tipo personalizado `NameValuePair`. En la clase `NameValuePair`, puede utilizar los métodos `setName` y `setValueAsNumeric` para cambiar el valor. El nombre debe coincidir. En los datos de sesión, el nombre no es sensible a las mayúsculas y minúsculas. Por lo tanto, los nombres `creditscore` o `CrEdItScOrE` deben sobrescribir `creditScore`.

Sólo se mantienen los últimos datos escritos en los datos de sesión. Por ejemplo, `startSession` carga los datos del perfil del valor de `lastOffer`. Un método `postEvent` sobrescribe `lastOffer`. A continuación, un segundo método `postEvent` sobrescribe `lastOffer`. El entorno de ejecución sólo conserva los datos escritos por el segundo método `postEvent` en los datos de sesión.

Cuando la sesión finaliza, los datos se pierden, a menos que haya creado consideraciones especiales como, por ejemplo, la utilización de un proceso Instantánea en el diagrama de flujo interactivo para escribir los datos en una tabla de base de datos. Si tiene previsto utilizar procesos Instantánea, recuerde que los nombres deben coincidir con las limitaciones de la base de datos. Por ejemplo, si sólo tiene permiso para 256 caracteres en el nombre de una columna, el nombre del par nombre-valor no debe exceder los 256 caracteres.

Acerca de la clase InteractAPI

La clase InteractAPI contiene los métodos que se utilizan para integrar el punto de encuentro con el servidor de ejecución. Las demás clases y métodos de la API de Interact dan soporte a los métodos de esta clase.

Debe compilar su implementación en `interact_client.jar`, que se encuentra en el directorio `lib` de la instalación del entorno de ejecución de Interact.

endSession

```
endSession(String sessionID)
```

El método `endSession` marca el final de la sesión de ejecución. Cuando el servidor de ejecución recibe este método, el servidor de ejecución se registra en el historial, borra la memoria, etc.

- **sessionID** — Cadena exclusiva que identifica la sesión.

Si no se invoca el método endSession, las sesiones de ejecución exceden el tiempo de espera. El período de tiempo de espera puede configurarse con la propiedad `sessionTimeout`.

Valor de retorno

El servidor de ejecución responde al método `endSession` con el objeto `Respose` con los siguientes atributos completados:

- SessionID
- ApiVersion
- StatusCode
- AdvisoryMessages

Ejemplo

El siguiente ejemplo muestra el método `endSession` y cómo puede analizar la respuesta. `sessionID` es la misma cadena que identifica la sesión utilizada por la llamada `startSession` que ha iniciado esta sesión.

```
response = api.endSession(sessionId);
    // comprobar si la respuesta es satisfactoria o no
    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");
    }
    // Para los errores, deben aparecer mensajes de aviso explicando por qué
    if(response.getStatusCode() != Response.STATUS_SUCCESS)
        printDetailMessageOfWarningOrError("endSession",
   response.getAdvisoryMessages());
```

executeBatch

`executeBatch(String *sessionID*, *CommandImpl*[] commands)`

El método `executeBatch` permite ejecutar varios métodos con una sola solicitud en el servidor de ejecución.

- **sessionID** — una cadena que identifica el ID de sesión. Este ID de sesión se utiliza para todos los comandos ejecutados por esta llamada de método.
- **commandImpl[]** — una matriz de objetos CommandImpl, uno para cada comando que desea ejecutar.

El resultado de invocar este método equivale a invocar explícitamente cada método en la matriz de comandos. Este método minimiza el número de solicitudes reales en el servidor de ejecución. El servidor de ejecución ejecuta cada método en serie; para cada llamada, se capturan los errores o avisos en el objeto de respuesta correspondiente a esa llamada de método. Si se encuentra un error, executeBatch continúa con el resto de las llamadas del lote. Si la ejecución de un método da como resultado un error, el estado de nivel superior del objeto BatchResponse refleja ese error. Si no se produce ningún error, el estado de nivel superior refleja todos los avisos que se han producido. Si no se produce ningún aviso, el estado de nivel superior refleja una ejecución satisfactoria del lote.

Valor de retorno

El servidor de ejecución responde a executeBatch con un objeto BatchResponse.

Ejemplo

El siguiente ejemplo muestra cómo llamar a todos los métodos getOffer y postEvent con una sola llamada executeBatch, y una sugerencia sobre cómo manejar la respuesta.

```
/** Definir todas las variables para todos los miembros de executeBatch*/
String sessionId="MySessionID-123";
String interactionPoint = "Overview Page Banner 1";
int numberRequested=1;
String eventName = "logOffer";

/** compilar el comando getOffers */
Command getOffersCommand = new CommandImpl();
getOffersCommand.setMethodIdentifier(Command.COMMAND_GETOFFERS);
getOffersCommand.setInteractionPoint(interactionPoint);
getOffersCommand.setNumberRequested(numberRequested);

/** compilar el comando postEvent */
Command postEventCommand = new CommandImpl();
postEventCommand.setMethodIdentifier(Command.COMMAND_POSTEVENT);
postEventCommand.setEventParameters(postEventParameters);
postEventCommand.setEvent(eventName);

/** Crear la matriz de comandos */
Command[] commands =
{
  getOffersCommand,
  postEventCommand,
};

/** Realizar la llamada */
BatchResponse batchResponse = api.executeBatch(sessionId, commands);

/** Procesar la respuesta según corresponda */
// El código de estado de nivel superior es un atajo para determinar si hay
// errores en la matriz de objetos de respuesta
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");
}

// Iterar por la matriz e imprimir el mensaje de los errores
for(Response response : batchResponse.getResponses())
{
  if(response.getStatusCode()!=Response.STATUS_SUCCESS)
  {
  printDetailMessageOfWarningOrError("executeBatchCommand",
 response.getAdvisoryMessages());
  }
}
```

getInstance

```
getInstance(String URL)
```

El método `getInstance` crea una instancia de la API de Interact que se comunica con el servidor de ejecución especificado.

Importante: Cada aplicación que escriba utilizando la API de Interact debe invocar `getInstance` para crear una instancia de un objeto InteractAPI que se correlaciona con un servidor de ejecución especificado por el parámetro de URL.

Para los grupos de servidores, si está utilizando un equilibrador de carga, utilice el nombre de host y el puerto que ha configurado con el equilibrador de carga. Si no tiene un equilibrador de carga, deberá incluir la lógica para rotar entre los servidores de ejecución disponibles.

Este método sólo es aplicable para la serialización Java con el adaptador HTTP. No hay ningún método correspondiente definido en el WSDL de SOAP. Cada implementación de cliente SOAP tiene su propia forma de establecer el URL de punto final.

- **URL** — Una cadena que identifica el URL de la instancia de tiempo de ejecución. Por ejemplo, `http://localhost:7001/Interact/servlet/InteractJSService`.

Valor de retorno

El servidor de ejecución devuelve la InteractAPI.

Ejemplo

El siguiente ejemplo muestra cómo crear una instancia de un objeto InteractAPI que apunta a una instancia de servidor de ejecución que se ejecuta en la misma máquina que su punto de encuentro.

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

getOffers

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

El método `getOffers` permite solicitar ofertas del servidor de ejecución.

- **sessionID** — una cadena que identifica la sesión actual.
- **interactionPoint** — una cadena que identifica el nombre del punto de interacción al que hace referencia este método.

  Nota: Este nombre debe coincidir exactamente con el nombre del punto de interacción definido en el canal interactivo.
- **numberOfOffers** — un entero que identifica el número de ofertas solicitadas.

El método `getOffers` espera el número de milisegundos definido en la propiedad `segmentationMaxWaitTimeInMS` a que finalice toda la resegmentación antes de ejecutarse. Por lo tanto, si invoca un método `postEvent` que desencadena una resegmentación o un método `setAudience` inmediatamente antes de una llamada `getOffers`, puede producirse un retardo.

Valor de retorno

El servidor de ejecución responde a `getOffers` con un objeto de respuesta con los siguientes atributos completados:

- AdvisoryMessages
- ApiVersion
- OfferList
- SessionID
- StatusCode

Ejemplo

Este ejemplo muestra la solicitud una oferta individual para el punto de interacción Overview Page Banner 1 y una forma de manejar la respuesta.

`sessionID` es la misma cadena que identifica la sesión de ejecución utilizada por la llamada `startSession` que ha iniciado esta sesión.

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

/** Realizar la llamada */
response = api.getOffers(sessionId, interactionPoint, numberRequested);

/** Procesar la respuesta según corresponda */
// comprobar si la respuesta es satisfactoria o no
 if(response.getStatusCode() == Response.STATUS_SUCCESS)
 {
  System.out.println("getOffers call processed with no warnings or errors");

        /** Comprobar si hay alguna oferta */
        OfferList offerList=response.getOfferList();

        if(offerList.getRecommendedOffers() != null)
        {
            for(Offer offer : offerList.getRecommendedOffers())
            {
                // imprimir oferta
                System.out.println("Nombre de oferta:"+offer.getOfferName());
            }
        }
        else // recuento en la cadena de oferta predeterminada
        System.out.println("Oferta predeterminada:"+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");
    }
    // Para los errores, deben aparecer mensajes de aviso explicando la causa
    if(response.getStatusCode() != Response.STATUS_SUCCESS)
        printDetailMessageOfWarningOrError("getOffers",
   response.getAdvisoryMessages());
```

getOffersForMultipleInteractionPoints

`getOffersForMultipleInteractionPoints(String sessionID, String requestStr)`

El método `getOffersForMultipleInteractionPoints` permite solicitar ofertas del servidor de ejecución para varios IP con desduplicación.

- **sessionID** — una cadena que identifica la sesión actual.
- **requestStr** — una cadena que proporciona una matriz de objetos `GetOfferRequest`.

  Cada objeto `GetOfferRequest` especifica:

  – **ipName** — El nombre del punto de interacción (IP) para el que el objeto solicita ofertas

  – **numberRequested** — El número de ofertas exclusivas necesarias para el IP especificado

  – **offerAttributes** — Requisitos de los atributos de las ofertas entregadas utilizando una instancia de `OfferAttributeRequirements`

  – **duplicationPolicy** — ID de política de duplicación de las ofertas que se entregarán

    Las políticas de duplicación determinan si se devolverán ofertas duplicadas en los distintos puntos de interacción en una única llamada de método. (*Dentro* de un punto de interacción individual, nunca se devuelven ofertas duplicadas). Actualmente, se da soporte a dos políticas de duplicación.

    - NO_DUPLICATION (Valor de ID = 1). Ninguna de las ofertas que se han incluido en las instancias de `GetOfferRequest` anteriores se incluirán en esta instancia de `GetOfferRequest` (es decir, Interact aplicará la desduplicación).
    - ALLOW_DUPLICATION (Valor de ID = 2). Se incluirán todas las ofertas que cumplan los requisitos especificados en esta instancia de `GetOfferRequest`. Las ofertas que se han incluido en las instancias de `GetOfferRequest` anteriores no se reconciliarán.

  El orden de las solicitudes en el parámetro de matriz es también el orden de prioridad cuando se entregan las ofertas.

  Por ejemplo, supongamos que los IP en la solicitud son IP1 e IP2, que no se permiten ofertas duplicadas (un ID de política de duplicación = 1) y que cada uno solicita cada dos ofertas. Si Interact encuentra las ofertas A, B y C para IP1 y las ofertas A y D para IP2, la respuesta contendrá las ofertas A y B para IP1, y sólo la oferta D para IP2.

  Asimismo, tenga en cuenta que cuando el ID de política de duplicación es 1, las ofertas que se hayan entregado con un IP con una prioridad mayor no se entregará con este IP.

El método `getOffersForMultipleInteractionPoints` espera el número de milisegundos definido en la propiedad `segmentationMaxWaitTimeInMS` a que finalice toda la resegmentación antes de ejecutarse. Por lo tanto, si invoca un método `postEvent` que desencadena una resegmentación o un método `setAudience` inmediatamente antes de una llamada `getOffers`, puede producirse un retardo.

Valor de retorno

El servidor de ejecución responde a `getOffersForMultipleInteractionPoints` con un objeto de respuesta con los siguientes atributos completados:

- AdvisoryMessages
- ApiVersion
- matriz de OfferList
- SessionID
- StatusCode

Ejemplo

```
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) {
  // Comprobar si hay alguna oferta
  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());
 }
```

Tenga en cuenta que la sintaxis de `requestStr` es la siguiente:

```
requests_for_IP[<requests_for_IP]
```

donde

```
<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
```

En el ejemplo anterior, `requestForIP1 ({IP1,5,1,(5,attr1=1|numeric; attr2=value2|string, (3,attr3=value3|string)(3,attr4=4|numeric))})` significa que, para el punto de interacción denominado IP1, se entregan como máximo 5 ofertas diferentes que no pueden devolverse para otros puntos de interacción durante la misma llamada de método. Las 5 ofertas deben tener un atributo numérico denominado `attr1` con el valor 1, y deben tener un atributo de cadena denominado `attr2` con el valor *value2*. Aparte de esas 5 ofertas, un máximo de 3 deben tener un atributo de cadena denominado `attr3` con el valor *value3*, y un máximo de 3 deben tener un atributo numérico denominado `attr4` con el valor *4*.

Los tipos de atributo permitidos son numéricos, cadena, y fecha y hora; el formato de un valor de atributo de fecha y hora debe ser `MM/dd/yyyy HH:mm:ss`. Para recuperar las ofertas devueltas, utilice el método `Response.getAllOfferLists()`. Para entender mejor la sintaxis, el ejemplo de `setGetOfferRequests` crea la misma instancia de `GetOfferRequests` utilizando objetos Java, que es la opción preferida.

getProfile

```
getProfile(String sessionID)
```

El método getProfile permite recuperar la información de perfil y temporal sobre el visitante que visita el punto de encuentro.

- **sessionID** — una cadena que identifica el ID de sesión.

Valor de retorno

El servidor de ejecución responde a getProfile con un objeto de respuesta con los siguientes atributos completados:

- AdvisoryMessages
- ApiVersion
- ProfileRecord
- SessionID
- StatusCode

Ejemplo

El siguiente es un ejemplo de utilización de getProfile y una forma de manejar la respuesta.

sessionID es la misma cadena que identifica la sesión utilizada por la llamada startSession que ha iniciado esta sesión.

```
response = api.getProfile(sessionId);
/** Procesar la respuesta según corresponda */
    // comprobar si la respuesta es satisfactoria o no
    if(response.getStatusCode() == Response.STATUS_SUCCESS)
    {
        System.out.println("getProfile call processed with no warnings or errors");
        // Imprimir el perfil - es sólo una matriz de objetos NameValuePair
        for(NameValuePair nvp : response.getProfileRecord())
        {
            System.out.println("Nombre:"+nvp.getName());
            if(nvp.getValueDataType().equals(NameValuePair.DATA_TYPE_DATETIME))
            {
                System.out.println("Valor:"+nvp.getValueAsDate());
            }
            else if(nvp.getValueDataType().equals(NameValuePair.DATA_TYPE_NUMERIC))
            {
                System.out.println("Valor:"+nvp.getValueAsNumeric());
            }
            else
            {
                System.out.println("Valor:"+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");
    }
    // Para los errores, deben aparecer mensajes de aviso explicando por qué
    if(response.getStatusCode() != Response.STATUS_SUCCESS)
        printDetailMessageOfWarningOrError("getProfile",
   response.getAdvisoryMessages());
```

getVersion

```
getVersion()
```

El método `getVersion` devuelve la versión de la implementación actual del servidor de ejecución de Interact.

Se recomienda utilizar este método cuando inicialice el punto de encuentro con la API de Interact.

Valor de retorno

El servidor de ejecución responde a `getVersion` con un objeto de respuesta con los siguientes atributos completados:

- AdvisoryMessages
- ApiVersion
- StatusCode

Ejemplo

Este ejemplo muestra una forma simple de invocar `getVersion` y procesar los resultados.

```
response = api.getVersion();
/** Procesar la respuesta según corresponda */
    // comprobar si la respuesta es satisfactoria o no
    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");
    }

    // Para los errores, deben aparecer mensajes de aviso explicando por qué
    if(response.getStatusCode() != Response.STATUS_SUCCESS)
        printDetailMessageOfWarningOrError("getVersion",
   response.getAdvisoryMessages());
```

postEvent

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

El método `postEvent` permite ejecutar cualquier evento definido en el canal interactivo.

- **sessionID** — una cadena que identifica el ID de sesión.
- **eventName** — una cadena que identifica el nombre del evento.

  Nota: El nombre del evento debe coincidir con el nombre del evento tal como está definido en el canal interactivo. Este nombre es sensible a mayúsculas y minúsculas.
- **eventParameters** — Objetos NameValuePairImpl que identifican los parámetros que deben pasarse con el evento. Estos valores se almacenan en los datos de sesión.

  Si este evento desencadena la resegmentación, debe asegurarse de que todos los datos necesarios para los diagramas de flujo interactivos estén disponibles en los datos de sesión. Si alguno de estos valores no se ha completado mediante las

acciones anteriores (por ejemplo, a partir de `startSession` o `setAudience`, o al cargar la tabla de perfil), debe incluir un eventParameter para cada valor que falte. Por ejemplo, si ha configurado todas las tablas de perfil para que se carguen en la memoria, debe incluir un NameValuePair para los datos temporales necesarios para los diagramas de flujo interactivos.

Si utiliza más de un nivel de audiencia, lo más probable es que tenga distintos conjuntos de eventParameters para cada nivel de audiencia. Debe incluir alguna lógica para asegurarse de que está seleccionando el conjunto de parámetros correcto para el nivel de audiencia.

Importante: Si este evento se registra en un historial de respuestas, debe pasar el código de tratamiento para la oferta. Debe definir el nombre de NameValuePair como "`UACIOfferTrackingCode`".

Sólo puede pasar un código de tratamiento por evento. Si no pasa el código de tratamiento para un contacto de oferta, Interact registra un contacto de oferta para cada oferta en la última lista de ofertas recomendada. Si no pasa el código de tratamiento para una respuesta, Interact devuelve un error.

- Existen otros parámetros reservados que se utilizan con `postEvent` y otros métodos, que se describen más adelante en esta sección.

Toda solicitud de resegmentación o escritura en el historial de contactos o de respuestas no espera una respuesta.

A menos que se especifique con el parámetro `UACIExecuteFlowchartByName`, la resegmentación ejecuta todos los diagramas de flujo interactivos asociados con este canal interactivo para el nivel de audiencia actual. El método `getOffers` espera a que la resegmentación finalice antes de ejecutarse. Por lo tanto, si invoca un método `postEvent` que desencadena una resegmentación inmediatamente antes de una llamada `getOffers`, puede producirse un retardo.

Valor de retorno

El servidor de ejecución responde a `postEvent` con un objeto de respuesta con los siguientes atributos completados:

- AdvisoryMessages
- ApiVersion
- SessionID
- StatusCode

Ejemplo

El siguiente ejemplo de `postEvent` muestra el envío de nuevos parámetros para un evento que desencadena la resegmentación, y una forma de manejar la respuesta.

`sessionID` es la misma cadena que identifica la sesión utilizada por la llamada `startSession` que ha iniciado esta sesión.

```
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
};

/** Realizar la llamada */
response = api.postEvent(sessionId, eventName, postEventParameters);

/** Procesar la respuesta según corresponda */
    // comprobar si la respuesta es satisfactoria o no
    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");
    }

    // Para los errores, deben aparecer mensajes de aviso explicando por qué
    if(response.getStatusCode() != Response.STATUS_SUCCESS)
        printDetailMessageOfWarningOrError("postEvent",
   response.getAdvisoryMessages());
```

setAudience

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

El método `setAudience` permite establecer el ID y el nivel de audiencia para un visitante.

- **sessionID**: cadena que identifica el ID de sesión.
- **audienceID**: matriz de objetos `NameValuePairImpl` que define el ID de audiencia.
- **audienceLevel**: cadena que define el nivel de audiencia.

- **parameters**: objetos NameValuePairImpl que identifican los parámetros que se deben pasar con setAudience. Estos valores se almacenan en los datos de sesión y pueden utilizarse para la segmentación.

  Debe tener un valor para cada columna de su perfil. Este es un superconjunto de todas las columnas de todas las tablas definidas para el canal interactivo y los datos en tiempo real. Si ya ha completado todos los datos de sesión con startSession o postEvent, no es necesario enviar nuevos parámetros.

El método setAudience desencadena una resegmentación. El método getOffers espera a que la resegmentación finalice antes de ejecutarse. Por lo tanto, si invoca un método setAudience inmediatamente antes de una llamada getOffers, puede producirse un retardo.

El método setAudience también carga los datos del perfil para el ID de audiencia. Puede utilizar el método setAudience para forzar que se vuelvan a cargar los mismos datos del perfil cargados por el método startSession.

Valor de retorno

El servidor de ejecución responde a setAudience con un objeto de respuesta con los siguientes atributos completados:

- AdvisoryMessages
- ApiVersion
- SessionID
- StatusCode

Ejemplo

En este ejemplo, el nivel de audiencia permanece igual, pero el ID cambia, como si un usuario anónimo inicia una sesión y pasa a ser conocido.

sessionId y audienceLevel son las mismas cadenas que identifican la sesión y el nivel de audiencia utilizados por la llamada startSession que ha iniciado esta sesión.

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

NameValuePair[] newAudienceId = { custId2 };

/** Los parámetros también pueden pasarse. Para este ejemplo, no hay parámetros y se pasa null */
NameValuePair[] noParameters=null;

/** Realizar la llamada */
response = api.setAudience(sessionId, newAudienceId, audienceLevel, noParameters);

 /** Procesar la respuesta según corresponda */
    // comprobar si la respuesta es satisfactoria o no
    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");
    }
```

```
    // Para los errores, deben aparecer mensajes de aviso explicando por qué
    if(response.getStatusCode() != Response.STATUS_SUCCESS)
        printDetailMessageOfWarningOrError("setAudience",
   response.getAdvisoryMessages());
```

setDebug

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

El método setDebug le permite establecer el nivel de detalle de registro para todas las rutas de código de la sesión.

- **ID_sesión**: serie que identifica el ID de sesión.
- **debug**: booleano que habilita o inhabilita la información de depuración. Los valores válidos son true o false. Si es true, Interact registra información de depuración en el registro del servidor de ejecución.

Valor de retorno

El servidor de ejecución responde a setDebug con un objeto Response con los siguientes atributos completados:

- AdvisoryMessages
- ApiVersion
- SessionID
- StatusCode

Ejemplo

El ejemplo siguiente muestra el nivel de depuración de la sesión.

ID_sesión es la misma serie para identificar la sesión utilizada por la llamada a startSession que ha iniciado esta sesión.

```
boolean newDebugFlag=false;
/** realizar la llamada */
response = api.setDebug(sessionId, newDebugFlag);

/** Procesar la respuesta de la forma adecuada */
    // comprobar si la respuesta ha sido satisfactoria o no
    if(response.getStatusCode() == Response.STATUS_SUCCESS)
    {
        System.out.println("llamada a setDebug procesada sin avisos o errores");
    }
    else if(response.getStatusCode() == Response.STATUS_WARNING)
    {
        System.out.println("llamada a setDebug procesada con un aviso");
    }
    else
    {
        System.out.println("llamada a setDebug procesada con un error");
    }

    // Para resultados no satisfactorios, debería haber mensajes de
    // recomendación que explicaran porqué
    if(response.getStatusCode() != Response.STATUS_SUCCESS)
        printDetailMessageOfWarningOrError("setDebug",
   response.getAdvisoryMessages());
```

startSession

```
startSession(String ID_sesión,
   boolean relyOnExistingSession,
   boolean debug,
```

```
String canalInteractivo,
NameValuePairImpl[] IDAudiencia,
String nivelAudiencia,
NameValuePairImpl[] parameters)
```

El método `startSession` crea y define una sesión de ejecución. `startSession` puede desencadenar hasta cinco acciones:

- crear una sesión de ejecución.
- cargar los datos de perfil del visitante para el nivel de audiencia actual en la sesión de ejecución, incluidas las tablas de dimensiones marcadas para la carga en la correlación de tablas definida para el canal interactivo.
- activar segmentación, ejecutando todos los diagramas de flujo interactivos para el nivel de audiencia actual.
- cargar datos de supresión de oferta en la sesión, si la propiedad `enableOfferSuppressionLookup` está establecida en true.
- cargar los datos de sustitución de puntuación en la sesión, si la propiedad `enableScoreOverrideLookup` está establecida en true.

El método `startSession` requiere los parámetros siguientes:

- **ID_sesión**: serie que identifica el ID de sesión. Debe definir el ID de sesión. Por ejemplo, podría utilizar una combinación de ID de cliente e indicación de fecha y hora.

  Para definir qué compone una sesión de ejecución, se debe especificar el ID de sesión. Este valor lo gestiona el cliente. El cliente debe sincronizar todas las llamadas a método para el mismo ID de sesión. El comportamiento para las llamadas a API simultáneas con el mismo ID de sesión no está definido.
- **relyOnExistingSession**: booleano que define si esta sesión utiliza una sesión nueva o existente. Los valores válidos son `true` o `false`. Si es `true`, debe proporcionar un ID de sesión existente con el método `startSession`. Si es `false`, debe proporcionar un nuevo ID de sesión.

  Si se establece `relyOnExistingSession` en `true` y ya existe una sesión, el entorno de ejecución utiliza los datos de sesión existentes y no vuelve a cargar datos o desencadena segmentación. Si la sesión no existe, el entorno de ejecución crea una nueva sesión, lo que incluye cargar datos y desencadenar segmentación. Establecer `relyOnExistingSession` en true y utilizarlo con todas las llamadas a `startSession` resulta útil si el punto de encuentro tiene una duración de sesión superior a la sesión de ejecución. Por ejemplo, una sesión de sitio web está activa durante 2 horas, pero la sesión de ejecución solo está activa 20 minutos.

  Si llama a `startSession` dos veces con el mismo ID de sesión, todos los datos de sesión de la primera llamada a `startSession` se perderán si `relyOnExistingSession` es false.
- **debug**: booleano que habilita o inhabilita la información de depuración. Los valores válidos son `true` o `false`. Si es true, Interact registra la información de depuración en los registros del servidor de ejecución. El indicador de depuración se establece individualmente para cada sesión. Por lo tanto, puede rastrear los datos de depuración para una sesión individual.
- **canalInteractivo**: serie que define el nombre del canal interactivo al que hace referencia esta sesión. Este nombre debe coincidir con el nombre del canal interactivo tal como está definido exactamente en Campaign.
- **IDAudiencia**: matriz de objetos NameValuePairImpl donde los nombres deben coincidir con los nombres de columna física de cualquier tabla que contenga el ID de audiencia.
- **nivelAudiencia**: serie que define el nivel de audiencia.

- **parámetros**: objetos NameValuePairImpl que identifican los parámetros que es necesario pasar con `startSession`. Estos valores se almacenan en los datos de sesión y se pueden utilizar para la segmentación.

  Si tiene varios diagramas de flujo interactivos para el mismo nivel de audiencia, debe incluir un superconjunto de todas las columnas de todas las tablas. Si configura el tiempo de ejecución para cargar la tabla de perfil, y la tabla de perfil contiene todas las columnas que necesita, no es necesario pasar ningún parámetro, a menos que desee sobrescribir los datos de la tabla de perfil. Si la tabla de perfil contiene un subconjunto de las columnas necesarias, debe incluir las columnas que faltan como parámetros.

Si `IDAudiencia` o `nivelAudiencia` no son válidos y `relyOnExistingSession` es false, la llamada a `startSession` fallará. Si `interactiveChannel` no es válido, `startSession` fallará, independientemente de si `relyOnExistingSession` es true o false.

Si `relyOnExistingSession` es true y realiza una segunda llamada a `startSession` utilizando el mismo `ID_sesión`, pero la primera sesión ha caducado, Interact creará una nueva sesión.

Si `relyOnExistingSession` es true y realiza una segunda llamada a `startSession` utilizando el mismo `ID_sesión` pero un `IDAudiencia` o `nivelAudiencia` distinto, el servidor de ejecución cambiará la audiencia para la sesión existente.

Si `relyOnExistingSession` es true y realiza una segunda llamada a `startSession` utilizando el mismo `ID_sesión` pero un `canalInteractivo` distinto, el servidor de ejecución creará una nueva sesión.

Valor de retorno

El servidor de ejecución responde a `startSession` con un objeto Response que tiene los siguientes atributos completados:

- AdvisoryMessages (si StatusCode no es igual a 0)
- ApiVersion
- SessionID
- StatusCode

Ejemplo

El ejemplo siguiente muestra una forma de llamar a `startSession`.

```
String sessionId="MySessionID-123";
String audienceLevel="Cliente";
NameValuePair custId = new NameValuePairImpl();
custId.setName("IDCliente");
custId.setValueAsNumeric(1.0);
custId.setValueDataType(NameValuePair.DATA_TYPE_NUMERIC);
NameValuePair[] initialAudienceId = { custId };
boolean relyOnExistingSession=false;
boolean initialDebugFlag=true;
String interactiveChannel="Sitio web Cuentas";
NameValuePair parm1 = new NameValuePairImpl();
parm1.setName("SerieBúsqueda");
parm1.setValueAsString("");
parm1.setValueDataType(NameValuePair.DATA_TYPE_STRING);

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

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

NameValuePair parm3 = new NameValuePairImpl();
parm3.setName("Navegador");
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("TemaPágina");
parm6.setValueAsString("");
parm6.setValueDataType(NameValuePair.DATA_TYPE_STRING);

/** Especificación de parámetros (opcional) */
NameValuePair[] initialParameters = { parm1,
 parm2,
 parm3,
 parm4,
 parm5,
 parm6
 };

/** Realizar la llamada */
response = api.startSession(sessionId, relyOnExistingSession, initialDebugFlag,
 interactiveChannel, initialAudienceId, audienceLevel, initialParameters);

/** Procesar la respuesta de la forma adecuada */
processStartSessionResponse(response);
```

processStartSessionResponse es un método que maneja el objeto de respuesta devuelto por startSession.

```
public static void processStartSessionResponse(Response response)
{
  // comprobar si la respuesta ha sido satisfactoria o no
  if(response.getStatusCode() == Response.STATUS_SUCCESS)
 {
  System.out.println("llamada a startSession procesada sin avisos o errores");
 }
  else if(response.getStatusCode() == Response.STATUS_WARNING)
 {
  System.out.println("llamada a startSession procesada con un aviso");
 }
  else
 {
  System.out.println("llamada a startSession procesada con un error");
 }

 // Para resultados no satisfactorios, debería haber mensajes de
    // recomendación que explicaran porqué
 if(response.getStatusCode() != Response.STATUS_SUCCESS)
  printDetailMessageOfWarningOrError("StartSession",
   response.getAdvisoryMessages());
}
```

Parámetros reservados

Hay algunos parámetros reservados que se utilizan con la API de Interact. Algunos son necesarios para el servidor de ejecución y otros se pueden utilizar para características adicionales.

Características de postEvent

Característica	Parámetro	Descripción
Registrar en tabla personalizada	`UACICustomLoggerTableName`	Nombre de una tabla del origen de datos de tablas de ejecución. Si proporciona este parámetro con un nombre de tabla válido, el entorno de ejecución graba todos los datos de sesión en la tabla seleccionada. Todos los nombres de columna de la tabla que coinciden con los datos de sesión NameValuePair se completan. El entorno de ejecución completa con un nulo las columnas que no coincidan con el par nombre-valor de sesión. Puede gestionar el proceso que graba en la base de datos con las propiedades de configuración `customLogger`.
Varios tipos de respuesta	`UACILogToLearning`	Un entero con un valor 1 ó 0. 1 indica que el entorno de ejecución debe registrar el evento como una aceptación para aprendizaje. 0 indica que el entorno de ejecución no debe registrar el evento para aprendizaje. Este parámetro le permite crear varios métodos `postEvent` registrando distintos tipos de respuesta sin influenciar el aprendizaje. No necesita definir este parámetro para eventos establecidos para registrar un contacto, una aceptación o un rechazo. Debe utilizar este parámetro junto con `UACIResponseTypeCode`. Si no define `UACILOGTOLEARNING`, el entorno de ejecución asume el valor predeterminado de 0 (a menos que el evento desencadene un contacto, aceptación o rechazo de registro).
	`UACIResponseTypeCode`	Valor que representa un código de tipo de respuesta. El valor debe ser una entrada válida en la tabla `UA_UsrResponseType`

Característica	Parámetro	Descripción
Seguim. respuestas	UACIOfferTrackingCode	Código de tratamiento de la oferta. Debe definir este parámetro si el evento se registra en el historial de respuestas o contactos. Sólo puede pasar un código de tratamiento por evento. Si no pasa el código de tratamiento para un contacto de oferta, el entorno de ejecución registra un contacto de oferta para cada oferta en la última lista recomendada de ofertas. Si no pasa el código de tratamiento para una respuesta, el entorno de ejecución devuelve un error. Si configura el seguimiento de respuestas de sesiones cruzadas, puede utilizar el parámetro UACIOfferTrackingcodeType para definir qué tipo de código de seguimiento utiliza que no es el código de tratamiento.
Seguimiento de respuestas de sesiones cruzadas	UACIOfferTrackingCodeType	Número que define el tipo de código de seguimiento. 1 es el código de tratamiento predeterminado y 2 es el código de oferta. Todos los códigos deben ser entradas válidas de la tabla UACI_TrackingType. Puede añadir otros códigos personalizadas a esta tabla.
Ejecución de diagrama de flujo específica	UACIExecuteFlowchartByName	Si define este parámetro para cualquier método que desencadene segmentación (startSession, setAudience o postEvent que desencadena re-segmentación), en lugar de ejecutar todos los diagramas de flujo para el nivel de audiencia actual, Interact ejecuta solo los diagramas de flujo con nombre. Puede proporcionar una lista de diagramas de flujo separados por un carácter de barra vertical (|).

Parámetros reservados del entorno de ejecución

El entorno de ejecución utiliza los siguientes parámetros reservados. No utilice estos parámetros para sus parámetros de evento.

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

Acerca de la clase AdvisoryMessage

La clase advisoryMessage contiene métodos que definen el objeto de mensaje de aviso. El objeto de mensaje de aviso está contenido en el objeto de respuesta. Cada método de la InteractAPI devuelve un objeto de respuesta.(Excepto el método executeBatch, que devuelve un objeto batchResponse.) Si se produce un error o un

aviso, el servidor de Interact completa el objeto de mensaje de aviso. El objeto de mensaje de aviso contiene los siguientes atributos:

- **DetailMessage** — una descripción detallada del mensaje de aviso. Este atributo no estará disponibles para todos los mensajes de aviso. Si está disponible, DetailMessage no puede localizarse.
- **Message** — una descripción breve del mensaje de aviso.
- **MessageCode** — un número de código para el mensaje de aviso.
- **StatusLevel** — un número de código para la gravedad del mensaje de aviso.

Los objetos advisoryMessage se recuperan utilizando el método `getAdvisoryMessages`.

getDetailMessage

```
getDetailMessage()
```

El método `getDetailMessage` devuelve la descripción detallada de un objeto de mensaje de aviso.

No todos los mensajes tienen un mensaje detallado.

Valor de retorno

El objeto de mensaje de aviso devuelve una cadena.

Ejemplo

```
// Para los errores, deben aparecer mensajes de aviso explicando por qué
if(response.getStatusCode() != Response.STATUS_SUCCESS)
{
 for(AdvisoryMessage msg : response.getAdvisoryMessages())
 {
  System.out.println(msg.getMessage());
// Algunos mensajes de aviso pueden tener detalles adicionales:
  System.out.println(msg.getDetailMessage());
 }
}
```

getMessage

```
getMessage()
```

El método `getMessage` devuelve una descripción breve de un objeto de mensaje de aviso.

Valor de retorno

El objeto de mensaje de aviso devuelve una cadena.

Ejemplo

El siguiente método imprime el mensaje y el mensaje detallado de un objeto AdvisoryMessage.

```
// Para los errores, deben aparecer mensajes de aviso explicando por qué
if(response.getStatusCode() != Response.STATUS_SUCCESS)
{
 for(AdvisoryMessage msg : response.getAdvisoryMessages())
 {
  System.out.println(msg.getMessage());
```

```
// Algunos mensajes de aviso pueden tener detalles adicionales:
  System.out.println(msg.getDetailMessage());
 }
}
```

getMessageCode

```
getMessageCode()
```

El método `getMessageCode` devuelve el código de error interno asociado con un objeto de mensaje de aviso si el nivel de estado es 2 (STATUS_LEVEL_ERROR).

Valor de retorno

El objeto AdvisoryMessage devuelve un entero.

Ejemplo

El siguiente método imprime el código de mensaje de un objeto AdvisoryMessage.

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

getStatusLevel

```
getStatusLevel()
```

El método `getStatusLevel` devuelve el nivel de estado de un objeto de mensaje de aviso.

Valor de retorno

El objeto de mensaje de aviso devuelve un entero.

- 0 - STATUS_LEVEL_SUCCESS — El método invocado ha finalizado sin errores.
- 1 - STATUS_LEVEL_WARNING — El método invocado ha finalizado con al menos un aviso (pero sin errores).
- 2 - STATUS_LEVEL_ERROR — El método invocado no ha finalizado satisfactoriamente y tiene al menos un error.

Ejemplo

El siguiente método imprime el nivel de estado de un objeto AdvisoryMessage.

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

Acerca de la clase AdvisoryMessageCode

La clase `advisoryMessageCode` contiene métodos que definen los códigos de mensaje de aviso. Puede recuperar los códigos de mensajes de aviso con el método `getMessageCode`.

Códigos de mensaje de aviso

Código	Descripción
1	INVALID_SESSION_ID - el ID de sesión no hace referencia a una sesión válida
2	ERROR_TRYING_TO_ABORT_SEGMENTATION - se ha encontrado un error cuando se intentaba abortar la segmentación durante una `endSession`
3	INVALID_INTERACTIVE_CHANNEL - el argumento pasado para el canal interactivo no hace referencia a un canal interactivo válido
4	INVALID_EVENT_NAME - el argumento pasado para el evento no hace referencia a un evento válido para el canal interactivo actual
5	INVALID_INTERACTION_POINT - el argumento pasado para el punto de interacción no hace referencia a un punto de interacción válido para el canal interactivo actual
6	ERROR_WHILE_MAKING_INITIAL_SEGMENTATION_REQUEST - se ha encontrado un error al enviar una solicitud de segmentación
7	SEGMENTATION_RUN_FAILED - la segmentación ejecutado parcialmente, pero ha dado como resultado un error
8	PROFILE_LOAD_FAILED - el intento de cargar las tablas de dimensiones o perfiles ha fallado
9	OFFER_SUPPRESSION_LOAD_FAILED - el intento de cargar la tabla de supresión de ofertas ha fallado
10	COMMAND_METHOD_UNRECOGNIZED - el método de comando especificado para un comando en un `executeBatch` no es válido
11	ERROR_TRYING_TO_POST_EVENT_PARAMETERS - se ha producido un error al publicar los parámetros de evento
12	LOG_SYSTEM_EVENT_EXCEPTION - se ha producido una excepción al intentar enviar el evento de sistema (Finalizar sesión, Obtener oferta, Obtener perfil, Establecer audiencia, Establecer depuración o Iniciar sesión) para el registro
13	LOG_USER_EVENT_EXCEPTION - se ha producido una excepción al intentar enviar el evento de registro para el registro
14	ERROR_TRYING_TO_LOOK_UP_EVENT - se ha producido un error al intentar buscar el nombre de evento
15	ERROR_TRYING_TO_LOOK_UP_INTERACTIVE_CHANNEL - se ha producido un error al intentar buscar el nombre de canal interactivo
16	ERROR_TRYING_TO_LOOK_UP_INTERACTION_POINT - se ha producido un error al intentar consultar el nombre del punto de interacción
17	RUNTIME_EXCEPTION_ENCOUNTERED - se ha encontrado excepción de tiempo de ejecución inesperada
18	ERROR_TRYING_TO_EXECUTE_ASSOCIATED_ACTION - error al intentar ejecutar la acción asociada (Desencadenar resegmentación, Registrar contacto de oferta, Registrar aceptación de oferta, Registrar rechazo de oferta).
19	ERROR_TRYING_RUN_FLOWCHART - error al intentar ejecutar el diagrama de flujo
20	FLOWCHART_FAILED - el diagrama de flujo ha fallado
21	FLOWCHART_ABORTED - el diagrama de flujo ha abortado
22	FLOWCHART_NEVER_RUN - el diagrama de flujo nunca se ha ejecutado
23	FLOWCHART_STILL_RUNNING - el diagrama de flujo todavía se está ejecutando

Código	Descripción
24	ERROR_WHILE_READING_PARAMETERS - error al leer los parámetros
25	ERROR_WHILE_LOADING_RECOMMENDED_OFFERS - error al cargar las ofertas recomendadas
26	ERROR_WHILE_LOGGING_DEFAULT_TEXT_STATISTICS - error al registrar las estadísticas de texto predeterminadas (el número de veces que aparece la cadena predeterminada para el punto de interacción)
27	SCORE_OVERRIDE_LOAD_FAILED - la carga de la tabla de alteración temporal de puntuaciones ha fallado
28	NULL_AUDIENCE_ID - el ID de audiencia está vacío
29	UNRECOGNIZED_AUDIENCE_LEVEL - nivel de audiencia no reconocido
30	MISSING_AUDIENCE_FIELD - falta el campo de audiencia
31	INVALID_AUDIENCE_FIELD_TYPE - tipo de campo de audiencia no válido
32	UNSUPPORTED_AUDIENCE_FIELD_TYPE - tipo de campo de audiencia no soportado
33	TIMEOUT_REACHED_ON_GET_OFFERS_CALL - la llamada getOffers ha alcanzado el tiempo de espera sin devolver ofertas
34	INTERACT_INITIALIZATION_NOT_COMPLETED_SUCCESSFULLY - la inicialización de tiempo de ejecución no se ha completado satisfactoriamente
35	SESSION_ID_UNDEFINED - ID de sesión sin definir
36	INVALID_NUMBER_OF_OFFERS_REQUESTED - el número de ofertas solicitado no es válido
37	NO_SESSION_EXIST_BUT_WILL_CREATE_NEW_ONE
38	AUDIENCE_ID_NOT_FOUND_IN_PROFILE_TABLE
39	LOG_CUSTOM_LOGGER_EVENT_EXCEPTION - se ha producido una excepción al intentar enviar el evento de datos de registro personalizado
40	SPECIFIED_FLOWCHART_FOR_EXECUTION_DOES_NOT_EXIST
41	AUDIENCE_NOT_DEFINED_IN_CONFIGURATION

Acerca de la clase BatchResponse

La clase `BatchResponse` contiene métodos que definen los resultados del método `executeBatch`. El objeto de respuesta por lotes contiene los siguientes atributos:

- **BatchStatusCode** — El valor máximo de Código de estado para todas las respuestas solicitadas por el método `executeBatch`.
- **Responses** — Una matriz de los objetos de respuesta solicitados por el método `executeBatch`.

getBatchStatusCode

```
getBatchStatusCode()
```

El método `getBatchStatusCode` devuelve el código de estado máximo de la matriz de comandos ejecutados por el método `executeBatch`.

Valor de retorno

El método `getBatchStatusCode` devuelve un entero.

- 0 - STATUS_SUCCESS — El método invocado ha finalizado sin errores.

- 1 - STATUS_WARNING — El método invocado ha finalizado con al menos un aviso (pero sin errores).
- 2 - STATUS_ERROR — El método invocado no ha finalizado satisfactoriamente y tiene al menos un error.

Ejemplo

El siguiente código de ejemplo proporciona un ejemplo de cómo recuperar el `BatchStatusCode`.

```
// El código de estado de nivel superior es un atajo para determinar si hay
// errores en la matriz de objetos de respuesta
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");
}

// Iterar por la matriz e imprimir el mensaje de los errores
for(Response response : batchResponse.getResponses())
{
  if(response.getStatusCode()!=Response.STATUS_SUCCESS)
  {
      printDetailMessageOfWarningOrError("executeBatchCommand",
   response.getAdvisoryMessages());
  }
}
```

getResponses

```
getResponses()
```

El método `getResponses` devuelve la matriz de objetos de respuesta correspondiente a la matriz de comandos ejecutados por el método `executeBatch`.

Valor de retorno

El método `getResponses` devuelve una matriz de objetos `Response`.

Ejemplo

El siguiente ejemplo selecciona todas las respuestas e imprime los mensajes de aviso si el comando no ha sido satisfactorio.

```
for(Response response : batchResponse.getResponses())
{
    if(response.getStatusCode()!=Response.STATUS_SUCCESS)
    {
        printDetailMessageOfWarningOrError("executeBatchCommand",
   response.getAdvisoryMessages());
    }
}
```

Acerca de la interfaz Comando

El método executeBatch requiere que pase una matriz de objetos que implemente la interfaz Comando. Debe utilizar la implementación predeterminada, CommandImpl para pasar objetos Comando.

La tabla siguiente lista el comando, el método de la clase InteractAPI que representa el comando, y los métodos de la interfaz Comando que se deben utilizar para cada comando. No es necesario incluir un ID de sesión porque el método executeBatch ya incluye el ID de sesión.

Comando	Método de la API de Interact	Métodos de la interfaz Comando
COMMAND_ENDSESSION	endSession	Ninguno.
COMMAND_GETOFFERS	getOffers	• setInteractionPoint • setNumberRequested
COMMAND_GETPROFILE	getProfile	Ninguno.
COMMAND_GETVERSION	getVersion	Ninguno.
COMMAND_POSTEVENT	postEvent	• setEvent • setEventParameters
COMMAND_SETAUDIENCE	setAudience	• setAudienceID • setAudienceLevel • setEventParameters
COMMAND_SETDEBUG	setDebug	setDebug
COMMAND_STARTSESSION	startSession	• setAudienceID • setAudienceLevel • setDebug • setEventParameters • setInteractiveChannel • setRelyOnExistingSession

setAudienceID

setAudienceID(*IDAudiencia*)

El método setAudienceID define el IDAudiencia para los comandos setAudience y startSession.

- **IDAudiencia**: matriz de objetos NameValuePair que definen el IDAudiencia.

Valor de retorno

Ninguno.

Ejemplo

El ejemplo siguiente es un fragmento de un método executeBatch que llama a startSession y setAudience.

```
NameValuePair custId = new NameValuePairImpl();
custId.setName("IDCliente");
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);
. . .
/** Crear matriz de comandos */
Command[] commands =
    {
        startSessionCommand,
        setAudienceCommand,
    };
/** Realizar la llamada */
    BatchResponse batchResponse = api.executeBatch(sessionId, commands);

/** Procesar la respuesta de la forma adecuada */
    processExecuteBatchResponse(batchResponse);
```

setAudienceLevel

`setAudienceLevel(`*`nivelAudiencia`*`)`

El método `setAudienceLevel` define el Nivel de audiencia de los comandos `setAudience` y `startSession`.

- *nivelAudiencia*: serie que contiene el Nivel de audiencia.

  Importante: El nombre del *nivelAudiencia* debe coincidir con el nombre del nivel de audiencia tal como está definido exactamente en Campaign. Distingue entre mayúsculas y minúsculas.

Valor de retorno

Ninguno.

Ejemplo

El ejemplo siguiente es un fragmento de un método `executeBatch` que llama a `startSession` y `setAudience`.

```
String audienceLevel="Cliente";
. . .
Command startSessionCommand = new CommandImpl();
startSessionCommand.setAudienceID(initialAudienceId);
. . .
Command setAudienceCommand = new CommandImpl();
setAudienceCommand.setAudienceLevel(audienceLevel);
. . .
/** Crear matriz de comandos */
Command[] commands =
    {
        startSessionCommand,
        setAudienceCommand,
    };
/** Realizar la llamada */
    BatchResponse batchResponse = api.executeBatch(sessionId, commands);

/** Procesar la respuesta de la forma adecuada */
    processExecuteBatchResponse(batchResponse);
```

setDebug

`setDebug(debug)`

El método `setDebug` define el nivel de depuración del comando `startSession`. Si es `true`, el servidor de ejecución registra la información de depuración en el registro del servidor de ejecución. Si es `false`, el servidor de ejecución no registra información de depuración. El indicador de depuración se establece individualmente para cada sesión. Por lo tanto, puede rastrear los datos de depuración para una sesión de ejecución individual.

- **debug**: booleano (`true` o `false`).

Valor de retorno

Ninguno.

Ejemplo

El ejemplo siguiente es un fragmento de un método `executeBatch` que llama a `startSession` y `setDebug`.

```
boolean initialDebugFlag=true;
boolean newDebugFlag=false;
. . .
/* crear el comando startSession */
Command startSessionCommand = new CommandImpl();
startSessionCommand.setDebug(initialDebugFlag);
. . .

/* crear el comando setDebug */
Command setDebugCommand = new CommandImpl();
setDebugCommand.setMethodIdentifier(Command.COMMAND_SETDEBUG);
setDebugCommand.setDebug(newDebugFlag);

/** Crear matriz de comandos */
Command[] commands =
    {
        startSessionCommand,
        setDebugCommand,
    };
/** Realizar la llamada */
    BatchResponse batchResponse = api.executeBatch(sessionId, commands);

/** Procesar la respuesta de la forma adecuada */
    processExecuteBatchResponse(batchResponse);
```

setEvent

`setEvent(evento)`

El método `setEvent` define el nombre del evento utilizado por el comando `postEvent`.

- **evento**: serie que contiene el nombre del evento.

Importante: El nombre del *evento* debe coincidir con el nombre del evento tal como está definido exactamente en el canal interactivo. Distingue entre mayúsculas y minúsculas.

Valor de retorno

Ninguno.

Ejemplo

El ejemplo siguiente es un fragmento de un método executeBatch que llama a postEvent.

```
String eventName = "SearchExecution";

Command postEventCommand = new CommandImpl();
postEventCommand.setMethodIdentifier(Command.COMMAND_POSTEVENT);
postEventCommand.setEventParameters(postEventParameters);
postEventCommand.setEvent(eventName);
```

setEventParameters

setEventParameters(*parámetrosEvento*)

El método setEventParameters define los parámetros de evento utilizados por el comando postEvent. Estos valores se almacenan en los datos de la sesión.

- **parámetrosEvento**: matriz de objetos NameValuePair que define los parámetros del evento.

Por ejemplo, si el evento está registrando una oferta en el historial de contactos, debe incluir el código de tratamiento de la oferta.

Valor de retorno

Ninguno.

Ejemplo

El ejemplo siguiente es un fragmento de un método executeBatch que llama a postEvent.

```
NameValuePair parmB1 = new NameValuePairImpl();
parmB1.setName("SerieBúsqueda");
parmB1.setValueAsString("hipoteca");
parmB1.setValueDataType(NameValuePair.DATA_TYPE_STRING);

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

NameValuePair parmB3 = new NameValuePairImpl();
parmB3.setName("Navegador");
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("TemaPágina");
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

setGetOfferRequests(*númeroSolicitado*)

El método **setGetOfferRequests** establece el parámetro para recuperar ofertas utilizadas por el comando getOffersForMultipleInteractionPoints.

- **númeroSolicitado**: matriz de objetos de GetOfferRequest que define el parámetro para recuperar ofertas.

Valor de retorno

Ninguno.

Ejemplo

El ejemplo siguiente es un fragmento de un método GetOfferRequest que llama a setGetOfferRequests.

```
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

`setInteractiveChannel(`*canalInteractivo*`)`

El método `setInteractiveChannel` define el nombre del canal interactivo utilizado por el comando `startSession`.

- **canalInteractivo**: serie que contiene el nombre de canal interactivo.

Importante: *canalInteractivo* debe coincidir con el nombre del canal interactivo tal como está definido exactamente en Campaign. Distingue entre mayúsculas y minúsculas.

Valor de retorno

Ninguno.

Ejemplo

El ejemplo siguiente es un fragmento de un método `executeBatch` que llama a `startSession`.

```
String interactiveChannel="Sitio web Cuentas";
. . .
Command startSessionCommand = new CommandImpl();
startSessionCommand.setInteractiveChannel(interactiveChannel);
```

setInteractionPoint

`setInteractionPoint(`*puntoInteracción*`)`

El método `setInteractionPoint` define el nombre del punto de interacción utilizado por los comandos `getOffers` y `postEvent`.

- **puntoInteracción**: serie que contiene el nombre de punto de interacción.

Importante: *puntoInteracción* debe coincidir con el punto de interacción tal como está definido exactamente en el canal interactivo. Distingue entre mayúsculas y minúsculas.

Valor de retorno

Ninguno.

Ejemplo

El ejemplo siguiente es un fragmento de un método `executeBatch` que llama a `getOffers`.

```
String interactionPoint = "Banner de la página de visión general 1";
int numberRequested=1;

Command getOffersCommand = new CommandImpl();
getOffersCommand.setMethodIdentifier(Command.COMMAND_GETOFFERS);
getOffersCommand.setInteractionPoint(interactionPoint);
getOffersCommand.setNumberRequested(numberRequested);
```

setMethodIdentifier

`setMethodIdentifier(`*`métodoIdentificador`*`)`

El método `setMethodIdentifier` define el tipo de comando contenido en el objeto de comando.

- **métodoIdentificador**: serie que contiene el tipo de comando.

  Los valores válidos son:
  - **COMMAND_ENDSESSION**: representa el método `endSession`.
  - **COMMAND_GETOFFERS**: representa el método `getOffers`.
  - **COMMAND_GETPROFILE**: representa el método `getProfile`.
  - **COMMAND_GETVERSION**: representa el método `getVersion`.
  - **COMMAND_POSTEVENT**: representa el método `postEvent`.
  - **COMMAND_SETAUDIENCE**: representa el método `setAudience`.
  - **COMMAND_SETDEBUG**: representa el método `setDebug`.
  - **COMMAND_STARTSESSION**: representa el método `startSession`.

Valor de retorno

Ninguno.

Ejemplo

El ejemplo siguiente es un fragmento de un método `executeBatch` que llama a `getVersion` y `endSession`.

```
Command getVersionCommand = new CommandImpl();
getVersionCommand.setMethodIdentifier(Command.COMMAND_GETVERSION);

Command endSessionCommand = new CommandImpl();
endSessionCommand.setMethodIdentifier(Command.COMMAND_ENDSESSION);

Command[] commands =
{
    getVersionCommand,
    endSessionCommand
};
```

setNumberRequested

`setNumberRequested(`*`numberRequested`*`)`

El método `setNumberRequested` define el número de ofertas solicitadas por el comando `getOffers`.

- **numberRequested** — un entero que define el número de ofertas solicitadas por el comando `getOffers`.

Valor de retorno

Ninguno.

Ejemplo

El siguiente ejemplo es un extracto de un método `executeBatch` que invoca `getOffers`.

```
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

`setRelyOnExistingSession(`*relyOnExistingSession*`)`

El método `setRelyOnExistingSession` define un booleano que define si el comando `startSession` utiliza una sesión existente o no.

Si es `true`, el ID de sesión de `executeBatch` debe coincidir con un ID de sesión existente. Si es `false`, debe proporcionar un nuevo ID de sesión con el método `executeBatch`.

- **relyOnExistingSession**: booleano (`true` o `false`).

Valor de retorno

Ninguno.

Ejemplo

El ejemplo siguiente es un fragmento de un método `executeBatch` que llama a `startSession`.

```
boolean relyOnExistingSession=false;
. . .
Command startSessionCommand = new CommandImpl();
startSessionCommand.setRelyOnExistingSession(relyOnExistingSession);
```

Acerca de la interfaz NameValuePair

Muchos métodos de la API de Interact devuelven objetos NameValuePair o requieren que pase objetos NameValuePair como argumentos. Cuando se pasan como argumentos a un método, debe utilizar la implementación predeterminada `NameValuePairImpl`.

getName

`getName()`

El método `getName` devuelve el componente de nombre de un objeto NameValuePair.

Valor de retorno

El método `getName` devuelve una serie.

Ejemplo

El ejemplo siguiente es un fragmento de un método que procesa el objeto de respuesta para `getProfile`.

```
for(NameValuePair nvp : response.getProfileRecord())
{
    System.out.println("Nombre:"+nvp.getName());
}
```

getValueAsDate

```
getValueAsDate()
```

El método `getValueAsDate` devuelve el valor de un objeto NameValuePair.

Debe utilizar `getValueDataType` antes de utilizar `getValueAsDate` para confirmar que está haciendo referencia al tipo de datos correcto.

Valor de retorno

El método `getValueAsDate` devuelve una fecha.

Ejemplo

El ejemplo siguiente es un fragmento de un método que procesa un NameValuePair e imprime el valor si es una fecha.

```
if(nvp.getValueDataType().equals(NameValuePair.DATA_TYPE_DATE))
{
    System.out.println("Valor:"+nvp.getValueAsDate());
}
```

getValueAsNumeric

```
getValueAsNumeric()
```

El método `getValueAsNumeric` devuelve el valor de un objeto NameValuePair.

Debe utilizar `getValueDataType` antes de utilizar `getValueAsNumeric` para confirmar que está haciendo referencia al tipo de datos correcto.

Valor de retorno

El método `getValueAsNumeric` devuelve un valor doble. Si, por ejemplo, está recuperando un valor almacenado originalmente en la tabla de perfil como entero, `getValueAsNumeric` devuelve un valor doble.

Ejemplo

El ejemplo siguiente es un fragmento de un método que procesa un NameValuePair e imprime el valor si es numérico.

```
if(nvp.getValueDataType().equals(NameValuePair.DATA_TYPE_NUMERIC))
{
    System.out.println("Valor:"+nvp.getValueAsNumeric());
}
```

getValueAsString

```
getValueAsString()
```

El método `getValueAsString` devuelve el valor de un objeto NameValuePair.

Debe utilizar `getValueDataType` antes de utilizar `getValueAsString` para confirmar que está haciendo referencia al tipo de datos correcto.

Valor de retorno

El método getValueAsString devuelve una serie. Si, por ejemplo, está recuperando un valor almacenado originalmente en la tabla de perfil como char, varchar o char[10], getValueAsString devuelve un serie.

Ejemplo

El ejemplo siguiente es un fragmento de un método que procesa NameValuePair e imprime el valor si es una serie.

```
if(nvp.getValueDataType().equals(NameValuePair.DATA_TYPE_STRING))
{
    System.out.println("Valor:"+nvp.getValueAsString());
}
```

getValueDataType

```
getValueDataType()
```

El método getValueDataType devuelve el tipo de datos de un objeto NameValuePair.

Debe utilizar getValueDataType antes de utilizar getValueAsDate, getValueAsNumeric o getValueAsString para confirmar que está haciendo referencia al tipo de datos correcto.

Valor de retorno

El método getValueDataType devuelve una serie que indica si NameValuePair contiene datos, un número o una serie.

Los valores válidos son:

- **DATA_TYPE_DATETIME**: fecha que contiene un valor de fecha y hora.
- **DATA_TYPE_NUMERIC**: valor doble que contiene un valor de número.
- **DATA_TYPE_STRING**: serie que contiene un valor de texto.

Ejemplo

El ejemplo siguiente es un fragmento de un método que procesa el objeto de respuesta de un método getProfile.

```
for(NameValuePair nvp : response.getProfileRecord())
{
    System.out.println("Nombre:"+nvp.getName());
    if(nvp.getValueDataType().equals(NameValuePair.DATA_TYPE_DATETIME))
    {
        System.out.println("Valor:"+nvp.getValueAsDate());
    }
    else if(nvp.getValueDataType().equals(NameValuePair.DATA_TYPE_NUMERIC))
    {
        System.out.println("Valor:"+nvp.getValueAsNumeric());
    }
    else
    {
        System.out.println("Valor:"+nvp.getValueAsString());
    }
}
```

setName

`setName(nombre)`

El método `setName` define el componente de nombre de un objeto NameValuePair.

- **nombre**: serie que contiene el componente de nombre de un objeto NameValuePair.

Valor de retorno

Ninguno.

Ejemplo

El ejemplo siguiente muestra cómo definir el componente de nombre de un NameValuePair.

```
NameValuePair custId = new NameValuePairImpl();
custId.setName("IDCliente");
custId.setValueAsNumeric(1.0);
custId.setValueDataType(NameValuePair.DATA_TYPE_NUMERIC);
NameValuePair[] initialAudienceId = { custId };
```

setValueAsDate

`setValueAsDate(valorComoFecha)`

El método `setValueAsDate` define el valor de un objeto NameValuePair.

- **valorComoFecha**: fecha que contiene el valor de fecha y hora de un objeto NameValuePair.

Valor de retorno

Ninguno.

Ejemplo

El ejemplo siguiente muestra cómo definir el componente de valor de un NameValuePair si el valor es una fecha.

```
NameValuePair parm2 = new NameValuePairImpl();
parm2.setName("IndicaciónFechaHora");
parm2.setValueAsDate(new Date());
parm2.setValueDataType(NameValuePair.DATA_TYPE_DATETIME);
```

setValueAsNumeric

`setValueAsNumeric(valorComoNumérico)`

El método `setValueAsNumeric` define el valor de un objeto NameValuePair.

- **valorComoNumérico**: valor doble que contiene el valor numérico de un objeto NameValuePair.

Valor de retorno

Ninguno.

Ejemplo

El ejemplo siguiente muestra cómo definir el componente de valor de un NameValuePair si el valor es numérico.

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

setValueAsString

`setValueAsString(`*`valorComoSerie`*`)`

El método `setValueAsString` define el valor de un objeto NameValuePair.

- **valorComoSerie**: serie que contiene el valor de un objeto NameValuePair

Valor de retorno

Ninguno.

Ejemplo

El ejemplo siguiente muestra cómo definir el componente de valor de un NameValuePair si el valor es numérico.

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

setValueDataType

`getValueDataType(`*`valorTipoDatos`*`)`

El método `setValueDataType` define el tipo de datos de un objeto NameValuePair.

Los valores válidos son:

- **DATA_TYPE_DATETIME**: fecha que contiene un valor de fecha y hora.
- **DATA_TYPE_NUMERIC**: valor doble que contiene un valor de número.
- **DATA_TYPE_STRING**: serie que contiene un valor de texto.

Valor de retorno

Ninguno.

Ejemplo

Los ejemplos siguientes muestran cómo establecer el tipo de datos del valor de un NameValuePair.

```
NameValuePair parm2 = new NameValuePairImpl();
parm2.setName("IndicaciónFechaHora");
parm2.setValueAsDate(new Date());
parm2.setValueDataType(NameValuePair.DATA_TYPE_DATETIME);

NameValuePair parm3 = new NameValuePairImpl();
parm3.setName("Navegador");
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);
```

Acerca de la clase Offer

La clase `Offer` contiene métodos que definen un objeto de oferta. Este objeto de oferta contiene muchas de las mismas propiedades que una oferta en Campaign. El objeto de oferta contiene los siguientes atributos:

- **AdditionalAttributes** — NameValuePairs que contienen los atributos de oferta personalizados que ha definido en Campaign.
- **Description** — La descripción de la oferta.
- **EffectiveDate** — La fecha efectiva de la oferta.
- **ExpirationDate** — La fecha de caducidad de la oferta.
- **OfferCode** — El código de oferta de la oferta.
- **OfferName** — El nombre de la oferta.
- **TreatmentCode** — El código de tratamiento de la oferta.
- **Score** — La puntuación de marketing de la oferta, o la puntuación definida por `ScoreOverrideTable` si la propiedad `enableScoreOverrideLookup` es true.

getAdditionalAttributes

```
getAdditionalAttributes()
```

El método `getAdditionalAttributes` devuelve los atributos de oferta personalizados definidos en Campaign.

Valor de retorno

El método `getAdditionalAttributes` devuelve una matriz de objetos NameValuePair.

Ejemplo

El ejemplo siguiente ordena todos los atributos adicionales, comprobando la fecha efectiva y la fecha de caducidad, e imprimiendo los demás parámetros.

```
for(NameValuePair offerAttribute : offer.getAdditionalAttributes())
{
    // comprobar si existe la fecha efectiva
    if(offerAttribute.getName().equalsIgnoreCase("fechaEfectiva"))
    {
        System.out.println("Se ha encontrado la fecha efectiva");
    }
    // comprobar si existe la fecha de caducidad
    else if(offerAttribute.getName().equalsIgnoreCase("fechaCaducidad"))
    {
        System.out.println("Se ha encontrado la fecha de caducidad");
    }
    printNameValuePair(offerAttribute);
    }
}
public static void printNameValuePair(NameValuePair nvp)
{
    // imprimir el nombre:
    System.out.println("Nombre:"+nvp.getName());

    // en función del tipo de datos, llame al método adecuado para obtener el valor
```

```
        if(nvp.getValueDataType()==NameValuePair.DATA_TYPE_DATETIME)
            System.out.println("Valor de fecha:"+nvp.getValueAsDate());
        else if(nvp.getValueDataType()==NameValuePair.DATA_TYPE_NUMERIC)
            System.out.println("Valor numérico:"+nvp.getValueAsNumeric());
        else
            System.out.println("Valor de serie:"+nvp.getValueAsString());
    }
```

getDescription

```
getDescription()
```

El método `getDescription` devuelve la descripción de la oferta definida en Campaign.

Valor de retorno

El método `getDescription` devuelve una serie.

Ejemplo

El ejemplo siguiente imprime la descripción de una oferta.

```
for(Offer offer : offerList.getRecommendedOffers())
{
    // imprimir oferta
     System.out.println("Descripción de oferta:"+offer.getDescription());
}
```

getOfferCode

```
getOfferCode()
```

El método `getOfferCode` devuelve el código de oferta de la oferta tal como está definido en Campaign.

Valor de retorno

El método `getOfferCode` devuelve una matriz de series que contiene el código de oferta de la oferta.

Ejemplo

El ejemplo siguiente imprime el código de oferta de una oferta.

```
for(Offer offer : offerList.getRecommendedOffers())
{
    // imprimir oferta
   System.out.println("Código de oferta:"+offer.getOfferCode());
}
```

getOfferName

```
getOfferName()
```

El método `getOfferName` devuelve el nombre de la oferta tal como está definido en Campaign.

Valor de retorno

El método `getOfferName` devuelve una serie.

Ejemplo

El ejemplo siguiente imprime el nombre de una oferta.

```
for(Offer offer : offerList.getRecommendedOffers())
{
// imprimir oferta
System.out.println("Nombre de oferta:"+offer.getOfferName());
}
```

getScore

```
getScore()
```

El método `getScore` devuelve uno de los elementos siguientes:

- Si no ha habilitado la tabla de ofertas predeterminadas, la tabla de sustituciones de puntuación o el aprendizaje incorporado, este método devuelve la puntuación de marketing de la oferta tal como está definida en la pestaña de estrategia de interacción.
- Si ha habilitado las ofertas predeterminadas o la tabla de sustituciones de puntuación y no ha habilitado el aprendizaje incorporado, este método devuelve la puntuación de la oferta tal como está definida por el orden de prioridad entre la tabla de ofertas predeterminadas, la puntuación del usuario de marketing y la tabla de sustituciones de puntuación.
- Si ha habilitado el aprendizaje incorporado, este método devuelve la puntuación final que ha utilizado el aprendizaje incorporado para ordenar las ofertas.

Valor de retorno

El método `getScore` devuelve un entero que representa la puntuación de la oferta.

Ejemplo

El ejemplo siguiente imprime la puntuación de una oferta.

```
for(Offer offer : offerList.getRecommendedOffers())
{
// imprimir oferta
System.out.println("Puntuación de oferta:"+offer.getOfferScore());
}
```

getTreatmentCode

```
getTreatmentCode()
```

El método `getTreatmentCode` devuelve el código de tratamiento de la oferta tal como está definido en Campaign.

Dado que Campaign utiliza el código de tratamiento para identificar la instancia de la oferta ofrecida, este código se debe devolver como parámetro de evento cuando se utilice el método `postEvent` para registrar un evento de contacto, aceptación o rechazo de la oferta. Si está registrando la aceptación o el rechazo de una oferta, debe establecer el valor de nombre de NameValuePair que representa el código de tratamiento en `UACIOfferTrackingCode`.

Valor de retorno

El método `getTreatmentCode` devuelve una serie.

Ejemplo

El ejemplo siguiente imprime el código de tratamiento de una oferta.

```
for(Offer offer : offerList.getRecommendedOffers())
{
    // imprimir oferta
    System.out.println("Código de tratamiento de oferta:"+offer.getTreatmentCode());
}
```

Acerca de la clase OfferList

La clase `OfferList` contiene métodos que definen los resultados del método `getOffers`. El objeto OfferList contiene los siguientes atributos:

- **DefaultString** — La cadena predeterminada definida para el punto de interacción en el canal interactivo.
- **RecommendedOffers** — Una matriz de los objetos de oferta solicitados por el método `getOffers`.

La clase `OfferList` trabaja con listas de ofertas. Esta clase no está relacionada con las listas de ofertas de Campaign.

getDefaultString

```
getDefaultString()
```

El método `getDefaultString` devuelve la serie predeterminada para el punto de interacción tal como está definido en Campaign.

Si el objeto RecommendedOffers está vacío, debe configurar el punto de encuentro para presentar esta serie para asegurarse de que el contenido se presenta. Interact completa el objeto DefaultString solo si el objeto RecommendedOffers está vacío.

Valor de retorno

El método `getDefaultString` devuelve una serie.

Ejemplo

El ejemplo siguiente obtiene la serie predeterminada si el objeto offerList no contiene ofertas.

```
OfferList offerList=response.getOfferList();
if(offerList.getRecommendedOffers() != null)
{
    for(Offer offer : offerList.getRecommendedOffers())
    {
  System.out.println("Nombre de oferta:"+offer.getOfferName());
    }
}
else // basarse en la serie de oferta predeterminada
    System.out.println("Oferta predeterminada:"+offerList.getDefaultString());
```

getRecommendedOffers

```
getRecommendedOffers()
```

El método `getRecommendedOffers` devuelve una matriz de objetos Offer solicitados por el método `getOffers`.

Si la respuesta a getRecommendedOffer está vacía, el punto de encuentro debe presentar el resultado de getDefaultString.

Valor de retorno

El método getRecommendedOffers devuelve un objeto Offer.

Ejemplo

El ejemplo siguiente procesa el objeto OfferList e imprime el nombre de oferta para todas las ofertas recomendadas.

```
OfferList offerList=response.getOfferList();
if(offerList.getRecommendedOffers() != null)
{
    for(Offer offer : offerList.getRecommendedOffers())
    {
        // imprimir oferta
        System.out.println("Nombre de oferta:"+offer.getOfferName());
    }
}
else // basarse en la serie de oferta predeterminada
System.out.println("Oferta predeterminada:"+offerList.getDefaultString());
```

Acerca de la clase Response

La clase Response contiene métodos que definen los resultados de cualquiera de los métodos de la clase InteractAPI. El objeto Response contiene los atributos siguientes:

- **AdvisoryMessages**: matriz de mensajes de recomendación. Este atributo se completa solo si hay avisos o errores al ejecutar el método.
- **ApiVersion**: serie que contiene la versión de la API. Este atributo lo completa el método getVersion.
- **OfferList**: objeto OfferList que contiene las ofertas que solicita el método getOffers.
- **ProfileRecord**: matriz de NameValuePairs que contiene datos de perfil. Este atributo lo completa el método getProfile.
- **SessionID**: serie que define el ID de sesión. Lo devuelven todos los métodos de clase InteractAPI.
- **StatusCode**: número que indica si el método se ha ejecutado sin error, con un aviso, o con errores. Lo devuelven todos los métodos de clase InteractAPI.

getAdvisoryMessages

```
getAdvisoryMessages()
```

El método getAdvisoryMessages devuelve una matriz de Mensajes de recomendación del objeto Response.

Valor de retorno

El método getAdvisoryMessages devuelve una matriz de objetos Advisory Message.

Ejemplo

El ejemplo siguiente obtiene los objetos AdvisoryMessage de un objeto Response e itera a través de ellos, imprimiendo los mensajes.

```
AdvisoryMessage[] messages = response.getAdvisoryMessages();
    for(AdvisoryMessage msg : messages)
    {
        System.out.println(msg.getMessage());
        // Algunos mensajes de aviso pueden tener detalles adicionales:
        System.out.println(msg.getDetailMessage());
    }
```

getApiVersion

```
getApiVersion()
```

El método `getApiVersion` devuelve la versión de la API de un objeto Response.

El método `getVersion` completa el atributo ApiVersion de un objeto Response.

Valor de retorno

EL objeto Response devuelve una serie.

Ejemplo

El ejemplo siguiente es un fragmento de un método que procesa el objeto de respuesta de `getVersion`.

```
if(response.getStatusCode() == Response.STATUS_SUCCESS)
{
    System.out.println("llamada a getVersion procesada sin avisos o errores");
    System.out.println("Versión de la API:" + response.getApiVersion());
}
```

getOfferList

```
getOfferList()
```

El método `getOfferList` devuelve el objeto OfferList de un objeto Response.

El método `getOffers` completa el objeto OfferList de un objeto Response.

Valor de retorno

El objeto Response devuelve un objeto OfferList.

Ejemplo

El ejemplo siguiente es un fragmento de un método que procesa el objeto de respuesta para `getOffers`.

```
OfferList offerList=response.getOfferList();
if(offerList.getRecommendedOffers() != null)
{
    for(Offer offer : offerList.getRecommendedOffers())
    {
    // imprimir oferta
    System.out.println("Nombre de oferta:"+offer.getOfferName());
    }
}
```

getAllOfferLists

```
getAllOfferLists()
```

El método `getAllOfferLists` devuelve una matriz de todas las OfferLists de un objeto Response.

Lo utiliza el método `getOffersForMultipleInteractionPoints` que completa el objeto de matriz OfferList de un objeto Response.

Valor de retorno

El objeto Response devuelve una matriz OfferList.

Ejemplo

El ejemplo siguiente es un fragmento de un método que procesa el objeto de respuesta para `getOffers`.

```
OfferList[] allOfferLists = response.getAllOfferLists();
 if (allOfferLists != null) {
  for (OfferList ol : allOfferLists) {
   System.out.println("Las ofertas siguientes se entregan para el punto de interacción "
     + ol.getInteractionPointName() + ":");
   for (Offer o : ol.getRecommendedOffers()) {
    System.out.println(o.getOfferName());
   }
  }
 }
```

getProfileRecord

```
getProfileRecord()
```

El método `getProfileRecord` devuelve los registros de perfil para la sesión actual como una matriz de objetos NameValuePair. Estos registros de perfil incluyen también los `parámetrosEvento` añadidos anteriormente en la sesión de ejecución.

El método `getProfile` completa los objetos NameValuePair del registro del perfil de un objeto Response.

Valor de retorno

El objeto Response devuelve una matriz de objetos NameValuePair.

Ejemplo

El ejemplo siguiente es un fragmento de un método que procesa el objeto de respuesta para `getOffers`.

```
for(NameValuePair nvp : response.getProfileRecord())
{
    System.out.println("Nombre:"+nvp.getName());
    if(nvp.getValueDataType().equals(NameValuePair.DATA_TYPE_DATETIME))
    {
        System.out.println("Valor:"+nvp.getValueAsDate());
    }
    else if(nvp.getValueDataType().equals(NameValuePair.DATA_TYPE_NUMERIC))
    {
         System.out.println("Valor:"+nvp.getValueAsNumeric());
    }
    else
```

```
    {
        System.out.println("Valor:"+nvp.getValueAsString());
    }
}
```

getSessionID

```
getSessionID()
```

El método `getSessionID` devuelve el ID de sesión.

Valor de retorno

El método `getSessionID` devuelve una serie.

Ejemplo

El ejemplo siguiente muestra un mensaje que se puede visualizar al final o al inicio del error para indicar a qué sesión corresponde cualquier error.

```
System.out.println("Esta respuesta corresponde a sessionId:"+response.getSessionID());
```

getStatusCode

```
getStatusCode()
```

El método `getStatusCode` devuelve el código de estado de un objeto Response.

Valor de retorno

El objeto Response devuelve un entero.

- 0 - STATUS_SUCCESS: el método al que se ha llamado se ha completado sin errores. Puede haber Mensajes de recomendación o no.
- 1 - STATUS_WARNING: el método al que se ha llamado se ha completado con un mensaje de aviso como mínimo (pero sin errores). Consulte Mensajes de recomendación para ver más detalles.
- 2 - STATUS_ERROR: el método al que se ha llamado no se ha completado satisfactoriamente y tiene como mínimo un mensaje de error. Consulte Mensajes de recomendación para ver más detalles.

Ejemplo

A continuación se muestra un ejemplo de cómo puede utilizar `getStatusCode` en el manejo de errores.

```
public static void processSetDebugResponse(Response response)
{
    // comprobar si la respuesta ha sido satisfactoria o no
    if(response.getStatusCode() == Response.STATUS_SUCCESS)
    {
        System.out.println("llamada a setDebug procesada sin avisos o errores");
    }
    else if(response.getStatusCode() == Response.STATUS_WARNING)
    {
        System.out.println("llamada a setDebug procesada con un aviso");
    }
    else
    {
        System.out.println("llamada a setDebug procesada con un error");
    }
```

```
    // Para resultados no satisfactorios, debería haber mensajes de
    // recomendación que explicaran porqué
    if(response.getStatusCode() != Response.STATUS_SUCCESS)
        printDetailMessageOfWarningOrError("setDebug",
    response.getAdvisoryMessages());
}
```

Capítulo 8. Acerca de la API de ExternalCallout

Interact ofrece una macro ampliable, EXTERNALCALLOUT, que se utiliza con los diagramas de flujo interactivos. Esta macro permite ejecutar la lógica personalizada para comunicarse con sistemas externos durante las ejecuciones del diagrama de flujo. Por ejemplo, si desea calcular la puntuación de crédito de un cliente durante una ejecución de diagrama de flujo, puede crear una clase Java (una llamada) y, a continuación, utilizar la macro EXTERNALCALLOUT en un proceso Selección en el diagrama de flujo interactivo para obtener la puntuación de crédito de la llamada.

La configuración de EXTERNALCALLOUT consta de dos pasos principales. En primer lugar, debe crear una clase Java que implemente la API ExternalCallout. En segundo lugar, debe configurar las propiedades de configuración necesarias de la Marketing Platform en el servidor de ejecución en la categoría Interact | flowchart | ExternalCallouts.

Además de la información de esta sección, hay disponible un JavaDoc para la API ExternalCallout para cualquier servidor de ejecución de Interact en el directorio Interact/docs/externalCalloutJavaDoc.

Interfaz IAffiniumExternalCallout

La API de ExternalCallout está contenida en la interfaz IAffiniumExternalCallout. Debe implementar la interfaz IAffiniumExternalCallout para utilizar la macro EXTERNALCALLOUT.

La clase que implementa IAffiniumExternalCallout debe tener un constructor que pueda inicializar el servidor de ejecución.

- Si no hay constructores en la clase, el compilador Java crea un constructor predeterminado y ya es suficiente.
- Si hay constructores con argumentos, debe proporcionarse un constructor público sin argumento, que será utilizado por el servidor de ejecución.

Cuando desarrolle la llamada externa, recuerde lo siguiente:

- Cada evaluación de expresión con una llamada externa crea una nueva instancia de la clase. Debe gestionar los problemas de seguridad de subprocesos para los miembros estáticos en la clase.
- Si la llamada externa utiliza recursos del sistema como, por ejemplo, archivos o una conexión de base de datos, debe gestionar las conexiones. El servidor de ejecución no tiene un recurso para limpiar las conexiones automáticamente.

Debe compilar su implementación en interact_externalcallout.jar, que se encuentra en el directorio lib de la instalación del entorno de ejecución de IBM Unica Interact.

IAffiniumExternalCallout habilita el servidor de ejecución para solicitar datos de la clase Java. La interfaz consta de cuatro métodos:

- getNumberOfArguments
- getValue
- initialize
- shutdown

Adición de un servicio web para su uso con EXTERNALCALLOUT

La macro EXTERNALCALLOUT sólo reconoce las llamadas si ha definido la propiedades de configuración adecuadas.

En la Marketing Platform del entorno de ejecución, añada o defina las siguientes propiedades de configuración en la categoría Interact > flowchart > externalCallouts.

Propiedad de configuración	Valor
Categoría externalCallouts	Crear una nueva categoría para la llamada externa
class	los nombres de clase para la llamada externa
classpath	la ruta de clase para los archivos de clase de llamada externa
Categoría Parameter Data	Si la llamada externa requiere parámetros, cree nuevas propiedades de configuración de parámetros para ellos y asígneles un valor a cada uno

getNumberOfArguments

```
getNumberOfArguments()
```

El método getNumberOfArguments devuelve el número de argumentos esperados por la clase Java con la que se está integrando.

Valor de retorno

El método getNumberOfArguments devuelve un entero.

Ejemplo

El ejemplo siguiente muestra la impresión del número de argumentos.

```
public int getNumberOfArguments()
{
    return 0;
}
```

getValue

```
getValue(audienceID, configData, arguments)
```

El método getValue ejecuta la funcionalidad principal de la llamada y devuelve los resultados.

El método getValue requiere los siguientes parámetros:

- **audienceID** — un valor que identifica el ID de audiencia.
- **configData** — una correlación con pares clave-valor de datos de configuración necesarios para la llamada.
- **arguments** — los argumentos necesarios para la llamada. Cada argumento puede ser Cadena, Doble, Fecha o una Lista de alguno de estos. Una argumento de lista puede contener valores nulos. No obstante, una lista no puede contener, por ejemplo, una cadena y un doble.

  Debe ejecutarse la comprobación de tipos de argumentos en su implementación.

Si el método getValue falla por alguna razón, devuelve CalloutException.

Valor de retorno

El método getValue devuelve una lista de cadenas.

Ejemplo

```
public List<String> getValue(AudienceId audienceId, Map<String,
  String> configurationData, Object... arguments) throws CalloutException
{
 Long customerId = (Long) audienceId.getComponentValue("Customer");
  // ahora consulte scoreQueryUtility para obtener la puntuación de crédito de customerId
 Double score = scoreQueryUtility.query(customerId);
 String str = Double.toString(score);
 List<String> list = new LinkedList<String>();
 list.add(str);
 return list;
}
```

initialize

```
initialize(configData)
```

El método initialize se invoca una vez cuando se inicia el servidor de ejecución. Si hay operaciones que pueden obstaculizar el rendimiento durante el tiempo de ejecución como, por ejemplo, la carga de una tabla de base de datos, debe ejecutarlas este método.

El método initialize requiere el siguiente parámetro:

- **configData** — una correlación con pares clave-valor de datos de configuración necesarios para la llamada.

  Interact lee estos valores de los parámetros de llamada externa definidos en la categoría Interact > Flowchart > External Callouts > [External Callout] > Parameter Data.

Si el método initialize falla por alguna razón, devuelve CalloutException.

Valor de retorno

Ninguno.

Ejemplo

```
public void initialize(Map<String, String> configurationData) throws CalloutException
{
      // configurationData tiene los pares de clave-valor específicos del entorno
    // donde se ejecuta el servidor
      // inicializar scoreQueryUtility aquí
}
```

shutdown

```
shutdown(configData)
```

El método shutdown se invoca una vez cuando se cierra el servidor de ejecución. Si hay tareas de limpieza necesarias para la llamada, deben ejecutarse en este momento.

El método shutdown requiere el siguiente parámetro:

- **configData** — una correlación con pares clave-valor de datos de configuración necesarios para la llamada.

Si el método shutdown falla por alguna razón, devuelve CalloutException.

Valor de retorno

Ninguno.

Ejemplo

```
public void shutdown(Map<String, String> configurationData) throws CalloutException
{
     // cerrar scoreQueryUtility aquí
}
```

Ejemplo de API de ExternalCallout

1. Cree un archivo denominado GetCreditScore.java con el siguiente contenido. En este archivo se supone que hay una clase denominada ScoreQueryUtility que capta una puntuación de una aplicación de modelado.

```
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
{
// la clase que tiene la lógica para consultar un sistema externo para obtener una puntuación de crédito de un clientes
 private static ScoreQueryUtility scoreQueryUtility;
 public void initialize(Map<String, String> configurationData) throws CalloutException
 {
  // configurationData tiene los pares de clave-valor específicos del entorno donde se ejecuta el servidor
  // inicializar scoreQueryUtility aquí
 }

 public void shutdown(Map<String, String> configurationData) throws CalloutException
 {
 // cerrar scoreQueryUtility aquí
 }

 public int getNumberOfArguments()
 {
  // no espere ningún argumento adicional aparte del ID de cliente
  return 0;
 }

 public List<String> getValue(AudienceId audienceId, Map<String, String> configurationData,
  Object... arguments) throws CalloutException
 {
  Long customerId = (Long) audienceId.getComponentValue("Customer");
  // ahora consulte scoreQueryUtility para obtener la puntuación de crédito de customerId
  Double score = scoreQueryUtility.query(customerId);
  String str = Double.toString(score);
  List<String> list = new LinkedList<String>();
  list.add(str);
  return list;
 }
}
```

2. Compile GetCreditScore.java en GetCreditScore.class.
3. Cree un archivo jar llamado creditscore.jar que contenga GetCreditScore.class y los otros archivos de clase que utiliza.

4. Copie el archivo jar en alguna ubicación en el servidor de ejecución, por ejemplo, /data/interact/creditscore.jar.
5. Cree una llamada externa con el nombre GetCreditScore y una ruta de clase /data/interact/creditscore.jar en la categoría externalCallouts en la página Gestionar configuraciones.
6. En un diagrama de flujo interactivo, la llamada se puede utilizar como EXTERNALCALLOUT('GetCreditScore').

Interfaz IInteractProfileDataService

La API de Profile Data Services está contenida en la interfaz iInteractProfileDataService. Esta interfaz le permite importar datos jerárquicos a una sesión de Interact a través de uno o más orígenes de datos externos (tales como un archivo sin formato, servicio web, etc) en el momento en que se inicia la sesión de Interact o cambia el ID de audiencia de una sesión de Interact.

Para importar datos jerárquicos utilizando la API de Profile Data Services, debe escribir una clase Java que extrae información de cualquier origen de datos y la correlaciona con un objeto ISessionDataRootNode; luego haga referencia a estos datos correlacionados utilizando la macro EXTERNALCALLOUT

Debe compilar su implementación para interact_externalcallout.jar, que se encuentra en el directorio lib de la instalación del entorno de ejecución de IBM Unica Interact.

Para obtener un conjunto completo de documentación Javadoc para utilizar esta interfaz, vea los archivos contenidos en *directorio_inicio_Interact*/docs/externalCalloutJavaDoc con cualquier navegador web.

Para ver una implementación de muestra de cómo utilizar Profile Data Services, así como descripciones de cómo se implementó el ejemplo, consulte *directorio_inicio_Interact*/samples/externalcallout/XMLProfileDataService.java.

Añadir un origen de datos para utilizarlo con Profile Data Services

La macro EXTERNALCALLOUT reconoce un origen de datos para una importación de datos jerárquicos de Profile Data Services sólo si se han definido las propiedades de configuración apropiadas.

En la Marketing Platform correspondiente al entorno de ejecución, añada o defina las propiedades de configuración siguientes en la categoría Interact > Perfil > Niveles de audiencia > [AudienceLevelName] > Profile Data Services.

Propiedad de configuración	Valor
Categoría Nombre de categoría nuevo	Nombre del origen de datos que está definiendo. El nombre que especifique aquí debe ser exclusivo entre los orígenes de datos para un mismo nivel de audiencia.
enabled	Indica si el origen de datos está habilitado para el nivel de audiencia en el que está definido.
className	Nombre completo de la clase de origen de datos que implementa IInteractProfileDataService

Propiedad de configuración	Valor
classPath	Vía de acceso de los archivos de clase de Profile Data Services. Si no especifica un valor, se utiliza por omisión la vía de acceso de clases del servidor de aplicaciones
Categoría prioridad	Prioridad del origen de datos dentro de este nivel de audiencia. Debe ser un valor exclusivo entre todos los orígenes de datos para cada nivel de audiencia. (Es decir, si la prioridad se establece en 100 para un origen de datos, ningún otro origen de datos dentro del nivel de audiencia puede tener una prioridad de 100.)

Capítulo 9. Programas de utilidad de IBM Unica Interact

Esta sección describe los programas de utilidad de administración disponibles en Interact.

Ejecución del programa de utilidad de despliegue (runDeployment.sh/.bat)

La herramienta de línea de mandatos `runDeployment` le permite desplegar un canal interactivo para un grupo de servidores determinado desde la línea de mandatos, utilizando los valores proporcionados por un archivo `deployment.properties` que describe todos los parámetros posibles y reside en la misma ubicación que la propia herramienta `runDeployment`. La capacidad de ejecutar un despliegue de canal interactivo desde la línea de mandatos es especialmente útil cuando se utiliza la función `OffersBySQL`. Por ejemplo, puede configurar un diagrama de flujo por lotes de Campaign para que se ejecute de forma periódica. Cuando finaliza la ejecución del diagrama de flujo, se puede activar un desencadenante para inicializar el despliegue de las ofertas en la tabla `OffersBySQL` mediante la herramienta de línea de mandatos.

Descripción

La herramienta de línea de mandatos `runDeployment` se instala automáticamente en el servidor de tiempo de tiempo de diseño de Interact, en la ubicación siguiente:

directorio_inicio_Interact`/interactDT/tools/deployment/runDeployment.sh` (o `runDeployment.bat` en un servidor Windows)

El único argumento que se pasa al mandato es la ubicación de un archivo denominado `deployment.properties` que describe todos los parámetros posibles necesarios para desplegar la combinación canal interactivo/grupo de servidores de ejecución. Se proporciona un archivo de muestra para referencia.

Nota: Antes de utilizar el programa de utilidad `runDeployment`, primero debe editarlo con un editor de texto cualquiera para proporcionar la ubicación del entorno de ejecución Java en el servidor. Por ejemplo, puede especificar *directorio_inicio_Interact*`/jre` o *directorio_inicio_Platform*`/jre` como vía de acceso si cualquiera de esos dos directorios contiene el entorno de ejecución Java que desea que sea utilizado por el programa de utilidad. Como alternativa, puede proporcionar la vía de acceso de cualquier entorno de ejecución Java que se pueda utilizar con este release de los productos IBM Unica .

Utilización del programa de utilidad runDeployment en un entorno seguro (SSL)

Para utilizar el programa de utilidad runDeployment cuando se ha habilitado la seguridad en el servidor de Interact (y por lo tanto la conexión se realiza a través de un puerto SSL), debe añadir la propiedad trustStore de Java como se muestra a continuación:

1. Cuando esté editando el archivo `deployment.properties` para el despliegue del canal interactivo, modifique la propiedad `deploymentURL` para utilizar el URL seguro de SSL, como en este ejemplo:

deploymentURL=https://*<HOST>*.*<DOMAIN>*:*<PORT>*/Campaign/interact/InvokeDeploymentServlet

2. Edite el script runDeployment.sh o runDeployment.bat utilizando un editor de texto cualquiera para añadir el argumento siguiente a la línea que empieza por ${JAVA_HOME}:

   -Djavax.net.ssl.trustStore=*<TrustStorePath>*

   Por ejemplo, la línea podría tener este aspecto después de añadir el argumento trustStore:

   ```
   ${JAVA_HOME}/bin/java -Djavax.net.ssl.trustStore=<TrustStorePath>
   -cp ${CLASSPATH}com.unicacorp.Campaign.interact.deployment.tools.
   InvokeDeploymentClient $1
   ```

   Sustituya *<TrustStorePath>* por la vía de acceso del almacén de confianza de SSL.

Ejecución del programa de utilidad

Después de haber editado el programa de utilidad para proporcionar el entorno de ejecución Java, y haber personalizado una copia del archivo deployment.properties de acuerdo con el entorno utilizado, puede ejecutar el programa de utilidad con este mandato:

Interact_home/interactDT/tools/deployment/runDeployment.sh *deployment.properties*

Sustituya *directorio_inicio_Interact* por el valor real de la instalación de tiempo de diseño de Interact, y sustituya *deployment.properties* por la vía de acceso y el nombre del archivo de propiedades que ha personalizado para este despliegue.

Archivo deployment.properties de muestra

El archivo deployment.properties de muestra contiene una lista, en forma de comentario, de todos los parámetros que debe personalizar para que se adecuen al entorno utilizado. El archivo de muestra también contiene comentarios que explican lo que hace cada parámetro, y por qué puede ser necesario personalizar un valor determinado.

```
##############################################################################
#
# The following properties feed into the InvokeDeploymentClient program.
# The program will look for a deploymentURL setting.  The program will post a
# request against that url; all other settings are posted as parameters in
# that request. The program then checks the status of the deployment and
# returns back when the deployment is at a terminal state (or if the
# specified waitTime has been reached).
#
# the output of the program will be of this format:
# <STATE> : <Misc Detail>
#
# where state can be one of the following:
# ERROR
# RUNNING
# SUCCESS
#
# Misc Detail is data that would normally populate the status message area
#  in the deployment gui of the IC summary page. NOTE: HTML tags may exist
# in the Misc Detail
#
##############################################################################

##############################################################################
```

```
# deploymentURL: url to the InvokeDeployment servlet that resides in Interact
# Design time.  should be in the following format:
# http://dt_host:port/Campaign/interact/InvokeDeploymentServlet
################################################################################
deploymentURL=http://localhost:7001/Campaign/interact/InvokeDeploymentServlet

################################################################################
# dtLogin:  this is the login that you would use to login to the Design Time if
# you had wanted to deploy the IC via the deployment gui inside the IC summary
# page.
################################################################################
dtLogin=asm_admin

################################################################################
# dtPW:  this is the PW that goes along with the dtLogin
################################################################################
dtPW=

################################################################################
# icName:  this is the name of the Interactive Channel that you want to deploy
################################################################################
icName=ic1

################################################################################
# partition:  this is the name of the partition
################################################################################
partition=partition1

################################################################################
# request:  this is the type of request that you want this tool to execute
# currently, there two behaviors.  If the value is "deploy", then the deployment
# will be executed.  All other values would cause the tool to simply return the
# status of the last deployment of the specified IC.
################################################################################
request=deploy

################################################################################
# serverGroup:  this is the name of the server group that you would like to
# deploy the IC.
################################################################################
serverGroup=defaultServerGroup

################################################################################
# serverGroupType:  this will indicate whether or not this deployment is going
# against production server group or a test server group. 1 denotes production
# 2 denotes test.
################################################################################
serverGroupType=1

################################################################################
# rtLogin:  this is the account used to authenticate against the server group
# that you are deploying to.
################################################################################
rtLogin=asm_admin

################################################################################
# rtPW:  this is the password associated to the rtLogin
################################################################################
rtPW=

################################################################################
# waitTime:  Once the tool submits the deployment request, the tool will check
# the status of the deployment.  If the deployment has not completed (or
# failed), then the tool will continue to poll the system for the status until
# a completed state has been reached, OR until the specified waitTime (in
# seconds) has been reached.
################################################################################
```

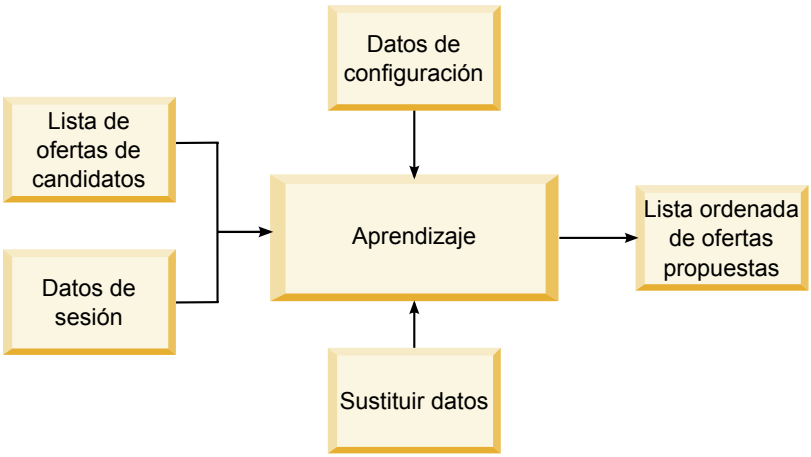
```
waitTime=5

################################################################################
# pollTime: If the status of a deployment is still in running state, then the
# tool will continue to check the status.  It will sleep in between status
# checks a number of seconds based on the pollTime setting .
################################################################################
pollTime=3

################################################################################
# global: Setting to false will make the tool NOT deploy the global settings.
# Non-availability of the property will still deploy the global settings.
################################################################################
global=true
```

Capítulo 10. Acerca de la API de aprendizaje

Interact ofrece un módulo de aprendizaje que utiliza un algoritmo Naive Bayesian para supervisar las acciones de los visitantes y proponer las ofertas óptimas (en términos de aceptación). Puede implementar la misma interfaz Java con sus propios algoritmos utilizando la API de aprendizaje para crear su propio módulo de aprendizaje.

Nota: Si utiliza el aprendizaje externo, los informes de ejemplo sobre el aprendizaje (los informes Detalles de aprendizaje de ofertas interactivas y Análisis de elevación de segmento interactivo) no devuelven datos válidos.

En el nivel más simple, la API de aprendizaje proporciona métodos para recopilar datos del entorno de ejecución y devolver una lista ordenada de ofertas recomendadas.



Puede recopilar los siguientes datos de Interact

- Datos de contactos de ofertas
- Datos de aceptación de ofertas
- Todos los datos de sesión
- Datos de ofertas específicos de Campaign
- Propiedades de configuración definidas en la categoría `learning` para el entorno de diseño y la categoría `offerserving` para el entorno de ejecución

Puede utilizar estos datos en los algoritmos para crear una lista de ofertas propuestas. A continuación, devuelva una lista de ofertas recomendadas, en el orden de mayor a menor recomendación.

Aunque no se muestra en el diagrama, también puede utilizar la API de aprendizaje para recopilar datos para su implementación de aprendizaje. Puede conservar estos datos en la memoria o registrarlos en un archivo o una base de datos para continuar el análisis.

Después de crear las clases Java, puede convertirlas en archivos jar. Una vez creados los archivos jar, también debe configurar el entorno de ejecución para reconocer el módulo de aprendizaje externo editando las propiedades de

configuración. Debe copiar los archivos jar o las clases Java para cada servidor de ejecución utilizando el módulo de aprendizaje externo.

Además de la información de esta guía, hay disponible un JavaDoc para la API del optimizador de aprendizaje en cualquier servidor de ejecución en el directorio `Interact/docs/learningOptimizerJavaDoc`.

Debe compilar su implementación en `interact_learning.jar`, que se encuentra en el directorio `lib` de la instalación del entorno de ejecución de Interact.

Cuando escribe su implementación de aprendizaje personalizada, debe tener en cuenta las siguientes directrices.

- El rendimiento es crítico.
- Debe trabajar con varios subprocesos y garantizar la seguridad para los subprocesos.
- Debe gestionar todos los recursos externos teniendo en cuenta los modos de error y el rendimiento.
- Utilice las excepciones, el registro (log4j) y la memoria según corresponda.

Habilitación del aprendizaje externo

Puede utilizar la API Java de aprendizaje para escribir su propio módulo de aprendizaje. Debe configurar el entorno de ejecución para que reconozca su utilidad de aprendizaje en la Marketing Platform.

En la Marketing Platform del entorno de ejecución, edite las siguientes propiedades de configuración en la categoría `Interact > offerserving`. Las propiedades de configuración de la API del optimizador de aprendizaje se encuentran en la categoría `Interact > offerserving > External Learning Config`.

Propiedad de configuración	**Valor**
`optimizationType`	**ExternalLearning**
`externalLearningClass`	Nombre de clase del aprendizaje externo
`externalLearningClassPath`	La ruta de la clase o los archivos jar en el servidor de ejecución para el aprendizaje externo. Si utiliza un grupo de servidores y todos los servidores de ejecución hacen referencia a la misma instancia de Marketing Platform, cada servidor debe tener una copia de la clase o los archivos jar en la misma ubicación.

Debe reiniciar el servidor de ejecución de Interact para que estos cambios entren en vigor.

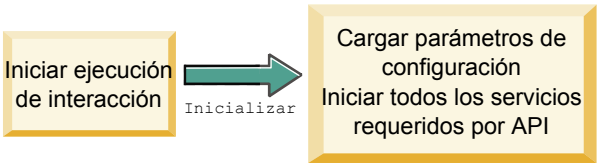
Interfaz ILearning

La API de aprendizaje se construye a partir de la interfaz `ILearning`. Debe implementar la interfaz `ILearning` para dar soporte a la lógica personalizada del módulo de aprendizaje.

Entre otras cosas, la interfaz `ILearning` permite recopilar datos del entorno de ejecución para la clase Java, y para enviar una lista de ofertas recomendadas al servidor de ejecución.

initialize

```
initialize(ILearningConfig config, boolean debug)
```



El método `initialize` se invoca una vez cuando se inicia el servidor de ejecución. Si hay operaciones que no se deben repetir, pero pueden obstaculizar el rendimiento durante el tiempo de ejecución como, por ejemplo, la carga de datos estáticos de una tabla de base de datos, debe ejecutarlas este método.

- **config** — un objeto `ILearningConfig` define todas las propiedades de configuración relevantes para el aprendizaje.
- **debug** — un booleano. Si es `true`, indica el nivel de detalle del nivel de registro en que se ha establecido el sistema del entorno de ejecución para la depuración. Para obtener los mejores resultados, seleccione este valor antes de escribir en un registro.

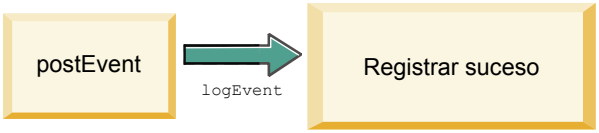
Si el método `initialize` falla por alguna razón, genera `LearningException`.

Valor de retorno

Ninguno.

logEvent

```
logEvent(ILearningContext context,
             IOffer offer,
             IClientArgs clientArgs,
             IInteractSession session,
             boolean debug)
```



El servidor de ejecución invoca el método `logEvent` siempre que la API de Interact publica un evento que está configurado para registrarse como un contacto o una respuesta. Utilice este método para registrar datos de contacto y de respuesta en una base de datos o un archivo a efectos de informes y aprendizaje. Por ejemplo, si desea determinar mediante algoritmos la probabilidad de que un cliente acepte una oferta basándose en los criterios, utilice este método para registrar los datos.

- **context** — un objeto `ILearningContext` que define el contexto de aprendizaje del evento, por ejemplo, contacto, aceptación o rechazo.
- **offer** — un objeto `IOffer` que define la oferta sobre la que se está registrando este evento.
- **clientArgs** — un objeto `IClientArgs` que define los parámetros. Actualmente, `logEvent` no necesita `clientArgs`, por lo que este parámetro puede estar vacío.
- **session** — un objeto `IInteractSession` que define todos los datos de sesión.

- **debug** — un booleano. Si es `true`, indica el nivel de detalle del nivel de registro en que se ha establecido el sistema del entorno de ejecución para la depuración. Para obtener los mejores resultados, seleccione este valor antes de escribir en un registro.

Si el método `logEvent` falla, genera una `LearningException`.

Valor de retorno

Ninguno.

optimizeRecommendList

```
optimizeRecommendList(list(ITreatment) recList,
  IClientArgs clientArg, IInteractSession session,
  boolean debug)
```

El método `optimizeRecommendList` debe utilizar la lista de ofertas recomendadas y los datos de sesión, y devolver una lista que contenga el número de ofertas solicitadas. El método `optimizeRecommendList` debe ordenar las ofertas de alguna manera, con su propio algoritmo de aprendizaje. La lista de ofertas debe ordenarse de modo que las ofertas que desee presentar primero estén al principio de la lista. Por ejemplo, si el algoritmo de aprendizaje proporciona una baja puntuación a las mejores ofertas, las ofertas deben ordenarse como 1, 2, 3. Si el algoritmo de aprendizaje proporciona una alta puntuación a las mejores ofertas, las ofertas deben ordenarse como 100, 99, 98.

El método `optimizeRecommendList` requiere los siguientes parámetros:

- **recList** — una lista de los objetos de tratamiento (ofertas) recomendados por el entorno de ejecución.
- **clientArg** — un objeto IClientArgs que contiene como mínimo el número de ofertas solicitadas por el entorno de ejecución.
- **session** — un objeto IInteractSession que contiene todos los datos de sesión.
- **debug** — un booleano. Si es `true`, indica el nivel de detalle del nivel de registro en que se ha establecido el sistema del entorno de ejecución para la depuración. Para obtener los mejores resultados, seleccione este valor antes de escribir en un registro.

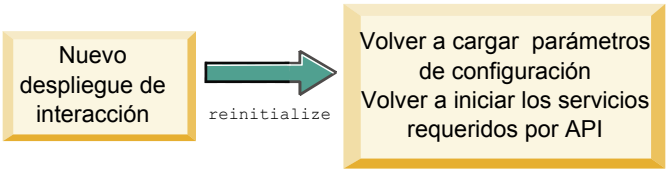
Si el método `optimizeRecommendList` falla, genera una `LearningException`.

Valor de retorno

El método `optimizeRecommendList` devuelve una lista de objetos `ITreatment`.

reinitialize

```
reinitialize(ILearningConfig config,
                  boolean debug)
```



El entorno de ejecución invoca el método `reinitialize` cada vez que hay un nuevo despliegue. Este método pasa todos los datos de configuración de aprendizaje. Si tiene servicios necesarios para la API de aprendizaje que leen propiedades de configuración, esta interfaz debe reiniciarlos.

- **config** — un objeto `ILearningConfig` que contiene todas las propiedades de configuración.
- **debug** — un booleano. Si es `true`, indica el nivel de detalle del nivel de registro en que se ha establecido el sistema del entorno de ejecución para la depuración. Para obtener los mejores resultados, seleccione este valor antes de escribir en un registro.

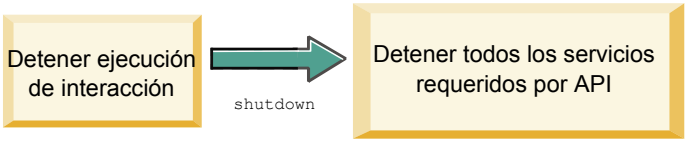
Si el método `logEvent` falla, genera una `LearningException`.

Valor de retorno

Ninguno.

shutdown

```
shutdown(ILearningConfig config, boolean debug)
```



El entorno de ejecución invoca el método `shutdown` cuando se cierra el servidor de ejecución. Si hay tareas de limpieza necesarias para el módulo de aprendizaje, deben ejecutarse en este momento.

El método `shutdown` requiere los siguientes parámetros.

- **config** — un objeto `ILearningConfig` que define todas las propiedades de configuración.
- **debug** — un booleano. Si es `true`, indica el nivel de detalle del nivel de registro en que se ha establecido el sistema del entorno de ejecución para la depuración. Para obtener los mejores resultados, seleccione este valor antes de escribir en un registro.

Si el método `shutdown` falla por alguna razón, genera `LearningException`.

Valor de retorno

Ninguno.

Interfaz IAudienceID

La interfaz `IAudienceID` da soporte a la interfaz `IInteractSession`. Es una interfaz para el ID de audiencia. Como el ID de audiencia puede estar formado por varias partes, esta interfaz permite acceder a todos los elementos del ID de audiencia, así como al nombre del nivel de audiencia.

getAudienceLevel

```
getAudienceLevel()
```

El método `getAudienceLevel` devuelve el nivel de audiencia.

Valor de retorno

El método `getAudienceLevel` devuelve una cadena que define el nivel de audiencia.

getComponentNames

```
getComponentNames()
```

El método `getComponentNames` obtiene un conjunto de los nombres de los componentes que forman el ID de audiencia. Por ejemplo, si el ID de audiencia está formado por los valores `customerName` y `accountID`, `getComponentNames` devolverá un conjunto que contiene las cadenas customerName y accountID.

Valor de retorno

Un conjunto de cadenas que contienen los nombres de los componentes del ID de audiencia.

getComponentValue

```
getComponentValue(String componentName)
```

El método `getComponentValue` devuelve el valor del componente denominado.

- **componentName** — una cadena que define el nombre del componente para el que desea recuperar el valor. Esta cadena no es sensible a las mayúsculas y minúsculas.

Valor de retorno

El método `getComponentValue` devuelve un objeto que define el valor del componente.

IClientArgs

La interfaz `IClientArgs` da soporte a la interfaz `ILearning`. Esta interfaz es una abstracción para cubrir los datos que se pasan al servidor desde el punto de encuentro que no están cubiertos por los datos de sesión. Por ejemplo, el número de ofertas solicitadas por el método `getOffers` de la API de Interact. Estos datos se almacenan en una correlación.

getValue

```
getValue(int clientArgKey)
```

El método `getValue` devuelve el valor del elemento de correlación solicitado.

Los siguientes elementos son necesarios en la correlación.

- **1** — NUMBER_OF_OFFERS_REQUESTED. El número de ofertas solicitadas por el método `getOffers` de la API de Interact. Esta constante devuelve un entero.

Valor de retorno

El método `getValue` devuelve un objeto que define el valor de la constante de correlación solicitada.

IInteractSession

La interfaz `IInteractSession` da soporte a la interfaz `ILearning`. Es una interfaz para la sesión actual en el entorno de ejecución.

getAudienceId

```
getAudienceId()
```

El método `getAudienceId` devuelve un objeto `AudienceID`. Utilice la interfaz `IAudienceID` para extraer los valores.

Valor de retorno

El método `getAudienceId` devuelve un objeto `AudienceID`.

getSessionData

```
getSessionData()
```

El método `getSessionData` devuelve una correlación no modificable de datos de sesión, donde el nombre de la variable de sesión es la clave. El nombre de la variable de sesión está siempre en mayúsculas. Utilice la interfaz `IInteractSessionData` para extraer los valores.

Valor de retorno

El método `getSessionData` devuelve un objeto `IInteractSessionData`.

Interfaz IInteractSessionData

La interfaz `IInteractSessionData` da soporte a la interfaz `ILearning`. Es una interfaz para los datos de sesión de ejecución del visitante actual. Los datos de sesión se almacenan como una lista de pares nombre-valor. También puede utilizar esta interfaz para cambiar el valor de los datos en la sesión de ejecución.

getDataType

```
getDataType(string parameterName)
```

El método `getDataType` devuelve el tipo de datos para el nombre de parámetro especificado.

Valor de retorno

El método `getDataType` devuelve un objeto InteractDataType. IntearctDataType es una enumeración Java representada por Desconocido, Cadena, Doble, Fecha o Lista.

getParameterNames

```
getParameterNames()
```

El método `getParameterNames` devuelve un conjunto de todos los nombres de los datos en la sesión actual.

Valor de retorno

El método `getParameterNames` devuelve un conjunto de nombres para los que se han establecido valores. Cada nombre del conjunto puede pasarse a getValue(String) para devolver un valor.

getValue

```
getValue(parameterName)
```

El método `getValue` devuelve el valor de objeto correspondiente al `parameterName` especificado. El objeto puede ser una Cadena, un Doble o una Fecha.

El método `getValue` requiere el siguiente parámetro:

- **parameterName** — una cadena que define el nombre del par nombre-valor de los datos de sesión.

Valor de retorno

El método `getValue` devuelve un objeto que contiene el valor de parámetro especificado.

setValue

```
setValue(string parameterName, object value)
```

El método `setValue` permite establecer un valor para el `parameterName` especificado. El valor puede ser una Cadena, un Doble o una Fecha.

El método `setValue` requiere los siguientes parámetros:

- **parameterName** — una cadena que define el nombre del par nombre-valor de los datos de sesión.
- **valor** — un objeto que define el valor del parámetro designado.

Valor de retorno

Ninguno.

ILearningAttribute

La interfaz `ILearningAttribute` da soporte a la interfaz `ILearningConfig`. Esta es una interfaz para los atributos de aprendizaje definidos en las propiedades de configuración en la categoría `learningAttributes`.

getName

```
getName()
```

El método `getName` devuelve el nombre del atributo de aprendizaje.

Valor de retorno

El método `getName` devuelve una cadena que define el nombre del atributo de aprendizaje.

ILearningConfig

La interfaz `ILearningConfig` da soporte a la interfaz `ILearning`. Es una interfaz para las propiedades de configuración de aprendizaje. Todos estos métodos devuelven el valor de la propiedad.

La interfaz consta de 15 métodos:

- **getAdditionalParameters** — devuelve una correlación de las propiedades adicionales definidas en la categoría `External Learning Config`
- **getAggregateStatsIntervalInMinutes** — devuelve un entero
- **getConfidenceLevel** — devuelve un entero
- **getDataSourceName** — devuelve una cadena
- **getDataSourceType** — devuelve una cadena
- **getInsertRawStatsIntervalInMinutes** — devuelve un entero
- **getLearningAttributes** — devuelve una lista de objetos `ILearningAttribute`
- **getMaxAttributeNames** — devuelve un entero
- **getMaxAttributeValues** — devuelve un entero
- **getMinPresentCountThreshold** — devuelve un entero
- **getOtherAttributeValue** — devuelve una cadena
- **getPercentRandomSelection** — devuelve un entero
- **getRecencyWeightingFactor** — devuelve un flotante
- **getRecencyWeightingPeriod** — devuelve un entero
- **isPruningEnabled** — devuelve un booleano

ILearningContext

La interfaz `ILearningContext` da soporte a la interfaz `ILearning`.

getLearningContext

```
getLearningContext()
```

El método `getLearningContext` devuelve una constante que indica si es un escenario de contactos, aceptaciones o rechazos.

- **1**—LOG_AS_CONTACT
- **2**—LOG_AS_ACCEPT
- **3**—LOG_AS_REJECT

4 y 5 están reservados para uso en el futuro.

Valor de retorno

El método `getLearningContext` devuelve un entero.

getResponseCode

```
getResponseCode()
```

El método `getResponseCode` devuelve el código de respuesta asignado a esta oferta. Este valor debe existir en la tabla `UA_UsrResponseType` en las tablas del sistema de Campaign.

Valor de retorno

El método `getResponseCode` devuelve una cadena que define el código de respuesta.

IOffer

La interfaz `IOffer` da soporte a la interfaz `ITreatment`. Es una interfaz para el objeto de oferta definido en el entorno de diseño. Utilice la interfaz `IOffer` para recopilar los detalles de oferta del entorno de ejecución.

getCreateDate

```
getCreateDate()
```

El método `getCreateDate` devuelve la fecha de creación de la oferta.

Valor de retorno

El método `getCreateDate` devuelve una fecha que define la fecha de creación de la oferta.

getEffectiveDateFlag

```
getEffectiveDateFlag()
```

El método `getEffectiveDateFlag` devuelve un número que define la fecha efectiva de la oferta.

- **0**: la fecha efectiva es una fecha absoluta, por ejemplo, 15 de marzo de 2010.
- **1**: la fecha efectiva es la fecha de recomendación.

Valor de retorno

El método `getEffectiveDateFlag` devuelve un entero que define la fecha efectiva de la oferta.

getExpirationDateFlag

```
getExpirationDateFlag()
```

El método `getExpirationDateFlag` devuelve un valor entero que describe la fecha de caducidad de la oferta.

- **0**: una fecha absoluta, por ejemplo, 15 de marzo de 2010.
- **1**: cierto número de días después de la recomendación, por ejemplo, 14.

- **2**: fin de mes después de la recomendación. Si una oferta se presenta el 31 de marzo, la oferta caducará ese día.

Valor de retorno

El método `getExpirationDateFlag` devuelve un entero que describe la fecha de caducidad de la oferta.

getOfferAttributes

```
getOfferAttributes()
```

El método `getOfferAttributes` devuelve atributos de oferta definidos para la oferta como un objeto `IOfferAttributes`.

Valor de retorno

El método `getOfferAttributes` devuelve un objeto `IOfferAttributes`.

getOfferCode

```
getOfferCode()
```

El método `getOfferCode` devuelve el código de oferta de la oferta tal como está definido en Campaign.

Valor de retorno

El método `getOfferCode` devuelve un objeto `IOfferCode`.

getOfferDescription

```
getOfferDescription()
```

El método `getOfferDescription` devuelve la descripción de la oferta definida en Campaign.

Valor de retorno

El método `getOfferDescription` devuelve una serie.

getOfferID

```
getOfferID()
```

El método `getOfferID` devuelve el ID de oferta tal como está definido en Campaign.

Valor de retorno

El método `getOfferID` devuelve un valor largo que define el ID de oferta.

getOfferName

```
getOfferName()
```

El método `getOfferName` devuelve el nombre de la oferta tal como está definido en Campaign.

Valor de retorno

El método `getOfferName` devuelve una serie.

getUpdateDate

```
getUpdateDate()
```

El método `getUpdateDate` devuelve la fecha de la última actualización de la oferta.

Valor de retorno

El método `getUpdateDate` devuelve una fecha que define cuándo se ha actualizado una oferta por última vez.

IOfferAttributes

La interfaz `IOfferAttributes` da soporte a la interfaz `IOffer`. Es una interfaz para los atributos de oferta definidos para una oferta en el entorno de diseño. Utilice la interfaz `IOfferAttributes` para recopilar los atributos de oferta del entorno de ejecución.

getParameterNames

```
getParameterNames()
```

El método `getParameterNames` devuelve una lista de los nombres de parámetros de oferta.

Valor de retorno

El método `getParameterNames` devuelve un conjunto que define la lista de nombres de parámetros de oferta.

getValue

```
getValue(String nombreParámetro)
```

El método `getValue` devuelve el valor del atributo de oferta especificado.

Valor de retorno

El método `getValue` devuelve un objeto que define el valor del atributo de oferta.

Interfaz IOfferCode

La interfaz `IOfferCode` da soporte a la interfaz `ILearning`. Es una interfaz para el código de oferta definido para una oferta en el entorno de diseño. Un código de oferta puede estar formado por una o varias cadenas. Utilice la interfaz `IOfferCode` para recopilar el código de oferta del entorno de ejecución.

getPartCount

```
getPartCount()
```

El método `getPartCount` devuelve el número de partes que componen un código de oferta.

Valor de retorno

El método `getPartCount` devuelve un valor entero que define el número de partes del código de oferta.

getParts

```
getParts()
```

El método `getParts` obtiene una lista no modificable de las partes del código de oferta.

Valor de retorno

El método `getParts` devuelve una lista no modificable de las partes del código de oferta.

LearningException

La clase `LearningException` da soporte a la interfaz `ILearning`. Algunos métodos en la interfaz requerirán que las implementaciones generen una `LearningException`, que es una subclase simple de `java.lang.Exception`. Se recomienda especialmente a efectos de depuración que la `LearningException` se construya con la excepción raíz, si existe alguna.

IScoreOverride

La interfaz `IScoreOverride` da soporte a la interfaz `ITreatment`. Esta interfaz permite leer los datos definidos en la tabla de alteración temporal de puntuaciones o la tabla de ofertas predeterminadas.

getOfferCode

```
getOfferCode()
```

El método `getOfferCode` devuelve el valor de las columnas de código de oferta en la tabla de sustituciones de puntuación para este miembro de audiencia.

Valor de retorno

El método `getOfferCode` devuelve un objeto `IOfferCode` que define el valor de las columnas de código de oferta en la tabla de sustituciones de puntuación.

getParameterNames

```
getParameterNames()
```

El método `getParameterNames` devuelve la lista de parámetros.

Valor de retorno

El método `getParameterNames` devuelve un conjunto que define la lista de parámetros.

El método `IScoreOverride` contiene los parámetros siguientes. A menos que se indique lo contrario, estos parámetros son los mismos que la tabla de sustituciones de puntuación.

- ADJ_EXPLORE_SCORE_COLUMN
- CELL_CODE_COLUMN
- ENABLE_STATE_ID_COLUMN
- ESTIMATED_PRESENT_COUNT: para sustituir el recuento actual estimado (durante el cálculo del peso de la oferta)
- FINAL_SCORE_COLUMN
- LIKELIHOOD_SCORE_COLUMN
- MARKETER_SCORE
- OVERRIDE_TYPE_ID_COLUMN
- PREDICATE_COLUMN: para crear una expresión booleana para determinar la elegibilidad de la oferta
- PREDICATE_SCORE: para crear una expresión que genere una puntuación numérica
- SCORE_COLUMN
- ZONE_COLUMN

También puede hacer referencia a cualquier columna que añada a la tabla de ofertas predeterminadas o de sustituciones de puntuación utilizando el mismo nombre que la columna.

getValue

`getValue(String nombreParámetro)`

El método `getValue` devuelve el valor de la columna de zona de la tabla de sustituciones de puntuación para este miembro de audiencia.

- **parameterName**: a serie que define el nombre del parámetro para el que desea el valor.

Valor de retorno

El método `getValue` devuelve un objeto que define el valor del parámetro solicitado.

ISelectionMethod

La interfaz `ISelection` indica el método utilizado para encontrar la lista recomendada. El valor predeterminado para el objeto de tratamiento es EXTERNAL_LEARNING, por lo que no tiene que establecer este valor. El valor se almacena en última instancia en el historial de contactos detallado para la creación de informes.

Puede ampliar esta interfaz más allá de las constantes existentes si desea almacenar los datos para su análisis posterior. Por ejemplo, puede crear dos módulos de aprendizaje diferentes e implementarlos en grupo de servidores diferentes. Puede ampliar la interfaz `ISelection` para incluir SERVER_GROUP_1 y SERVER_GROUP_2. A continuación, puede comparar los resultados de los dos módulos de aprendizaje.

Interfaz ITreatment

La interfaz `ITreatment` da soporte a la interfaz `ILearning` como una interfaz para la información de tratamiento. Un tratamiento representa la oferta asignada a una determinada celda tal como está definida en el entorno de diseño. A partir de esta interfaz, puede obtener información de celda y oferta, así como la puntuación de marketing asignada.

getCellCode

```
getCellCode()
```

El método `getCellCode` devuelve el código de celda tal como está definido en Campaign. La celda es la celda asignada al segmento inteligente asociado a esta oferta.

Valor de retorno

El método `getCellCode` devuelve una serie que define el código de celda.

getCellId

```
getOfferName()
```

El método `getCellId` devuelve el ID interno de la celda tal como está definido en Campaign. La celda es la celda asignada al segmento inteligente asociado a esta oferta.

Valor de retorno

El método `getCellId` devuelve un valor largo que define el ID de celda.

getCellName

```
getCellName()
```

El método `getCellName` devuelve el nombre de la celda tal como está definida en Campaign. La celda es la celda asignada al segmento inteligente asociado a esta oferta.

Valor de retorno

El método `getCellName` devuelve una serie que define el nombre de celda.

getLearningScore

```
getLearningScore()
```

El método `getLearningScore` devuelve la puntuación de este tratamiento. La prioridad es la siguiente.

1. Devolver el valor de sustitución, si está presente en la correlación Sustituir valores especificada en `IScoreoveride.PREDICATE_SCORE_COLUMN`
2. Devolver puntuación de predicado si el valor no es nulo
3. Devolver la puntuación de los usuarios de marketing, si está presente en la correlación Sustituir valores especificada en `IScoreoveride.SCORE`
4. Devolver la puntuación de los usuarios de marketing

Valor de retorno

El método `getLearningScore` devuelve un entero que define la puntuación determinada por el algoritmo de aprendizaje.

getMarketerScore

```
getMarketerScore()
```

El método `getMarketerScore` devuelve la puntuación del usuario de marketing definida mediante el graduador en la pestaña de estrategia de interacción para la oferta.

Para recuperar la puntuación de un usuario de marketing definida mediante las opciones avanzadas de la pestaña de estrategia de interacción, utilice `getPredicateScore`.

Para recuperar la puntuación del usuario de marketing utilizada realmente por el tratamiento, utilice `getLearningScore`.

Valor de retorno

El método `getMarketerScore` devuelve un entero que define la puntuación del usuario de marketing.

getOffer

```
getOffer()
```

El método `getOffer` devuelve la oferta para el tratamiento.

Valor de retorno

El método `getOffer` devuelve un objeto `IOffer` que define la oferta para este tratamiento.

getOverrideValues

```
getOverrideValues()
```

El método `getOverrideValues` devuelve las sustituciones definidas en las ofertas predeterminadas o tabla de sustituciones de puntuación.

Valor de retorno

El método `getOverrideValues` devuelve un objeto `IScoreOverride`.

getPredicate

```
getPredicate()
```

El método `getPredicate` devuelve el predicado definido por la columna de predicado de la tabla de ofertas predeterminadas, la tabla de sustituciones de puntuación o las opciones avanzadas de las reglas de tratamiento.

Valor de retorno

El método `getPredicate` devuelve una serie que define el predicado definido por la columna de predicado de la tabla de ofertas predeterminadas, la tabla de sustituciones de puntuación o las opciones avanzadas de las reglas de tratamiento.

getPredicateScore

```
getPredicateScore()
```

El método `getPredicateScore` devuelve la puntuación establecida por la columna de predicado de la tabla de ofertas predeterminadas, la tabla de sustituciones de puntuación o las opciones avanzadas de las reglas de tratamiento.

Valor de retorno

El método `getPredicateScore` devuelve un valor doble que define la puntuación establecida por la columna de predicado de la tabla de ofertas predeterminadas, la tabla de sustituciones de puntuación o las opciones avanzadas de las reglas de tratamiento.

getScore

```
getScore()
```

El método `getScore` devuelve uno de los elementos siguientes:

- La puntuación de marketing de la oferta tal como está definida en la pestaña de estrategia de interacción en Campaign si la propiedad `enableScoreOverrideLookup` está establecida en false.
- La puntuación de la oferta tal como está definida en `scoreOverrideTable` si la propiedad `enableScoreOverrideLookup` se establece en true.

Valor de retorno

El método `getScore` devuelve un entero que representa la puntuación de la oferta.

getTreatmentCode

```
getTreatmentCode()
```

El método `getTreatmentCode` devuelve el código de tratamiento.

Valor de retorno

El método `getTreatmentCode` devuelve una serie que define el código de tratamiento.

setActualValueUsed

```
setActualValueUsed(string nombreParm, object valor)
```

Utilice el método `setActualValueUsed` para definir qué valores se utilizan en las distintas etapas de la ejecución del algoritmo de aprendizaje.

Por ejemplo, si utiliza este método para grabar en las tablas de historial de respuestas y de contactos, y modificar los ejemplos de muestra existentes, puede incluir datos del algoritmo de aprendizaje en la creación de informes.

- **nombreParm**: serie que define el nombre del parámetro que está estableciendo.
- **valor**: objeto que define el valor del parámetro que está estableciendo.

Valor de retorno

Ninguno.

Ejemplo de API de aprendizaje

Esta sección contiene una implementación de muestra de ILearningInterface. Tenga en cuenta que esta implementación es sólo una muestra y no está diseñada para utilizarse en un entorno de producción.

Este ejemplo realiza un seguimiento de los recuentos de aceptaciones y contactos, y utiliza la proporción de aceptaciones y contactos para una determinada oferta como la tasa de probabilidad de aceptación de la oferta. Las ofertas que no se presentan reciben una mayor prioridad de recomendación. Las ofertas con al menos un contacto se ordenan según la tasa de probabilidad de aceptación descendente.

En este ejemplo, todos los recuentos se mantienen en memoria. No es un escenario realista, ya que el servidor de ejecución se quedará sin memoria. En un escenario de producción real, los recuentos deben ser persistentes en una base de datos.

```
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;

/**
 * Es una implementación de muestra del optimizador de aprendizaje.
 * La interfaz ILearning puede encontrarse en la biblioteca interact.jar.
 *
 * Para utilizar realmente esta implementación, seleccione ExternalLearning como optimizationType en el nodo offerServing
 * de la aplicación de Interact en la configuración de plataforma. En el nodo offerserving también hay
 * una categoría External Learning Config, donde debe establecer el nombre de la clase en:
 * com.unicacorp.interact.samples.learning.v2.SampleLearning. Tenga en cuenta que esta implementación es sólo una muestra
 * y no está diseñada para utilizarse en un entorno de producción.
 *
 *
 * Este ejemplo realiza un seguimiento de los recuentos de aceptaciones y contactos, y utiliza la proporción de aceptaciones y contactos
 * para una determinada oferta como la tasa de probabilidad de aceptación de la oferta.
 *
 *
 * Las ofertas que no se presenten recibirán una mayor prioridad de recomendación.
 * Las ofertas con al menos un contacto se ordenarán según la tasa de probabilidad de aceptación descendente.
 *
 * Nota: todos los recuentos se mantienen en memoria. No es un escenario realista, ya que se quedará sin memoria antes o
 * después. En un escenario de producción real, los recuentos deben ser persistentes en una base de datos.
 *
 */

public class SampleLearning implements ILearning
{
```

```
// Una correlación de ID de oferta con recuentos de contactos para el ID de oferta
private Map<Long,Integer> _offerToContactCount = new HashMap<Long, Integer>();

// Una correlación de ID de oferta con recuentos de contactos para el ID de oferta
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
{
    // Si se necesitan conexiones remotas, es un buen lugar para inicializarlas ya que este
    // método se invoca una vez al inicio de la aplicación web de tiempo de ejecución de Interact.
    // Este ejemplo no tiene conexiones remotas e imprime a efectos de depuración que este método
    // se invocará
    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
{
    // Si se despliega un IC, se invoca este método de reinicialización para que la implementación
    // renueve los valores de configuración actualizados
    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");

        // Realizar un seguimiento de todos los contactos en la memoria
        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");
        // Realizar un seguimiento de todas las cuentas de aceptación en la memoria mediante la adición a la correlación
        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");

        // Ordenar los tratamientos candidatos invocando el clasificador definido en esta clase y devolver la lista ordenada
        Collections.sort(recList,new MyOfferSorter());

        // ahora devolver sólo lo que se ha solicitado mediante la variable "numberRequested"
        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
    {
        // Si existen conexiones remotas, es un buen lugar para desconectarse ordenadamente
        // de ellas ya que este método se invoca en la conclusión de la aplicación web de tiempo de ejecución de
        // Interact. Para este ejemplo, no hay nada que hacer
        // excepto imprimir una sentencia de depuración.
        System.out.println("Calling shutdown for SampleLearning");

    }
    // Ordenar por:
    // 1. ofertas con cero contactos - para los vínculos, el orden se basa en la entrada original
    // 2. tasa de probabilidad de aceptación descendente - para los vínculos, el orden se basa en la entrada original

    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)
        {

            // obtener el recuento de contactos para ambos tratamientos
            Integer contactCount1 = _offerToContactCount.get(treatment1.getOffer().getOfferId());
            Integer contactCount2 = _offerToContactCount.get(treatment2.getOffer().getOfferId());

            // si no se ha puesto en contacto con el tratamiento, será el ganador
            if(contactCount1 == null || contactCount1 == 0)
                return -1;

            if(contactCount2 == null || contactCount2 == 0)
                return 1;

            // obtener los recuentos de aceptación
            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;

            // orden descendente
            return (int) (acceptProbability2 - acceptProbability1);

        }
    }

}
```

Apéndice A. IBM Unica Interact WSDL

```
<?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>
```

Apéndice B. Propiedades de configuración del entorno de ejecución de Interact

Esta sección describe todas las propiedades de configuración del entorno de ejecución de Interact.

Interact | general

Estas propiedades de configuración definen los valores generales del entorno de ejecución, incluidos el nivel de registro predeterminado y el valor de configuración regional.

log4jConfig

Descripción

La ubicación del archivo que contiene las propiedades log4j. Esta ruta debe ser relativa a la variable de entorno `INTERACT_HOME`. `INTERACT_HOME` es la ubicación del directorio de instalación de Interact.

Valor predeterminado

`./conf/interact_log4j.properties`

asmUserForDefaultLocale

Descripción

La propiedad `asmUserForDefaultLocale` define el usuario de IBM Unica Marketing del que Interact deriva sus valores de configuración regional.

Los valores de configuración regional definen qué idioma aparece en el tiempo de diseño y en qué idioma están los mensajes de aviso de la API de Interact. Si el valor de configuración regional no coincide con los valores del sistema operativo de las máquinas, Interact todavía funciona, aunque la pantalla del tiempo de diseño y los mensajes de aviso pueden aparecer en otro idioma.

Valor predeterminado

No se ha definido ningún valor predeterminado.

Interact | general | learningTablesDataSource

Estas propiedades de configuración definen la configuración del origen de datos para las tablas de aprendizaje incorporadas. Debe definir este origen de datos si utiliza el aprendizaje incorporado de Interact.

Si crea su propia implementación de aprendizaje utilizando la API de aprendizaje, puede configurar su implementación de aprendizaje personalizada para leer estos valores utilizando la interfaz `ILearningConfig`.

jndiName

Descripción

Utilice esta propiedad jndiName para identificar el origen de datos JNDI (Java Naming and Directory Interface) que está definido en el servidor de aplicaciones (WebSphere o WebLogic) para las tablas de aprendizaje a las que acceden los servidores de ejecución de Interact.

Las tablas de aprendizaje se crean utilizando el archivo ddl aci_lrntab y contienen las siguientes tablas (entre otras): UACI_AttributeValue y UACI_OfferStats.

Valor predeterminado

No se ha definido ningún valor predeterminado.

type

Descripción

El tipo de base de datos para el origen de datos utilizado por las tablas de aprendizaje a las que acceden los servidores de ejecución de Interact.

Las tablas de aprendizaje se crean utilizando el archivo ddl aci_lrntab y contienen las siguientes tablas (entre otras): UACI_AttributeValue y UACI_OfferStats.

Valor predeterminado

SQLServer

Valores válidos

SQLServer | DB2 | ORACLE

connectionRetryPeriod

Descripción

La propiedad ConnectionRetryPeriod especifica la cantidad de tiempo en segundos que Interact reintenta automáticamente la solicitud de conexión de base de datos anómala para las tablas de aprendizaje. Interact intenta reconectarse automáticamente a la base de datos durante este periodo de tiempo antes de informar de un error o una anomalía de base de datos. Si el valor se establece en 0, Interact intentará reconectarse indefinidamente; si el valor se establece en -1, no se realizará ningún intento.

Las tablas de aprendizaje se crean utilizando el archivo ddl aci_lrntab y contienen las siguientes tablas (entre otras): UACI_AttributeValue y UACI_OfferStats.

Valor predeterminado

-1

connectionRetryDelay

Descripción

La propiedad ConnectionRetryDelay especifica la cantidad de tiempo en segundos que Interact espera antes de intentar reconectarse a la base de datos después de una anomalía para las tablas de aprendizaje. Si el valor se establece en -1, no realiza ningún intento.

Las tablas de aprendizaje se crean utilizando el archivo ddl aci_lrntab y contienen las siguientes tablas (entre otras): UACI_AttributeValue y UACI_OfferStats.

Valor predeterminado

-1

schema

Descripción

El nombre del esquema que contiene las tablas para el módulo de aprendizaje incorporado. Interact inserta el valor de esta propiedad antes de todos los nombres de tabla, por ejemplo, `UACI_IntChannel` pasa a ser `schema.UACI_IntChannel`.

No tiene que definir un esquema. Si no define un esquema, Interact supone que el propietario de las tablas es el mismo que el del esquema. Debe establecer este valor para eliminar ambigüedades.

Valor predeterminado

No se ha definido ningún valor predeterminado.

Interact | general | prodUserDataSource

Estas propiedades de configuración definen la configuración del origen de datos para las tablas de perfil de producción. Debe definir este origen de datos. Es el origen de datos al que hace referencia el entorno de ejecución cuando ejecuta diagramas de flujo interactivos después del despliegue.

jndiName

Descripción

Utilice esta propiedad `jndiName` para identificar el origen de datos JNDI (Java Naming and Directory Interface) que está definido en el servidor de aplicaciones (WebSphere o WebLogic) para las tablas de clientes a las que acceden los servidores de ejecución de Interact.

Valor predeterminado

No se ha definido ningún valor predeterminado.

type

Descripción

El tipo de base de datos para las tablas de clientes a las que acceden los servidores de ejecución de Interact.

Valor predeterminado

SQLServer

Valores válidos

SQLServer | DB2 | ORACLE

aliasPrefix

Descripción

La propiedad `AliasPrefix` especifica la manera en que Interact forma el nombre de alias que Interact crea automáticamente cuando utiliza una tabla de dimensiones y escribe en una nueva tabla en las tablas de clientes a las que acceden los servidores de ejecución de Interact.

Tenga en cuenta que cada base de datos tiene una longitud de identificador máxima; revise la documentación para la base de datos que está utilizando para asegurarse de que el valor que ha establecido no excede la longitud de identificador máxima para su base de datos.

Valor predeterminado

A

connectionRetryPeriod

Descripción

La propiedad `ConnectionRetryPeriod` especifica la cantidad de tiempo en segundos que Interact reintenta automáticamente la solicitud de conexión de base de datos anómala para las tablas de clientes de tiempo de ejecución. Interact intenta reconectarse automáticamente a la base de datos durante este periodo de tiempo antes de informar de un error o una anomalía de base de datos. Si el valor se establece en 0, Interact intentará reconectarse indefinidamente; si el valor se establece en -1, no se realizará ningún intento.

Valor predeterminado

-1

connectionRetryDelay

Descripción

La propiedad `ConnectionRetryDelay` especifica la cantidad de tiempo en segundos que Interact espera antes de intentar reconectarse a la base de datos después de una anomalía para las tablas de clientes de tiempo de ejecución de Interact. Si el valor se establece en -1, no realiza ningún intento.

Valor predeterminado

-1

schema

Descripción

El nombre del esquema que contiene las tablas de datos de perfil. Interact inserta el valor de esta propiedad antes de todos los nombres de tabla, por ejemplo, `UACI_IntChannel` pasa a ser `schema.UACI_IntChannel`.

No tiene que definir un esquema. Si no define un esquema, Interact supone que el propietario de las tablas es el mismo que el del esquema. Debe establecer este valor para eliminar ambigüedades.

Valor predeterminado

No se ha definido ningún valor predeterminado.

Interact | general | systemTablesDataSource

Estas propiedades de configuración definen la configuración del origen de datos para las tablas del sistema del entorno de ejecución. Debe definir este origen de datos.

jndiName

Descripción

Utilice esta propiedad `jndiName` para identificar el origen de datos JNDI (Java Naming and Directory Interface) que está definido en el servidor de aplicaciones (WebSphere o WebLogic) para las tablas del entorno de ejecución.

La base de datos del entorno de ejecución es la base de datos que se completa con los scripts dll `aci_runtime` y `aci_populate_runtime` y, por ejemplo, contiene las siguientes tablas (entre otras): `UACI_CHOfferAttrib` y `UACI_DefaultedStat`.

Valor predeterminado

No se ha definido ningún valor predeterminado.

type

Descripción

El tipo de base de datos para las tablas del sistema del entorno de ejecución.

La base de datos del entorno de ejecución es la base de datos que se completa con los scripts dll `aci_runtime` y `aci_populate_runtime` y, por ejemplo, contiene las siguientes tablas (entre otras): `UACI_CHOfferAttrib` y `UACI_DefaultedStat`.

Valor predeterminado

SQLServer

Valores válidos

SQLServer | DB2 | ORACLE

connectionRetryPeriod

Descripción

La propiedad `ConnectionRetryPeriod` especifica la cantidad de tiempo en segundos que Interact reintenta automáticamente la solicitud de conexión de base de datos anómala para las tablas del sistema de tiempo de ejecución. Interact intenta reconectarse automáticamente a la base de datos durante este periodo de tiempo antes de informar de un error o una anomalía de base de datos. Si el valor se establece en 0, Interact intentará reconectarse indefinidamente; si el valor se establece en -1, no se realizará ningún intento.

La base de datos del entorno de ejecución es la base de datos que se completa con los scripts dll `aci_runtime` y `aci_populate_runtime` y, por ejemplo, contiene las siguientes tablas (entre otras): `UACI_CHOfferAttrib` y `UACI_DefaultedStat`.

Valor predeterminado

-1

connectionRetryDelay

Descripción

La propiedad ConnectionRetryDelay especifica la cantidad de tiempo en segundos que Interact espera antes de intentar reconectarse a la base de datos después de una anomalía para las tablas del sistema de tiempo de ejecución de Interact. Si el valor se establece en -1, no realiza ningún intento.

La base de datos del entorno de ejecución es la base de datos que se completa con los scripts dll aci_runtime y aci_populate_runtime y, por ejemplo, contiene las siguientes tablas (entre otras): UACI_CHOfferAttrib y UACI_DefaultedStat.

Valor predeterminado

-1

schema

Descripción

El nombre del esquema que contiene las tablas para el entorno de ejecución. Interact inserta el valor de esta propiedad antes de todos los nombres de tabla, por ejemplo, UACI_IntChannel pasa a ser schema.UACI_IntChannel.

No tiene que definir un esquema. Si no define un esquema, Interact supone que el propietario de las tablas es el mismo que el del esquema. Debe establecer este valor para eliminar ambigüedades.

Valor predeterminado

No se ha definido ningún valor predeterminado.

Interact | general | systemTablesDataSource | loaderProperties

Estas propiedades de configuración definen la configuración de una utilidad de carga de base de datos para las tablas del sistema del entorno de ejecución. Sólo debe definir estas propiedades si utiliza una utilidad de carga de base de datos.

databaseName

Descripción

El nombre de la base de datos a la que se conecta el cargador de base de datos.

Valor predeterminado

No se ha definido ningún valor predeterminado.

LoaderCommandForAppend

Descripción

El parámetro LoaderCommandForAppend especifica el comando emitido para invocar la utilidad de carga de base de datos para añadir registros a las tablas de base de datos de preparación del historial de contactos y respuestas en Interact. Debe establecer este parámetro para habilitar la utilidad de carga de base de datos para los datos del historial de contactos y respuestas.

Este parámetro se especifica como un nombre de ruta completo al ejecutable de la utilidad de carga de base de datos o a un script que inicia la utilidad de carga de base de datos. La utilización de un script permite realizar configuraciones adicionales antes de invocar la utilidad de carga.

La mayoría de las utilidades de carga de base de datos requieren varios argumentos para poder iniciarse satisfactoriamente. Por ejemplo, la especificación del archivo de datos y el archivo de control desde los que se carga, y la base de datos y la tabla donde se carga. Los tokens se sustituyen por los elementos especificados cuando se ejecuta el comando.

Consulte la documentación de la utilidad de carga de base de datos para conocer la sintaxis correcta que se debe utilizar cuando se invoca la utilidad de carga de base de datos.

Este parámetro no está definido de forma predeterminada.

Los tokens disponibles para `LoaderCommandForAppend` se describen en la siguiente tabla.

Token	**Descripción**
<CONTROLFILE>	Este token se sustituye por la ruta completa y el nombre de archivo del archivo de control temporal que Interact genera de acuerdo con la plantilla que se especifica en el parámetro `LoaderControlFileTemplate`.
<DATABASE>	Este token se sustituye por el nombre del origen de datos en el que Interact está cargando datos. Es el mismo nombre de origen de datos que se utiliza en el nombre de categoría para este origen de datos.
<DATAFILE>	Este token se sustituye por la ruta completa y el nombre de archivo del archivo de datos temporal creado por Interact durante el proceso de carga. Este archivo se encuentra en el directorio temporal de Interact, `UNICA_ACTMPDIR`.
<DBCOLUMNNUMBER>	Este token se sustituye por el ordinal de columna en la base de datos.
<FIELDLENGTH>	Este token se sustituye por la longitud del campo que se está cargando en la base de datos.
<FIELDNAME>	Este token se sustituye por el nombre del campo que se está cargando en la base de datos.
<FIELDNUMBER>	Este token se sustituye por el número del campo que se está cargando en la base de datos.

Token	Descripción
<FIELDTYPE>	Este token se sustituye por el literal "CHAR( )". La longitud de este campo se especifica entre paréntesis (). Si la base de datos no entiende el tipo de campo, CHAR, puede especificar manualmente el texto correspondiente para el tipo de campo y utilizar el token <FIELDLENGTH>. Por ejemplo, para SQLSVR y SQL2000, puede utilizar "SQLCHAR(<FIELDLENGTH>)".
<NATIVETYPE>	Este token se sustituye por el tipo de base de datos en el que se carga este campo.
<NUMFIELDS>	Este token se sustituye por el número de campos en la tabla.
<PASSWORD>	Este token se sustituye por la contraseña de base de datos de la conexión de diagrama de flujo actual con el origen de datos.
<TABLENAME>	Este token se sustituye por el nombre de tabla de base de datos en el que Interact está cargando datos.
<USER>	Este token se sustituye por el usuario de base de datos de la conexión de diagrama de flujo actual con el origen de datos.

Valor predeterminado

No se ha definido ningún valor predeterminado.

LoaderControlFileTemplateForAppend

Descripción

La propiedad `LoaderControlFileTemplateForAppend` especifica la ruta completa y el nombre de archivo de la plantilla de archivo de control que se ha configurado previamente en Interact. Cuando se establece este parámetro, Interact crea dinámicamente un archivo de control temporal basado en la plantilla que se especifica aquí. La ruta y el nombre de este archivo de control temporal están disponibles en el token `<CONTROLFILE>` que está disponible en la propiedad `LoaderCommandForAppend`.

Antes de utilizar Interact en modo de utilidad de carga de la base de datos, debe configurar la plantilla de archivo de control que ha especificado este parámetro. La plantilla de archivo de control da soporte a los siguientes tokens, que se sustituyen dinámicamente cuando Interact crea el archivo de control temporal.

Consulte la documentación de la utilidad de carga de base de datos para conocer la sintaxis correcta que requiere el archivo de control. Los tokens disponibles para su plantilla de archivo de control son los mismos que los de la propiedad `LoaderControlFileTemplate`.

Este parámetro no está definido de forma predeterminada.

Valor predeterminado

No se ha definido ningún valor predeterminado.

LoaderDelimiterForAppend

Descripción

La propiedad LoaderDelimiterForAppend especifica si el archivo de datos temporal de Interact es un archivo sin formato con delimitadores o un archivo de ancho fijo y, si está delimitado, el carácter o el juego de caracteres utilizado como delimitadores.

Si el valor no está definido, Interact crea el archivo de datos temporal como un archivo sin formato de ancho fijo.

Si especifica un valor, se utilizará cuando se invoque al cargador para completar una tabla que no se sabe que está vacía. Interact crea el archivo de datos temporal como un archivo sin formato con delimitadores, utilizando el valor de esta propiedad como delimitador.

Esta propiedad no está definida de forma predeterminada.

Valor predeterminado

Valores válidos

Caracteres, que puede escribir entre comillas dobles, si lo desea.

LoaderDelimiterAtEndForAppend

Descripción

Algunas utilidades de carga externas requieren que el archivo de datos esté delimitado y que cada línea finalice con el delimitador. Para satisfacer este requisito, establezca el valor de LoaderDelimiterAtEndForAppend en TRUE, para que cuando se invoque el cargador para completar una tabla que no se sabe que está vacía, Interact utilice delimitadores al final de cada línea.

Valor predeterminado

FALSE

Valores válidos

TRUE | FALSE

LoaderUseLocaleDP

Descripción

La propiedad LoaderUseLocaleDP especifica, cuando Interact escribe valores numéricos en archivos que va a cargar una utilidad de carga de base de datos, si se utiliza el símbolo específico de la configuración regional para el separador decimal.

Establezca este valor en FALSE para especificar que se utilice el punto (.) como separador decimal.

Establezca este valor en TRUE para especificar que e utilice el símbolo de separador decimal correspondiente a la configuración regional.

Valor predeterminado

FALSE

Valores válidos

TRUE | FALSE

Interact | general | testRunDataSource

Estas propiedades de configuración definen la configuración del origen de datos para las tablas de ejecución de prueba del entorno de diseño de Interact. Debe definir este origen de datos para al menos uno de los entornos de ejecución. Estas son las tablas que se utilizan cuando realiza una ejecución de prueba del diagrama de flujo interactivo.

jndiName

Descripción

Utilice esta propiedad `jndiName` para identificar el origen de datos JNDI (Java Naming and Directory Interface) que está definido en el servidor de aplicaciones (WebSphere o WebLogic) para las tablas de clientes a las que accede el entorno de diseño cuando ejecuta las ejecuciones de prueba de diagramas de flujo interactivos.

Valor predeterminado

No se ha definido ningún valor predeterminado.

type

Descripción

El tipo de base de datos para las tablas de clientes a las que accede el entorno de diseño cuando ejecuta las ejecuciones de prueba de diagramas de flujo interactivos.

Valor predeterminado

SQLServer

Valores válidos

SQLServer | DB2 | ORACLE

aliasPrefix

Descripción

La propiedad `AliasPrefix` especifica la manera en que Interact forma el nombre de alias que Interact crea automáticamente cuando utiliza una tabla de dimensiones y escribe en una nueva tabla para las tablas de clientes a las que accede el entorno de diseño cuando ejecuta las ejecuciones de prueba de diagramas de flujo interactivos.

Tenga en cuenta que cada base de datos tiene una longitud de identificador máxima; revise la documentación para la base de datos que está utilizando para asegurarse de que el valor que ha establecido no excede la longitud de identificador máxima para su base de datos.

Valor predeterminado

A

connectionRetryPeriod

Descripción

La propiedad ConnectionRetryPeriod especifica la cantidad de tiempo en segundos que Interact reintenta automáticamente la solicitud de conexión de base de datos anómala para las tablas de ejecución de prueba. Interact intenta reconectarse automáticamente a la base de datos durante este periodo de tiempo antes de informar de un error o una anomalía de base de datos. Si el valor se establece en 0, Interact intentará reconectarse indefinidamente; si el valor se establece en -1, no se realizará ningún intento.

Valor predeterminado

-1

connectionRetryDelay

Descripción

La propiedad ConnectionRetryDelay especifica la cantidad de tiempo en segundos que Interact espera antes de intentar reconectarse a la base de datos después de una anomalía para las tablas de ejecución de prueba. Si el valor se establece en -1, no realiza ningún intento.

Valor predeterminado

-1

schema

Descripción

El nombre del esquema que contiene las tablas para las ejecuciones de prueba del diagrama de flujo interactivo. Interact inserta el valor de esta propiedad antes de todos los nombres de tabla, por ejemplo, UACI_IntChannel pasa a ser schema.UACI_IntChannel.

No tiene que definir un esquema. Si no define un esquema, Interact supone que el propietario de las tablas es el mismo que el del esquema. Debe establecer este valor para eliminar ambigüedades.

Valor predeterminado

No se ha definido ningún valor predeterminado.

Interact | general | contactAndResponseHistoryDataSource

Estas propiedades de configuración definen los valores de conexión para el origen de datos de historial de respuestas y contactos necesario para el seguimiento de respuestas de sesiones cruzadas de Interact.

Estos valores no están relacionados con el módulo de historial de contactos y respuestas.

jndiName

Descripción

Utilice esta propiedad jndiName para identificar el origen de datos JNDI (Java Naming and Directory Interface) que está definido en el servidor de aplicaciones (WebSphere o WebLogic) para el origen de datos de historial de respuestas y contactos necesario para el seguimiento de respuestas de sesiones cruzadas de Interact.

Valor predeterminado

type

Descripción

El tipo de base de datos para el origen de datos utilizado por el origen de datos de historial de respuestas y contactos necesario para el seguimiento de respuestas de sesiones cruzadas de Interact.

Valor predeterminado

SQLServer

Valores válidos

SQLServer | DB2 | ORACLE

connectionRetryPeriod

Descripción

La propiedad `ConnectionRetryPeriod` especifica la cantidad de tiempo en segundos que Interact reintenta automáticamente la solicitud de conexión de base de datos anómala para el seguimiento de respuestas de sesiones cruzadas de Interact.Interact intenta reconectarse automáticamente a la base de datos durante este periodo de tiempo antes de informar de un error o una anomalía de base de datos. Si el valor se establece en 0, Interact intentará reconectarse indefinidamente; si el valor se establece en -1, no se realizará ningún intento.

Valor predeterminado

-1

connectionRetryDelay

Descripción

La propiedad `ConnectionRetryDelay` especifica la cantidad de tiempo en segundos que Interact espera antes de intentar reconectarse a la base de datos después de una anomalía para el seguimiento de respuestas de sesiones cruzadas Interact. Si el valor se establece en -1, no realiza ningún intento.

Valor predeterminado

-1

schema

Descripción

El nombre del esquema que contiene las tablas para el seguimiento de respuestas de sesiones cruzadas de Interact. Interact inserta el valor de esta propiedad antes de todos los nombres de tabla, por ejemplo, `UACI_IntChannel` pasa a ser `schema.UACI_IntChannel`.

No tiene que definir un esquema. Si no define un esquema, Interact supone que el propietario de las tablas es el mismo que el del esquema. Debe establecer este valor para eliminar ambigüedades.

Valor predeterminado

No se ha definido ningún valor predeterminado.

Interact | general | idsByType

Estas propiedades de configuración definen los valores de los números de ID utilizados por el módulo de historial de contactos y respuestas.

initialValue

Descripción

El valor de ID inicial utilizado cuando se generan ID utilizando la tabla UACI_IDsByType.

Valor predeterminado

1

Valores válidos

Un valor mayor que 0.

retries

Descripción

El número de reintentos antes de producir una excepción cuando se generan ID utilizando la tabla UACI_IDsByType.

Valor predeterminado

20

Valores válidos

Un entero mayor que 0.

Interact | flowchart

En esta sección se definen los valores de configuración de los diagramas de flujo interactivos.

defaultDateFormat

Descripción

El formato de fecha predeterminado utilizado por Interact para convertir la fecha en una cadena de caracteres y la cadena de caracteres en una fecha.

Valor predeterminado

MM/dd/yy

idleFlowchartThreadTimeoutInMinutes

Descripción

El número de minutos que Interact permite que un subproceso dedicado a un diagrama de flujo interactivo esté desocupado antes de liberar el subproceso.

Valor predeterminado

5

idleProcessBoxThreadTimeoutInMinutes

Descripción

El número de minutos que Interact permite que un subproceso dedicado a un proceso de diagrama de flujo interactivo esté desocupado antes de liberar el subproceso.

Valor predeterminado

5

maxSizeOfFlowchartEngineInboundQueue

Descripción

El número máximo de solicitudes de ejecución de diagrama de flujo que Interact mantiene en cola. Si se alcanza este número de solicitudes, Interact dejará de aceptar solicitudes.

Valor predeterminado

1000

maxNumberOfFlowchartThreads

Descripción

El número máximo de subprocesos dedicados a solicitudes de diagramas de flujo interactivos.

Valor predeterminado

25

maxNumberOfProcessBoxThreads

Descripción

El número máximo de subprocesos dedicados a procesos de diagramas de flujo interactivos.

Valor predeterminado

50

maxNumberOfProcessBoxThreadsPerFlowchart

Descripción

El número máximo de subprocesos dedicados a procesos de diagramas de flujo interactivos por instancia de diagrama de flujo.

Valor predeterminado

3

minNumberOfFlowchartThreads

Descripción

El número mínimo de subprocesos dedicados a solicitudes de diagramas de flujo interactivos.

Valor predeterminado

10

minNumberOfProcessBoxThreads

Descripción

El número mínimo de subprocesos dedicados a procesos de diagramas de flujo interactivos.

Valor predeterminado

20

sessionVarPrefix

Descripción

El prefijo de las variables de sesión.

Valor predeterminado

SessionVar

Interact | flowchart | ExternalCallouts | [ExternalCalloutName]

En esta sección se definen los valores de clase para las llamadas externas personalizadas que ha escrito con la API de llamadas externas.

class

Descripción

El nombre de la clase Java representada por esta llamada externa.

Es la clase Java a la que puede acceder con la macro de IBM Unica `EXTERNALCALLOUT`.

Valor predeterminado

No se ha definido ningún valor predeterminado.

classpath

Descripción

La ruta de clase de la clase Java representada por esta llamada externa. La ruta de clase debe hacer referencia a los archivos jar en el servidor del entorno de ejecución. Si utiliza un grupo de servidores y todos los servidores de ejecución utilizan la misma Marketing Platform, cada servidor debe tener una copia del archivo jar en la misma ubicación. La ruta de clase debe estar formada por las ubicaciones absolutas de los archivos jar, separadas por el delimitador de ruta del sistema operativo del servidor del entorno de ejecución, por ejemplo, un punto y coma (;) en Windows y dos puntos (:) en los sistemas UNIX. Los directorios que contienen archivos de clase no se aceptan. Por ejemplo, en un sistema UNIX: `/path1/file1.jar:/path2/file2.jar`.

Esta ruta de clase debe tener menos de 1024 caracteres. Puede utilizar el archivo de manifiesto en un archivo .jar para especificar otros archivos .jar para que sólo tenga que aparecer un archivo .jar en su ruta de clase.

Es la clase Java a la que puede acceder con la macro de IBM Unica `EXTERNALCALLOUT`.

Valor predeterminado

No se ha definido ningún valor predeterminado.

Interact | flowchart | ExternalCallouts | [ExternalCalloutName] | Parameter Data | [parameterName]

En esta sección se definen los valores de parámetro para una llamadas externa personalizada que ha escrito con la API de llamadas externas.

value

Descripción

El valor de un parámetro necesario para la clase para la llamada externa.

Valor predeterminado

No se ha definido ningún valor predeterminado.

Ejemplo

Si la llamada externa requiere el nombre de host de un servidor externo, debe crear una categoría de parámetro denominada `host` y definir la propiedad `value` como el nombre del servidor.

Interact | monitoring

Este conjunto de propiedades de configuración permite definir los valores de supervisión JMX. Sólo debe configurar estas propiedades si está utilizando la supervisión JMX.

Existen otras propiedades de supervisión JMX que se deben definir para el módulo del historial de contactos y respuestas en las propiedades de configuración del entorno de diseño de Interact.

protocol

Descripción

Define el protocolo del servicio de mensajería de Interact.

Si elige JMXMP, debe incluir los siguientes archivos JAR en su ruta de clase en orden:

`Interact/lib/InteractJMX.jar;Interact/lib/jmxremote_optional.jar`

Valor predeterminado

JMXMP

Valores válidos

JMXMP | RMI

port

Descripción

El número de puerto del servicio de mensajería.

Valor predeterminado

9998

enableSecurity

Descripción

Un booleano que habilita o inhabilita la seguridad del servicio de mensajería JMXMP para el servidor de ejecución de Interact. Si se establece en `true`, debe proporcionar un nombre de usuario y una contraseña para acceder al servicio JMX de tiempo de ejecución de Interact. Esta credencial de usuario se autentica mediante la Marketing Platform del servidor de ejecución. Jconsole no permite el inicio de sesión con contraseña vacía.

Esta propiedad no tiene ningún efecto si el protocolo es RMI. Esta propiedad no tiene ningún efecto en JMX para Campaign (el tiempo de diseño de Interact).

Valor predeterminado

True

Valores válidos

True | False

Interact | profile

Este conjunto de propiedades de configuración controla varias de las características de presentación de ofertas opcionales, incluida la supresión de ofertas y la alteración temporal de puntuaciones.

enableScoreOverrideLookup

Descripción

Si se establece en `True`, Interact carga los datos de alteración temporal de puntuaciones desde `scoreOverrideTable` al crear una sesión. Si se establece en `False`, Interact no carga los datos de alteración temporal de puntuaciones de marketing cuando se crea una sesión.

Si es true, también debe configurar la propiedad `Unica > Interact > profile > Audience Levels > (Audience Level) > scoreOverrideTable`. Sólo debe definir la propiedad `scoreOverrideTable` para los niveles de audiencia que necesite. Si deja `scoreOverrideTable` en blanco para un nivel de audiencia, se inhabilita la tabla de alteración temporal de puntuaciones para el nivel de audiencia.

Valor predeterminado

`False`

Valores válidos

True | False

enableOfferSuppressionLookup

Descripción

Si se establece en `True`, Interact carga los datos de supresión de ofertas desde `offerSuppressionTable` al crear una sesión. Si se establece en `False`, Interact no carga los datos de supresión de ofertas cuando se crea una sesión.

Si es true, también debe configurar la propiedad `Unica > Interact > profile > Audience Levels > (Audience Level) > offerSuppressionTable`. Sólo debe definir la propiedad `enableOfferSuppressionLookup` para los niveles de audiencia que necesite.

Valor predeterminado

`False`

Valores válidos

True | False

enableProfileLookup

Descripción

En una nueva instalación de Interact, esta propiedad está en desuso. En una instalación actualizada de Interact, esta propiedad es válida hasta el primer despliegue.

El comportamiento de carga para una tabla utilizada en un diagrama de flujo interactivo pero no correlacionada en el canal interactivo. Si se establece en `True`, Interact carga los datos del perfil desde `profileTable` al crear una sesión.

Si es true, también debe configurar la propiedad `Unica > Interact > profile > Audience Levels > (Audience Level) > profileTable`.

El valor **Cargar estos datos en la memoria cuando comience una sesión de visita** en el asistente de correlación de tablas de canal interactivo altera temporalmente esta propiedad de configuración.

Valor predeterminado

`False`

Valores válidos

True | False

defaultOfferUpdatePollPeriod

Descripción

El número de segundos que espera el sistema antes de actualizar las ofertas predeterminadas en la memoria caché desde la tabla de ofertas predeterminadas. Si se establece en -1, el sistema no actualiza las ofertas predeterminadas en la memoria caché después de cargar la lista inicial en la memoria caché cuando se inicia el servidor de ejecución.

Valor predeterminado

-1

Interact | profile | Audience Levels | [AudienceLevelName]

Este conjunto de propiedades de configuración permite definir los nombres de tabla necesarios para las características adicionales de Interact. Sólo tiene que definir el nombre de tabla si está utilizando la característica asociada.

scoreOverrideTable

Descripción

El nombre de la tabla que contiene la información de alteración temporal de puntuaciones para este nivel de audiencia. Esta propiedad sólo es aplicable si establece `enableScoreOverrideLookup` en true. Debe definir esta propiedad para los niveles de audiencia para los que desea habilitar una tabla de alteración temporal de puntuaciones. Si no tiene una tabla de

alteración temporal de puntuaciones para este nivel de audiencia, puede dejar esta propiedad sin definir, aunque `enableScoreOverrideLookup` se haya establecido en true.

Interact busca esta tabla en las tablas del cliente a las que acceden los servidores de ejecución de Interact, definidos por las propiedades `prodUserDataSource`.

Si ha definido la propiedad `schema` para este origen de datos, Interact añade el esquema delante de este nombre de tabla, por ejemplo, `schema.UACI_ScoreOverride`. Si especifica un nombre completo, por ejemplo, `mySchema.UACI_ScoreOverride`, Interact no añade el nombre de esquema.

Valor predeterminado

UACI_ScoreOverride

offerSuppressionTable

Descripción

El nombre de la tabla que contiene la información de supresión de ofertas para este nivel de audiencia. Debe definir esta propiedad para los niveles de audiencia para los que desea habilitar una tabla de supresión de ofertas. Si no tiene una tabla de supresión de ofertas para este nivel de audiencia, puede dejar esta propiedad sin definir, aunque `enableOfferSuppressionLookup` se haya establecido en true.

Interact busca esta tabla en las tablas del cliente a las que acceden los servidores de ejecución, definidos por las propiedades `prodUserDataSource`.

Valor predeterminado

UACI_BlackList

profileTable

Descripción

En una nueva instalación de Interact, esta propiedad está en desuso. En una instalación actualizada de Interact, esta propiedad es válida hasta el primer despliegue.

El nombre de la tabla que contiene los datos del perfil para este nivel de audiencia.

Interact busca esta tabla en las tablas del cliente a las que acceden los servidores de ejecución, definidos por las propiedades `prodUserDataSource`.

Si ha definido la propiedad `schema` para este origen de datos, Interact añade el esquema delante de este nombre de tabla, por ejemplo, `schema.UACI_usrProd`. Si especifica un nombre completo, por ejemplo, `mySchema.UACI_usrProd`, Interact no añade el nombre de esquema.

Valor predeterminado

No se ha definido ningún valor predeterminado.

contactHistoryTable

Descripción

El nombre de la tabla de preparación de los datos del historial de contactos para este nivel de audiencia.

Esta tabla se almacena en las tablas del entorno de ejecución (systemTablesDataSource).

Si ha definido la propiedad schema para este origen de datos, Interact añade el esquema delante de este nombre de tabla, por ejemplo, schema.UACI_CHStaging. Si especifica un nombre completo, por ejemplo, mySchema.UACI_CHStaging, Interact no añade el nombre de esquema.

Valor predeterminado

UACI_CHStaging

chOfferAttribTable

Descripción

El nombre de la tabla de atributos de oferta del historial de contactos para este nivel de audiencia.

Esta tabla se almacena en las tablas del entorno de ejecución (systemTablesDataSource).

Si ha definido la propiedad schema para este origen de datos, Interact añade el esquema delante de este nombre de tabla, por ejemplo, schema.UACI_CHOfferAttrib. Si especifica un nombre completo, por ejemplo, mySchema.UACI_CHOfferAttrib, Interact no añade el nombre de esquema.

Valor predeterminado

UACI_CHOfferAttrib

responseHistoryTable

Descripción

El nombre de la tabla de preparación del historial de respuestas para este nivel de audiencia.

Esta tabla se almacena en las tablas del entorno de ejecución (systemTablesDataSource).

Si ha definido la propiedad schema para este origen de datos, Interact añade el esquema delante de este nombre de tabla, por ejemplo, schema.UACI_RHStaging. Si especifica un nombre completo, por ejemplo, mySchema.UACI_RHStaging, Interact no añade el nombre de esquema.

Valor predeterminado

UACI_RHStaging

crossSessionResponseTable

Descripción

El nombre de la tabla para este nivel de audiencia que se necesita para el seguimiento de respuestas de sesiones cruzadas en las tablas del historial de contactos y respuestas a las que puede acceder la característica de seguimiento de respuestas.

Si ha definido la propiedad schema para este origen de datos, Interact añade el esquema delante de este nombre de tabla, por ejemplo, schema.UACI_XSessResponse. Si especifica un nombre completo, por ejemplo, mySchema.UACI_XSessResponse, Interact no añade el nombre de esquema.

Valor predeterminado

UACI_XSessResponse

Interact | profile | Audience Levels | [AudienceLevelName] | Offers by Raw SQL

Este conjunto de propiedades de configuración permite definir los nombres de tabla necesarios para las características adicionales de Interact. Sólo tiene que definir el nombre de tabla si está utilizando la característica asociada.

enableOffersByRawSQL

Descripción

Si se establece en True, Interact habilita la característica offersBySQL para este nivel de audiencia, que permite configurar el código SQL que se debe ejecutar para crear el conjunto deseado de ofertas candidatas en el tiempo de ejecución. Si es False, Interact no utiliza la característica offersBySQL.

Si establece esta propiedad en true, también puede configurar la propiedad Unica | Interact | profile | Audience Levels | (Audience Level) | Offers by Raw SQL | SQL Template para definir una o varias plantillas SQL.

Valor predeterminado

False

Valores válidos

True | False

cacheSize

Descripción

Tamaño de la memoria caché utilizada para almacenar los resultados de las consultas de OfferBySQL. Tenga en cuenta que la utilización de una memoria caché puede tener un impacto negativo si los resultados de consulta son exclusivos para la mayoría de las sesiones.

Valor predeterminado

-1 (off)

Valores válidos

-1 | Value

cacheLifeInMinutes

Descripción

Si la memoria caché está habilitada, indica el número de minutos antes de que el sistema borre la memoria caché para evitar la obsolescencia.

Valor predeterminado

-1 (off)

Valores válidos

-1 | Value

defaultSQLTemplate

Descripción

El nombre de la plantilla SQL que se utiliza si no se ha especificado una mediante las llamadas de API.

Valor predeterminado

`Ninguno`

Valores válidos

Nombre de plantilla SQL

Interact | profile | Audience Levels | [AudienceLevelName] | SQL Template

Estas propiedades de configuración permiten definir una o varias plantillas de consulta SQL utilizadas por la característica offersBySQL de Interact.

name

Descripción

El nombre que desea asignar a esta plantilla de consulta SQL. Escriba un nombre descriptivo que sea significativo cuando utilice esta plantilla SQL en las llamadas de API. Tenga en cuenta que si utiliza un nombre aquí que es *idéntico* a un nombre definido en el cuadro de proceso Lista de interacción para un tratamiento offerBySQL, se utilizará el SQL del cuadro de proceso en lugar del SQL que especifique aquí.

Valor predeterminado

Ninguno

SQL

Descripción

Contiene la consulta SQL que invocará esta plantilla. La consulta SQL puede contener referencias a nombres de variable que forman parte de los datos de sesión del visitante (perfil). Por ejemplo, `select * from MyOffers where category = ${preferredCategory}` se basará en la sesión que contiene una variable denominada `preferredCategory`.

Debe configurar el SQL para consultar las tablas de oferta específicas que ha creado durante el tiempo de diseño para utilizarlas con esta característica. Tenga en cuenta que los procedimientos almacenados no están soportados aquí.

Valor predeterminado

Ninguno

Interact | profile | Audience Levels | [AudienceLevelName | Profile Data Services | [DataSource]

Este conjunto de propiedades de configuración permite definir los nombres de tabla necesarios para las características adicionales de Interact. Sólo tiene que definir el nombre de tabla si está utilizando la característica asociada. La categoría Profile Data Services proporciona información acerca de un origen de datos incorporado (denominado Database) que se crea para todos los niveles de audiencia, y que está configurado previamente con una prioridad de 100. Pero puede modificar o inhabilitar este valor. Esta categoría también contiene una

plantilla para orígenes de datos externos adicionales. Cuando pulsa la plantilla denominada **External Data Services**, puede completar los valores de configuración que se describen aquí.

Nuevo nombre de categoría

Descripción

(No está disponible para la entrada Database predeterminada). Nombre del origen de datos que está definiendo. El nombre que especifique aquí debe ser exclusivo entre los orígenes de datos para un mismo nivel de audiencia.

Valor predeterminado

Ninguno

Valores válidos

Se puede utilizar una cadena de texto cualquiera.

enabled

Descripción

Si el valor se establece en `True`, Interact, este origen de datos está habilitado para el nivel de audiencia al que está asignado. Si el valor es `False`, Interact no utiliza este origen de datos para este nivel de audiencia.

Valor predeterminado

`True`

Valores válidos

True | False

className

Descripción

(No está disponible para la entrada Database predeterminada). Nombre completo de la clase de origen de datos que implementa `IInteractProfileDataService`.

Valor predeterminado

Ninguno.

Valores válidos

Nombre de clase completo en forma de cadena de caracteres.

classPath

Descripción

(No está disponible para la entrada Database predeterminada). Valor de configuración opcional que proporciona la vía de acceso para cargar esta clase de implementación del origen de datos. Si no especifica un valor, se utiliza por omisión la vía de acceso de clases del servidor de aplicaciones

Valor predeterminado

No se muestra, pero por omisión se utiliza la vía de acceso de clases del servidor de aplicaciones si no se proporciona ningún valor aquí.

Valores válidos

Vía de acceso de clases en forma de cadena de caracteres.

priority

Descripción

Prioridad del origen de datos dentro de este nivel de audiencia. Debe ser un valor exclusivo entre todos los orígenes de datos para cada nivel de audiencia. (Es decir, si la prioridad se establece en 100 para un origen de datos, ningún otro origen de datos dentro del nivel de audiencia puede tener una prioridad de 100.)

Valor predeterminado

`100` para el origen de datos predeterminado Database, `200` para un origen de datos definido por el usuario

Valores válidos

Se puede utilizar cualquier número entero no negativo.

Interact | offerserving

Estas propiedades de configuración definen las propiedades de configuración del aprendizaje genérico.

Si utiliza el aprendizaje incorporado, para ajustar la implementación de aprendizaje, utilice las propiedades de configuración del entorno de diseño.

optimizationType

Descripción

La propiedad `optimizationType` define si Interact utiliza un motor de aprendizaje como ayuda en las asignaciones de ofertas. Si se establece en `NoLearning`, Interact no utiliza el aprendizaje. Si se establece en `BuiltInLearning`, Interact utiliza el motor de aprendizaje baysean que se incorpora con Interact. Si se establece en `ExternalLearning`, Interact utiliza el motor de aprendizaje que proporcione. Si selecciona `ExternalLearning`, debe definir las propiedades `externalLearningClass` y `externalLearningClassPath`.

Valor predeterminado

NoLearning

Valores válidos

NoLearning | BuiltInLearning | ExternalLearning

segmentationMaxWaitTimeInMS

Descripción

El número máximo de milisegundos que el servidor de ejecución espera a que se complete un diagrama de flujo interactivo antes de recibir ofertas.

Valor predeterminado

`5000`

treatmentCodePrefix

Descripción

El prefijo que se añade a los códigos de tratamiento.

Valor predeterminado

No se ha definido ningún valor predeterminado.

Interact | offerserving | Built-in Learning Config

Estas propiedades de configuración definen la configuración de escritura de base de datos para el aprendizaje incorporado.

Para ajustar la implementación de aprendizaje, utilice las propiedades de configuración del entorno de diseño.

insertRawStatsIntervalInMinutes

Descripción

El número de minutos que el módulo de aprendizaje de Interact espera antes de insertar más filas en las tablas de preparación de aprendizaje. Deberá modificar este tiempo según la cantidad de datos que el módulo de aprendizaje esté procesando en el entorno.

Valor predeterminado

5

aggregateStatsIntervalInMinutes

Descripción

El número de minutos que el módulo de aprendizaje de Interact espera entre la agregación de datos en las tablas de preparación de aprendizaje. Deberá modificar este tiempo según la cantidad de datos que el módulo de aprendizaje esté procesando en el entorno.

Valor predeterminado

15

Valores válidos

Un entero mayor que cero.

Interact | offerserving | External Learning Config

Estas propiedades de configuración definen los valores de clase para un módulo de aprendizaje externo que ha escrito utilizando la API de aprendizaje.

class

Descripción

Si `optimizationType` se establece en `ExternalLearning`, establezca `externalLearningClass` en el nombre de clase del motor de aprendizaje externo.

Valor predeterminado

No se ha definido ningún valor predeterminado.

Disponibilidad

Esta propiedad sólo es aplicable si `optimizationType` se ha establecido en `ExternalLearning`.

classPath

Descripción

Si `optimizationType` se establece en `ExternalLearning`, establezca `externalLearningClass` en la ruta de clase del motor de aprendizaje externo.

La ruta de clase debe hacer referencia a los archivos jar en el servidor del entorno de ejecución. Si utiliza un grupo de servidores y todos los servidores de ejecución utilizan la misma Marketing Platform, cada servidor debe tener una copia del archivo jar en la misma ubicación. La ruta de clase debe estar formada por las ubicaciones absolutas de los archivos jar, separadas por el delimitador de ruta del sistema operativo del servidor del entorno de ejecución, por ejemplo, un punto y coma (;) en Windows y dos puntos (:) en los sistemas UNIX. Los directorios que contienen archivos de clase no se aceptan. Por ejemplo, en un sistema UNIX: `/path1/file1.jar:/path2/file2.jar`.

Esta ruta de clase debe tener menos de 1024 caracteres. Puede utilizar el archivo de manifiesto en un archivo .jar para especificar otros archivos .jar para que sólo tenga que aparecer un archivo .jar en su ruta de clase.

Valor predeterminado

No se ha definido ningún valor predeterminado.

Disponibilidad

Esta propiedad sólo es aplicable si `optimizationType` se ha establecido en `ExternalLearning`.

Interact | offerserving | External Learning Config | Parameter Data | [parameterName]

Estas propiedades de configuración definen los parámetros del módulo de aprendizaje externo.

value

Descripción

El valor de un parámetro necesario para la clase para un módulo de aprendizaje externo.

Valor predeterminado

No se ha definido ningún valor predeterminado.

Ejemplo

Si el módulo de aprendizaje externo requiere una ruta a una aplicación de resolución de algoritmos, debe crear una categoría de parámetro denominada `solverPath` y definir la propiedad `value` como la ruta a la aplicación.

Interact | services

Las propiedades de configuración de esta categoría definen los valores de todos los servicios que gestionan la recopilación de datos y estadísticas del historial de contactos y respuestas para informar y escribir en las tablas del sistema del entorno de ejecución.

externalLoaderStagingDirectory

Descripción

Esta propiedad define la ubicación del directorio de preparación para una utilidad de carga de base de datos.

Valor predeterminado

No se ha definido ningún valor predeterminado.

Valores válidos

Una ruta relativa al directorio de instalación de Interact o una ruta absoluta a un directorio de preparación.

Si habilita una utilidad de carga de base de datos, debe establecer la propiedad `cacheType` de las categorías `contactHist` y `responstHist` en `External Loader File`.

Interact | services | contactHist

Las propiedades de configuración de esta categoría definen los valores del servicio que recopila datos para las tablas de preparación del historial de contactos.

enableLog

Descripción

Si es `true`, habilita el servicio que recopila datos para registrar los datos del historial de contactos. Si es `false`, no se recopilan datos.

Valor predeterminado

True

Valores válidos

True | False

cacheType

Descripción

Define si los datos recopilados para el historial de contactos se mantienen en la memoria (`Memory Cache`) o en un archivo (`External Loader File`). Sólo puede utilizar `External Loader File` si ha configurado Interact para utilizar una utilidad de carga de base de datos.

Si selecciona `Memory Cache`, utilice la configuración de categoría `cache`. Si selecciona `External Loader File`, utilice la configuración de categoría `fileCache`.

Valor predeterminado

Memory Cache

Valores válidos

Memory Cache | External Loader File

Interact | services | contactHist | cache

Las propiedades de configuración de esta categoría definen la configuración de la memoria caché del servicio que recopila datos para la tabla de preparación del historial de contactos.

threshold

Descripción

El número de registros acumulados antes de que el servicio flushCacheToDB escriba los datos recopilados del historial de contactos en la base de datos.

Valor predeterminado

100

insertPeriodInSecs

Descripción

El número de segundos entre las escrituras forzadas en la base de datos.

Valor predeterminado

3600

Interact | services | contactHist | fileCache

Las propiedades de configuración de esta categoría definen la configuración de la memoria caché del servicio que recopila los datos del historial de contactos si utiliza una utilidad de carga de base de datos.

threshold

Descripción

El número de registros acumulados antes de que el servicio flushCacheToDB escriba los datos recopilados del historial de contactos en la base de datos.

Valor predeterminado

100

insertPeriodInSecs

Descripción

El número de segundos entre las escrituras forzadas en la base de datos.

Valor predeterminado

3600

Interact | services | defaultedStats

Las propiedades de configuración de esta categoría definen los valores del servicio que recopila las estadísticas sobre el número de veces que se ha utilizado la cadena predeterminada para el punto de interacción.

enableLog

Descripción

Si es `true`, habilita el servicio que recopila las estadísticas sobre el número de veces que se ha utilizado la cadena predeterminada para el punto de interacción en la tabla `UACI_DefaultedStat`. Si es `false`, no se recopilan estadísticas de cadenas predeterminadas.

Si no utiliza los informes de IBM, puede establecer esta propiedad en `false`, ya que la colección de datos no es necesaria.

Valor predeterminado

True

Valores válidos

True | False

Interact | services | defaultedStats | cache

Las propiedades de configuración de esta categoría definen la configuración de la memoria caché del servicio que recopila las estadísticas sobre el número de veces que se ha utilizado la cadena predeterminada para el punto de interacción.

threshold

Descripción

El número de registros acumulados antes de que el servicio flushCacheToDB escriba las estadísticas recopiladas de cadenas predeterminadas en la base de datos.

Valor predeterminado

100

insertPeriodInSecs

Descripción

El número de segundos entre las escrituras forzadas en la base de datos.

Valor predeterminado

3600

Interact | services | eligOpsStats

Las propiedades de configuración de esta categoría definen los valores del servicio que escribe las estadísticas de las ofertas elegibles.

enableLog

Descripción

Si es `true`, habilita el servicio que recopila las estadísticas de las ofertas elegibles. Si es `false`, no se recopilan estadísticas de ofertas elegibles.

Si no utiliza los informes de IBM, puede establecer esta propiedad en `false`, ya que la colección de datos no es necesaria.

Valor predeterminado

True

Valores válidos

True | False

Interact | services | eligOpsStats | cache

Las propiedades de configuración de esta categoría definen la configuración de la memoria caché del servicio que recopila las estadísticas de ofertas elegibles.

threshold

Descripción

El número de registros acumulados antes de que el servicio flushCacheToDB escriba las estadísticas recopiladas de ofertas elegibles en la base de datos.

Valor predeterminado

100

insertPeriodInSecs

Descripción

El número de segundos entre las escrituras forzadas en la base de datos.

Valor predeterminado

3600

Interact | services | eventActivity

Las propiedades de configuración de esta categoría definen los valores del servicio que recopila las estadísticas de actividades de eventos.

enableLog

Descripción

Si es `true`, habilita el servicio que recopila las estadísticas de actividades de eventos. Si es `false`, no se recopilan estadísticas de eventos.

Si no utiliza los informes de IBM, puede establecer esta propiedad en `false`, ya que la colección de datos no es necesaria.

Valor predeterminado

True

Valores válidos

True | False

Interact | services | eventActivity | cache

Las propiedades de configuración de esta categoría definen la configuración de la memoria caché del servicio que recopila las estadísticas de actividades de eventos.

threshold

Descripción

El número de registros acumulados antes de que el servicio flushCacheToDB escriba las estadísticas recopiladas de actividades de eventos en la base de datos.

Valor predeterminado

100

insertPeriodInSecs

Descripción

El número de segundos entre las escrituras forzadas en la base de datos.

Valor predeterminado

3600

Interact | services | customLogger

Las propiedades de configuración de esta categoría definen los valores del servicio que recopila datos personalizados para escribir en una tabla (un evento que utiliza el parámetro de evento `UACICustomLoggerTableName`).

enableLog

Descripción

Si es `true`, habilita la característica de registro personalizado en la tabla. Si es `false`, el parámetro de evento `UACICustomLoggerTableName` no tiene ningún efecto.

Valor predeterminado

True

Valores válidos

True | False

Interact | services | customLogger | cache

Las propiedades de configuración de esta categoría definen la configuración de la memoria caché del servicio que recopila datos personalizados en una tabla (un evento que utiliza el parámetro de evento `UACICustomLoggerTableName`).

threshold

Descripción

El número de registros acumulados antes de que el servicio flushCacheToDB escriba los datos personalizados recopilados en la base de datos.

Valor predeterminado

100

insertPeriodInSecs

Descripción

El número de segundos entre las escrituras forzadas en la base de datos.

Valor predeterminado

3600

Interact | services | responseHist

Las propiedades de configuración de esta categoría definen los valores del servicio que escribe en las tablas de preparación del historial de respuestas.

enableLog

Descripción

Si es `true`, habilita el servicio que escribe en las tablas de preparación del historial de respuestas. Si es `false`, no se escriben datos en las tablas de preparación del historial de respuestas.

La tabla de preparación del historial de respuestas está definida por la propiedad `responseHistoryTable` del nivel de audiencia. El valor predeterminado es `UACI_RHStaging`.

Valor predeterminado

True

Valores válidos

True | False

cacheType

Descripción

Define si la memoria caché se mantiene en la memoria o en un archivo. Sólo puede utilizar `External Loader File` si ha configurado Interact para utilizar una utilidad de carga de base de datos.

Si selecciona `Memory Cache`, utilice la configuración de categoría `cache`. Si selecciona `External Loader File`, utilice la configuración de categoría `fileCache`.

Valor predeterminado

Memoria caché

Valores válidos

Memory Cache | External Loader File

Interact | services | responseHist | cache

Las propiedades de configuración de esta categoría definen la configuración de la memoria caché del servicio que recopila los datos del historial de respuestas.

threshold

Descripción

El número de registros acumulados antes de que el servicio flushCacheToDB escriba los datos recopilados del historial de respuestas en la base de datos.

Valor predeterminado

100

insertPeriodInSecs

Descripción

El número de segundos entre las escrituras forzadas en la base de datos.

Valor predeterminado

3600

Interact | services | responseHist | fileCache

Las propiedades de configuración de esta categoría definen la configuración de la memoria caché del servicio que recopila los datos del historial de respuestas si utiliza una utilidad de carga de base de datos.

threshold

Descripción

El número de registros acumulados antes de que Interact los escriba en la base de datos.

`responseHist` - La tabla definida por la propiedad `responseHistoryTable` del nivel de audiencia. El valor predeterminado es `UACI_RHStaging`.

Valor predeterminado

100

insertPeriodInSecs

Descripción

El número de segundos entre las escrituras forzadas en la base de datos.

Valor predeterminado

3600

Interact | services | crossSessionResponse

Las propiedades de configuración de esta categoría definen los valores generales para el servicio crossSessionResponse y el proceso xsession. Sólo necesita configurar estos valores si está utilizando el seguimiento de respuestas de sesiones cruzadas de Interact.

enableLog

Descripción

Si es `true`, habilita el servicio crossSessionResponse y Interact escribe datos en las tablas de preparación de seguimiento de respuestas de sesiones cruzadas. Si se establece en `false`, inhabilita el servicio crossSessionResponse.

Valor predeterminado

False

xsessionProcessIntervalInSecs

Descripción

El número de segundos entre las ejecuciones del proceso xsession. Este proceso mueve los datos de las tablas de preparación de seguimiento de respuestas de sesiones cruzadas a la tabla de preparación del historial de respuestas y el módulo de aprendizaje incorporado.

Valor predeterminado

180

Valores válidos

Un entero mayor que cero

purgeOrphanResponseThresholdInMinutes

Descripción

El número de minutos que espera el servicio crossSessionResponse antes de marcar las respuestas que no coinciden con los contactos en las tablas del historial de contactos y respuestas.

Si una respuesta no tiene ninguna coincidencia en las tablas de historial de contactos y respuestas después de `purgeOrphanResponseThresholdInMinutes` minutos, Interact marca la respuesta con un valor -1 en la columna `Mark` de la tabla de preparación xSessResponse. A continuación, puede hacer coincidir o suprimir manualmente estas respuestas.

Valor predeterminado

180

Interact | services | crossSessionResponse | cache

Las propiedades de configuración de esta categoría definen la configuración de la memoria caché del servicio que recopila los datos de respuestas de sesiones cruzadas.

threshold

Descripción

El número de registros acumulados antes de que el servicio flushCacheToDB escriba los datos de respuestas de sesiones cruzadas recopilados en la base de datos.

Valor predeterminado

100

insertPeriodInSecs

Descripción

El número de segundos entre las escrituras forzadas en la tabla `XSessResponse`.

Valor predeterminado

3600

Interact | services | crossSessionResponse | OverridePerAudience | [AudienceLevel] | TrackingCodes | byTreatmentCode

Las propiedades de esta sección definen cómo el seguimiento de respuestas de sesiones cruzadas hace coincidir los códigos de tratamiento con el historial de contactos y respuestas.

SQL

Descripción

Esta propiedad define si Interact utiliza el SQL generado por el sistema o el SQL personalizado definido en la propiedad `OverrideSQL`.

Valor predeterminado

Utilizar SQL generado por el sistema

Valores válidos

Utilizar SQL generado por el sistema | Alterar temporalmente SQL

OverrideSQL

Descripción

Si no utiliza el comando SQL predeterminado para hacer coincidir el código de tratamiento con el historial de contactos y respuestas, especifique aquí el SQL o el procedimiento almacenado.

Este valor se ignora si `SQL` se establece en `Utilizar SQL generado por el sistema`.

Valor predeterminado

useStoredProcedure

Descripción

Si se establece en true, `OverrideSQL` debe incluir una referencia a un procedimiento almacenado que haga coincidir el código de tratamiento con el historial de contactos y respuestas.

Si se establece en false, `OverrideSQL`, cuando se utiliza, debe ser una consulta SQL.

Valor predeterminado

false

Valores válidos

true | false

Type

Descripción

El TrackingCodeType asociado definido en la tabla UACI_TrackingType en las tablas del entorno de ejecución. A menos que revise la tabla UACI_TrackingType, Type debe ser 1.

Valor predeterminado

1

Valores válidos

Un entero definido en la tabla UACI_TrackingType.

Interact | services | crossSessionResponse | OverridePerAudience | [AudienceLevel] | TrackingCodes | byOfferCode

Las propiedades de esta sección definen cómo el seguimiento de respuestas de sesiones cruzadas hace coincidir los códigos de oferta con el historial de contactos y respuestas.

SQL

Descripción

Esta propiedad define si Interact utiliza el SQL generado por el sistema o el SQL personalizado definido en la propiedad OverrideSQL.

Valor predeterminado

Utilizar SQL generado por el sistema

Valores válidos

Utilizar SQL generado por el sistema | Alterar temporalmente SQL

OverrideSQL

Descripción

Si no utiliza el comando SQL predeterminado para hacer coincidir el código de oferta con el historial de contactos y respuestas, especifique aquí el SQL o el procedimiento almacenado.

Este valor se ignora si SQL se establece en Utilizar SQL generado por el sistema.

Valor predeterminado

useStoredProcedure

Descripción

Si se establece en true, OverrideSQL debe incluir una referencia a un procedimiento almacenado que haga coincidir el código de oferta con el historial de contactos y respuestas.

Si se establece en false, OverrideSQL, cuando se utiliza, debe ser una consulta SQL.

Valor predeterminado

false

Valores válidos

true | false

Type

Descripción

El TrackingCodeType asociado definido en la tabla UACI_TrackingType en las tablas del entorno de ejecución. A menos que revise la tabla UACI_TrackingType, Type debe ser 2.

Valor predeterminado

2

Valores válidos

Un entero definido en la tabla UACI_TrackingType.

Interact | services | crossSessionResponse | OverridePerAudience | [AudienceLevel] | TrackingCodes | byAlternateCode

Las propiedades de esta sección definen cómo el seguimiento de respuestas de sesiones cruzadas hace coincidir un código alternativo definido por el usuario con el historial de contactos y respuestas.

Name

Descripción

Esta propiedad define el nombre del código alternativo. Debe coincidir con el valor de Name en la tabla UACI_TrackingType en las tablas del entorno de ejecución.

Valor predeterminado

OverrideSQL

Descripción

El comando SQL o el procedimiento almacenado que hace coincidir el código alternativo con el historial de contactos y respuestas por código de oferta o código de tratamiento.

Valor predeterminado

useStoredProcedure

Descripción

Si se establece en true, OverrideSQL debe incluir una referencia a un procedimiento almacenado que haga coincidir el código alternativo con el historial de contactos y respuestas.

Si se establece en false, OverrideSQL, cuando se utiliza, debe ser una consulta SQL.

Valor predeterminado

false

Valores válidos

true | false

Type

Descripción

El TrackingCodeType asociado definido en la tabla UACI_TrackingType en las tablas del entorno de ejecución.

Valor predeterminado

3

Valores válidos

Un entero definido en la tabla UACI_TrackingType.

Interact | services | threadManagement | contactAndResponseHist

Las propiedades de configuración de esta categoría definen los valores de gestión de subprocesos para los servicios que recopilan datos para las tablas de preparación del historial de contactos y respuestas.

corePoolSize

Descripción

El número de subprocesos que se mantendrán en la agrupación, aunque estén desocupados, para recopilar los datos del historial de contactos y respuestas.

Valor predeterminado

5

maxPoolSize

Descripción

El número máximo de subprocesos que se mantendrán en la agrupación para recopilar los datos del historial de contactos y respuestas.

Valor predeterminado

5

keepAliveTimeSecs

Descripción

Cuando el número de subprocesos es mayor que el principal, es el tiempo máximo que los subprocesos desocupados excedentes esperan nuevas tareas antes de finalizar para recopilar los datos del historial de contactos y respuestas.

Valor predeterminado

5

queueCapacity

Descripción

El tamaño de la cola utilizada por la agrupación de subprocesos para recopilar los datos del historial de contactos y respuestas.

Valor predeterminado

1000

termWaitSecs

Descripción

Al cerrar el servidor de ejecución, es el número de segundos que se espera a que los subprocesos de servicio terminen de recopilar los datos del historial de contactos y respuestas.

Valor predeterminado

5

Interact | services | threadManagement | allOtherServices

Las propiedades de configuración en esta categoría definen los valores de gestión de subprocesos para los servicios que recopilan las estadísticas de elegibilidad de ofertas, las estadísticas de actividades de eventos, las estadísticas de uso de cadenas predeterminadas y el registro personalizado en los datos de la tabla.

corePoolSize

Descripción

El número de subprocesos que se mantendrán en la agrupación, aunque estén desocupados, para los servicios que recopilan las estadísticas de elegibilidad de ofertas, las estadísticas de actividades de eventos, las estadísticas de uso de cadenas predeterminadas y el registro personalizado en los datos de la tabla.

Valor predeterminado

5

maxPoolSize

Descripción

El número máximo de subprocesos que se mantendrán en la agrupación para los servicios que recopilan las estadísticas de elegibilidad de ofertas, las estadísticas de actividades de eventos, las estadísticas de uso de cadenas predeterminadas y el registro personalizado en los datos de la tabla.

Valor predeterminado

5

keepAliveTimeSecs

Descripción

Cuando el número de subprocesos es mayor que el principal, es el tiempo máximo que los subprocesos desocupados excedentes esperan nuevas tareas antes de finalizar para los servicios que recopilan las estadísticas de elegibilidad de ofertas, las estadísticas de actividades de eventos, las estadísticas de uso de cadenas predeterminadas y el registro personalizado en los datos de la tabla.

Valor predeterminado

5

queueCapacity

Descripción

El tamaño de la cola utilizada por la agrupación de subprocesos para los servicios que recopilan las estadísticas de elegibilidad de ofertas, las estadísticas de actividades de eventos, las estadísticas de uso de cadenas predeterminadas y el registro personalizado en los datos de la tabla.

Valor predeterminado

1000

termWaitSecs

Descripción

Al cerrar el servidor de ejecución, es el número de segundos que se espera a que finalicen los subprocesos de servicio para para los servicios que recopilan las estadísticas de elegibilidad de ofertas, las estadísticas de actividades de eventos, las estadísticas de uso de cadenas predeterminadas y el registro personalizado en los datos de la tabla.

Valor predeterminado

5

Interact | services | threadManagement | flushCacheToDB

Las propiedades de configuración de esta categoría definen los valores de gestión de subprocesos para los subprocesos que escriben datos recopilados de la memoria caché en las tablas de base de datos del entorno de ejecución.

corePoolSize

Descripción

El número de subprocesos que se mantienen en la agrupación para los subprocesos planificados que escriben datos guardados en caché en el almacén de datos.

Valor predeterminado

5

maxPoolSize

Descripción

El número máximo de subprocesos que se mantienen en la agrupación para los subprocesos planificados que escriben datos guardados en caché en el almacén de datos.

Valor predeterminado

5

keepAliveTimeSecs

Descripción

Cuando el número de subprocesos es mayor que el principal, es el tiempo máximo que los subprocesos desocupados excedentes esperan nuevas tareas antes de finalizar para los subprocesos planificados que escriben datos guardados en caché en el almacén de datos.

Valor predeterminado

5

queueCapacity

Descripción

El tamaño de la cola utilizada por la agrupación de subprocesos para los subprocesos planificados que escriben datos guardados en caché en el almacén de datos.

Valor predeterminado

1000

termWaitSecs

Descripción

Al cerrar el servidor de ejecución, es el número de segundos que se espera a que finalicen los subprocesos de servicio para los subprocesos planificados que escriben datos guardados en caché en el almacén de datos.

Valor predeterminado

5

Interact | sessionManagement

Este conjunto de propiedades de configuración define los valores de las sesiones de ejecución.

cacheType

Descripción

Define el tipo de enfoque de memoria caché para los servidores de ejecución.

Valor predeterminado

`Local`

Valores válidos

Distributed | Local

maxNumberOfSessions

Descripción

El número máxima de sesiones de ejecución que mantiene la memoria caché en cualquier momento. Si se produce una solicitud de añadir una nueva sesión de ejecución cuando la memoria caché ha alcanzado este número máximo, la memoria caché elimina la sesión de ejecución más antigua que se encuentre inactiva.

Valor predeterminado

999999999

Valores válidos

Un entero mayor que 0.

multicastIPAddress

Descripción

Si `cacheType` es `Distributed`, escriba la dirección IP utilizada por la memoria caché distribuida. También debe definir `multicastPort`.

Si `cacheType` es `Local`, puede dejar `multicastIPAddress` sin definir.

Valor predeterminado

230.0.0.1

Valores válidos

Una dirección IP válida.

multicastPort

Descripción

Si `cacheType` es `Distributed`, escriba el número de puerto utilizado por la memoria caché distribuida. También debe definir `multicastIPAddress`.

Si `cacheType` es `Local`, puede dejar `multicastPort` sin definir.

Valor predeterminado

6363

Valores válidos

1024 – 49151

sessionTimeoutInSecs

Descripción

La cantidad de tiempo, en segundos, que una sesión puede permanecer inactiva. Una vez transcurrido el número de segundos de `sessionTimeout`, Interact finaliza la sesión.

Valor predeterminado

`300`

Valores válidos

Un entero mayor que cero.

Apéndice C. Propiedades de configuración del entorno de diseño de Interact

Esta sección describe todas las propiedades de configuración del entorno de diseño de Interact.

Campaign | partitions | partition[n] | reports

Estas propiedades de configuración definen carpetas para los informes.

offerAnalysisTabCachedFolder

Descripción

La propiedad `offerAnalysisTabCachedFolder` especifica la ubicación de la carpeta que contiene la especificación para informes de oferta ampliados (expandidos) listados en la pestaña Análisis cuando llega a ella pulsando la pestaña Análisis del panel de navegación. La ruta se especifica mediante notación XPath.

Valor predeterminado

```
/content/folder[@name='Affinium Campaign - Informes específicos del objeto']/folder[@name='offer']/folder[@name='cached']
```

segmentAnalysisTabOnDemandFolder

Descripción

La propiedad `segmentAnalysisTabOnDemandFolder` especifica la ubicación de la carpeta que contiene los informes de segmento que se listan en la pestaña Análisis de un segmento. La ruta se especifica mediante notación XPath.

Valor predeterminado

```
/content/folder[@name='Affinium Campaign - Informes específicos de objeto']/folder[@name='segment']/folder[@name='cached']
```

offerAnalysisTabOnDemandFolder

Descripción

La propiedad `offerAnalysisTabOnDemandFolder` especifica la ubicación de la carpeta que contiene los informes de oferta que se listan en la pestaña Análisis de una oferta. La ruta se especifica mediante notación XPath.

Valor predeterminado

```
/content/folder[@name='Affinium Campaign - Informes específicos del objeto']/folder[@name='offer']
```

segmentAnalysisTabCachedFolder

Descripción

La propiedad `segmentAnalysisTabCachedFolder` especifica la ubicación de la carpeta que contiene la especificación para informes de segmento

ampliados (expandidos) que se lista en la pestaña Análisis al llegar a ella pulsando el enlace Análisis del panel de navegación. La ruta se especifica mediante notación XPath.

Valor predeterminado

```
/content/folder[@name='Affinium Campaign - Informes específicos de objeto']/folder[@name='segment']
```

analysisSectionFolder

Descripción

La propiedad `analysisSectionFolder` especifica la ubicación de la carpeta raíz donde se almacenan las especificaciones de informe. La ruta se especifica mediante notación XPath.

Valor predeterminado

```
/content/folder[@name='Affinium Campaign']
```

campaignAnalysisTabOnDemandFolder

Descripción

La propiedad `campaignAnalysisTabOnDemandFolder` especifica la ubicación de la carpeta que contiene los informes de campaña que se listan en la pestaña Análisis de una campaña. La ruta se especifica mediante notación XPath.

Valor predeterminado

```
/content/folder[@name='Affinium Campaign - Informes específicos de objeto']/folder[@name='campaign']
```

campaignAnalysisTabCachedFolder

Descripción

La propiedad `campaignAnalysisTabCachedFolder` especifica la ubicación de la carpeta que contiene la especificación para informes de campaña ampliados (expandidos) que se listan en la pestaña Análisis al llegar a ella pulsando el enlace Análisis en el panel de navegación. La ruta se especifica mediante notación XPath.

Valor predeterminado

```
/content/folder[@name='Affinium Campaign - Informes específicos del objeto']/folder[@name='campaign']/folder[@name='cached']
```

campaignAnalysisTabEmessageOnDemandFolder

Descripción

La propiedad `campaignAnalysisTabEmessageOnDemandFolder` especifica la ubicación de la carpeta que contiene los informes de eMessage que se listan en la pestaña Análisis de una campaña. La ruta se especifica mediante notación XPath.

Valor predeterminado

```
/content/folder[@name='Affinium Campaign']/folder[@name='Informes de eMessage']
```

campaignAnalysisTabInteractOnDemandFolder

Descripción

Serie de la carpeta del servidor de informes para informes de Interact.

Valor predeterminado

/content/folder[@name='Affinium Campaign']/folder[@name='Informes de Interact']

Disponibilidad

Esta propiedad es aplicable solo si ha instalado Interact.

interactiveChannelAnalysisTabOnDemandFolder

Descripción

Serie de la carpeta del servidor de informes para los informes de la pestaña Análisis de canal interactivo

Valor predeterminado

/content/folder[@name='Affinium Campaign - Informes específicos del objeto']/folder[@name='canal interactivo']

Disponibilidad

Esta propiedad es aplicable solo si ha instalado Interact.

Campaign | partitions | partition[n] | Interact | contactAndResponseHistTracking

Estas propiedades de configuración definen los valores del módulo del historial de contactos y respuestas de Interact.

isEnabled

Descripción

Si se establece en `yes`, habilita el módulo del historial de contactos y respuestas de Interact, que copia el historial de contactos y respuestas de Interact desde las tablas de preparación del tiempo de ejecución de Interact en las tablas de historial de contactos y respuestas de Campaign. La propiedad `interactInstalled` también debe establecerse en `yes`.

Valor predeterminado

no

Valores válidos

yes | no

Disponibilidad

Esta propiedad sólo es aplicable si ha instalado Interact.

runOnceADay

Descripción

Especifica si se debe ejecutar el ETL del historial de contactos y respuestas una vez al día. Si establece esta propiedad en `Yes`, el ETL se ejecuta durante el intervalo planificado especificado por `preferredStartTime` y `preferredEndTime`.

Si el ETL tarda más de 24 horas en ejecutarse y, por lo tanto, no llega a la hora de inicio del día siguiente, omitirá ese día y se ejecutará a la hora planificada del día siguiente. Por ejemplo, si ETL está configurado para ejecutarse entre la 1:00 a.m. y las 3:00 a.m., y el proceso se inicia a la 1:00 a.m. del lunes y termina a las 2:00 a.m. del martes, la siguiente ejecución, planificada originalmente para la 1:00 a.m. del martes, se omitirá y el siguiente ETL empezará a la 1:00 a.m. del miércoles.

La planificación de ETL no tiene en cuenta los cambios del horario de verano. Por ejemplo, si el ETL está planificado para ejecutarse entre la 1:00 a.m. y las 3:00 a.m., puede ejecutarse a las 12:00 a.m. o las 2:00 a.m. cuando se produce el cambio de horario de verano.

Valor predeterminado

No

Disponibilidad

Esta propiedad sólo es aplicable si ha instalado Interact.

processSleepIntervalInMinutes

Descripción

El número de minutos que espera el módulo del historial de contactos y respuestas de Interact entre la copia de los datos de las tablas de preparación de tiempo de ejecución de Interact en las tablas del historial de contactos y respuestas de Campaign.

Valor predeterminado

60

Valores válidos

Un entero mayor que cero.

Disponibilidad

Esta propiedad sólo es aplicable si ha instalado Interact.

preferredStartTime

Descripción

La hora preferida para iniciar el proceso de ETL diario. Esta propiedad, cuando se utiliza con la propiedad preferredEndTime, configura el intervalo de tiempo preferido en el que desea ejecutar el ETL. El ETL se iniciará durante el intervalo de tiempo especificado y procesará como máximo el número de registros especificado utilizando `maxJDBCFetchBatchSize`. El formato es HH:mm:ss AM o PM, utilizando un reloj de 12 horas.

Valor predeterminado

12:00:00 AM

Disponibilidad

Esta propiedad sólo es aplicable si ha instalado Interact.

preferredEndTime

Descripción

La hora preferida para finalizar el proceso de ETL diario. Esta propiedad, cuando se utiliza con la propiedad preferredStartTime, configura el intervalo de tiempo preferido en el que desea ejecutar el ETL. El ETL se iniciará durante el intervalo de tiempo especificado y procesará como máximo el número de registros especificado utilizando `maxJDBCFetchBatchSize`. El formato es HH:mm:ss AM o PM, utilizando un reloj de 12 horas.

Valor predeterminado

2:00:00 AM

Disponibilidad

Esta propiedad sólo es aplicable si ha instalado Interact.

purgeOrphanResponseThresholdInMinutes

Descripción

El número de minutos que espera el módulo del historial de contactos y respuestas de Interact antes de depurar las respuestas sin un contacto correspondiente. Esto evita que se registren respuestas sin que se registren contactos.

Valor predeterminado

180

Valores válidos

Un entero mayor que cero.

Disponibilidad

Esta propiedad sólo es aplicable si ha instalado Interact.

maxJDBCInsertBatchSize

Descripción

El número máximo de registros de un lote de JDBC antes de confirmar la consulta. No es el número máximo de registros que procesa el módulo del historial de contactos y respuestas de Interact en una iteración. Durante cada iteración, el módulo del historial de contactos y respuestas de Interact procesa todos los registros disponibles de las tablas de preparación. No obstante, todos esos registros están divididos en fragmentos de `maxJDBCInsertSize`.

Valor predeterminado

1000

Valores válidos

Un entero mayor que cero.

Disponibilidad

Esta propiedad sólo es aplicable si ha instalado Interact.

maxJDBCFetchBatchSize

Descripción

El número máximo de registros de un lote de JDBC que se captan de la base de datos de preparación. Es probable que deba aumentar este valor para ajustar el rendimiento del módulo del historial de contactos y respuestas.

Por ejemplo, para procesar 2,5 millones de registros del historial de contacto al día, debe establecer `maxJDBCFetchBatchSize` en un número mayor que 2.500.000 para que se procesen todos los registros de un día.

A continuación, puede establecer `maxJDBCFetchChunkSize` y `maxJDBCInsertBatchSize` en valores menores (en este ejemplo, quizá 50.000 y 10.000, respectivamente). También se podrán procesar algunos registros del día siguiente, pero se retendrán hasta el día siguiente.

Valor predeterminado

1000

Valores válidos

Un entero mayor que cero

maxJDBCFetchChunkSize

Descripción

El número máximo de un tamaño de fragmento de JDBC de lectura de datos durante el proceso de ETL (Extracción, Transformación y Carga). En algunos casos, un tamaño de fragmento mayor que el tamaño de inserción puede mejorar la velocidad del proceso de ETL.

Valor predeterminado

1000

Valores válidos

Un entero mayor que cero

deleteProcessedRecords

Descripción

Especifica si se retienen los registros del historial de contactos y el historial de respuestas una vez procesados.

Valor predeterminado

Yes

completionNotificationScript

Descripción

Especifica la ruta absoluta de un script que se ejecuta cuando finaliza el ETL. Si especifica un script, se pasan cuatro argumentos al script de notificación de finalización: hora de inicio, hora de finalización, número total de registros de CH procesados y número total de registros de RH procesados. La hora de inicio y la hora de finalización son valores numéricos que representan el número de milisegundos transcurridos desde 1970.

Valor predeterminado

Ninguno

fetchSize

Descripción

Permite establecer JDBC fetchSize cuando recupera los registros de las tablas de preparación.

En las bases de datos Oracle especialmente, ajuste el valor en el número de registros que JDBC debe recuperar con cada viaje de ida y vuelta de red. Para los lotes de gran tamaño de 100.000 o más, pruebe 10000. Tenga cuidado de no utilizar un valor demasiado grande, ya que tendrá un impacto en el uso de la memoria y las ventajas serán insignificantes, si no negativas.

Valor predeterminado

Ninguno

Campaign | partitions | partition[n] | Interact | contactAndResponseHistTracking | runtimeDataSources | [runtimeDataSource]

Estas propiedades de configuración definen el origen de datos del módulo del historial de contactos y respuestas de Interact.

jndiName

Descripción

Utilice la propiedad `systemTablesDataSource` para identificar el origen de datos JNDI (Java Naming and Directory Interface) que está definido en el servidor de aplicaciones (WebSphere o WebLogic) para las tablas de ejecución de Interact.

La base de datos de tiempo de ejecución de Interact es la base de datos que se completa con los scripts dll `aci_runtime` y `aci_populate_runtime` y, por ejemplo, contiene las siguientes tablas (entre otras): `UACI_CHOfferAttrib` y `UACI_DefaultedStat`.

Valor predeterminado

No se ha definido ningún valor predeterminado.

Disponibilidad

Esta propiedad sólo es aplicable si ha instalado Interact.

databaseType

Descripción

Tipo de base de datos para el origen de datos de tiempo de ejecución de Interact.

Valor predeterminado

SQLServer

Valores válidos

SQLServer | Oracle | DB2

Disponibilidad

Esta propiedad sólo es aplicable si ha instalado Interact.

schemaName

Descripción

El nombre del esquema que contiene las tablas de preparación del módulo de historial de contactos y respuestas. Debería ser el mismo que el de las tablas del entorno de ejecución.

No tiene que definir un esquema.

Valor predeterminado

No se ha definido ningún valor predeterminado.

Campaign | partitions | partition[n] | Interact | contactAndResponseHistTracking | contactTypeMappings

Estas propiedades de configuración definen el tipo de contacto de la campaña que se correlaciona con un 'contact' para la creación de informes o el aprendizaje.

contacted

Descripción

El valor asignado a la columna `ContactStatusID` de la tabla `UA_DtlContactHist` de las tablas del sistema de Campaign para un contacto de oferta. El valor debe ser una entrada válida de la tabla `UA_ContactStatus`. Consulte la publicación *Campaign Guía del administrador* para obtener detalles sobre cómo añadir tipos de contacto.

Valor predeterminado

2

Valores válidos

Un entero mayor que cero.

Disponibilidad

Esta propiedad es aplicable solo si ha instalado Interact.

Campaign | partitions | partition[n] | Interact | contactAndResponseHistTracking | responseTypeMappings

Estas propiedades de configuración definen las respuestas para aceptar o rechazar los informes y el aprendizaje.

accept

Descripción

El valor asignado a la columna `ResponseTypeID` de la tabla `UA_ResponseHistory` en las tablas del sistema de Campaign para una oferta aceptada. El valor debe ser una entrada válida en la tabla `UA_UsrResponseType`. Debe asignar a la columna `CountsAsResponse` el valor `1`, una respuesta.

Consulte la publicación *Campaign Administrator's Guide* para obtener información detallada sobre cómo añadir tipos de respuesta.

Valor predeterminado

3

Valores válidos

Un entero mayor que cero.

Disponibilidad

Esta propiedad sólo es aplicable si ha instalado Interact.

reject

Descripción

El valor asignado a la columna `ResponseTypeID` de la tabla `UA_ResponseHistory` en las tablas del sistema de Campaign para una oferta rechazada. El valor debe ser una entrada válida en la tabla `UA_UsrResponseType`. Debe asignar a la columna `CountsAsResponse` el valor `2`, un rechazo. Consulte la publicación *Campaign Administrator's Guide* para obtener información detallada sobre cómo añadir tipos de respuesta.

Valor predeterminado

8

Valores válidos

Un entero mayor que cero.

Disponibilidad

Esta propiedad sólo es aplicable si ha instalado Interact.

Campaign | partitions | partition[n] | Interact | report

Estas propiedades de configuración definen los nombres de informe cuando se realiza la integración con Cognos.

interactiveCellPerformanceByOfferReportName

Descripción

Nombre del informe Rendimiento de celda interactiva por oferta. Este nombre debe coincidir con el nombre de este informe en el servidor Cognos.

Valor predeterminado

Rendimiento de celda interactiva por oferta

treatmentRuleInventoryReportName

Descripción

Nombre del informe Inventario de reglas de tratamiento. Este nombre debe coincidir con el nombre de este informe en el servidor Cognos.

Valor predeterminado

Inventario de reglas de tratamiento de canal

deploymentHistoryReportName

Descripción

Nombre del informe Historial de despliegues. Este nombre debe coincidir con el nombre de este informe en el servidor Cognos.

Valor predeterminado

Historial de despliegues de canal

Campaign | partitions | partition[n] | Interact | learning

Estas propiedades de configuración permiten ajustar el módulo de aprendizaje incorporado.

confidenceLevel

Descripción

Un porcentaje que indica la confianza que se desea que tenga la utilidad de aprendizaje antes de cambiar de la exploración a la explotación. Un valor 0 desactiva efectivamente la exploración.

Esta propiedad sólo es aplicable si la propiedad `Interact > offerserving > optimizationType` del tiempo de ejecución de Interact se establece en `BuiltInLearning`.

Valor predeterminado

95

Valores válidos

Un entero entre 0 y 95, divisible por 5 o 99.

enableLearning

Descripción

Si se establece en `Yes`, el tiempo de diseño de Interact espera que el aprendizaje esté habilitado. Si establece `enableLearning` en `yes`, debe configurar `Interact > offerserving > optimizationType` en `BuiltInLearning` o `ExternalLearning`.

Si se establece en `No`, el tiempo de diseño de Interact espera que el aprendizaje esté inhabilitado. Si establece `enableLearning` en `no`, debe configurar `Interact > offerserving > optimizationType` en `NoLearning`.

Valor predeterminado

No

maxAttributeNames

Descripción

El número máximo de atributos de aprendizaje que supervisa la utilidad de aprendizaje de Interact.

Esta propiedad sólo es aplicable si la propiedad `Interact > offerserving > optimizationType` del tiempo de ejecución de Interact se establece en `BuiltInLearning`.

Valor predeterminado

10

Valores válidos

Un entero cualquiera.

maxAttributeValues

Descripción

El número máximo de valores para los que el módulo de aprendizaje de Interact realiza un seguimiento para cada atributo de aprendizaje.

Esta propiedad sólo es aplicable si la propiedad `Interact > offerserving > optimizationType` del tiempo de ejecución de Interact se establece en `BuiltInLearning`.

Valor predeterminado

5

otherAttributeValue

Descripción

El nombre predeterminado del valor de atributo que se utiliza para representar todos los valores de atributo más allá de `maxAttributeValues`.

Esta propiedad sólo es aplicable si la propiedad `Interact > offerserving > optimizationType` del tiempo de ejecución de Interact se establece en `BuiltInLearning`.

Valor predeterminado

`Other`

Valores válidos

Una cadena o un número.

Ejemplo

Si maxAttributeValues se establece en 3 y otherAttributeValue se establece en other, el módulo de aprendizaje realiza un seguimiento de los tres primeros valores. Los demás valores se asignan a la otra categoría. Por ejemplo, si realiza un seguimiento del atributo de visitante de color de cabello, y los cinco primeros visitantes tienen el cabello de color moreno, castaño, rubio, pelirrojo y gris, la utilidad de aprendizaje realiza un seguimiento de los colores de cabello moreno, castaño y rubio. Los colores pelirrojo y gris se agrupan en `otherAttributeValue`.

percentRandomSelection

Descripción

El porcentaje de tiempo que el módulo de aprendizaje presenta una oferta aleatoria. Por ejemplo, si percentRandomSelection se establece en 5, esto significa que el módulo de aprendizaje presenta una oferta aleatoria el 5% del tiempo (5 de cada 100 recomendaciones).

Valor predeterminado

5

Valores válidos

Un entero cualquiera de 0 a 100.

recencyWeightingFactor

Descripción

La representación decimal de un porcentaje del conjunto de datos definido por `recencyWeightingPeriod`. Por ejemplo, el valor predeterminado 0,15 significa que el 15% de los datos utilizados por la utilidad de aprendizaje provienen de `recencyWeightingPeriod`.

Esta propiedad sólo es aplicable si la propiedad `Interact > offerserving > optimizationType` del tiempo de ejecución de Interact se establece en `BuiltInLearning`.

Valor predeterminado

0,15

Valores válidos

Un valor decimal menor que 1.

recencyWeightingPeriod

Descripción

El tamaño en horas de datos a los que el módulo de aprendizaje ha otorgado el porcentaje `recencyWeightingFactor` de ponderación. Por ejemplo, el valor predeterminado 120 significa que el `recencyWeightingFactor` de los datos utilizados por el módulo de aprendizaje provienen de las últimas 120 horas.

Esta propiedad sólo es aplicable si `optimizationType` se ha establecido en `builtInLearning`.

Valor predeterminado

120

minPresentCountThreshold

Descripción

El número mínimo de veces que debe presentarse una oferta antes de que sus datos se utilicen en los cálculos y el módulo de aprendizaje entre en el modo de exploración.

Valor predeterminado

0

Valores válidos

Un entero mayor o igual que cero.

enablePruning

Descripción

Si se establece en `Yes`, el módulo de aprendizaje de Interact determina con algoritmos cuándo un atributo de aprendizaje (estándar o dinámico) no es predictivo. Si un atributo de aprendizaje no es predictivo, el módulo de aprendizaje no lo considerará cuando determine la ponderación de una oferta. Esto continúa hasta que el módulo de aprendizaje agregue datos de aprendizaje.

Si se establece en `No`, el módulo de aprendizaje siempre utiliza todos los atributos de aprendizaje. Si no se podan los atributos no predictivos, el módulo de aprendizaje no tendrá la precisión que debería.

Valor predeterminado

Yes

Valores válidos

Yes | No

Campaign | partitions | partition[n] | Interact | learning | learningAttributes | [learningAttribute]

Estas propiedades de configuración definen los atributos de aprendizaje.

attributeName

Descripción

Cada `attributeName` es el nombre de un atributo de visitante que se desea supervisar con el módulo de aprendizaje. Debe coincidir con el nombre de un par nombre-valor en los datos de sesión.

Esta propiedad sólo es aplicable si la propiedad `Interact > offerserving > optimizationType` del tiempo de ejecución de Interact se establece en `BuiltInLearning`.

Valor predeterminado

No se ha definido ningún valor predeterminado.

Campaign | partitions | partition[n] | Interact | deployment

Estas propiedades de configuración definen los valores de despliegue.

chunkSize

Descripción

El tamaño máximo de fragmentación en KB para cada paquete de despliegue de Interact.

Valor predeterminado

500

Disponibilidad

Esta propiedad sólo es aplicable si ha instalado Interact.

Campaign | partitions | partition[n] | Interact | serverGroups | [serverGroup]

Estas propiedades de configuración definen los valores del grupo de servidores.

serverGroupName

Descripción

El nombre del grupo de servidores de ejecución de Interact. Es el nombre que aparece en la pestaña Resumen del canal interactivo.

Valor predeterminado

No se ha definido ningún valor predeterminado.

Disponibilidad

Esta propiedad sólo es aplicable si ha instalado Interact.

Campaign | partitions | partition[n] | Interact | serverGroups | [serverGroup] | instanceURLs | [instanceURL]

Estas propiedades de configuración definen los servidores de ejecución de Interact.

instanceURL

Descripción

El URL del servidor de ejecución de Interact. Un grupo de servidores puede contener varios servidores de ejecución de Interact; no obstante, cada servidor se debe crear con una nueva categoría.

Valor predeterminado

No se ha definido ningún valor predeterminado.

Ejemplo

`http://server:port/interact`

Disponibilidad

Esta propiedad sólo es aplicable si ha instalado Interact.

Campaign | partitions | partition[n] | Interact | flowchart

Estas propiedades de configuración definen el entorno de ejecución de Interact utilizado para las ejecuciones de prueba de diagramas de flujo interactivos.

serverGroup

Descripción

El nombre del grupo de servidores de Interact que Campaign utiliza para ejecutar una prueba de ejecución. Este nombre debe coincidir con el nombre de categoría que crea en `serverGroups`.

Valor predeterminado

No se ha definido ningún valor predeterminado.

Disponibilidad

Esta propiedad sólo es aplicable si ha instalado Interact.

dataSource

Descripción

Utilice la propiedad `dataSource` para identificar el origen de datos físico que Campaign utiliza cuando realiza las ejecuciones de prueba de diagramas de flujo interactivos. Esta propiedad debe coincidir con el origen

de datos definido por la propiedad Campaign > partitions > partitionN > dataSources del origen de datos de ejecución de prueba definido para el tiempo de diseño de Interact.

Valor predeterminado

No se ha definido ningún valor predeterminado.

Disponibilidad

Esta propiedad sólo es aplicable si ha instalado Interact.

Campaign | partitions | partition[n] | Interact | whiteList | [AudienceLevel] | DefaultOffers

Estas propiedades de configuración definen el código de celda predeterminado para la tabla de ofertas predeterminadas. Sólo debe configurar estas propiedades si está definiendo asignaciones de oferta globales.

DefaultCellCode

Descripción

El código de celda predeterminado que Interact utiliza si no define un código de celda en la tabla de ofertas predeterminadas.

Valor predeterminado

No se ha definido ningún valor predeterminado.

Valores válidos

Una cadena que coincide con el formato de código de celda definido en Campaign

Disponibilidad

Esta propiedad sólo es aplicable si ha instalado Interact.

Campaign | partitions | partition[n] | Interact | whiteList | [AudienceLevel] | offersBySQL

Estas propiedades de configuración definen el código de celda predeterminado para la tabla offersBySQL. Sólo debe configurar estas propiedades si utiliza consultas SQL para obtener el conjunto deseado de ofertas candidatas.

DefaultCellCode

Descripción

El código de celda predeterminado que Interact utiliza para una oferta en las tablas OffersBySQL que tiene un valor nulo en la columna de código de celda (o si la columna de código de celda no aparece en absoluto). El valor debe ser un código de celda válido.

Valor predeterminado

No se ha definido ningún valor predeterminado.

Valores válidos

Una cadena que coincide con el formato de código de celda definido en Campaign

Disponibilidad

Esta propiedad sólo es aplicable si ha instalado Interact.

Campaign | partitions | partition[n] | Interact | whiteList | [AudienceLevel] | ScoreOverride

Estas propiedades de configuración definen el código de celda predeterminado para la tabla de alteración temporal de puntuaciones. Sólo debe configurar estas propiedades si está definiendo asignaciones de oferta individuales.

DefaultCellCode

Descripción

El código de celda predeterminado que Interact utiliza si no define un código de celda en la tabla de alteración temporal de puntuaciones.

Valor predeterminado

No se ha definido ningún valor predeterminado.

Valores válidos

Una cadena que coincide con el formato de código de celda definido en Campaign

Disponibilidad

Esta propiedad sólo es aplicable si ha instalado Interact.

Campaign | partitions | partition[n] | server | internal

Las propiedades de esta categoría especifican valores de integración y los límites de internalID para la partición de Campaign seleccionada. Si la instalación de Campaign tiene varias particiones, establezca estas propiedades para cada partición que desee modificar.

internalIdLowerLimit

Descripción

Las propiedades `internalIdUpperLimit` e `internalIdLowerLimit` limitan los ID internos de Campaign para que estén dentro del rango especificado. Tenga en cuenta que los valores son inclusivos, es decir, Campaign puede utilizar el límite inferior y el límite superior.

Valor predeterminado

`0` (cero)

internalIdUpperLimit

Descripción

Las propiedades `internalIdUpperLimit` e `internalIdLowerLimit` limitan los ID internos de Campaign para que estén dentro del rango especificado. Tenga en cuenta que los valores son inclusivos, es decir, Campaign puede utilizar el límite inferior y el límite superior.

Valor predeterminado

`4294967295`

eMessageInstalled

Descripción

Indica que eMessage se ha instalado. Cuando selecciona `yes`, las características de eMessage están disponibles en la interfaz de Campaign.

El instalador de IBM establece esta propiedad en `yes` para la partición predeterminada en su instalación de eMessage. Para las particiones adicionales donde ha instalado eMessage, debe configurar esta propiedad manualmente.

Valor predeterminado

`no`

Valores válidos

`yes | no`

interactInstalled

Descripción

Después de instalar el entorno de diseño de Interact, esta propiedad de configuración se debe establecer en `yes` para habilitar el entorno de diseño de Interact en Campaign.

Si no tiene instalado Interact, establezca esta propiedad en `no`. Si se establece esta propiedad en `no`, no elimina los menús y las opciones de Interact de la interfaz de usuario. Para eliminar los menús y las opciones, debe anular manualmente el registro de Interact utilizando la utilidad `configTool`.

Valor predeterminado

no

Valores válidos

yes | no

Disponibilidad

Esta propiedad sólo es aplicable si ha instalado Interact.

MO_UC_integration

Descripción

Habilita la integración con Marketing Operations para esta partición. Si tiene previsto establecer alguna de las siguientes tres opciones en Yes, debe establecer **MO_UC_integration** en Yes. Para obtener más información sobre cómo configurar esta integración, consulte *IBM Unica Marketing Operations and Campaign Integration Guide*.

Valor predeterminado

no

Valores válidos

yes | no

MO_UC_BottomUpTargetCells

Descripción

Permite las celdas ascendentes para las Hojas de cálculo de celda objetivo en esta partición. Si se establece en Yes, están visibles las celdas objetivo descendentes y ascendentes, aunque las celdas objetivo son de sólo lectura. Tenga en cuenta que **MO_UC_integration** debe estar habilitado. Para obtener más información sobre cómo configurar esta integración, consulte *IBM Unica Marketing Operations and Campaign Integration Guide*.

Valor predeterminado

no

Valores válidos

yes | no

Legacy_campaigns

Descripción

Cuando la propiedad **MO_UC_integration** se establece en **Yes**, la propiedad **Legacy_campaigns** habilita el acceso a las campañas creadas antes de habilitar la integración, incluidas las campañas creadas en Campaign 7.x y enlazadas a los proyectos de Plan 7.x. Para obtener más información sobre cómo configurar esta integración, consulte *IBM Unica Marketing Operations and Campaign Integration Guide*.

Valor predeterminado

no

Valores válidos

yes | no

IBM Unica Marketing Operations - Integración de oferta

Descripción

Habilita la posibilidad de utilizar Marketing Operations para ejecutar tareas de gestión de ciclo de vida de las ofertas en esta partición. (**MO_UC_integration** debe estar habilitado. Asimismo, **Integración de campaña** debe estar habilitado en **Configuración > Configuración > Unica > Plataforma**.) Para obtener más información sobre cómo configurar esta integración, consulte *IBM Unica Marketing Operations and Campaign Integration Guide*.

Valor predeterminado

no

Valores válidos

yes | no

UC_CM_integration

Descripción

Habilita la integración de segmentos en línea de IBM Coremetrics para una partición de campaña. Si establece esta opción en Yes, el cuadro Proceso Selección en un diagrama de flujo proporcionará la opción para seleccionar **Segmentos de IBM Coremetrics** como entrada. Para configurar la integración para cada partición, seleccione **Configuración > Configuración > Campaign | partitions | partition[n] | Coremetrics**.

Valor predeterminado

no

Valores válidos

yes | no

Campaign | monitoring

Las propiedades de esta categoría especifican si está habilitada la característica de supervisión operativa, el URL del servidor de supervisión operativa y el comportamiento del almacenamiento en caché. La supervisión operativa se visualiza y permite controlar los diagramas de flujo activos.

cacheCleanupInterval

Descripción

La propiedad `cacheCleanupInterval` especifica el intervalo, en segundos, entre las limpiezas automáticas de la memoria caché de estado del diagrama de flujo.

Esta propiedad no está disponible en las versiones de Campaign anteriores a la 7.0.

Valor predeterminado

`600` (10 minutos)

cacheRunCompleteTime

Descripción

La propiedad `cacheRunCompleteTime` especifica la cantidad de tiempo, en minutos, que se guardan en caché las ejecuciones completas y se muestran en la página de supervisión.

Esta propiedad no está disponible en las versiones de Campaign anteriores a la 7.0.

Valor predeterminado

`4320`

monitorEnabled

Descripción

La propiedad `monitorEnabled` especifica si el supervisor está activado.

Esta propiedad no está disponible en las versiones de Campaign anteriores a la 7.0.

Valor predeterminado

`yes`

serverURL

Descripción

La propiedad Campaign > monitoring > serverURL especifica el URL del servidor de supervisión operativa. Es un valor obligatorio; modifique el valor si el URL del servidor de supervisión operativa no es el valor predeterminado.

Si Campaign se ha configurado para utilizar comunicaciones SSL (Secure Sockets Layer), establezca el valor de esta propiedad para utilizar HTTPS. Por ejemplo: serverURL=https://*host:SSL_port*/Campaign/OperationMonitor, donde:

- *host* es el nombre o la dirección IP de la máquina donde está instalada la aplicación web
- *SSL_Port* es el puerto SSL de la aplicación web.

Observe el https en el URL.

Valor predeterminado

http://localhost:7001/Campaign/OperationMonitor

monitorEnabledForInteract

Descripción

Si se establece en yes, habilita el servidor del conector JMX de Campaign para Interact. Campaign no tiene seguridad JMX.

Si se establece en no, no puede conectarse al servidor del conector JMX de Campaign.

Esta supervisión JMX sólo está indicada para el módulo de historial de contactos y respuestas de Interact.

Valor predeterminado

False

Valores válidos

True | False

Disponibilidad

Esta propiedad sólo es aplicable si ha instalado Interact.

protocol

Descripción

Protocolo de escucha del servidor del conector JMX de Campaign, si monitorEnabledForInteract está establecido en yes.

Esta supervisión JMX sólo está indicada para el módulo de historial de contactos y respuestas de Interact.

Valor predeterminado

JMXMP

Valores válidos

JMXMP | RMI

Disponibilidad

Esta propiedad sólo es aplicable si ha instalado Interact.

port

Descripción

Puerto de escucha del servidor del conector JMX de Campaign, si `monitorEnabledForInteract` está establecido en yes.

Esta supervisión JMX sólo está indicada para el módulo de historial de contactos y respuestas de Interact.

Valor predeterminado

2004

Valores válidos

Un entero entre 1025 y 65535.

Disponibilidad

Esta propiedad sólo es aplicable si ha instalado Interact.

Apéndice D. Personalización de ofertas en tiempo real en el lado del cliente

Puede haber situaciones en las que desee proporcionar personalización de ofertas en tiempo real sin implementar código Java de bajo nivel o llamadas SOAP al servidor Interact. Por ejemplo, cuando un visitante carga inicialmente una página web donde el contenido de Javascript es la única programación ampliada disponible, o cuando un visitante abre un mensaje de correo electrónico donde sólo es posible contenido HTML. IBM Unica Interact proporciona varios conectores que proporcionan gestión de ofertas en tiempo real en situaciones en las que solo se tiene control sobre el contenido web que se carga en el lado del cliente, o en las que se desea simplificar la interfaz a Interact.

La instalación de Interact incluye dos conectores para personalización de ofertas iniciada en el lado del cliente:

- "Acerca de Interact Message Connector". Mediante Message Connector, el contenido web de los mensajes de correo electrónico (por ejemplo) u otro soporte electrónico puede contener etiquetas de enlaces e imágenes para realizar llamadas al servidor Interact para presentación de ofertas de carga de página o páginas de destino de pulsación.
- "Acerca de Interact Web Connector" en la página 232. Mediante Web Connector (también denominado JS Connector), las páginas web pueden utilizar JavaScript del lado del cliente para gestionar el arbitraje, presentación e historial de contactos/respuestas de ofertas en la presentación de ofertas al cargar página o las páginas de destino de pulsación.

Acerca de Interact Message Connector

Interact Message Connector permite que los mensajes de correo electrónico y otros soportes electrónicos realicen llamadas a IBM Unica Interact para permitir que se presenten ofertas personalizadas cuando se abra o el cliente pulse el mensaje y vaya al sitio especificado. Esto se consigue mediante la utilización de dos etiquetas clave: la etiqueta de imagen (`IMG`), que carga las ofertas personalizadas al abrir, o la etiqueta de enlace (`A`), que captura información cuando el cliente pulsa y lo redirige a una página de destino específica.

Ejemplo

El ejemplo siguiente muestra código HTML que se podría incluir en una ubicación de marketing (por ejemplo, en un mensaje de correo electrónico) que contenga un URL de etiqueta `IMG` (que pasa información cuando se abre el documento en el servidor Interact y recupera la imagen de oferta adecuada en la respuesta) y un URL de etiqueta `A` (que determina qué información se pasa al servidor Interact cuando se pulsa):

```
<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>
```

En el ejemplo siguiente, una etiqueta `IMG` está incluida en un código `A`, lo que causa el comportamiento siguiente:

1. Cuando se abre el mensaje de correo electrónico, Message Connector recibe una solicitud que contiene información codificada en la etiqueta IMG: el IDmsj e IDEnlace de este mensaje y los parámetros de cliente que incluyen ID de usuario, nivel de ingresos y tipo de ingresos.
2. Esta información se pasa mediante una llamada a la API al servidor de ejecución de Interact.
3. El servidor de ejecución devuelve una oferta a Message Connector, que recibe el URL de la imagen de la oferta, y proporciona ese URL (con los parámetros adicionales incluidos) y redirecciona la solicitud de imagen a ese URL de oferta.
4. El cliente ve la oferta como una imagen.

En este punto, el cliente puede pulsar dicha imagen para responder a la oferta de alguna forma. Esa pulsación, mediante la etiqueta A y su atributo HREF especificado (que especifica el URL de destino), envía otra solicitud a Message Connector de una página de destino enlazada al URL de esa oferta. A continuación, el explorador del cliente se redirecciona a la página de destino tal como está configurado en la oferta.

Tenga en cuenta que una etiqueta A de pulsación no es estrictamente necesaria; la oferta puede constar solo de una imagen, como un vale para que el cliente lo imprima.

Instalación de Message Connector

Los archivos que necesita para instalar, desplegar y ejecutar Message Connector se han incluido automáticamente con la instalación del servidor de ejecución de IBM Unica Interact. Esta sección resume los pasos necesarios para hacer que Message Connector este listo para su utilización.

La instalación y el despliegue de Message Connector implican las tareas siguientes:

- Opcionalmente, la configuración de los valores predeterminados de Message Connector, tal como se describe en "Configuración de Message Connector".
- La creación de tablas de base de datos necesarias para almacenar los datos de transacciones de Message Connector, tal como se describe en "Creación de tablas de Message Connector" en la página 227.
- La instalación de la aplicación web de Message Connector, tal como se describe en "Despliegue y ejecución de Message Connector" en la página 228.
- La creación de las etiquetas IMG y A en sus ubicaciones de marketing (correos electrónicos o páginas web, por ejemplo) necesarias para llamar a ofertas de Message Connector cuando se abran y pulsen, tal como se describe en "Creación de los enlaces de Message Connector" en la página 229.

Configuración de Message Connector

Para desplegar Message Connector, debe personalizar el archivo de configuración incluido con la instalación para que coincida con su entorno específico. Puede modificar el archivo XML denominado MessageConnectorConfig.xml que se encuentra en el directorio de Message Connector en el servidor de ejecución de Interact, similar a <inicio_Interact>/msgconnector/config/MessageConnectorConfig.xml.

El archivo MessageConnectorConfig.xml contiene algunos valores de configuración que son necesarios, y otros que son opcionales. Los valores que utilice deben estar personalizados para su instalación específica. Siga los pasos siguientes para modificar la configuración.

1. Si Message Connector está desplegado y en ejecución en el servidor de aplicaciones web, anule el despliegue de Message Connector antes de continuar.
2. En el servidor de ejecución de Interact, abra el archivo `MessageConnectorConfig.xml` en cualquier editor de texto o XML.
3. Modifique los valores de configuración según se requiera, asegurándose de que los siguientes valores *necesarios* sean correctos para su instalación.
   - `<URL_interact>`, URL del servidor de ejecución de Interact al que se deben conectar las etiquetas de página de Message Connector y en el que se está ejecutando Message Connector.
   - `<enlace_error_imagen>`, URL al que redirigirá Message Connector si se produce un error al procesar una solicitud para una imagen de oferta.
   - `<enlace_error_página_destino>`, URL al que redirigirá Message Connector si se produce un error al procesar una solicitud para una página de destino de oferta.
   - `<niveles_audiencia>`, una sección del archivo de configuración que contiene uno o más valores de nivel de audiencia, y que especifica el nivel de audiencia predeterminado si el enlace de Message Connector no ha especificado ninguno. Debe haber como mínimo un nivel de audiencia configurado.

   Todos los valores de configuración se describen más detalladamente en "Valores de configuración de Message Connector".
4. Cuando haya completado los cambios de configuración, guarde y cierre el archivo `MessageConnectorConfig.xml`.
5. Continúe configurando y desplegando Message Connector.

Valores de configuración de Message Connector:

Para configurar Message Connector, puede modificar el archivo XML denominado `MessageConnectorConfig.xml` que se encuentra en el directorio de Message Connector en el servidor de ejecución de Interact, normalmente `<inicio_Interact>/msgconnector/config/MessageConnectorConfig.xml`. A continuación se describe cada una de las configuraciones de este archivo XML. Tenga en cuenta que si modifica este archivo una vez que Message Connector esté desplegado y en ejecución, debe asegurarse de anular el despliegue y volver a desplegar Message Connector o reiniciar el servidor de aplicaciones para volver a cargar estos valores una vez que haya acabado de modificar el archivo.

Valores generales

La tabla siguiente contiene una lista de valores opcionales y necesarios incluidos en la sección `generalSettings` del archivo `MessageConnectorConfig.xml`.

Tabla 20. Valores generales de Message Connector

Elemento	Descripción	Valor predeterminado
<interactURL>	URL del servidor de ejecución de Interact para manejar las llamadas de las etiquetas de página de Message Connector, como por ejemplo el servidor de ejecución en el que se está ejecutando Message Connector. Este elemento es necesario.	http://localhost:7001/interact
<defaultDateTimeFormat>	Formato de fecha predeterminado.	MM/dd/aaaa
<log4jConfigFileLocation>	Ubicación del archivo de propiedades Log4j. Es relativa a la variable de entorno *$MESSAGE_CONNECTOR_HOME* si está establecida; de lo contrario, este valor es relativo a la ruta raíz de la aplicación web de Message Connector.	config/ MessageConnectorLog4j.properties

Valores de parámetros predeterminados

La tabla siguiente contiene una lista de valores predeterminados y necesarios contenidos en la sección defaultParameterValues del archivo MessageConnectorConfig.xml.

Tabla 21. Valores de parámetros predeterminados de Message Connector

Elemento	Descripción	Valor predeterminado
<interactiveChannel>	Nombre del canal interactivo predeterminado.	
<interactionPoint>	Nombre del punto de interacción predeterminado.	
<debugFlag>	Determina si la depuración está habilitada. Los valores permitidos son true y false.	false
<contactEventName>	Nombre predeterminado del evento de contacto que se publica.	
<acceptEventName>	Nombre predeterminado del evento de aceptación que se publica.	
<imageUrlAttribute>	Nombre de atributo de oferta predeterminado que contiene el URL de la imagen de oferta, si no se especifica ninguno en el enlace de Message Connector.	
<landingPageUrlAttribute>	URL predeterminado para la página de destino de pulsación si no se especifica ninguno en el enlace de Message Connector.	

Valores de comportamiento

La tabla siguiente contiene una lista de los valores opcionales y necesarios contenidos en la sección behaviorSettings del archivo MessageConnectorConfig.xml.

Tabla 22. Valores de comportamiento de Message Connector

Elemento	Descripción	Valor predeterminado
<imageErrorLink>	URL al que redirige el conector si se produce un error al procesar una solicitud para una imagen de oferta. Este valor es necesario.	/images/default.jpg
<landingPageErrorLink>	URL al que redirige el conector si se produce un error al procesar una solicitud de una página de destino de pulsación. Este valor es necesario.	/jsp/default.jsp
<alwaysUseExistingOffer>	Determina si la oferta en caché se debe devolver, incluso si ya ha caducado. Los valores permitidos son true y false.	false
<offerExpireAction>	Acción que se realizará si se encuentra la oferta original pero ya ha caducado. Los valores permitidos son: • GetNewOffer • RedirectToErrorPage • ReturnExpiredOffer	RedirectToErrorPage

Valores de almacenamiento

La tabla siguiente contiene una lista de valores opcionales y predeterminados contenidos en la sección storageSettings del archivo MessageConnectorConfig.xml.

Tabla 23. Valores de almacenamiento de MessageConnector

Elemento	Descripción	Valor predeterminado
<persistenceMode>	Cuando la memoria caché persiste nuevas entradas en la base de datos. Los valores permitidos son WRITE-BEHIND (donde los datos se graban en la memoria caché inicialmente, y se actualizan en la base de datos posteriormente) y WRITE-THROUGH (donde los datos se graban en la memoria caché y en la base de datos simultáneamente).	WRITE-THROUGH
<maxCacheSize>	Número máximo de entradas en la memoria caché.	5000
<maxPersistenceBatchSize>	Tamaño máximo de proceso por lotes al persistir entradas en la base de datos.	200
<macCachePersistInterval>	Tiempo máximo en segundos que una entrada permanece en la memoria caché antes de que se persista en la base de datos.	3
<maxElementOnDisk>	Número máximo de entradas en la memoria caché de disco.	5000
<cacheEntryTimeToExpireInSeconds>	Periodo máximo de tiempo que las entradas en la caché de disco permanecerán antes de caducar.	60000

Tabla 23. Valores de almacenamiento de MessageConnector (continuación)

Elemento	Descripción	Valor predeterminado
<jdbcSettings>	Sección del archivo XML que contiene información específica si se utiliza una conexión JDBC. Se excluye mutuamente con la sección <dataSourceSettings>.	Configurada de forma predeterminada para conectar a una base de datos de SQLServer configurada en el servidor local, pero si habilita esta sección debe proporcionar los valores de JDBC y las credenciales para iniciar sesión.
<dataSourceSettings>	Sección del archivo XML que contiene información específica si se utiliza una conexión de origen de datos. Se excluye mutuamente con la sección <jdbcSettings>.	Configurado de manera predeterminada para conectar al origen de datos InteractDS definido en el servidor de aplicaciones web local.

Niveles de audiencia

La siguiente tabla contiene una lista de los valores opcionales y necesarios contenidos en la sección audienceLevels del archivo MessageConnectorConfig.xml.

Tenga en cuenta que el elemento audienceLevels se utiliza opcionalmente para especificar el nivel de audiencia predeterminado que se utilizará si no se especifica ninguno en el enlace de Message Connector, como en el ejemplo siguiente:

```
<audienceLevels default="Cliente">
```

En este ejemplo, el valor del atributo predeterminado coincide con el nombre de un audienceLevel definido en esta sección. Debe haber como mínimo un nivel de audiencia definido en este archivo de configuración.

Tabla 24. Valor de nivel de audiencia de MessageConnector

Elemento	Elemento	Descripción	Valor predeterminado
<audienceLevel>		Elemento que contiene la configuración del nivel de audiencia. Proporcione un atributo de nombre, como en <audienceLevel name="Cliente">	
	<messageLogTable>	Nombre de la tabla de registro. Este valor es necesario.	UACI_MESSAGE_CONNECTOR_LOG
<fields>	<field>	Definición de uno o más campos de ID de audiencia para este audienceLevel.	
	<name>	Nombre del campo de ID de audiencia, tal como se especifica en el tiempo de ejecución de Interact.	
	<httpParameterName>	Nombre de parámetro correspondiente para este campo de ID de audiencia.	
	<dbColumnName>	Nombre de columna correspondiente en la base de datos para este campo de ID de audiencia.	

Tabla 24. Valor de nivel de audiencia de MessageConnector (continuación)

Elemento	Elemento	Descripción	Valor predeterminado
	<type>	Tipo del campo de ID de audiencia, tal como se especifica en el tiempo de ejecución de Interact. Los valores pueden ser de tipo serie o numérico.	

Creación de tablas de Message Connector

Para desplegar IBM Unica Interact Message Connector, primero debe crear las tablas en la base de datos donde están almacenados los datos de tiempo de ejecución de Interact. Creará una tabla para cada nivel de audiencia que haya definido. Para cada nivel de audiencia, Interact utilizará las tablas que cree para registrar información sobre transacciones de Message Connector.

Utilice el cliente de base de datos para ejecutar el script SQL de Message Connector en la base de datos o el esquema apropiado para crear las tablas necesarias. Los scripts SQL para la base de datos soportada se instalan automáticamente al instalar el servidor de ejecución de Interact. Consulte las hojas de trabajo que ha completado en la publicación *IBM Unica Interact Guía de instalación* para ver detalles sobre cómo conectarse a la base de datos que contiene las tablas de ejecución de Interact.

1. Lance el cliente de base de datos y conéctese a la base de datos en la que están almacenadas actualmente las tablas de ejecución de Interact.
2. Ejecute el script adecuado en el directorio <*inicio_Interact*>/msgconnector/scripts/ddl (donde <*inicio_Interact*> es el directorio en el que ha instalado el tiempo de ejecución de Interact, por ejemplo, C:\Unica\Interact o /Unica/Interact). La lista siguiente muestra los scripts SQL de muestra que se pueden utilizar para crear manualmente tablas de Message Connector:

Tabla 25. Scripts de creación de tablas de Message Connector

Tipo de origen de datos	Nombre del script
IBM DB2	db_scheme_db2.sql
Microsoft SQL Server	db_scheme_sqlserver.sql
Oracle	db_scheme_oracle.sql

Tenga en cuenta que estos scripts se proporcionan como muestras. Es posible que utilice un convenio de denominación o estructura distinto para los valores de ID de audiencia, por lo que puede que necesite modificar el script antes de ejecutarlo. Normalmente, un método recomendado es tener una tabla dedicada para cada nivel de audiencia.

Las tablas se crean para contener la siguiente información:

Tabla 26. Información creada por los scripts SQL de muestra

Nombre de columna	Descripción
IDRegistro	Clave primaria de esta entrada.
IDMensaje	Identificador exclusivo de cada instancia de mensaje.
IDEnlace	Identificador exclusivo de cada enlace en el soporte electrónico (por ejemplo, un mensaje de correo electrónico).
URLImagenOferta	URL a la imagen relacionada de la oferta devuelta.

Tabla 26. Información creada por los scripts SQL de muestra (continuación)

Nombre de columna	Descripción
URLPáginaDestinoOferta	URL a la página de destino relacionada de la oferta devuelta.
CódigoTratamiento	Código de tratamiento de la oferta devuelta.
FechaCaducidadOferta	Fecha y hora de caducidad de la oferta devuelta.
FechaContactoOferta	Fecha y hora en la que la oferta se ha devuelto al cliente.
IDAudiencia	ID de audiencia del soporte electrónico.

Tenga en cuenta lo siguiente sobre esta tabla:

- En función del nivel de audiencia, habrá una columna IDAudiencia para cada componente de la clave de audiencia.
- La combinación de IDMensaje, IDEnlace e IDAudiencia forma una clave exclusiva de esta tabla.

Cuando los scripts hayan acabado de ejecutarse, se habrán creado las tablas necesarias para Message Connector.

Ahora está preparado para desplegar la aplicación web Message Connector.

Despliegue y ejecución de Message Connector

Message Connector de IBM Unica Interact se despliega como una aplicación web autónoma en un servidor de aplicaciones web soportado.

Para desplegar Message Connector, asegúrese de que se hayan completado las tareas siguientes:

- Debe haber instalado el servidor de ejecución de IBM Unica Interact. La aplicación Message Connector desplegable se instala automáticamente con el servidor de ejecución, y está lista para su despliegue desde el directorio de inicio de Interact.
- También debe haber ejecutado los scripts SQL proporcionados con la instalación para crear las tablas necesarias en la base de datos de ejecución de Interact para que las utilice Message Connector tal como se describe en "Creación de tablas de Message Connector" en la página 227

Exactamente de la misma forma en que se despliegan las aplicaciones de IBM Unica en un servidor de aplicaciones web antes de que se ejecuten, se debe desplegar la aplicación Message Connector para que esté disponible para proporcionar las ofertas.

1. Conéctese a la interfaz de gestión del servidor de aplicaciones web con los privilegios necesarios para desplegar una aplicación.
2. Siga las instrucciones del servidor de aplicaciones web para desplegar y ejecutar le archivo denominado <*inicio_Interact*>/msgconnector/MessageConnector.war Sustituya <*inicio_Interact*> por el directorio real en el que está instalado el servidor de ejecución de Interact, como por ejemplo C:\Unica\Interact, o /Unica/Interact.

Message Connector ahora estará disponible para su uso. Una vez que haya configurado la instalación de Interact para crear los datos subyacentes que utilizará Message Connector para proporcionar ofertas, como por ejemplo canales interactivos y estrategias, diagramas de flujo, ofertas, etc., puede crear los enlaces en el soporte electrónico que aceptará Message Connector.

Creación de los enlaces de Message Connector

Para utilizar Message Connector para proporcionar imágenes de oferta personalizadas cuando un usuario final interactúe con el soporte electrónico (por ejemplo, abriendo un mensaje de correo electrónico) y páginas de destino personalizadas cuando el usuario final pulse en la oferta, debe crear los enlaces que se incorporarán en el mensaje. Esta sección proporciona un resumen de las etiquetas HTML de estos enlaces.

Independientemente del sistema que utilice para generar los mensajes salientes a los usuarios finales, debe generar las etiquetas HTML para que contengan los campos adecuados (proporcionados en los códigos HTML como atributos) que contienen la información que desea pasar al servidor de ejecución de Interact. Siga los pasos siguientes para configurar la información mínima necesaria para un mensaje de Message Connector.

Tenga en cuenta que aunque las instrucciones aquí se refieren específicamente a mensajes que contienen enlaces de Message Connector, puede seguir los mismos pasos y la misma configuración para añadir enlaces a páginas web y otros soportes electrónicos.

1. Cree el enlace `IMG` que aparecerá en el mensaje con, como mínimo, los parámetros siguientes:
   - `IDmsj`, que indica el identificador exclusivo para este mensaje.
   - `IDEnlace`, que indica el identificador exclusivo para el enlace en el mensaje.
   - `IDAudiencia`, el identificador de la audiencia a la que pertenece el destinatario del mensaje.

     Tenga en cuenta que si el ID de audiencia es un ID compuesto, todos estos componentes se deben incluir en el enlace.

   También puede incluir parámetros opcionales, que incluyen nivel de audiencia, nombre de canal interactivo, nombre de punto de interacción, URL de ubicación de imagen y los parámetros de personalización no utilizados específicamente por Message Connector.
2. Opcionalmente, cree un enlace `A` que incluya un enlace `IMG` de forma que, cuando el usuario pulse la imagen, el navegador cargue una página que contenga la oferta para el usuario. El enlace `A` debe contener también los tres parámetros listados anteriormente (`IDmsj`, `IDEnlace` y `IDAudiencia`), además de los parámetros adicionales (nivel de audiencia, nombre de canal interactivo y nombre de punto interactivo) y parámetros personalizados no utilizados específicamente por Message Connector. Tenga en cuenta que el enlace `A` muy probablemente contendrá un enlace `IMG` de Message Connector, pero también puede aparecer sólo, según se requiera. Si el enlace sí contiene un enlace `IMG`, el enlace `IMG` podría contener el mismo conjunto de parámetros que el enlace `A` que lo incluye (incluidos los parámetros opcionales o personalizados).
3. Cuando los enlaces se hayan definido correctamente, genere y envíe los mensajes de correo electrónico.

Para obtener información detallada sobre los parámetros disponibles, y los enlaces de muestra, consulte “Parámetros de solicitud HTTP de etiqueta "IMG" y "A"”

Parámetros de solicitud HTTP de etiqueta "IMG" y "A"

Cuando Message Connector recibe una solicitud, ya sea porque un usuario final ha abierto un correo electrónico que contiene una etiqueta `IMG` codificada por Message Connector o porque el usuario final ha pulsado una etiqueta `A`, analiza los parámetros incluidos con la solicitud para devolver los datos de oferta adecuados.

Esta sección proporciona una lista de los parámetros que se pueden incluir en el URL solicitante (la etiqueta IMG (cargada automáticamente cuando se visualiza una imagen etiquetada al abrir un correo electrónico) o la etiqueta A (cargada cuando la persona que visualiza el correo electrónico pulsa el mensaje y va al sitio especificado).

Parámetros

Cuando Message Connector recibe una solicitud, analiza los parámetros incluidos con la solicitud. Estos parámetros incluyen algunos de los siguientes, o todos ellos:

Nombre de parámetro	Descripción	¿Necesario?	Valor predeterminado
IDmsj	Identificador exclusivo de la instancia de correo electrónico o página web.	Sí	Ninguno. Lo proporciona el sistema que crea la instancia exclusiva del mensaje de correo electrónico o página web que contiene la etiqueta.
IDEnlace	Identificador exclusivo del enlace de este correo electrónico o página web.	Sí	Ninguno. Lo proporciona el sistema que crea la instancia exclusiva del mensaje de correo electrónico o página web que contiene la etiqueta.
nivelAudiencia	Nivel de audiencia al que pertenece el destinatario de esta comunicación.	No	audienceLevel especificado como valor predeterminado en el elemento audienceLevels que se encuentra en el archivo MessageConnectorConfig.xml.
ic	Nombre del canal interactivo de destino (IC)	No	Valor del elemento interactiveChannel que se encuentra en la sección defaultParameterValues del archivo MessageConnectorConfig.xml, que es "interactiveChannel" de forma predeterminada.
ip	Nombre del punto de interacción (IP) que se aplica	No	Valor del elemento interactionPoint que se encuentra en la sección defaultParameterValues del archivo MessageConnectorConfig.xml, que es "headBanner" de forma predeterminada.
URLImagenoferta	URL de la imagen de oferta de destino para el URL IMG en el mensaje.	No	Ninguno.
AtrURLImagenOferta	Nombre del atributo de oferta que tiene el URL de la imagen de oferta de destino	No	Valor del elemento imageUrlAttribute que se encuentra en la sección defaultParameterValues del archivo MessageConnectorConfig.xml.
URLPáginaDestinoOferta	URL de la página de destino correspondiente a la oferta de destino.	No	Ninguno.
AtribPáginaDestinoOferta	Nombre del atributo de oferta que tiene el URL de la página de destino correspondiente a la oferta de destino.	No	Valor del elemento landingPageUrlAttribute que se encuentra en la sección defaultParameterValues del archivo MessageConnectorConfig.xml.
eventoContacto	Nombre del evento de contacto.	No	Valor del elemento contactEventName que se encuentra en la sección defaultParameterValues del archivo MessageConnectorConfig.xml, que es "contact" de forma predeterminada.
eventoRespuesta	Nombre del evento de aceptación.	No	Valor del elemento acceptEventName que se encuentra en la sección defaultParameterValues del archivo MessageConnectorConfig.xml, que es "accept" de forma predeterminada.
depuración	Indicador de depuración. Establezca este parámetro en "true" solo para la resolución de problemas y si se lo indica el personal de soporte técnico de IBM Unica .	No	Valor del elemento debugFlag que se encuentra en la sección defaultParameterValues del archivo MessageConnectorConfig.xml, que es "false" de forma predeterminada.

Nombre de parámetro	Descripción	¿Necesario?	Valor predeterminado
<ID de audiencia>	ID de audiencia de este usuario. El nombre de este parámetro se define en el archivo de configuración.	Sí	Ninguno.

Cuando Message Connector recibe un parámetro que no se reconoce (es decir, no aparece en la lista anterior, se maneja de dos posibles maneras:

- Si se proporciona un parámetro no reconocido (por ejemplo, "attribute", como en `attribute="attrValue"`) y hay un parámetro coincidente del mismo nombre más la palabra "Type" (por ejemplo, "attributeType", como en `attributeType="string"`), esto hace que Message Connector cree un parámetro de Interact coincidente y lo pase al tiempo de ejecución de Interact.

  Los valores del parámetro Type pueden ser los siguientes:

  – string
  – numeric
  – datetime

    Para un parámetro de tipo "datetime," Message Connector busca también un parámetro con el mismo nombre más la palabra "Pattern" (por ejemplo, "atributoPattern") cuyo valor es un formato válido de fecha/hora. Por ejemplo, podría proporciona el parámetro `attributePattern="MM/dd/aaaa"`.

    Tenga en cuenta que si especifica un tipo de parámetro de "datetime" pero no proporciona un patrón de fecha coincidente, se utilizará el valor especificado en el archivo de configuración de Message Connector (que se encuentra en `<directorio_instalación>/msgconnector/config/MessageConnectorConfig.xml`) en el servidor Interact.
- Si se proporciona un parámetro no reconocido y no hay ningún valor Type coincidente, Message Connector pasa ese parámetro al URL de redireccionamiento de destino.

Para todos los parámetros no reconocidos, Message Connector los pasa al servidor de ejecución de Interact sin procesarlos o guardarlos.

Código de Message Connector de ejemplo

La siguiente etiqueta `A` contiene un ejemplo de un conjunto de enlaces de Message Connector que podría aparecer en un mensaje de correo electrónico:

```
<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>
```

En este ejemplo, la etiqueta `IMG` se carga automáticamente cuando se abre el mensaje de correo electrónico. Recuperando la imagen de la página especificada, el mensaje pasa los parámetros para el identificador de mensaje exclusivo (IDmsj), identificador de enlace exclusivo (IDEnlace) e identificador de usuario exclusivo (ID_usuario), junto con dos parámetros adicionales (códigoIngresos y tipoIngresos) para que se pasen al tiempo de ejecución de Interact.

La etiqueta `A` proporciona el atributo HREF (Hypertext Reference) que convierte la imagen de oferta en un enlace del mensaje de correo electrónico que se puede pulsar. Si el visor del mensaje, al ver la imagen, pulsa y va a la página de destino, el identificador de mensaje exclusivo (`IDmsj`), identificador de enlace (`IDEnlace`) e

identificador de usuario (ID_usuario) se pasan al servidor, así como un parámetro adicional (recomendación) que se pasa al URL de redireccionamiento de destino.

Acerca de Interact Web Connector

Interact WebConnector (también denominado JavaScript Connector, o JSConnector) proporciona un servicio en el servidor de ejecución de Interact que permite que el código JavaScript llame a la API de Interact Java. Esto permite a las páginas web realizar llamadas a Interact para la personalización de ofertas en tiempo real utilizando solo código JavaScript incorporado, sin tener que basarse en lenguajes de desarrollo web (por ejemplo, Java, PHP, JSP, etc.). Por ejemplo, podría incorporar un pequeño fragmento de código JavaScript en cada página del sitio web que proporcione ofertas recomendadas por Interact de forma que cada vez que se cargue la página se realicen llamadas a la API de Interact para garantizar que se visualizan las mejores ofertas en la página de destino para el visitante del sitio.

Utilice Interact Web Connector en situaciones donde desee visualizar ofertas a los visitantes de una página donde es posible que no tenga control programático en el lado del servidor sobre la visualización de la página (como tendría, por ejemplo, con PHP u otros scripts basados en el servidor), pero aún puede incorporar código JavaScript en el contenido de la página que ejecutará el navegador web del visitante.

Consejo: Los archivos de Interact Web Connector se instalan automáticamente en el servidor de ejecución de Interact, en el directorio <*inicio_Interact*>/jsconnector. En el directorio <*inicio_Interact*>/jsconnector, encontrará un archivo ReadMe.txt que contiene notas importantes y detalles sobre las características de Web Connector, así como archivos de muestra y el código fuente de Web Connector que se utilizará como base para desarrollar sus propias soluciones. Si no encuentra aquí información que responda a sus preguntas, consulte el directorio jsconnector para obtener más información.

Instalación de Web Connector en el servidor de ejecución

Una instancia de Web Connector se instala automáticamente con el servidor de ejecución de IBM Unica Interact y se habilita de forma predeterminada. Sin embargo, existen algunos valores que debe modificar para configurar y utilizar Web Connector.

Los valores que debe modificar para poder utilizar el Web Connector instalado en el servidor de ejecución se añaden a la configuración del servidor de aplicaciones web. Por este motivo, necesitará reiniciar el servidor de aplicaciones web después de completar estos pasos.

1. Para el servidor de aplicaciones web en el que está instalado el servidor de ejecución de Interact, establezca las siguientes propiedades Java:

   -DUI_JSCONNECTOR_ENABLE_INPROCESS=true

   -DUI_JSCONNECTOR_HOME=<*inicio_jsconnector*>

   Sustituya <*inicio_jsconnector*> por la ruta al directorio jsconnector del servidor de ejecución, que es <directorio_instalación_Interact>/jsconnector.

   Por ejemplo, en una instalación Windows, podría ser C:\Unica\Interact\jsconnector. En un sistema UNIX, podría especificar /Unica/Interact/jsconnector para este valor.

La manera en la que establece las propiedades Java depende del servidor de aplicaciones web. Por ejemplo, en WebLogic, editaría el archivo `startWebLogic.sh` o `startWebLogic.cmd` para actualizar el valor `JAVA_OPTIONS`, como en este ejemplo:

```
JAVA_OPTIONS="${SAVE_JAVA_OPTIONS} -DUI_JSCONNECTOR_HOME=/UnicaFiles/
jsconnector"
```

En WebSphere Application Server, establecería esta propiedad en el panel de la máquina virtual Java de la consola de administración.

Consulte la documentación del servidor de aplicaciones web para ver instrucciones específicas sobre cómo establecer las propiedades Java.

2. Reinicie el servidor de aplicaciones web si ya está en ejecución, o inicie ahora el servidor de aplicaciones web, para asegurarse de que se utilizan las nuevas propiedades Java.

Cuando el servidor de aplicaciones web haya completado su proceso de inicio, habrá finalizado la instalación de Web Connector en el servidor de ejecución. El paso siguiente será conectarse a esta página web de Configuración en `http://<host>:<puerto>/interact/jsp/WebConnector.jsp`, donde *<host>* es el nombre de servidor de ejecución de Interact, y *<puerto>* es el puerto en el que Web Connector está a la escucha, tal como ha especificado el servidor de aplicaciones web.

Instalación de Web Connector como aplicación web independiente

Una instancia de Web Connector se instala automáticamente con el servidor de ejecución de IBM Unica Interact y se habilita de forma predeterminada. Sin embargo, también puede desplegar Web Connector como su propia aplicación web (por ejemplo, en un servidor de aplicaciones web en un sistema individual) y configurarlo para comunicarse con el servidor de ejecución de Interact remoto.

Estas instrucciones describen el proceso de desplegar Web Connector como una aplicación web individual con acceso a un servidor de ejecución de Interact remoto.

Para desplegar Web Connector, debe tener instalado el servidor de ejecución de IBM Unica Interact y debe tener un servidor de aplicaciones web en otro sistema con acceso a la red (no bloqueado por ningún cortafuegos) al servidor de ejecución de Interact.

1. Copie el directorio `jsconnector` que contiene los archivos de Web Connector del servidor de ejecución de Interact en el sistema donde el servidor de aplicaciones web (por ejemplo, WebSphere Application Server) ya está configurado y en ejecución. Puede encontrar el directorio `jsconnector` en el directorio de instalación de Interact, por ejemplo, `C:\Unica\Interact` o `/Unica/Interact`.
2. En el sistema donde desplegará la instancia de Web Connector, configure el archivo `jsconnector/jsconnector.xml` utilizando cualquier editor de texto o XML para modificar el atributo `interactURL`.

   Se establece de forma predeterminada en `http://localhost:7001/interact`, pero debe modificarlo para que coincida con el URL del servidor de ejecución remoto de Interact, como por ejemplo `http://runtime.example.com:7011/interact`.

Una vez que haya desplegado Web Connector, podrá utilizar una interfaz web para personalizar los demás valores del archivo jsconnector.xml. Consulte "Configuración de Web Connector" para obtener más información.

3. Para el servidor de aplicaciones web en el que desplegará Web Connector, establezca la siguiente propiedad Java:

   -DUI_JSCONNECTOR_HOME=<*inicio_jsconnector*>

   Sustituya <*inicio_jsconnector*> por la ruta real donde ha copiado el directorio jsconnector en el servidor de aplicaciones web.

   Por ejemplo, en una instalación Windows, podría ser C:\Unica\Interact\jsconnector. En un sistema UNIX, podría especificar /Unica/Interact/jsconnector para este valor.

   La manera en la que establece las propiedades Java depende del servidor de aplicaciones web. Por ejemplo, en WebLogic, editaría el archivo startWebLogic.sh o startWebLogic.cmd para actualizar el valor JAVA_OPTIONS, como en este ejemplo:

   JAVA_OPTIONS="${SAVE_JAVA_OPTIONS} -DUI_JSCONNECTOR_HOME=/UnicaFiles/jsconnector"

   En WebSphere Application Server, establecería esta propiedad en el panel de la máquina virtual Java de la consola de administración.

   Consulte la documentación del servidor de aplicaciones web para ver instrucciones específicas sobre cómo establecer las propiedades Java.

4. Reinicie el servidor de aplicaciones web si ya estaba en ejecución, o inícielo en este paso, para asegurarse de que se utiliza la nueva propiedad Java.

   Espere a que el servidor de aplicaciones web complete su proceso de inicio antes de continuar.

5. Conéctese a la interfaz de gestión del servidor de aplicaciones web con los privilegios necesarios para desplegar una aplicación.
6. Siga las instrucciones del servidor de aplicaciones web para desplegar y ejecutar el archivo siguiente: jsConnector/jsConnector.war

Web Connector ahora está desplegado en la aplicación web. Una vez que tenga el servidor Interact completamente configurado activo y en ejecución, el paso siguiente es conectarse a la página de Configuración de Web Connector en http://<*host*>:<*puerto*>/interact/jsp/WebConnector.jsp, donde <*host*> es el sistema que ejecuta el servidor de aplicaciones web en el que acaba de desplegar Web Connector y <*puerto*> es el puerto en el que Web Connector está a la escucha, tal como especifica el servidor de aplicaciones web.

Configuración de Web Connector

Los valores de configuración de Interact Web Connector se almacenan en un archivo denominado jsconnector.xml que se almacena en el sistema donde está desplegado Web Connector (por ejemplo, el propio servidor de ejecución de Interact o en un sistema distinto que ejecute el servidor de aplicaciones web). Puede editar el archivo jsconnector.xml directamente utilizando cualquier editor de texto o XML; sin embargo, una forma fácil de configurar casi todos los valores de configuración disponibles es utilizar la página de configuración de Web Connector desde el navegador web.

Para utilizar la interfaz web para configurar Web Connector, debe instalar y desplegar la aplicación web que proporciona Web Connector. En el servidor de ejecución de Interact se instala automáticamente una instancia de Web Connector al instalar y desplegar Interact. En cualquier otro servidor de aplicaciones web, debe

instalar y desplegar la aplicación web de Web Connector tal como se describe en "Instalación de Web Connector como aplicación web independiente" en la página 233.

1. Abra el navegador web soportado y abra un URL similar al siguiente:

   `http://<host>:<puerto>/interact/jsp/WebConnector.jsp`

   - Sustituya *<host>* por el servidor en el que se ejecuta Web Connector, por ejemplo, el nombre de host del servidor de ejecución o el nombre del servidor en el que se ha desplegado una instancia individual de Web Connector.
   - Sustituya *<puerto>* por el número de puerto en el que la aplicación web de Web Connector está a la escucha de conexiones, el cual normalmente coincide con el puerto predeterminado del servidor de aplicaciones web.
2. En la página Configuraciones que aparece, complete las secciones siguientes:

Tabla 27. Resumen de valores de la configuración de Web Connector.

Sección	Configuración
Valores básicos	Utilice la página Valores básicos para configurar el comportamiento global de Web Connector para el sitio en el que desplegará las páginas etiquetadas. Estos valores incluyen el URL base del sitio, información que requiere utilizar Interact sobre los visitantes del sitio, y valores similares que se aplican a todas las páginas que tiene previsto etiquetar con el código de Web Connector. Consulte "Opciones básicas de la configuración de WebConnector" en la página 237 para obtener más detalles.
Tipos de visualización HTML	Utilice la página Tipos de visualización HTML para determinar el código HTML que se proporcionará para cada punto de interacción en la página. Puede elegir en la lista de plantillas predeterminadas (archivos .flt) que contienen alguna combinación de código de hoja de estilo en cascada (CSS), código HTML y código Javascript para utilizar para cada punto de interacción. Puede utilizar las plantillas tal como se proporcionan, personalizarlas según se requiera o crear las suyas propias. Los valores de configuración de esta página se corresponden con la sección `interactionPoints` del archivo de configuración `jsconnector.xml`. Consulte "Tipos de visualización HTML de la configuración de WebConnector" en la página 238 para obtener más detalles.
Páginas ampliadas	Utilice la página Páginas ampliadas para correlacionar valores específicos de página con un patrón de URL. Por ejemplo, podría configurar una correlación de página como por ejemplo que cualquier URL que contuviera el texto "index.htm" se visualizara en la página de bienvenida general, con eventos de carga de página específicos y puntos de interacción definidos para esa correlación. Los valores de configuración de esta página se corresponden con la sección `pageMapping` del archivo de configuración `jsconnector.xml`. Consulte "Páginas ampliadas de configuración de WebConnector" en la página 241 para obtener más detalles.

3. En la página Valores básicos, verifique que los valores del sitio sean válidos para la instalación y, opcionalmente, especifique la modalidad de depuración (no recomendada a menos que está realizando la resolución de un problema), la integración de etiqueta de página de NetInsight y los valores predeterminados

para la mayoría de puntos de interacción y, a continuación, pulse el enlace Tipos de visualización HTML en Configuraciones.

4. En la página Tipos de visualización HTML, siga estos pasos para añadir o modificar plantillas de visualización que definan los puntos de interacción en la página web del cliente.

   De forma predeterminada, las plantillas de visualización (archivos .flt) se almacenan en *<inicio_jsconnector>*/conf/html.

   a. Seleccione en la lista el archivo .flt que desee examinar o utilizar como punto de partida, o pulse Añadir un tipo para crear una nueva plantilla de punto de interacción en blanco para utilizar.

      La información sobre el contenido de la plantilla, si la hay, aparece junto a la lista de plantillas.

   b. Opcionalmente, modifique el nombre de la plantilla en el campo **Nombre de archivo para este tipo de visualización**. En el caso de una plantilla nueva, actualice CHANGE_ME.flt a algo más descriptivo.

      Si renombra aquí la plantilla, Web Connector creará un nuevo archivo con ese nombre la próxima vez que se guarde la plantilla. Las plantillas se guardan cuando se modifica el cuerpo del texto y a continuación se pasa a otro campo.

   c. Modifique o complete la información de Fragmento HTML según se requiera, incluyendo el código de hojas de estilo (CSS), JavaScript y HTML que desee incluir. Tenga en cuenta que también puede incluir variables que sustituyan parámetros de Interact en tiempo de ejecución. Por ejemplo, ${offer.HighlightTitle} se sustituye automáticamente por el título de la oferta en la ubicación especificada del punto de interacción.

      Utilice los ejemplos que aparecen a continuación del campo Fragmento HTML para ver indicaciones sobre cómo dar formato a los bloques de código CSS, JavaScript o HTML.

5. Utilice la página Páginas ampliadas según se requiera para configurar las correlaciones de página que determinan cómo se manejan patrones de URL específicos en las páginas.

6. Cuando haya acabado de establecer las propiedades de configuración, pulse **Desplegar los cambios**. Al pulsar **Desplegar los cambios** se realizarán las acciones siguientes:

   - Se visualiza la etiqueta de página de IBM Unica Interact Web Connector, que contiene el código JavaScript que se puede copiar de la página de Web Connector e insertar en sus páginas web.
   - Se realiza una copia de seguridad del archivo de configuración existente de Web Connector en el servidor Interact (el archivo jsconnector.xml en el servidor donde está instalado Web Connector) y se crea un nuevo archivo de configuración con los valores que ha definido.

     Los archivos de configuración de copia de seguridad se almacenan en *<inicio_jsconnector>*/conf/archive/jsconnector.xml.*<fecha>*.*<hora>*, como en jsconnector.xml.20111113.214933.750-0500 (donde la serie de fecha es 20111113 y la serie de hora, incluyendo el indicador de huso horario, es 214933.750-0500)

Ahora ha completado la configuración de Web Connector.

Para modificar la configuración, puede volver al inicio de estos pasos y realizarlos de nuevo con nuevos valores, o puede abrir el archivo de configuración (<inicio_Interact>/jsconnector/conf/jsconnector.xml) en cualquier editor de texto o XML y modificarlo según se requiera.

Opciones básicas de la configuración de WebConnector

Utilice la página Valores básicos de la página Configuraciones de Web Connector para configurar el comportamiento global de Web Connector para el sitio en el que estará desplegando las páginas etiquetadas. Estos valores incluyen el URL base del sitio, información que requiere utilizar Interact sobre los visitantes del sitio, y valores similares que se aplican a todas las páginas que tiene previsto etiquetar con el código de Web Connector.

Valores del sitio

Las opciones de configuración de Valores del sitio son valores globales que afectan al comportamiento global de la instalación del Web Connector que está configurando. Puede especificar los valores siguientes:

Tabla 28. Valores del sitio para la instalación de Web Connector

Valor	Descripción	Valor equivalente en jsconnector.xml
URL de la API de Interact	URL base del servidor de ejecución de Interact. **Nota:** Este valor sólo se utiliza si Web Connector no se ejecuta en el servidor de ejecución de Interact (es decir, lo ha desplegado por separado).	*<URLinteract>*
URL de Web Connector	URL base utilizado para generar el URL de pulsación.	*<URLjsConnector>*
Nombre de canal interactivo para el sitio web de destino	Nombre del canal interactivo que se ha definido en el servidor Interact que representa esta correlación de página.	*<canalInteractivo>*
Nivel de audiencia de visitantes	El nivel de audiencia de Campaign para el visitante entrante; se utiliza en la llamada de la API al tiempo de ejecución de Interact.	*<nivelAudiencia>*
Nombre de campo de ID de audiencia en la tabla de perfil	Nombre del campo IDAudiencia que se utilizará en la llamada de la API a Interact. Tenga en cuenta que actualmente no se da soporte a identificadores de audiencia de varios campos.	*<campoIDAudiencia>*
Tipo de datos del campo de ID de audiencia	Tipo de datos del campo de ID de audiencia ("numérico" o "serie") que se utilizará en la llamada de la API a Interact.	*<tipoCampoIDAudiencia>*
Nombre de cookie que representa el ID de sesión	Nombre de la cookie que contendrá el ID de sesión.	*<cookieIDsesión>*
Nombre de cookie que representa el ID de visitante	Nombre de la cookie que contendrá el ID de visitante.	*<cookieIDvisitante>*

Características opcionales

Las opciones de configuración de Características opcionales son valores globales opcionales de la instalación del Web Connector que está configurando. Puede especificar los valores siguientes:

Tabla 29. Valores opcionales del sitio para la instalación de Web Connector

Valor	Descripción	Valor equivalente en jsconnector.xml
Habilitar modalidad de depuración	Especifica (con una respuesta de `sí` o `no`) si se utilizará una modalidad de depuración especial. Si habilita esta característica, el contenido devuelto de Web Connector incluye una llamada de Javascript a 'alerta', que informa al cliente de la correlación de página determinada que se acaba de producir. El cliente debe tener una entrada en el archivo especificado por el valor *<authorizedDebugClients>* para obtener la alerta.	*<habilitarModalidadDepuración>*
Archivo autorizado de host de clientes de depuración	Ruta de un archivo que contiene la lista de hosts o direcciones IP (Internet Protocol) que califican para modalidad de depuración. Para que se pueda recopilar la información de depuración, el nombre de host o la dirección IP de un cliente debe aparecer en el archivo especificado.	*<clientesDepuraciónAutorizados>*
Habilitar integración de etiqueta de página de NetInsight	Especifica (con una respuesta de `sí` o `no`) si Web Connector debe adjuntar la etiqueta especificada de IBM Unica NetInsight al final del contenido de la página.	`<habilitarEtiquetasNetInsight>`
Archivo de plantilla HTML de etiqueta de NetInsight	Plantilla HTML/Javascript utilizada para integrar una llamada con la etiqueta de NetInsight. Normalmente, debe aceptar el valor predeterminado a menos que se le indique que proporcione una plantilla distinta.	`<etiquetaNetInsight>`

Tipos de visualización HTML de la configuración de WebConnector

Utilice la página Tipos de visualización HTML para determinar el código HTML que se proporcionará para cada punto de interacción en la página. Puede elegir en la lista de plantillas predeterminadas (archivos .flt) que contienen alguna combinación de código de hoja de estilo en cascada (CSS), código HTML y código JavaScript para utilizar para cada punto de interacción. Puede utilizar las plantillas tal como se proporcionan, personalizarlas según se requiera o crear las suyas propias.

Nota: Los valores de configuración de esta página se corresponden con la sección `interactionPoints` del archivo de configuración `jsconnector.xml`.

El punto de interacción también puede contener marcadores de posición (zonas) en las que se pueden soltar automáticamente atributos de oferta. Por ejemplo, podría incluir `${offer.TREATMENT_CODE}`, que se sustituiría con el código de tratamiento asignado a esa oferta durante la interacción.

Las plantillas que aparecen en esta página se cargan automáticamente desde los archivos almacenados en el directorio `<Interact_home>/jsconnector/conf/html` del servidor de Web Connector. Las nuevas plantillas que cree aquí se almacenarán también en ese directorio.

Para utilizar la página Tipos de visualización HTML para visualizar o modificar cualquiera de las plantillas existentes, seleccione el archivo `.flt` en la lista.

Para crear una nueva plantilla en la página Tipo de visualización HTML, pulse **Añadir un tipo**.

Independientemente del método que elija para crear o modificar una plantilla, la siguiente información aparecerá junto a la lista de plantillas:

Valor	Descripción	Valor equivalente en jsconnector.xml
Nombre de archivo para este tipo de visualización	Nombre asignado a la plantilla que está editando. Este nombre debe ser válido para el sistema operativo en el que se ejecuta Web Connector; por ejemplo, no puede utilizar una barra inclinada (/) en el nombre si el sistema operativo es Microsoft Windows. Si está creando una nueva plantilla, este campo está preestablecido en `CHANGE_ME.flt`. Debe cambiarlo a un valor significativo antes de continuar.	*<fragmento_html>*

Valor	Descripción	Valor equivalente en jsconnector.xml
Fragmento HTML	Contenido específico que Web Connector debe insertar en el punto de interacción en la página web. Este fragmento puede contener código HTML, información de formato CSS o JavaScript que se ejecutará en la página. Cada uno de estos tres tipos de contenido debe estar entre los códigos BEGIN y END, como en los ejemplos siguientes: • ${BEGIN_HTML} <*su contenido HTML*> ${END_HTML} • ${BEGIN_CSS} <*su información de hoja de estilo específica del punto de interacción*> ${END_CSS} • ${BEGIN_JAVASCRIPT} <*su código JavaScript específico del punto de interacción*> ${END_JAVASCRIPT} Puede especificar también diversos códigos especiales predefinidos que se sustituyen automáticamente cuando se carga la página, incluyendo lo siguiente: • `${logAsAccept}`: una macro que toma dos parámetros (un URL de destino, y el TreatmentCode utilizado para identificar la aceptación de la oferta) y lo sustituye por el URL de pulsación. • `${offer.AbsoluteLandingPageURL}` • `${offer.OFFER_CODE}` • `${offer.TREATMENT_CODE}` • `${offer.TextVersion}` • `$offer.AbsoluteBannerURL}` Cada uno de los códigos de oferta que se lista aquí representa atributos de oferta definidos en la plantilla de oferta en IBM Unica Campaign que ha utilizado el usuario de marketing para crear las ofertas que devuelve Interact. Tenga en cuenta que Web Connector utiliza un motor de plantillas denominado FreeMarker que proporciona muchas opciones adicionales que puede encontrar útiles al configurar códigos en las plantillas de páginas. Consulte http://freemarker.org/docs/index.html para obtener más información.	No hay ningún equivalente porque el fragmento HTML reside en su propio archivo aparte de jsconnector.xml.

Valor	Descripción	Valor equivalente en jsconnector.xml
Códigos especiales de ejemplo	Contiene ejemplos de tipo de códigos especiales, que incluyen códigos que identifican bloques como HTML, CSS o JAVASCRIPT, y zonas para soltar que se pueden insertar para hacer referencia a metadatos de oferta específicos.	Sin equivalente.

Los cambios que realice en esta página se guardarán automáticamente al navegar a otra página de configuración de Web Connector.

Páginas ampliadas de configuración de WebConnector

Utilice la página Páginas ampliadas para correlacionar valores específicos de página con un patrón de URL. Por ejemplo, podría configurar una correlación de página como por ejemplo que cualquier URL entrante que contuviera el texto "index.htm" se visualiza en la página de bienvenida general, con eventos de carga de página específicos y puntos de interacción definidos para esa correlación.

Nota: Los valores de configuración de esta página se corresponden con la sección `pageMapping` del archivo de configuración `jsconnector.xml`.

Para utilizar la página Páginas ampliadas para crear una nueva correlación de página, pulse el enlace **Añadir una página** y complete la información necesaria para la correlación.

Información sobre la página

Las opciones de configuración de Información sobre la página para la correlación de página definen el patrón de URL que actúa como el desencadenante de esta correlación, además de algunos valores adicionales para la manera en la que Interact maneja esta correlación de página.

Valor	Descripción	Valor equivalente en jsconnector.xml
URL contiene	Es el patrón de URL que debe vigilar Web Connector en la solicitud de página entrante. Por ejemplo, si el URL solicitante contiene "mortgage.htm", podría correlacionarlo con su página de información de hipoteca.	*<patrón_url>*
Nombre descriptivo de esta página o conjunto de páginas	Nombre descriptivo para su propia referencia que describe para qué es esta correlación de página, por ejemplo, "Página de información de hipoteca".	*<nombre_descriptivo>*
También devolver ofertas como datos JSON para uso de JavaScript	Lista desplegable para indicar si desea que Web Connector incluya los datos de oferta sin formato con el formato de notación de objeto JavaScript (http://www.json.org/) al final del contenido de la página.	*<habilitarRetornoDatosSinFormato>*

Eventos a activar (desencadenar) cuando se realice una visita a esta página o conjunto de páginas

Este conjunto de opciones de configuración de la correlación de página define el patrón de URL que actúa como desencadenante de esta correlación, además de algunos valores adicionales para la manera en la que Interact maneja esta correlación de página.

Nota: Los valores de configuración de esta sección se corresponden con la sección `<pageLoadEvents>` de `jsconnector.xml`.

Valor	Descripción	Valor equivalente en jsconnector.xml
Eventos individuales	Lista de eventos disponibles para esta página o este conjunto de páginas. Los eventos de esta lista son los que ha definido en Interact. Seleccione uno o más eventos que desee que se produzcan cuando se cargue la página. La secuencia de las llamadas a la API de Interact es la siguiente: 1. `startSession` 2. `postEvent` para cada evento de carga de página individual (siempre que haya definido los eventos individuales en Interact) 3. Para cada punto de interacción: • `getOffers` • `postEvent(ContactEvent)`	*<evento>*

Puntos de interacción (ubicaciones de visualización de oferta) en esta página o este conjunto de páginas

Este conjunto de opciones de configuración para la correlación de página le permite seleccionar qué puntos de interacción aparecerán en la página de Interact.

Nota: Los valores de configuración de esta sección se corresponden con la sección `<correlaciónPágina> | <página> | <puntosInteracción>` de `jsconnector.xml`.

Valor	Descripción	Valor equivalente en jsconnector.xml
Casilla de verificación del nombre del punto de interacción	Cada punto de interacción que se ha definido en el archivo de configuración aparece en esta sección de la página. Si selecciona la casilla de verificación junto al nombre del punto de interacción, se visualizará un número de opciones disponibles para ese punto de interacción.	*<puntoInteracción>*
ID de elemento HTML (Interact establecerá innerHTML)	Nombre del elemento HTML que debe recibir el contenido para este punto de interacción. Por ejemplo, si ha especificado `<div id="welcomebanner">` en la página, especificaría `welcomebanner` (el valor de ID) en este campo.	*<IDElementoHTML>*

Valor	Descripción	Valor equivalente en jsconnector.xml
Tipo de visualización HTML	Lista desplegable que le permite seleccionar el tipo de visualización HTML (los fragmentos HTML, o archivos .flt, definidos en una página de configuración anterior de Web Connector) para utilizar para este punto de interacción.	*<fragmento_html>*
Número máximo de ofertas a presentar (si se trata de un carrusel o flipbook)	Número máximo de ofertas que debe recibir Web Connector del servidor Interact para este punto de interacción. Este campo es opcional y se aplica solo para un punto de interacción que actualiza regularmente las ofertas presentadas sin volver a cargar la página, como en el escenario de carrusel donde se recuperan varias ofertas de forma que se pueden hacer disponibles una cada vez.	*<NúmeroMáxOfertas>*
Evento a activar cuando se presente la oferta	Nombre del evento de contacto que se publicará para este punto de interacción.	*<eventoContacto>*
Evento a activar cuando se acepte la oferta	Nombre del evento de aceptación que se publicará para este punto de interacción cuando se pulse el enlace de la oferta.	*<eventoAceptación>*
Evento a activar cuando se rechace la oferta	Nombre del evento de rechazo que se publicará para este punto de interacción. **Nota:** En este momento esta característica no se utiliza aún.	*<eventoRechazo>*

Opciones de configuración de Web Connector

Normalmente, puede utilizar una interfaz gráfica de Web Connector para configurar los valores de Web Connector. Todos los valores que especifica se almacenan también en un archivo denominado jsconnector.xml, que se encuentra en el directorio `jsconnector/conf`. Aquí se describen todos los parámetros que se guardan en el archivo de configuración jsconnector.xml.

Parámetros y sus descripciones

Los parámetros siguientes se almacenan en el archivo `jsconnector.xml` y se utilizan para las interacciones de Web Connector. Hay dos maneras de modificar estos valores:

- Mediante la página web de Configuración de Web Connector que está automáticamente disponible una vez que se ha desplegado e iniciado la aplicación Web Connector. Para utilizar la página web de Configuración, utilice el navegador web para abrir un URL similar al siguiente: `http://<host>:<puerto>/interact/jsp/WebConnector.jsp`.

  Los cambios que realiza en la página web de Administración se almacenan en el archivo `jsconnector.xml` en el servidor donde está desplegado Web Connector.
- Edite el archivo `jsconnector.xml` directamente mediante cualquier editor de texto o XML. Asegúrese de que se siente cómodo editando etiquetas y valores XML antes de utilizar este método.

Nota: Cada vez que edita manualmente el archivo `jsconnector.xml`, puede volver a cargar estos valores abriendo la página de Administración de Web Connector (que se encuentra en `http://<host>:<puerto>/interact/jsp/jsconnector.jsp`) y pulsando **Volver a cargar configuración**.

La tabla siguiente describe las opciones de configuración que se pueden establecer cuando aparecen en el archivo `jsconnector.xml`.

Tabla 30. Opciones de configuración de Web Connector

Grupo de parámetros	Parámetro	Descripción
`defaultPageBehavior`		
	`friendlyName`	Identificador legible por personas para el patrón de URL para que se visualice en la página de configuración web de Web Connector.
	`interactURL`	URL base del servidor de ejecución de Interact. Nota: Debe establecer este parámetro solo si el servicio de Web Connector (jsconnector) se ejecuta como aplicación web desplegada. No necesita establecer este parámetro si WebConnector se ejecuta automáticamente como parte del servidor de ejecución de Interact.
	`jsConnectorURL`	URL base utilizado para generar el URL de pulsación, por ejemplo, `http://host:puerto/jsconnector/clickThru`
	`interactiveChannel`	Nombre del canal interactivo que representa esta correlación de página.
	`sessionIdCookie`	Nombre de la cookie que contiene el ID de sesión que se utiliza en las llamadas de la API a Interact.
	`visitorIdCookie`	Nombre de la cookie que contiene el ID de audiencia.
	`audienceLevel`	Nivel de audiencia de campaña para el visitante entrante, utilizado en la llamada de la API al tiempo de ejecución de Interact.
	`audienceIdField`	Nombre del campo `IDAudiencia` utilizado en la llamada de la API al tiempo de ejecución de Interact. **Nota:** Nota: Actualmente no hay soporte para identificadores de audiencia de varios campos.
	`audienceIdFieldType`	Tipo de datos del campo de ID de audiencia [`numérico | serie`] utilizado en la llamada de la API al tiempo de ejecución de Interact
	`audienceLevelCookie`	Nombre de la cookie que contendrá el nivel de audiencia. Es opcional. Si no establece este parámetro, el sistema utiliza lo que se define para `audienceLevel`.
	`relyOnExistingSession`	Se utiliza en la llamada de la API al tiempo de ejecución de Interact. Normalmente, este parámetro se establece en "true".

Tabla 30. Opciones de configuración de Web Connector (continuación)

Grupo de parámetros	Parámetro	Descripción
	enableInteractAPIDebug	Se utiliza en la llamada de la API al tiempo de ejecución de Interact para habilitar la salida de depuración a los archivos de registro.
	pageLoadEvents	Evento que se publicará una vez que se cargue esta página determinada. Especifique uno o más eventos en esta etiqueta, con un formato similar a <evento>evento1</evento>.
	interactionPointValues	Todos los elementos de esta categoría actúan como valores predeterminados para los valores que faltan en las categorías específicas de puntos de interacción.
	interactionPointValuescontactEvent	Nombre predeterminado del evento de contacto que se publicará para este punto de interacción específico.
	interactionPointValuesacceptEvent	Nombre predeterminado del evento de aceptación que se publicará para este punto de interacción específico.
	interactionPointValuesrejectEvent	Nombre predeterminado del evento de rechazo que se publicará para este punto de interacción específico. (Nota: en este momento, esta característica no se utiliza.)
	interactionPointValueshtmlSnippet	Nombre predeterminado de la plantilla HTML que se utilizará para este punto de interacción.
	interactionPointValuesmaxNumberOfOffers	Número máx. predeterminado de ofertas que se recuperarán de Interact para este punto de interacción.
	interactionPointValueshtmlElementId	Nombre predeterminado del elemento HTML que recibirá el contenido de este punto de interacción.
	interactionPoints	Esta categoría contiene la configuración de cada punto de interacción. Para las propiedades que faltan, el sistema se basará en lo que se ha configurado en la categoría interactionPointValues.
	interactionPointname	Nombre del punto de interacción (IP).
	interactionPointcontactEvent	Nombre del evento de contacto que se publicará para este punto de contacto específico.
	interactionPointacceptEvent	Nombre del evento de aceptación que se publicará para este punto de contacto específico.
	interactionPointrejectEvent	Nombre del evento de rechazo que se publicará para este punto de contacto específico. (Tenga en cuenta que esta característica no se utiliza aún.)
	interactionPointhtmlSnippet	Nombre de la plantilla HTML que se utilizará para este punto de interacción.

Tabla 30. Opciones de configuración de Web Connector (continuación)

Grupo de parámetros	Parámetro	Descripción
	interactionPointmaxNumberOfOffers	Número máximo de ofertas que se recuperarán de Interact para este punto de interacción
	interactionPointhtmlElementId	Nombre del elemento HTML para recuperar el contenido para este punto de interacción.
	enableDebugMode	Indicador booleano (valores aceptables: true o false) para activar modalidad de depuración especial. Si se establece en true, el contenido devuelto de Web Connector incluye una llamada de JavaScript a 'alerta' que informa al cliente sobre la correlación de página determinada que se acaba de producir. El cliente debe tener una entrada en el archivo authorizedDebugClients para generar la alerta.
	authorizedDebugClients	Archivo utilizado por la modalidad de depuración especial que consta de una lista de nombres de host o direcciones IP (IP) que califican para la modalidad de depuración.
	enableRawDataReturn	Indicador booleano (valores aceptables: true o false) para determinar si Web Connector se conecta a datos de oferta sin formato con formato JSON al final del contenido.
	enableNetInsightTagging	Indicador booleano (valores aceptables: true o false) para determinar si Web Connector se conecta al indicador de NetInsight al final del contenido.
	apiSequence	Representa una implementación de la interfaz de APISequence, que dicta la secuencia de las llamadas de la API realizadas por Web Connector cuando se llama a un pageTag. De forma predeterminada, la implementación utiliza una secuencia de StartSession, pageLoadEvents, getOffers y logContact, donde los dos últimos son específicos de cada punto de interacción.
	clickThruApiSequence	Representa una implementación de la interfaz de APISequence, que dicta una secuencia de las llamadas de la API realizadas por Web Connector cuando se llama a un clickThru. De forma predeterminada, la implementación utiliza una secuencia de StartSession y logAccept.
	netInsightTag	Representa la plantilla HTML y JavaScript utilizada para integrar una llamada con la etiqueta de NetInsight. Normalmente, no debería ser necesario cambiar esta opción.

Utilización de la página de administración de Web Connector

Web Connector incluye una página de administración que proporciona algunas herramientas para ayudarle a gestionar y probar la configuración tal como se

podría utilizar con patrones de URL específicos. También puede utilizar la página de administración para volver a cargar la configuración que ha modificado.

Acerca de la página de administración

Mediante el navegador web soportado, puede abrir `http://host:puerto/interact/jsp/jsconnector.jsp`, donde *host:puerto* es el nombre de host en el que se ejecuta Web Connector y el puerto en el que está a la escucha de conexiones, por ejemplo, `runtime.example.com:7001`

Puede utilizar la página de administración de cualquiera de las formas siguientes:

Tabla 31. Opciones de la página de administración de Web Connector

Opción	Finalidad
Volver a cargar configuración	Pulse el enlace **Volver a cargar configuración** para volver a cargar en la memoria los cambios de configuración que se hayan guardado en disco. Esto es necesario si ha realizado cambios directamente en el archivo de configuracion jsconnector.xml del conector web en lugar de mediante las páginas web de configuración.
Ver configuración	Visualice la configuración de WebConnector en función del patrón de URL que especifique en el campo **Ver configuración**. Cuando especifique el URL de una página y pulse **Ver configuración**, Web Connector devolverá la configuración que utilizará el sistema en función de esta correlación de patrón. Si no se encuentra ninguna coincidencia, se devolverá la configuración predeterminada. Esto es útil para probar si se está utilizando la configuración correcta para una página determinada.
Ejecutar etiqueta de página	Si se completan los campos de esta página y se pulsa **Ejecutar etiqueta de página**, Web Connector devolverá el resultado `pageTag` en función del patrón de URL. Esto simula la llamada a una etiqueta de página. La diferencia entre llamar a pageTag desde esta herramienta y utilizando un sitio web real es que si se utiliza esta página de administración se visualizarán los errores o excepciones. En el caso de un sitio web real, las excepciones no se devolverán y serán visibles solo en el archivo de registro de Web Connector.

Página de Web Connector de muestra

Por ejemplo, se ha incluido con Interact Web Connector un archivo denominado testPage.html que muestra cuántas características de Web Connector se etiquetarían en una página. Por comodidad, esta misma página de muestra se presenta aquí.

Página HTML de Web Connector de muestra

```
<?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[
/* ###############################################################
Esta es una página de prueba que contiene pageTag de WebConnector.  Dado que
el nombre de este archivo tiene incorporado TestPage, WebConnector detectará una
coincidencia de patrón de URL con el patrón de URL "testpage" de la versión
```

```
predeterminada de jsconnector.xml - aquí se aplicará la definición de configuración
correlacionada con ese patrón de URL "testpage". Esto significa que esta
página debería tener los ID de elemento HTML correspondientes a las IP de este
patrón de URL (es decir, 'welcomebanner', 'crosssellcarousel' y 'textservicemessage')
###################################################################### */

/* ##################################################################
Esta sección establece las cookies para sessionId y visitorId.
Tenga en cuenta que en un sitio web de producción real, esto lo hace
muy probablemente el componente de inicio de sesión. En el caso de la
prueba, se hace aquí... el nombre de la cookie debe coincidir con lo
que se ha configurado en 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("IDSesión","123");
     setCookie("IDCliente","1");

   /* ##################################################
   Ahora se configuran los ID de elemento HTML correspondientes a las IP
   ################################################## */
     document.writeln("<div id='welcomebanner'> Esto se debe cambiar, "
  + "si no algo es incorrecto <\/div>");
     document.writeln("<div id='crosssellcarousel'> Esto se debe cambiar, "
  + "si no algo es incorrecto <\/div>");
     document.writeln("<div id='textservicemessage'> Esto se debe cambiar, "
  + "si no algo es incorrecto <\/div>");
   //]]&gt;
</script><!--
##############################################################################
Esto es lo que se pega del archivo pageTag.txt en el directorio conf de la
instalación de WebConnector... la var unicaWebConnectorBaseURL se debe
ajustar para adecuarse al entorno local de WebConnector
##############################################################################
-->
<!-- BEGIN: Unica Interact Web Connector Page Tag -->
<!-- Copyright 2011, IBM Corporation All rights reserved. -->
<script language="javascript" type="text/javascript">
//<![CDATA[
   var unicaWebConnectorBaseURL = "http://localhost:7001/interact/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>Página de Interact Web Connector de ejemplo</title>
   </head>
   <body>
   <!-- END: Unica Interact Web Connector Page Tag -->
   <!--
################################################################################
end of pageTag paste
################################################################################
-->
   </body>
</html>
```

Apéndice E. Recomendaciones de productos integradas de Interact e Intelligent Offer

IBM Unica Interact se puede combinar con IBM Coremetrics Intelligent Offer para proporcionar recomendaciones de productos dirigidas por Interact. Ambos productos pueden proporcionar recomendaciones de productos para ofertas, pero utilizando métodos diferentes. Intelligent Offer utiliza el comportamiento web de un visitante (filtrado de colaboración) para crear correlaciones entre visitantes y ofertas recomendadas. Interact se basa en el comportamiento pasado del cliente, atributos, historial, y menos en las ofertas a nivel de visualización, y aprende qué ofertas se ajustan mejor al perfil de comportamiento de un cliente (basándose en datos demográficos y otra información sobre el cliente). Las tasas de aceptación de ofertas ayudan a crear un modelo predictivo a través del autoaprendizaje. Cuando se utilizan ambos productos, Interact puede utilizar un perfil personal para definir ofertas que pasarán un ID de categoría a Intelligent Offer y recuperarán productos recomendados basándose en su popularidad para ser mostrados al visitante como parte de las ofertas seleccionadas. Esto puede proporcionar mejores recomendaciones para los clientes que darán como resultado más pulsaciones de enlaces y mejores resultados que la actuación en solitario de cualquiera de los dos productos.

Las secciones siguientes describen cómo funciona esta integración, y cómo utilizar la aplicación de muestra proporcionada para crear su propia integración de oferta personalizada.

Visión general de la integración de Interact con Intelligent Offer

Esta sección describe cómo IBM Unica Interact se puede integrar con IBM Coremetrics Intelligent Offer para proporcionar recomendaciones de productos dirigidas por Interact, e incluye una descripción del proceso y los mecanismos mediante los cuales se realiza la integración.

IBM Unica Interact se integra con IBM Coremetrics Intelligent Offer a través de la API REST (Representational State Transfer), que se proporciona al instalar Intelligent Offer. Mediante la realización de llamadas de la API REST con el ID de categoría adecuado, Interact puede recuperar productos recomendados e incluirlos en la información sobre ofertas que se muestra en la página personalizada que está viendo el visitante.

Cuando un visitante abre el URL de la página web (tal como la página JSP de muestra que se incluye con la instalación de Interact), la página llama a Interact para buscar una oferta. Si la oferta se ha configurado dentro de Interact con los parámetros correctos, en el caso más simple tienen lugar los pasos siguientes:

1. La lógica de página identifica el ID de cliente del visitante.
2. Se realiza una llamada de API a Interact y se pasa la información necesaria para generar una oferta para ese cliente.
3. La oferta devuelta proporciona la página web con al menos tres atributos: el URL de la imagen de la oferta, el URL de la página de destino cuando el cliente pulsa el enlace y el ID de categoría que se debe utilizar para determinar qué productos recomendar.

4. El ID de categoría se utiliza para llamar a Intelligent Offer a fin de recuperar los productos recomendados. Estos productos están en formato JSON (JavaScript Object Notation) y ordenados por nivel de ventas dentro de esa categoría.
5. La oferta y los productos se visualizan en el navegador del visitante.

Esta integración es útil para combinar juntos la recomendación de ofertas y las recomendaciones de productos. Por ejemplo, en una página web puede tener dos puntos de interacción: uno para una oferta, y otro para las recomendaciones correspondientes a esa oferta. Para conseguir esto, la página web hace una llamada a Interact para realizar una segmentación en tiempo real y determinar la mejor oferta (por ejemplo, para un descuento del 10% en todos los electrodomésticos pequeños). Cuando la página recibe la oferta desde Interact, esa oferta contiene el ID de categoría (en este ejemplo, para electrodomésticos pequeños). La página pasa el ID de categoría de electrodomésticos pequeños a Intelligent Offer utilizando una llamada de API, y en respuesta recibe las mejores recomendaciones de productos para esa categoría de acuerdo con el nivel de ventas.

En un caso más simple, una página web hace una llamada a Interact sólo para encontrar una categoría (por ejemplo, cubertería de gama alta) que coincida con el perfil del cliente. La página pasa el ID de categoría recibido a Intelligent Offer y en respuesta recibe las recomendaciones de productos de cubertería.

Requisitos previos de la integración

Antes de poder utilizar la integración Intelligent Offer - Interact, compruebe que se cumplan los requisitos previos descritos en esta sección.

Asegúrese de que se cumplan los siguientes requisitos previos:

- Está familiarizado con el uso de la API de Interact tal como está documentada en la *Guía del administrador* y la ayuda en línea.
- Está familiarizado con la API REST de Intelligent Offer tal como está descrita en la documentación del desarrollador de Intelligent Offer.
- Tiene un conocimiento básico de HTML, JavaScript, CSS y JSON (JavaScript Object Notation).

  JSON es importante porque la API de REST de Intelligent Offer devuelve la información solicitada sobre productos en forma de datos con formato JSON.
- Está familiarizado con la codificación de páginas web en el extremo servidor, pues la aplicación de demostración proporcionada con Interact utiliza JSP (aunque JSP no es necesario).
- Tiene una cuenta válida de Intelligent Offer y la lista de los ID de categoría que piensa transferir a Interact para recuperar recomendaciones de productos (productos más vendidos o más populares dentro de la categoría especificada).
- Tiene el enlace de la API REST de Intelligent Offer (un URL correspondiente el entorno utilizado de Intelligent Offer).

  Consulte la aplicación de muestra que se incluye con la instalación de Interact para obtener un ejemplo, o consulte el código de muestra en "Utilización del proyecto de muestra de integración" en la página 254 para obtener más información.

Configuración de una oferta para la integración con Intelligent Offer

Para que su página web pueda invocar IBM Coremetrics Intelligent Offer a fin de recuperar un producto recomendado, primero debe configurar la oferta de IBM Unica Interact con la información necesaria a fin de pasarla a Intelligent Offer.

Para configurar una oferta a fin de enlazarla con Intelligent Offer, compruebe primero que se cumplen las condiciones siguientes:

- Asegúrese de que el servidor de ejecución de Interact se ha configurado y se ejecuta correctamente.
- Asegúrese de que el servidor de ejecución puede establecer una conexión con el servidor de Intelligent Offer, y que el cortafuegos no impide el establecimiento de una conexión web estándar de salida (puerto 80).

Para configurar una oferta para la integración con Intelligent Offer, realice los pasos siguientes.

1. Cree o edite una oferta para Interact.

   Para obtener información sobre cómo crear y modificar ofertas, consulte la publicación *IBM Unica Interact User's Guide*, y la documentación de IBM Unica Campaign.
2. Además de los otros valores contenidos en la oferta, asegúrese de que la oferta incluye los atributos de oferta siguientes:
   - El URL (localizador universal de recursos) que enlaza con la imagen correspondiente a la oferta.
   - El URL que enlaza con la página de destino de la oferta.
   - Un ID de categoría de Intelligent Offer asociado a la oferta.

     Puede recuperar el ID de categoría manualmente desde la configuración de Intelligent Offer. Interact no puede recuperar valores de ID de categoría directamente.

   En la aplicación web de demostración que se incluye con la instalación de Interact, estos atributos de oferta se denominan `ImageURL`, `ClickThruURL` y `CategoryID`. Puede utilizar cualquier nombre que sea descriptivo para usted, siempre que la aplicación web concuerde con los valores esperados por la oferta.

   Por ejemplo, puede definir un oferta llamada "10PercentOff" que contiene esos atributos, donde el ID de categoría (tal como se recupera de la configuración de Intelligent Offer) es `PROD1161127`, el URL de la oferta es `http://www.example.com/success`, y el URL de la imagen que se debe mostrar para la oferta es `http://localhost:7001/sampleIO/img/10PercentOffer.jpg` (un URL que, en este caso, es local respecto del servidor de ejecución de Interact).
3. Defina las reglas de tratamiento de un canal interactivo para incluir esta oferta, y despliegue el canal interactivo de la forma habitual.

La oferta está ahora definida con la información necesaria para la integración con Intelligent Offer. El trabajo restante para permitir que Intelligent Offer proporcione recomendaciones de productos a Interact se realiza configurando páginas web para realizar las llamadas de API apropiadas.

Cuando configure la aplicación web para presentar la página integrada a los visitantes, asegúrese de que los archivos siguientes estén incluidos en el directorio `WEB-INF/lib`:

- *`Interact_Home`*`/lib/interact_client.jar`, archivo necesario para gestionar las llamadas realizadas desde la página web a la API de Interact.

- *Interact_Home*/lib/JSON4J_Apache.jar, archivo necesario para gestionar los datos devueltos por la llamada a la API REST de Intelligent Offer, que devuelve datos con formato JSON.

Consulte "Utilización del proyecto de muestra de integración" para obtener más información sobre cómo presentar las ofertas a los clientes.

Utilización del proyecto de muestra de integración

Cada instalación de tiempo de ejecución de Interact incluye un proyecto de muestra que describe el proceso de la integración Intelligent Offer - Interact. El proyecto de muestra describe cómo crear una página web que llama a una oferta que contiene un ID de categoría, el cual se pasa a Intelligent Offer para recuperar una lista de productos recomendados para su presentación en los puntos de interacción de la página.

Visión general

Puede utilizar el proyecto de muestra tal como se proporciona si desea probar el proceso de integración, o puede utilizar el proyecto como punto de partida para desarrollar sus propias páginas personalizadas. El proyecto de muestra se encuentra en el archivo siguiente:

directorio_inicio_Interact/samples/IntelligentOfferIntegration/MySampleStore.jsp

Este archivo, además de contener un ejemplo completo funcional del proceso de integración, también contiene comentarios extensos que describen qué debe configurar en Interact, qué debe personalizar en el archivo .jsp , y cómo desplegar la página correctamente para ejecutarla con la instalación.

MySampleStore.jsp

Para su comodidad, el archivo MySampleStore.jsp se muestra aquí. Este ejemplo se puede actualizar en versiones posteriores de Interact; utilice pues el archivo que se incluye con la instalación como punto de partida para cualquier ejemplo que necesite.

```
<!--
# ********************************************************************************
#  Licensed Materials - Property of IBM
#  Unica 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 IntelligentOffer.
```

```
*
* 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 IntelligentOffer 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 : IntelligentOffer 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 Intelligent Offers 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 IO 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
 *   Intelligent Offer
 *****************************************************************************/

 /*****************************************************************************
 * Call IntelligentOffer 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());
    }
%>
```

Contactar con el soporte técnico de IBM Unica

Si encuentra un problema que no puede resolver consultando la documentación, el contacto de soporte indicado de la compañía puede registrar una llamada con el soporte técnico de IBM Unica . Utilice la información de esta sección para asegurarse de que el problema se resuelve eficaz y satisfactoriamente.

Si no es un contacto de soporte asignado de la compañía, póngase en contacto con el administrador de IBM Unica para obtener información.

Información que se debe reunir

Antes de ponerse en contacto con el soporte técnico de IBM Unica , reúna la siguiente información:

- Una breve descripción de la naturaleza del problema.
- Mensajes de error detallados para ver cuándo se produce el problema.
- Pasos detallados para reproducir el problema.
- Archivos de registro, archivos de sesión, archivos de configuración y archivos de datos relacionados.
- Información acerca del producto y del entorno del sistema, que puede obtener según se describe en "Información del sistema".

Información del sistema

Cuando llame al soporte técnico de IBM Unica , es posible que se le solicite que proporcione información sobre el entorno.

Si el problema no le impide iniciar sesión, gran parte de esta información está disponible en la página Acerca de, que proporciona información sobre las aplicaciones IBM Unica instaladas.

Puede acceder a la página Acerca de seleccionado **Ayuda > Acerca de**. Si no se puede acceder a la página Acerca de, puede obtener el número de versión de cualquier aplicación IBM Unica visualizando el archivo `version.txt` ubicado en el directorio de instalación de cada aplicación.

Información de contacto para el soporte técnico de IBM Unica

Para ver las maneras de contactar con el soporte técnico de IBM Unica , consulte el sitio web de soporte técnico del producto de IBM Unica : (http://www.unica.com/about/product-technical-support.htm).

Avisos

Esta información se ha desarrollado para productos y servicios ofrecidos en EE.UU.

Es posible que IBM no ofrezca en otros países los productos, servicios o características que se describen en este documento. Consulte con el representante de IBM de su localidad si desea información sobre los productos y servicios disponibles actualmente en su zona. Cualquier referencia a un producto, programa o servicio de IBM no pretende afirmar ni implicar que solamente se pueda utilizar ese producto, programa o servicio de IBM. En su lugar, se puede utilizar cualquier producto, programa o servicio funcionalmente equivalente que no infrinja ninguno de los derechos de propiedad intelectual de IBM. Sin embargo, es responsabilidad del usuario evaluar y verificar el funcionamiento de cualquier producto, programa o servicio que no sea de IBM.

IBM puede tener patentes o solicitudes de patentes pendientes que cubran la materia descrita en este documento. El abastecimiento de este documento no confiere ninguna licencia sobre dichas patentes. Puede enviar consultas sobre licencias, por escrito, a:

IBM Director of Licensing
IBM Corporation
North Castle Drive
Armonk, NY 10504-1785
EE.UU.

Para consultas sobre licencias relacionadas con información de doble byte (DBCS), póngase en contacto con el departamento de propiedad intelectual de IBM de su país o envíe sus consultas, por escrito, a:

Intellectual Property Licensing
Legal and Intellectual Property Law
IBM Japan Ltd.
1623-14, Shimotsuruma, Yamato-shi
Kanagawa 242-8502 Japón

El párrafo siguiente no se aplica al Reino Unido ni a ningún otro país donde estas disposiciones sean incompatibles con la legislación vigente: INTERNATIONAL BUSINESS MACHINES CORPORATION PROPORCIONA ESTA PUBLICACIÓN "TAL CUAL" SIN GARANTÍAS DE NINGÚN TIPO, NI EXPLÍCITAS NI IMPLÍCITAS, INCLUYENDO PERO NO LIMITÁNDOSE A ELLAS, LAS GARANTÍAS IMPLÍCITAS DE NO INFRACCIÓN DE DERECHOS DE TERCEROS, COMERCIALIZACIÓN O IDONEIDAD PARA UN FIN CONCRETO. Algunos países no permiten la renuncia a garantías explícitas o implícitas en determinadas transacciones, por lo que puede que esta declaración no sea aplicable en su caso.

Esta información puede incluir imprecisiones técnicas o errores tipográficos. Periódicamente se realizan cambios en la información aquí contenida; estos cambios se incorporarán en nuevas ediciones de la publicación. IBM puede efectuar mejoras y/o cambios en los productos y/o programas descritos en esta publicación en cualquier momento y sin previo aviso.

Cualquier referencia incluida en esta información a sitios web que no sean de IBM sólo se proporciona para comodidad del usuario y en ningún modo constituye una aprobación de dichos sitios web. El material de esos sitios web no forma parte del material de este producto de IBM y la utilización de esos sitios web corre a cuenta y riesgo del usuario.

IBM puede utilizar o distribuir cualquier información que se le proporcione en la forma que considere adecuada, sin incurrir por ello en ninguna obligación para con el remitente.

Los titulares de licencias de este programa que deseen obtener información sobre el mismo con el fin de permitir: (i) el intercambio de información entre programas creados independientemente y otros programas (incluido éste) y (ii) el uso mutuo de información que se haya intercambiado, deben ponerse en contacto con:

IBM Corporation
170 Tracer Lane
Waltham, MA 02451
EE.UU.

Dicha información puede estar disponible, sujeta a los términos y condiciones correspondientes, incluyendo, en algunos casos, el pago de una tarifa.

El programa bajo licencia que se describe en este documento y todo el material bajo licencia disponible los proporciona IBM bajo los términos de las Condiciones Generales de IBM, Acuerdo Internacional de Programas Bajo Licencia de IBM o cualquier acuerdo equivalente entre las partes.

Los datos de rendimiento contenidos aquí se han determinado en un entorno controlado. Por lo tanto, los resultados que se obtengan en otros entornos operativos pueden variar significativamente. Es posible que algunas medidas se hayan realizado en sistemas en nivel de desarrollo, y no existen garantías de que estas medidas sean las mismas en los sistemas de disponibilidad general. Además, algunas medidas se pueden haber estimado mediante extrapolación. Los resultados reales pueden variar. Los usuarios de este documento deberían verificar los datos aplicables para sus entornos específicos.

La información relativa a productos que no son de IBM se ha obtenido de los proveedores de estos productos, sus anuncios publicados y otras fuentes públicamente disponibles. IBM no ha probado esos productos y no puede confirmar la exactitud del rendimiento, de la compatibilidad ni de ninguna otra declaración relacionada con productos que no sean de IBM. Las consultas sobre las prestaciones de productos que no sean de IBM se deben dirigir a los proveedores de esos productos.

Todas las declaraciones relativas a la orientación o intención futura de IBM están sujetas a cambio o anulación sin previo aviso y representan solamente metas y objetivos.

Todos los precios de IBM mostrados son precios actuales de venta al por menor sugeridos por IBM y sujetos a modificaciones sin previo aviso. Los precios de venta pueden variar.

Esta información contiene ejemplos de datos e informes utilizados en las operaciones comerciales diarias. Para ilustrar estas operaciones de forma tan completa como sea posible, los ejemplos incluyen nombres de personas,

compañías, marcas y productos. Todos estos nombres son ficticios y cualquier similitud con nombres y direcciones utilizados por una compañía real es totalmente casual.

LICENCIA DE COPYRIGHT:

Esta información contiene programas de aplicación de muestra en lenguaje fuente, que ilustran las técnicas de programación en diversas plataformas operativas. Puede copiar, modificar y distribuir estos programas de muestra de cualquier forma, sin pagar nada a IBM, con los fines de desarrollar, utilizar, comercializar o distribuir programas de aplicación de acuerdo con la interfaz de programación de aplicaciones para la plataforma operativa para la cual se han escrito los programas de muestra. Estos ejemplos no se han probado exhaustivamente en todas las condiciones. Por lo tanto, IBM no puede garantizar ni dar por sentada la fiabilidad, capacidad de servicio o el funcionamiento de estos programas. Los programas de muestra se proporcionan "TAL CUAL" sin garantías de ningún tipo. IBM no será responsable de ningún daño resultante de la utilización de los programas de muestra por parte del usuario.

Si está viendo esta información en copia software, es posible que las fotografías y las ilustraciones en color no aparezcan.

Marcas registradas

IBM, el logotipo de IBM e ibm.com son marcas registradas o marcas comerciales registradas de International Business Machines Corp., registrada en muchas jurisdicciones en todo el mundo. Otros nombres de servicios y productos podrían ser marcas registradas de IBM u otras compañías. Hay disponible una lista actual de marcas registradas de IBM en la Web en “Información de marca registrada y copyright en ”www.ibm.com/legal/copytrade.shtml.

Impreso en España