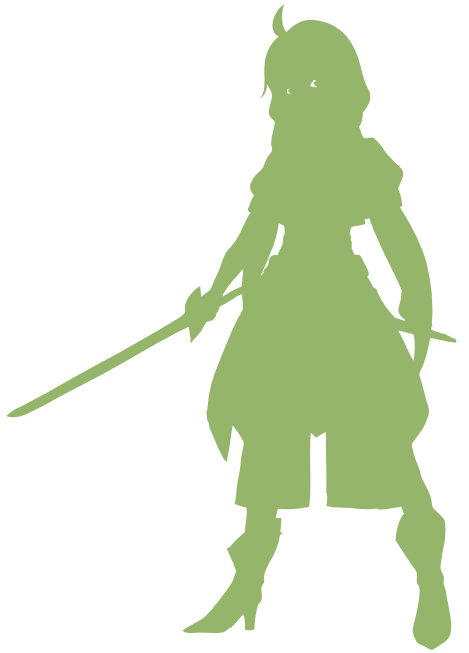
既存のC/Sアプリケーションを Sencha を使ってWeb対応する

第35回 エンバカデロ・デベロッパーキャンプ

エンバカデロ・テクノロジーズ
セールスコンサルタント
井之上 和弘

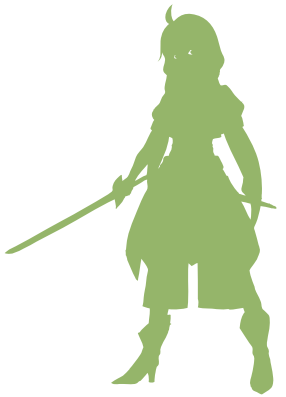


embarcadero®
DEVELOPER CAMP

セッションサマリ

- DelphiやC++Builderなどで開発した既存のC/Sシステムを「Web対応してほしい」という要望が出ていませんか？
- 新たに加わったSenchaが有力な解決策となります。Gridなどを用いた既存のアプリはどのようにWeb対応できるのか。
- また、現状のC/Sアプリは併用できるのか。
- このような課題を解決しつつ、既存のDelphi/C++Builderのアプリケーションをweb拡張する方法をご案内します。

なぜWeb対応するのか？



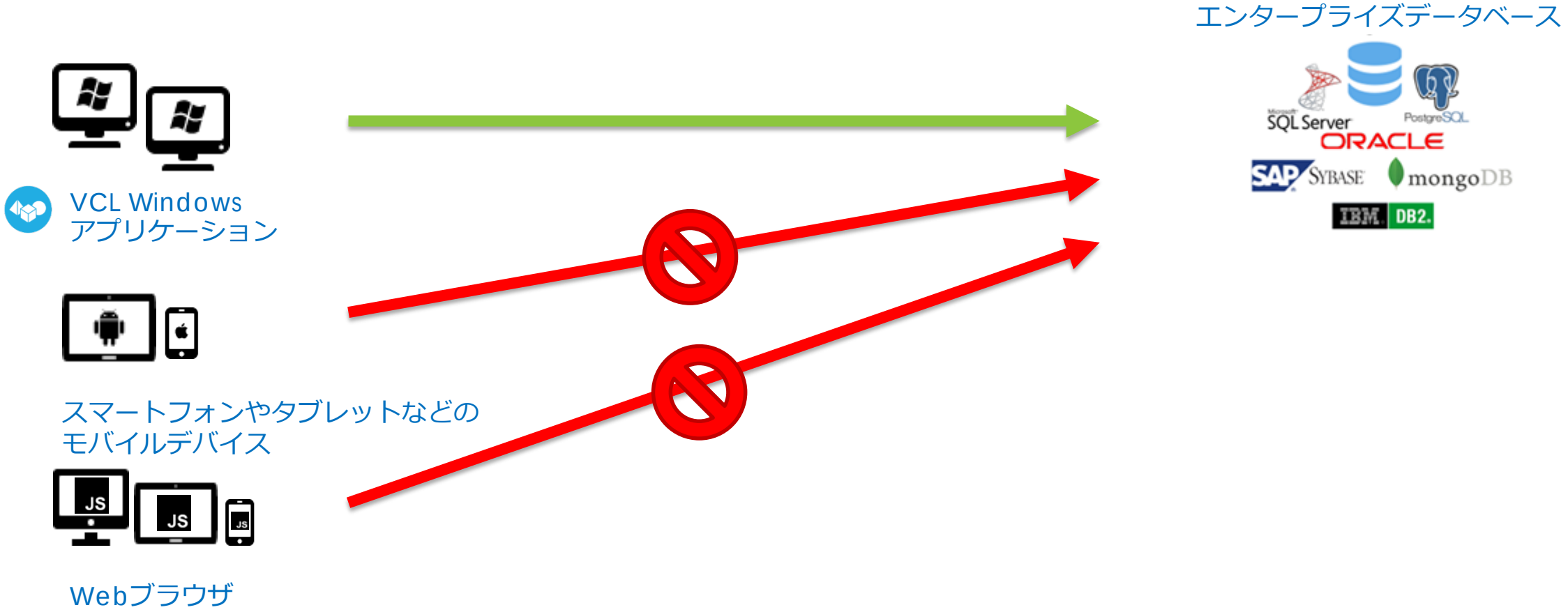
embarcadero®
DEVELOPER CAMP

既存のC/Sは閉じた世界で利用

VCL Windows
アプリケーション

エンタープライズデータベース

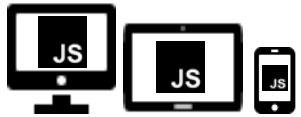
データベースはモバイル端末やブラウザでは利用できない



DB以外のデータソースも登場、どう利用するか？

VCL Windows
アプリケーション

スマートフォンやタブレットなどの
モバイルデバイス



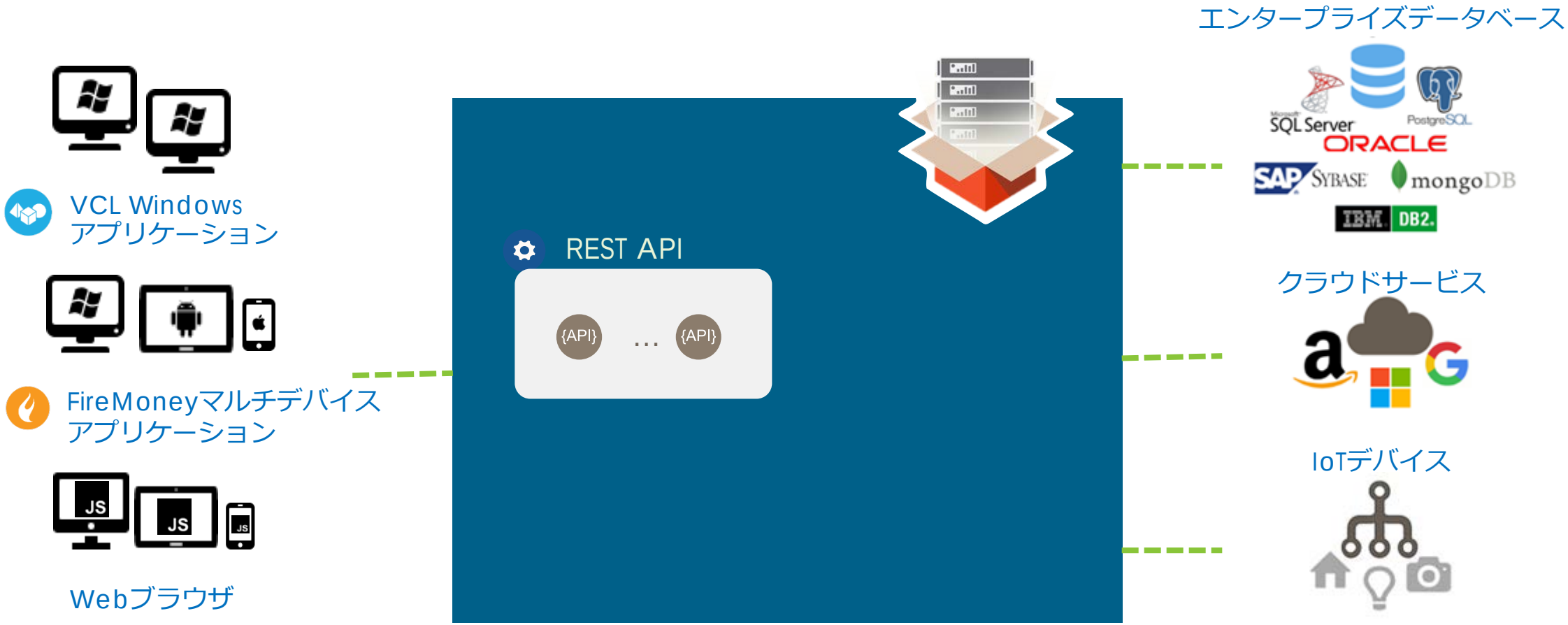
Webブラウザ

エンタープライズデータベース

クラウドサービス

IoTデバイス

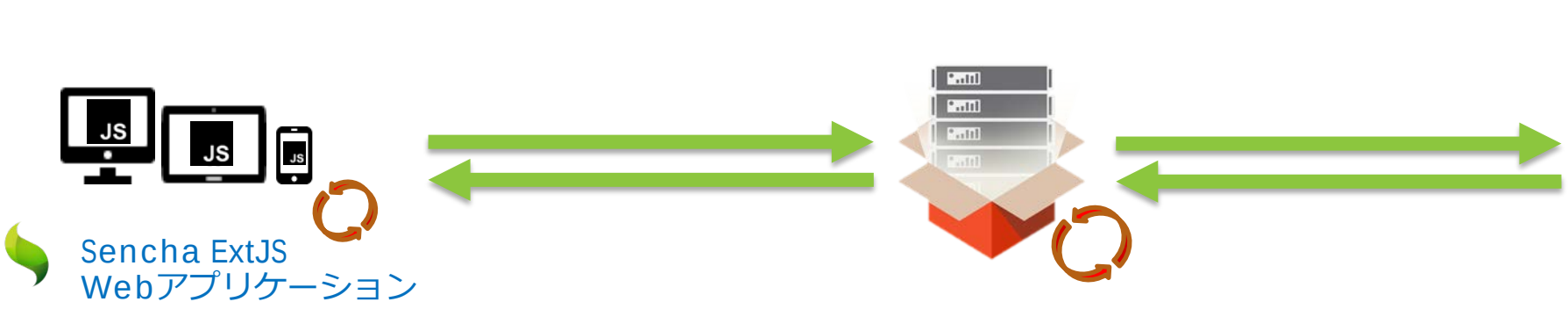
Web対応の中間サーバでつながるようになる



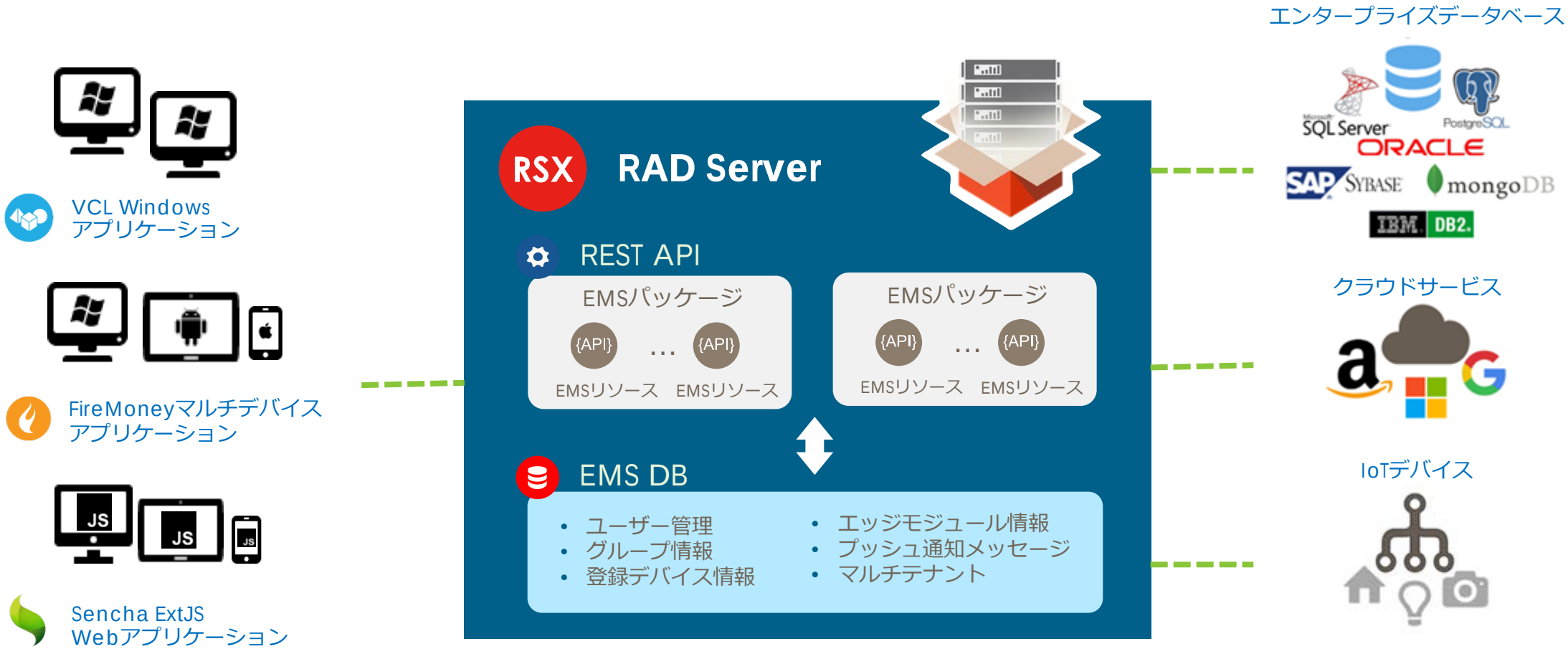
Web対応するには何が必要か？

- C/Sでは、基本的にクライアント側で処理を行う

- Web対応した場合はどこで処理を行うのか？
  - 古い方法では、中間サーバが処理した結果をHTMLで返す
  - 今の方法は、中間サーバはデータだけを返して、JavaScript で表示

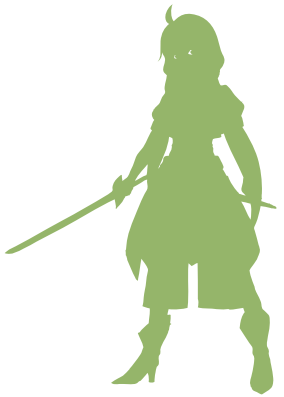


RAD Server と Sencha による Web 対応



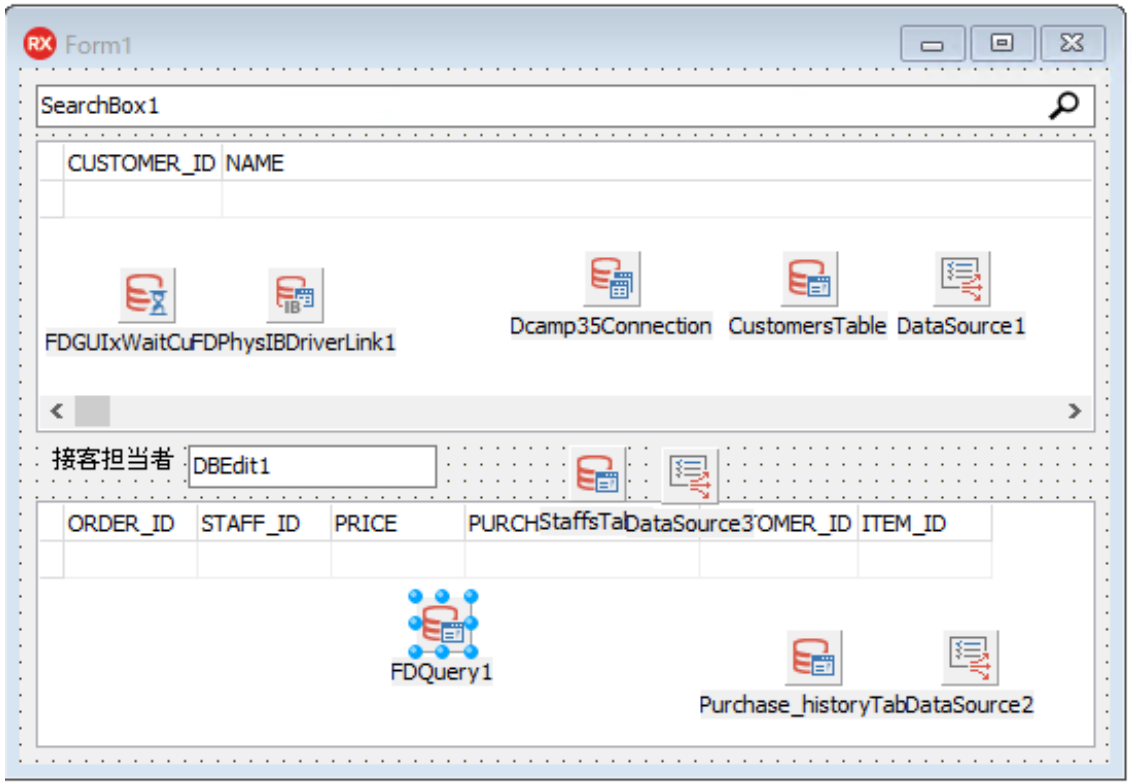
まずは実際に作るアプリのデモ

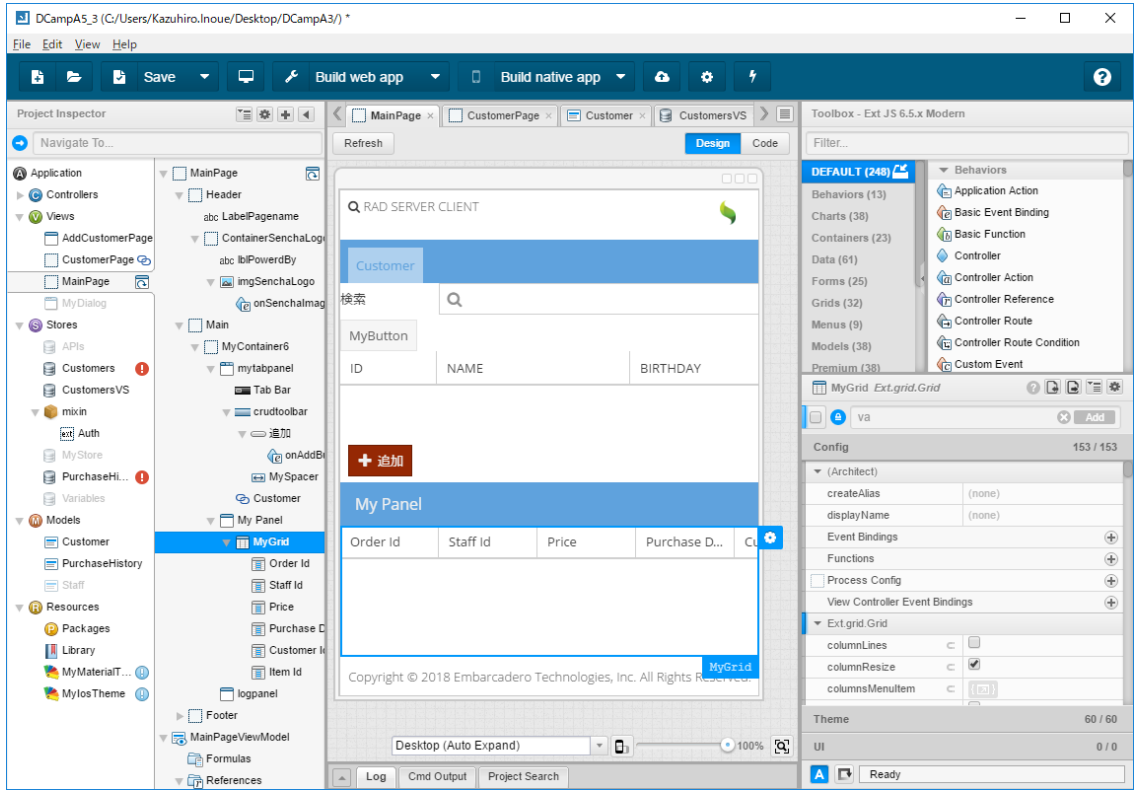
これから作成していくアプリをご覧いただき、こういうものを作る、ということのイメージを持っていただきます



embarcadero DEVELOPER CAMP

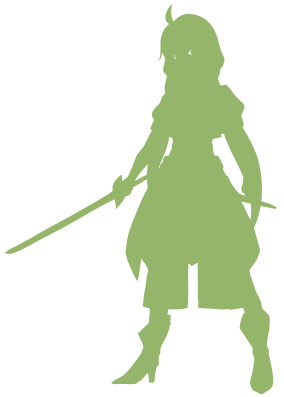
DEMO：単体のC/S アプリ、RAD Server と組み合わせる Sencha アプリの紹介





C/Sアプリをリファクタリングで Web対応する

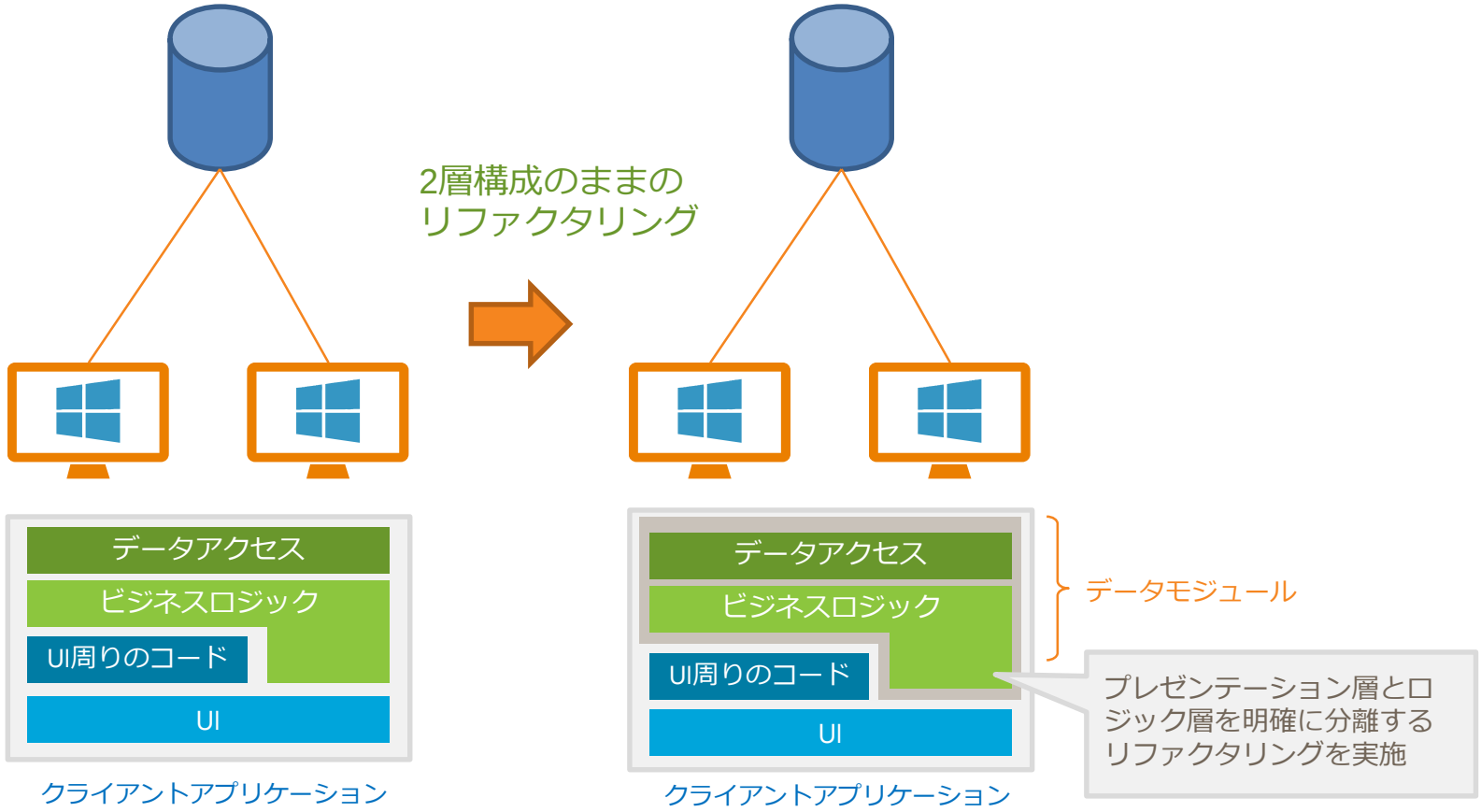
では、Sencha 向けのWeb対応を、既存のC/Sアプリからの
リファクタリングで実装してみましょう



embarcadero®
DEVELOPER CAMP

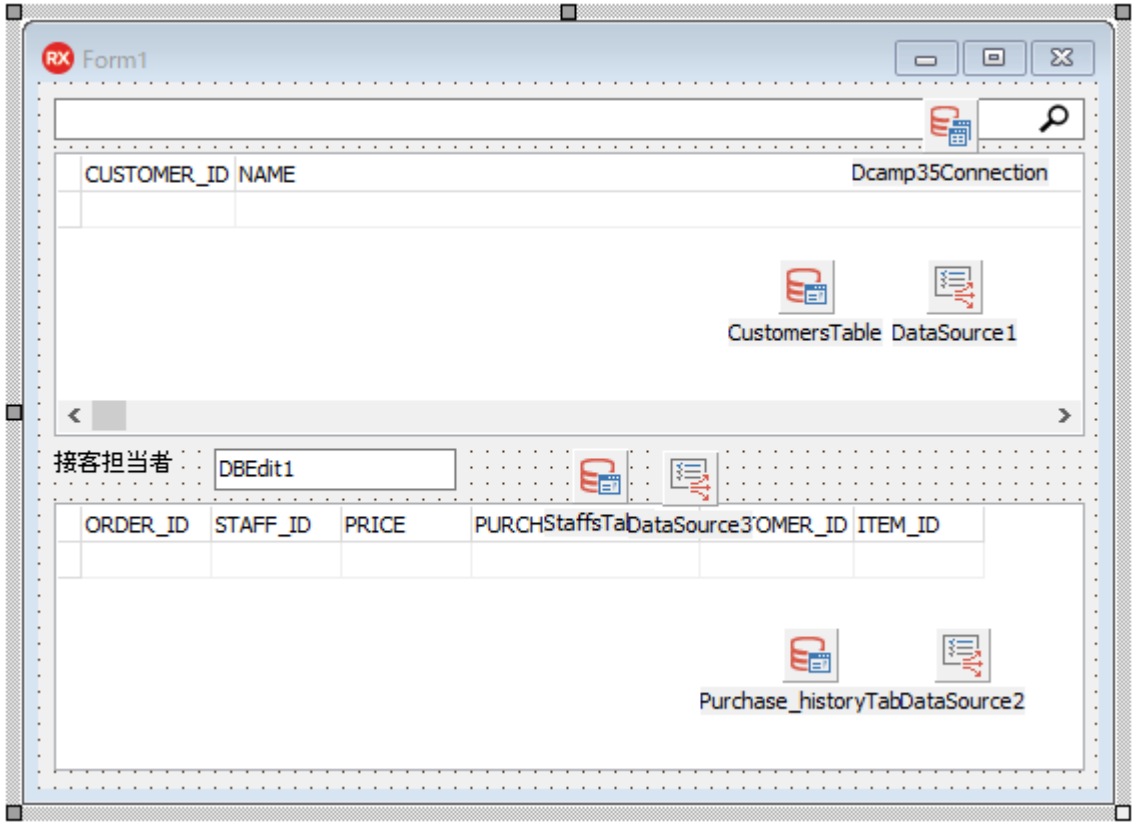
従来アプリケーションのWeb対応（前半：リファクタリング）

- 2層アプリケーションは、ロジック層の分離を行う



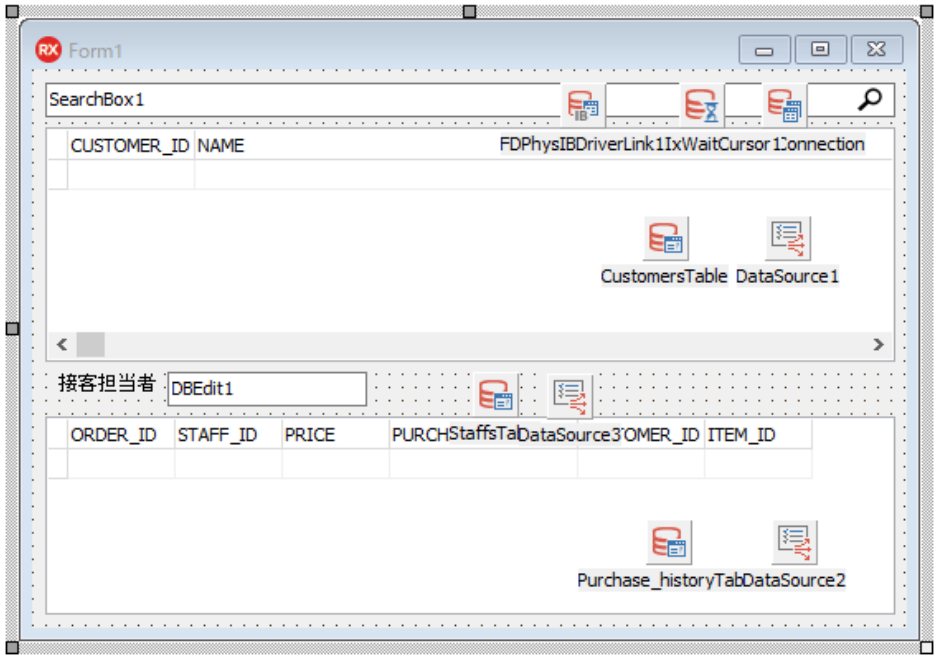
DEMO：データモジュール利用に変える

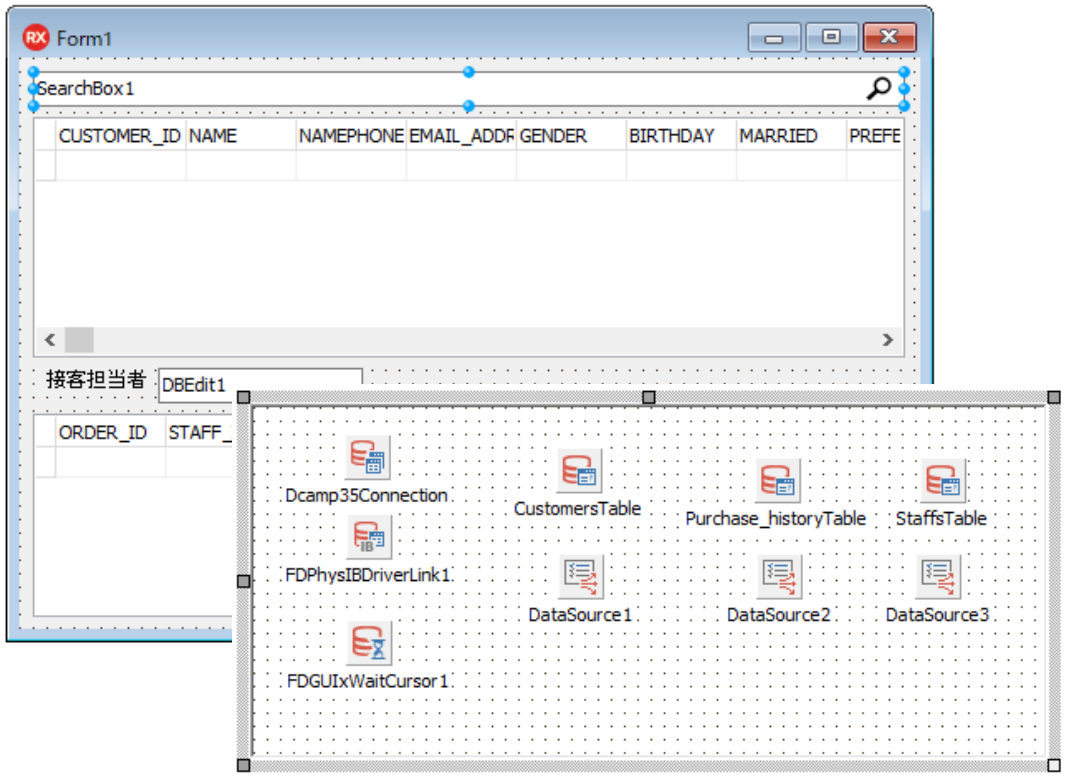
- フォーム上にビジュアルコンポーネントと非ビジュアルコンポーネントが混在したアプリを、データモジュールを用いる形に変更



確認：既存アプリケーションのリファクタリング

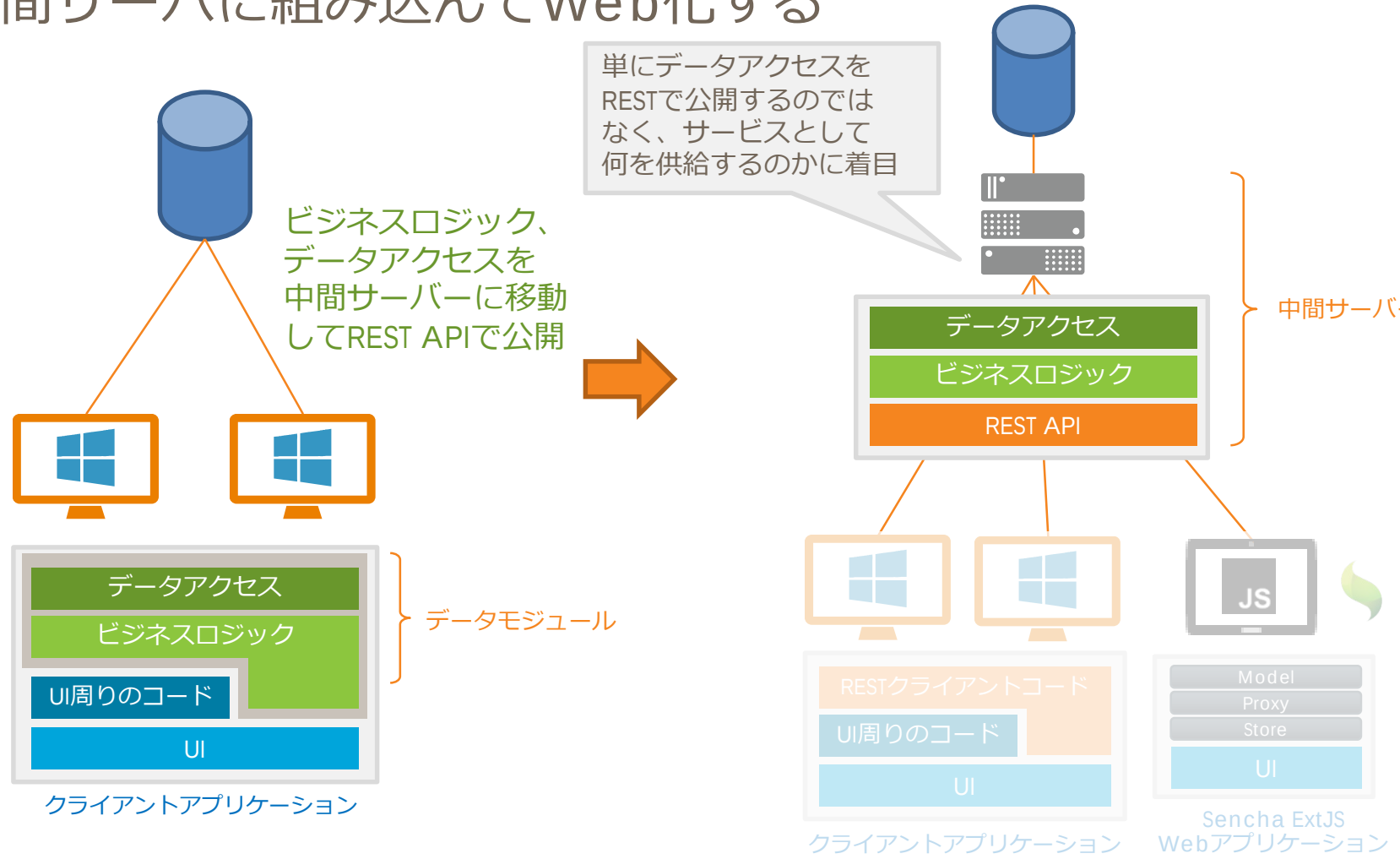
- データアクセスとビジネスロジックはデータモジュールに分離



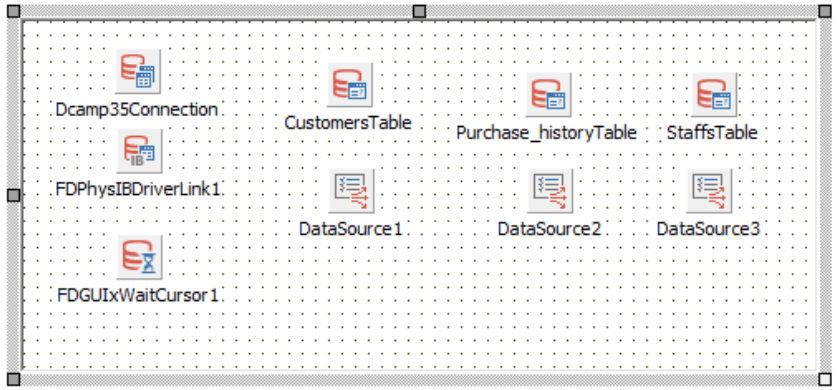


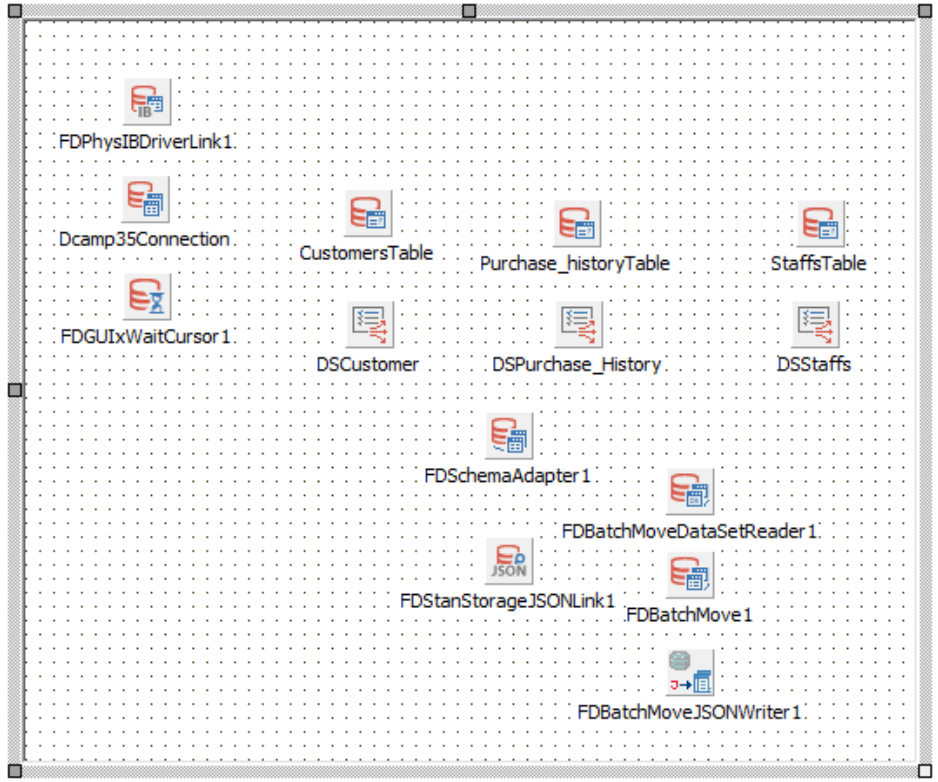
従来アプリケーションのWeb対応（後半：3層化）

- 分離したロジックを中間サーバに組み込んでWeb化する



確認：RAD Server プロジェクトにデータモジュールを取り込む





JSONはクライアントが使いやすい形式で作る

- 普通のJSON （ FireDAC 内部形式ではないもの ）を受け渡す
  - さまざまな Web クライアントで汎用的に使える
  - TFDBatchMove でカンタンに扱える（10.2.2 以降）


```
{
    customer:
    [
        {
            CUSTOMER_ID: 1,
            NAME: "神野 天音",
            NAMEPHONETIC: "かみの あまね",
            EMAIL_ADDRESS: "kamino_amane@example.com",
            GENDER: "女",
            BIRTHDAY: "1995-02-14T00:00:00.000+09:00",
            MARRIED: "未婚",
            PREFECTURE: "東京都",
            PHONE_NUMBER: "03-4540-4148",
            MOBILEPHONE: "03-4540-4148"
        }
    ],
    total: 1
}
```

DEMO：RAD Server のインタフェース実装

- リファクタリングによって分離されたデータモジュールを RAD Serverで用いる
- REST API は RAD Server の基本機能で実装

大量データを Sencha の VirtualStore で扱えるようにする

- VirtualStore はデータをページ単位で自動的に取得する
  - 以下の２つのクエリパラメータを用いる

count	1回のリクエストで取得したいレコード数
page	取得するページ番号（初期値 = 1）

- VirtualStore はトータルのレコード数も使用する
  - JSON データの { “total”:nn } を参照して操作時のページ数を算出

- サーバ側がこれらのデータを返せば VritualStore が利用できる

EMSServer.ini での認証設定例

- 設定例

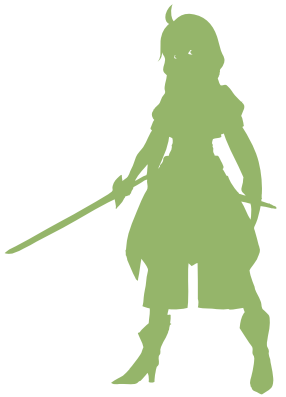
```
[Server.Authorization]
; 標準のリソースを認証必須にする
Version={"public": false}
Users={"public": false}
Groups={"public": false}
Installations={"public": false}
Push={"public": false}
Edgemodules={"public": false}

; Users.LoginUser は認証処理に使用するため、未ログイン状態のアクセスを許可する
Users.LoginUser={"public": true}

; DCamp35 リソースは group = users に所属する認証済みユーザだけがアクセスできる
DCamp35={"public": false, "groups": ["users"]}
```

クライアント側の実装

できあがったサーバに接続するクライアントを作ります



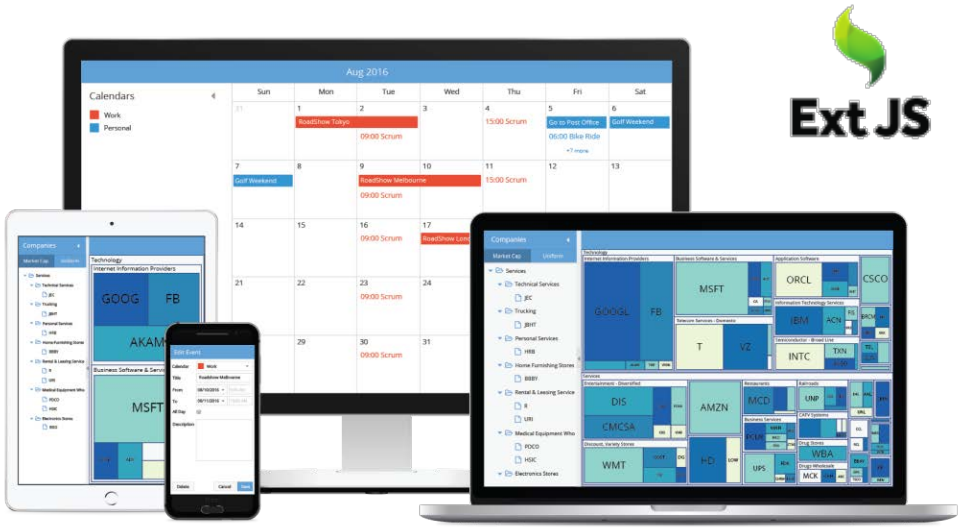
embarcadero DEVELOPER CAMP

今回使用するSenchaとは？

- エンタープライズWebアプリケーションを迅速に構築

Ext JS

- JavaScriptによる
  クロスプラットフォームアプリ開発
- カスタマイズ可能なUI部品
- 幅広いブラウザをサポート



Sencha Architect

- ドラッグ＆ドロップによる
  HTML5アプリケーション構築をサポート

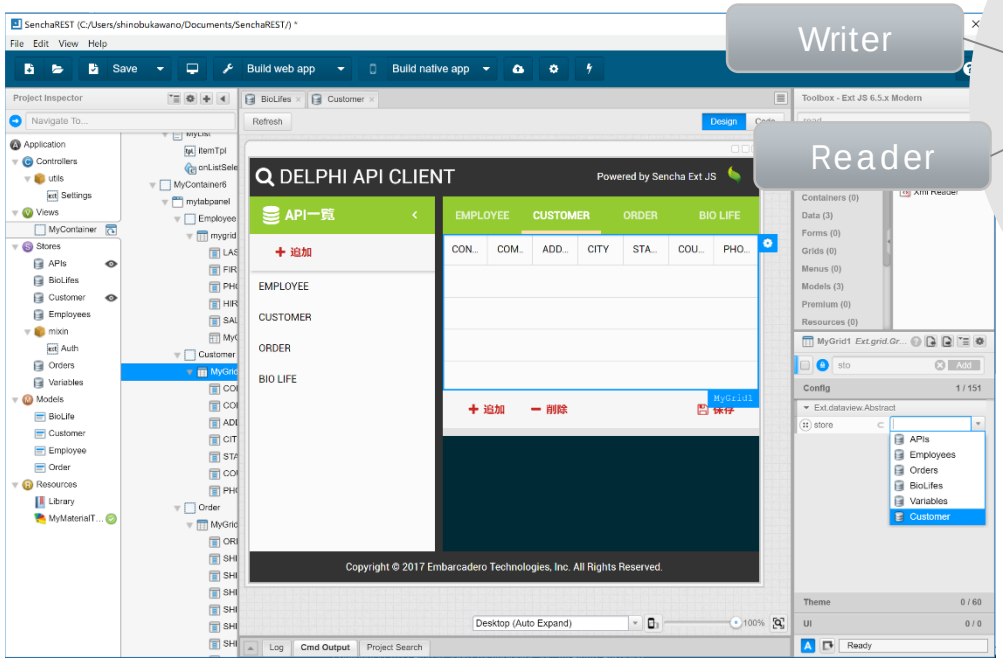
Sencha Test

- ExtJSアプリの単体・機能テストの最善のソリューション

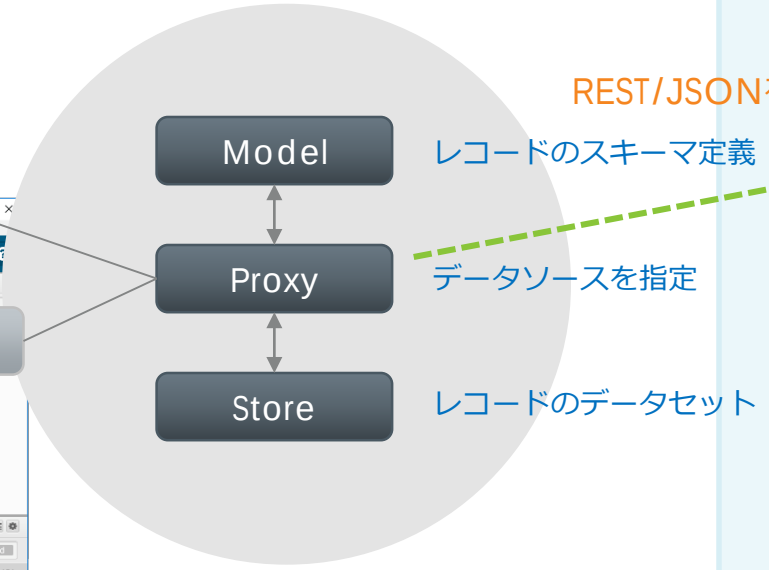
Sencha の典型的な開発スタイル

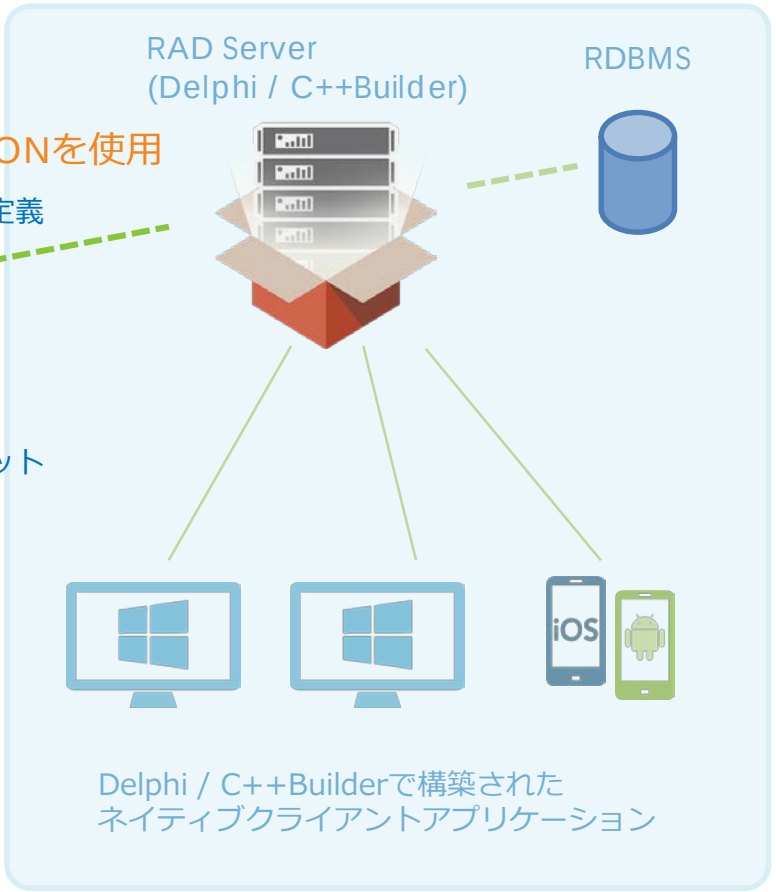
- Architect で UI を設計
  - Ext JS コンポーネントを使用
- MVC や MVVM による実装
  - アプリをデータ構造や手続き、画面に分割して設計実装する手法
  - MVC = **M**odel, **V**iew, **C**ontroller
  - MVVM = **M**odel, **V**iew, **V**iew**M**odel
- REST API に接続し、JSONデータを利用
  - 多くの Web API で標準的な方法

Senchaでデータ処理に使用する機能



Sencha ArchitectでWebアプリケーションを構築





既存のDelphi / C++Builderアプリケーション資産

開発の手順

- ツールキットを選ぶ (Modern, Classic)
- 取り扱うデータのスキーマを設定する (Model)
- データの保存先 (Store) 、データソース（Proxy）や
  受け渡されるデータの Reader/Writer を設定する
- アプリの画面を作る (View)
- アプリで発生するイベントに対応したコードを実装する
  (Contoller)

ツールキット (Modern, Classic) を選ぶ

- スマートフォン対応が必要な場合は Modern
- Internet Explorer 8 -10 の対応が必要なら Classic

	Internet Explorer 8-10	Internet Explorer 11	Edge, Chrome, FireFox,, Opera	タブレット	スマートフォン
Ext JS Modern	×	○	○	○	○
Ext JS Classic	○	○	○	○	×

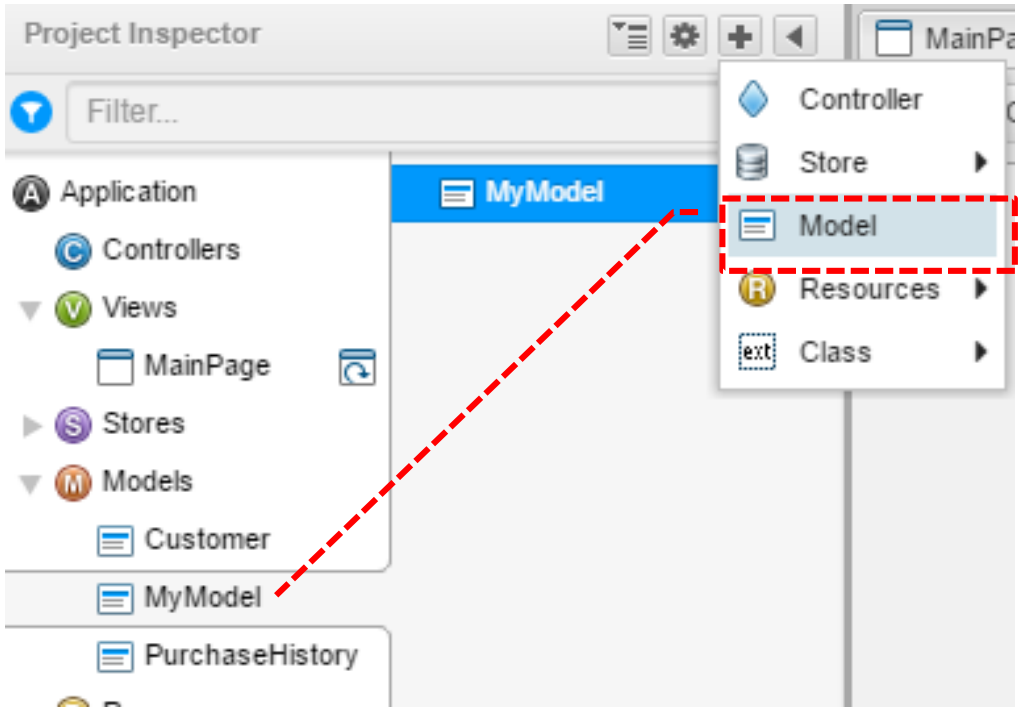
* Classic のページはスマートフォンもPC向けの表示となり、操作性に問題が出る
* Modern のページは Internet Explorer では正しく表示できない場合がある

- 今回は Modern を使います

DEMO：Modelを作成する

確認：Model を作成する

- プロジェクトインスペクタで Model を追加

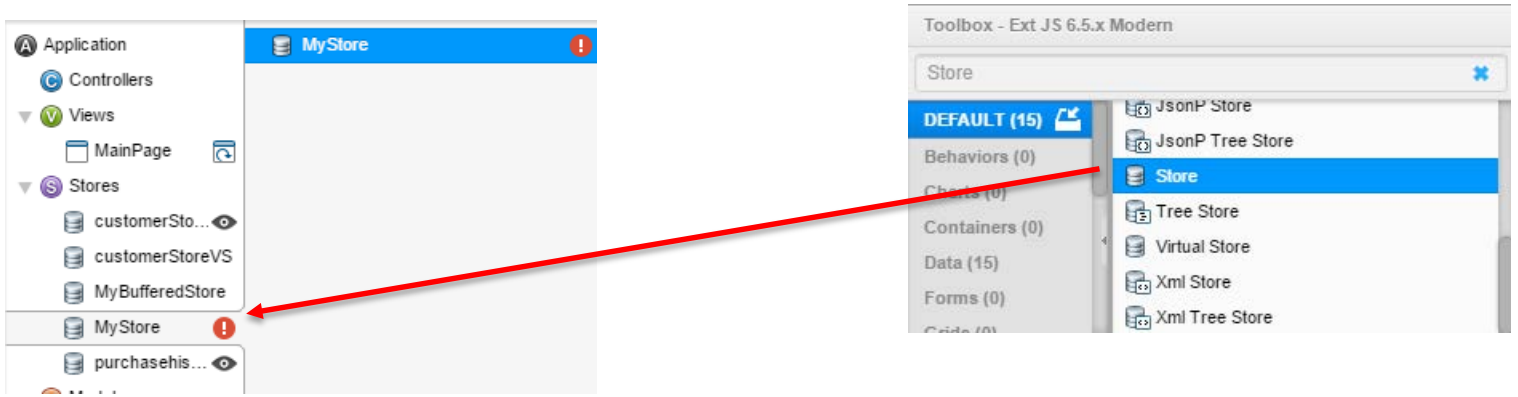


- 名前とフィールドを設定

DEMO：Storeを作成する

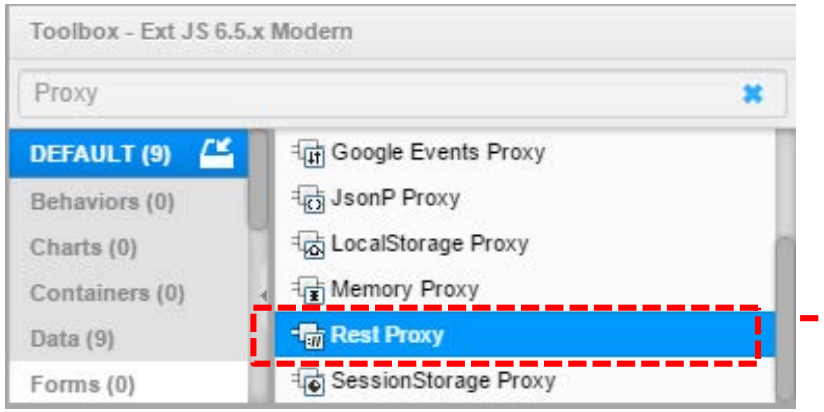
確認：Store を作成する

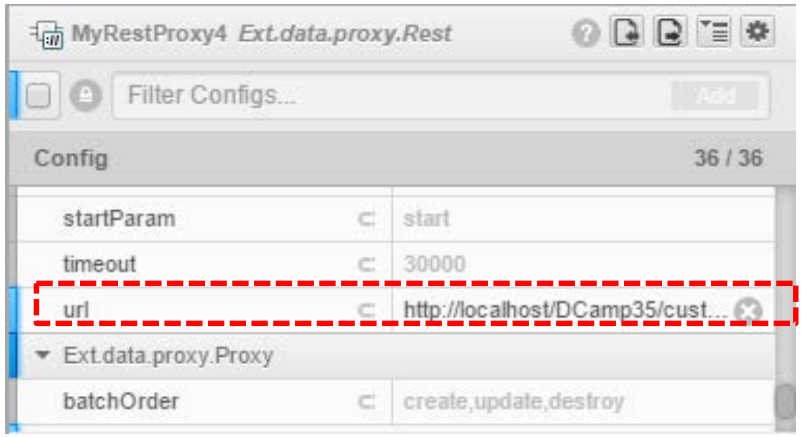
- ツールボックスから Store を配置する

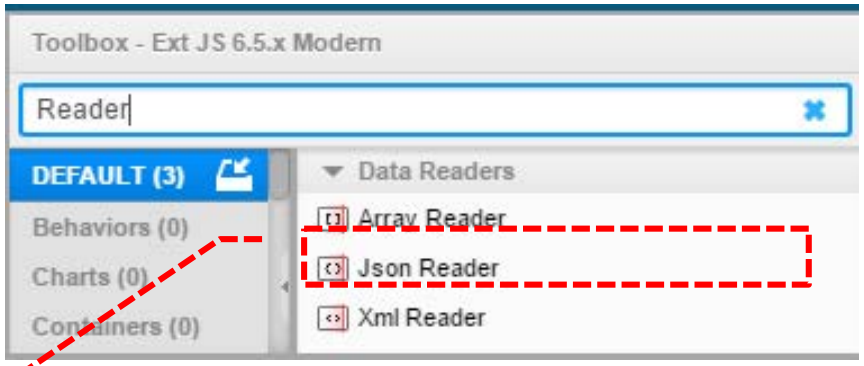


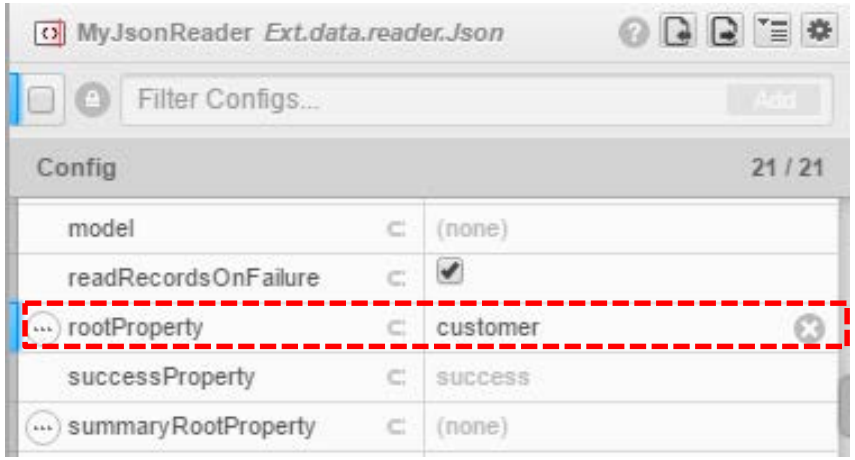
- 大量のデータを扱う場合は、VirtualStore を選ぶ
  - VritualStore ではデータをオンデマンドで必要な分だけ取得できる
  - サーバ側がページや件数のクエリに対応していれば利用可能

確認：Store に Proxy, Reader を配置、設定する





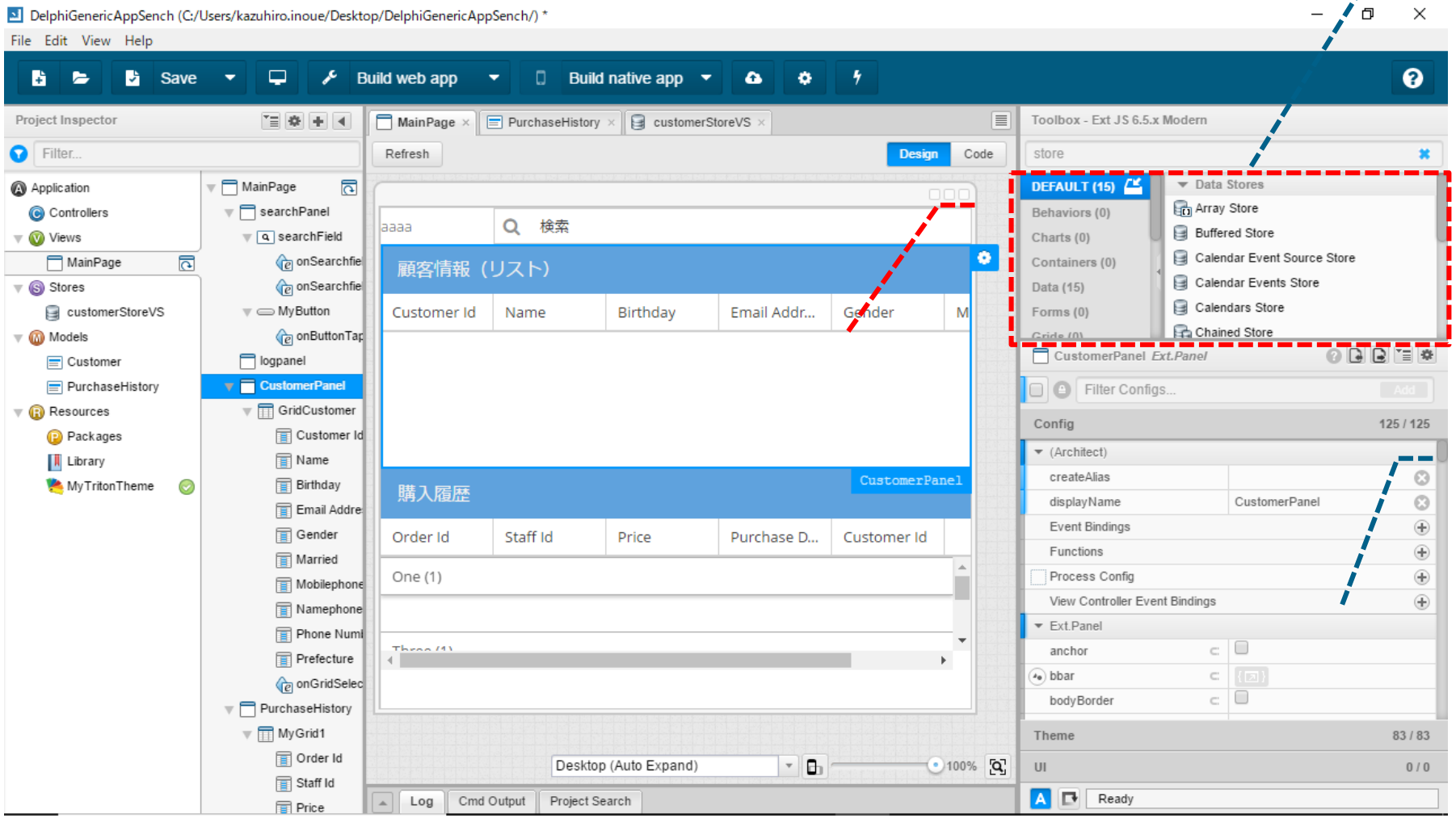




DEMO:画面を作成する

- Grid を配置し、作成した Store, Model でデータ表示を行います

確認：Grid を配置する



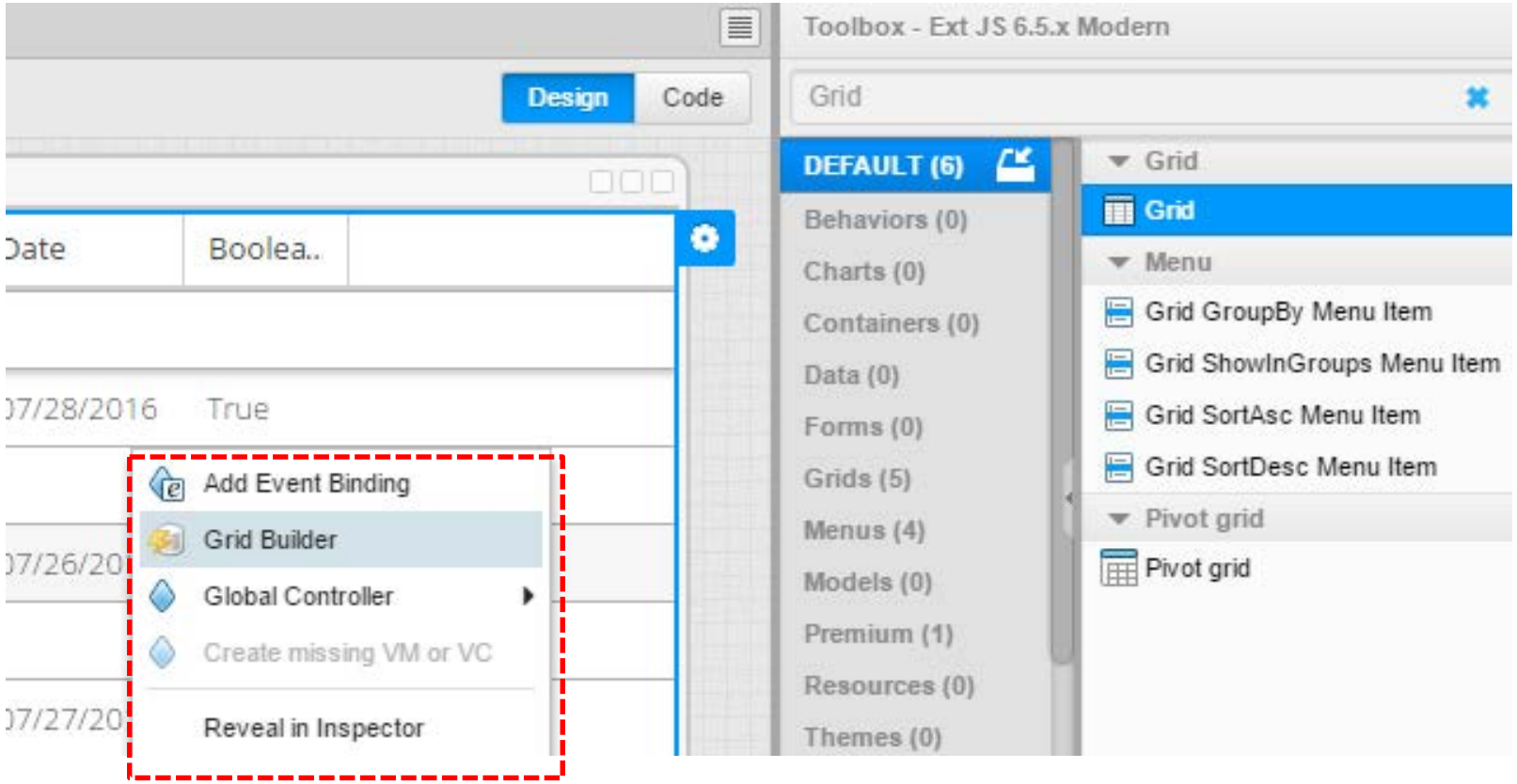
コンポーネントを
配置して画面を作る

GridにStoreを設定

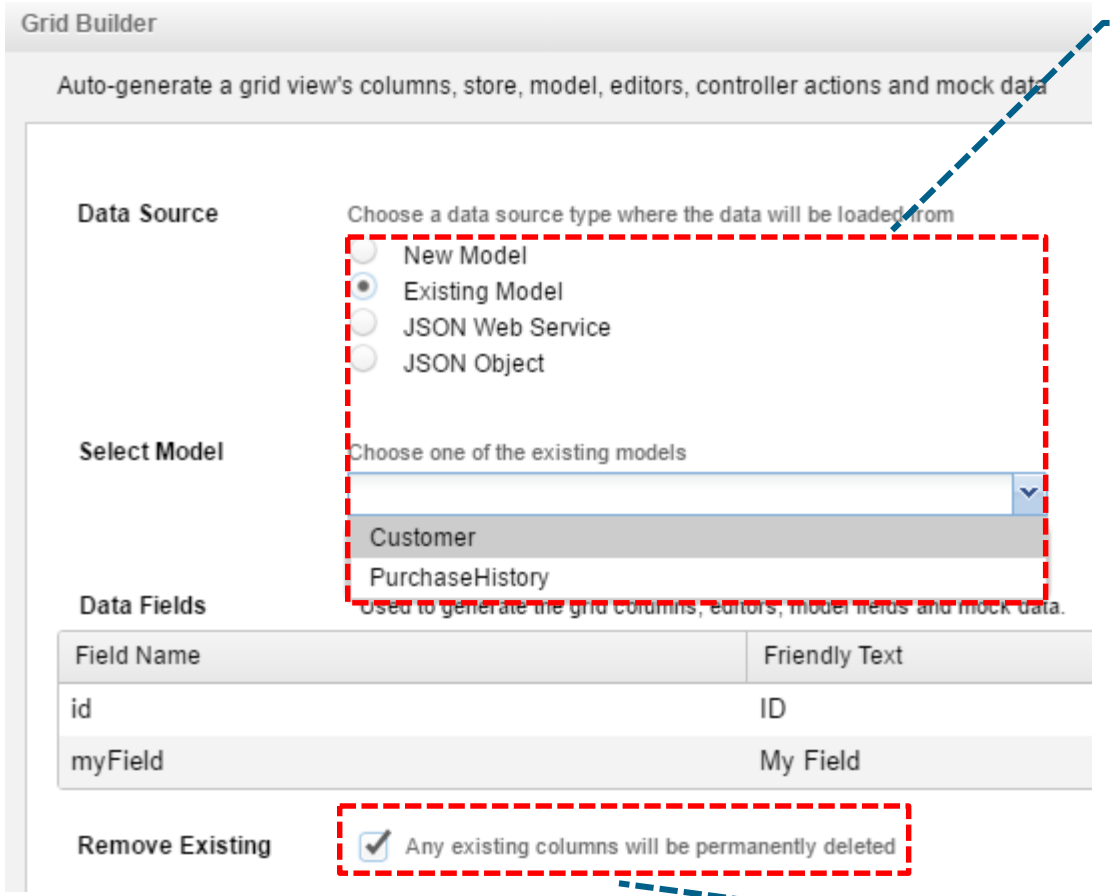
カラムを追加して
カラム名や Model を
設定

確認：Grid Builder を使って一括割当

- Views の Grid で右クリック、 Grid Builder を選ぶ



確認：データソースを選んでGrid Builderを実行

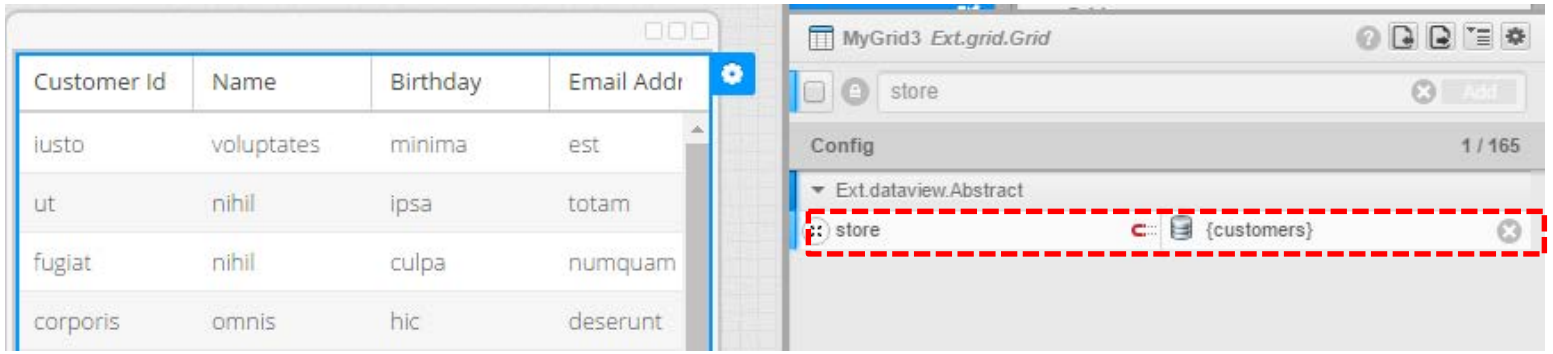


作成ずみのモデルを選ぶ

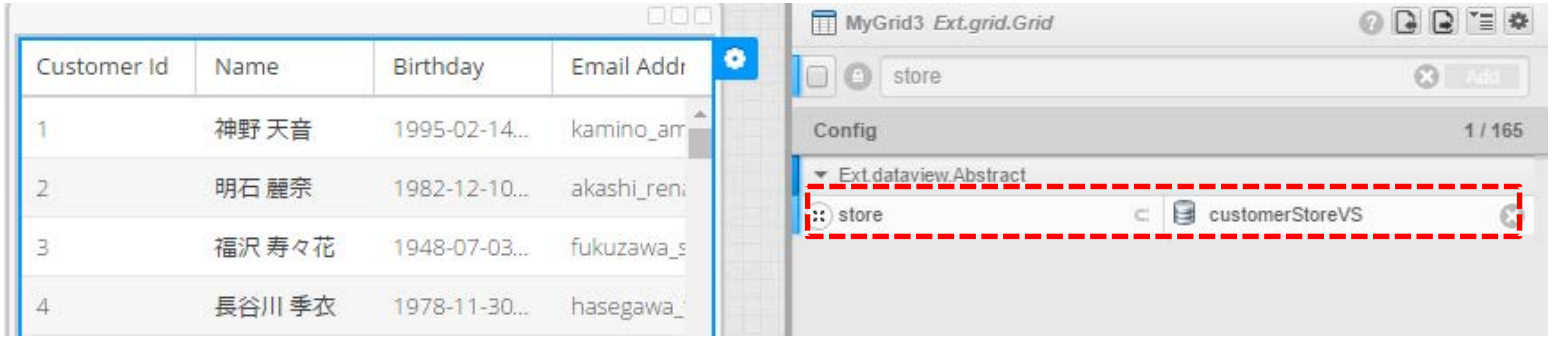
既存のカラムを消す指定

確認：生成されたGridに Store を割り当てる

- Store にはダミーデータが割り当てられている

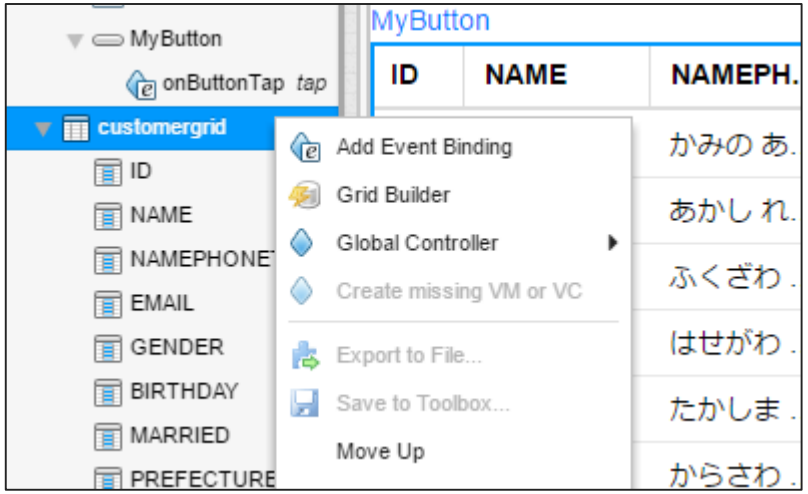


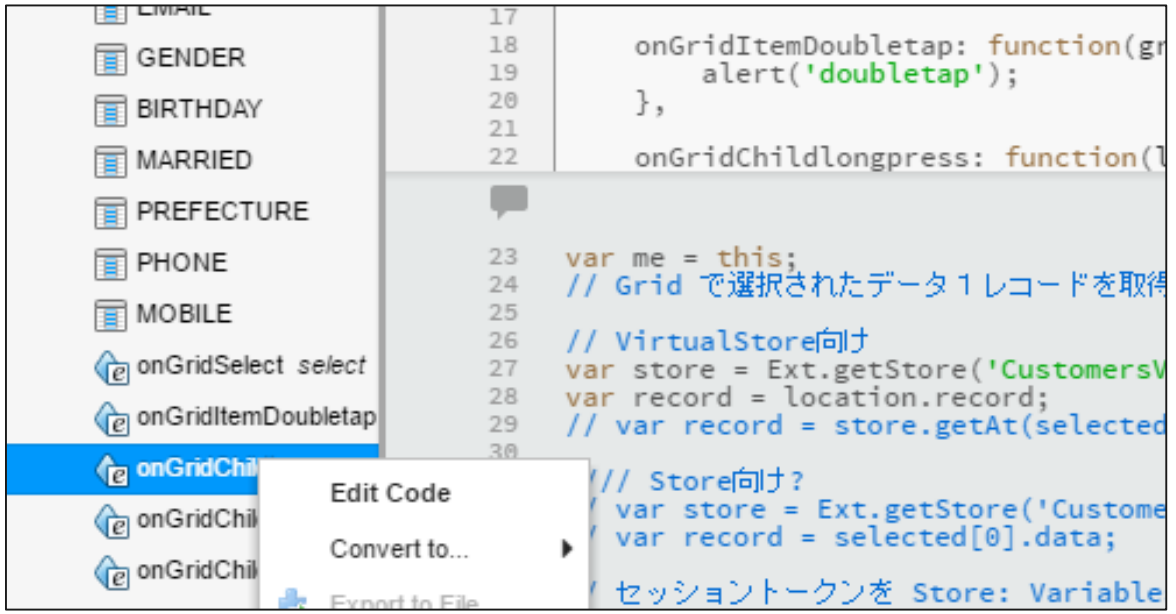
- 実際のデータに変更する



DEMO：イベントに紐づく処理を書く

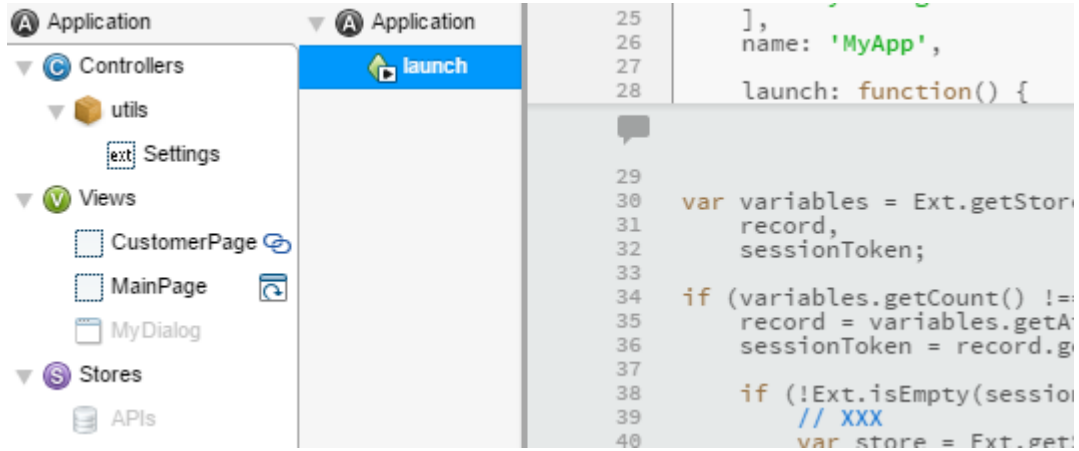
- コンポーネントで「Add Event Binding」してコードを書く

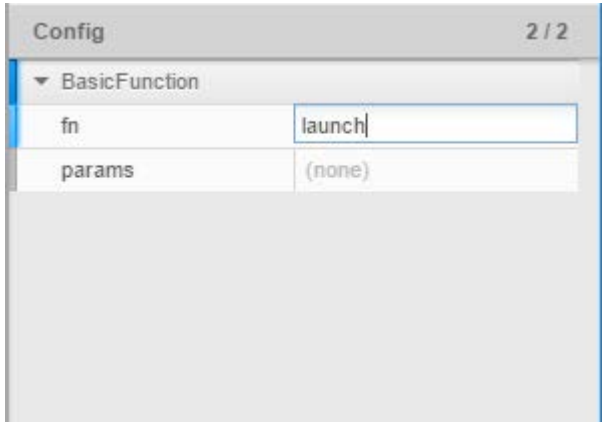




DEMO：認証を行う

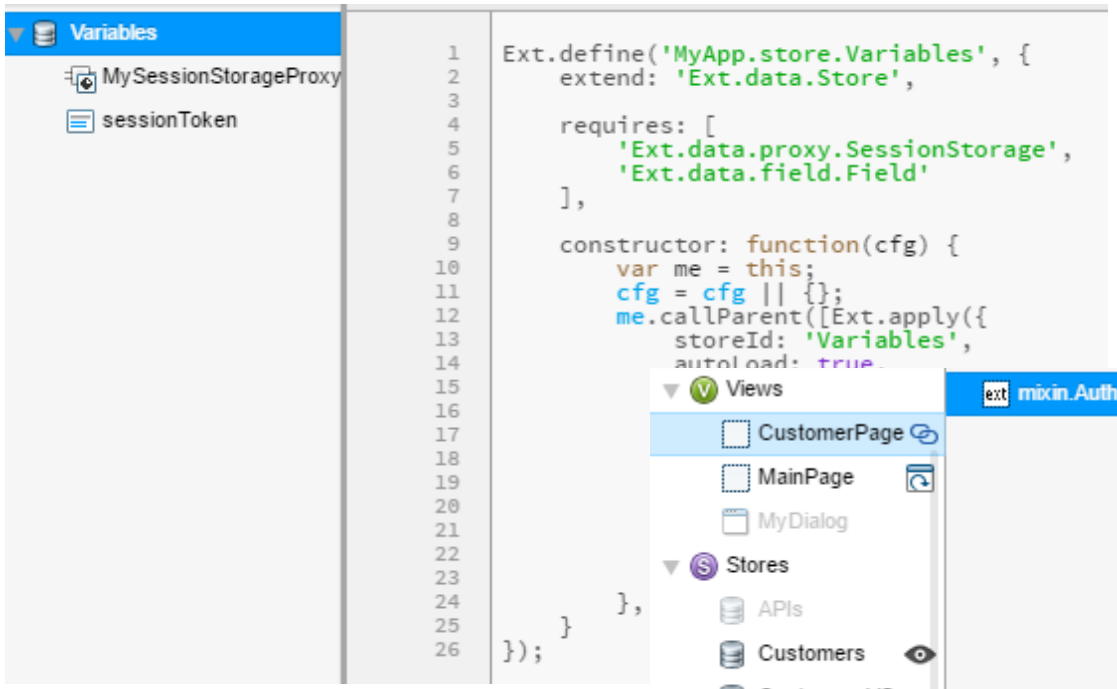
- サイトアクセス時には最初に必ずログイン画面を表示するようにする (BASIC Function を Application に設定し、fn = launch として処理を書く)





DEMO：認証を行う(認証情報の保存、個別リクエストへのセッション情報の付与)

- ログイン後のセッション情報は、専用の Store に保持しておく
- サーバへのHTTP接続時にセッション情報をリクエストヘッダに付与する




```
Ext.define('MyApp.store.Variables', {
    extend: 'Ext.data.Store',

    requires: [
        'Ext.data.proxy.SessionStorage',
        'Ext.data.field.Field'
    ],

    constructor: function(cfg) {
        var me = this;
        cfg = cfg || {};
        me.callParent([Ext.apply({
            storeId: 'Variables',
            autoLoad: true,
        },
    }
});
```

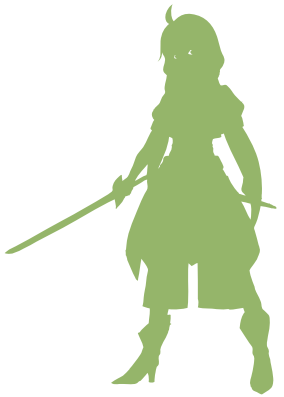


```
Ext.define('MyApp.store.mixin.Auth', {
    extend: 'Ext.Base',

    listeners: {
        beforeLoad: function (store, operation, eOpts) {
            var variables = Ext.getStore('Variables');
            var record = variables.getAt(0);
            var proxy = store.getProxy();
            proxy.setHeaders({
                "X-Embarcadero-Session-Token": record.get('sessionToken')
            });
        }
    }
});
```

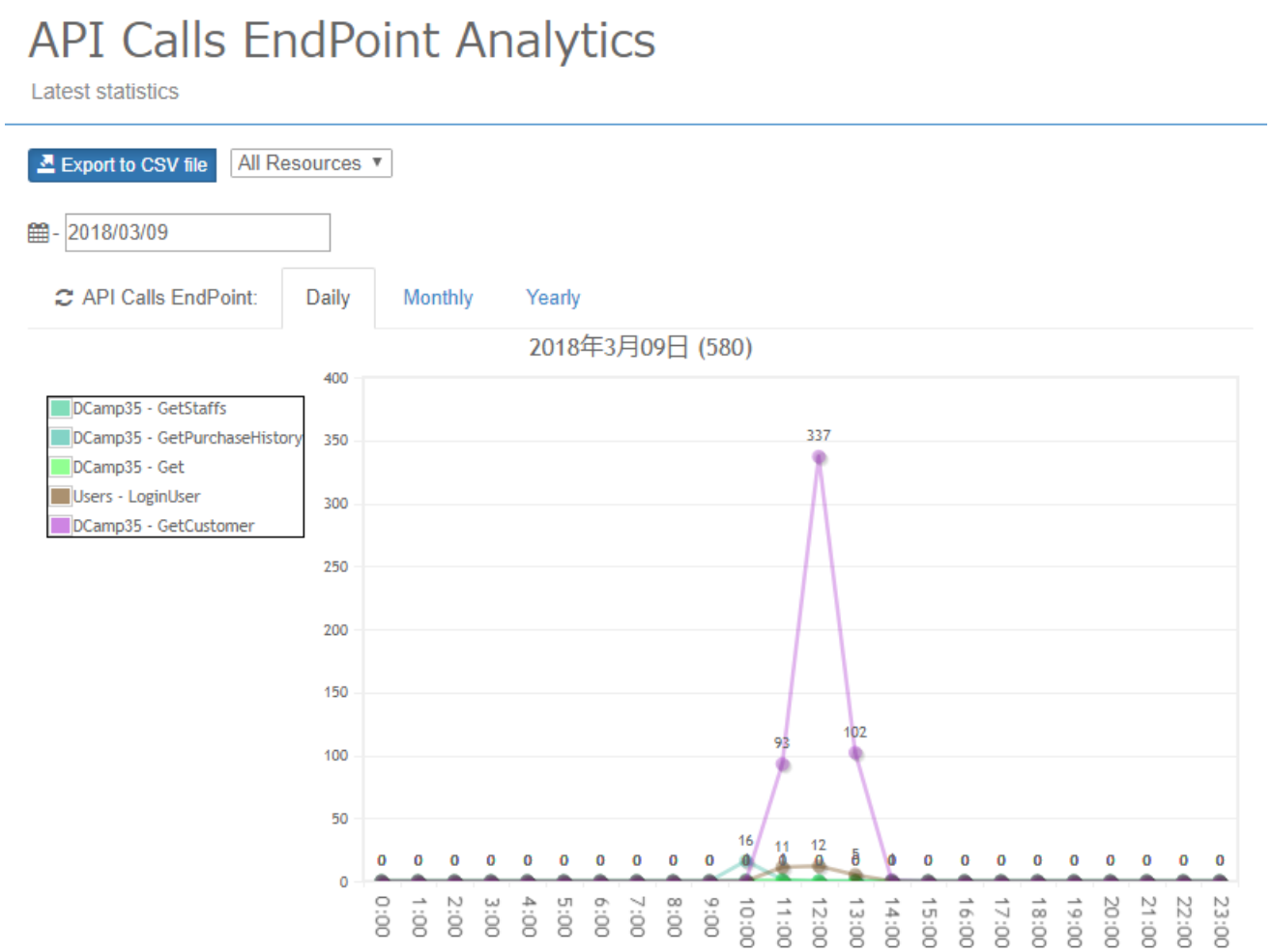
▪管理コンソールで統計情報の表示

RAD Server には統計情報の管理コンソールが含まれますが、
実際の利用状況がどのように表示されるかを確認してみましょう



embarcadero®
DEVELOPER CAMP

API呼び出しはクライアントによらず合算集計されます



まとめ

- 既存のC/Sアプリも３層化でWeb対応できます
- まずはリファクタリングでビジネスロジックやデータアクセスを分離することから
- Sencha Architect のビジュアル開発は Delphi/C++Builder と同様のスタイルで効率的に開発できます

THANKS!

www.embarcadero.com/jp

第35回 エンバカデロ・デベロッパーキャンプ