Unica JourneyV12.1.1管理者ガイド

Contents

Chapter 1. 紹介のUnica Journey

Unica Journeyは、コンテキスト主導型のパーソナライズされたマルチステップのオムニチャネル カスタマー エクスペリエンスを作成、実行、視覚化するための目標ベースのオーケストレーション ソリューションです。

マーケターが使用できるUnica Journeyに:

- カスタマー エクスペリエンスの目標を定義する
- ジャーニーをリアルタイムで簡単に調整して達成する
- 洗練された直感的なツールを使用して、チャネル/タッチポイントおよびイベント全体のカスタマー ジャーニー全体を作成および視覚化します。Journeyキャンバス

カスタマー ジャーニーは完全に自動化され、顧客のブランド エンゲージメントのすべてのステップと同期されます。内でリアルタイムのインサイトを使用するJourney顧客の行動を理解するJourney.

特徴Unica Journey

以下の通り特徴Unica Journey:

- **目標主導のエクスペリエンス**: カスタマー エクスペリエンスの目標を定義し、ジャーニーをリアルタイムで簡単に調整して達成します。
- **オーケストレーション キャンバス**: 洗練された直感的な操作で、チャネル/タッチポイントおよびイベント全体のカスタマー ジャーニー全体を作成および視覚化します。Journeyキャンバス。
- **Always on Engagement** : 顧客のブランド エンゲージメントのすべてのステップと同期する、完全に自動化された実行。
- **リアルタイムのインサイト**: カスタマー ジャーニーでの出来事を反映したインサイトを使用して、顧客の行動を理解します。
- **タッチポイントの選択**: デジタル チャネル用のすぐに使用できるネイティブ タッチポイントを活用するか、カスタム タッチポイントを作成して、エコ システム全体のジャーニーをシームレスに調整します。
- **動的データ フレームワーク**: 柔軟なデータ定義とエントリ ソースにより、複数のタッチポイントからのさまざまな形式 (ファイル、API など) のコンテキスト データとイベントでカスタマー ジャーニーを強化します。

メリットのUnica Journey

利点Unica Journey以下の通り:

- **ブランド ロイヤルティの向上**: ターゲットを絞った自動化されたジャーニーにより、顧客を獲得、育成、変換、維持することで、ブランド フォローを強化します。
- 強化された**オムニ チャネル エンゲージメント**: アウトバウンドのネイティブ統合により、チャネル全体で一貫したカスタマー エクスペリエンスを提供します (Unica Campaign) およびインバウンド エンゲージメント (Unica Interact 、Unica Discover 、 とUnica Deliver) 。
- **顧客コンバージョン サイクルを短縮する**: 一歩先を行き、タイムリーなネクスト ベスト アクションで顧客を目標に導きます。

- **瞬間に反応する**: 顧客が旅のどこにいるかを知る機会を逃さず、適切な体験で顧客を喜ばせます。
- **マーケティングの TCO**の削減 : 自動化されたフローとプラグ アンド プレイによる MarTech エコシステムへの統合により、マーケティングの TCO を削減します。Unica Link .

Chapter 2. Unica Journey積算

Unica Journey電子メール用実行エンジン

Unica Journeyは、メール配信のためにUnica Deliver とUnica Link をサポートしています。Journey との連携には、どちらを使ってもかまいません。

Unica JourneyとともにUnica Link

Unica Link電子メール、SMS、CRM、ADTECH、およびJDBCチャネルを介して通信を送信する機能を提供します。Unica Link電子メール、SMS、CRM、ADTECH、およびJDBCチャネルに通信を配信するための次の参照コネクタを提供します。

以下のリファレンスコネクタをお好みで取り付けてください。

- **MailChimp**- Eメール用
- **Mandrill**- 電子メール用
- **Twilio**- SMS用
- **Salesforce**- CRM向け

Unica Link との統合により、Journey は、メール、SMS、CRM、ADTECH、JDBC 実行のみ、あらゆるサードパーティベンダーと統合することができます。

Table 1. のインストールと設定Unica Link

タスク	資料
のインストールと設定Unica Link	*Unica Link* V12.1 インストールガイド」を参照してください。
のUnica Link コネクタアプリをインストールします。Journey	*Unica Link* V12.1 インストールガイド」を参照してください。
Unica Link コネクタのインストール - MailChimp	*Unica Link* Mailchimp Connectorユーザーガイドをご覧ください。
Unica Link コネクタの取り付け - Mandrill	*Unica Link* Mandrill Connector ユーザーガイド を参照してください。
Unica Link コネクタのインストール - Twilio	*Unica Link* Twilio Connectorユーザーガイドをご覧ください。
Unica Link コネクタのインストール - Salesforce	*Unica Link* Salesforce Connector ユーザーガイドを参照してください。

Note: HCLは、これらのデリバリーチャネルベンダーのアカウントやアクセスを提供するものではありません。あなたの好みに基づいて、これらのベンダーから権利やアカウントを取得することができます。

Unica JourneyとともにUnica Deliver

Unica Journeyは、Unica Deliver の機能を利用して、電子メール通信を送信します。これはまた、リアルタイムでメールの応答をキャプチャし、聴衆を処理するのに役立ちますJourney.Unica Deliver とUnica Journey の統合を有効にするための詳細については、*Unica Journey* インストールガイドを参照してください。

Unica Journeyとの統合を実現し、Unica Campaign Unica Interact

Unica Journey Unica Campaign およびUnica Interact とシームレスに統合されます。Unica Campaign およびUnica Interact は、特定の Kafka トピックでUnica Journey に視聴者データを送信します。視聴者データはKafkaのエントリーソースを通じて送信され、これらのエントリーソースからデータを消費するすべてのジャーニーにプッシュされます。

Unica Campaign Unica Journey と の連携については、以下のドキュメントマップに記載されているガイドを参照してください。Unica Interact

Journeyは、Campaignの複数のパーティションからのデータをサポートします。

ジャーニーサポートデータを複数のパーティションに分割してキャンペーンを実施。

1. Journeyアプリケーションは、マルチパーティションに対応していません。
2. ジャーニーで処理できるのは、Campaign/Interact/Deliverの複数のパーティションのデータのみです。このジャーニーでは、1つのパーティションで実行されます。

構成プラットフォームとユーザーの役割と権限を変更する必要があります。

- エントリーソースの下に表示されるキャンペーンフローチャートの詳細は、複数のパーティションからのものです。
- パーティションに基づき、メール/SMS/WhatsAppのタッチポイントにDeliverテンプレートが表示されます。

Table 2. Unica Campaign と他のHCL製品との統合

タスク	資料
Unica Campaign の統合とUnica Journey	*Unica Campaign* 管理者ガイド、*Unica Campaign* ユーザーガイドを参照してください。
Unica Campaign の統合とUnica Interact	*Unica Interact* 管理者ガイドを参照してください。

Unica JourneyUnica Discoverとの連携

Unica Journeyユニカディスカバーとシームレスに統合。Unica Discoverは、オーディエンスの闘争データをUnica Journeyに送信します。視聴者データはRESTの入力ソースを通じて送信され、これらの入力ソースからデータを消費するすべてのジャーニーにプッシュされます。4つのスクリプトが提供されますので、Journeyをインストールした後、すぐにスクリプトを実行する必要があります。これにより、2つのエントリソースと、CART用Discover Entryソースとフォーム用Discover Entryソースという2つのデータ定義が作成されます。

DD名	カート
説明	顧客がカートや選択した商品セットを放棄した場合、このイベントをトリガーすることができます。

Table 3. 送信する属性

名前	タイプ	長さ	注記
電子メール※1	TEXT	200	これは必須フィールドです。
名前	TEXT	200	
DiscoverSessionId	TEXT	50	Discover Session IDはリンクバックするために必要です。
CartId	TEXT	50	カートを識別するための一意なID。
カートバリュー	NUMBER		
イベント日時	TIMESTAMP		イベントの日付と時刻（UTC） 経度
イベントタイプ	TEXT		イベントタイプはCART_-ABANDONEDにすることができます。
クッキーID	TEXT	1024	
TLT_BROWSER	TEXT	50	ブラウザの詳細
TLT_MODEL	TEXT	50	デバイスの詳細
HTTP_ACCEPT_LANGUAGE	TEXT	50	言語

DD名	Form
説明	顧客がWebフォームに入力すると、このイベントを公開することができます。

Table 4. 送信する属性

名前	タイプ	長さ	注記
電子メール※1	TEXT	200	これは必須フィールドです。
名前	TEXT	200	
DiscoverSessionId	TEXT	50	Discover Session IDはリンクバックするために必要です。

Table 4. 送信する属性 (continued)

名前	タイプ	長さ	注記
FormId	TEXT	50	フォームを識別するための一意なID
フォーム名	TEXT	100	
イベント日時	TIMESTAMP		イベントの日付と時刻（UTC） 経度
クッキーID	TEXT	1024	
TLT_BROWSER	TEXT	50	ブラウザの詳細
TLT_MODEL	TEXT	50	デバイスの詳細
HTTP_ACCEPT_LANGUAGE	TEXT	50	言語
イベントタイプ	TEXT		イベントタイプはFORM_SUBMITTED, FORM_ABANDONEDのいずれかになります。

Note: Fixpack 3以降では、Unica JourneyとUnica Discover機能の連携が可能になります。

紹介のUnica Deliver

Unica Deliverは、Web ベースのエンタープライズ規模のマーケティング メッセージ ソリューションであり、アウトバウンド バルク メッセージングおよびトランザクション メッセージング キャンペーンを実施するために使用できます。Deliverと統合しますUnica Campaignによってホストされている安全なメッセージ作成、送信、および追跡リソースを使用します。Unica .

Deliverパーソナライズされた電子メール通信を作成、送信、および追跡するため使用できます。Deliverをインストールして操作するのでCampaign、フローチャートを使ってCampaign受信者情報を正確に選択およびセグメント化し、各メッセージをカスタマイズすることができます。

オーディエンスを選択します

Campaign使用する各メッセージをパーソナライズするために使用できるメッセージ受信者と各人に関するデータを選択します。

Deliverを使用すると、多数の電子メール受信者にすばやく個別に連絡できます。ただし、トランザクションに応答して単一の電子メール メッセージを自動的に送信するようにメーリングを構成することもできます。

メッセージを作成します

DeliverのDocument Composer には、パーソナライズされたメッセージ コンテンツのデザイン、プレビュー、および公開に使用できる編集ツールが用意されています。Document Composer にアップロードするコンテン

ツ、または外部コンテンツへのリンクを含むメッセージを作成できます。Deliverメッセージを作成して送信します。Deliverは、各受信者の個人データに基づいて条件付きでコンテンツを表示するメッセージを設計する方法をいくつか提供します。

メッセージを送信して応答を追跡する

目標に応じて、メッセージング キャンペーンをできるだけ早く実行するようにスケジュールしたり、後で実行するようにスケジュールしたりできます。Deliverメッセージ配信を監視し、受信者の応答を追跡します。システムは、連絡先と応答データをDeliverの一部としてインストールされるシステム テーブルCampaignデータベース スキーマ。

どうやって始めるのか

開始するには、インストールする必要がありますCampaignホステッド メッセージング アカウントを持っていること。

システム管理者は、ホストされたメッセージング アカウントを要求し、操作する必要がありますUnicaリモート メッセージングおよび追跡システムへの安全なアクセスを構成します。一部のメッセージング機能は、リクエストがあった場合にのみ利用できますUnica.ホステッド メッセージング アカウントの確立とへのアクセスの構成の詳細については、Unicaホステッド メッセージングについては、*Unica Deliver*スタートアップおよび管理者ガイド。

Kafkaの統合

Journey ノードに対して、Unica Platform で Kafka を設定する必要があります。

でKafka_Configurationsにアクセスする。Unica Platform

Kafka_Configurationsにアクセスするには、次の手順を実行します。

1. Unica Platform で、 **設定方法 > 設定**次の場所に移動します。
2. ノード **Journey**を展開する。
3. **Kafka_Configurationsを**選択します。
4. **Edit settings**を選択します。

CommunicationMechanism 値に基づく必須構成。

(Kafka_Configurations)ページで、CommunicationMechanism フィールドに次の値のいずれかを選択できます。

- NO_SASLPLAINTEXT_SSL
- SASL_PLAINTEXT
- SSL
- SASL_PLAINTEXT_SSL

選択した内容に応じて、以下の項目が必須となります。

フィールド名	NO_SASLPLAIN TEXT_SSL	SASL_PLAIN TEXT	SSL	SASL_PLAIN TEXT_SSL
KafkaBrokerURL	はい	はい	はい	はい
トピック名	はい	はい	はい	はい
sasl.mechanism		はい		はい
ユーザーForKafkaData		はい	はい	はい
sasl.jaas.config.data ソース		はい		はい
truststore.location			はい	はい
truststore.password.data ソース			はい	はい
keystore.location			はい	はい
keystore.password.data ソース			はい	はい
key.password.dataSource			オプション	オプション
ssl.endpoint.identification. algorithm			はい	はい

必要な設定を行い、[**変更を保存**]をクリックします。

Note: Kafkaログファイルのサイズが大きいため、ディスクストレージが不足し、Kafkaサーバーを突然シャットダウンしたこと。

Chapter 3. Journeyの始まりと終わりのプロセス

About this task

プロセスを開始

1. プロセスウェブの開始
    a. KafkaとZookeeperの設定
        i. IP - Zookeeper/Kafkaが動作しているIP。
        ii. PORT- Kafka（デフォルト9092）、Zookeeperデフォルトポート2181
        iii. ログのパス
        iv. auto.create.topic.enable = true, このプロパティは、Engine Publishサービスを動作させるためにtrueに設定する必要があります。
    b. Zookeeperを起動し、10秒待つ
    c. kafkaを起動する
    d. configure Journey.xml –(Doc、Doc2参照)
    e. configure フォルダ下の Log4j2.xml を設定する。
    f. ウェブサーバー（JBOSS/TOMCAT/WebSphere）の起動
    g. Start Journey Web アプリケーション
2. エンジン始動
    a. application.portiesの設定
        i. DBの詳細を追加
        ii. Kafkaの詳細を追加する（例：spring.kafka.bootstrap-servers=127.0.0.1:9092, 127.0.0.2:9092)
            - Ignite Storageのパス、spring.ignite.storage.path、エンジンを実行するユーザがIgniteフォルダのパスを読み書きできること。
        iii. configure フォルダ下の Log4j2.xml を設定する。
        iv. プロパティ spring.ignite.ipFinder.List を以下のように設定します。
            - `spring.ignite.ipFinder.List=127.0.0.1:63501,127.0.0.1:63502, 127.0.0.1:63503,127.0.0.1:63504`
        v. エンジンの起動 (java -jar journeyEngine.jar)

停止処理（データを失わないための手順)

a. ウェブサーバーを停止する
b. エンジンの停止(grep and kill Pid)またはDirectorを使用する。
c. Kafkaを停止する
d. Zookeeperを止める

Chapter 4. Journeyユーザーの役割と権限

使い始める前にUnica Journey、ユーザーに役割と権限を割り当てる必要があります。

-
-
-

Note: 構成の変更には、再起動が必要ですUnica Journey.セキュリティ構成に関する詳細については、次を参照してください。*Unica Platform*管理者ガイド。

Chapter 5. Journey相互作用のロギング

の相互作用ログJourneyスケジュールされたジョブとして実行されます。スケジューリング パラメータは、アプリケーションの`application.properties`ファイルで設定されますJourneyエンジ.ン。以下に設定例を示します。

```
engine.logging.cron=0 15 3 * * ?
```

スケジュールされたジョブは、JourneyEngineの`application.properties`ファイルで再度定義される代替スキーマにデータをエキスポートします.

```
journey.report.datasource.url = journey.report.datasource.username = journey.report.datasource.password
 = journey.report.datasource.driver-class-name=
```

インタラクション ロギングは、Journey彼らがそれぞれを移動するときのアプリケーションJourney、公開済みまたは完了済み。公開されているが一時停止されているジャーニーも、対話ログの対象と見なされます。

すべてのタッチポイント、電子メール、SMS、または CRM は、それぞれのチャネルを介して構成された統合を使用してオーディエンス データが送信されるため、Interaction Logging の対象と見なされます。すべての連絡先から受信した応答も取得されます。

Log4j2

両方JourneyWebとJourneyエンジンは標準を使用ロギング用。両方の`log4j2.xml`ファイルJourneyWebとJourneyエンジンは、インストール場所の`conf`フォルダー内に配置されます。

両方JourneyWebとJourneyエンジンは、通常のアプリケーション ログとパフォーマンス ログを生成します。JourneyWeb の場合、ログの既定の場所は`ログ`フォルダー内です。Journeyログのデフォルトの場所は`performancelogs`フォルダー内です。両方のためのJourneyWebとJourneyエンジン、前述のフォルダーはインストール場所内に配置されます。

Chapter 6. JourneyGDPR

アクセスJourney GDPR

GDPRツールは、Journey のアプリケーションフォルダからアクセスできます。場所は以下の通りです。

```
<Journey_Home>\Journey\tools\GDPR\
```

GDPR > MariaDB MS SQL サーバー、**OneDB データベース**のほか、**Oracle**もサポートしています。

Journey GDPRを実行します

Journey GDPR を実行するには、次の手順を実行します。

1. gdpr.propertiesファイル内の以下のプロパティを変更します。

プロパティー名	値の例	メモ
Journey.audience.DBType	ORACLE	現在、Journey は Oracle のみサポートしています。
Journey.audience.Db.スキーマ名	Journeyユーザー	Journey データベースで使用されるスキーマ名。
Journey.audience.Field	電子メール/携帯電話番号	フィールド名はCSVファイルの入力。
Journey.audience.Csv	<GDPR_HOME>/sample/JourneyAudiences.csv	<GDPR_HOME>をカレントディレクトリのパスに置き換えます。 これは、Journey からオプトアウトする必要があるレコードを含む入力csvファイルです。
Journey.audience.Output	<GDPR_HOME>/JourneyAudiences.sql	JourneyAudiences.sql は、Journey アプリケーションからすべてのレコードを削除するために使用されたすべての SQL クエリを含む出力ファイル名です。<GDPR_HOME>をカレントディレクトリのパスに置き換えます。
Journey.audience.Output.FileSizeLimit	10	数値はMBs単位です。ファイルサイズが入力した値を超えると、以下のサフィックスを持つ複数のファイルが生成されます。JourneyAudiences _0、JourneyAudiences _1、…

プロパティー名	値の例	メモ
		といった具合に、複数のファイルが生成されます。

2. **Note:** エラーが表示された場合は、このログファイルを使用して追跡できます。
3. ファイルを実行するには、次のいずれかの手順を実行します。
    a. Windowsの場合、`gdpr_purge.bat`ファイルを探して実行します。例えば、`gdpr_purge.bat`が`D:\workspace\HCL_GDPR\dist\journey\`にある場合、`gdpr_purge.bat`ファイルをを実行します。
    b. UNIXベースのシステムの場合、`gdpr_purge.sh`ファイルを探して実行します。例えば、`gdpr_purge.sh`が`\workspace\HCL_GDPR\dist\journey\`にある場合、./gdpr_purge.sh コマンドを実行します。
4. `gdpr_purge.bat`（Windows用）または`gdpr_purge.sh`（Linux用）を実行すると、上記手順で指定した`<GDPR_HOME>`に *"JourneyAudiences 0","JourneyAudiences _1","JourneyAudiences _2 "*などの出力ファイルが作成されます。生成されるファイル数は、指定されたファイルサイズに依存します。
5. *"JourneyAudiences_x "*ファイルには、JourneyAudiences.csvに記載されたレコードの削除クエリが含まれます。
6. これらのクエリーは、「Journey」データベースで必要に応じて手動で実行し、「Journeyaudiences」テーブルからレコードを削除させる必要があります。

GDPRユーティリティは、以下のテーブルからJourneyAudiences、AudienceResponse、AudienceResponseMetaData、AudienceResponseInteraction、JourneyAudienceMileのレコードを削除します。ただし、集計結果が格納されている各テーブルからのデータ削除は行いません。例えば、journeyFlow、journeyAudienceFlow、JourneyGoalContactTransactionなどのテーブルがあります。そのため、UI上でカウントの不一致が発生します。

GDPRツールでは、Publish Kafkaトピックからも、ファイルシステム上のファイルからも、顧客データを削除することはできません。このデータは、ユーザーが必要に応じて手動で削除する必要があります。

GDPRツールでは、JDBCコネクタからエクスポートされた顧客データを削除することはできません。

Chapter 7. SSL を使用した Kafka 認証

組織の Kafka インスタンスを使用している場合は、その Kafka インスタンス用に構成された証明書を使用できます。SSL キーと証明書を生成し、Journey アプリケーションのプロパティで構成するためのクライアント証明書を取得する必要はありません。

証明書がない場合は、自己署名認証局 (CA) を生成できます。これは、単なる公開鍵と秘密鍵のペアと証明書です。

各なKafka クライアントとブローカーの信頼ストアに, 同じ CA 証明書を追加する必要があります。

各なKafka ブローカーに SSL キーと証明書を生成します

Kafka サーバーの自己署名証明書を生成するには、次の手順を実行します。

前提条件

- 証明書と信頼ストアを生成するには、Java keytool と OpenSSL が必要です。
- 必要に応じて、OpenSSL の代わりに任意の SSL 証明書生成ユーティリティを使用できます。

1. SSL をデプロイするには、クラスター内の各マシンのキーと証明書を生成します。最初にキーを一時キーストアに生成して、後で CA でエクスポートして署名できるようにします。

   ```
   keytool -keystore kafka.server.keystore.jks -alias localhost -validity 365 -genkey
   ```

   - キーストア: 証明書を格納するキーストア ファイル。キーストア ファイルには、証明書の秘密鍵が含まれているため、安全に保管する必要があります。
   - 有効性: 証明書の有効期間 (日数)。
2. 独自の CA (認証局) を作成する

   ```
   openssl req -new -x509 -keyout ca-key -out ca-cert -days 365
   ```

   生成された CA は、単に公開鍵と秘密鍵のペアと証明書であり、他の証明書に署名することを目的としています。
3. 生成された CA をクライアントの信頼ストアに追加して、クライアントがこの CA を信頼できるようにします。
   - `keytool -keystore kafka.server.truststore.jks -alias CARoot -import -file ca-cert`
   - `keytool -keystore kafka.client.truststore.jks -alias CARoot -import -file ca-cert`
4. 生成された CA を使用して、キーストア内のすべての証明書に署名します。
   a. キーストアから証明書をエクスポートします。

      ```
      keytool -keystore kafka.server.keystore.jks -alias localhost -certreq -file cert-file
      ```
5. CA で署名します。

   ```
   openssl x509 -req -CA ca-cert -CAkey ca-key -in cert-file -out cert-signed -days 365 -CAcreateserial
   -passin pass:<password>
   ```
6. CA の証明書と署名付き証明書の両方をキーストアにインポートします。

   ```
   keytool -keystore kafka.server.keystore.jks -alias CARoot -import -file ca-cert
   ```

   ```
   keytool -keystore kafka.server.keystore.jks -alias localhost -import -file cert-signed
   ```

7. クライアント キーストアを作成し、CA の証明書と署名付き証明書の両方をクライアント キーストアにインポートします。これらのクライアント証明書は、アプリケーション プロパティで使用されます。

```
keytool -keystore kafka.client.keystore.jks -alias localhost -validity 365 -genkey

keytool -keystore kafka.client.keystore.jks -alias localhost -certreq -file cert-file

openssl x509 -req -CA ca-cert -CAkey ca-key -in cert-file -out cert-signed -days 365 -CAcreateserial
-passin pass:<password>

keytool -keystore kafka.client.keystore.jks -alias CARoot -import -file ca-cert

keytool -keystore kafka.client.keystore.jks -alias localhost -import -file cert-signed
```

Kafka サーバー, JourneyおよびLinkコンポーネントをSSLで構成する

Kafka サーバーとクライアント証明書に使用されるサーバー証明書は、Kafka サーバーに接続するすべてのアプリケーションで使用する必要があります。Journeyウェブ、Journeyエンジン、Unica Link – この Kafka サーバーに接続するために必要な Kafka-link またはその他のツール。

Kafka サーバーを構成するには、Journeyコンポーネント、およびLinkコンポーネントを SSL 認証で使用するには、次のセクションで説明する手順を実行します。

Kafka サーバーをSSL 認証で 構成する

次のサーバー証明書は、Kafka サーバーのみに使用する必要があります。これらの証明書を必要なマシンで共有し、パスワードをメモします。

- `kafka.server.keystore.jks`
- `Kafka.server.truststore.jks`

Kafka サーバーの config ディレクトリにある次の server.properties を更新します。

```
listeners=SSL://<KAFKA_HOST>:<KAFKA_PORT> ssl.keystore.location=/PATH/kafka.server.keystore.jks
 ssl.keystore.password= password ssl.key.password= password
 ssl.truststore.location= /PATH/kafka.server.truststore.jks ssl.truststore.password= password
 ssl.endpoint.identification.algorithm= ssl.client.auth=required security.inter.broker.protocol=SSL
```

JourneyエンジンをKafka SSLで設定する

次のクライアント証明書を使用し、必要なマシンでこれらの証明書を共有し、パスワードをメモします。

- `Kafka.client.keystore.jks`
- `kafka.client.truststore.jks`

1. アップデートJourney<JOURNEY_HOME>/Engine/conf/ディレクトリのエンジンlog4j2.xmlファイル。log4j2.xmlの次の行のコメントを外します。

```
<Property name="security.protocol" >${sys:security.protocol}</ Property> <Property
 name="ssl.truststore.location"> ${sys:ssl.truststore.location}</Property>
 <Property name="ssl.truststore.password"> ${sys:ssl.truststore.password}</Property>
```

```
<Property name="ssl.keystore.location">${sys:ssl.keystore.location}</ Property>
<Property name="ssl.keystore.password">${sys:ssl.keystore.password}</
Property> <Property name="ssl.key.password">${sys:ssl.key.password}</Property>
<Property name="ssl.endpoint.identification.algorithm">
${sys:ssl.endpoint.identification.algorithm}</Property>
```

2. <JOURNEY_HOME>/Engine/ディレクトリーからjourney_engine_master.configを更新します。
3. 次のプロパティ値を更新します。

```
kafka.security.enabled=Y kafka.security.protocols.enabled=SSL security.protocol=SSL
 ssl.truststore.location= /PATH/kafka.client.truststore.jks ssl.truststore.password=<ENCYPTED
 PASSWORD WITH JOURNEY ENCRYPTION TOOL> ssl.keystore.location= /PATH/kafka.client.keystore.jks
 ssl.keystore.password=<ENCYPTED PASSWORD WITH JOURNEY ENCRYPTION TOOL>
 ssl.key.password=<ENCYPTED PASSWORD WITH JOURNEY ENCRYPTION TOOL>
 ssl.endpoint.identification.algorithm=
```

Kafka SSLでJourneywebを構成する

1. <JOURNEY_HOME>/Web/properties/ディレクトリから Web application.propertiesファイルをJourney更新します。
2. 次のプロパティ値を更新します。

```
kafka.security.enabled=Y kafka.security.protocols.enabled=SSL
 ssl.truststore.location= /PATH/kafka.client.truststore.jks ssl.truststore.password= <ENCYPTED
 PASSWORD WITH JOURNEY ENCRYPTION TOOL> ssl.keystore.location= /PATH/kafka.client.keystore.jks
 ssl.keystore.password= <ENCYPTED PASSWORD WITH JOURNEY ENCRYPTION TOOL> ssl.key.password=
 <ENCYPTED PASSWORD WITH JOURNEY ENCRYPTION TOOL> ssl.endpoint.identification.algorithm=
```

Unica LinkコンポーネントとSSL を構成する

の次のプロパティ値を更新します。Unica Linkインストール - kafkalink.propertiesファイル。

```
security.ssl=true security.protocol=SSL ssl.truststore.location= /PATH/kafka.client.truststore.jks
 ssl.truststore.password=password security.authentication=username
 ssl.keystore.location= /PATH/kafka.client.keystore.jks ssl.keystore.password=password
 ssl.key.password=passwordssl.endpoint.identification.algorithm=
```

Kafka サーバー, JourneyおよびLinkコンポーネントをSSLで構成する

Kafka サーバーを構成するには、Journeyコンポーネント、およびLinkコンポーネントを SASL 認証で使用するには、次のセクションで説明する手順を実行します。

Kafka サーバーをSASL 認証で 構成する

1. kafka-run-class.bat/shで JVM パラメータを指定します。

   JAVA_OPTS=%JAVA_OPTS%を設定する

   -Djava.security.auth.login.config=/PATH/kafka_server_jaas.conf

   コマンドの設定=%JAVA% %JAVA_OPTS% %KAFKA_HEAP_OPTS%
   %KAFKA_JVM_PERFORMANCE_OPTS% %KAFKA_JMX_OPTS% %KAFKA_LOG4J_OPTS% -cp

```
"%CLASSPATH%" %KAFKA_OPTS% %*
```

サンプルの jaas.configファイル:

```
KafkaServer { org.apache.kafka.common.security.plain.PlainLoginModule required username="admin"
 password="admin-secret" user_admin="admin-secret" user_alice="alice-secret"; };
```

```
KafkaClient { org.apache.kafka.common.security.plain.PlainLoginModule required username="alice"
 password="alice-secret"; };
```

2. KAFKA_SERVER/config/server.propertiesから次の Kafka サーバー プロパティ ファイルを更新します。

```
listeners=SASL_PLAINTEXT:// <KAFKA_HOST>:<KAFKA_PORT>
 security.inter.broker.protocol=SASL_PLAINTEXT sasl.mechanism.inter.broker.protocol=PLAIN
 sasl.enabled.mechanisms=PLAIN
```

JourneyエンジンとKafka SASLを設定する

1. アップデートJourney<JOURNEY_HOME>/Engine/conf/ディレクトリのエンジンlog4j2.xmlファイル。log4j2.xmlの次の行のコメントを外します。

```
<!-- Kafka SASL configuration --> <Property
 name="security.protocol">${sys:security.protocol}</Property> <Property
 name="sasl.mechanism">${sys:sasl.mechanism}</Property>
```

2. <JOURNEY_HOME>/Engine/ディレクトリーからjourney_engine_master.configを更新します。次のプロパティ値を更新します。

```
kafka.security.enabled=Y kafka.security.protocols.enabled=SASL_PLAINTEXT
 security.protocol=SASL_PLAINTEXT sasl.mechanism=PLAIN
 java.security.auth.login.config=./kafka_client_jaas.conf
```

JourneyWebとKafka SASLで構成する

<JOURNEY_HOME>/Web/properties/ディレクトリから Web application.propertiesファイルをJourney更新します。

```
kafka.security.enabled=はい kafka.security.protocols.enabled=SASL_PLAINTEXT
 java.security.auth.login.config=/PATH/kafka_client_jaas.conf
```

Kafka SASLでUnica Linkコンポーネントを設定する

Unica Linkインストール - kafkalink.propertiesファイルは次のプロパティ値を更新します。

```
security.sasl =true security.protocol=SASL_PLAINTEXT security.sasl.auth.login.config
 =/PATH/kafka_client_jaas.conf sasl.mechanism=PLAIN
```

Kafka サーバーの構成とJourneySASL_SSL 構成のコンポーネント

Kafka サーバーなどを構成するにはJourneyコンポーネントを SASL 認証で使用するには、次のセクションで説明する手順を完了してください。

Note: Unica LinkSASL_SSL 認証メカニズムを使用した Kafka-link への接続をサポートしていません。SASL または SSL 認証メカニズムを使用する必要があります。

Kafka SASL_SSL でKafka サーバーを設定する

Kafka サーバー構成ディレクトリーで以下の server.properties を更新します。

```
listeners=SASL_SSL:// <KAFKA_HOST>:<KAFKA_PORT> security.inter.broker.protocol=SASL_PLAINTEXT
 sasl.mechanism.inter.broker.protocol=PLAIN sasl.enabled.mechanisms=PLAIN
 ssl.keystore.location=/PATH/kafka.server.keystore.jks ssl.keystore.password=password ssl.key.password=
 password ssl.truststore.location=/PATH/kafka.server.truststore.jks ssl.truststore.password= password
 ssl.endpoint.identification.algorithm= ssl.client.auth=required security.inter.broker.protocol=SSL
```

エンジンとKafka SASLを設定するJourney

1. アップデートJourney<JOURNEY_HOME>/Engine/conf/ディレクトリのエンジンlog4j2.xmlファイル。

   log4j2.xmlの次の行のコメントを外します。

   ```
   <Property name="sasl.mechanism">${sys:sasl.mechanism}</Property> <Property
    name="security.protocol" >${sys:security.protocol}</Property> <Property
    name="ssl.truststore.location" >${sys:ssl.truststore.location}</Property>
    <Property name="ssl.truststore.password">${sys:ssl.truststore.password}</Property>
    <Property name="ssl.keystore.location">${sys:ssl.keystore.location}</Property>
    <Property name="ssl.keystore.password">${sys:ssl.keystore.password}</Property>
    <Property name="ssl.key.password">${sys:ssl.key.password}</Property> <Property
    name="ssl.endpoint.identification.algorithm">${sys:ssl.endpoint.identification.algorithm}</Prope
   rty>
   ```

2. <JOURNEY_HOME>/Engine/ディレクトリから以下のtour_engine_master.configを更新します。

   次のプロパティ値を更新します。

   ```
   kafka.security.enabled=Y kafka.security.protocols.enabled=SASL_SSL
    ssl.truststore.location=/PATH/kafka.client.truststore.jks
    ssl.truststore.password=<ENCYPTED PASSWORD WITH JOURNEY ENCRYPTION TOOL>
    ssl.keystore.location=/PATH/kafka.client.keystore.jks ssl.keystore.password=<ENCYPTED
    PASSWORD WITH JOURNEY ENCRYPTION TOOL> ssl.key.password=<ENCYPTED PASSWORD
    WITH JOURNEY ENCRYPTION TOOL> ssl.endpoint.identification.algorithm=
    java.security.auth.login.config=/PATH/kafka_client_jaas.conf
   ```

JourneyWeb と Kafka SASL_SSLを設定する

以下を更新しますJourney<JOURNEY_HOME>/Web/properties/ディレクトリの Web application.propertiesファイル。

```
kafka.security.enabled=Y kafka.security.protocols.enabled=SASL_SSL
 ssl.truststore.location=/PATH/kafka.client.truststore.jks ssl.truststore.password=<ENCYPTED
 PASSWORD WITH JOURNEY ENCRYPTION TOOL> ssl.keystore.location=/PATH/kafka.client.keystore.jks
 ssl.keystore.password=<ENCYPTED PASSWORD WITH JOURNEY ENCRYPTION TOOL> ssl.key.password=<ENCYPTED
 PASSWORD WITH JOURNEY ENCRYPTION TOOL> ssl.endpoint.identification.algorithm=
 java.security.auth.login.config=/PATH/kafka_client_jaas.conf
```

Chapter 8. SSL 用に Web アプリケーション サーバー Tomcat を構成する

Unicaアプリケーションを配置するすべてのアプリケーションサーバーで、採用することを決定した証明書を使用するようにWebアプリケーションサーバーを設定する。

これらの手順の実行について詳しくは、アプリケーション・サーバーの資料を参照してください。

Cookie のセキュリティーの確認

いくつかの cookie は、クライアント・ブラウザーで適切に保護されない場合があります。cookies を保護しないと、アプリケーションは中間者攻撃およびセッション・ハイジャック攻撃に対してぜい弱なままになります。この問題を修正するには、以下の予防手段を取ります。

- 常に SSL の使用を強制して、ワイヤー上で代行受信されている cookie のリスクを減らします。
- Web アプリケーション・サーバーで、`secure` フラグおよび `httponly` フラグをすべての cookie に設定します。
    - `secure` フラグは、ブラウザーが HTTPS 接続のみを使用して cookie を送信するよう指示します。このフラグを設定する場合、相互に通信するすべてのアプリケーションで SSL を有効にする必要があります。
    - `httponly` フラグは、cookie がクライアント・サイドのスクリプトを介してアクセスされないようにします。

Tomcat での SSL のフラグの設定

Tomcat で `secure` フラグと `httponly` フラグを設定するには 、Tomcat の `.xml` サーバーで以下の変更を行います。

このタスクについて

```
<Connector port="7003" protocol="org.apache.coyote.http11.Http11NioProtocol"
maxThreads="150" SSLEnabled="true" scheme="https" acceptCount="100" clientAuth="false"
disableUploadTimeout="true" enableLookups="false" secure="true" sslProtocol="TLS"
 keystoreFile="/opt/v12.1/v12.1.0.1.1/Campaign/SSL_NEW/PlatformClientIdentity.jks"
 keystorePass="password" >     </Connector>
```

Unica JourneyをSSLで 構成する

構成するにはUnica JourneySSL を使用するには、いくつかの構成プロパティを設定する必要があります。インストールに適した、このセクションの手順を使用します。Unica Journey SSL を使用して保護する必要がある通信。

About this task

アクセスするとUnicaまた、次の手順で説明するようにアプリケーションのナビゲーション プロパティを設定する場合は、URL で`https`とセキュア ポート番号を使用する必要があります。Tomcat のデフォルトの SSL ポートは`8443`です。

この手順に従って Journey with SSL を設定します

1. ログインするUnicaをクリックし、**[設定] > [構成]** をクリックします。
2. Affinium |の値を設定します。旅 | ナビゲーションプロパティUnica JourneyURL。

   例: `https:// host.domain:SSL_port /unica`

   各部の意味は以下のとおりです。

   - *host* は、マシンの名前または IP アドレスです。Unica Journeyインストールされています
   - *domain* は、あなたの会社のドメインです。Unica製品がインストールされています
   - *SSL_Port* は、アプリケーション サーバーの SSL ポートです。Unica Journey展開されます

   URL のhttpsに注意してください。

Chapter 9. 設定

設定メニューを使用して、Journey電子メール コネクタ、SMS コネクタ、CRM 接続、REST 統合などの統合。

デフォルトの電子メール接続の設定する

複数のコネクタがある場合Unica Link電子メールを送信する場合、[設定] メニューでデフォルトの電子メール接続を**設定**できます。

About this task

デフォルトの電子メール接続を設定するには、次の手順を実行します。

1.  **> Link > 電子メール**
   Result を選択する。
   [電子メール]ページが表示されます。
2. **[使用可能な接続]**リストから、接続を選択します。
   利用可能な接続には、Mandril、Mailchimp などが含まれます。
3. **[保存]**をクリックします。
   既存の接続を選択解除して、**[保存]** をクリックすることもできます。これにより、デフォルトの接続が設定されていないことが保証されます。

デフォルトの SMS 接続の設定します

複数のコネクタがある場合Unica LinkSMS を送信する場合は、[設定] メニューでデフォルトの SMS 接続を**設定**できます。

About this task

デフォルトの SMS 接続を設定するには、次の手順を実行します。

1.  **> Link > SMS**
   Result 選択します 。
   SMSページが表示されます。
2. **[使用可能な接続]**リストから、接続を選択します。

   Note:

   電話番号の形式は、配信チャネルの仕様に従って記載する必要があります。Journey電話番号を同じ形式で配信チャネルに送信します。たとえば、参照でサポートされている Twilio 接続の電話番号形式Journey以下のとおりであります:

   - <プラス記号><国コード><*10* 桁の電話番号> - `+15403241212` .
   - <プラス記号> <国コード <(市外局番)> <*3*桁の数字><*4*桁の数字> - `+1 (540) 324 1212` .

- <プラス記号>-<国コード>-<市外局番>-<*3* 桁の数字>-<*4* 桁の数字> - `+1-540-324-1212` .
- <プラス記号> <国コード>-<市外局番>-<*3* 桁の数字>-<*4* 桁の数字> - `+1 540-324-1212` .

入力した電話番号の形式に関係なく、Unica Journey次の形式で番号を保存します: <プラス記号><国コード><*10* 桁の電話番号> 。たとえば、電話番号を`+1 540-324-1212`として指定すると、Unica Journey電話番号を`+15403241212`として保存します。

デフォルトの SMS 接続として Twilio を選択すると、次の形式の電話番号のみが受け入れられます: <プラス記号><国コード><*10* 桁の電話番号> 。たとえば、 `+15403241212`です。

3. **[保存]**をクリックします。

デフォルトの CRM 接続の設定

複数の CRM 接続がある場合は、[設定] メニューで既定の CRM 接続を**設定**できます。

About this task

デフォルトの CRM 接続を設定するには、次の手順を実行します。

1. **> Link > CRM**
   Result を選択する 。
   CRMページが表示されます。
2. **[使用可能な接続]**リストから、接続を選択します。
3. **[保存]**をクリックします。

ADTECHのデフォルト接続を設定する

ADTECH接続が複数ある場合は、**設定**メニューでADTECHのデフォルト接続を設定できます。

About this task

ADTECHのデフォルト接続を設定するには、次の手順を実行します。

1. **> Link > ADTECH**
   選択
   Result
   ADTECHページが表示されます
2. **利用可能な接続の**一覧から、接続を選択します。
3. **[保存]**をクリックします。

デフォルトのデータベース接続を設定する

複数のデータベース接続がある場合、**設定**メニューでデフォルトのデータベース接続を設定することができます。

About this task

データベース接続をデフォルトで設定するには、次の手順を実行します。

1.  **> リンク > データベース**
   選択
   Result
   データベースのページが表示される
2. **利用可能な接続の**一覧から、接続を選択します。
3. **[保存]**をクリックします。

接続を管理する

Unica Link接続からこのメニューを管理できます。

About this task

Unica Link Mailchimp、Mandrill、Salesforce、Twilio などのコネクタを接続を作成できます。すべての既存は[**Existing Connections** (*n*)] にパネルでの接続を表示できます。nは接続の数です。

1. Mailchimp 接続を作成するには、次の手順を実行します。

   a.  **> Link > 接続の管理 > 新規作成**
   Result を選択する.
   [新しい接続の作成]ページが表示されます。

   b. 次のフィールドに値を入力します。
      - **名前**-必須
      - **説明**-オプション

   c. **[次]**をクリックします。

   d. **[接続の選択]** パネルから、 [**Mailchimp**] を選択します。

   e. **[接続プロパティ]** パネルで、次の必須フィールドに値を指定します。

      Note: 入力するフィールドと値については、を参照してください。 *Unica LinkMailchimp* コネクタ ユーザー ガイド.

      - **ベース URL**
      - **ユーザー ID**
      - **API キー**
      - **アクティビティのフェッチ頻度**
      - **アクティビティ フェッチ ユニット**

   f. **[テスト]**をクリックして、接続をテストします。指定された値が正しい場合は、成功メッセージが表示されます。指定した値が正しくない場合は、エラー メッセージが表示されます。

   g. 接続を保存するには、**[保存]** をクリックします。

Result

新しい接続が正常に保存され、[**既存の接続**] パネルに表示されます。

2. Mandril 接続を作成するには、次の手順を実行します。

a.  **> Link > 接続の管理 > 新規作成**

Result を選択する.

[新しい接続の作成]ページが表示されます。

b. 次のフィールドに値を入力します。

- **名前**-必須
- **説明**-オプション

c. **[次]**をクリックします。

d. [**接続の選択**] パネルから、 **Mandrill**を選択します。

e. [**接続プロパティ**] パネルで、次の必須フィールドに値を指定します。

Note: 入力するフィールドと値については、を参照してください。*Unica LinkMandrill*ユーザー ガイド.

- **API キー**
- **アクティビティのフェッチ頻度**
- **アクティビティ フェッチ ユニット**

f. **[テスト]**をクリックして、接続をテストします。指定された値が正しい場合は、成功メッセージが表示されます。指定した値が正しくない場合は、エラー メッセージが表示されます。

g. 接続を保存するには、[**保存**] をクリックします。

Result

新しい接続が正常に保存され、[**既存の接続**] パネルに表示されます。

3. Salesforce 接続を作成するには、次の手順を実行します。

a.  **> Link > 接続の管理 > 新規作成**

Result を選択する.

[新しい接続の作成]ページが表示されます。

b. 次のフィールドに値を入力します。

- **名前**-必須
- **説明**-オプション

c. **[次]**をクリックします。

d. [**Choose Connection**] パネルから、 [**Salesforce**] を選択します。

e. **[接続プロパティ]** パネルで、次の必須フィールドに値を指定します。

Note: 入力するフィールドと値については、を参照してください。 *Unica LinkSalesforce* ユーザー ガイド.

- **インスタンス URL**
- **アクセストークン**
- **バージョン**

f. **[テスト]**をクリックして、接続をテストします。指定された値が正しい場合は、成功メッセージが表示されます。指定した値が正しくない場合は、エラー メッセージが表示されます。

g. 接続を保存するには、**[保存]** をクリックします。

Result

新しい接続が正常に保存され、**[既存の接続]** パネルに表示されます。

4. Twilio 接続を作成するには、次の手順を実行します。

a. **> Link > 接続の管理 > 新規作成**

Result を選択する.

[新しい接続の作成]ページが表示されます。

b. 次のフィールドに値を入力します。

- **名前**-必須
- **説明**-オプション

c. **[次]**をクリックします。

d. **[接続の選択]** パネルから、[**Twilio**] を選択します。

e. **[接続プロパティ]** パネルで、次の必須フィールドに値を指定します。

Note: 入力するフィールドと値については、を参照してください。 *Unica LinkTwilio* ユーザー ガイド.

- **ベース URL**
- **アカウント SID**
- **認証トークン**
- **番号から**
- **再試行間隔**
- **再試行回数**

f. **[テスト]**をクリックして、接続をテストします。指定された値が正しい場合は、成功メッセージが表示されます。指定した値が正しくない場合は、エラー メッセージが表示されます。

g. 接続を保存するには、**[保存]** をクリックします。

Result

新しい接続が正常に保存され、**[既存の接続]** パネルに表示されます。

REST の統合

REST キーは、アプリケーションへのサードパーティ ログインに使用されます。キーと値のペアを生成し、キーとバリューのペアを使用してJourneyサードパーティアプリケーションにログインすることができます。

新しい REST 統合の作成します

新しい REST 統合キー ペアを作成するには、次の手順を実行します。

1.  **> REST**

   Result を選択します。

   RESTページが表示されます。
2. **+ REST 統合**をクリックします。

   Result

   新しい REST 統合ページが表示されます。
3. 次のフィールドに値を入力します。
   - **アプリ名**- 必須。
   - **説明**- オプション。
4. **[キーの生成] を**クリックします。

   Result

   システムは**ClientID**と**ClientSecret**を生成します。
5. トグル バーを使用して、**[ステータス]**を [`Active`]または [`Inactive`] に変更します。デフォルトでは、**ステータス**は`Active`です。
6. REST 統合を保存するには、**[保存]** をクリックします。

   オーディエンス データを送信するにはJourney、REST エンドポイントの構成に使用される REST エントリ ソースに記載されている詳細に従います。ステップ (4) を実行したときに受け取った**ClientID**と**ClientSecret**を使用して、エントリ ソースで REST エンドポイントを構成します。

REST 統合リストの表示します

Unica Journey作成された REST 統合のリストを維持します。

About this task

REST 統合のリストを表示するには、次の手順を実行します。

1.  **> REST**

   Result を選択します。

   RESTページが表示されます。

2. 次の操作のいずれかを実行します。
    a. [名前] フィールドで REST 統合のリストを昇順または降順で表示するには、**[名前]** をクリックします。
    b. [説明] フィールドで REST 統合のリストを昇順または降順で表示するには、**[説明]** をクリックします。

既存の REST 統合を変更

既存の REST 統合を説明とステータスのみを変更できます。

About this task

既存の REST 統合を変更するには、次の手順を実行します。

1.  **> REST**
   Result を選択します。
   RESTページが表示されます。
2. 残りの統合を変更するには、次のいずれかを実行できます。
   Choose from:
   - リストから必要な REST 統合を選択します
   - 選択する **>** 

   Result
   REST 統合の更新ページが表示されます。
3. 次のフィールドのみを更新できます。
   Choose from:
   - **説明**
   - **状況**
4. 変更を保存するには、**[保存]** をクリックします。

REST 統合を削除

使用されなくなった、または不要になった非アクティブな REST 統合のみを削除できます。

Before you begin

REST 統合エントリのステータスを変更するには、既存の REST 統合を変更 on page 27を参照してください。 .

About this task

既存の非アクティブな REST 統合を削除するには、次の手順を実行します。

1.  **> REST**
   Result を選択します。
   RESTページが表示されます。

2. 次のいずれかの手順を実行します。
   Choose from:

   - REST 統合を削除するには、 >  リスト内のREST統合を成功させます。
   - 複数の REST 統合を削除するには、リストで削除する REST 統合の前にあるチェックボックスを選択し、[**削除**] をクリックします。
3. 確認ボックスが表示されます。削除を続行するには、[**OK**] をクリックします。

Journey Proxy 統合

Proxyサーバーは、旅ウェブとエンジンのプロジェクトに統合され、これにより、ユーザーはセキュリティを追加し、アプリケーションサーバーをProxyサーバーの背後に保つことができるようになりました。Proxyサーバーは、Deliver、Link、Platformの各サーバーとやり取りを行います。

Journey Web - Deliver、Link、Platformサーバーと通信し、構成の詳細を取得したり、Journeyにメール/SMS/AdTech Pointを統合したりします。

Journey Engine - Proxyを使用してDeliver/Link Serverとの通信を行い、Eメール/SMS/Adtechの詳細をエンドサーバーに送信します。

Journey Webでサポートされ ているProxy

1. SOCKS
2. HTTP
3. HTTPS

Journeyエンジンでサポートされ ているProxy

1. HTTP

Note: EngineがDeliverと通信するために使用するSOAP（Apache Axis2）では、SOCKSおよびHTTPS Proxyはサポートされていません。

Engineのapplication.propertiesファイルでEngineに設定するプロパティです。

- journey.proxy.type=NONE
- spring.proxy.host=[IP]
- spring.proxy.port=[PORT]
- spring.proxy.username=[username]
- spring.proxy.password=[password]

Web アプリケーション.properties ファイルで Web 用に設定されるプロパティ

- journey.proxy.type=NONE
- spring.proxy.host=[IP]

- spring.proxy.port=[PORT]
- spring.proxy.username=[USERNAME]
- spring.proxy.password=[PASSWORD]
- server.use-forward-headers=true

Note: journey.proxy.typeプロパティのデフォルト値はNONEで、NONEに設定するとProxyは無効となる。

デベロッパーツール

デベロッパーツールの一覧を表示します。

APIドキュメント

ユーザーは、JourneyのRESTAPIのリストを見つけることができます。