

Unica JourneyV12.1.0管理者ガイド



Contents

Chapter 1. 紹介のUnica Journey	4
特徴Unica Journey.....	4
メリットのUnica Journey.....	4
Chapter 2. Unica Journey積算	6
紹介のUnica Deliver.....	9
Unica Deliver統合.....	10
Kafkaの統合.....	10
Chapter 3. Journeyの始まりと終わりのプロセス	13
Chapter 4. Journeyユーザーの役割と権限	14
アクセス許可の割り当てJourney役割.....	14
ユーザーへの JourneyAdmin ロールを割り当てします.....	15
JourneyUserロールをユーザーに割り当てる.....	16
Chapter 5. Journey相互作用のログイン	17
Chapter 6. JourneyGDPR	18
Chapter 7. SSL を使用した Kafka 認証	20
Kafka サーバー、Journey、SSL を使用したコンポーネントのリンクの設定.....	21
Kafka サーバーをSSL 認証で 構成する.....	21
JourneyエンジンをKafka SSLで設定する	21
Kafka SSLでJourneywebを構成する	22
Unica LinkコンポーネントとSSL を構成する.....	22
Kafka サーバー、JourneyおよびLinkコンポーネントをSSLで構成する	23
Kafka サーバーをSASL 認証で 構成する	23
JourneyエンジンとKafka SASLを設定する	23
JourneyWebとKafka SASLで構成する.....	24
Kafka SASL を使用した Unica Link コンポーネントの設定.....	24
SASL_SSL 設定を使用して、Kafka サーバーとジャーニーコンポーネントを設定します.....	24
Kafka SASL_SSL でKafka サーバーを設定する.....	24
エンジンとKafka SASLを設定するJourney	24
JourneyWeb と Kafka SASL_SSLを設定する.....	25
Chapter 8. SSL 用に Web アプリケーション サーバー Tomcat を構成する	26
Unica JourneyをSSLで 構成する.....	26

Chapter 9. 設定	27
デフォルトの電子メール接続の設定する.....	27
デフォルトの SMS 接続の設定します.....	27
デフォルトの CRM 接続の設定.....	28
ADTECHのデフォルト接続を設定する.....	28
デフォルトのデータベース接続を設定する.....	28
接続を管理する.....	28
REST の統合.....	31
新しい REST 統合の作成します.....	31
REST 統合リストの表示します.....	32
既存の REST 統合を変更.....	32
REST 統合を削除.....	33
Journey Proxy 統合.....	33
デベロッパーツール.....	34
APIドキュメント.....	34

Chapter 1. 紹介のUnica Journey

Unica Journey#####
#####

#####Unica Journey##

- 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

#####Journey#####
###Journey.

特徴Unica Journey

#####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

##Unica Journey#####

- **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#####

Unica Journey#####Unica Deliver #Unica Link #####Journey #####

Unica Journey####Unica Link

Unica Link#####SMS#CRM#ADTECH####JDBC#####Unica Link####

##SMS#CRM#ADTECH####JDBC#####

#####

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

Unica Link #####Journey #####SMS#CRM#ADTECH#JDBC #####

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####Unica Deliver

Unica Journey##Unica Deliver #####
 ##Journey.Unica Deliver #Unica Journey #####Unica Journey #####

Unica Journey#####Unica Campaign Unica Interact

Unica Journey Unica Campaign ###Unica Interact #####Unica Campaign ###Unica Interact #####
 Kafka #####Unica Journey #####Kafka#####
 #####

Unica Campaign Unica Journey # #####Unica Interact

Journey##Campaign#####

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	<i>Unica Campaign</i> 管理者ガイド、 <i>Unica Campaign</i> ユーザーガイドを参照してください。
Unica Campaign の統合と Unica Interact	<i>Unica Interact</i> 管理者ガイドを参照してください。

Unica JourneyUnica Discover####

Unica Journey#####Unica Discover#####Unica Journey#####REST
 #####4#####Journey#####
 #####2#####CART#Discover Entry#####Discover Entry#####2
 #####

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##Web #####
#####Deliver#####Unica Campaign#####
#####Unica .
```

```
Deliver#####Deliver#####Campaign#####
#Campaign#####
```

オーディエンスを選択します

```
Campaign#####
```

```
Deliver#####
#####
```

メッセージを作成します

```
Deliver#Document Composer #####
##Document Composer #####Deliver#####
##Deliver#####
```

メッセージを送信して応答を追跡する

```
##### Deliver#####  
##### Deliver##### Campaign#####
```

どうやって始めるのか

```
##### Campaign#####  
  
##### Unica#####  
##### Unica.##### Unica##  
### ##### Unica Deliver#####
```

Unica Deliver統合

```
##### Unica Deliver#Unica Journey##### Unica Platform .
```

1. Unica Platformで、**]]> 構成**に移動します。
[構成カテゴリ]ページが表示されます。
2. **Journey** 選択する。
「ジャーニー」 ページの**設定**が表示されます。
3. **[設定の編集]**を選択します。
(ジャーニー) ページが表示されます。
4. 次の手順を実行してください。
 - a. **Deliver_Configured**フィールドのために、**[Yes]** を選択します。
 - b. **[変更を保存]** をクリックします。
5. 展開されたジャーニー ノードで、**Deliver_構成**を選択します。
「Deliver_Configurations」 ページの**設定**が表示されます。
6. **[設定の編集]**を選択します。
(Deliver_Configurations)ページが表示されます。
7. 次の手順を実行してください。
 - a. 次のフィールドに値を入力します。
 - **Deliver_URL** : 構成された URLDeliver .
 - **Deliver_Partition** : 資格証明がアクセスするパーティション**Deliver_URL**が格納されます。
 - b. **[変更を保存]** をクリックします。

Kafkaの統合

```
Journey #####Unica Platform # Kafka #####
```

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)#####CommunicationMechanism #####

- NO_SASLPLAINTEXT_SSL
- SASL_PLAINTEXT
- SSL
- SASL_PLAINTEXT_SSL

#####

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 ###		はい		はい
truststore.location			はい	はい
truststore.password.data ###			はい	はい
keystore.location			はい	はい
keystore.password.data ###			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

#####

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

#####Unica Journey#####

- アクセス許可の割り当てJourney役割 (on page 14)
- ユーザーへの JourneyAdmin ロールを割り当てします (on page 15)
- JourneyUserロールをユーザーに割り当てる (on page 16)



**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。

## アクセス許可の割り当てJourney役割

#####

Journey## 2 #####

- Journey管理者
- Journeyユーザー

#####

1. からUnica Platformホームページで、**[設定] [> ユーザーの役割と権限]**を選択します。  
**[ユーザーの役割と権限]**ページが表示されます。
2. 左側のパネルで、展開します**Unica Journey > パーティション 1**。  
**partition1**ページが表示されます。
3. **[権限の割り当て]**を選択します。  
**(管理役割のプロパティ)**ページが表示されます。
4. **[権限を保存して編集]** をクリックします。  
**(パーティション 1 のアクセス許可)**ページが表示されます。
5. **アプリケーション**を展開します。
6. 次のフィールドに値を設定します。

| 操作         | Journey管理者のデフォルト設定 | Journeyユーザーデフォルト設定 |
|------------|--------------------|--------------------|
| データ定義の作成   | はい                 | いいえ                |
| データ定義の編集   | はい                 | いいえ                |
| データ定義の削除   | はい                 | いいえ                |
| エン트리ソースの作成 | はい                 | いいえ                |
| エン트리ソースの編集 | はい                 | いいえ                |
| エン트리ソースの削除 | はい                 | いいえ                |

| 操作          | Journey管理者のデフォルト設定 | Journeyユーザーデフォルト設定 |
|-------------|--------------------|--------------------|
| 作成Journey   | はい                 | はい                 |
| 編集Journey   | はい                 | はい                 |
| 削除Journey   | はい                 | いいえ                |
| 公開Journey   | はい                 | はい                 |
| 完了Journey   | はい                 | はい                 |
| 休止Journey   | はい                 | はい                 |
| 目標の追加/変更/削除 | はい                 | いいえ                |
| 目標ビュー       | はい                 | はい                 |
| 設定の追加・変更・削除 | はい                 | いいえ                |
| ビューの設定      | はい                 | はい                 |

**Note:**

- **Journey管理者**ロールの場合、アクセス許可を減らさず、デフォルトのアクセス許可を保持することをお勧めします。デフォルトでは、**Journey管理者**にはすべての権限があります。
- **Journeyユーザー**ロールは、適切と思われる権限を付与します。あなたは与えることができますが、すべての権限を**Journeyユーザー**に付与しますが、お勧めしません。

7. 権限を付与したら、[**変更を保存**] をクリックします。

## ユーザーへの JourneyAdmin ロールを割り当てします

#####JourneyAdmin#####

1. Marketing Platform のホームページから、[**設定**] [**> ユーザーの役割と権限**] を選択します。  
[**ユーザーの役割と権限**] ページが表示されます。
2. 左パネルで**Unica Journey**を展開します。
3. **partition1 > JourneyAdmin**を選択します。  
**JourneyAdmin**ページが表示されます。
4. [**ユーザー**]セクションで、ユーザーを選択します。たとえば、`asm_admin`です。  
**asm\_admin (asm\_admin)**ユーザーの詳細ページが表示されます。
5. [**ロールの編集**] を選択します。  
[**ロールの編集**] ページが表示されます。
6. [**使用可能なロール**] リストから、[**JourneyAdmin (Unica Journey)**] を選択し、[**>>**] ボタンをクリックして  
ロールを [**選択されたロール**] リストに移動します。
7. [**変更を保存**] をクリックします。

## JourneyUserロールをユーザーに割り当てる

JourneyUser#####

1. Marketing Platform のホームページから、**[設定] [> ユーザーの役割と権限]**を選択します。  
**[ユーザーの役割と権限]**ページが表示されます。
2. 左パネルで**Unica Journey**を展開します。
3. **partition1 > JourneyUser**を選択します。  
**JourneyUser**ページが表示されます。
4. **[ユーザー]**セクションで、ユーザーを選択します。たとえば、 `journey_example`です。  
**journey\_example (journey\_example)**ユーザーの詳細ページが表示されます。
5. **[ロールの編集]** を選択します。  
**[ロールの編集]**ページが表示されます。
6. **[使用可能なロール]** リストから**JourneyUser (Unica Journey)**を選択し、**[>>]** ボタンをクリックしてロールを  
**[選択されたロール]** リストに移動します。
7. **[変更を保存]** をクリックします。



# Chapter 5. Journey相互作用のロギング

```
#####Journey##### #####application.properties###
#####Journey###.#####
```

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

```
#####JourneyEngine#application.properties#####.
```

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

```
#####Journey#####Journey#####
#####
```

```
#####SMS#### CRM ##### Interaction
Logging #####
```

## Log4j2

```
##JourneyWeb#Journey#####log4j2.xml###JourneyWeb#Journey#####
#conf#####
```

```
##JourneyWeb#Journey##### ##### JourneyWeb ##### ログ###
#####Journey#####performancelogs#####JourneyWeb#Journey#####
#####
```

# 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 #####OneDB #####Oracle#####
```


## Journey GDPRを実行します

```
Journey GDPR #####
```

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

| プロパティ名                                             | 値の例                                                        | メモ                                                                                                                                                         |
|----------------------------------------------------|------------------------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <code>Journey.audience.DBType</code>               | ORACLE                                                     | 現在、Journey は Oracle のみサポートしています。                                                                                                                           |
| <code>Journey.audience.Db.スキーマ名</code>             | Journeyユーザー                                                | Journey データベースで使用されるスキーマ名。                                                                                                                                 |
| <code>Journey.audience.Field</code>                | 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。                                                                                                                                         |
| <code>Journey.audience.Csv</code>                  | <code>&lt;GDPR_HOME&gt;/sample/JourneyAudiences.csv</code> | <code>&lt;GDPR_HOME&gt;</code> をカレントディレクトリのパスに置き換えます。<br><br>#####Journey #####<br>#####csv#####                                                           |
| <code>Journey.audience.Output</code>               | <code>&lt;GDPR_HOME&gt;/JourneyAudiences.sql</code>        | <code>JourneyAudiences.sql</code> 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。 <code>&lt;GDPR_HOME&gt;</code> をカレントディレクトリのパスに置き換えます。 |
| <code>Journey.audience.Output.FileSizeLimit</code> | 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。 <code>JourneyAudiences_0</code> 、 <code>JourneyAudiences_1</code> 、...といった具合                     |

| プロパティ名 | 値の例 | メモ                |
|--------|-----|-------------------|
|        |     | 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#####
JourneyAudiences#AudienceResponse#AudienceResponseMetaData#AudienceResponseInteraction#JourneyAudienceMilestone#Jour
#####
##journeyFlow#journeyAudienceFlow#JourneyGoalContactTransaction#####UI#####
####

GDPR#####Publish Kafka#####
#####

GDPR#####JDBC#####
```

## Chapter 7. SSL を使用した Kafka 認証

```
Kafka ##### Kafka #####SSL #####Journey #####
#####
```

```
(CA)
```

```
##Kafka #####, ## CA #####
```

### 各なKafka ブローカーに SSL キーと証明書を生成します

```
Kafka #####
```

```
####
```

- 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
```

```
CA
```

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を署名します。

- 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、SSL を使用したコンポーネントのリンクの設定

```
Kafka #####Kafka ##### Journey
Web#Journey Engine#Unica Link - Kafka-link ##### Kafka #####
```

```
Kafka ##### SSL #####
```

### 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 ##### config ##### server.properties #####
```

```
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=<ENCRYPTED
PASSWORD WITH JOURNEY ENCRYPTION TOOL> ssl.keystore.location= /PATH/kafka.client.keystore.jks
ssl.keystore.password=<ENCRYPTED PASSWORD WITH JOURNEY ENCRYPTION TOOL> ssl.key.password=<ENCRYPTED
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= <ENCRYPTED
PASSWORD WITH JOURNEY ENCRYPTION TOOL> ssl.keystore.location= /PATH/kafka.client.keystore.jks
ssl.keystore.password= <ENCRYPTED PASSWORD WITH JOURNEY ENCRYPTION TOOL> ssl.key.password= <ENCRYPTED
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

#####Unica Link##### - kafkalink.properties#####

```
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 #####Journey#####Link##### SASL #####

### 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#####

-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/##### Web application.properties#####Journey#####
```

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

## Kafka SASL を使用した Unica Link コンポーネントの設定

```
Unica Link ##### - kafkalink.properties #####
```

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

## SASL\_SSL 設定を使用して、Kafka サーバーとジャーニーコンポーネントを設定します

Kafka サーバーやその他の Journeys コンポーネントを SASL 認証で設定するには、以下のセクションで説明する手順を実行します。



**Note:** Unica Link は、SASL\_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 ##### server.properties #####
```

```
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#####
```

```
<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}</Property>
```

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。

#####

```
kafka.security.enabled=Y kafka.security.protocols.enabled=SASL_SSL
ssl.truststore.location=/PATH/kafka.client.truststore.jks ssl.truststore.password=<ENCRYPTED
PASSWORD WITH JOURNEY ENCRYPTION TOOL> ssl.keystore.location=/PATH/kafka.client.keystore.jks
ssl.keystore.password=<ENCRYPTED PASSWORD WITH JOURNEY ENCRYPTION TOOL> ssl.key.password=<ENCRYPTED
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

#####Journey<JOURNEY\_HOME>/Web/properties/##### Web application.properties#####

```
kafka.security.enabled=Y kafka.security.protocols.enabled=SASL_SSL
ssl.truststore.location=/PATH/kafka.client.truststore.jks ssl.truststore.password=<ENCRYPTED
PASSWORD WITH JOURNEY ENCRYPTION TOOL> ssl.keystore.location=/PATH/kafka.client.keystore.jks
ssl.keystore.password=<ENCRYPTED PASSWORD WITH JOURNEY ENCRYPTION TOOL> ssl.key.password=<ENCRYPTED
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#####Web#####  
#####

## Unica JourneyをSSLで構成する

#####Unica JourneySSL #####Unica Journey SSL #####

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。

##### Journey with SSL #####

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。

#. https:// host.domain:SSL\_port /unica

#####

- 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 #https#####


# Chapter 9. 設定

#####Journey#####SMS#####CRM###REST#####

## デフォルトの電子メール接続の設定する

#####Unica Link#####[##]#####


#####

-  > **Link > 電子メール**を選択する。  
[電子メール]ページが表示されます。
- [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。
- [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

#####Unica LinkSMS#####[##]#####SMS#####

#####SMS#####

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



Note:

#####Journey#####

##### Twilio #####Journey#####

- <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.

#####Unica Journey#####.<#####><#####><10#####>#####

##+1 540-324-1212#####Unica Journey#####+15403241212#####


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

### CRM #####[##] ##### CRM #####


##### CRM #####

1.  > **Link > CRM** 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#####ADTECH#####


ADTECH#####

1. 選択  > **Link > ADTECH**  
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

#####

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. 選択  > **リンク > データベース**  
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#####

Unica Link Mailchimp#Mandrill#Salesforce#Twilio ##### **[ Existing Connections ( n ) ] ##**  
#####n#####

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** 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。  
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** 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。  
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** 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。  
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**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。  
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 ##### Journey#####
#####
```

### 新しい REST 統合の作成します

```
REST
```


1.  > **REST**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。  
**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。  
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##### REST #####




REST #####

1.  **> REST**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

### REST #####

### REST #####

1.  **> REST**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。
  - リストから必要な REST 統合を選択します
  - 選択する   **>****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。
  - **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

```
REST
REST ##### REST ##### (on page 32)#####.
REST
```

1.  > **REST**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。
  - 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#####Proxy#####
#####Proxy#####Deliver#Link#Platform#####
```

```
Journey Web - Deliver#Link#Platform#####Journey####/SMS/AdTech Point#####
```

```
Journey Engine - Proxy#####Deliver/Link Server#####E###/SMS/Adtech#####
```

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

#####

## APIドキュメント

#####Journey#RESTAPI#####