HCL Software Academy
for Digital Solutions

HCL

HCL Volt MX

Business Object Data

HVMX-FOU-300 Volt MX Foundry Development

HCL SOFTWARE

オブジェクトサービス

- Object Services でモデル駆動型アプリケーション設計を実現

- Object Services を使用すると、好みのデータモデルを定義できます。このデータモデルは、アプリケーションがデータとどのようにやり取りしたいかを定義します。

- データモデルと、それがバックエンドシステムにどのようにマッピングされるかは、明確に分離されています。

- Object Services の完全な情報については下記参照。
  - https://opensource.hcltechsw.com/volt-mx-docs/docs/documentation/Foundry/voltmx_foundry_user_guide/Content/Objectservices.html

オブジェクトサービス（続き）

- Object Services を使用すると、ビジネスライン（LOB）オブジェクトからデータモデルを作成したり、企業内の既存の API からサービスドリブンなオブジェクトを定義したりできます。
- これらのビジネスアダプターを使用すると、LOB システムによって公開されるエンティティを視覚的に検出および選択できます。
- 既存のVolt MX Foundry - Integration/Orchestration - Servicesのセットから、サービス駆動型オブジェクトを作成できます。これらの統合サービスは、既存の API エンドポイントまたは Volt MX Foundry オーケストレーションサービスに接続し、複数の API を組み合わせて新しい複合 API または集約 API を作成します。
- オブジェクト・サービスは、その名前が示すように、データを参照するためのオブジェクトを作成し、操作できます。
  - 単純化されたアプリケーションでは、統合サービスの使用で十分かもしれません。
  - しかし、より大規模で複雑なアプリケーションを構築するようになると、再利用性が重要な役割を果たすようになります。このような場合、Objects Services を使用することで、作業を容易にできます。
- このサービスは、以前に定義したオブジェクトデータモデルを使用して、バックエンドデータと対話するのに役立ちます。

オブジェクトはどのように機能するのか

- オブジェクトサービスは、バックエンドデータとクライアントアプリケーションの間の橋渡しとなるデータモデルを作成することで、アプリケーション開発を容易にします。
- データモデルとは、オブジェクトの定義です。
- データモデルが定義されると、それを中心にアプリケーションを構築し始めることができます。

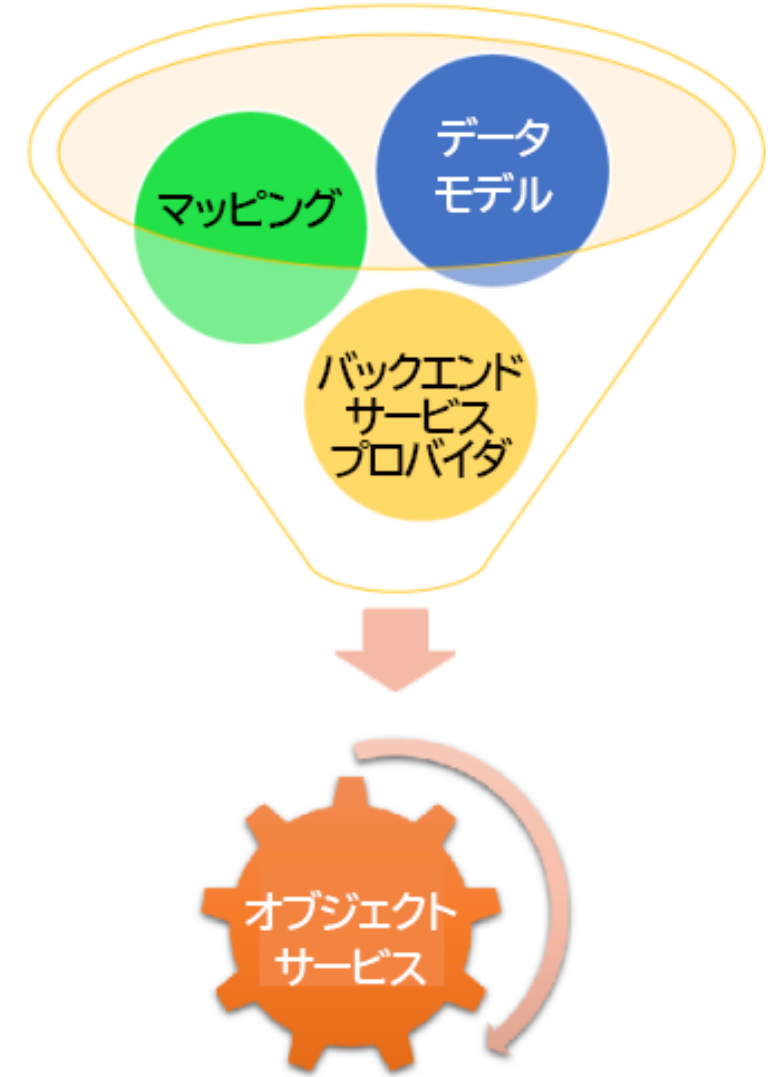
- 注: オフラインオブジェクト機能は、オフラインアクセス用にクライアントアプリケーションにデータを同期させるための簡便なアプローチとして役立ちます。
- Offline Objects 機能は Object Services と直接接続するため、同期ランタイムサーバーは必要ありません。

Object Services を使用する理由

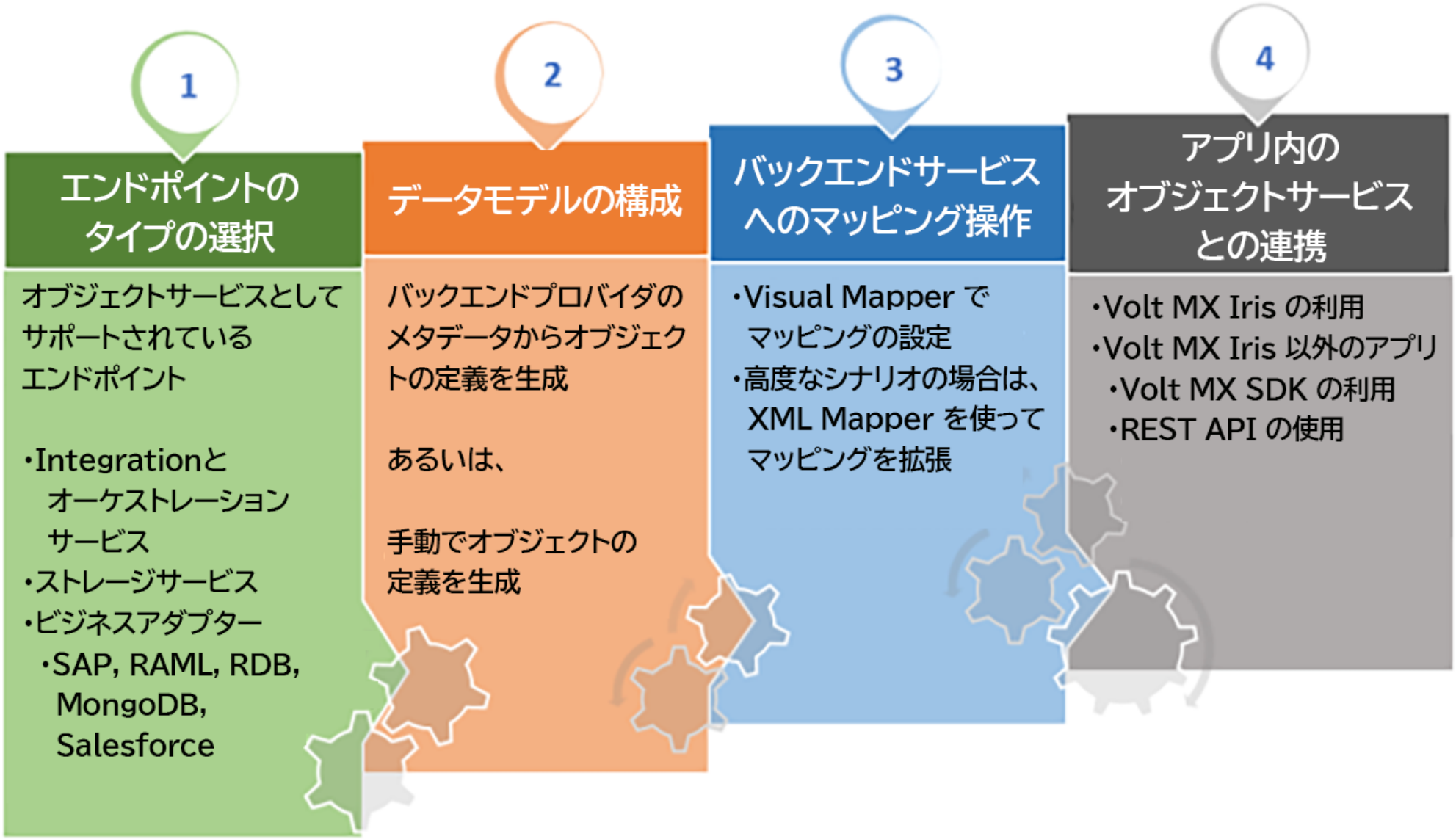
オブジェクトサービス	従来のAPIへのアプローチ
データモデルの設計とバックエンドのデータへのマッピングに重点を置く。	REST APIの設計と呼び出しに注力。
フロントエンドとバックエンドの独立した開発。	フロントエンドとバックエンドの開発間の緊密な相互作用。
アプリケーションロジックとユーザーを魅了する機能にフォーカス。	バックエンドのインタラクションの作成にエネルギーを費やす。
サービスの作成と呼び出しが容易。	より多くのコードが必要で、エラーが発生しやすい。
異なるアプリケーション間で同じデータモデルを再利用する。	パターン主導の開発を促進しない。
メンテナンスが容易、コードが自動生成される。	開発者のコーディングスタイルによって、メンテナンスの大変さが変わる。
同じデータモデルを再利用することで、リリースが3倍速くなる。長期的なコスト削減が可能。	再利用ができないので、アプリケーションごとに一からサービスを呼び出す必要がある。
統合が4倍速くなる。データにアクセスするためのコネクタの設定に集中できる。	膨大なカスタムコードが必要。

オブジェクトサービスビュー

- アプリケーションの Objects タブ、または API Management View から Object Services View に移動できます。このビューから、サービスを作成したり、アカウントに既に追加されている既存のサービスを使用したりできます。
- オブジェクトサービスを作成するには、以下の基本コンポーネントを理解する必要があります。
  - データモデル
  - バックエンドサービスプロバイダ
  - マッパー

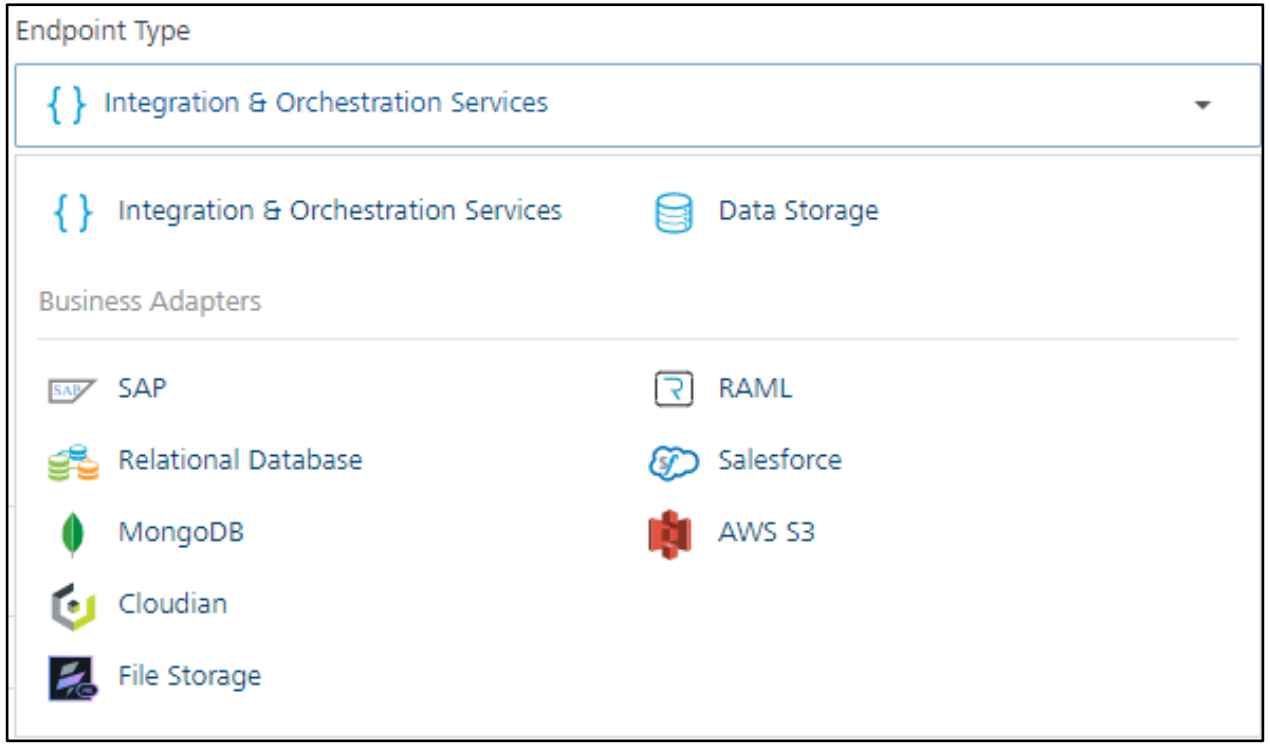


オブジェクトサービスのワークフロー



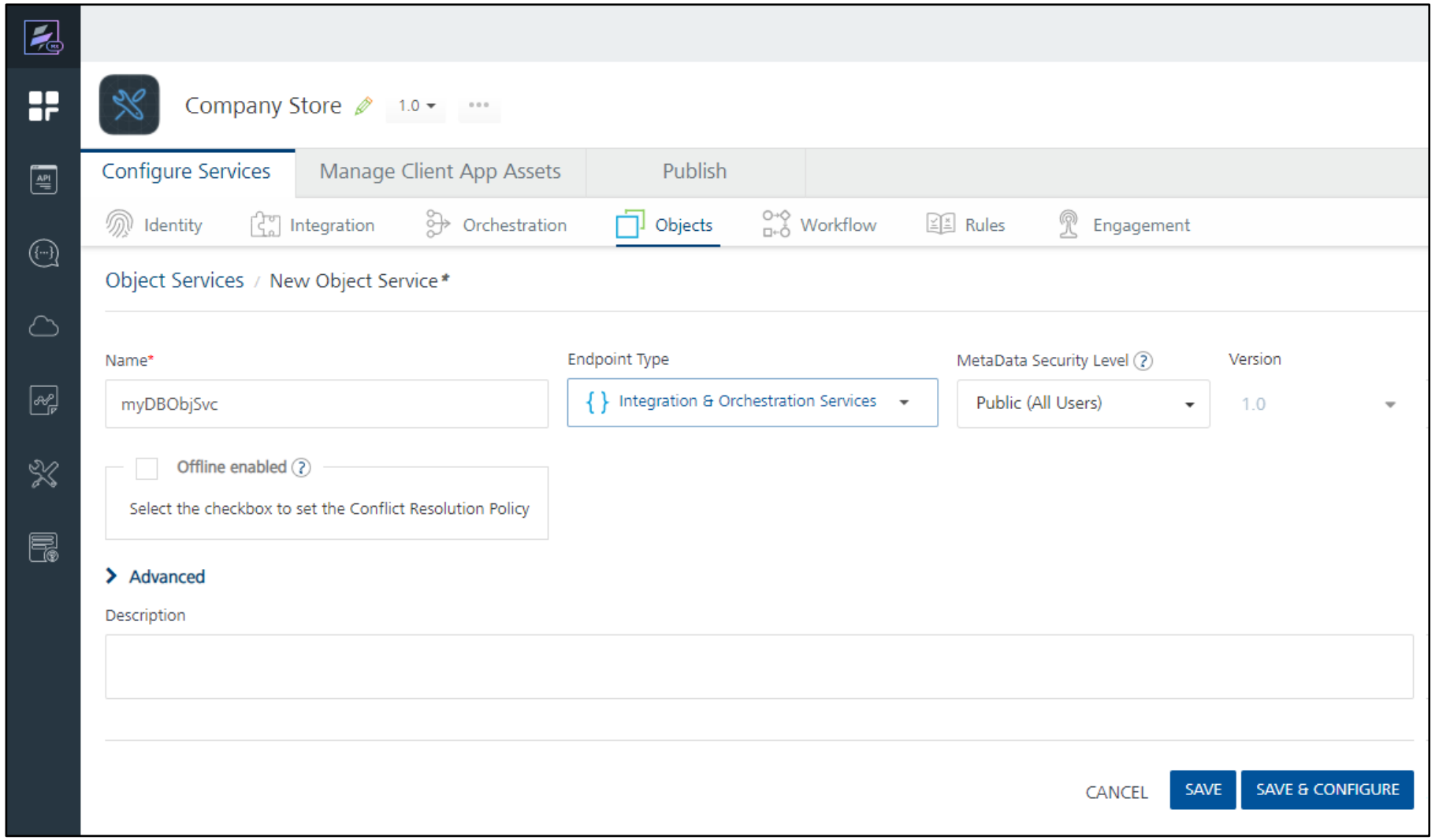
Foundry でオブジェクトを作成する

- Object Servicesは、以下に対して作成できます。
  - 統合サービス
  - ストレージ(自己定義テーブル)
  - ビジネスアダプタ
- この例では、先ほど作成したIntegration Servicesをベースにオブジェクトを作成します。



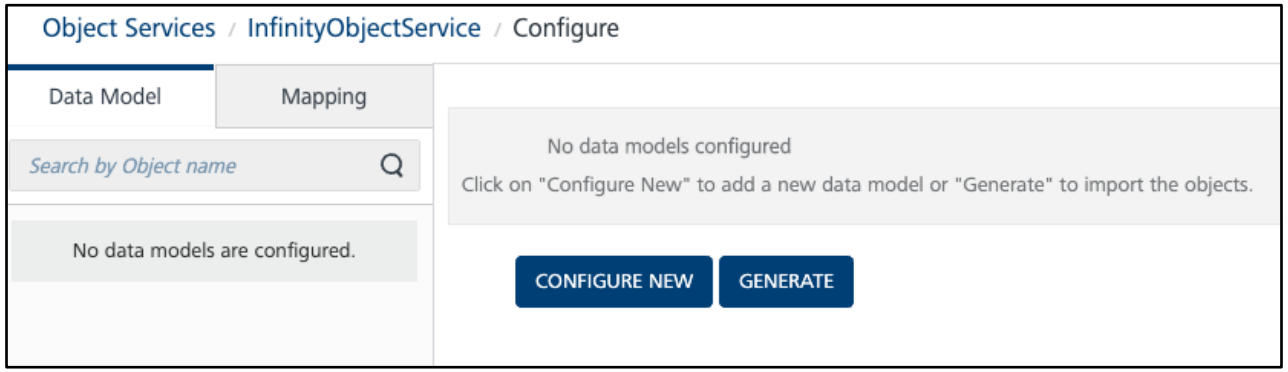
オブジェクトサービス - 統合サービス - ステップ 1

- Object Service を選択します。
- 新規作成し、Integration Services を選択します。
- この例では、セキュリティレベルを Publicに設定します。
- Save（または Save and Configure）を選択します。

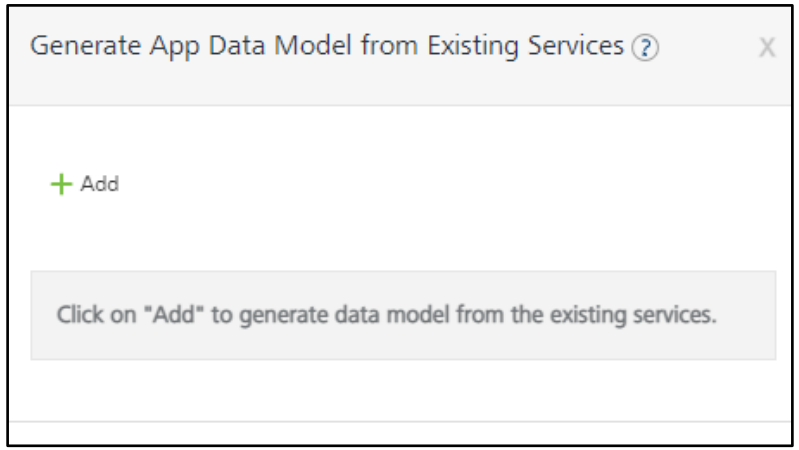


オブジェクトサービス - Integration Services - ステップ 2

- 次のステップは、データモデルを定義することです。

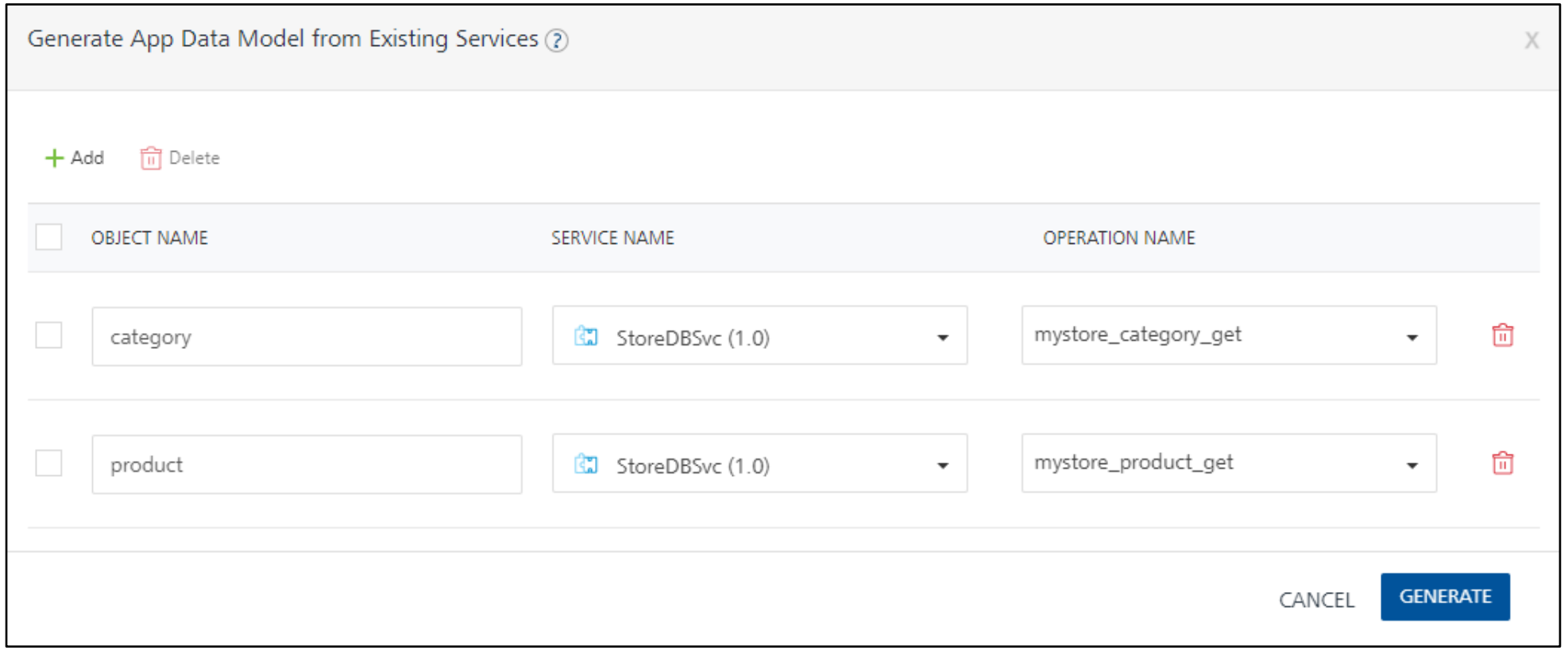


- GENERATEオプションを選択すると、プロジェクトで定義されたIntegration Servicesに基づいて、オブジェクトを定義できるようになります。



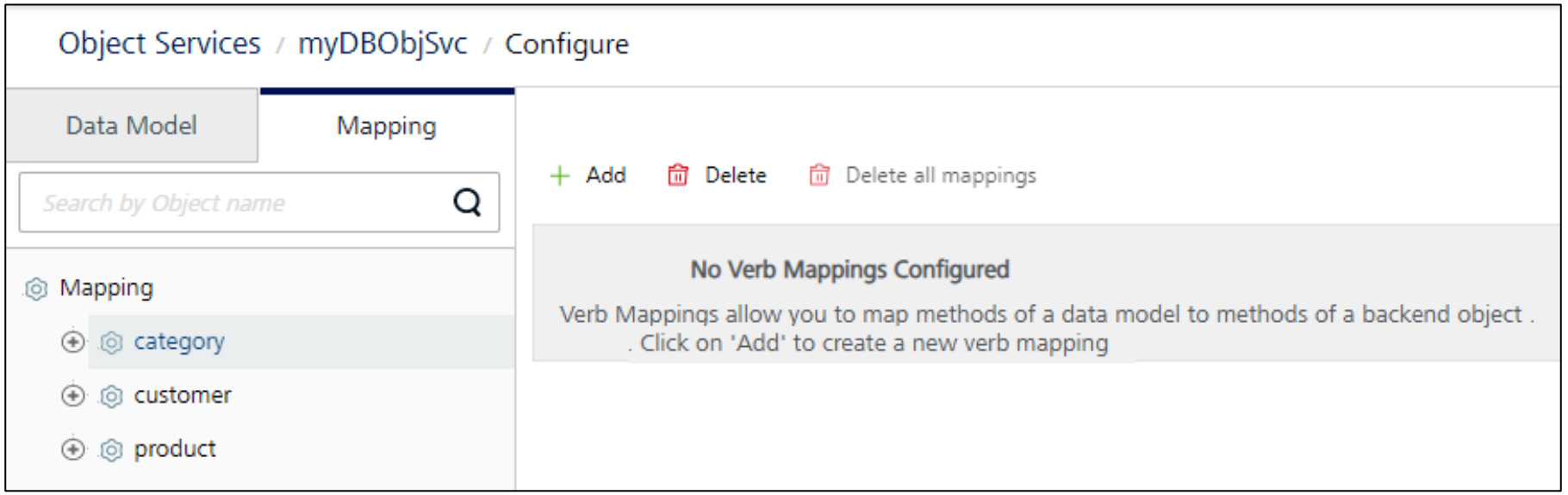
オブジェクト・サービス - Integration Services - Step 3

- このプロジェクトでは 3つのオブジェクトがあります。
  - 名前を入力し、サービスから選択し、適切な”get”サービスを選択します。
  - GENERATEを選択



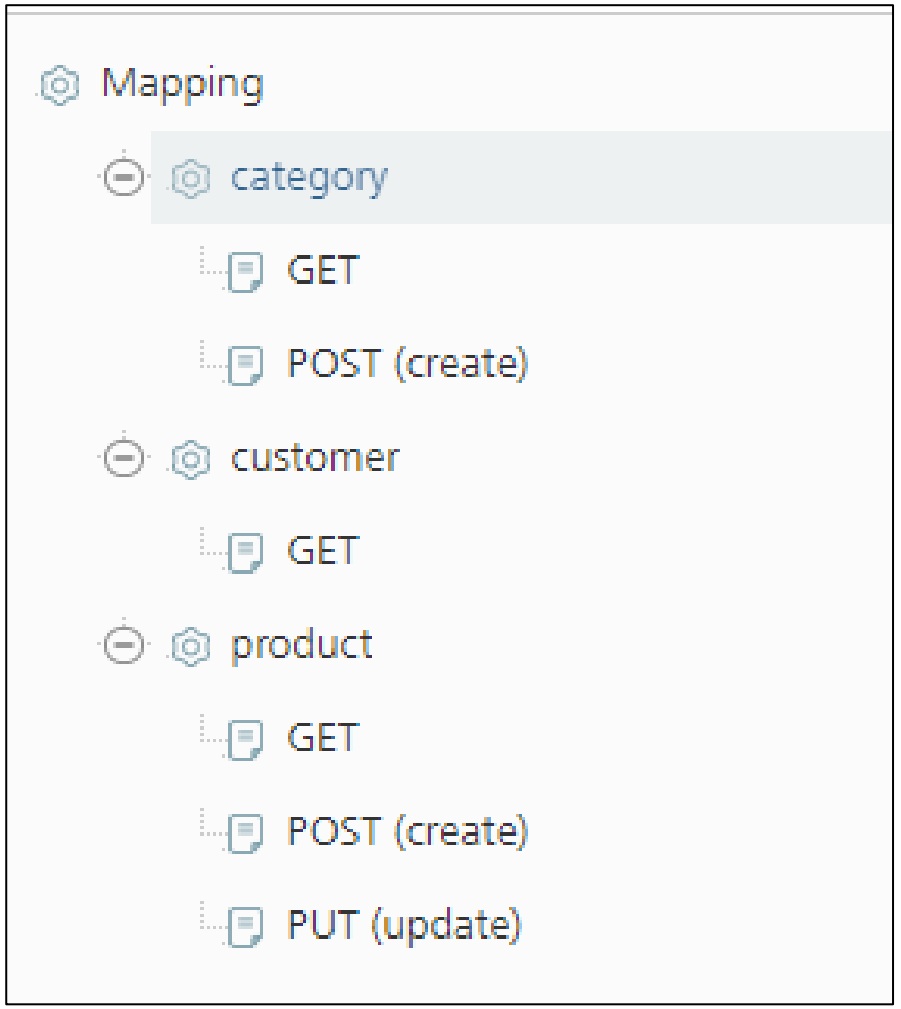
オブジェクトサービス - インテグレーションサービス - ステップ4

- 最後のステップは、”マッピング”操作です
  - オブジェクトを選択した状態で、マッピングタブを選択し、ADDを実行します。



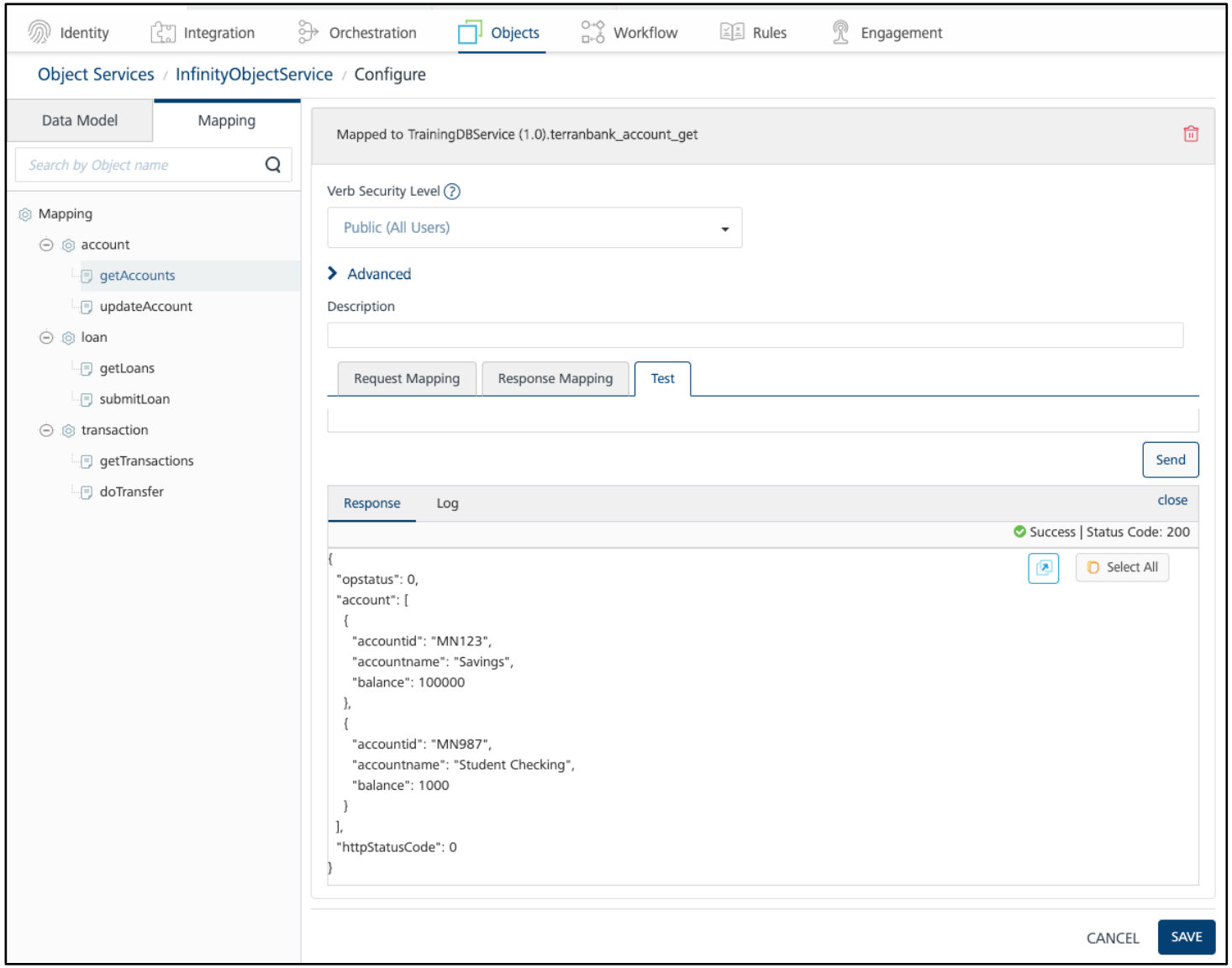
オブジェクトサービス マッピングの完了

- これで、6つのオブジェクトサービスが完成しました。
  - Category: get, create
  - Customer: get
  - Product: get, create, update



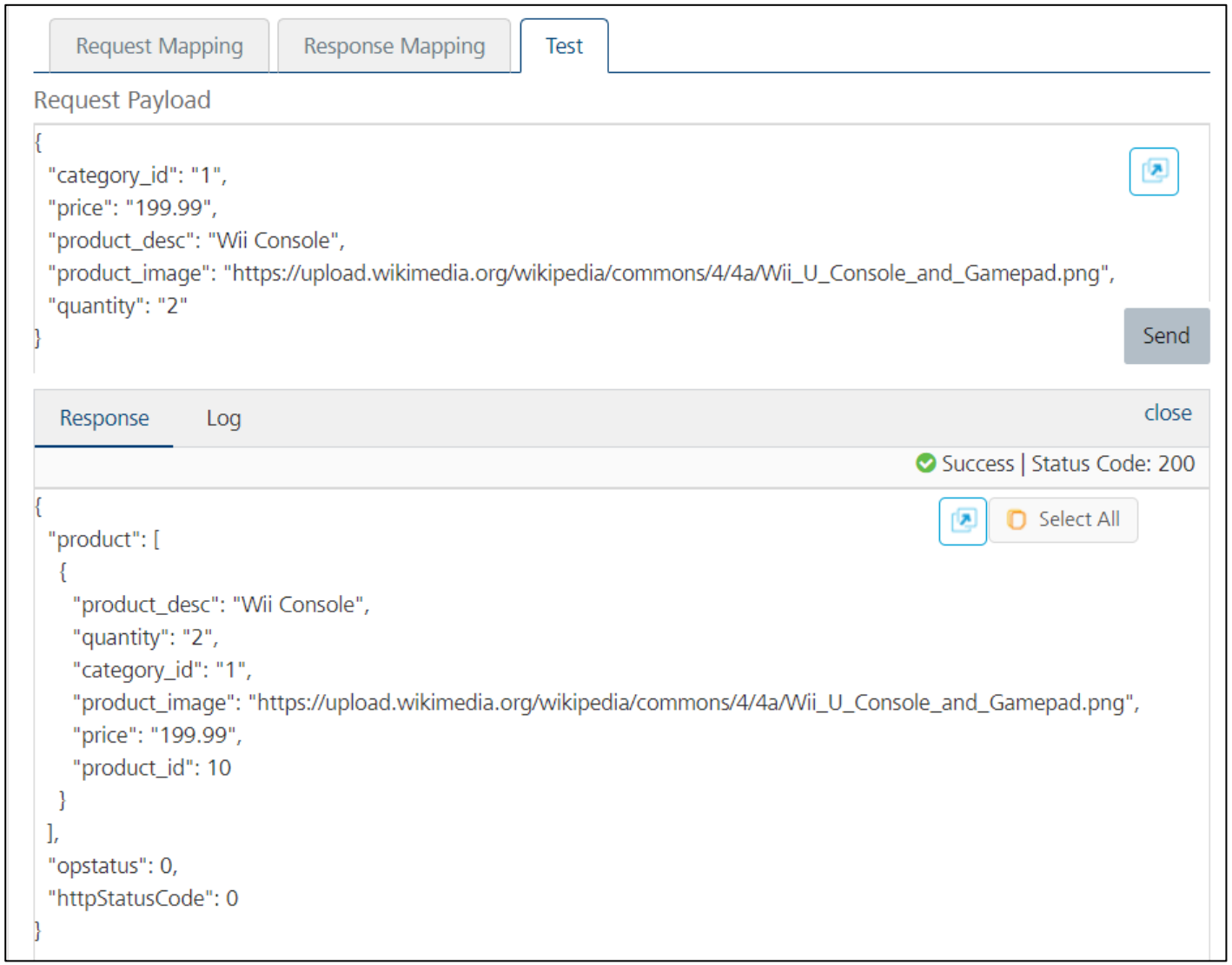
オブジェクトサービス - Foundryでgetをテストする

- いずれかの「get」オブジェクトを選択し、「Test」タブを選択します。
- Send を選択して、Foundry で呼び出しをテストします。
- 初回は必ずPublishしてください。



オブジェクトサービス - Foundry での挿入/更新のテスト

- 挿入または更新をテストするには、ペイロードを指定する必要があります。
- ヒント:挿入/更新のための有効なペイロードを取得するために取得からレコードのいずれかをコピーします。
- Createアクションは作成されたキーを返します。
- 注意: 更新時には、提供されたカラムのみが返るので、必要とするすべてのカラムを送信することを確認します。
- Foundryアプリを公開することを忘れないでください。



オブジェクトサービスの呼び出し

- オブジェクトサービスを呼び出す方法について、ドキュメントを参照して確認してみましょう。
  - https://opensource.hcltechsw.com/volt-mx-docs/docs/documentation/Foundry/voltmx foundry user guide/Content/ObjectsAPIReference/Objects API Reference.html
- すべてのオブジェクトに customVerb メソッド を使用しています。
- このメソッドのシグネチャーは以下のとおりです。
  - customVerb(verbName, options, successCallback, failureCallback);
- このメソッドをどのように呼び出すか見てみましょう...

- 参考
  - OnlineObjectService Class
    - https://docs.kony.com/konylibrary/konyfabric/kony_fabric_user_guide/Content/ObjectsAPIReference/OnlineObjectService_Class.htm

オブジェクト取得サービスを呼び出すサンプルコード

```
1.  var objSvc = voltmx.sdk.getCurrentInstance().getObjectService("serviceName", {
2.     "access":"online"});
3.  var dataObject = new voltmx.sdk.dto.DataObject("ObjectName");
4.  dataObject.addField("field1","value1");
5.  dataObject.addField("primaryKeyField","value");
6.  var options = {"dataObject": dataObject };
7.  objSvc.customVerb("verbName", options,
8.      function(response) {
9.          voltmx.print("Custom operation performed:"+ JSON.stringify(response));
10.     },
11.     function(error) {
12.         voltmx.print("Error in custom operation:"+ JSON.stringify(error));
13.     });
```

getProductsを呼び出すサンプルコード

```
1.  getProducts : function ()
2.  {
3.    var objSvc = voltmx.sdk.getCurrentInstance().getObjectService("DBObjectService",
        {"access":"online"});
4.    var dataObject = new voltmx.sdk.dto.DataObject("product");
5.    var options = {"dataObject": dataObject};
6.    objSvc.customVerb("GetProducts", options, successCB.bind(this),failureCB.bind(this));
7.    function successCB (response)
8.    {
9.      voltmx.print("#### Custom operation performed:"+ JSON.stringify(response));
10.   }
11.   function failureCB (error)
12.   {
13.     voltmx.print("#### Error in custom operation:"+ JSON.stringify(error));
14.   }
15. }
```

練習: オブジェクトの使用

- クライアントアプリをアップデートしてみよう
- サービスの代わりにオブジェクトの呼び出しを練習して使ってみよう

- 選択したカテゴリに新しい製品を作成する

www.hcltech.com

$10 BILLION | 159,000+ IDEAPRENEURS | 50 COUNTRIES