

FUJITSU Hybrid IT Service FJcloud-O

API Management

ユーザーズガイド

第 1.15 版

富士通株式会社

FUJITSU Hybrid IT Service FJcloud-O
API Management ユーザーズガイド 第 1.15 版

発行日 2022 年 4 月
All Right Reserved, Copyright© 富士通株式会社

●本書の無断複製・転載を禁じます。

はじめに

本書は、FUJITSU Hybrid IT Service FJcloud-O API Management(以下、本サービス)を利用していただくための、操作方法や使用例について記載したマニュアル(以下、本マニュアル)です。本サービスは、様々な Web サービスの API を管理する統合 API プラットフォームです。

本サービスでは、プラン及びオプションによってご利用いただける機能が異なります。本書では、特定のプラン及びオプションでご利用いただける機能については、以下のアイコンで表現しています。

Pro	Pro のプランでご利用いただける機能です。 (東日本リージョン 1 のみ)
ゲートウェイ拡張	ゲートウェイ拡張構成でご利用いただける機能です。
バックエンドセキュア接続	バックエンドセキュア構成でご利用いただける機能です。
フルアナリティクス	フルアナリティクスオプションでご利用いただける機能です。 (東日本リージョン 3、西日本リージョン 3 のみ)

※ アイコンがない機能については、すべてのプランでご利用いただけます。

お願い

●本書の内容について予告無く変更をおこなうことがありますのでご了承ください。

目次

1. はじめに	7
1.1. サービス概要	8
1.2. 機能概要	9
1.3. ログイン	11
1.3.1. ログイン方法	11
1.3.2. 管理コンソールのログインパスワードの変更方法	13
1.4. パスワードポリシー	14
1.4.1. パスワードポリシー	14
1.4.2. パスワード強度	14
2. チュートリアル	15
2.1. API Proxy の作成	16
2.2. API Proxy のエンハンス	25
2.3. API Proxy の修正	32
3. 画面一覧	34
3.1. Dashboard	35
3.2. APIs	36
3.2.1. API Proxies	36
3.2.1.1. List	36
3.2.1.2. Analytics	37
3.2.1.3. New API Proxy	38
3.2.1.4. Offline Trace	43
3.2.1.5. Roles	43
3.2.2. Shared Flows	44
3.2.2.1. New Shared Flow	44
3.2.3. Environment Configuration	47
3.2.3.1. Caches	47
3.2.3.1.1. New Cache	47
3.2.3.2. Flow Hooks	49
3.2.3.3. Key Value Maps	50
3.2.3.4. Target Servers	51
3.2.3.5. Virtual Hosts	55
3.2.4. API Proxy 画面	56
3.2.4.1. OVERVIEW	56
3.2.4.2. DEVELOP	58
3.2.4.3. TRACE	63
3.2.4.4. Node.js Logs	67
3.2.4.5. PERFORMANCE	68
3.2.5. Shared Flows 画面	72
3.2.5.1. OVERVIEW	72
3.2.5.2. DEVELOP	73
3.3. Publish	77
3.3.1. Products	77
3.3.1.1. List	77
3.3.1.2. Analytics	79
3.3.1.3. Product history	80
3.3.1.4. Roles	81
3.3.1.5. New Product	82
3.3.2. Developers	84
3.3.2.1. List	84

3.3.2.2. Analytics.....	86
3.3.2.3. Developer history	87
3.3.2.4. New Developer	88
3.3.3. Developer Apps.....	89
3.3.3.1. List	89
3.3.3.2. Analytics.....	91
3.3.3.3. New Developer App	92
3.4. Analytics.....	93
3.4.1. Proxy Performance	93
3.4.2. Target Performance	95
3.4.3. Cache Performance	97
3.4.4. Latency Analytics.....	98
3.4.5. Error Code Analytics	99
3.4.6. Developer Engagement	101
3.4.7. Traffic Composition.....	104
3.4.8. Devices.....	106
3.4.9. Reports.....	108
3.4.9.1. New Custom Report.....	111
3.4.10. 統計情報取得用 WebAPI	115
3.5. Admin	116
3.5.1. Organization Users	116
3.5.1.1. List	116
3.5.1.2. New User	118
3.5.2. Organization Roles	119
3.5.2.1. List	119
3.5.2.2. New Custom Role	121
3.5.3. TLS Certificates.....	123
3.5.3.1. 証明書管理用 WebAPI.....	124
3.5.4. Organization History	125
4. Policy 一覧.....	126
4.1. TRAFFIC MANAGEMENT	127
4.2. SECURITY.....	128
4.3. MEDIATION	129
4.4. EXTENSION.....	130
4.4.1. Java Callout Policy	131
4.4.1.1. Java Callout Policy 管理用 WebAPI.....	131
付録.....	132
A.1. 用語集	133

1. はじめに

ここでは、本サービスのご利用にあたり、前提となる情報を説明します。

1.1. サービス概要

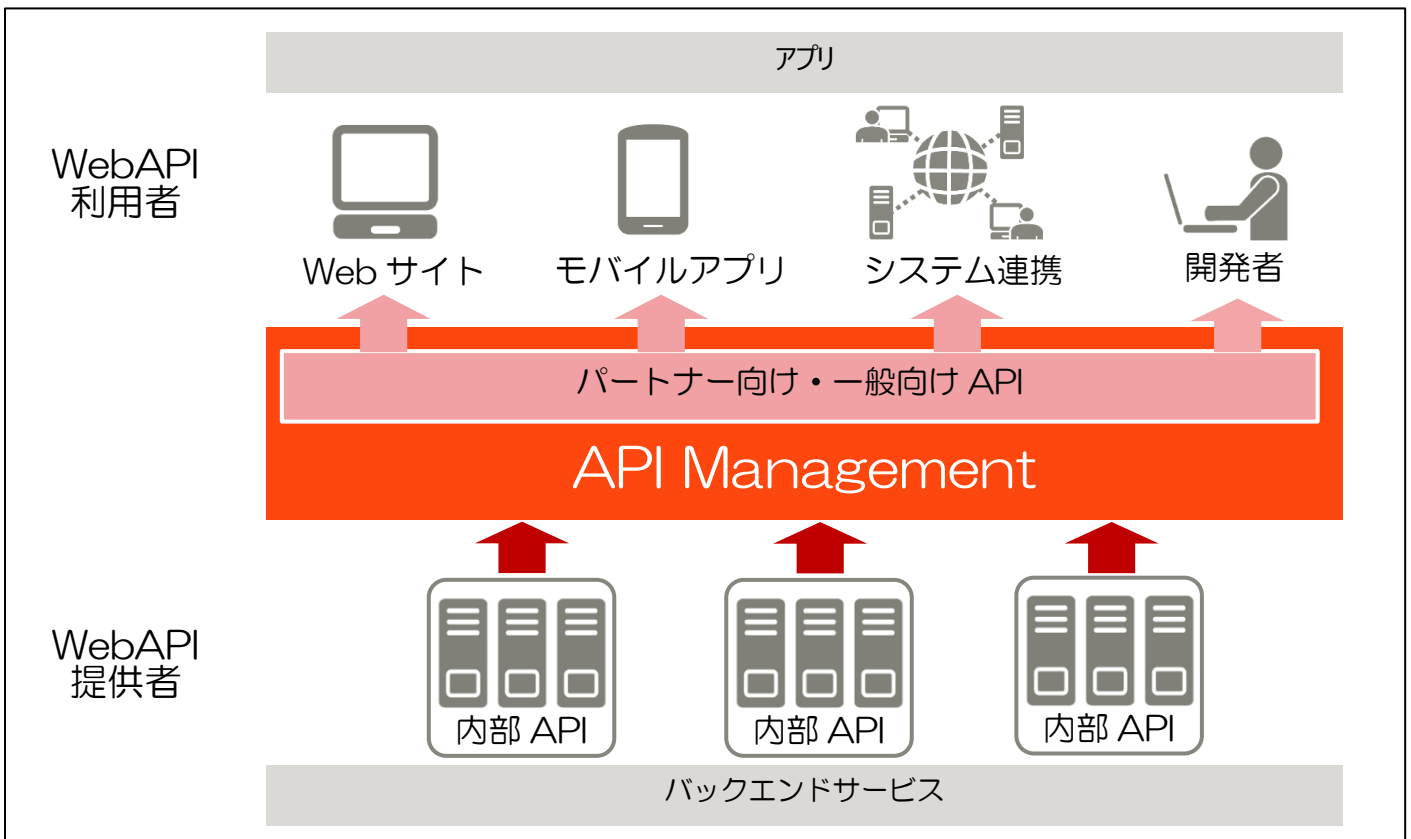
本サービスは、Web API の Gateway サービスです。

Web API サービスを提供されているプロバイダ様や、システムに Web API のインタフェースを開発予定のお客様に価値をもたらします。

API Management は、API を提供するサービスとそれを利用するクライアントアプリとの間に位置し、API の Gateway として機能します。

API Management は、トラフィック制御、認証、セキュリティ、ペイロード加工などのさまざまな機能を備えています。

API Management 経由で API を実行することで、API にそれらの機能を持たせることができるほか、API 自体の開発も可能です。



➤ メリット

- ◇ コーディングなしで機能を追加できるため、API 開発のコストダウンに繋がります
- ◇ デバッグ機能、デプロイ機能、開発用環境などの開発機能を利用することで、効率的な開発を実現します
- ◇ API の詳細な統計をグラフィカルに確認できるため、パフォーマンス改善、エラー検知、利用状況確認に役立ちます
- ◇ API のパッケージング機能によりビジネスプランに合わせた柔軟な API 製品の提供が可能です

1.2. 機能概要

➤ API への機能追加

追加できる機能	トラフィック	リクエスト数の制限
		スパイクによるバックエンドサービスへの負荷の抑止
		バックエンドサービスへの同時接続数の制御
	キャッシュ	レスポンスキャッシュ
		通信データのキャッシュ
		Key Value Store
	認証	Basic 認証
		OAuth1.0a/2.0
		API Key 認証
		SAML アサーションのチェック/生成
		署名付き JWT の生成/検証
	セキュリティ	LDAP を利用したアクセス制御
		IP 制限
		コンテンツフィルタリング（正規表現、JSON/XML 構造の制限、SOAP (XSD/WSDL チェック)）
	データ	データ形式変換 (JSON⇄XML)
		リクエスト/レスポンスデータの編集
		XSL 変換
		エラー処理
	カスタム	ユーザースクリプト (JavaScript) による独自機能の追加
		独自ディメンションの定義（統計での確認）
	その他	API Management の独自情報取得 (Product、Developer、App 等)
		ロギング
機能の追加時に利用できる開発機能	開発者管理	ユーザー登録
		ロール設定
	テスト環境	
	デバッグ	
	バージョン管理	

➤ API の提供

API のパッケージング	複数 API のパッケージング
	使用量制限（リクエスト数の上限設定）
	アクセス制限（リクエストメソッドの制限）
API 利用者管理	API を利用するアプリの管理
	API を利用するアプリの開発者管理

➤ 統計

レスポンス数
レスポンス時間
キャッシュのヒット率
エラー統計
リクエストを実行したアプリ毎の統計
独自統計の作成

1.3. ログイン

1.3.1. ログイン方法

本サービスの管理コンソールにログインするには、下記手順に従ってください。

1. 以下の URL へアクセスしてください。

`https://{FQDN}/login`

※利用可能なブラウザは「API Management 動作環境」を参照してください。

※{FQDN}は下記を参照し、ご契約のリージョンに対応した FQDN に書き換えて使用してください。

リージョン名称	FQDN
東日本リージョン 1	rev-jp1.apimng.com
東日本リージョン 3	rev-jpe3.apimng.com
西日本リージョン 3	rev-jpw3.apimng.com
マルチリージョン	rev-jpe3.apimng.com または rev-jpw3.apimng.com

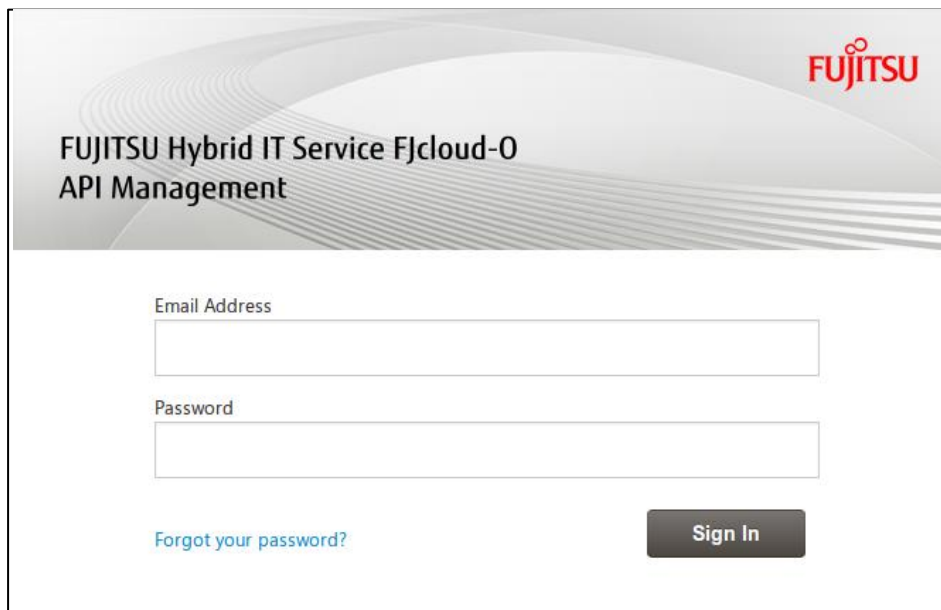
2. 認証ウィンドウが表示されたら、下記の認証情報を入力後、「ログイン」ボタンを押下してください。

ユーザー名：各ユーザーに通知された OTP 認証 ID

パスワード：認証アプリに表示された 6 桁の数字



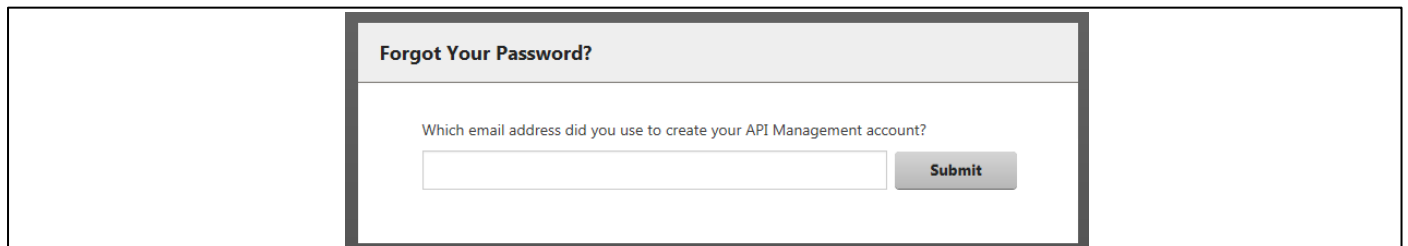
3. API Management 管理コンソールのログイン画面が表示されたら、サービス利用開始通知書に記載（またはご利用者が追加）された Email Address および Password を入力後、「Sign In」ボタンを押下してください。



The screenshot shows the login interface for the Fujitsu Hybrid IT Service FJcloud-0 API Management. At the top right is the Fujitsu logo. The main heading is "FUJITSU Hybrid IT Service FJcloud-0 API Management". Below this, there are two input fields: "Email Address" and "Password". To the left of the "Password" field is a blue link that says "Forgot your password?". To the right of the input fields is a dark grey button labeled "Sign In".

- Forgot your password?

パスワードを忘れてしまった場合は、「Forgot your password?」のリンクをクリックしてください。E-mailの送信フォームに、ご利用いただいているE-mailアドレスを入力し、「Submit」ボタンを押下するとパスワードリセットのメールが届きます。メールに従って、パスワードを再設定してください。



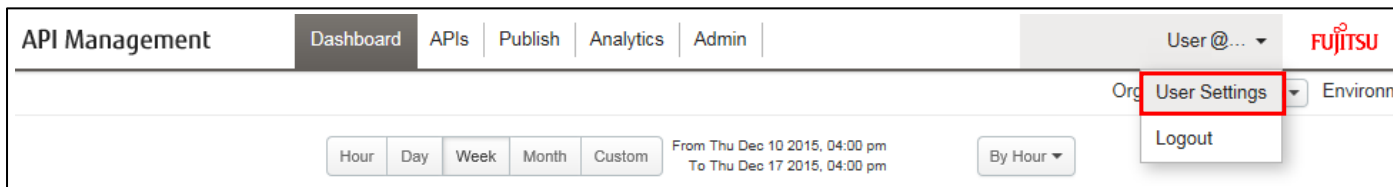
The screenshot shows a form titled "Forgot Your Password?". The form asks the user, "Which email address did you use to create your API Management account?". There is a single input field for the email address and a grey button labeled "Submit" to the right of the field.

1.3.2. 管理コンソールのログインパスワードの変更方法

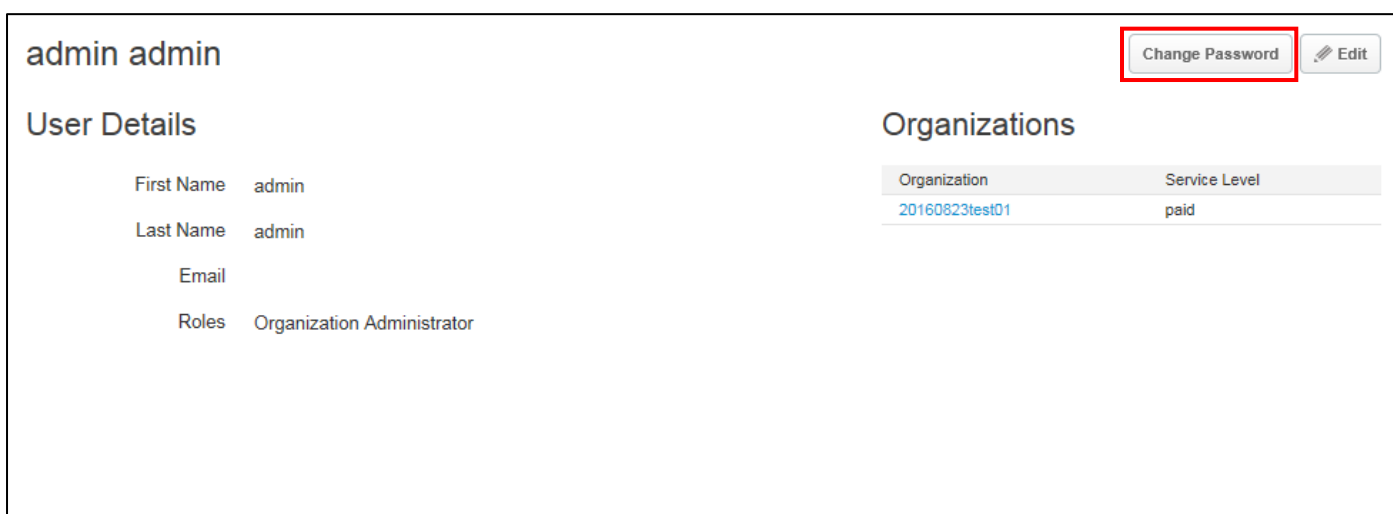
管理コンソールのログインパスワードを変更する場合は、ログイン後に下記手順を実施してください。ログインパスワードを忘れた場合は、1.3.1 ログイン方法の「Forgot your Password」の項を参照してください。

- パスワードの変更手順

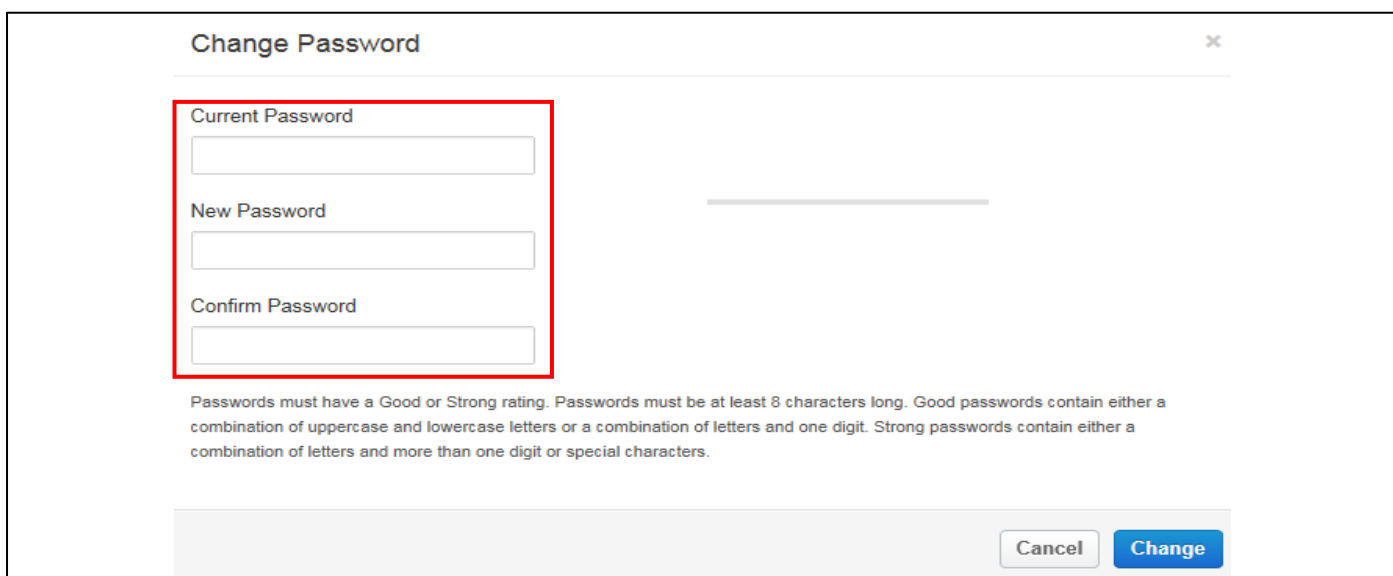
①画面上部のメニューバーから「User Settings」を選択します。



②「Change Password」ボタンを押下します。



③現在のパスワードおよび新しいパスワードを2回入力して「Change」ボタンを押下します。



④ログアウトして、新しく設定したパスワードでログインできることを確認してください。

1.4. パスワードポリシー

1.4.1. パスワードポリシー

パスワードポリシーは下記の通りです。

パスワード連続誤りによりロックされるまでの回数	5回	ログイン時に、誤ったパスワードを入力し、5回連続でログインに失敗した場合、15分間ログインできなくなります。
ロック期間	15分	
パスワードの有効期間	90日	90日以内にパスワードを変更しなかった場合、ログインできなくなります。 ログインできなくなってしまった場合には、ログイン画面上の「Forgot your password?」からパスワードを再発行する必要があります。
パスワード世代チェック	4世代	パスワード変更時に、4世代前までのパスワードを再設定できません。

1.4.2. パスワード強度

ユーザー作成時のパスワード設定時やパスワード変更時には、パスワードが以下の条件を満たしている必要があります。

パスワード長	16文字以上
パスワード強度	全ての条件を満たす必要があります。 <ul style="list-style-type: none">・英小文字を1文字以上含む・英大文字を1文字以上含む・数字を1文字以上含む・記号を1文字以上含む・ユーザー名を含まない

2. チュートリアル

ここでは、例を取り上げて API Proxy 開発の一連のフローについて説明します。

本チュートリアルでサンプルとして使用しております「livedoor 天気予報 API」は、2020年7月31日でサービスが終了しました。

設定例につきましては、参考手順としてご覧いただけますよう、お願い申し上げます。

2.1. API Proxy の作成

API Proxy の作成から API の実行までのフローについて、具体例を示しながら説明します。

今回取り上げる例の概要およびフローは下記の通りです。

- 概要

レスポンスとして JSON データを返す API に対して、データ形式変換をおこなう API Proxy を設定し、XML のレスポンスを返すようにします。

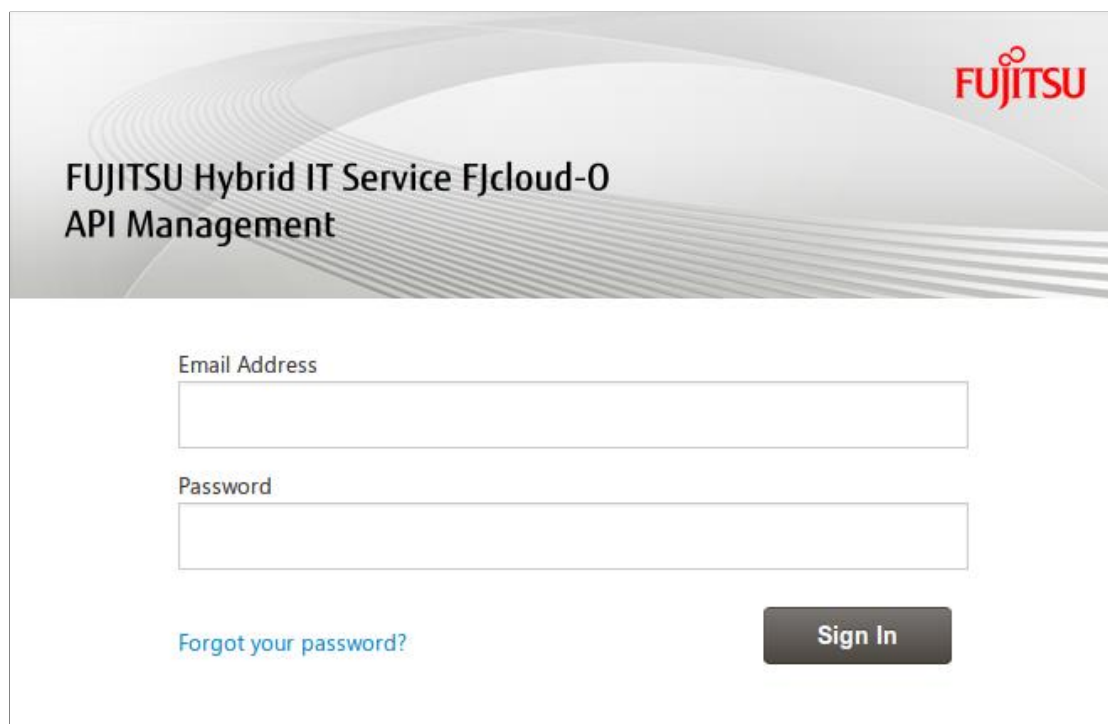
- フロー

1. ログイン
2. ユーザーの作成
3. API Proxy の作成
4. JSON to XML Policy の追加
5. デバッグ
6. 本番環境へのデプロイ
7. API の実行
8. 統計の確認

【作成手順】

1. ログイン

ユーザーID、パスワードを入力しログインします。



The screenshot shows the login interface for the Fujitsu Hybrid IT Service FJcloud-0 API Management. At the top right is the Fujitsu logo. The main heading reads "FUJITSU Hybrid IT Service FJcloud-0 API Management". Below this, there are two input fields: "Email Address" and "Password". To the left of the "Password" field is a link that says "Forgot your password?". To the right of the "Password" field is a dark grey button labeled "Sign In".

2. ユーザーの作成

API Proxy 作成時に使用するアカウントを作成します。

※サービス契約時に払い出された管理者アカウントでも API Proxy の作成は可能ですが、開発用ユーザーの作成をお勧めします。今回の例では開発用のユーザーを利用します。

「Admin」メニュー→「Organization Users」→「+User」ボタン→ユーザー情報の入力→「Save」ボタン

➤ ユーザー情報の入力

New User

Email

Roles Business User Operations Administrator User

Email	指定したメールアドレスがユーザーID になります。「Save」ボタンをクリックすると、アカウントを有効にするためのログインに必要な情報がメールで届きます。
Roles	「Business User」「Operations Administrator」「User」を指定します。 ※ユーザー管理、環境設定を除いた全機能を利用できるようになります

メールの内容に従って、アカウントのアクティベートおよびパスワード設定を行います。
完了したら、作成したアカウントを使って再度ログインします。

3. API Proxy の作成

JSON データを返す API に対する API Proxy を作成します。

- ① 「APIs」メニュー→「API Proxies」→「+API Proxy」ボタン→「Reverse proxy (most common)」にチェック→「Next」ボタン

➤ Build a Proxy (TYPE)

Build a Proxy

Reverse proxy (most common)
Route inbound requests to backend services.
 Optionally associate the proxy with an OpenAPI (Swagger) document

SOAP service
Create a RESTful or pass-through proxy for a SOAP service.

No Target
Create a simple API proxy that does not route to any backend target.

Node.js App
Create a new app in JavaScript and optionally add policies.

Proxy bundle
Import an existing proxy from a zip archive.

② Proxy を下記の通り設定後、「Next」ボタンを押下します。

➤ Proxy 情報の入力

Proxy Name	sampleproxy
Proxy Base Path	/v1/sampleproxy
Existing API	http://weather.livedoor.com/forecast/webservice/json/v1 ※JSON データをレスポンスで返す API
Description	Proxy の説明文を入力します。(省略可)

➤ Build a Proxy (DETAILS)

Build a Proxy

TYPE > **DETAILS** > SECURITY > VIRTUAL HOSTS > BUILD > SUMMARY

Specify the proxy details.

Proxy Name*
Valid characters are letters, numbers, dash (-), and underscore (_).

Proxy Base Path*
A path component that uniquely identifies this API proxy. The public-facing URL of this API proxy is comprised of your organization name, an environment where this API proxy is deployed, and this Proxy Base Path. Example URL <http://example-test.apigee.net/v1/sampleproxy>

Existing API*
Defines the target URL invoked on behalf of this API proxy. Any URL that is accessible over the open Internet can be used. Example: <https://api.usergrid.com>

Description

③ Proxy を下記の通り設定後、「Next」ボタンを押下します。

Authorization	Pass through (none)
Browser	チェックしない

➤ Build a Proxy (SECURITY)

The screenshot shows the 'Build a Proxy' configuration page with the 'SECURITY' tab selected. The breadcrumb navigation includes TYPE, DETAILS, SECURITY, VIRTUAL HOSTS, BUILD, and SUMMARY. The main heading is 'Secure access for users and clients.' Under 'Authorization', 'Pass through (none)' is selected with a radio button, while 'API Key' and 'OAuth 2.0' are unselected. Under 'Browser', 'Add CORS headers' is unselected. At the bottom, there are 'Previous', 'Exit Without Saving', and 'Next' buttons.

④ 設定を変えずに「Next」ボタンを押下します。

➤ Build a Proxy (VIRTUAL HOSTS)

The screenshot shows the 'Build a Proxy' configuration page with the 'VIRTUAL HOSTS' tab selected. The breadcrumb navigation includes TYPE, DETAILS, SECURITY, VIRTUAL HOSTS, BUILD, and SUMMARY. The main heading is 'Select the virtual hosts this proxy will bind to when it is deployed. You must select at least one virtual host. [Learn more...](#)' Below this is a table with columns for Name, Environment, and Host Aliases. The 'Name' column has checkboxes for 'default' and 'secure', both of which are checked. The 'Environment' column lists 'prod' and 'test' for each name. The 'Host Aliases' column contains input fields for each combination. At the bottom, there are 'Previous', 'Exit Without Saving', and 'Next' buttons.

<input checked="" type="checkbox"/> Name	Environment	Host Aliases
<input checked="" type="checkbox"/> default	prod	<input type="text"/>
	test	<input type="text"/>
<input checked="" type="checkbox"/> secure	prod	<input type="text"/>
	test	<input type="text"/>

⑤ 「Build and Deploy」 ボタンを押下します。

➤ Build a Proxy (BUILD)

The screenshot shows the 'Build a Proxy' interface with the 'BUILD' step selected in the progress bar. The message reads: 'You are ready to build and deploy your API proxy.' Below this, there are several configuration options:

- Deploy Environments: prod test
- Proxy Name: sampleproxy
- Proxy Type: Reverse proxy
- Virtual Hosts: default
- Security: None
- Browser: Do not allow direct requests from a browser via CORS

At the bottom, there are three buttons: 'Previous', 'Exit Without Saving', and 'Build and Deploy'.

⑥ 作成結果が緑色のチェックで表示されることを確認します。

➤ Build a Proxy (SUMMARY)

The screenshot shows the 'Build a Proxy' interface with the 'SUMMARY' step selected in the progress bar. The message reads: 'View [sampleproxy](#) proxy in the editor.'

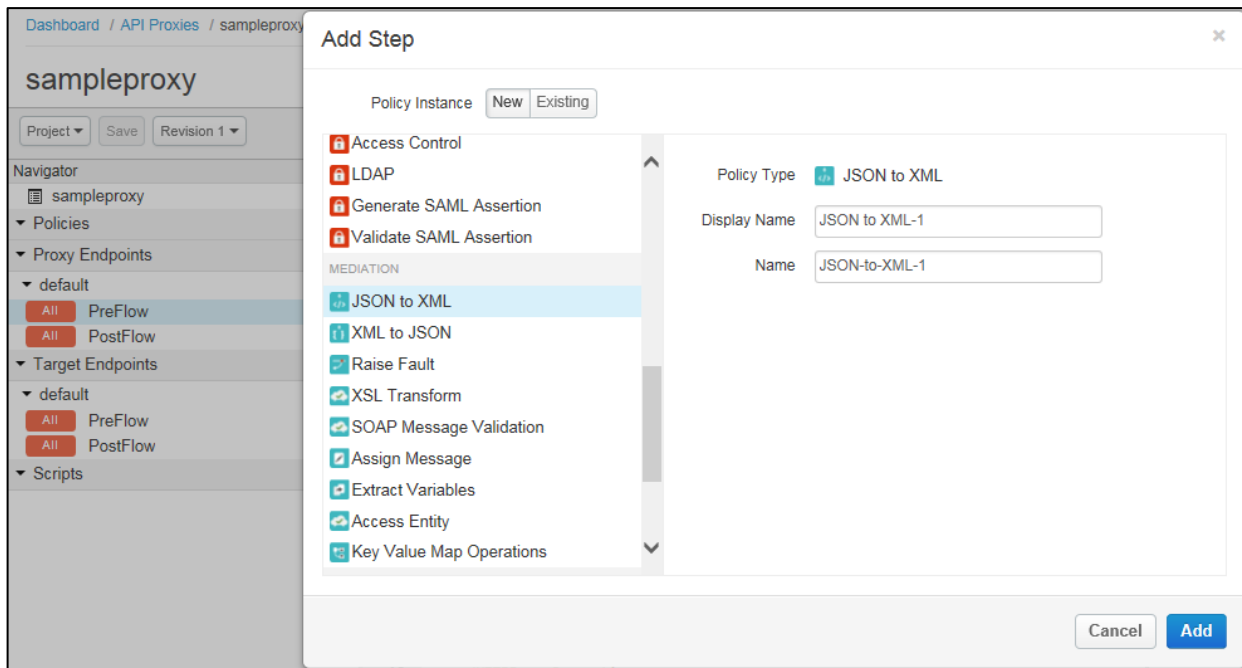
Below the message, there are three green bars, each with a green checkmark and text:

- Generated proxy
- Uploaded proxy
- Deployed to test

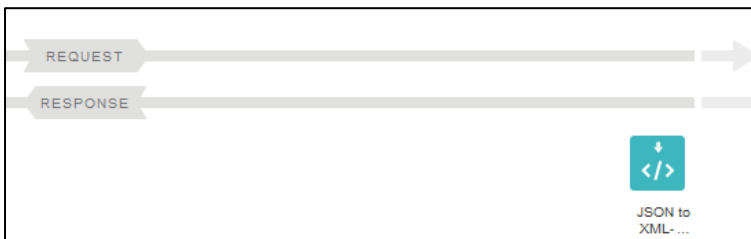
4. JSON to XML Policy の追加

API Proxy に対して、JSON データを XML データに変換する Policy を追加します。

「APIs」メニュー→「API Proxies」→「sampleproxy」リンク→「DEVELOP」タブ→「+Step」ボタン（「Flow: PreFlow」ペインの RESPONSE 側）→「JSON to XML」を選択→「Add」ボタン



Policy が追加されます。「Save」をクリックし、テスト環境にデプロイします。



5. デバッグ

XML に変換されることを TRACE 画面で確認します。

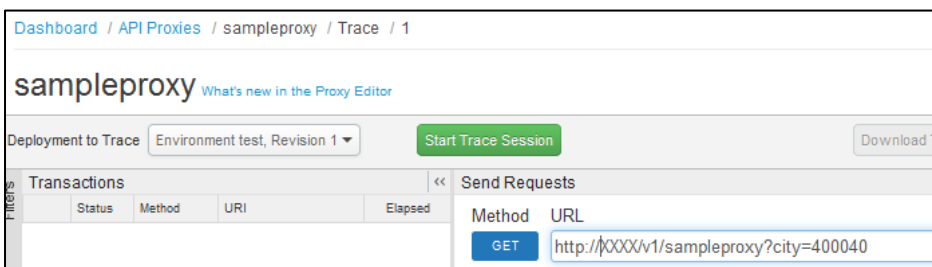
「TRACE」タブ→「URL」の指定→「Start Trace Session」ボタン→「Send」ボタン→レスポンスデータの確認→「Stop Trace Session」ボタン

➤ 「URL」の指定

<http://XXXX/v1/sampleproxy?city=400040>

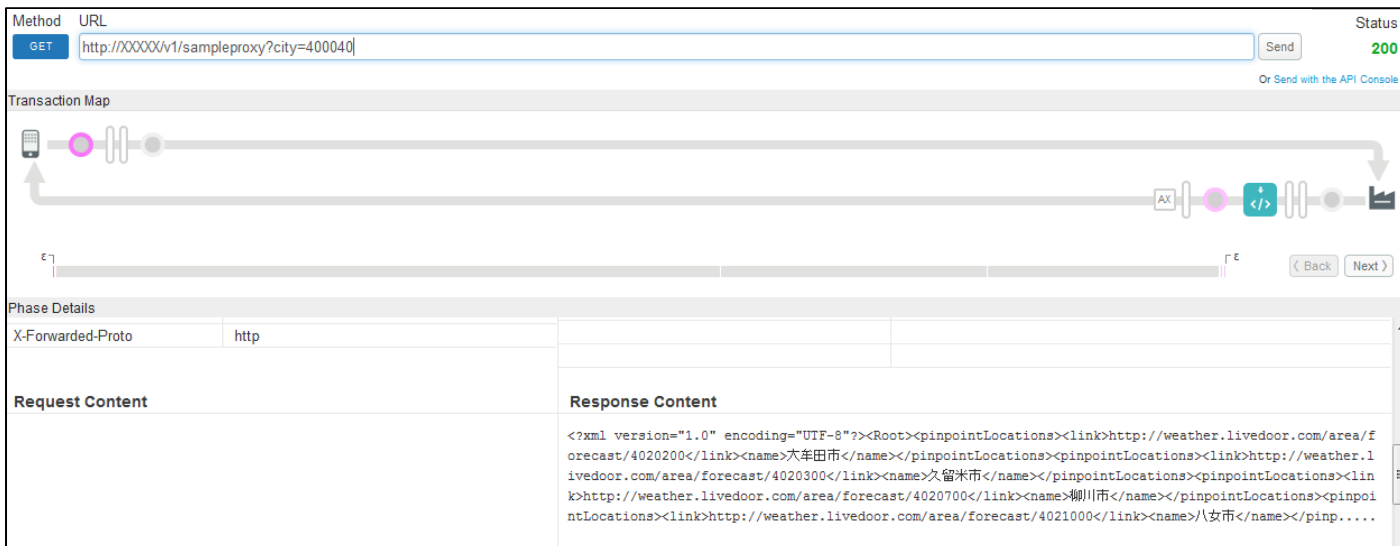
※XXXX 部分はお客様毎に異なります。TRACE 画面の表示時に元々入力されていたものを使用して下さい。

※クエリは、API Proxy 作成時に「Backend Service URL」で指定した URL に依存します。今回の例では「city=400040」を指定して下さい。



➤ レスポンスデータの確認

「Phase Details」ペインの「Response Content」にXMLデータが入っていることを確認します。



The screenshot shows the API proxy interface. At the top, the Method is GET and the URL is `http://XXXXXX/v1/sampleproxy?city=400040`. The Status is 200. Below the URL bar is a Transaction Map showing a sequence of events. The Phase Details section is expanded to show the Response Content, which is XML data:

```
<?xml version="1.0" encoding="UTF-8"?><Root><pinpointLocations><link>http://weather.livedoor.com/area/forecast/4020200</link><name>大牟田市</name></pinpointLocations><pinpointLocations><link>http://weather.livedoor.com/area/forecast/4020300</link><name>久留米市</name></pinpointLocations><pinpointLocations><link>http://weather.livedoor.com/area/forecast/4020700</link><name>柳川市</name></pinpointLocations><pinpointLocations><link>http://weather.livedoor.com/area/forecast/4021000</link><name>八女市</name></pinp.....
```

6. 本番環境へのデプロイ

テストが完了したので、API Proxy を本番環境にデプロイします。

「APIs」メニュー→「API Proxies」→「sampleproxy」リンク→「Deployment」→「prod」→「Deploy」

デプロイが完了すると、prod も test 同様に緑になります。



The screenshot shows the API proxy dashboard for 'sampleproxy'. The breadcrumb is 'Dashboard / API Proxies / sampleproxy / Overview / 1'. The main heading is 'sampleproxy'. Below it are buttons for 'Project', 'Save', 'Revision 1', and 'Deployment'. The 'Deployment' dropdown menu is open, showing two options: 'prod' and 'test', both with green circles next to them, indicating they are active. Below the deployment menu is the 'Revision 1 Summary' section, which shows 'Created: Apr 18th, 2017' and 'Updated: Apr 18th, 2017'.

7. API の実行

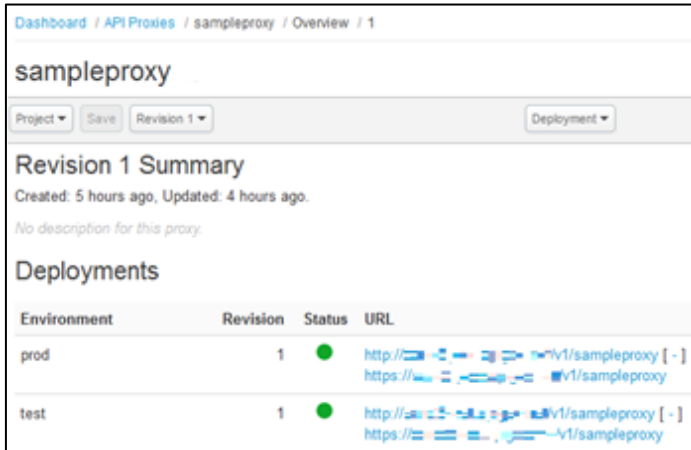
本番環境へのデプロイが完了し、API を実行するための準備が整ったので API を実行します。

URL の確認→API の実行

➤ URL の確認

「APIs」メニュー→「API Proxies」→「sampleproxy」リンク→「Deployments」

本番環境（prod）の HTTPS の URL を確認します。



Dashboard / API Proxies / sampleproxy / Overview / 1

sampleproxy

Project Save Revision 1 Deployment

Revision 1 Summary

Created: 5 hours ago, Updated: 4 hours ago.
No description for this proxy.

Deployments

Environment	Revision	Status	URL
prod	1	●	http://.../v1/sampleproxy [-] https://.../v1/sampleproxy
test	1	●	http://.../v1/sampleproxy [-] https://.../v1/sampleproxy

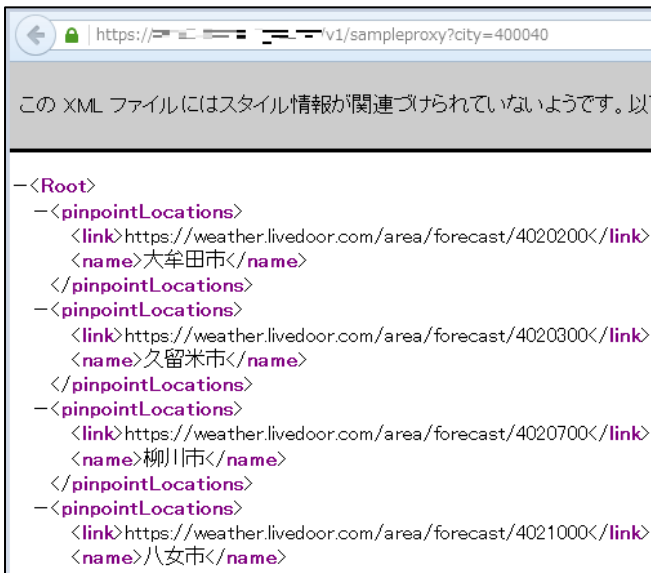
➤ API の実行

ブラウザから API を実行します。

URL は、上記で確認した URL に「?city=400040」を追加した URL になります。

例、<https://XXXX/v1/sampleproxy?city=400040>

XML データが取得できることを確認します。



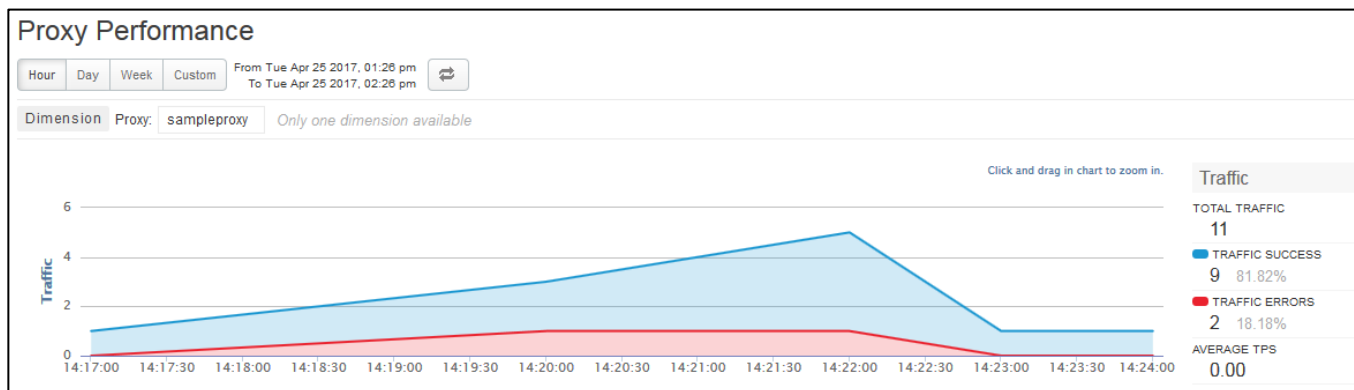
この XML ファイルにはスタイル情報が関連づけられていないようです。以

```
-<Root>
-<pinpointLocations>
  <link>https://weather.livedoor.com/area/forecast/4020200</link>
  <name>大牟田市</name>
</pinpointLocations>
-<pinpointLocations>
  <link>https://weather.livedoor.com/area/forecast/4020300</link>
  <name>久留米市</name>
</pinpointLocations>
-<pinpointLocations>
  <link>https://weather.livedoor.com/area/forecast/4020700</link>
  <name>柳川市</name>
</pinpointLocations>
-<pinpointLocations>
  <link>https://weather.livedoor.com/area/forecast/4021000</link>
  <name>八女市</name>
</pinpointLocations>
```

8. 統計の確認

API を実行した分だけトラフィックが増えていることを確認します。

「Analytics」 → 「Proxy Performance」



※反映に時間がかかる場合があります。

2.2. API Proxy のエンハンス

既存の API Proxy に機能追加する際のフローについて、具体例を示しながら説明します。

今回取り上げる例の概要およびフローは下記の通りです。

- 概要

「2.1 API Proxy の作成」で作成した API に API Key 認証用の Policy を追加して、事前登録されたアプリからのみ API を受けるようにします。

Policy 追加後、バージョンを上げて API をデプロイします。

- フロー

1. Policy の追加
2. API のバージョンアップ
3. Product の作成
4. アプリ開発者の登録
5. アプリの登録
6. API Key の確認
7. デバッグ

【作成手順】

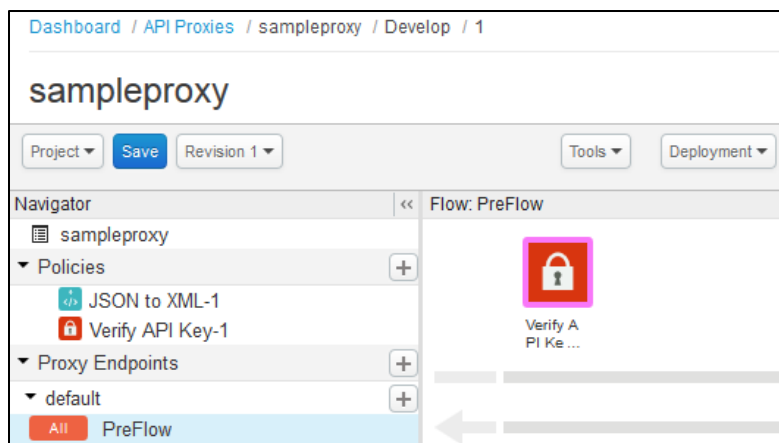
1. Policy の追加

API Proxy に事前登録されたアプリからの API のみ許可するために、Policy を2つ追加します。

1つ目の Policy を追加します。

API の実行時に、アプリから送られてくる API Key をチェックするための Policy を追加します。

「APIs」メニュー→「API Proxies」→「sampleproxy」リンク→「DEVELOP」タブ→「+Step」ボタン（「Flow: PreFlow」ペインの REQUEST 側）→「Verify API Key」を選択→「Add」ボタン



2つ目の Policy を追加します。
チェック後に API Key をリクエストから削除する Policy を追加します。
これによりバックエンドサービスには、API Key の情報は送信されなくなります。

「+Step」ボタン（「Flow: PreFlow」ペインの REQUEST 側）→「Assign Message」を選択→「Add」ボタン→Policy 属性の設定

➤ Policy 属性の設定

「追加されたアイコン（Assign Message-1）」をクリック→「Policy Assigning Message-1」ペインの内容を下記の内容で置き換える

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<AssignMessage async="false" continueOnError="false" enabled="true" name="remove-query-param-apikey">
  <DisplayName>Remove Query Param apikey</DisplayName>
  <Remove>
    <QueryParams>
      <QueryParam name="apikey"/>
    </QueryParams>
  </Remove>
  <IgnoreUnresolvedVariables>true</IgnoreUnresolvedVariables>
  <AssignTo createNew="false" transport="http" type="request"/>
</AssignMessage>
```

The screenshot shows the API Gateway console interface. At the top, there are two icons: a red padlock icon labeled 'Verify A PI Ke...' and a blue pencil icon labeled 'Remove Query...'. Below these icons, there are two horizontal bars representing 'REQUEST' and 'RESPONSE' flow. A '+ Step' button is visible on the left. The main area shows the configuration for the policy 'Remove Query Param apikey'. The XML configuration is displayed in a code editor with line numbers 1 through 11. The configuration is as follows:

```
1 <?xml version="1.0" encoding="UTF-8" standalone="yes"?>
2 <AssignMessage async="false" continueOnError="false" enabled="true" name="remove-query-param-apikey">
3   <DisplayName>Remove Query Param apikey</DisplayName>
4   <Remove>
5     <QueryParams>
6       <QueryParam name="apikey"/>
7     </QueryParams>
8   </Remove>
9   <IgnoreUnresolvedVariables>true</IgnoreUnresolvedVariables>
10  <AssignTo createNew="false" transport="http" type="request"/>
11 </AssignMessage>
```

2. APIのバージョンアップ

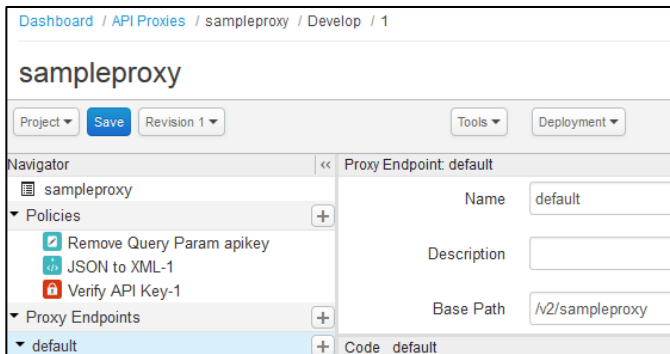
APIのバージョンを上げます。

作業としては、API実行時のURIを変更するだけです。

※エンハンス時にURIを変更したくないケースでは、本作業は不要になります

「Navigator」ペインの「Proxy Endpoints」の「default」をクリック→「Base Path」の変更

Base Path : /v1/sampleproxy → /v2/sampleproxy



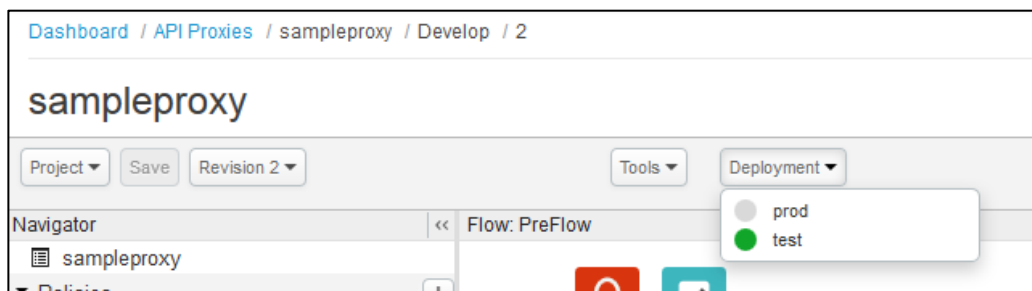
変更が完了したら保存後、デプロイします。

保存時にはリビジョンを上げます。

リビジョンを上げることで、エンハンス前のAPI Proxyを旧版として残しておくことができます。

※旧版が不要なケースでは、リビジョンを上げる必要はありません

「Project」ボタン→「Save as New Revision」→「Deployment」→「test」



3. Product の作成

Product は API Proxy のパッケージです。

API Key 認証により API を実行できるアプリを制限するためには、Product を作成する必要があります。アプリが利用する API (API Proxy) を含む Product を作成し、アプリに紐づけます (紐付は「5 アプリの登録」で行います)。

「Publish」メニュー → 「Products」 → 「+Product」ボタン → Product 情報の入力 → 「Save」ボタン

➤ Product 情報の入力

New Product Products give developers access to your APIs.

Product Details

Name: sampleProduct
Display Name: sampleProduct
Description:
Environment: prod test
Access: Internal only — Visible only to developers at example during app registration
 Private — Visible only to external developers with explicit permission during app registration
 Public — Visible only to any registered developer during app registration
Key Approval Type: Automatic Manual
Quota: requests every Select...
Allowed OAuth Scopes:
Comma separated scope names.

Resources

API Proxy: Select... Revision: Select... Resource Path: Select... Import Resource
Paths: Resource Path Actions
+ Custom Resource
API Proxies: sampleproxy Actions
Delete + API Proxy

Custom

Attributes: Name Value Actions
+ Custom Attribute
Cancel Save

Name	sampleProduct
Environment	test、 prod
Access	Internal only
Key Approval Type	Automatic
API Proxies	「+API Proxy」ボタン → 「sampleproxy」を選択

4. アプリ開発者の登録

API Key 認証を利用するためには、API Proxy を利用するアプリを事前に登録しておく必要があります。
アプリの登録時には、アプリ開発者を予め登録しておく必要があります。

「Publish」メニュー→「Developers」→「+Developer」ボタン→アプリ開発者情報の入力→「Save」ボタン

➤ アプリ開発者情報の入力

Dashboard / Developers / New Developer

New Developer

Developer Details

First Name

Last Name

Email

Username

Custom Attributes

Name

First Name	サンプル
Last Name	開発者
Email	※アプリ開発者のメールアドレス
Username	sampleDeveloper ※アプリ開発者のユーザーID（任意）

5. アプリの登録

アプリを登録します。

Product にアプリを紐づけることによって、Product に含まれる API (API Proxy) をアプリから実行できるようになります。

「Publish」 → 「Developer Apps」 → 「+Developer App」 → アプリ情報の入力 → 「Save」 ボタン

➤ アプリ情報の入力

Dashboard / Developer Apps / New Developer App Organization example

New Developer App

Developer App Details

Name: sampleApp
Display Name: サンプルアプリ
Developer: サンプル 開発者 (xxxxxxxx@j...)
App Status:
Callback URL:
A callback URL is required only for 3-legged OAuth.
Notes:

Credentials

Expiration: Never Duration Date
Products: sampleProduct
+ Product

Custom Attributes

Name	Value	Actions
		+ Custom Attribute

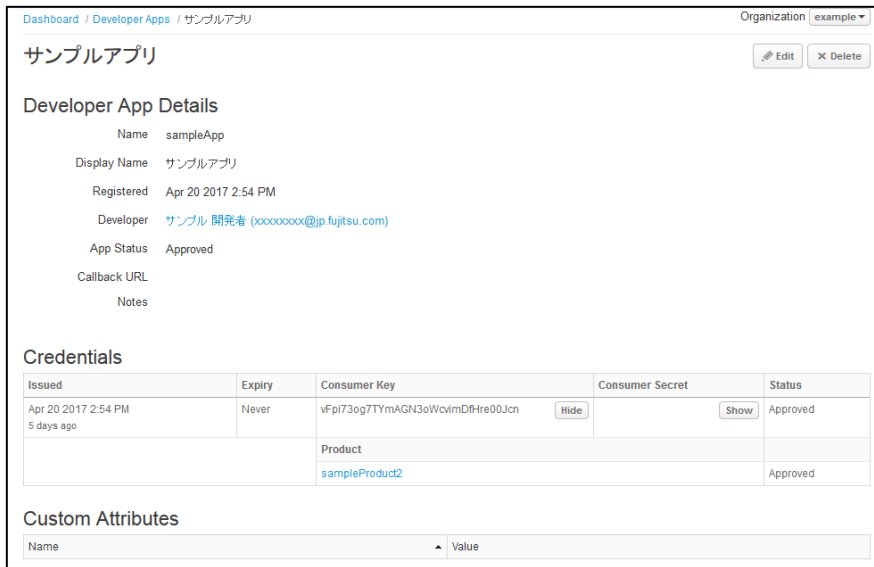
Cancel Save

Name	sampleApp
Display Name	サンプルアプリ
Developer	サンプル開発者
Products	「+Product」 → 「sampleProduct」 を選択

6. API Key の確認

アプリから API を実行するために必要になる API Key を確認します。

「Developer Apps」画面の「サンプルアプリ」リンク→「Consumer Key」の「Show」ボタン



7. デバッグ

API Key 認証ができることをテストします。

「APIs」→「API Proxies」→「sampleproxy」リンク→「TRACE」タブ→「URL」の指定→「Send」ボタン→Status コードが 200 になることの確認

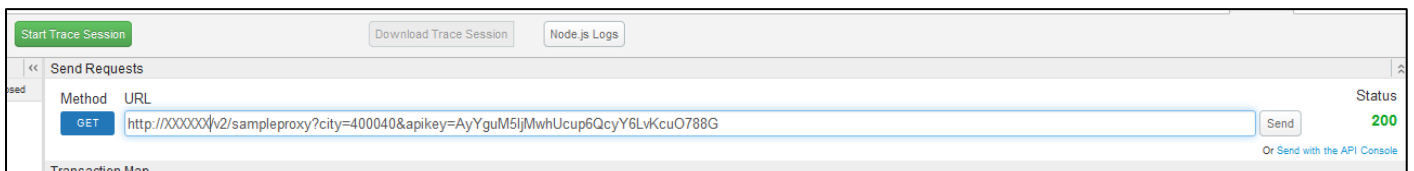
➤ 「URL」の指定

<http://XXXX/v2/sampleproxy?city=400040&apikey=YYYY>

※XXXX 部分はお客様毎に異なります。TRACE 画面の表示時に元々入力されていたものを使用して下さい

※YYYY には API Key を指定して下さい

※API をバージョンアップしたため、パスが v1 から v2 に変わっています



API Key を指定しない場合には、Status コードが 401 になることを確認します。

「URL」から apikey クエリの削除→「Start Trace Session」ボタン→「Send」ボタン→ステータスコードの確認→「Stop Trace Session」ボタン



残りの作業は、「2.1 API Proxy の作成」の「本番環境へのデプロイ」以降の作業と同様です。

2.3. API Proxy の修正

既存の API Proxy を修正する際のフローについて、具体例を示しながら説明します。

今回取り上げる例の概要およびフローは下記の通りです。

- 概要

「2.1 API Proxy の作成」で作成した API に変更を加えます。

現状の API では、JSON データを一律 XML に変換していますが、リクエスト元が Firefox の場合にだけ XML に変換するよう変更を加えます。

変更前の版を残しておくために、API Proxy のリビジョンを上げます。

- フロー

1. Policy 実行条件の設定
2. API Proxy のリビジョンアップ
3. デバッグ

【作成手順】

1. Policy 実行条件の設定

JSON を XML に変換する Policy の実行条件を設定します。

User-Agent ヘッダーに「Firefox」という文字列を含む場合にのみ、Policy を実行するようにします。

「APIs」 → 「API Proxies」 → 「sampleproxy」リンク → 「DEVELOP」タブ → Policy 属性の設定

➤ Policy 属性の設定

「Code default」ペインの内容に変更を加えます。

「<Name>JSON-to-XML-1</Name>」の一つ上に以下の設定を追加します。

```
<Condition>request.header.User-Agent JavaRegex ".*Firefox.*"</Condition>
```

The screenshot shows the configuration interface for an API Proxy. On the left, there are expandable sections for 'default', 'Target Endpoints', and 'Scripts'. The main area displays the 'Code default' section with XML code. The code is as follows:

```
5 <Request>
6   <Step>
7     <Name>Verify-API-Key-1</Name>
8   </Step>
9   <Step>
10    <Name>remove-query-param-apikey</Name>
11  </Step>
12 </Request>
13 <Response>
14   <Step>
15    <Condition>request.header.User-Agent JavaRegex ".*Firefox.*"</Condition>
16    <Name>JSON-to-XML-1</Name>
17  </Step>
18 </Response>
19 </Preflow>
```


2. API Proxy のリビジョンアップ

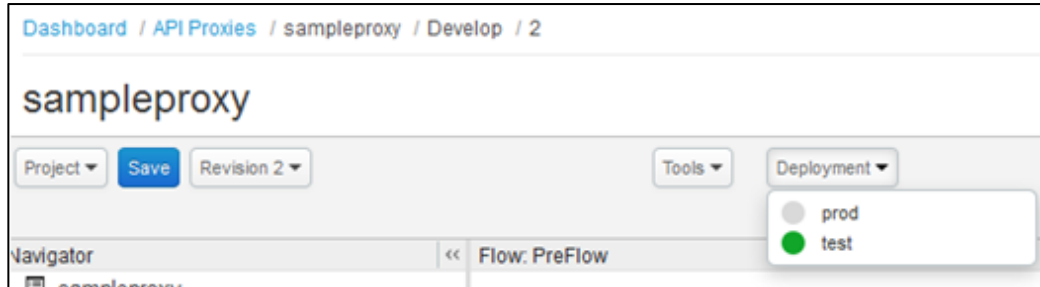
変更した API Proxy を保存した後、デプロイします。

今回は、リビジョンを上げて API Proxy を保存します。

リビジョンを上げることで、変更前の API Proxy を旧版として残しておくことができます。

※旧版が不要なケースでは、リビジョンを上げる必要はありません

「Project」 → 「Save as New Revision」 → 「Deployment」 → 「test」



※リビジョンを上げてデプロイすると、旧リビジョンの API Proxy はアンデプロイされます

3. デバッグ

JSON to XML Policy の実行がスキップされることを確認します。

※トレース画面で API を実行した場合、User-Agent ヘッダーの値は「NING/1.0」となるため、Policy の実行はスキップされます

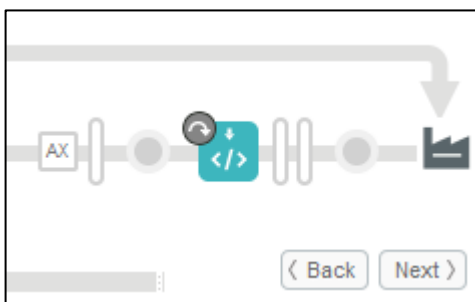
「TRACE」タブ→「URL」の指定→「Start Trace Session」ボタン→「Send」ボタン→「Stop Trace Session」ボタン→Policy がスキップされていることの確認

➤ 「URL」の指定

URL : <http://XXXX/v1/sampleproxy?city=400040>

※XXXX 部分はお客様毎に異なります。TRACE 画面の表示時に元々入力されていたものを使用して下さい

➤ Policy がスキップされていることを確認



Policy のアイコンの左上に、スキップされたことを示すマークが付きます

残りの作業は、「2.1 API Proxy の作成」の「本番環境へのデプロイ」以降の作業と同様です。

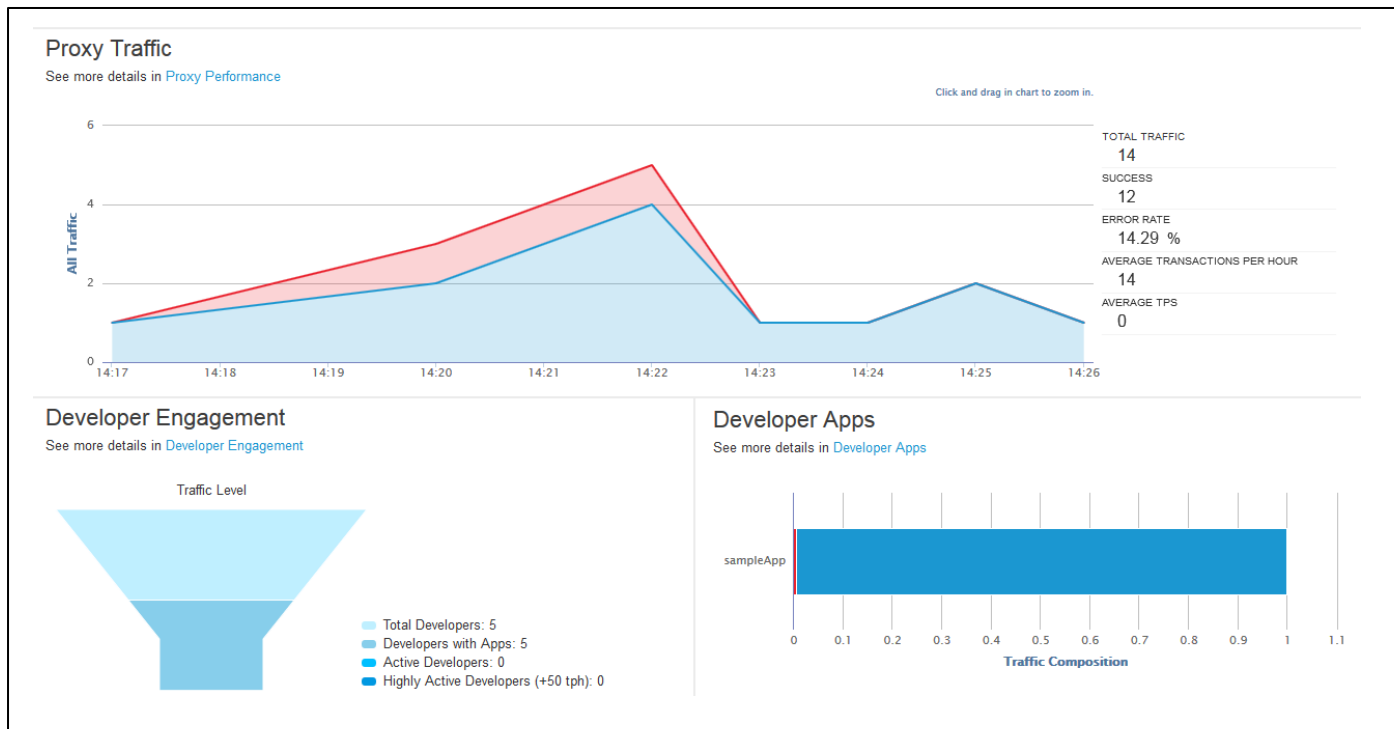
API の実行は Firefox から行い、レスポンスデータが XML であることを確認します。

3. 画面一覧

ここでは、各画面について説明します。

3.1. Dashboard

Dashboard 画面です。Proxy Traffic、Developer Engagement、Developer Apps の 3 グラフが 1 画面に表示されます。各グラフの「See more details in <graph name>」をクリックすることで、Analytics 機能の詳細ページを表示します。詳細ページにつきましては、3.4 Analytics を参照してください。



● Proxy Traffic

TOTAL TRAFFIC	API Proxy 全体のリクエスト数を表示します。
SUCCESS	成功したリクエスト数を表示します。
ERROR RATE	失敗したリクエスト数を表示します。
AVERAGE TRANSACTION PER HOUR	1 時間あたりの平均リクエスト・レスポンス数を表示します。
AVERAGE TPS	1 秒あたりの平均リクエスト・レスポンス数を表示します。

● Developer Engagement

Total Developers	組織内の API Proxy に関連するアプリ開発者の合計人数を表示します。
Developers with Apps	組織内のアプリに関連するアプリ開発者の合計人数を示します。
Active Developers	API Proxy 経由でリクエストを送信したアプリ開発者の人数を表示します。
Highly Active Developers (+50 tph)	1 時間あたりのトランザクション数が 50 を超えるアプリ開発者の人数を表示します。

● Developer Apps

アプリの名前とトラフィック量（成功・失敗）を表示します。

3.2. APIs

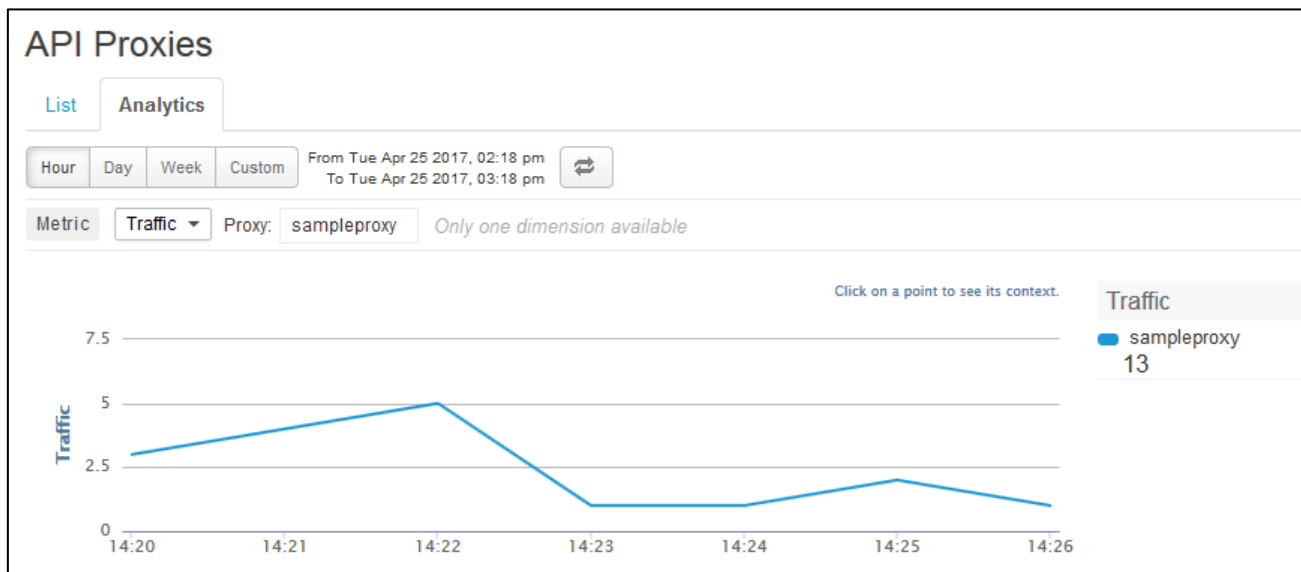
3.2.1. API Proxies

3.2.1.1. List

API Proxy の一覧画面です。

API Proxies								
List	Analytics							
Search	All ▾	<input type="text"/>	1-4 of 4		◀	▶	Offline Trace	+ API Proxy
Metrics for Last 24 Hours (test)								
API Proxy	Environments	Traffic	Message Trend by Hour	Avg Time	Error Rate	Modified ▾	Actions	
sampleproxy	test, prod	17		58.41 ms	35.29 %	15 minutes ago	Delete Roles	
api-test	test	10		8.19 ms	0 %	4 days ago	Delete Roles	
docs	test, prod	0				4 days ago	Delete Roles	
web-monitoring	test, prod	812		27.92 ms	0 %	11 days ago	Delete Roles	

3.2.1.2. Analytics



● Metrics

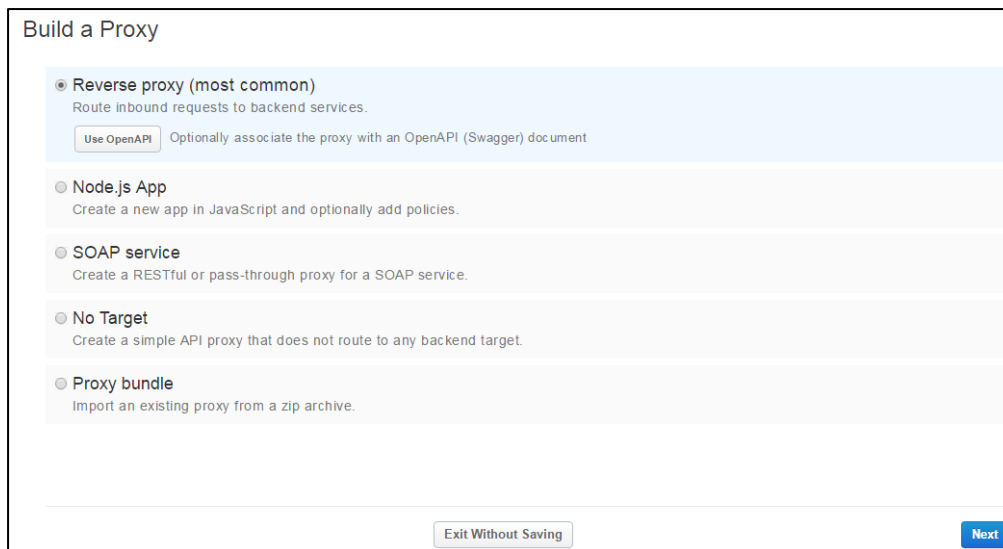
Traffic	リクエスト数を表示します。 ※他の Metric を選択した場合、再度選択することはできません。
Errors	エラーになったリクエスト数を表示します。
Average Data Exchange	リクエストとレスポンスのデータ量を表示します。
Average Target Response Time	API Proxy がバックエンドサービスにリクエストの全データを送り終えた時点から、バックエンドサービスからのレスポンスを全て受け取り終わるまでの時間の平均を表示します。
Average Response Time	API Proxy がリクエスト情報を受け取り終わった時点から、クライアントにレスポンスを返し始める時点までの、時間の平均を表示します。
Maximum Response Time	最大レスポンス時間を表示します。
Average Transactions per Second	1 秒あたりのリクエスト数+レスポンス数の平均を表示します。
Average Transactions per Minute	1 分あたりのリクエスト数+レスポンス数の平均を表示します。

3.2.1.3. New API Proxy

API Proxy の作成画面です。作成画面はウィザード形式となっており、複数の画面で分けて設定します。作成する API Proxy のタイプ選択、Base Path の指定、セキュリティの設定、バーチャルホストのバインド設定が可能です。

- Build a Proxy (TYPE)

API Proxy のタイプを選択します。



Reverse proxy (most common)	バックエンドサービスとして Web サービスを設定する場合に選択します。 ※OpenAPI document を利用して API Proxy を作成する事が可能です。
Node.jp App	バックエンドサービスとして Node.js を設定する場合に選択します。 ※OpenAPI document を利用して API Proxy を作成する事が可能です。
SOAP service	バックエンドサービスとして WSDL を設定する場合に選択します。
No Target	バックエンドサービスを指定しません。 ※OpenAPI document を利用して API Proxy を作成する事が可能です。
Proxy bundle	ファイルで指定した内容で API Proxy を作成します。 3.2.4.1 OVERVIEW や 3.2.4.2 DEVELOP でダウンロードした API Pro xy 情報 (zip 形式) を指定することが可能です。

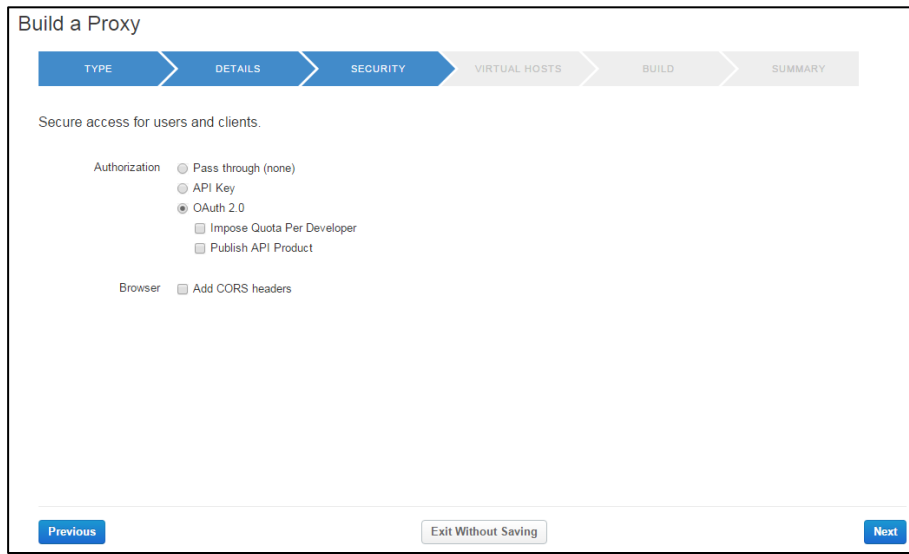
- Build a Proxy (DETAILS)

API Proxy の詳細を設定します。

Proxy Name	API Proxy の名前を指定します。
Proxy Base Path	API Proxy にアクセスする際のパスを指定します。API Proxy 毎に固有な識別子を指定する必要があります。
Existing API	API Proxy のタイプで Reverse proxy を指定した場合に表示されます。バックエンドサービスとして指定する Web サービスの URL を設定します。指定した URL とその子リソースに対してアクセスを許可します。URL の親パスにはアクセスできません。
Source	API Proxy のタイプで Node.js App を指定した場合に表示されます。サンプルの Hello World を選択するか JavaScript や JSON のファイルをアップロードすることが可能です。
WSDL	API Proxy のタイプで SOAP service を指定した場合に表示されます。WSDL の URL (サンプル有り) または、ファイルを指定することが可能です。
ZIP Bundle	API Proxy のタイプで Proxy Bundle を指定した場合に表示されます。ZIP 形式のファイルを指定する必要があります。
Description	API Proxy の説明を指定します。

● Build a Proxy (SECURITY)

Security に関する Policy やクロスドメイン通信を設定します。



Pass through (none)	Security に関する Policy が設定されていない状態の API Proxy が作成されます。 ※Policy は 3.2.4.2 DEVELOP で追加することが可能です。
API Keys	API Key に関する Policy (Verifi API Key、Assign Message) が設定された状態の API Proxy を作成します。
OAuth v2.0	OAuth v2.0 に関する Policy (OAuth2.0、Assign Message) が設定された状態の API Proxy を作成します。
Impose Quota per Developer	チェックを入れると、Policy (Impose Quota) が設定された状態の API Proxy を作成します。
Publish API Product	チェックを入れると、API Proxy と一緒に Products も作成します。
Add CORS headers	チェックを入れると、CORS によるクロスドメイン通信を許可する Policy (Assign Message) が追加されます。

- Build a Proxy (VIRTUAL HOSTS)

API Proxy をデプロイする際にバインドするバーチャルホストを選択します。少なくとも 1 つは選択する必要があります。

The screenshot shows the 'Build a Proxy' wizard at the 'VIRTUAL HOSTS' step. The progress bar at the top indicates the current step. Below the progress bar, there is a text instruction: 'Select the virtual hosts this proxy will bind to when it is deployed. You must select at least one virtual host. [Learn more...](#)'. A table lists available virtual hosts with checkboxes for selection.

<input checked="" type="checkbox"/>	Name	Environment	Host Aliases
<input checked="" type="checkbox"/>	default	prod	
<input checked="" type="checkbox"/>		test	
<input checked="" type="checkbox"/>	secure	prod	
<input checked="" type="checkbox"/>		test	

At the bottom of the form, there are three buttons: 'Previous', 'Exit Without Saving', and 'Next'.

- Build a Proxy (BUILD)

API Proxy 作成前の確認画面です。

The screenshot shows the 'Build a Proxy' wizard at the 'BUILD' step. The progress bar at the top indicates the current step. Below the progress bar, there is a text instruction: 'You are ready to build and deploy your API proxy.' The form contains several configuration options for the proxy.

Deploy Environments: prod test

Proxy Name: test

Proxy Type: Reverse proxy

Virtual Hosts: default, secure

Security: None

Browser: Do not allow direct requests from a browser via CORS

At the bottom of the form, there are three buttons: 'Previous', 'Exit Without Saving', and 'Build and Deploy'.

Deploy Environments

API Proxy 作成後、チェックを入れた環境 (prod、test) にデプロイします。

- Build a Proxy (SUMMARY)

API Proxy 作成結果が表示されます。青文字の Proxy 名をクリックすると 3.2.4.1 OVERVIEW の画面に遷移します。

Build a Proxy

TYPE > DETAILS > SECURITY > VIRTUAL HOSTS > BUILD > SUMMARY

- ✓ Generated proxy
- ✓ Uploaded proxy
- ✓ Deployed to test

[View test proxy in the editor](#) .

3.2.1.4. Offline Trace

保存しておいたトレース情報を分析するための画面です。トレース情報は、TRACE 画面の「Download Trace Session」ボタンで保存します。詳細は 3.2.4.3 TRACE を参照してください。

3.2.1.5. Roles

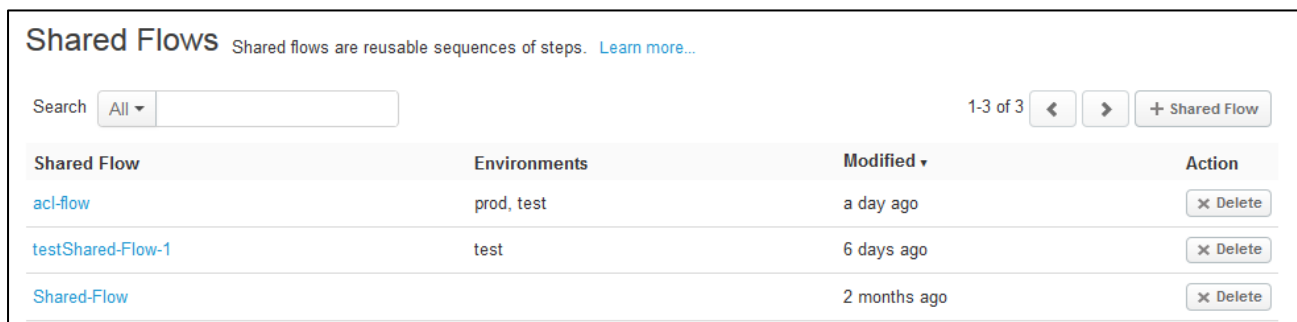
カスタムロール（3.5.2.2 New Custom Role 参照）に対して API Proxy へのアクセス権を設定する画面です。

Hello

Role	Operations	Actions
productRole	<input checked="" type="checkbox"/> View <input checked="" type="checkbox"/> Edit <input type="checkbox"/> Delete	<input type="button" value="× Delete"/>

3.2.2. Shared Flows

Shared Flow の一覧画面です。

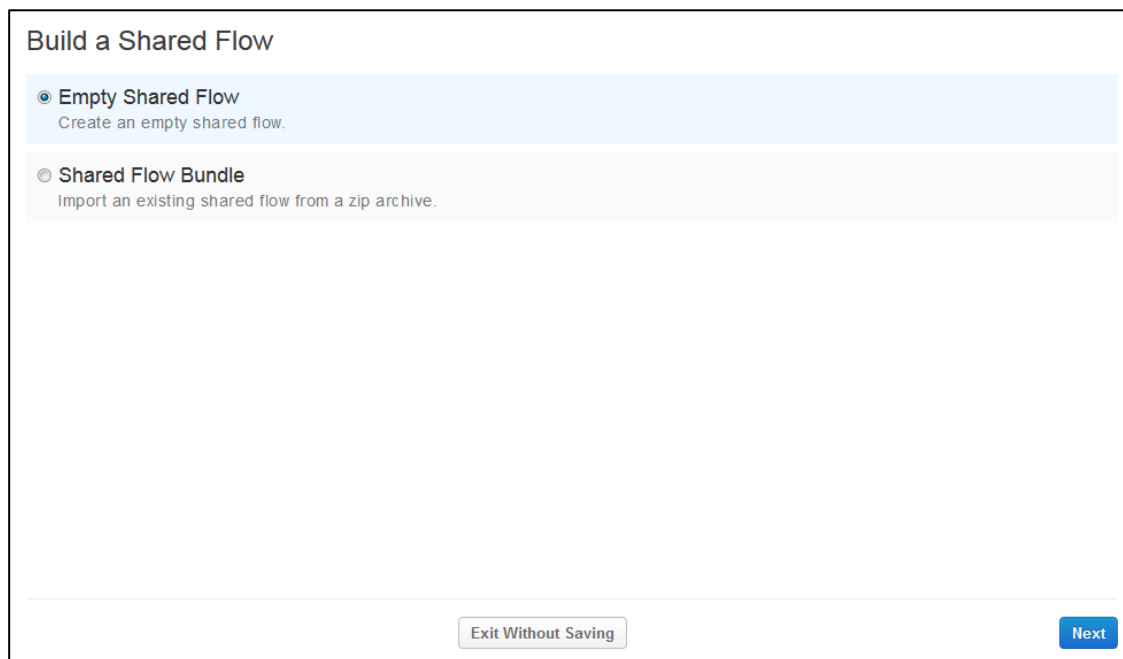


3.2.2.1. New Shared Flow

Shared Flow の作成画面です。作成画面はウィザード形式となっており、複数の画面で分けて設定します。作成する Shared Flow のタイプ選択によって、設定項目が異なります。

- Build a Shared Flow(TYPE)

Shared Flow のタイプを選択します。



Empty Shared Flow	Shared Flow を作成します。
Shared Flow Bundle	ファイルで指定した内容でShared Flow を作成します。3.2.5.1 OVERVIEW や3.2.5.2 DEVELOP でダウンロードしたShared Flow 情報 (zip 形式) を指定することが可能です。

- Build a Shared Flow(DETAILS)

Shared Flow の詳細を設定します。

Shared Flow Name	Shared Flow の名前を指定します。
Description	Shared Flow の説明を指定します。
ZIP Bundle	Shared Flow のタイプでShared Flow Bundleを指定した場合に表示されます。ZIP 形式のファイルを指定する必要があります。

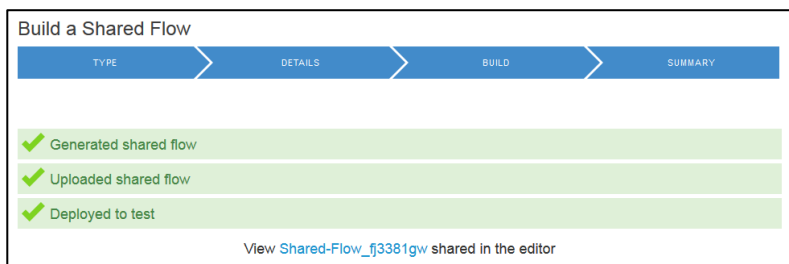
- Build a Shared Flow(BUILD)

Shared Flow 作成前の確認画面です。

Deploy Environments	Shared Flow 作成後、チェックを入れた環境 (prod、test) にデプロイします。
---------------------	--

- Build a Shared Flow(SUMMARY)

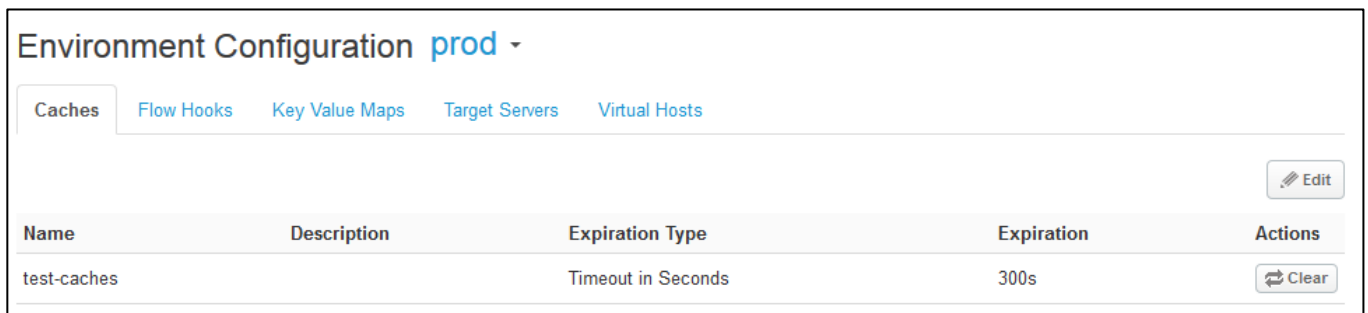
Shared Flow作成結果が表示されます。青文字のShared Flow 名をクリックすると3.2.5.1 OVERVIEWの画面に遷移します。



3.2.3. Environment Configuration

3.2.3.1. Caches

キャッシュの一覧画面です。「Clear」ボタンでキャッシュをクリアすることができます。



Environment Configuration **prod** ▾

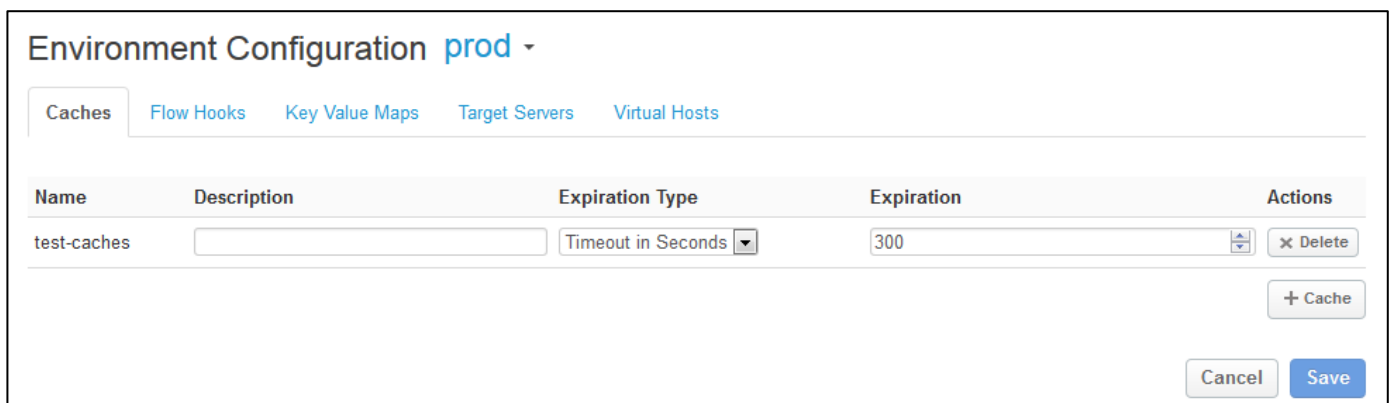
Caches Flow Hooks Key Value Maps Target Servers Virtual Hosts

Edit

Name	Description	Expiration Type	Expiration	Actions
test-caches		Timeout in Seconds	300s	Clear

3.2.3.1.1. New Cache

キャッシュの設定ができます。



Environment Configuration **prod** ▾

Caches Flow Hooks Key Value Maps Target Servers Virtual Hosts

Name	Description	Expiration Type	Expiration	Actions
test-caches	<input type="text"/>	Timeout in Seconds ▾	<input type="text" value="300"/>	× Delete

+ Cache

Cancel Save

- Expiration Type

Expiration 欄で指定する値の単位を指定します。

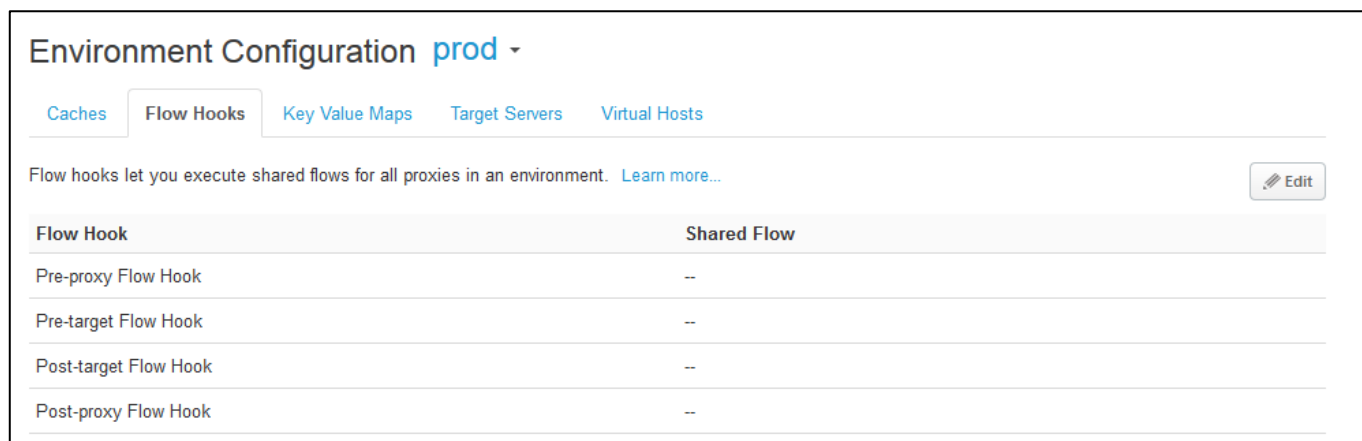
Timeout in Seconds	指定した秒数を経過するとキャッシュをクリアします。
Time of Day	毎日指定した時刻にキャッシュをクリアします。
Date	指定した日付でキャッシュをクリアします。

- Expiration

キャッシュクリアのタイミングを値で指定します。

3.2.3.2. Flow Hooks

Flow Hooks の一覧が表示されます。Flow Hook に Shared Flow を設定することで、同一環境内の API Proxy に対して共通したタイミングで Shared Flow を実行させることができます。



Environment Configuration **prod** ▾

[Caches](#) **Flow Hooks** [Key Value Maps](#) [Target Servers](#) [Virtual Hosts](#)

Flow hooks let you execute shared flows for all proxies in an environment. [Learn more...](#) Edit

Flow Hook	Shared Flow
Pre-proxy Flow Hook	--
Pre-target Flow Hook	--
Post-target Flow Hook	--
Post-proxy Flow Hook	--

- Edit ボタン

「Edit」ボタンを押下すると、それぞれの Flow Hook で実行する Shared Flow の追加・修正が可能になります。

3.2.3.3. Key Value Maps

Key と Value のマッピングリストを設定することができます。画面で設定したマッピングリストは、Key Value Map Operations Policy を使って操作することが可能です。

Environment Configuration **prod** ▾

Caches Flow Hooks **Key Value Maps** Target Servers Virtual Hosts

+ Key Value Map

Name	Action	
▼ tetsu-value	Delete	
	Edit	
Key	Value	Action
test-name-a	1	Delete
tett-name-b	2	Delete

- + Key Value Map ボタン

「+Key Value Map」 ボタンを押下すると、マッピングリストの作成画面を表示します。

New Key Value Map ×

Name

Encrypted


Cancel Add

- Edit ボタン

「Edit」 ボタンを押下すると、マッピングリストに対して Key と Value の追加・修正が可能になります。

3.2.3.4. Target Servers

Target Servers の一覧が表示されます。登録した Target Server は、Target Endpoint の HTTPTarget Connection に設定をおこなうことで、ロードバランシング (RoundRobin、Weighted、LeastConnection) させることができます。

Environment Configuration prod ▾			
Caches Flow Hooks Key Value Maps Target Servers Virtual Hosts			
 Edit			
Name	Host	Port	Enabled
target1	target1.example.com	80	✓
target2	target2.example.com	80	✓

- Edit ボタン

Target Servers の追加・修正が可能になります。

- 【参考】 TargetEndpoint の HTTPTargetConnection 設定例

- 上図の target1、target2 に対する RoundRobin の設定例
※均等な比率で target1 と target2 に負荷分散します。

```
<TargetEndpoint name="default">
  <HTTPTargetConnection>
    <LoadBalancer>
      <Algorithm>RoundRobin</Algorithm>
      <Server name="target1" />
      <Server name="target2" />
    </LoadBalancer>
  </HTTPTargetConnection>
</TargetEndpoint>
```

- リクエストに 5 回失敗した場合、失敗した Target Server をローテーションから除外する設定例
※<MaxFailures>タグで指定した数値に達すると除外します。

```
<TargetEndpoint name="default">
  <HTTPTargetConnection>
    <LoadBalancer>
      <Algorithm>RoundRobin</Algorithm>
      <Server name="target1" />
      <Server name="target2" />
      <MaxFailures>5</MaxFailures>
    </LoadBalancer>
  </HTTPTargetConnection>
</TargetEndpoint>
```

※除外された Target Server を復旧するには、手動で API Proxy を再デプロイ（アンデプロイ→デプロイ）するか、下記設定の例のように HealthMonitor 設定（自動復旧処理）を実装する必要があります。

➤ HealthMonitor の設定例（TCPMonitor）

下記例では、5 秒毎に Target Server の 80 ポートに対してポーリングします。接続できない状態または 10 秒以上応答が無かった場合は障害と判断して失敗カウントを 1 増やし、<MaxFailures>タグで設定した上限を超えると Target Server を除外します。接続に成功した場合は失敗カウントを 0 にリセットし、Target Server が除外されている場合は復旧します。

※HealthMonitor を有効にするには<LoadBalancer> / <MaxFailures>タグは 0 以外の値を設定する必要があります。

```
<TargetEndpoint name="default">
  <HTTPTargetConnection>
    <LoadBalancer>
      <Algorithm>RoundRobin</Algorithm>
      <Server name="target1" />
      <Server name="target2" />
      <MaxFailures>5</MaxFailures>
    </LoadBalancer>
    <HealthMonitor>
      <IsEnabled>true</IsEnabled>
      <IntervallnSec>5</IntervallnSec>
      <TCPMonitor>
        <ConnectTimeoutInSec>10</ConnectTimeoutInSec>
        <Port>80</Port>
      </TCPMonitor>
    </HealthMonitor>
  </HTTPTargetConnection>
</TargetEndpoint>
```

➤ HealthMonitor の設定例（HTTPMonitor）

下記例では、5 秒毎に Authorization ヘッダーを付与した GET リクエストを Target Server に対して送信します。Target Server からステータスコード 200 以外の値またはヘッダー“ImOK”が“YourOK”以外である場合は、障害と判断して失敗カウントを 1 増やし、<MaxFailures>タグで設定した上限を超えると Target Server を除外します。接続に成功した場合は失敗カウントを 0 にリセットし、Target Server が除外されている場合は復旧します。

※HealthMonitor を有効にするには<LoadBalancer> / <MaxFailures>タグは 0 以外の値を設定する必要があります。

```

<TargetEndpoint name="default">
  <HTTPTargetConnection>
    <LoadBalancer>
      <Algorithm>RoundRobin</Algorithm>
      <Server name="target1" />
      <Server name="target2" />
      <MaxFailures>5</MaxFailures>
    </LoadBalancer>
    <HealthMonitor>
      <IsEnabled>true</IsEnabled>
      <IntervalInSec>5</IntervalInSec>
      <HTTPMonitor>
        <Request>
          <ConnectTimeoutInSec>10</ConnectTimeoutInSec>
          <SocketReadTimeoutInSec>30</SocketReadTimeoutInSec>
          <Port>80</Port>
          <Verb>GET</Verb>
          <Path>/healthcheck</Path>
          <Header name="Authorization">Basic 1234567890abcd</Header>
        </Request>
        <SuccessResponse>
          <ResponseCode>200</ResponseCode>
          <Header name=" ImOK" >YourOK</Header>
        </SuccessResponse>
      </HTTPMonitor>
    </HealthMonitor>
  </HTTPTargetConnection>
</TargetEndpoint>

```

- 上図の target1、target2 に対する Weighted の設定例
 ※下記例では、3 リクエストのうち、target1 に 1 回、target2 に 2 回の比率で負荷分散をします。

```

<TargetEndpoint name="default">
  <HTTPTargetConnection>
    <LoadBalancer>
      <Algorithm>Weighted</Algorithm>
      <Server name="target1">
        <Weight>1</Weight>
      </Server>
      <Server name="target2">
        <Weight>2</Weight>
      </Server>
    </LoadBalancer>
  </HTTPTargetConnection>
</TargetEndpoint>

```

- 上図の target1、target2 に対する LeastConnection の設定例
※接続数が少ない TargetServer を優先します。

```
<TargetEndpoint name="default">  
  <HTTPTargetConnection>  
    <LoadBalancer>  
      <Algorithm>LeastConnection</Algorithm>  
      <Server name="target1" />  
      <Server name="target2" />  
    </LoadBalancer>  
  </HTTPTargetConnection>  
</TargetEndpoint>
```

3.2.3.5. Virtual Hosts

バーチャルホストの一覧を表示します。バーチャルホストの設定値は、修正することはできません。

Environment Configuration **prod** ▾

Caches Flow Hooks Key Value Maps Target Servers **Virtual Hosts**

Name	Port	Alias	Actions
default	80	[Redacted]	
🔒 secure	443	[Redacted]	Show

- Show ボタン

「Show」 ボタンを押下すると、SSL 通信の詳細を表示します。

Environment Configuration **prod** ▾

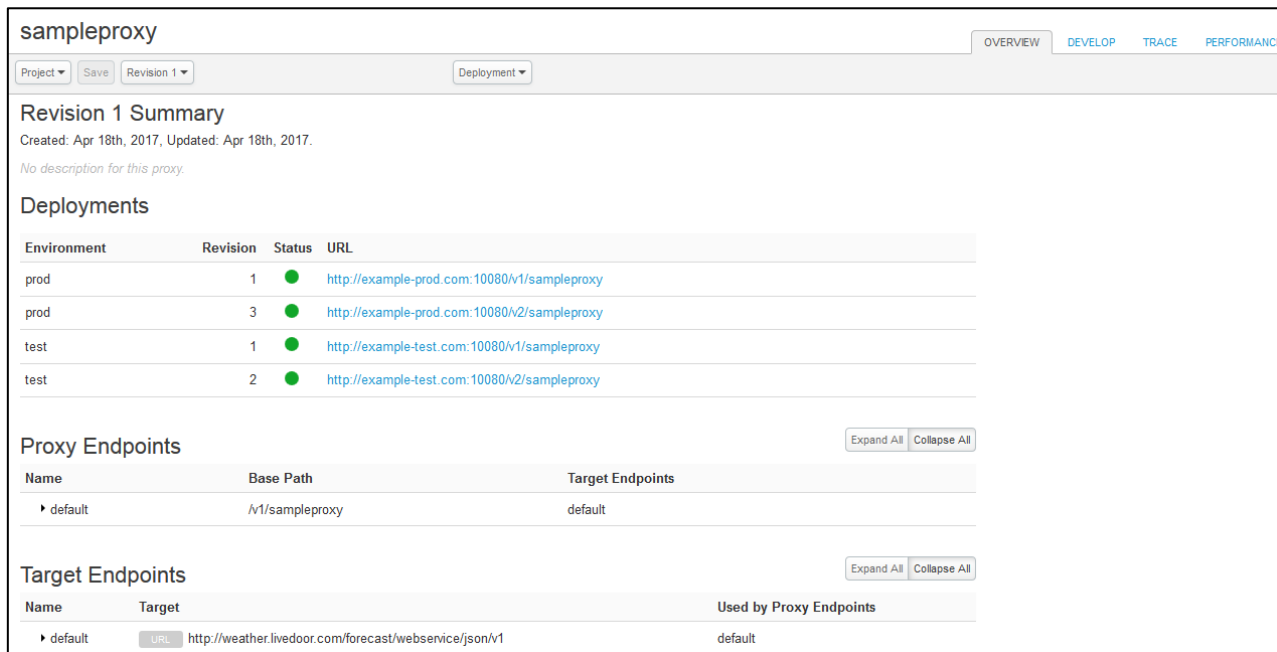
Caches Flow Hooks Key Value Maps Target Servers **Virtual Hosts**

Name	Port	Alias	Actions
default	80	[Redacted]	
Name secure	Port 443	Alias [Redacted]	
SSL Info			
Enabled	true	Client Auth	Disabled
Key Store	service-ks	Trust Store	Hide
Key Alias	ServiceKS	Ciphers	
Ignore Validation Errors	false		

3.2.4. API Proxy 画面

3.2.4.1. OVERVIEW

3.2.1.3 New API Proxy で作成した API Proxy の概要確認の他、API Proxy のリビジョン管理、環境へのデプロイをおこなうことができます。



● Project

Save as New Revision	表示中の API Proxy を新しいリビジョンとして保存します。
Save as New API Proxy...	表示中の API Proxy に新しい名前を付けて保存します。
Undo All	最後に保存した時点からの変更を取り消します。
Delete API Proxy...	表示中の API Proxy を削除します。
Delete Current Revision...	表示中のリビジョンを削除します。
Download Revision	表示中のリビジョンの API Proxy 情報を zip 形式でダウンロードします。
Upload a New Revision...	ダウンロードした API Proxy 情報を読み込み、新しいリビジョンとして保存します。
API Proxy History	表示中の API Proxy の変更履歴を表示します。

● Deployment

prod	表示中の API Proxy を prod 環境にデプロイします。
test	表示中の API Proxy を test 環境にデプロイします。

※名前の左側の●が灰色の場合はアンデプロイ状態、緑色の場合はデプロイ状態を表します。

- Node.js Logs

API Proxy に Node.js が設定されている場合に限り、「Node.js Logs」ボタンが表示されます。ボタンを押下すると、Node.js Logs 画面が表示されます。詳細は 3.2.4.4 Node.js Logs を参照してください。

- Deployments

表示中の API Proxy のデプロイ先情報が表示されます。

- Proxy Endpoints

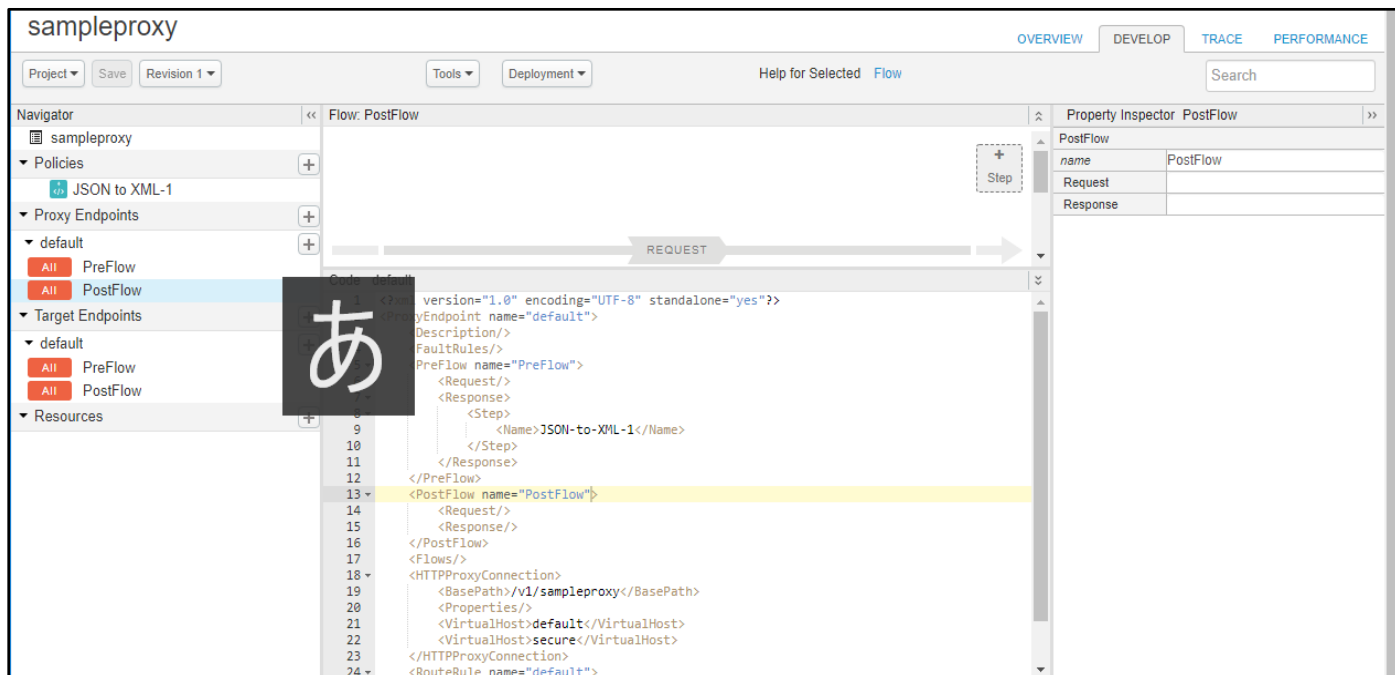
表示中の API Proxy の Base Path と、紐づいている Target Endpoints の名前が表示されます。

- Target Endpoints

バックエンドサービスとして指定した URL と、紐づいている Proxy Endpoints の名前が表示されます。

3.2.4.2. DEVELOP

3.2.1.3 New API Proxy で作成した API Proxy に対する Policy の設定の他、API Proxy のリビジョン管理、環境へのデプロイをおこなうことができます。



- Project

Save as New Revision	表示中の API Proxy を新しいリビジョンとして保存します。
Save as New API Proxy...	表示中の API Proxy に新しい名前を付けて保存します。
Undo All	最後に設定を保存した時の状態に画面を戻します。
Delete API Proxy...	表示中の API Proxy を削除します。
Delete Current Revision...	表示中のリビジョンを削除します。
Download Revision	表示中のリビジョンの API Proxy 情報を zip 形式でダウンロードします。
Upload a New Revision...	ダウンロードした API Proxy 情報を読み込み、新しいリビジョンとして保存します。
API Proxy History	表示中の API Proxy の変更履歴を表示します。

- Tools

Tools の「Custom Analytics Collection」をクリックすると、Solution Builder 画面（下図の通り）が表示されます。Solution Builder 画面では、Extract Statistics Request Policy および Collect Statistics Request Policy の設定をおこなうことができます。

Solution Builder: Custom Analytics Collection

1 Analyze HTTP Transaction and Define Collectors

Extract Data from HTTP Transaction		Save in Collector Variable		
Location Type	Location Source	Name	Data Type	Action
1	Select... <input type="button" value="v"/>	<input type="text"/>	String <input type="button" value="v"/>	

2 Select Flow

Proxy Endpoint default, Flow PreFlow

Solution Results

New collector variables are available in the Report editor as drilldown dimensions after this API proxy has been saved and deployed.

- Analyze HTTP Transaction and Define Collectors

Extract Data from HTTP Transaction	Extract Statistics Request Policy の設定ができます。 ※詳細は 4 Policy 一覧を参照してください。
Save in Collector Variable	Collect Statistics Request Policy の設定ができます。 ※詳細は 4 Policy 一覧を参照してください。

- Select Flow

Policy を設置する Flow を選択します。

- Deployment

prod	表示中の API Proxy を prod 環境にデプロイします。
test	表示中の API Proxy を test 環境にデプロイします。

※名前の左側の●が灰色の場合はアンデプロイ状態、緑色の場合はデプロイ状態を表します。

- Node.js Logs

API Proxy に Node.js が設定されている場合に限り、「Node.js Logs」ボタンが表示されます。ボタンを押下すると、Node.js Logs 画面が表示されます。詳細は 3.2.4.4 Node.js Logs を参照してください。

- Navigator ペイン

<API Proxy 名>	表示中の API Proxy の名前が表示されます。名前をクリックすると API Proxy の設定ペイン (Revision Metadata、Code、Property Inspector) が表示されます。
Policies	Policy の追加と削除が可能です。名前をクリックすると、Policy の設定ペイン (Policy、Code、Property Inspector) が表示されます。
Proxy Endpoints	Proxy Endpoint の追加と削除が可能です。名前をクリックすると Proxy Endpoint の設定ペイン (Proxy Endpoint、Flow、Code、Property Inspector) が表示されます。
Target Endpoints	Target Endpoints の追加と削除が可能です。名前をクリックすると、Target Endpoint の設定ペイン (Target Endpoint、Flow、Code、Property Inspector) が表示されます。
Resources/Scripts	Scripts (JAR、JavaScript、Node.js、Python、WSDL、XSD、XSL Transformation) の追加と削除が可能です。名前をクリックすると、Script の設定ペイン (Script、Code、Property Inspector) が表示されます。

- Revision Metadata ペイン

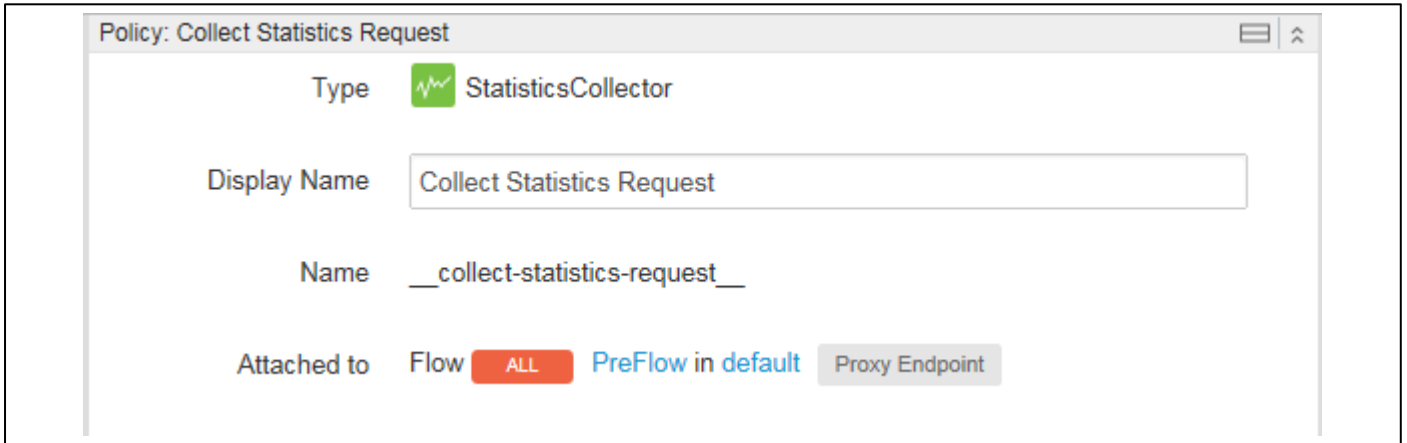
API Proxy の概要を表示します。Display Name と Description の編集が可能です。

Revision Metadata: weather ☰ ⬆

Display Name	<input style="width: 80%;" type="text" value="weather"/>
Description	<input style="width: 80%;" type="text"/>
Created At	Dec 3rd, 2015
Created By	
Last Modified At	Dec 3rd, 2015
Last Modified By	

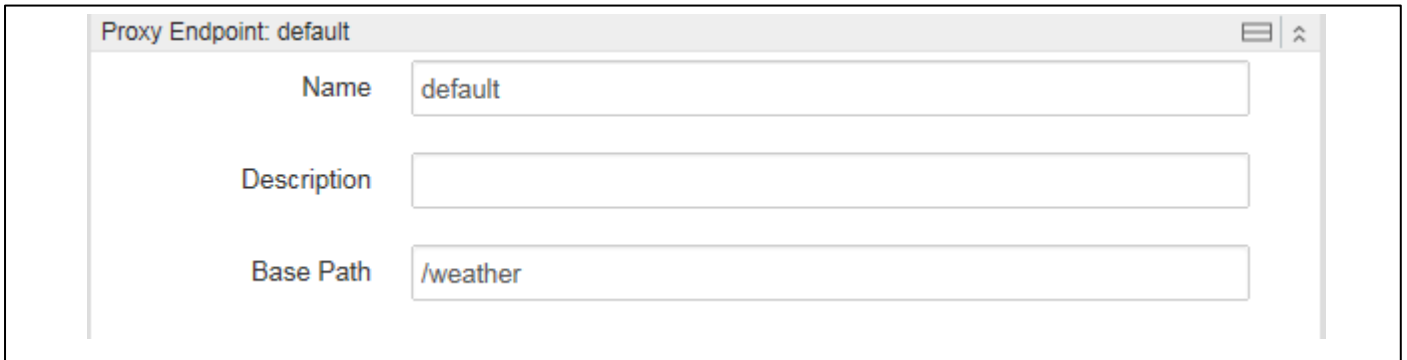
- Policy ペイン

Policy の概要を表示します。Display Name の編集が可能です。



- Proxy End point ペイン

Proxy Endpoint の概要を表示します。全ての項目が編集可能です。



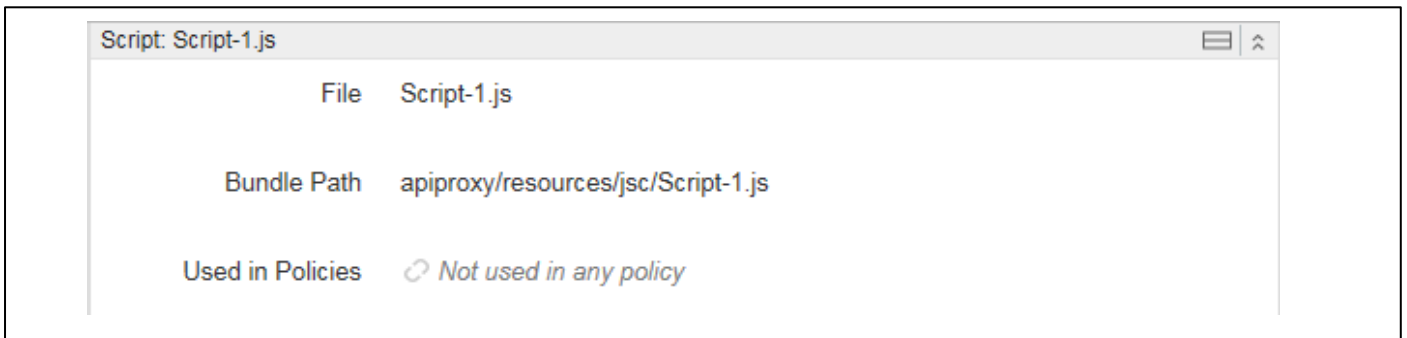
- Flow ペイン

選択中の Flow で実行される Policy を表示します。Policy の追加と削除が可能です。



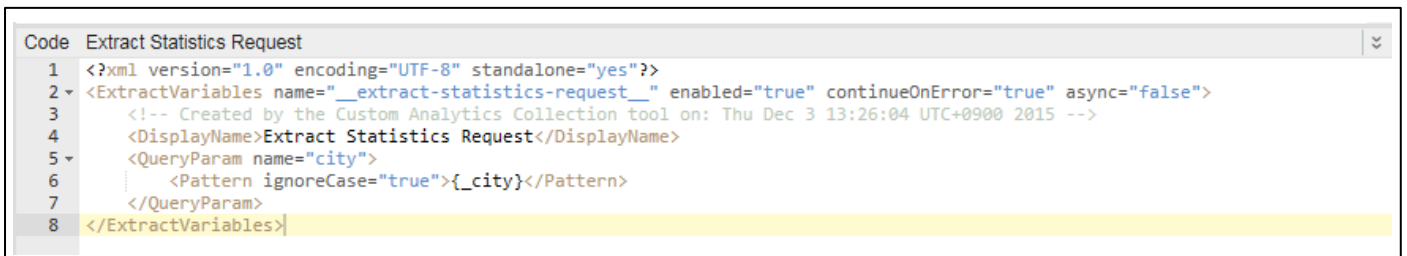
- Script ペイン

Script の概要を表示します。



- Code ペイン

選択中の項目（<API Proxy 名>、Policies、Proxy Endpoints、Target Endpoints、Flow、Scripts）の Code を表示します。Code は、編集可能です。



- Property Inspector ペイン

選択中の項目（<API Proxy 名>、Policies、Proxy Endpoints、Target Endpoints、Flow）の XML 要素を表示します。要素の値のみ編集する事が可能です。

※要素の追加・削除をおこなう場合は、Code ペインをご利用ください。

Property Inspector Extract Statistics Request	
ExtractVariables	
async	false
continueOnError	
enabled	true
name	__extract-statistics-request__
DisplayName	Extract Statistics Request
QueryParam	
name	city
Pattern	{_city}
ignoreCase	true

3.2.4.3. TRACE

TRACE 機能は、トラブルシューティングやデバッグ用に提供されているツールです。TRACE 機能を使うことで、API Proxy に送信されるリクエストおよびレスポンスを最大 10 分間までキャプチャし、トランザクションとして可視化することができます。

The screenshot displays the Apigee TRACE interface for a service named 'weather'. At the top, there are tabs for OVERVIEW, DEVELOP, TRACE, and PERFORMANCE. Below the tabs, there's a 'Deployment to Trace' section with 'Environment prod, Revision 1' and a 'Stop Trace Session' button with a remaining time of 04:55. A 'Download Trace Session' button and 'Node.js Logs' link are also present. The main area is divided into several sections: 'Filters' on the left, 'Transactions' in the center, 'Send Requests' on the right, and 'Phase Details' at the bottom. The 'Transactions' table shows one transaction with a status of 200, method GET, and a URL of /weather/forecast/webservice/json/v1?city=200020&test=100, with an elapsed time of 614 ms. The 'Phase Details' section shows the request and response headers for the selected transaction. The request headers include Accept, Accept-Encoding, Host, User-Agent, X-Apgee application, X-Apgee environment, X-Apgee organization, X-Apgee proxy, X-Apgee revision, X-Apgee vhost, X-Apgee-Trace-Id, X-Forwarded-For, X-Forwarded-Port, and X-Forwarded-Proto. The response headers include Connection, Content-Type, Date, Keep-Alive, Server, Set-Cookie, Transfer-Encoding, X-Content-Type-Options, and X-Frame-Options. The response status is 200 OK.

- Deployment to Trace

TRACE を実施する Environment とリビジョンを選択します。

- Start Trace Session / Stop Trace Session ボタン

「Start Trace Session」ボタンを押下すると TRACE が開始、「Stop Trace Session」ボタンを押下すると、TRACE が終了します。また、TRACE 開始後、10 分を経過すると自動的に TRACE が終了します。

- Download Trace Session

「Download Trace Session」ボタンを押下すると、TRACE の実施結果を xml 形式でダウンロードすることができます。

- Node.js Logs

Node.js Logs 画面を表示します。詳細は 3.2.4.4 Node.js Logs を参照してください。

- Filters ペイン

HTTP Header と Query Parameter を指定（複数指定した場合は AND 条件）することで、変数と値に一致するトランザクションのみキャプチャします。

Name	Value
HTTP Header	
<input type="text"/>	<input type="text"/>
Query Parameter	
<input type="text" value="city"/>	<input type="text" value="200020"/>
<input type="text" value="test"/>	<input type="text" value="100"/>
<input type="text"/>	<input type="text"/>

- Transactions ペイン

キャプチャしたトランザクションの一覧が表示されます。トランザクションをクリックすると、Transaction Map ペイン、Phase Details ペインに詳細情報を表示します。

Status	Method	URI	Elapsed
1 200	GET	/weather/forecast/... test=100&city=200020	614 ms

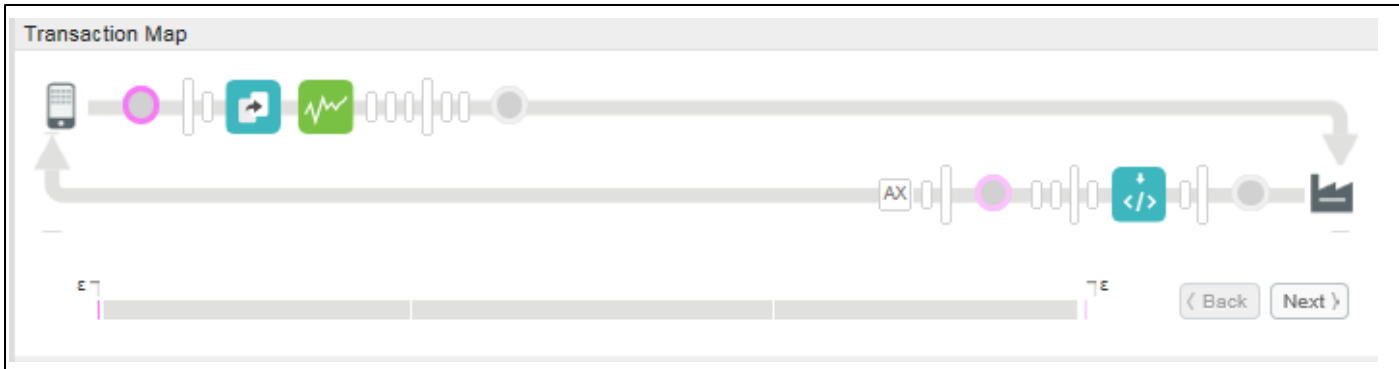
- Send Requests ペイン

指定した URL に GET リクエストを送信できます。右側の Status はリターンコードが表示されます。

Method	URL	Status
GET	<input type="text" value="http://ope_t.example.com/weather/forecast/webservice/json/v1?city=200020&test=100"/>	

● Transaction Map ペイン

トランザクションをアイコンで可視化した図が表示されます。図中のアイコンをクリックすることで、そのアイコンの詳細情報を Phase Details ペインに表示します。



■ Transaction Map ペインのアイコン一覧

	リクエストを送信するクライアントアプリです。
	エンドポイントの切り替わりで表示されるアイコンです。表示されるタイミングは下記の通りです。 <ul style="list-style-type: none"> ▶ クライアントからのリクエストを受信したタイミング ▶ バックエンドサービスにリクエストを送信したタイミング ▶ バックエンドサービスからの Respons を受信したタイミング ▶ クライアントに Respons を送信したタイミング
	小さいバーと大きいバーの 2 種類があり、FlowInfo と呼びます。小さいバーは Flow の中で Properties に変化があったことを示し、大きいバーは API Proxy の各 Flow で処理が開始されるタイミングを示します。
	Analytics のアクションが発生したことを示します。
	Conditional Flow の条件と一致した場合に表示されます。
	Conditional Flow の条件と一致しなかった場合に表示されます。
	各 Flow で設定された Policy のアイコンです。
	バックエンドサービスが、Node.js の場合に表示されます。
	リクエストを受信し、レスポンスを返すバックエンドサービスです。
	Transaction Map でクリックしたアイコンの処理時間（ミリ秒）をピンク色のタイムラインで示します。
	Policy が無効になっている場合、該当する Policy のアイコン上に表示されま
	Policy でエラーが発生した場合もしくは、RaiseFault Policy が実行された場
	Policy が処理条件に一致せずスキップされた場合、該当する Policy のアイコン

- Phase Details ペイン

Transaction Map ペインで選択中のアイコンの詳細情報が表示されます。

Request Received from Client GET /weather/forecast/webservice/json/v1?test=100&city=200020		Response Sent to Client 200 OK	
Request Headers		Response Headers	
Accept	*/*	Connection	keep-alive
Accept-Encoding	gzip,deflate	Content-Type	text/xml;charset=UTF-8
Host	yuuki_taki-prod.apigee.net	Date	Fri, 04 Dec 2015 01:32:
User-Agent	NING/1.0	Keep-Alive	timeout=3
X-Apigee.application	weather	Server	nginx
X-Apigee.environment	prod	Set-Cookie	Idsuid=CoJlJFZg7ShelC expires=Thu,03-Mar-16 GMT; path=/
X-Apigee.organization	yuuki_taki	Transfer-Encoding	chunked
X-Apigee.proxy	default	X-Content-Type-Options	nosniff
X-Apigee.revision	1	X-Frame-Options	DENY
X-Apigee.vhost	default		
X-Apigee-Trace-Id	982bb6ff-91c8-4a61-913f-046124dd7329		
X-Forwarded-For	54.64.120.29		
X-Forwarded-Port	80		
X-Forwarded-Proto	http		

- View Options ペイン

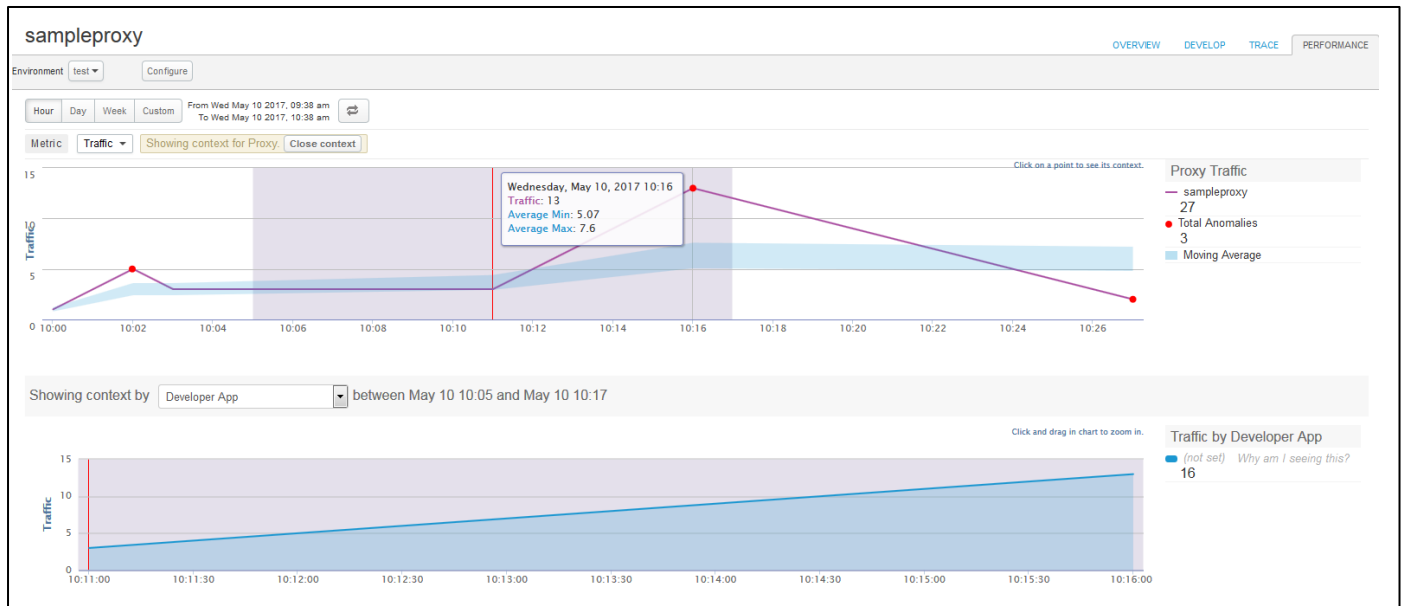
Show Disabled Policies	チェックを入れると、無効になっている Policy を Transaction Map ペインに表示します。
Show Skipped Phases	チェックを入れると、処理がスキップされた Policy を Transaction Map ペインに表示します。
Show All FlowInfos	チェックを入れると、FlowInfo を Transaction Map ペインに表示します。
Automatically Compare Selected Phase	チェックを入れると、処理前後の比較を Phase Details ペインに表示します。
Show Variables	チェックを入れると、変数を Phase Details ペインに表示します。
Show Properties	チェックを入れると、Properties を Phase Details ペインに表示します。

3.2.4.4. Node.js Logs

下図の通り、API Proxy に設定された Node.js のログ一覧が表示されます。

Node.js Logs			
Search	All ▾	Stop Auto Refresh	Last refresh at 16:57:42
1—9 of 9 < < > >			
Time	Level	Server	Message
Sep 12, 2016 9:00:28 PM	stdout	svr.7	Node HTTP server is listening
Sep 12, 2016 9:00:28 PM	stdout	svr.7	node.js application starting...
Sep 12, 2016 9:00:28 PM	stderr	svr.7	*** Starting script
Sep 12, 2016 9:00:05 PM	stdout	svr.130	Node HTTP server is listening
Sep 12, 2016 9:00:05 PM	stdout	svr.130	node.js application starting...
Sep 12, 2016 9:00:05 PM	stderr	svr.130	*** Starting script
Sep 12, 2016 8:59:54 PM	stdout	svr.314	Node HTTP server is listening
Sep 12, 2016 8:59:54 PM	stdout	svr.314	node.js application starting...
Sep 12, 2016 8:59:54 PM	stderr	svr.314	*** Starting script

API Proxy に対するリクエストの解析結果が表示されます。上のグラフには、API Proxy および Configure で設定された URI Pattern の Metric の値を表示します。表示された折れ線グラフをクリックすると、下のグラフにドリルダウン結果が表示されます。



- Metric

Traffic	リクエスト数を表示します。
Errors	エラーになったリクエスト数を表示します。
Average Request Size	リクエストの平均サイズを表示します。
Average Response Size	レスポンスの平均サイズを表示します。
Average Response Time	応答時間の平均を表示します。

- Showing context by

Access Token	OAuth のアクセストークン（OAuth Policy を使用した際）を表示します。
Developer App	リクエスト元のアプリの名前を表示します。
Developer ID	リクエスト元のアプリ開発者を表示します。
Flow Resource	リクエスト対象のバックエンドサービスのリソースを表示します。
Proxy Client IP	API Proxy の IP アドレスを表示します。
Proxy Path Suffix	API Proxy URI のベースパスより後ろの文字列を表示します。
Request Verb	リクエストメソッドを表示します。
Response Status Code	レスポンスコードを表示します。
Target Response Code	バックエンドサービスのレスポンスコードを表示します。
Target URL	バックエンドサービスの URL を表示します。
User Agent	User-Agent ヘッダーの値を表示します。
X Forwarded For	X Forwarded For ヘッダーの値を表示します。
ax_isp	ax_isp を表示します。
API Product	リクエストを受け取った API Proxy をメンバにもつ Product を表示します。
Business Unit ID	事業 ID を表示します。
Cache Key	Cache Key を表示します。
Cache Name	Cache Name を表示します。
Cache Source	Cache Source を表示します。
Channel ID	チャンネル ID を表示します。
City	リクエスト元の都市名を表示します。
Client Application Name	クライアントアプリの名前を表示します。
Client Host	クライアントアプリをホスティングしているコンピュータの名前を表示します。
Client ID	API Key を表示します。 ※Developers 画面で発行した API Key です。
Client IP Address	リクエスト元の IP アドレスを表示します。
Client Organization Name	クライアントの組織名を表示します。

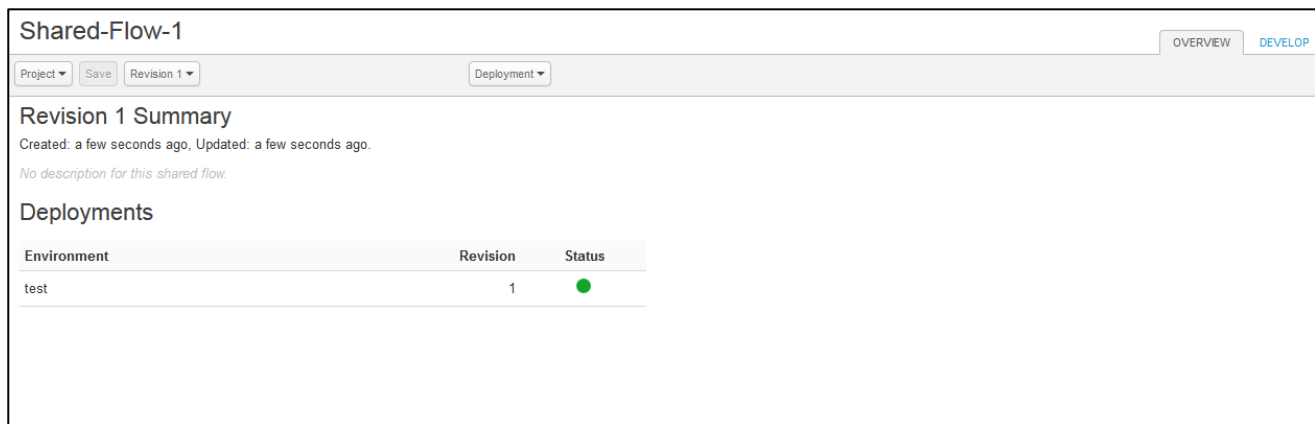
Client Request ID	クライアントのリクエスト ID を表示します。
Continent	リクエスト元の大陸名を表示します。
Country	リクエスト元の国名を表示します。
Day of week	リクエストした曜日を表示します。
Developer Email	開発者のメールアドレスを表示します。 ※Developers 画面で登録した Developer のメールアドレスです。
Device Category	デバイスの種類 (Personal computer 等) を表示します。 ※User-Agent に含まれる情報です。
Device ID	デバイス ID を表示します。
Environment	API Proxy がデプロイされている環境の名前 (test、prod 等) を表示します。
Flow Name on Error	エラーが発生したフロー名を表示します。
Flow State on Error	エラーが発生したフローの状態を表示します。
Gateway Flow ID	ゲートウェイフローの ID を表示します。(VM-8D5-S-0011_BTnU7OVE_RouterProxy-8-1047_6 等)
Gateway Source	ゲートウェイ (router、message_processor 等) を表示します。
Geographical Region	地理的地域を表示します。
Market ID	マーケット ID を表示します。
Month	リクエストした月を表示します。
Organization	組織名を表示します。
OS Family	OS の種類 (Windows) を表示します。 ※User-Agent に含まれている情報です。
OS Version	接続元の OS バージョンを表示します。 例) 6.1 (Windows7 のコードバージョン) ※User-Agent に含まれている情報です。
Partner ID	パートナー ID を表示します。
Policy Name on Error	エラーが発生した Policy の名前を表示します。
Proxy	Proxy Endpoint 名を表示します。
Proxy Base Path	API Proxy の Base Path を表示します。
Proxy Revision	API Proxy の Revision (1, 2, 3 . . . 等) を表示します。
Referred Client IP	参照されたクライアント IP を表示します。
Region	地域を表示します。
Request Path	リクエスト URI のパス部分を表示します。
Request URI	リクエスト URI を表示します。
Session ID	セッション ID を表示します。
Target	Target Endpoint 名を表示します。
Target Base Path	バックエンドサービスの Base Path を表示します。

Target Host	バックエンドサービスのドメイン名を表示します。
Target IP Address	バックエンドサービスの IP アドレスを表示します。
Time of Day	リクエスト時刻を表示します。
Time Zone	リクエスト元のタイムゾーンを表示します。
Traffic Referral ID	参照元の ID を表示します。
User Agent Family	エージェントのカテゴリ (Chrome、IE 等) を表示します。
User Agent Type	エージェントの型 (Browser 等) を表示します。
User Agent Version	エージェントのバージョン (Chrome→43.0、IE→11.0 等) を表示します。
Virtual Host	Virtual Host 名を表示します。
Week of Month	リクエストした週を表示します。

3.2.5. Shared Flows 画面

3.2.5.1. OVERVIEW

3.2.2.1 New Shared Flow で作成した Shared Flow の概要他、Shared Flow のリビジョン管理、環境へのデプロイをおこなうことができます。



● Project

Save as New Revision	表示中の Shared Flow を新しいリビジョンとして保存します。
Save as New Shared Flow...	表示中の Shared Flow に新しい名前を付けて保存します。
Undo All	最後に保存した時点からの変更を取り消します。
Delete Shared Flow...	表示中の Shared Flow を削除します。
Delete Current Revision...	表示中のリビジョンを削除します。
Download Revision	表示中のリビジョンの Shared Flow 情報を zip 形式でダウンロードします。
Upload a New Revision...	ダウンロードした Shared Flow 情報を読み込み、新しいリビジョンとして保存します。

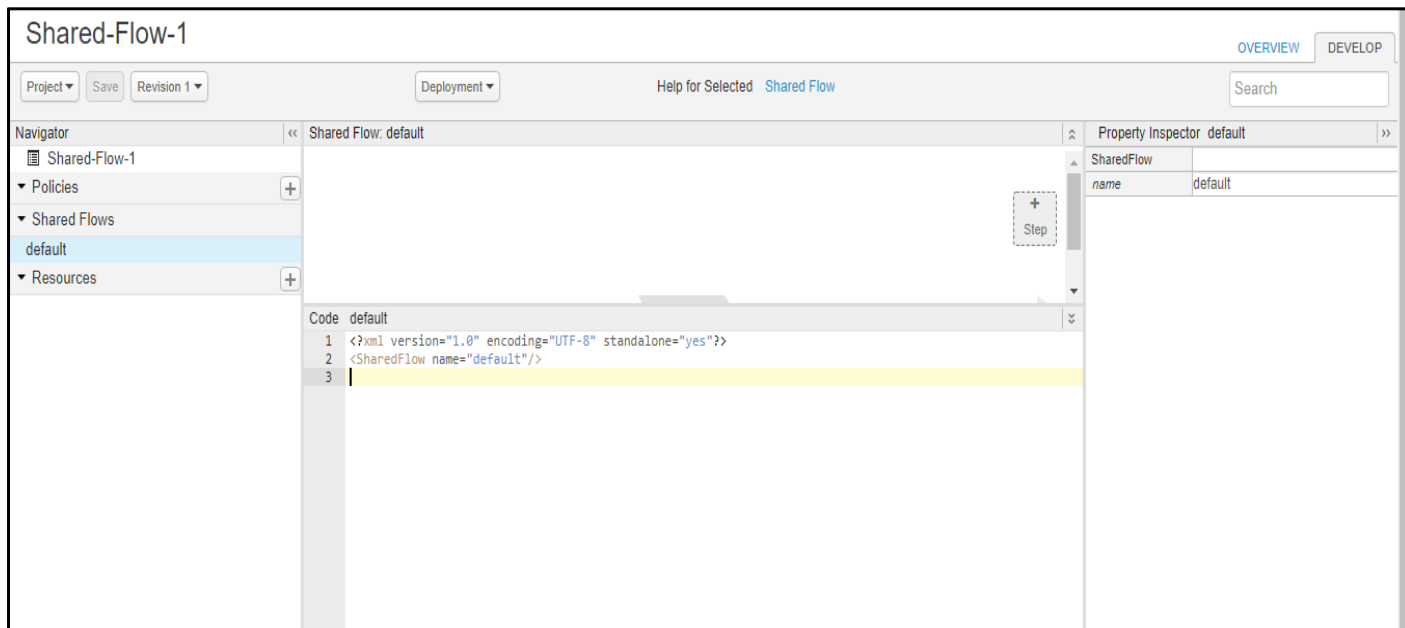
● Deployment

prod	表示中の Shared Flow を prod 環境にデプロイします。
test	表示中の Shared Flow を test 環境にデプロイします。

※名前の左側の●が灰色の場合はアンデプロイ状態、緑色の場合はデプロイ状態を表します。

3.2.5.2. DEVELOP

3.2.2.1 New Shared Flow で作成した Shared Flow に対する Policy の設定の他、Shared Flow のリビジョン管理、環境へのデプロイをおこなうことができます。



- Project

Save as New Revision	表示中の Shared Flow を新しいリビジョンとして保存します。
Save as New Shared Flow...	表示中の Shared Flow に新しい名前を付けて保存します。
Undo All	最後に保存した時点からの変更を取り消します。
Delete Shared Flow...	表示中の Shared Flow を削除します。
Delete Current Revision...	表示中のリビジョンを削除します。
Download Revision	表示中のリビジョンの Shared Flow 情報を zip 形式でダウンロードします。
Upload a New Revision...	ダウンロードした Shared Flow 情報を読み込み、新しいリビジョンとして保存します。

- Deployment

prod	表示中の Shared Flow を prod 環境にデプロイします。
test	表示中の Shared Flow を test 環境にデプロイします。

※名前の左側の●が灰色の場合はアンデプロイ状態、緑色の場合はデプロイ状態を表します。

- Navigator ペイン

<Shared Flow 名>	表示中の Shared Flow の名前が表示されます。名前をクリックすると Shared Flow の設定ペイン (Revision Metadata、Code、Property Inspector)が表示されます。
-----------------	---

Policies	Policy の追加と削除が可能です。名前をクリックすると、Policy の設定ペイン (Policy、Code、Property Inspector) が表示されます。
Shared Flows	Shared Flow の追加と削除が可能です。名前をクリックすると、Shared Flow の設定ペイン (Shared Flow、Code、Property Inspector) が表示されます。
Resources/Scripts	Scripts (JAR、JavaScript、Node、Python、WSDL、XSD、XSL Transformation) の追加と削除が可能です。名前をクリックすると、Script の設定ペイン (Script、Code、Property Inspector) が表示されます。

- Revision Metadata ペイン

Shared Flows の概要を表示します。Display Name と Description の編集が可能です。



Revision Metadata: Shared-Flow_fj3381gw

Display Name: Shared-Flow_fj3381gw

Description:

Created At: Apr 28th, 2017

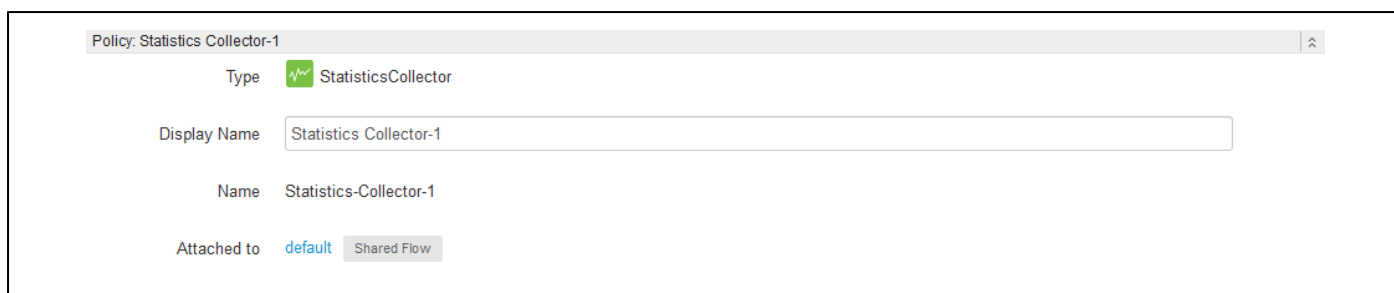
Created By: fst-webapi-apigee-prj@ml.jp.fujitsu.com

Last Modified At: Apr 28th, 2017

Last Modified By: fst-webapi-apigee-prj@ml.jp.fujitsu.com

- Policy ペイン

Policy の概要を表示します。Display Name の編集が可能です。



Policy: Statistics Collector-1

Type: StatisticsCollector

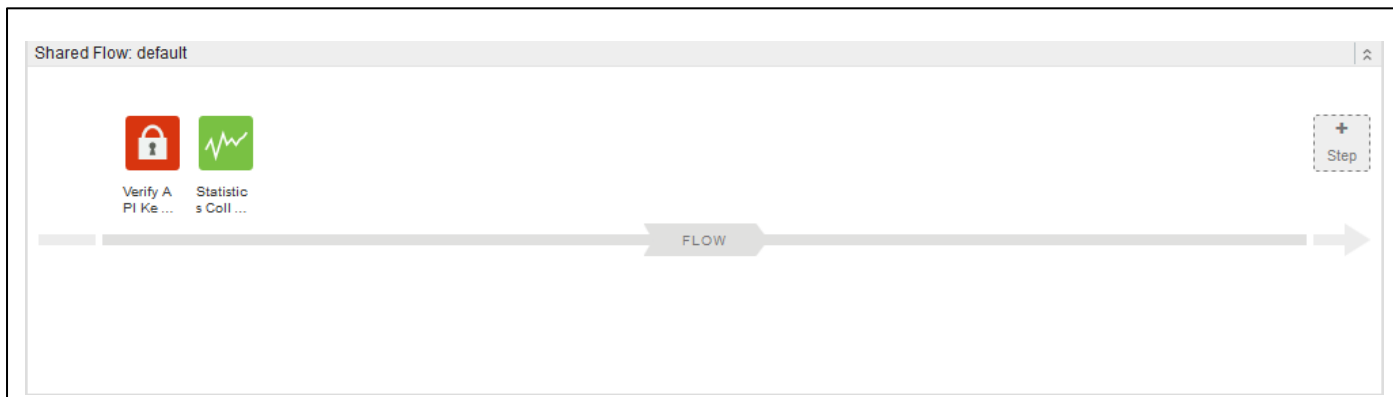
Display Name: Statistics Collector-1

Name: Statistics-Collector-1

Attached to: default Shared Flow

- Shared Flow ペイン

選択中の Shared Flow で実行される Policy を表示します。Policy の追加と削除が可能です。



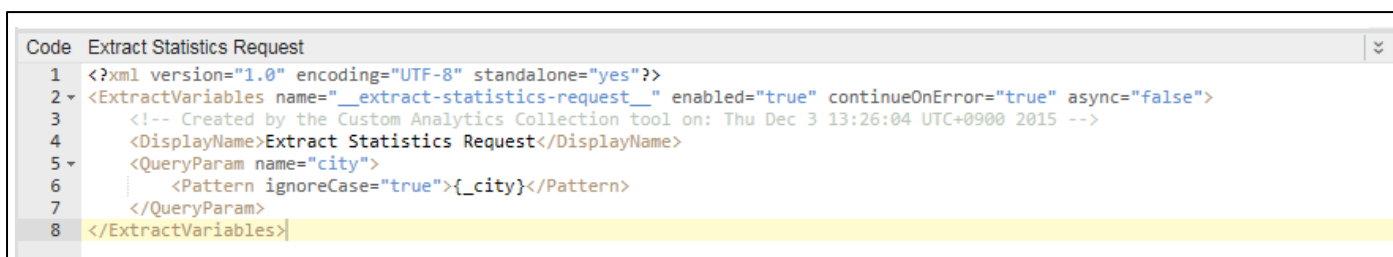
- Script ペイン

Script の概要を表示します。



- Code ペイン

選択中の項目（<Shared Flow 名>、Policies、Shared Flows、Scripts）の Code を表示します。Code は、編集可能です。



- Property Inspector ペイン

選択中の項目（<Shared Flow 名>、Policies、Shared Flows）の XML 要素を表示します。要素の値のみ編集する事が可能です。

※要素の追加・削除をおこなう場合は、Code ペインをご利用ください。

Property Inspector Extract Statistics Request >>	
ExtractVariables	
<i>async</i>	false
<i>continueOnError</i>	
<i>enabled</i>	true
<i>name</i>	__extract-statistics-request__
DisplayName	Extract Statistics Request
QueryParam	
<i>name</i>	city
Pattern	{_city}
<i>ignoreCase</i>	true

3.3. Publish

3.3.1. Products

3.3.1.1. List

Product の一覧画面です。

Product	Keys	Created	Modified	Actions
Premium Weather API	1	Aug 17, 2015 1:09:44 PM	Aug 17, 2015 1:09:44 PM	History Delete Roles

- +Product ボタン

「+Product」ボタンを押下すると、New Product 画面を表示します。詳細は、3.3.1.5 New Product を参照してください。

- History ボタン

「History」ボタンを押下すると、Product History 画面を表示します。詳細は、3.3.1.3 Product history を参照してください。

- Roles ボタン

「Roles」ボタンを押下すると、Role 一覧画面を表示します。詳細は、3.3.1.4 Roles を参照してください。

- Product 欄

Product 欄のリンクをクリックすると、Product の詳細画面が表示されます。

sampleProduct

Products give developers access to your APIs. [Learn more](#)

[Edit](#) [Delete](#)

Product Details

Name sampleProduct

Display Name sampleProduct

Description

Environment prod test

Access Internal only

Key Approval Type Automatic

Quota

Allowed OAuth Scopes

Resources

Paths

API Proxies

Istio Services

Custom

Attributes	Name	Value
------------	------	-------

Developer Apps

Search 1-1 of 1 [◀](#) [▶](#)

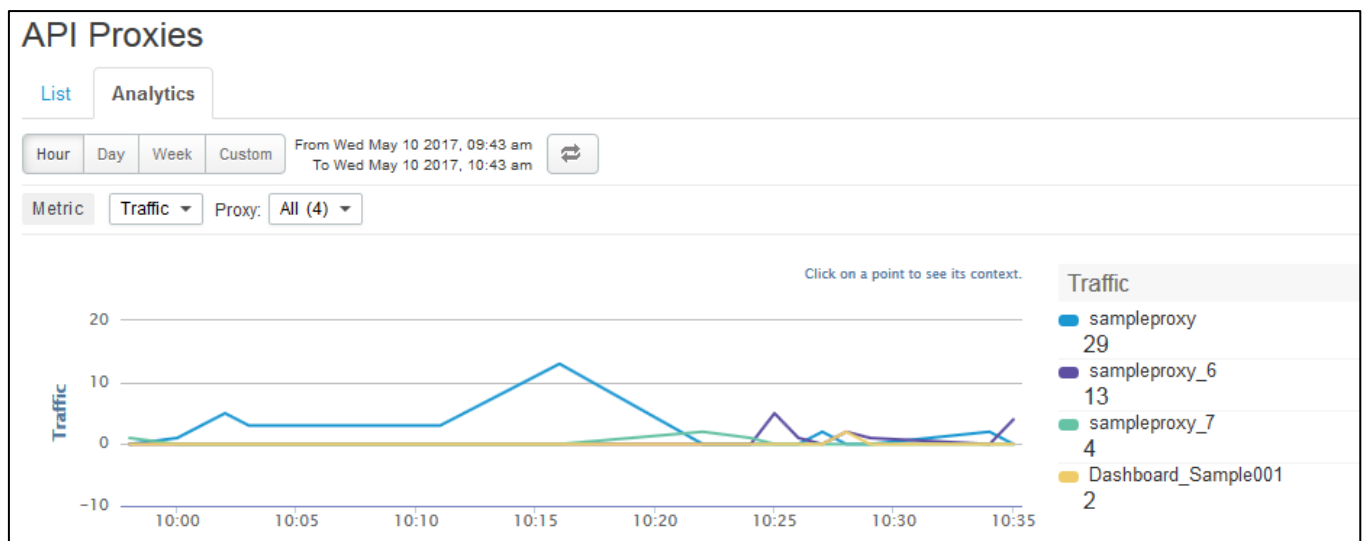
App	Developer	App Family	Key	Registered ▼
-----	-----------	------------	-----	--------------

- Edit ボタン

「Edit」ボタンを押下することで、Product Details の Name と Developer Apps を除く項目の修正が可能な状態になります。

3.3.1.2. Analytics

全 Products のリクエスト数が表示されます。



- Metric

Traffic	リクエスト数を表示します。
---------	---------------

3.3.1.3. Product history

Product の操作履歴が表示されます。

Product history: PremiumWeatherAPI				
Date Range: Year ▾				
Operation	User	Start Time ▾	Code	Request Body
CREATE	noreply_admin@a.com	8/17/15 1:09 PM	201	

3.3.1.4. Roles

Product の Role 一覧が表示されます。Role の追加と削除が可能です。

Premium Weather API

Role	Operations	Actions
all ok	<input type="checkbox"/> View <input type="checkbox"/> Edit <input type="checkbox"/> Delete	<input type="button" value="× Delete"/>

- Role 欄

3.5.2.2 New Custom Role で登録した Custom Role が選択可能です。

3.3.1.5. New Product

Product の作成画面です。

New Product

Products give developers access to your APIs. [Learn more](#)

Product Details

Name

Display Name

Description

Environment prod test

Access Internal only — Visible only to developers at sandbox during app registration
 Private — Visible only to external developers with explicit permission during app registration
 Public — Visible only to any registered developer during app registration

Key Approval Type Automatic Manual
[Learn more](#)

Quota requests every

Allowed OAuth Scopes
Comma-separated scope names. [Learn more](#)

Resources

API Proxy Revision Resource Path

Paths	Resource Path	Actions
		<input type="button" value="+ Custom Resource"/>

API Proxies	API Proxy	Actions
		<input type="button" value="+ API Proxy"/>

Istio Services	Service Name	Actions
		<input type="button" value="+ Istio Service"/>

Custom

Attributes	Name	Value	Actions
			<input type="button" value="+ Custom Attribute"/>

- Access

3つのオプション（Internal only、Private、Public）に機能上の違いはありません。

- Key Approval Type

Product に登録されたアプリの API Key の認証方式を選択します。

Automatic	アプリ作成時、API Key が Approved の状態で生成されます。
Manual	アプリ作成時、API Key が Pending の状態で生成されます。

- Quota

Product に登録されたリソースへのトラフィック量を、単位時間あたりのリクエスト数で制限します。

- Allowed OAuth Scopes

OAuth 機能でスコープの許可が必要な場合、Allowed OAuth Scopes に許可するスコープを設定します。
※複数のスコープを設定する場合は、カンマ区切りで入力してください。

- Resources

アクセスを許可する API Proxy とリビジョン、リソースパスを設定します。

- Import Resource ボタン

「Import Resource」ボタンは、簡易設定機能として用意されており、ボタンを押下することでコンボボックスに指定した値を、Paths 欄および API Proxies 欄に反映させます。

- Paths 欄

アクセスを許可するリソースパスを設定します。例として、/music/venues の API コールのみアクセスを許可したい場合は「/venues」と設定します。API コールが「/music/venues?name=paramount」の場合は許可され、「/music/artists?name=Fujitsu」は拒否されます。
※リソースパスは、ワイルドカードとして「/*」、「/**」を指定することができます。「/*」はベースパスから1つ下の階層に許可を与え、「/**」はベースパスのサブ URI 全てに対して許可を与えます。

- API Proxies 欄

アクセスを許可する API Proxy を設定します。API Proxy の詳細は、3.2 APIs を参照してください。

- Istio Services 欄

本サービスでは、Istio 機能をサポートしていません。

3.3.2. Developers

3.3.2.1. List

Developers の一覧画面です。Developers の削除とエクスポート（CSV 形式）が可能です。

Developer ▲	Email	Username	Apps	Keys	Actions
tarou fujitsu	test-t@example.co.jp	富士通 太郎	1	1	Delete

- Export

Developers	Developers の情報を CSV 形式でダウンロードできます。
Developers, Apps, and Products	Developers、Apps、Products の情報を CSV 形式でダウンロードできます。

- + Developer ボタン

「+ Developer」ボタンを押下すると、New Developer 画面を表示します。詳細は、3.3.2.4 New Developer を参照してください。

- Developer 欄

Developer 欄のリンクをクリックすると、Developer 詳細画面（Details タブ）が表示されます。

- Details タブ

Display Name	Keys	Message Count	Error Rate (%)	Registered ▼
Zabbix	1	788	0.00	Sep 1, 2016 10:22:36 AM

➤ History ボタン

「History」ボタンを押下すると、Product History 画面を表示します。詳細は、3.3.2.3 Developer history を参照してください。

➤ Edit ボタン

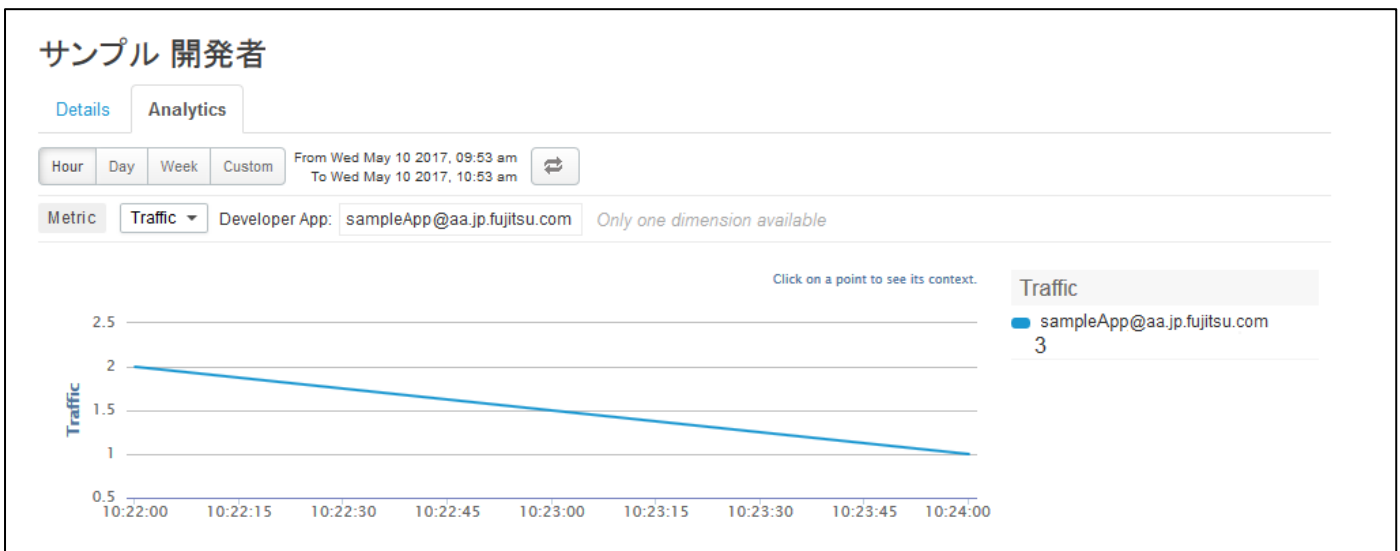
「Edit」ボタンを押下すると Developer Details の First Name、Last Name、Email、Username の 4 項目が編集可能になります。

➤ Apps

Developer に割り当てられている App が表示されます。

■ Analytics タブ

Developer のトラフィック量や応答時間、エラー件数を確認することができます。

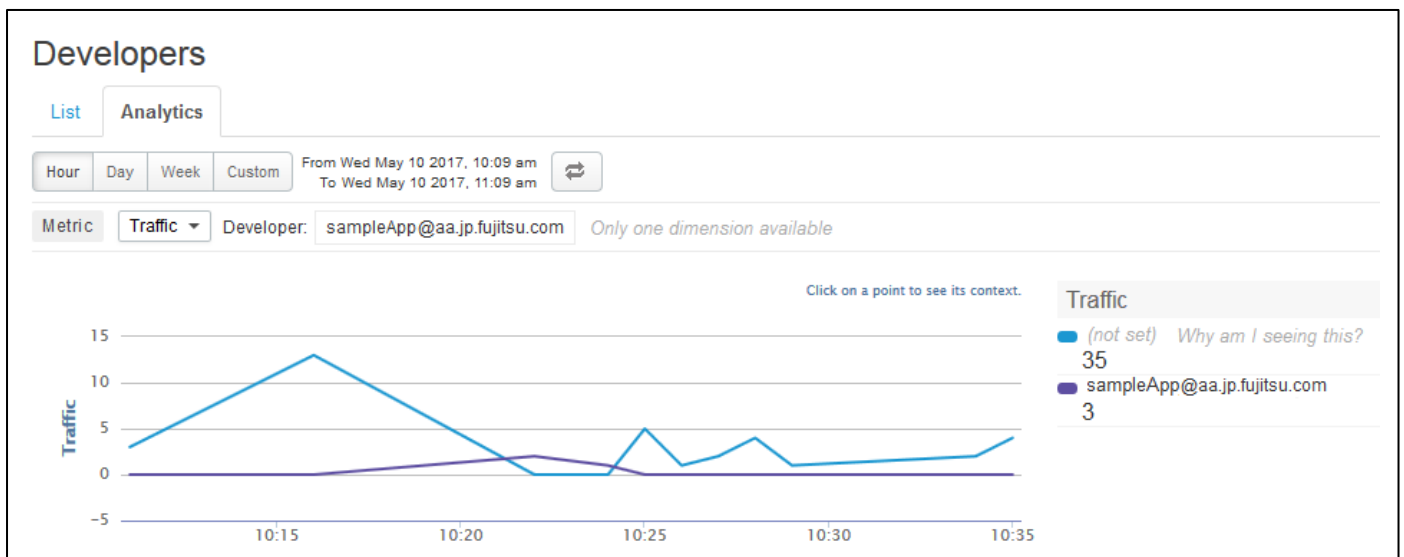


➤ Metrics

Traffic	リクエスト数を表示します。
Errors	エラーになったリクエスト数を表示します。
Average Target Response Time	API Proxy がバックエンドサービスにリクエストの全データを送り終えた時点から、バックエンドサービスからのレスポンスを全て受け取り終わるまでの時間の平均を表示します。
Average Response Time	Proxy がリクエスト情報を受け取り終わった時点から、クライアントにレスポンスを返し始める時点までの時間の平均を表示します。
Maximum Response Time	最大レスポンス時間を表示します。

3.3.2.2. Analytics

Developer 毎のリクエスト数を確認することができます。



● Metrics

Traffic	リクエスト数を表示します。
Errors	エラーになったリクエスト数を表示します。

3.3.2.3. Developer history

Developer の変更履歴が表示されます。

test-t@example.co.jp

Date Range:

Operation	User	Start Time	Code	Details
<input type="button" value="CREATE"/>	apimnq@example.com	12/15/15 1:10 PM	201	

3.3.2.4. New Developer

Developer の作成画面です。

New Developer

Developer Details

First Name	<input type="text" value="tarou"/>
Last Name	<input type="text" value="fujitsu"/>
Email	<input type="text" value="test@example.co.jp"/>
Username	<input type="text" value="富士通 太郎"/>

Custom Attributes

Name	Value	Actions
<input type="text"/>	<input type="text"/>	<input type="button" value="x Delete"/>

3.3.3. Developer Apps

3.3.3.1. List

Developer Apps の一覧画面です。

App	Developer	App Family	Company	Key	Metrics for Last 24 Hours		Registered	Actions
					Traffic	Error Rate (%)		
App test	tarou fujitsu	default		1			Dec 15, 2015 1:11:49 PM	Delete

- All ボタン

全ての App を表示します。

- Pending ボタン

「Pending」 ボタンを押下すると、API Key が Pending 状態の App を表示します。

- Revoked ボタン

「Revoked」 ボタンを押下すると、API Key が Revoked 状態の App を表示します。

- Approved ボタン

「Approved」 ボタンを押下すると、API Key が Approved 状態の App を表示します。

- + Developer App ボタン

「+ Developer App」 ボタン押下すると、New Developer App 画面を表示します。詳細は、3.3.3.3 New Developer App を参照してください。

- App 欄

App 欄のリンクをクリックすると App の詳細画面が表示されます。

サンプルアプリ ✎ Edit ✕ Delete

Developer App Details

Name sampleApp

Display Name サンプルアプリ

Registered Apr 20 2017 2:54 PM

Developer サンプル 開発者 (xxxxxxxx@jp.fujitsu.com)

App Status Approved

Callback URL

Notes

Credentials

Issued	Expiry	Consumer Key	Consumer Secret	Status
Apr 20 2017 2:54 PM 5 days ago	Never	Show	Show	Approved
		Product		
		sampleProduct		Approved

Custom Attributes

Name	Value

- Show/Hide ボタン

API Key (Consumer Key の値) と秘密鍵 (Consumer Secret の値) を表示/隠すことができます。

- Edit ボタン

「Edit」ボタンを押下すると、Display Name、Callback URL、Notes の修正、および Credentials の変更が可能になります。

Credentials の変更は、Expiration の修正、Product の追加・削除、および、API Key の Status を操作 (Revoke または Approve への変更) することができます。

Credentials

Issued	Expiry	Consumer Key	Consumer Secret	Status	Actions
Apr 18 2017 11:31 AM 21 days ago	Never	Show	wtoL6zL.CQJKNVOIG Hide	Approved	Revoke
		Product			
		sampleProduct		Approved	Revoke Remove
		+ Product			

[+ Credential](#)

- Developer 欄

Developer 欄のリンクをクリックすると、Developer の詳細画面が表示されます。詳細は、3.3.2 Developers を参照してください。

3.3.3.2. Analytics

Developer App 毎のトラフィック量や応答時間を確認することができます。



- Metrics

Traffic	リクエスト数を表示します。
Errors	エラーになったリクエスト数を表示します。
Average Target Response Time	API Proxy がバックエンドサービスにリクエストの全データを送り終えた時点から、バックエンドサービスからのレスポンスを全て受け取り終わるまでの時間の平均を表示します。
Average Response Time	Proxy がリクエスト情報を受け取り終わった時点から、クライアントにレスポンスを返し始める時点までの時間の平均を表示します。
Maximum Response Time	最大レスポンス時間を表示します。

3.3.3.3. New Developer App

Developer App の作成画面です。

New Developer App

Developer App Details

Name

Display Name

Developer

App Status

Callback URL

A callback URL is required only for 3-legged OAuth.

Notes

Credentials

Expiration	<input checked="" type="radio"/> Never <input type="radio"/> Duration <input type="radio"/> Date
Products	<input type="button" value="+ Product"/> <small>At least one product is required.</small>

Custom Attributes

Name	Value	Actions
<input type="button" value="+ Custom Attribute"/>		

- Developer

アプリの開発者を指定します。ドロップダウンのリストには、3.3.2 Developers で作成した Developer が表示されます。

- Callback URL

OAuth 認証後の戻り先となる URL を指定します。OAuth 認証を行わない場合は、設定が不要です。

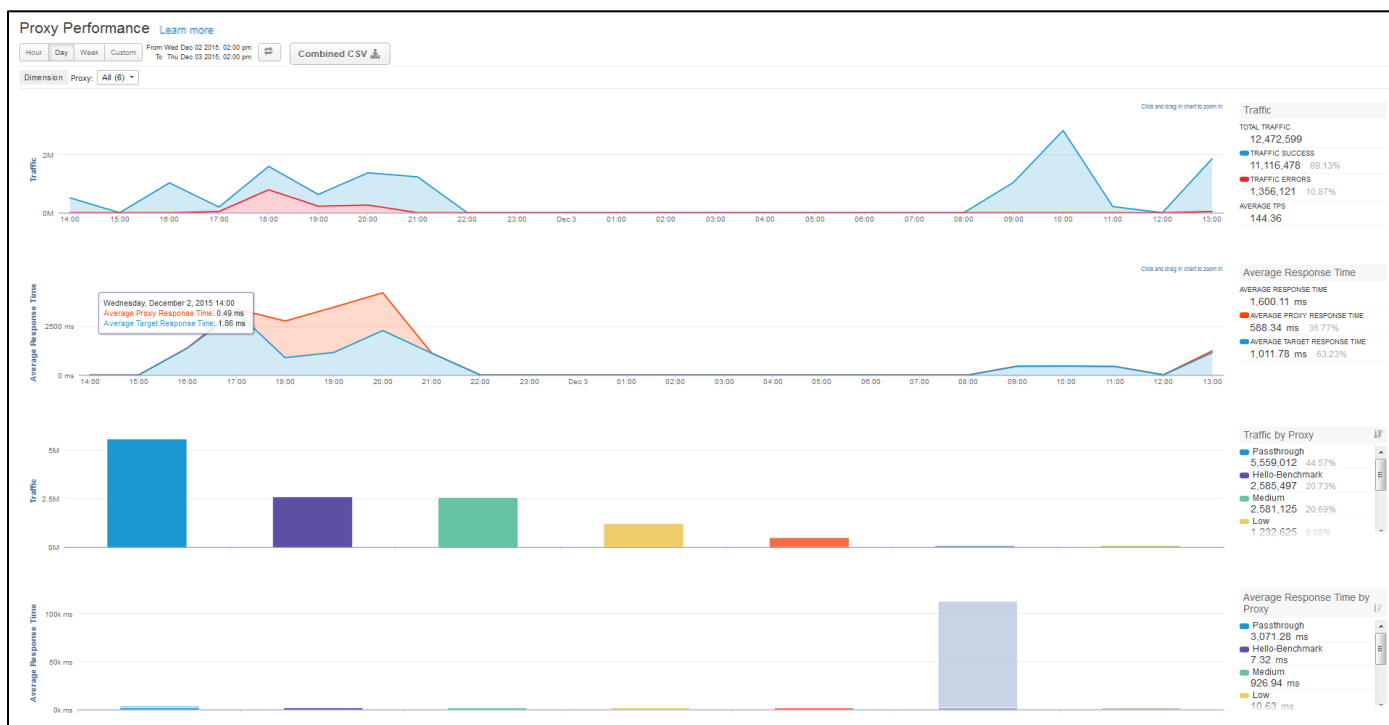
- Products

App に紐付ける Products を選択します。Products は複数追加することが可能で、「Show」ボタンを押下すると、Consumer Key および Consumer Secret の値が表示されます。

3.4. Analytics

3.4.1. Proxy Performance

API Proxy のトラフィック量と平均応答時間を確認できる画面です。



● Traffic

TOTAL TRAFFIC	リクエスト数を表示します。
TRAFFIC SUCCESS	成功したリクエスト数を表示します。
TRAFFIC ERRORS	エラーになったリクエスト数を表示します。
AVERAGE TPS	1秒あたりのリクエスト数を表示します。

● Average Response Time

AVERAGE RESPONSE TIME	API Proxy がリクエストを受け取り終わった時点から、クライアントにレスポンスを返し始める時点までの時間の平均を表示します。
AVERAGE PROXY RESPONSE TIME	API Proxy がリクエストを受け取り終わった時点から、バックエンドサービスにリクエスト情報を送り始める時点までの時間の平均を表示します。
AVERAGE TARGET RESPONSE TIME	API Proxy がバックエンドサービスからのレスポンスを受け取り終わった時点から、クライアントにレスポンスを返し始める時点までの時間の平均

- Traffic by Proxy

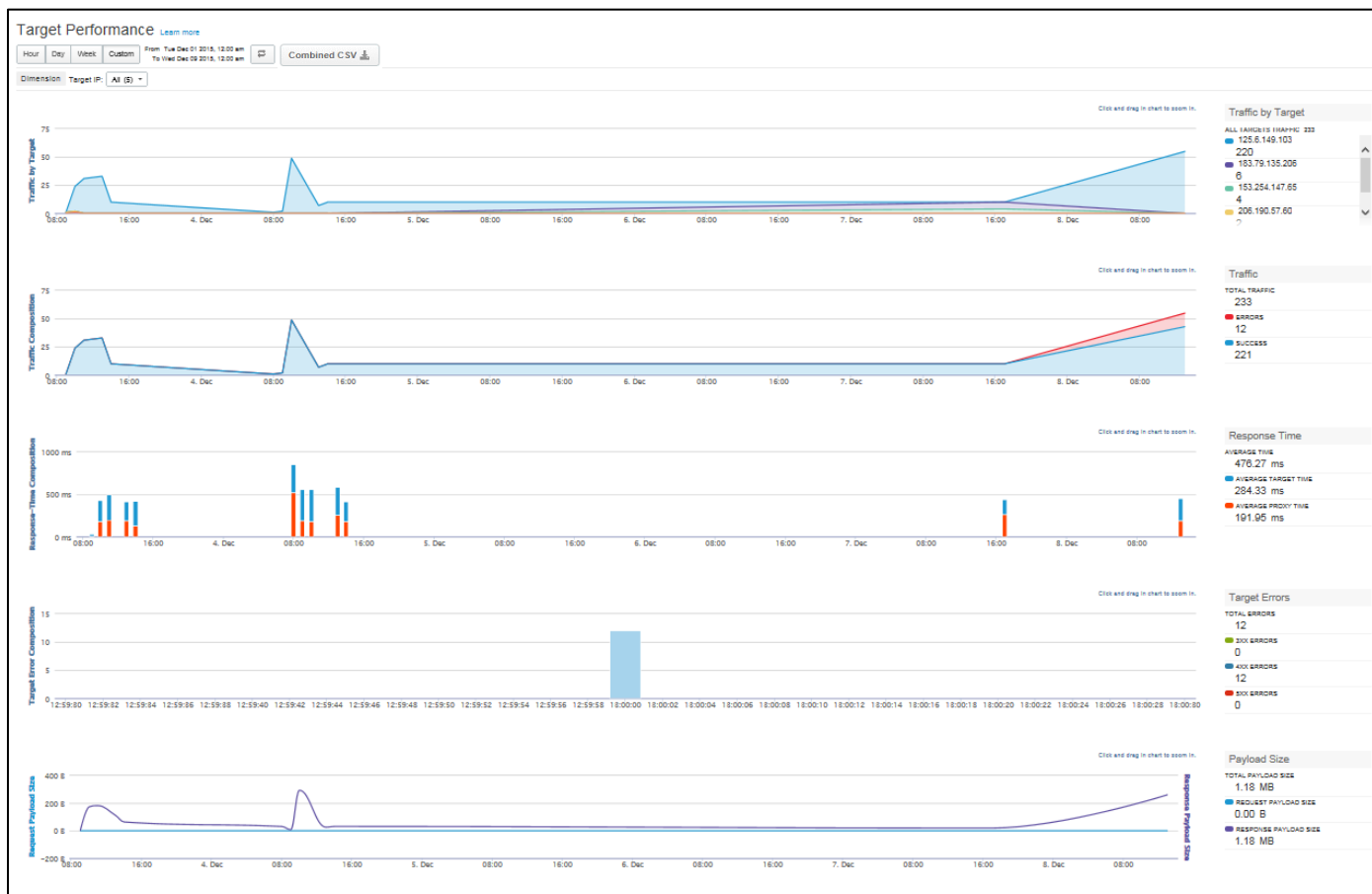
API Proxy 毎のリクエスト数を表示します。

- Average Response Time by Proxy

API Proxy 毎の「AVERAGE RESPONSE TIME」を表示します。

3.4.2. Target Performance

バックエンドサービスへのトラフィック量とリクエストの成功・失敗件数、応答時間、レスポンスの成功・失敗件数、ペイロードサイズを確認できる画面です。



- Traffic by Target

バックエンドサービス毎に、IP アドレスとそのバックエンドサービスに対するリクエスト数が表示されます。

- Traffic

TOTAL TRAFFIC	バックエンドサービスに対するリクエスト数が表示されます。
ERRORS	リクエストの失敗件数が表示されます。
SUCCESS	リクエストの成功件数が表示されます。

- Response Time

AVERAGE TIME	API Proxy がクライアントからリクエストを受けとり、クライアントレスポンスを返すまでの平均時間が表示されます。
AVERAGE TARGET TIME	API Proxy がバックエンドサービスへのリクエスト送信完了後からレスポンスを受け取るまでの平均時間が表示されます。
AVERAGE PROXY TIME	API Proxy がクライアントからリクエストを受信してから、バックエンドサービスへ送信するまでの平均時間が表示されます。

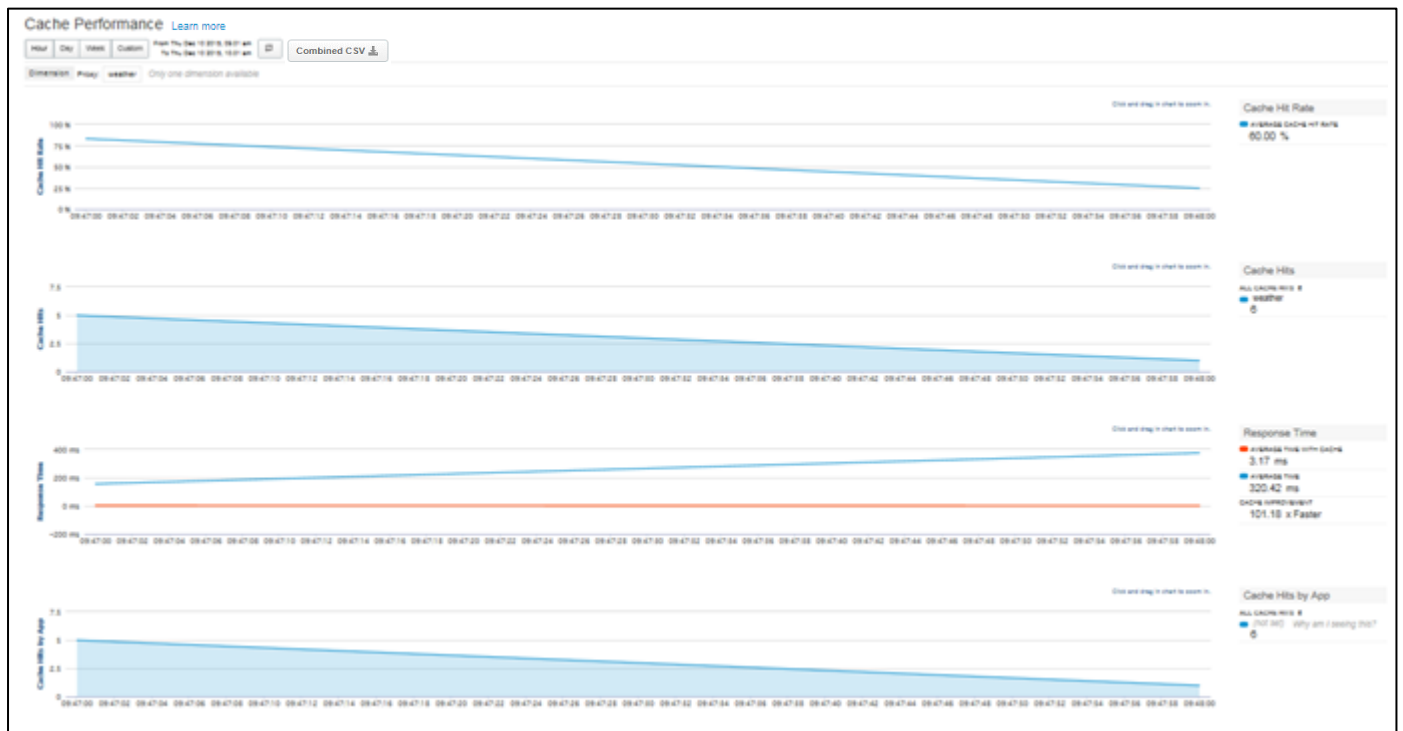
- Target Errors

TOTAL ERRORS	レスポンスの失敗件数が表示されます。
3XX ERRORS	ステータスコードが 300 系のレスポンス件数が表示されます。
4XX ERRORS	ステータスコードが 400 系のレスポンス件数が表示されます。
5XX ERRORS	ステータスコードが 500 系のレスポンス件数が表示されます。

- Payload Size

TOTAL PAYLOAD SIZE	リクエストとレスポンスのペイロードサイズ合計が表示されます。
REQUEST PAYLOAD SIZE	リクエストのペイロードサイズ合計が表示されます。
RESPONSE PAYLOAD SIZE	レスポンスのペイロードサイズ合計が表示されます。

Response Cache Policy のキャッシュヒット率や件数、応答時間を確認できる画面です。



- Cache Hit Rate

全体のキャッシュヒット率が表示されます。

- Cache Hits

ALL CACHE HITS	全体のキャッシュヒット件数が表示されます。
< Proxy name >	API Proxy 毎に名前とキャッシュヒット件数が表示されます。

- Response Time

AVERAGE TIME WITH CACHE	キャッシュがあった場合の平均応答時間が表示されます。
AVERAGE TIME WITHOUT CACHE	キャッシュをなかった場合の平均応答時間が表示されます。
CACHE IMPROVEMENT	キャッシュの有り無しでの速度の違いを倍率で表示します。

- Cache Hits by App

アプリの名前とキャッシュヒット件数が表示されます。

3.4.4. Latency Analytics

API Proxy の処理時間や、バックエンドサービスの応答時間を確認できる画面です。



- Latency Analysis の各グラフ項目

Mesian	API Proxy がクライアントからリクエストを受け、レスポンスを返すまでの時間に対する中央値が表示されます。
95th Percentile	API Proxy がクライアントからリクエストを受け、レスポンスを返すまでの時間に対する95 パーセンタイルの値が表示されます。
99th Percentile	API Proxy がクライアントからリクエストを受け、レスポンスを返すまでの時間に対する99 パーセンタイルの値が表示されます。

3.4.5. Error Code Analytics

API Proxy が処理するリクエストおよびレスポンスで発生したエラーの情報（件数やステータスコード等）を確認できる画面です。



● Error Composition

TOTAL ERRORS	全体のエラー発生件数が表示されます。
PROXY ERRORS	API Proxy で発生したエラーの合計件数が表示されます。
TARGET ERRORS	バックエンドサービスで発生したエラーの合計件数が表示されます。

- Proxy Errors

TOTAL PROXY ERRORS	API Proxy で発生したエラーの合計件数が表示されます。
4XX	API Proxy でHTTP ステータスコードが4XX 台となっているエラーの発生件数を表示します。
5XX	API Proxy でHTTP ステータスコードが5XX 台となっているエラーの発生件数を表示します。

- Target Errors

TOTAL TARGET ERRORS	バックエンドサービスで発生したエラーの合計件数が表示されます。
4XX	バックエンドサービスでHTTPステータスコードが4XX台となっているエラーの発生件数を表示します。
5XX	バックエンドサービスでHTTPステータスコードが5XX台となっているエラーの発生件数を表示します。

- Errors by Proxy

Proxy 毎に名前と発生したエラー件数が表示されます。

- Errors by Target

バックエンドサービス毎に IP アドレスと発生したエラー件数が表示されます。

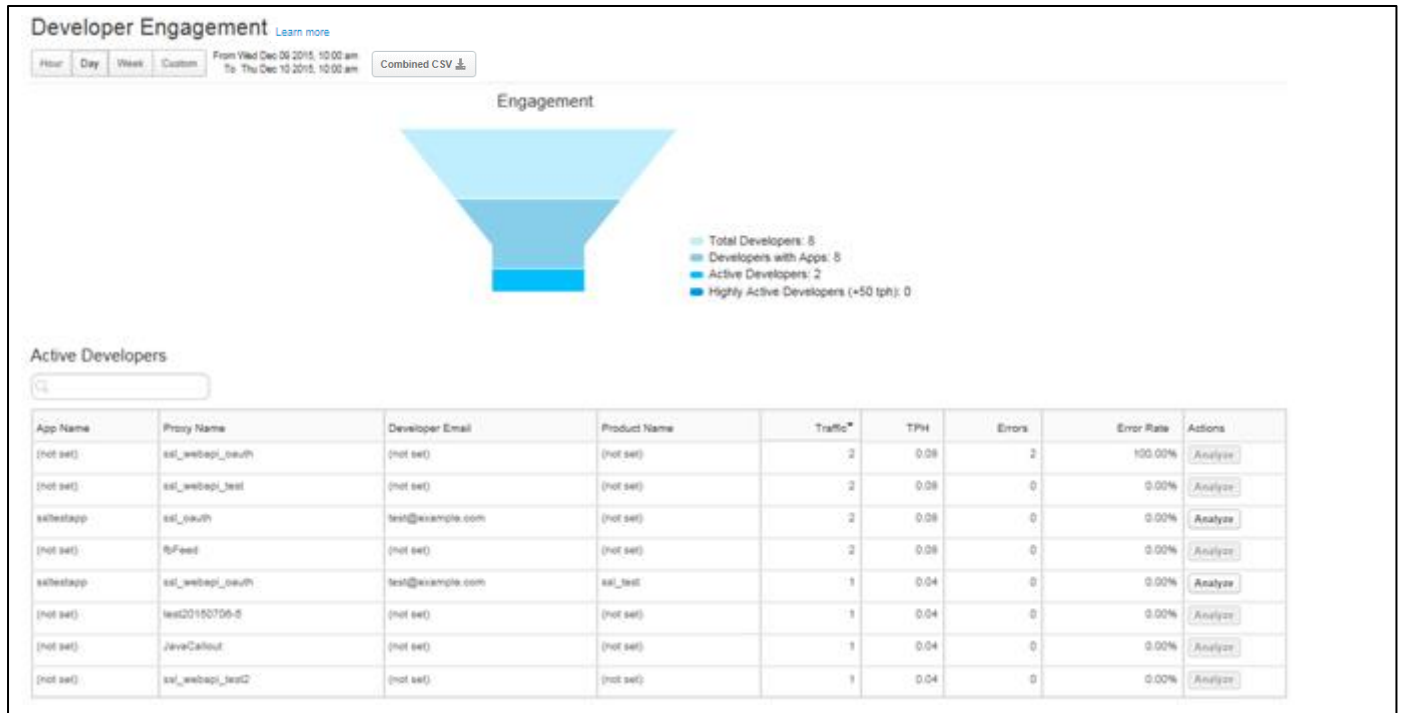
- Proxy Errors by Response Code

API Proxy で発生したエラーのステータスコードとエラー件数が表示されます。

- Target Errors by Response Code

エンドポイントで発生したエラーのステータスコードとエラー件数が表示されます。

アプリ開発者が生成したトラフィックの情報（アプリ開発者数、アクセス状況、トラフィック量、エラー率）が確認できる画面です。



● Engagement

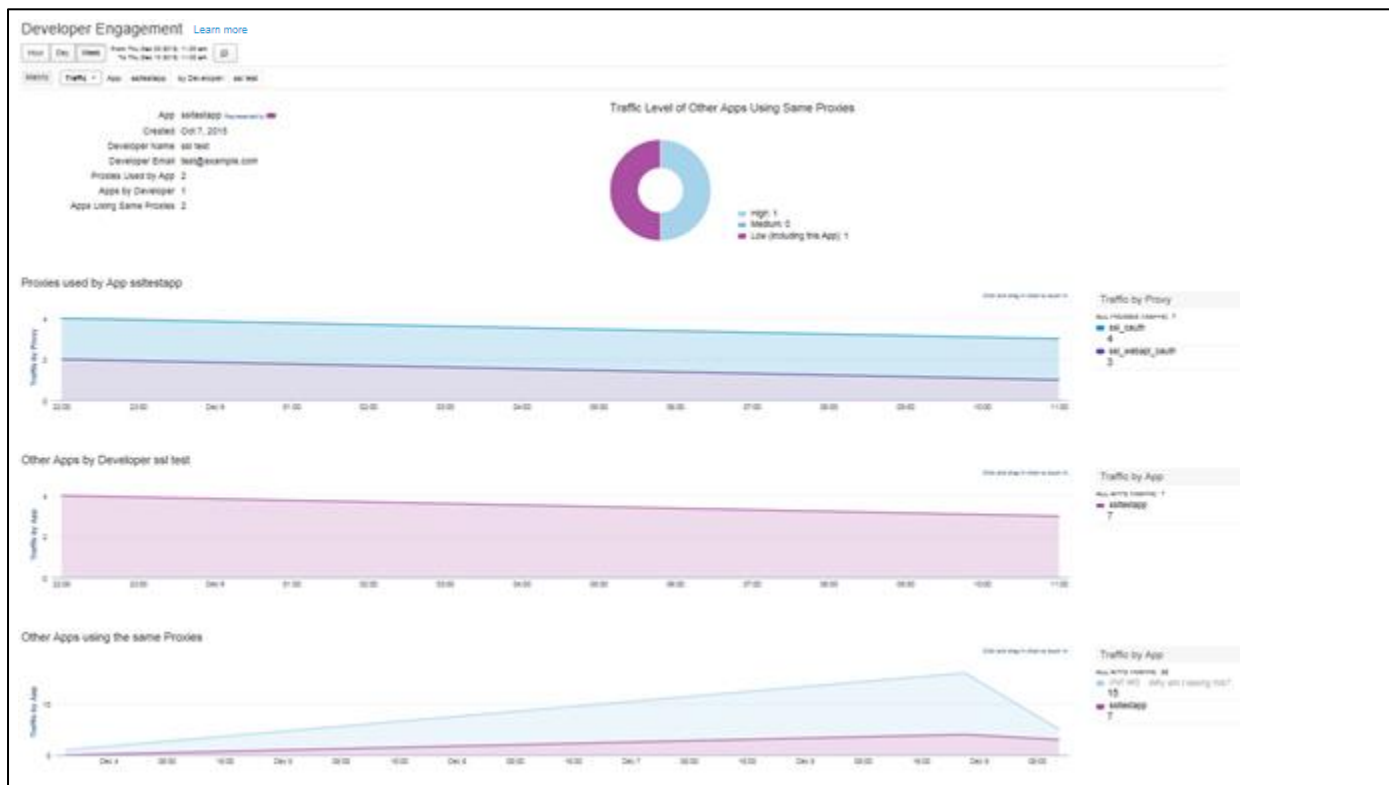
Total Developers	組織内のAPI Proxyに関連するアプリ開発者の合計人数を表示します。
Developers with Apps	組織内のアプリに関連するアプリ開発者の合計人数を表示します。
Active Developers	API Proxy経由でリクエストを送信した、アプリ開発者の人数を表示します。
Highly Active Developers (+50 tph)	1時間あたりのトランザクション数が、50以上のアプリ開発者人数を表示します。

● Active Developers

App Name	アプリの名前を表示します。
Proxy Name	アプリに関連するProxyの名前を表示します。
Developer Email	アプリを登録したアプリ開発者のメールアドレスを表示します。
Product Name	アプリに関連するProductの名前を表示します。
Traffic	アプリが生成したトラフィック量を表示します。
TPH	1時間あたりのトランザクション生成数を表示します。
Errors	アプリで発生したエラー数を表示します。
Error Rate	アプリで発生したエラーの割合（エラー数/リクエスト数）を表示します。

- Analyze ボタン

ボタンを押下すると、アプリの情報（トラフィック量、エラー率、応答時間）を表示する画面が表示されます。



- Metric

Traffic	トラフィック量を表示します。
Errors	エラー数を表示します。
Response Time	応答時間を表示します。

- Traffic Level of Other Apps Using Same Proxies

High	1時間あたりのトランザクション数が、500以上のアプリの数を表示します。
Medium	1時間あたりのトランザクション数が、50以上かつ500未満のアプリの数を表示します。
Low	1時間あたりのトランザクション数が、50未満のアプリの数を表示します。

- Proxies used by App

API Proxy 全体の Metric に対する合計値を表示します。

- Other Apps by Developer

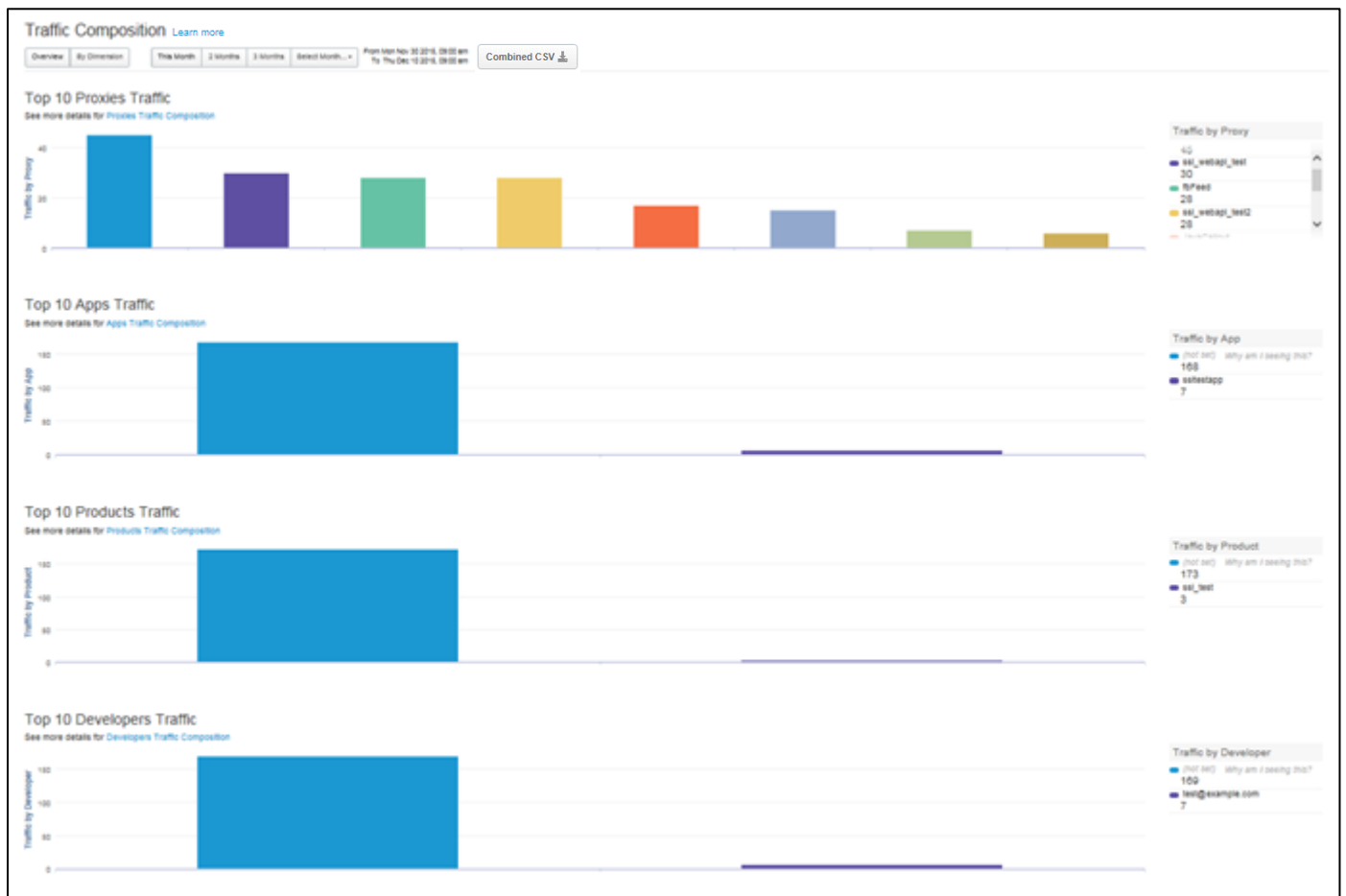
アプリ開発者が作成する全てのアプリの、Metric に対する合計値を表示します。

- Other Apps using the same Proxies

表示中のアプリが利用している API Proxy に対する、Metric の合計値（他のアプリを含む）を表示します。

3.4.7. Traffic Composition

API Proxy、App、Product、Developerそれぞれのトラフィック量 Top10を確認できる画面です。



- Top 10 Proxies Traffic

API Proxy の名前とトラフィック量を表示します。

- Top 10 Apps Traffic

App の名前とトラフィック量を表示します。

- Top 10 Products Traffic

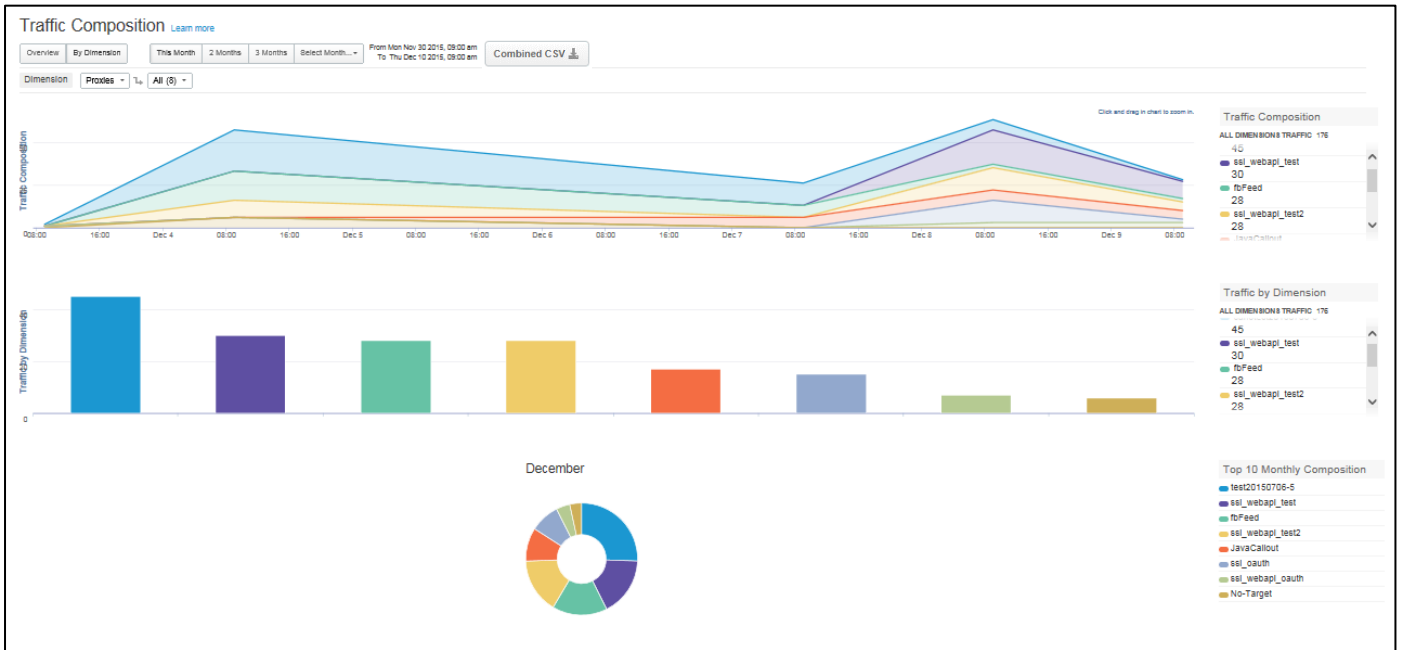
Product の名前とトラフィック量を表示します。

- Top 10 Developers Traffic

Developer の名前とトラフィック量を表示します。

- By Dimension ボタン

「By Dimension」 ボタンを押下すると、API Proxy、App、Product、Developer それぞれに対してドリルダウン可能な画面が表示されます。



- Dimension

Proxies	API Proxy毎のトラフィック量を表示します。
Developer Apps	アプリ毎のトラフィック量を表示します。
Developers	アプリ開発者毎のトラフィック量を表示します。
Products	Product毎のトラフィック量を表示します。

- Traffic Composition

トラフィック量の合計を表示します。

- Top 10 Monthly Composition

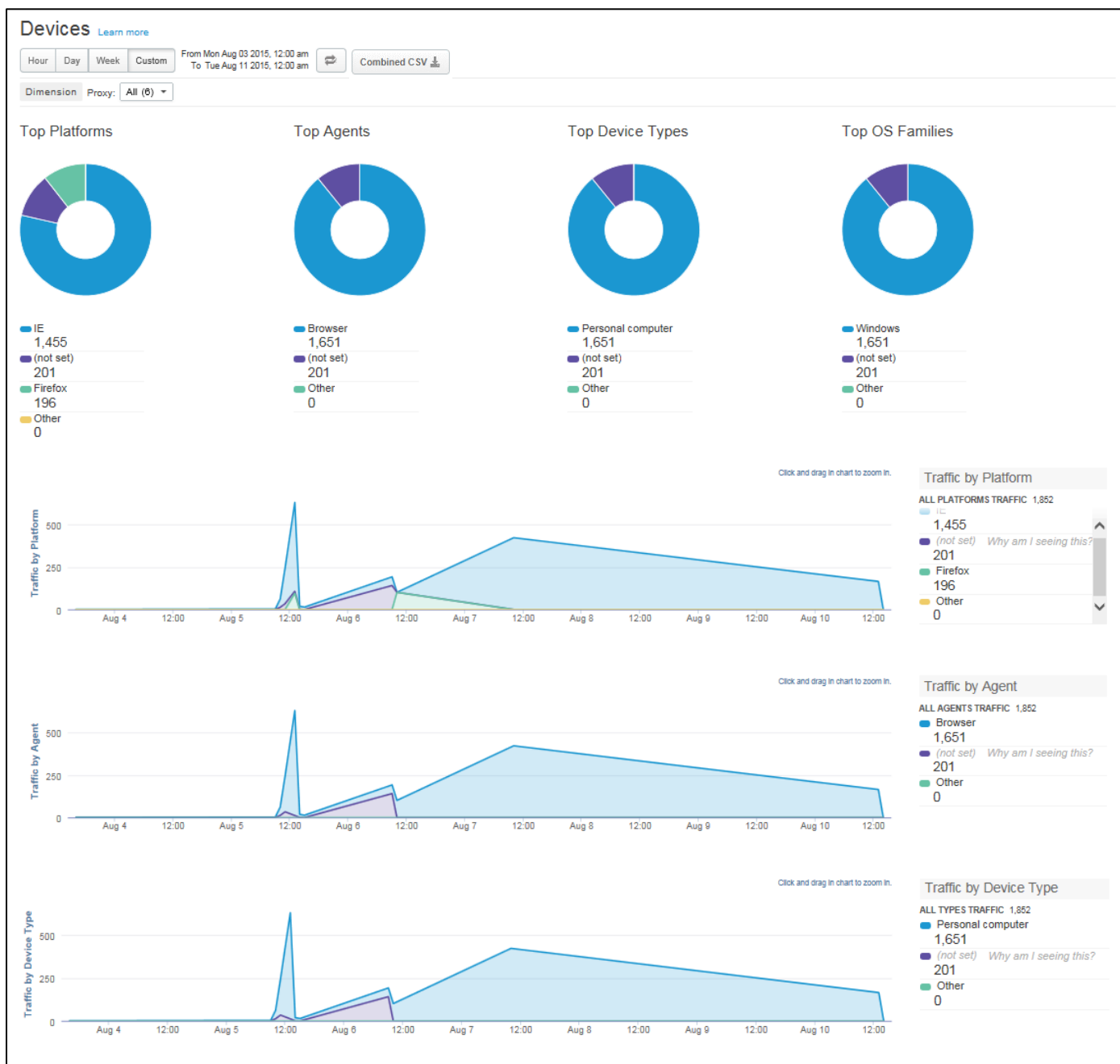
API Proxy 別のトラフィック量 Top10 を円グラフで割合表示します。

- Traffic by Dimension

API Proxy の名前が表示されます。

3.4.8. Devices

API に対するアクセス元のデバイス情報 (Platform、Agent、Device Type、OS Families) を確認できる画面です。また、表示しているデバイス情報は、API Proxy でドリルダウンすることができます。



- Traffic by Platform

アクセス元が利用している Platform (Google Chrome、Safari、FireFox、cURL、IE 等) とトラフィック量を表示します。

- Traffic by Agent

アクセス元が利用している Agent (browser、robot、library 等) とトラフィック量を表示します。

- Traffic by Device Type

アクセス元が利用している Device Type (Personal computer、mobile device 等) とトラフィック量を表示します。

3.4.9. Reports

Pro

フルアナリティクス

Custom Report の一覧画面です。

Custom Reports

Search All 1-1 of 1 < > + Custom Report

Report Name	Environment	Metric and Dimension	Description	Last Modified	Actions
test	prod, test	Metric: Sum of Traffic Dimension: Day of week		Nov 13, 2020 2:22:27 PM	Delete Roles

- + Custom Report ボタン

「+ Custom Report」 ボタンを押下すると、3.4.9.1 New Custom Report の画面を表示します。

- Report Name

Custom Report の Report Name をクリックすると、Custom Report のグラフを表示します。

testLine [Learn more](#)

Chart Table Thu 10 Nov 2016 00:00 - Fri 25 Nov 2016 00:00 Combined CSV 1-10 of 13

Dimension
Response Status Code All

Proportion

Sum of Traffic

Response Status Code	Sum of Traffic
200	8,500
304	2,897
201	2,056
401	634
504	249
404	87
202	58
400	52
503	15
500	5

Dispersion

Range 0.00 - 54,627.18
Quartile 1 1,254.19
Median 2,716.79
Quartile 3 8,164.21

Average of Target Response Time

Response Status Code	Average of Target Response Time
200	3,002.1
304	8,798.7
201	354.22
401	2,431.49
504	54,627.18
404	1,565.39
202	8,164.21
400	7,896.00
503	0.00
500	5,025.00

Dispersion

Range 0.00 - 59,939.00
Quartile 1 7,896.00
Median 47,931.00
Quartile 3 54,833.00

Maximum of Target Response Time

Response Status Code	Maximum of Target Response Time
200	54,833
304	55,000
201	54,236
401	47,230
504	59,939
404	48,632
202	43,945
400	7,896
503	0
500	6,123

Summary

Response Status Code	Sum of Traffic	Average of Target Response Time	Maximum of Target Response Time	Actions
200	8,500.00	3,002.10	54,833.00	Analyze
304	2,897.00	8,798.70	55,000.00	Analyze
201	2,056.00	354.22	54,236.00	Analyze
401	634.00	2,431.49	47,230.00	Analyze
504	249.00	54,627.18	59,939.00	Analyze
404	87.00	1,565.39	48,632.00	Analyze
202	58.00	8,164.21	43,945.00	Analyze
400	52.00	1,254.19	7,896.00	Analyze
503	15.00	0.00	0.00	Analyze
500	5.00	5,025.00	6,123.00	Analyze

■ Edit ボタン

「Edit」 ボタンを押下すると、Custom Report の編集画面を表示します。

testLine

Basics

Report Name

Report Description

Chart Type Column Line
For Column charts, the x-axis represents groups designated by dimensions. For Line charts, the x-axis represents time.

Metrics

The y-axis represents metric values.

	Metric	Aggregate Function	Actions
1	<input type="text" value="Traffic"/>	<input checked="" type="radio"/> Sum <input type="radio"/> Average <input type="radio"/> Min <input type="radio"/> Max	<input type="button" value="Delete"/>
2	<input type="text" value="Response Processing Latency"/>	<input type="radio"/> Sum <input checked="" type="radio"/> Average <input type="radio"/> Min <input type="radio"/> Max	<input type="button" value="Delete"/>

Dimensions

Dimensions have two purposes:

- Initially, the dimension is used to group data, similar to the GROUP BY clause in SQL.
- Once a dimension is selected, it becomes a filter, similar to the WHERE clause in SQL, working as a drill down, and the subsequent dimensions becomes the grouping mechanism.

	Dimension	Actions
1	<input type="text" value="Proxy Client IP"/>	

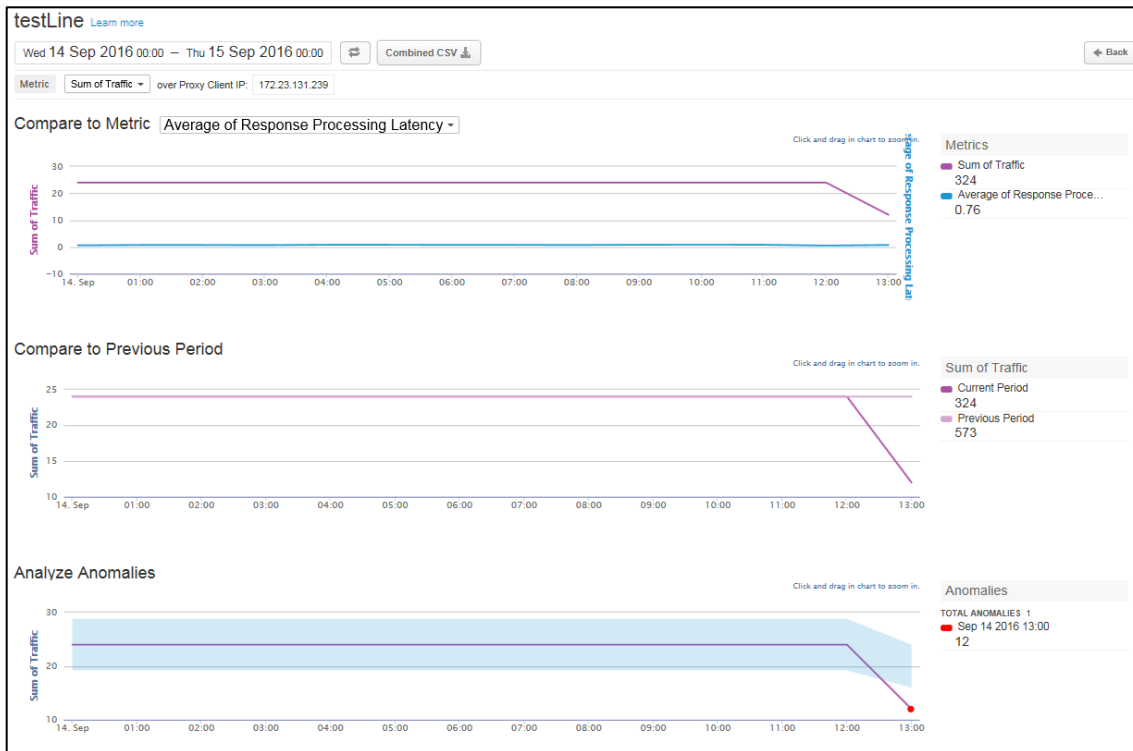
Filter

Filter Conditions

Connector	Name	Operator	Value	Actions
-----------	------	----------	-------	---------

■ Analyze ボタン

選択した Dimension のグラフを表示します。



■ Metrics

Custom Report を作成する際に選択した Metrics が表示されます。Metrics の詳細は、3.4.9.1 New Custom Report を参照してください。

■ Compare to Metric

Metrics と Compare to Metric で選択した Metric のグラフを重ねて表示します。Compare to Metrics で None を選択した場合は、選択中の Metrics のグラフのみ表示します。

■ Compare to Previous Period

指定した期間のグラフと、一つ前の期間のグラフを重ねて表示します。

例) 11月11日の00:00~23:59までを期間として指定した場合、下記2グラフを重ねて表示します。

- 11月10日の00:00~23:59のグラフ
- 11月11日の00:00~23:59のグラフ

■ Analyze Anomalies

選択した Metrics のグラフに対して、平均値の最少と最大を示す水色の帯を重ねて表示します。

3.4.9.1. New Custom Report

Custom Report の作成画面です。Y 軸となる Metrics と X 軸となる Dimension または時間を自由を選択してグラフを作成することができます。また、Dimensions は複数指定することで、ドリルダウンが可能です。

New Custom Report

Basics

Report Name

Report Description

Chart Type Column Line
For Column charts, the x-axis represents groups designated by dimensions. For Line charts, the x-axis represents time.

Metrics

The y-axis represents metric values.

	Metric	Aggregate Function	Actions
1	Select... <input type="button" value="v"/>	<input type="radio"/> Sum <input type="radio"/> Average <input type="radio"/> Min <input type="radio"/> Max	

Dimensions

Dimensions have two purposes:

- Initially, the dimension is used to group data, similar to the GROUP BY clause in SQL.
- Once a dimension is selected, it becomes a filter, similar to the WHERE clause in SQL, working as a drill down, and the subsequent dimensions becomes the grouping mechanism.

Dimension	Actions

Filter

Filter Conditions

Connector	Name	Operator	Value	Actions

- Basics

- Chart Type

Column	グラフの x 軸を指定した Dimensions で表示します。
Line	グラフの x 軸を時間で表示します。

- Metrics

- Metric

Average Transactions per Second	1 秒あたりのリクエストとレスポンスの件数を表示します。
Cache Hit	キャッシュにヒットしたリクエスト件数を表示します。キャッシュヒットしたリクエストは、バックエンドサービスに転送されません。
L1 Cache Elements Count	トランザクション中における L1 キャッシュ上の要素の数を表示します。
Policy Errors	Policy でエラーとなったリクエストの件数を表示します。
Proxy Errors	Proxy でエラーとなったリクエストの件数を表示します。
Request Processing Latency	Proxy がリクエストを処理する時間を表示します。
Request Size	リクエストのデータサイズ (KB) を表示します。
Response Cache Executed	Response Cache Policy の実行数を表示します。
Response Processing Latency	Proxy がレスポンスを処理する時間を表示します。
Response Size	レスポンスのデータサイズ (KB) を表示します。
Target Errors	バックエンドサービスでエラーとなったリクエストの件数を表示します。
Target Response Time	Proxy がバックエンドサービスにリクエストを送信してからクライアントにレスポンスを返し始めるまでの時間を表示します。
Total Response Time	Proxy がクライアントからリクエスト受け取ってからレスポンスを返し始めるまでの時間を表示します。
Traffic	リクエストとレスポンスの件数を表示します。

- + Metric ボタン

「+ Metric」ボタンを押下すると、Metric を 1 行追加します。Metrics は複数設定することが可能で、設定した件数分のグラフが画面に表示されます。

- Dimensions

Access Token	OAuth のアクセストークン (OAuth Policy を使用した際) を表示します。
Developer App	リクエスト元のアプリの名前を表示します。
Developer ID	リクエスト元のアプリ開発者を表示します。
Flow Resource	リクエスト対象のバックエンドサービスのリソースを表示します。
Proxy	API Proxy の名前を表示します。
Proxy Client IP	API Proxy の IP アドレスを表示します。
Proxy Path Suffix	API Proxy URI のベースパスより後ろの文字列を表示します。
Request Verb	リクエストメソッドを表示します。

Response Status Code	レスポンスコードを表示します。
Target Resonse Code	バックエンドサービスのレスポンスコードを表示します。
Target URL	バックエンドサービスの URL を表示します。
User Agent	User-Agent ヘッダーの値を表示します。
X Forwarded For	X Forwarded For ヘッダーの値を表示します。
API Product	リクエストを受け取った API Proxy をメンバにもつ Product を表示します。
Business Unit ID	事業 ID を表示します。
Cache Key	Cache Key を表示します。
Cache Name	Cache Name を表示します。
Cache Source	Cache Source を表示します。
Channel ID	チャンネル ID を表示します。
City	リクエスト元の都市名を表示します。
Client Application Name	クライアントアプリの名前を表示します。
Client Host	クライアントアプリをホスティングしているコンピュータの名前を表示します。
Client ID	API Key を表示します。 ※Developers 画面で発行した API Key です。
Client IP Address	リクエスト元の IP アドレスを表示します。
Client Organization Name	クライアントの組織名を表示します。
Client Request ID	クライアントのリクエスト ID を表示します。
Continent	リクエスト元の大陸名を表示します。
Country	リクエスト元の国名を表示します。
Day of week	リクエストした曜日を表示します。
Developer Email	開発者のメールアドレスを表示します。 ※Developers 画面で登録した Developer のメールアドレスです。
Device Category	デバイスの種類 (Personal computer 等) を表示します。 ※User-Agent に含まれる情報です。
Device ID	デバイス ID を表示します。
Environment	API Proxy がデプロイされている環境の名前 (test、prod 等) を表示します。
Flow Name on Error	エラーが発生したフロー名を表示します。
Flow State on Error	エラーが発生したフローの状態を表示します。
Gateway Flow ID	ゲートウェイフローの ID を表示します。(VM-8D5-S-0011_BTnU7OVE_RouterProxy-8-1047_6 等)
Gateway Source	ゲートウェイ (router、message_processor 等) を表示します。
Geographical Region	地理的地域を表示します。
Market ID	マーケット ID を表示します。

Month	リクエストした月を表示します。
OS Family	OSの種類 (Windows) を表示します。 ※User-Agent に含まれている情報です。
OS Version	接続元の OS バージョンを表示します。 例) 6.1 (Windows7 のコードバージョン) ※User-Agent に含まれている情報です。
Organization	組織名を表示します。
Partner ID	パートナーID を表示します。
Policy Name on Error	エラーが発生した Policy の名前を表示します。
Proxy Base Path	API Proxy の Base Path を表示します。
Proxy Revision	API Proxy の Revision(1, 2, 3 … 等) を表示します。
Referred Client IP	参照されたクライアント IP を表示します。
Region	地域を表示します。
Request Path	リクエスト URI のパス部分を表示します。
Request URI	リクエスト URI を表示します。
Session ID	セッション ID を表示します。
Target	Target Endpoint 名を表示します。
Target Base Path	バックエンドサービスの Base Path を表示します。
Target Host	バックエンドサービスのドメイン名を表示します。
Target IP Address	バックエンドサービスの IP アドレスを表示します。
Time Zone	リクエスト元のタイムゾーンを表示します。
Time of Day	リクエスト時刻を表示します。
Traffic Referral ID	参照元の ID を表示します。
User Agent Family	エージェントのカテゴリ (Chrome、IE 等) を表示します。
User Agent Type	エージェントの型 (Browser 等) を表示します。
User Agent Version	エージェントのバージョン (Chrome→43.0、IE→11.0 等) を表示します。
Virtual Host	Virtual Host 名を表示します。
Week of Month	リクエストした週を表示します。
ax_isp	ax_isp を表示します。

- Filter

Metrics または Dimensions の条件を指定することで、グラフに表示するデータをフィルタリングします。

3.4.10. 統計情報取得用 WebAPI

WebAPI 機能を使うことで、API Proxy および Publish 機能 (Products、Developers、Developer Apps) の統計情報を取得することができます。詳細は、別紙「API Management WebAPI リファレンス」をご参照ください。

3.5. Admin

3.5.1. Organization Users

3.5.1.1. List

ユーザーの一覧画面です。

Organization Users			
Search	All ▾		1-6 of 6 < > + User
User ▲	Email	Roles	Actions
a-test@example.co.jp	a-test@example.co.jp	all ok	Remove

- User 欄

User 欄のリンクをクリックすると、ユーザーの詳細画面が表示されます。

		Edit	Remove
Email	a-test@example.co.jp		
First Name			
Last Name			
Roles	all ok		

- Edit ボタン

「Edit」ボタンを押下すると、First Name、Last Name、Roles の3項目が修正可能な状態になります。

- Roles 欄

リンクをクリックすると Role の詳細画面が表示されます。ユーザーに割り当てられているロールの種類 (Built-in Role と Custom Role) によって、表示される画面が変わります。ロールの設定に関しましては、3.5.2 Organization Roles を参照してください。

- Built-in Role の場合

Organization Administrator			
Permissions	Resource	Path ▲	Operations
	Organization resource /		GET PUT DELETE

■ Custom Role の場合

all ok Edit Delete

API Proxies

All API Proxies Create View Edit Delete

Specific API Proxies

Deployment Status View

Environments

prod API Proxies Trace Deploy
 All Caches Create View Edit Delete

Specific Caches

test API Proxies Trace Deploy
 All Caches Create View Edit Delete

Specific Caches

Products

All Products Create View Edit Delete

Specific Products

Developers and Apps

Developers Create View Edit Delete

All Apps View

Developer Apps Create View Edit Delete

Company Apps Create View Edit Delete

Reports

All Reports Create View Edit Delete

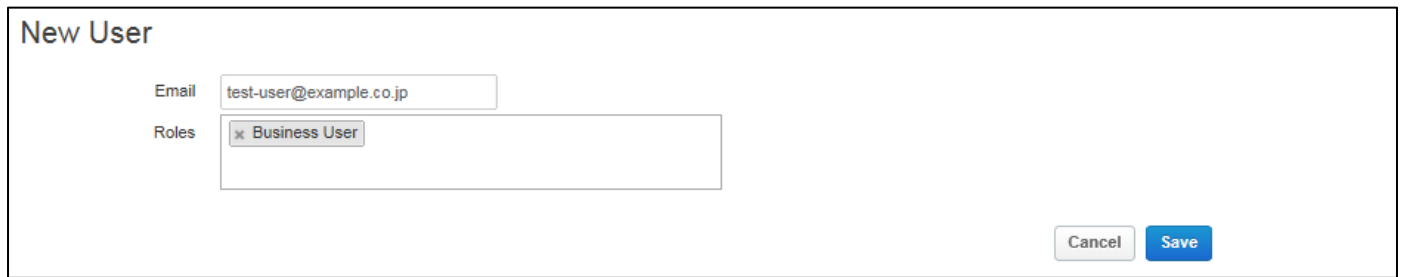
Specific Reports

■ Edit ボタン

「Edit」ボタンを押下すると、各項目の権限が編集可能な状態になります。

3.5.1.2. New User

ユーザーの作成画面です。



The screenshot shows a web form titled "New User". It contains two input fields: "Email" with the value "test-user@example.co.jp" and "Roles" with a dropdown menu showing "Business User". At the bottom right, there are two buttons: "Cancel" and "Save".

- Email

ユーザーの Email を設定します。ここで設定した Email のアドレス宛に、ユーザーのアクティベーションメールが送信されます。メールに従ってユーザーのアクティベーションをおこなうことで、本サービスにログインできるようになります。

Email アドレスとして使用可能な文字は下記のとおりです。

- 英数字 ([a-z][A-Z][0-9])
- -(マイナス), _(アンダースコア), .(ピリオド)

- Roles

ユーザーに割り当てるロールを指定します。入力欄にフォーカスが移ると、ロール一覧 (Built-in Role と Custom Role) が表示されます。

3.5.2. Organization Roles

3.5.2.1. List

Built-in Role と Custom Role の一覧が表示されます。Custom Roles は、追加・修正・削除が可能です。Built-in Roles は追加・修正・削除できません。

Organization Roles		
Built-in Roles		
Role ▲	Users	
Business User		
Operations Administrator		
Organization Administrator	admin admin	
User		
Custom Roles		
		+ Custom Role
Role ▲	Users	Action
test		× Delete

Only custom roles with no users can be deleted.

- Built-in Roles

システムで用意されている、デフォルトのロールが表示されます。デフォルトロールの権限は、下表を参照してください。

注意：デフォルトロールの権限は、追加、変更、削除ができません。

	test 環境						prod 環境						その他				
	API Proxy の表示	API Proxy の追加・修正・削除	API Proxy のデプロイ	トリーツールの利用	Publish 機能の表示	Publish 機能の追加・修正・削除	API Proxy の読み取り	API Proxy の追加・修正・削除	API Proxy のデプロイ	トリーツールの利用	Publish 機能の表示	Publish 機能の追加・修正・削除	CustomReport の表示	CustomReport の追加・修正・削除	Environment Configuration 画面	Admin の表示	Admin 画面の追加・修正・削除
Business User	○	※	×	○	○	○	○	※	×	○	○	○	○	×	×	×	○
Operations Administrator	○	×	○	○	○	×	○	×	○	○	×	○	×	×	×	×	○
Organization Administrator	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○
User	○	○	○	○	○	×	○	※	×	○	○	×	○	×	×	×	○

※…デプロイされている API Proxy は修正と削除ができません。アンデプロイすることで、修正と削除が可能になります。

- Custom Roles Roles

作成したロールが表示されます。ロールの作成画面については、3.5.2.2 New Custom Role を参照してください。

- + Custom Role ボタン

「+ Custom Role」ボタンを押下すると、3.5.2.2 New Custom Role の画面を表示します。

3.5.2.2. New Custom Role

Custom Role の作成画面です。チェックを入れることで、その権限を付与します。

- API Proxies

API Proxies を対象にした権限の設定が可能です。

All API Proxies	全ての API Proxy に対して、チェックを入れた操作（Create、View、Edit、Delete）の権限を設定します。
Specific API Proxies	選択した API Proxy に対して、チェックを入れた操作（View、Edit、Delete）の権限を設定します。また、環境毎に Trace と Deploy の実行権限を設定することが可能です。
Deployment Status	チェックを入れない場合、3.2.1.1 List の Environment 欄にマスクがかかります。

- Environments

環境 (test、prod) 毎に、API Proxy の Trace や Deploy、Cache の権限を設定することが可能です。

API Proxies	Trace と Deploy の実行権限を設定します。
All Caches	全ての Cache に対して、チェックを入れた操作 (Create、View、Edit、Delete) の権限を設定します。
Specific Caches	選択した Cache に対して、チェックを入れた操作 (View、Edit、Delete) の権限を設定します。

- Products

Products を対象にした権限の設定が可能です。

All Products	全ての Product に対して、チェックを入れた操作 (Create、View、Edit、Delete) の権限を設定します。
Specific Products	選択した Product に対して、チェックを入れた操作 (View、Edit、Delete) の権限を設定します。

- Developers and Apps

Developer と App を対象にした権限の設定が可能です。

Developers	全ての Developer に対して、チェックを入れた操作 (Create、View、Edit、Delete) の権限を設定します。
All Apps	全ての App に対して、操作 (View) の権限を設定します。
Developer Apps	Developer App に対して、チェックを入れた操作 (Create、View、Edit、Delete) の権限を設定します。
Company Apps	本機能はサポート対象外です。

- Reports

Custom Reports を対象にした権限の設定が可能です。

All Reports	全ての Custom Report に対して、チェックを入れた操作 (Create、View、Edit、Delete) の権限を設定します。
Specific Reports	選択した Custom Report に対して、チェックを入れた操作 (View、Edit、Delete) の権限を設定します。

3.5.3. TLS Certificates

TLS 証明書の詳細画面です。画面左側には、本サービスに登録されている証明書のツリー、画面右側には、左側のツリーで選択されている証明書の情報（Subject、Issuer）が表示されます。

The screenshot shows a web interface for managing TLS certificates. On the left, a tree view lists several keystore entries: 'Keystore service-ks', 'Keystore mySAMLKeystore', and 'Keystore onelogin'. Under 'Keystore onelogin', 'Truststore onelogin' is expanded, and 'OneLogin Account 92271' is selected. On the right, the 'TLS Certificate Details' are shown. The certificate is 'Valid' and expires on 'Sep 20, 2021 11:02:21 AM (in 4 years)'. The subject is 'OneLogin Account 92271' and the issuer is 'OneLogin IdP'.

TLS Certificate Details	
Status	Valid
Valid From	Sep 19, 2016 11:02:21 AM
Expires	Sep 20, 2021 11:02:21 AM (in 4 years)
Basic Constraints	CA:FALSE
Serial Number	[Hexadecimal string]
Signature Algorithm	SHA1withRSA
Public Key	RSA Public Key, 2048 bits
Subject	
Common Name	OneLogin Account 92271
Alternative Names	
Organization Unit	OneLogin IdP
Organization	[Hexadecimal string]
Locality	
State/Province	
Country	US
Issuer	
Common Name	OneLogin Account 92271
Organization Unit	OneLogin IdP
Organization	[Hexadecimal string]
Locality	
State/Province	
Country	US

- TLS Certificate Details

証明書の状態や有効期限、セキュリティ（鍵長やハッシュ方式）の情報が表示されます。

- Subject

証明書の発行先情報（識別名）が表示されます。

- Issuer

証明書の発行者情報（識別名）が表示されます。

3.5.3.1. 証明書管理用 WebAPI

ゲートウェイ拡張

バックエンドセキュア接続

WebAPI 機能を使うことで、TLS 証明書の Keystore/Truststore の一覧取得や作成、Cert ファイル/JAR ファイルのアップロードなどが可能です。詳細は、別紙「API Management WebAPI リファレンス」をご参照ください。

3.5.4. Organization History

組織に対する操作履歴が表示されます。

Organization History		Sep 7, 2016 - Sep 14, 2016 Change	
Search	All	1-20 of 102 < >	
Operation	Time	User	Status
Create report	Details 9/14/16 1:46 PM 37 minutes ago	admin admin	201 ✓
Create report	Details 9/14/16 1:21 PM an hour ago	admin admin	201 ✓
Create permission	Details 9/14/16 8:41 AM 6 hours ago	admin admin	201 ✓
Create permission	Details 9/14/16 8:40 AM 6 hours ago	admin admin	201 ✓
Create permission	Details 9/14/16 8:40 AM 6 hours ago	admin admin	201 ✓
Create /v1/organizations/scenario/environments/test/apipatterns	Details 9/13/16 5:01 PM 21 hours ago	admin admin	201 ✓

- Details

Details リンクをクリックすると、Operation の詳細が表示されます。

Create report ×

Operation **CREATE**

Time 9/14/16 1:46 PM, an hour ago

Request URI /v1/organizations/scenario/reports/31db0a18-9988-46ca-afa1-a15082edfa7d/

User admin admin

Response Code **201**

Request "scenario"









[Close](#)

4. Policy 一覧

ここでは、各 Policy について説明します。

4.1. TRAFFIC MANAGEMENT

APIのトラフィックに関する処理（流量制限、キャッシュの設定等）をおこなう Policy です。

 Quota	時間の単位（月、日、時、分）と回数を指定することで、アプリのリクエスト件数を制限します。
 Spike Arrest	アプリからバックエンドサービスに対する、秒間あたりのリクエスト件数を指定して制限します。 例) 毎分 30 件 (30pm) 許可する設定にした場合、2 秒毎に 1 件許可します。2 秒以内に 2 件来た場合、2 件目は処理されません。
 Concurrent Rate Limit	バックエンドサービスに同時接続できるアプリの数を制限します。
 Response Cache	バックエンドサービスからのレスポンスをキャッシュします。
 Lookup Cache	Populate Cache でキャッシュしたデータを取得します。
 Populate Cache	セッション ID や認証情報等、任意のデータをキャッシュします。
 Invalidate Cache	条件を指定して Populate Cache でキャッシュしたデータを削除します。
 Reset Quota	Quota でカウントしたリクエスト件数が、指定した値に達した場合件数をリセットします。

4.2. SECURITY










API のセキュリティに関する制御（認証、脆弱性対策等）をおこなう Policy です。

 Basic Authentication	Basic 認証（Base64 のエンコードまたはデコード）の設定ができます。
 XML Threat Protection	XML の脆弱性に対する攻撃を防ぐ設定ができます。
 JSON Threat Protection	JSON の脆弱性に対する攻撃を防ぐ設定ができます。
 Regular Expression Protection	正規表現でリクエストを拒否することができます。
 OAuth v2.0	OAuth2.0 のエンドポイントに対する設定（アクセストークンの生成やチェック等）ができます。
 Get OAuth v2.0 Info	OAuth2.0 のアクセストークンや認証コード等の情報を取得することができます。
 Set OAuth v2.0 Info	OAuth2.0 のアクセストークンに関連付けられたカスタム属性を追加・更新することができます。
 Delete OAuth v2.0 Info	OAuth2.0 のアクセストークンや認証コード等の情報を削除することができます。
 OAuth v1.0a	OAuth1.0a のエンドポイントに対する設定（アクセストークンの生成やチェック等）ができます。
 Get OAuth v1.0a Info	OAuth1.0a のアクセストークンや認証コード等の情報を取得することができます。
 Delete OAuth v1.0 Info	OAuth1.0a のアクセストークンや認証コード等の情報を削除することができます。
 Verify API Key	アクセスを許可する API Key を設定することができます。
 Access Control	IP アドレスによるアクセス許可・拒否設定ができます。
 LDAP	LDAP 認証の設定ができます。
 Generate SAML Assesion	送信する XML Request に SAML Assertion を追加します。
 Validate SAML Assertion	受信した SOAP Request に添付されている SAML Assertion をチェックし、無効なメッセージの場合は拒否します。
 Generate JWT	署名付き JWT を生成することができます。
 Verify JWT	署名付き JWT を検証することができます。
 Decode JWT	署名付き JWT を検証せずにデコードすることができます。
 Generate JWS *1	署名付き JWS を生成することができます。
 Verify JWS *1	署名付き JWS を検証することができます。
 Decode JWS *1	署名付き JWS を検証せずにデコードすることができます。

*1 東日本リージョン 3/西日本リージョン 3 のみ利用可能








4.3. MEDIATION

API のデータ加工（形式変換、メッセージ修正等）をおこなう Policy です。

 JSON to XML	JSON 形式を XML 形式に変換します。
 XML to JSON	XML 形式を JSON 形式に変換します。
 Raise Fault	ステータスコードに応じてカスタムメッセージが出力できます。
 XSL Transform	XML 形式を HTML やプレーンテキスト等の別フォーマットに変換します。
 SOAP Message Validation	受信した SOAP メッセージが、XSD スキーマまたは WSDL に準拠していない場合は拒否します。
 Assign Message	HTTP Request または Response メッセージの作成・修正が可能です。
 Extract Variables	リクエストまたはレスポンスが指定した条件（URI Path や Query Param 等）と一致した場合、変数を指定して追加することができます。
 Access Entity	API Management のデータストアから取得した情報（アプリ、API Product、Developer 等）を変数に設定します。
 Key Value Map Operations	PUT、GET、DELETE メソッドで Key/Value のペアを保存・検索・削除することができます。

4.4. EXTENSION

スクリプトの実行やメッセージ内のデータ収集、ログ記録をおこなう Policy です。

 Java Callout	Java を実行します。 ※詳細は 4.4.1 Java Callout Policy を参照してください。
 Python	Python を実行します。
 JavaScript	JavaScript を実行します。
 Service Callout	外部サービスを呼び出します。
 Flow Callout	Shared Flow を呼び出します。
 Statistics Collector	Analytics 用にメッセージ内のデータ (Product ID、価格、ターゲット URL 等) を収集することができます。
 Message Logging	syslog サーバにメッセージログを記録する事ができます。 ※ローカルディスクへの保存はご利用いただけません。

Java Callout Policy は、Java を使うことでカスタム Policy を実装することが可能です。他の Policy とは異なり API Management の GUI から API Proxy に対して追加することができないため、下記手順に従って追加してください。

※API のリクエストを送信する際は、追加対象の API Proxy をアンデプロイした状態にしてください。デプロイされた状態で追加のリクエストを送信した場合、レスポンスが 404 エラーとなり正常終了しません。

1. Java Callout Policy を追加する API Proxy の情報を確認します。
 - ✓ 組織の名前
 - ✓ API Proxy の名前
 - ✓ リビジョン番号
2. 追加対象の API Proxy に対して、Java Callout Policy で使う Java のクラスやパッケージを jar 形式でアップロードします。
3. 下記 4.4.1.1 Java Callout Policy 管理用 WebAPI を使って、Java Callout Policy を追加します。
4. API Management にログイン後、Java Callout Policy が追加されたことを確認します。

※Java Package Names 「io.apigee」、「com.apigee」は API Management で予約済みの名称のため、ご利用できません。また、Network I/O、ファイルシステムの読み込み/書き込み、現在のユーザー情報、プロセスリスト、CPU やメモリの使用率取得などのシステムコールはサポートされておりません。今後のアップデートで機能が無効になる恐れがあるため、Java Callout Policy でのご利用は避けてください。

4.4.1.1. Java Callout Policy 管理用 WebAPI

別紙「API Management WebAPI リファレンス」をご参照ください。

付録

A.1. 用語集

API Proxy	<p>API Proxy は、API Management のコアとなる要素で、バックエンドサービスとそれを利用するアプリとの間に位置し、アプリケーションからのリクエストをバックエンドサービスに渡すタイミング及びバックエンドサービスからのレスポンスをアプリケーションに返すタイミングで作用する機能です。</p> <p>API Management の機能を利用するためには、API Proxy を作成し、作成した API Proxy に対して Policy（作用する内容）を設定します。</p>
バックエンドサービス	<p>バックエンドサービスとは、API Management のバックエンドに指定する Web アクセスが可能なサービスを指します。EXTENSION Policy や Node.js を使用することで、複数のバックエンドサービスの呼び出し（マッシュアップ）や Web API 以外のバックエンドサービスの呼び出し（CRM、ERP、DWH、SOA 等）も可能です。</p>
Policy	<p>Policy はバックエンドサービスに対する付加機能であり、API Proxy に対して設定します。認証、Traffic 制限、キャッシュ、レスポンスの形式変換などの機能（4 Policy 一覧を参照してください）をコーディングなしで実装できます。</p>
Flow	<p>API Proxy に Policy の実行タイミングを設定することにより、API の振る舞いをプログラムすることができる機能です。</p>
Endpoint	<p>Endpoint は、Policy の実行タイミングです。大きく Proxy Endpoint と TargetEndpoint の2つに分かれており、ProxyEndpoint はアプリから API Proxy 間、TargetEndpoint は API Proxy からバックエンドサービス間の実行タイミングを指します。</p>
Base Path	<p>API Proxy のルートパス（URL）を示します。Base Path は、API Proxy 毎に重複しないよう設定する必要があります。</p>
Resource	<p>Base Path 配下に配置したリソースのパスを指します。Resource は API Proxy に複数指定することが可能です。</p>
Product	<p>Product は、API Proxy のグループを指します。グループ化することで、グループ単位にトラフィック制限や、アクセス権の設定が可能です。</p>
Developer	<p>本サービスでは、アプリの開発者を指します。バックエンドサービスの開発者は含まれません。</p>
Developer Apps	<p>Developer が開発する、バックエンドサービスと通信をおこなうアプリの事を指します。</p>

Environment	API Proxy のデプロイ先です。test と prod の 2 環境があり、一般的に、test 環境には試験用の API を公開し、prod 環境には本番用の API を公開します。
Organization	Organization はオブジェクト (API Proxy、Product、Developer、Developer Apps、Environment) のコンテナです。各オブジェクトの設定は、Organization User がおこないます。
バージョン	バージョンは API Proxy の Base Path に番号付けして管理することを指します。 例) test.example.co.jp/test_api/v1
リビジョン	リビジョンは、API Proxy の改定歴です。API Proxy の設定を、番号付けして保存することが可能で、過去のリビジョンをデプロイすることもできます。
Organization User	Organization に属しているユーザーの事を指します。このユーザーは、所属する Organization のオブジェクトを、与えられた権限 (Role) で操作することが可能です。
Virtual Host	Virtual Host はアプリがバックエンドサービスにアクセスする際 (API Proxy 経由) の URL とポート番号を定義します。default と secure の 2 つがあり、default には http プロトコルの URL とポート番号、secure には https プロトコルの URL とポート番号が定義されます。