

## 楽々ウェブクライアント/モジュール設定によるChart.jsの利用

### 1. 概要

Chart.jsはJavascriptでグラフを描画するモジュールです。  
楽々ウェブクライアントでは、これをWebClientで利用するためのインストーラー、変換ルール、パラメータ設定機能を提供しています。

ウィザードによるAngular環境の自動設定

モジュールの組み込み、定義ファイルの更新、変換ルールの追加等の操作をウィザードで簡単に行うことが可能です。

変換ルールとパラメータ設定によるテンプレート変換機能

Chart.js用の変換ルールを指定することで、テンプレートを変換します。  
また、パラメータをGUIで指定することにより、TSファイルにスクリプトを組み込みます。

サンプルプログラムの提供

グラフ描画用の共通プログラム、テスト用プログラムをサンプルとして用意しています。

### 2. MAGIC サンプルプログラム

サンプルプログラムを利用して、動作を確認してみましょう。

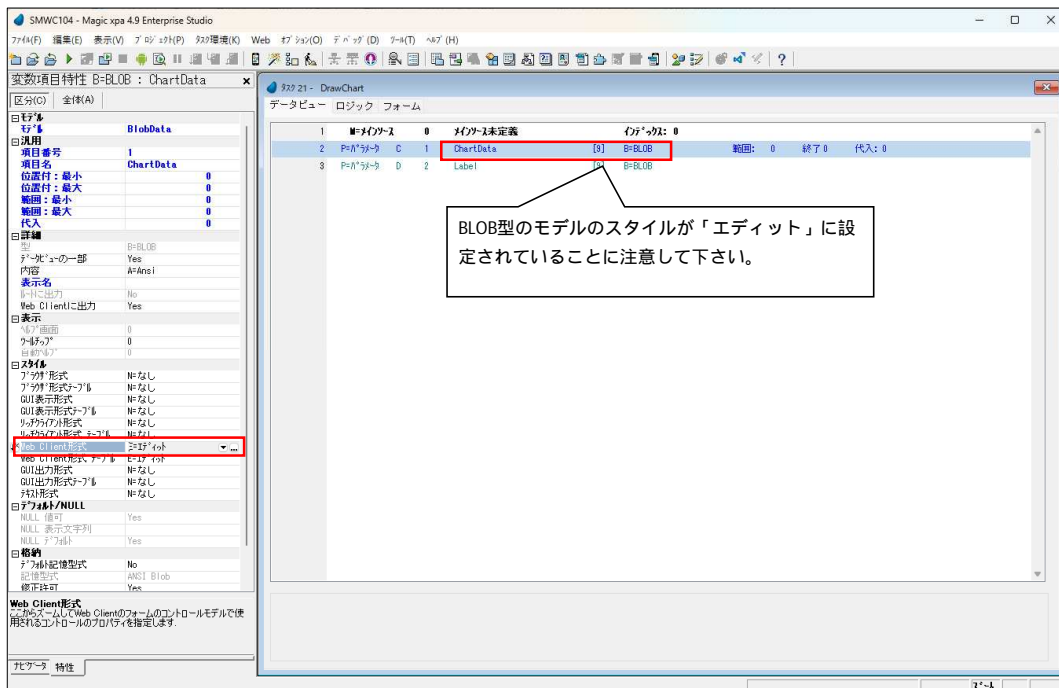
#### a) DrawChart

このプログラムはパラメータでデータを受け取りフォーム上にグラフを描画する基本プログラムです。  
例えば、配置したサブフォームの接続先にこのプログラムを指定することで、グラフを配置することが可能です。

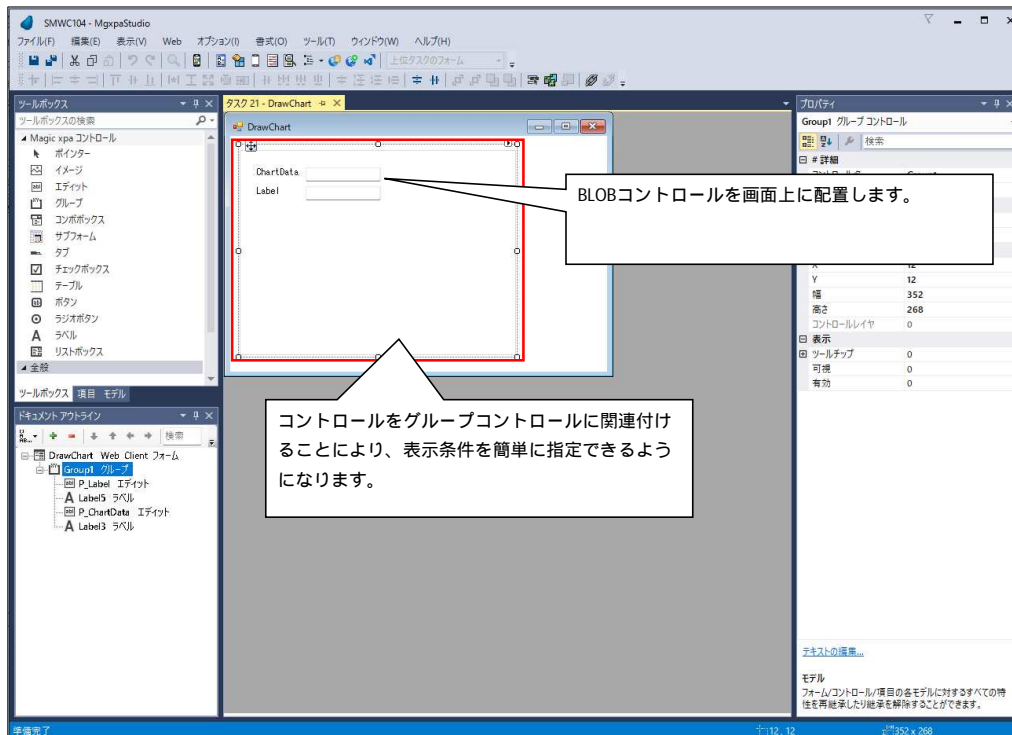
データビューでは、「ChartData」「Label」というパラメータが設定されています。

ChartData グラフのデータ(datasets)をJson形式で渡します  
Label グラフのラベル(label)をカンマ区切りで渡します

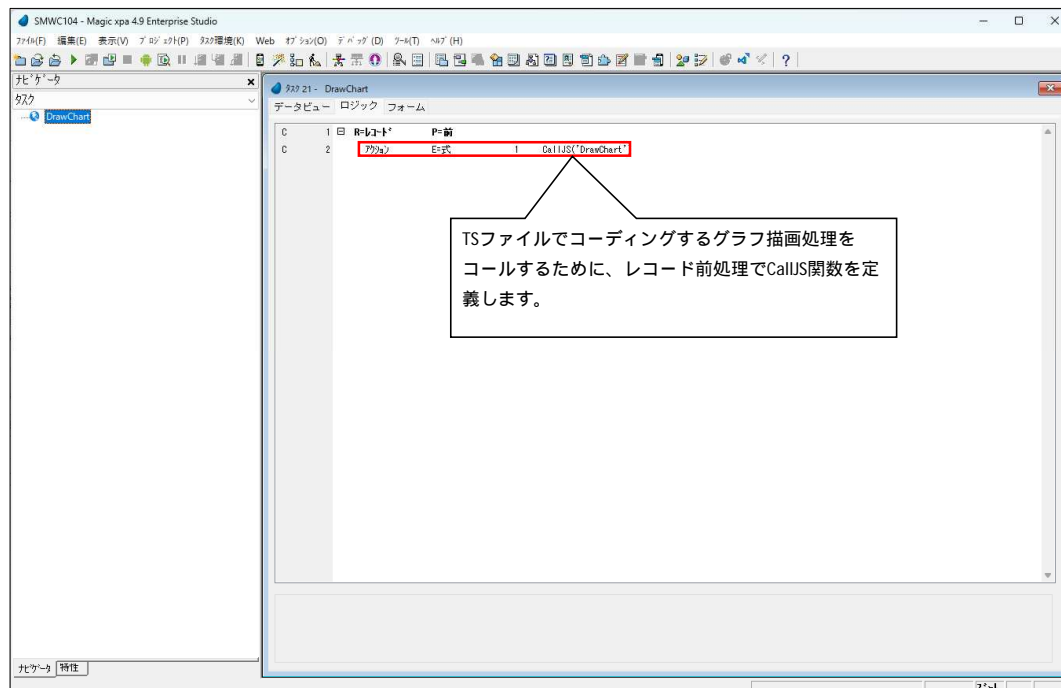
何れもBLOBタイプのモデルを使用しています。  
モデルのスタイル属性に「エディット」が設定されていることが分かります。  
これでフォーム上に配置することができるようになり、ロジック上、WebClientの変数として扱うことが可能になります。



二つの変数を配置しますが、このとき、グループコントロールに関連付けて配置します。  
これは、すべてのコントロールをグループでまとめ、一括して表示をコントロールするためのものです。(任意)



レコード前処理でCallJS関数を定義して下さい。  
指定する関数名は任意のもので構いません。(自動的に取得して使用します)



b) GraphTest

グラフ描画テスト用のサンプルプログラムです。  
 グラフデータをテーブル上で編集することが可能です。編集結果は即時でグラフを再描画します。

このプログラムで使用するデータは3つのテーブルに分かれています。

GraphData

データの番号「DataNo」とタイトル「Title」を保持します。「Check」は有効無効を保持します。  
 商品や製品の一覧などのイメージを想定して下さい。

| DataNo | Title | Check                               |
|--------|-------|-------------------------------------|
| 1      | A     | <input checked="" type="checkbox"/> |
| 2      | B     | <input checked="" type="checkbox"/> |
| 3      | C     | <input checked="" type="checkbox"/> |
| 4      | D     | <input checked="" type="checkbox"/> |

GraphXVal

X軸「Xno」のタイトル「LabelX」を保持します。  
 年度や年月をイメージして下さい。

| XNo | LabelX |
|-----|--------|
| 1   | 2020   |
| 2   | 2021   |
| 3   | 2022   |
| 4   | 2023   |

GraphXYVal

データ「DataNo」別、X軸「XNo」別の値「Val」を保持します。  
 商品別、年度別の売上数量等をイメージして下さい。

| DataNo | XNo | Val    |
|--------|-----|--------|
| 1      | 1   | 111.23 |
| 1      | 2   | 123.45 |
| 1      | 3   | 136.20 |
| 1      | 4   | 141.29 |
| 2      | 1   | 100.23 |
| 2      | 2   | 131.26 |
| 2      | 3   | 111.66 |
| 2      | 4   | 151.21 |
| 3      | 1   | 151.31 |
| 3      | 2   | 144.49 |
| 3      | 3   | 154.22 |
| 3      | 4   | 160.12 |
| 4      | 1   | 94.10  |
| 4      | 2   | 103.26 |
| 4      | 3   | 79.92  |
| 4      | 4   | 98.50  |

データをJson形式で取得するためのプログラムは「SetGraphData」です。  
 テーブルの値から下記のようなデータを返します。

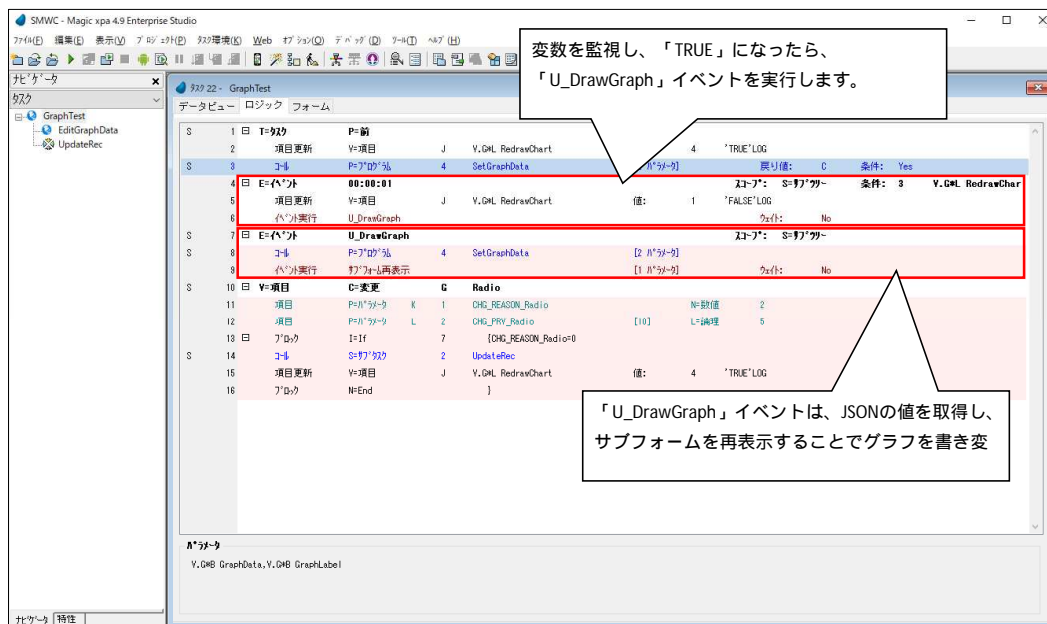
ChartData

```
[
  { "data": [111.23, 123.45, 136.20, 141.29 ], "label": "A"},
  { "data": [100.23, 131.26, 111.66, 151.21 ], "label": "B"},
  { "data": [151.31, 144.49, 154.22, 160.12 ], "label": "C"},
  { "data": [94.10, 103.26, 79.92, 98.50 ], "label": "D"}
]
```

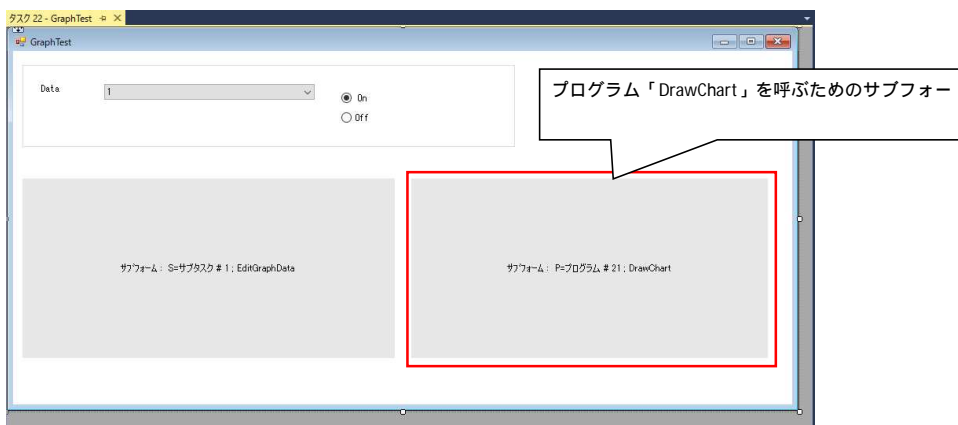
LabelData

```
2020, 2021, 2022, 2023
```

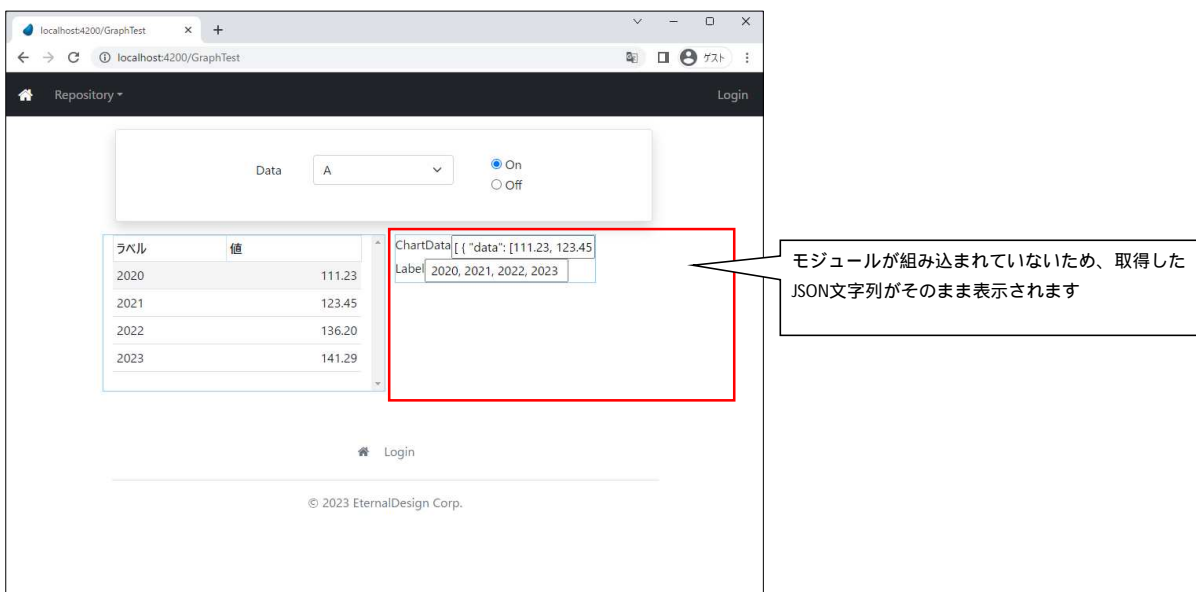
「GraphTest」プログラムに話を戻します。  
 ルートタスクでは、初期化や「RedrawChart」変数がTRUEになるとサブフォームを再描画する処理が組み込まれています。



フォームは下図のようになります。  
 上部にデータを選択するコンボボックスや表示 / 非表示を切り替えるラジオボタン、  
 下部左は、サブタスク(テーブルでデータを編集)をコールするためのサブフォーム、  
 下部右は、グラフ描画用プログラム「DrawChart」を表示するためのサブフォームをそれぞれ設置しています。



実行画面



### 3. モジュール設定ウィザードによるchart.js環境の設定

ウィザードを使ってChart.jsを組み込んでみましょう。

< 手順 >

#### a) 定義ファイルのインストール

解凍して得られるファイル「WCSetupOptionFile.xml」をSMSYSシステムフォルダ(論理名「SMSYS」で定義)にインストールします。

```

1 <?xml version="1.0" encoding="UTF-8" standalone="no" ?>
2 <System xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:noNamespaceSchemaLocation="WebClientSettings.xsd">
3
4 <option_settings>
5 <option_setting comment="グラフ描画用モジュール「chart.js」を組み込みます。" option_id="chart.js" seq="2" title="chart.js">
6 <node_modules module_disp_name="chart.js@2.9.4" module_id="chart.js" seq="1" version="2.9.4"/>
7 <node_modules module_disp_name="ng2-charts@2.4.3" module_id="ng2-charts" seq="2" version="2.4.3"/>
8 <json_settings data_id="1" file_name="angular.json" is_array="1" path="$projects.$PROJECT_NAME$.architect.build.options.scripts">
9 <array_vals module_id="chart.js" seq="1" val="./node_modules/chart.js/dist/Chart.min.js"/>
10 </json_settings>
11 <ts_settings file_name="magic.gen.lib.module.ts" seq="1">
12 <import_data from="ng2-charts" name="ChartsModule" seq="1"/>
13 <def_data name="imports" seq="1" val="ChartsModule"/>
14 </ts_settings>
15 <html_cnvs comment="chart.jsを利用してグラフ表示&#x0d;&#xA;(bar, line, doughnut, radar, pie)" data_id="1" imp_id="OUI_001" title="7-1表示">
16 <definitions element="div" insert_attr_to_new_path="class" insert_val_to_new_path="card bg-info" optype="2" path="/div/div/div"/>
17 <definitions element="div" insert_attr_to_new_path="class" insert_val_to_new_path="card-body" optype="2" path="/div/div/div/div"/>
18 <definitions element="div" optype="7" path="/div/div/div/div/div"/>
19 <graph_settings id="1" name="chartType" nange="bar,line,doughnut,radar,pie" type="1" val="ChartType"/>
20 <graph_settings id="2" name="datasets" type="1" val="ChartData"/>
21 <graph_settings id="3" name="labels" type="1" val="ChartLabels"/>
22 </html_cnvs>
23 </option_settings>
24 </option_settings>
25
26 </System>
27

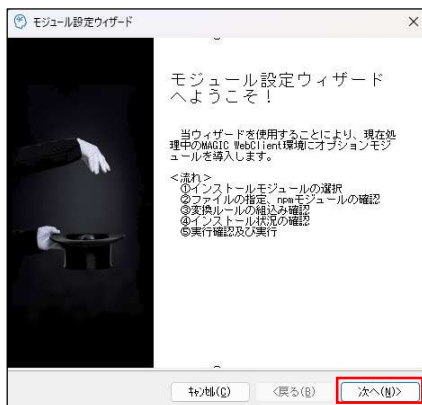
```

#### b) 初期設定アシスタント

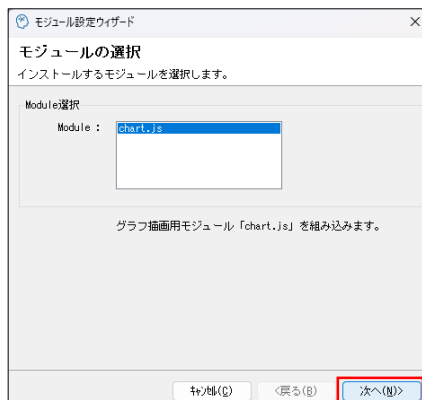
初期設定アシスタントを起動します。  
「モジュール設定ウィザード」を起動します。



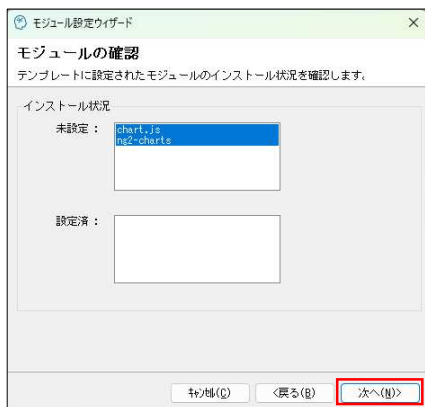
「次へ」を押します。



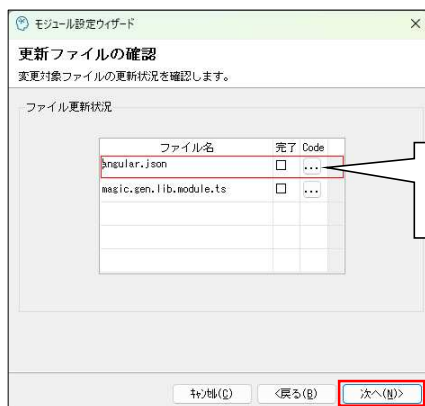
「chart.js」を選択し、「次へ」を押します。



インストール状況を確認し、「次へ」を押します。

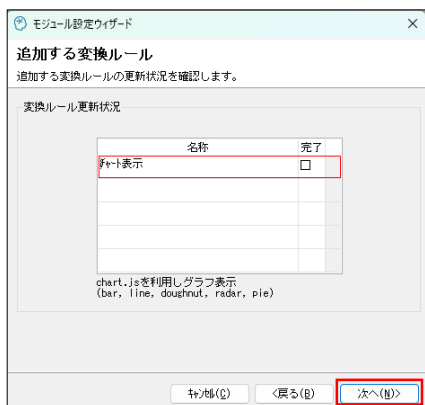


ファイル更新状況を確認し、「次へ」を押します。



「Code」のボタンを押すとそれぞれのファイルをVS Codeで開き、現在の内容を確認することができます

変換ルール更新状況を確認し、「次へ」を押します。



処理内容を確認し、「実行」を押します。



「実行」ボタンで処理を開始します。

途中、DOS窓が開き、モジュールのインストール状況が表示されます。  
終了したら、続行のため何かキーを押します。

```
npm install ng2-charts@2.4.3 x + v
added 3 packages, removed 1 package, and audited 1014 packages in 8s
122 packages are looking for funding
  run `npm fund` for details
8 vulnerabilities (2 moderate, 4 high, 2 critical)
To address all issues (including breaking changes), run:
  npm audit fix --force
Run `npm audit` for details.
added 3 packages, and audited 1017 packages in 3s
122 packages are looking for funding
  run `npm fund` for details
8 vulnerabilities (2 moderate, 4 high, 2 critical)
To address all issues (including breaking changes), run:
  npm audit fix --force
Run `npm audit` for details.
続行するには何かキーを押してください . . .
```

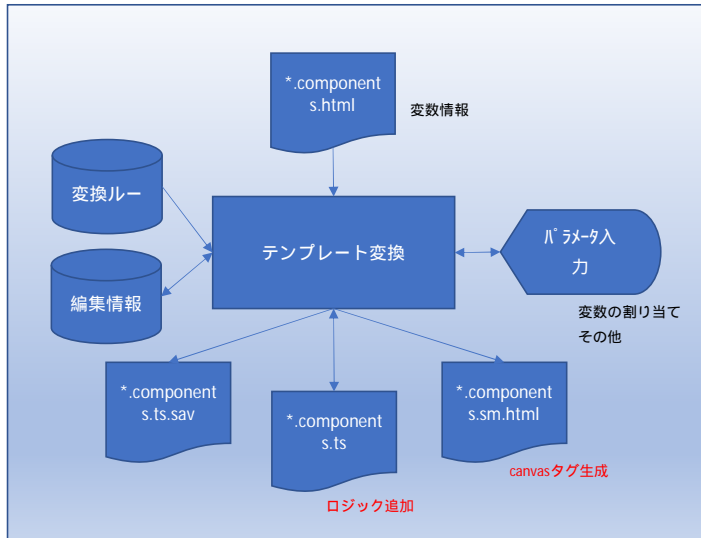
処理が終わるとダイアログが出ます。  
「OK」ボタンを押します。



これでインストールは終了です。

### 3. テンプレート変換

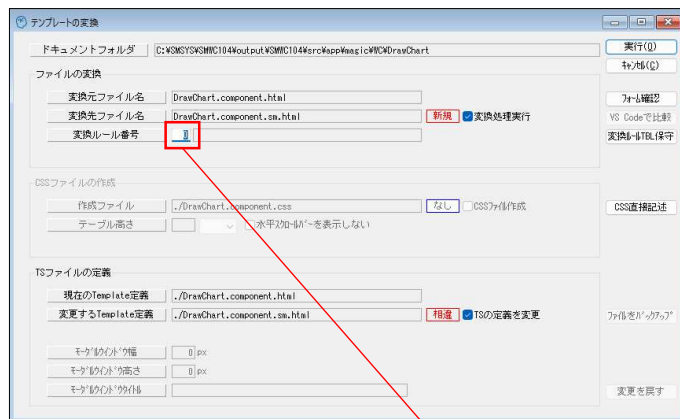
ウィザードで組み込まれた変換ルールを使用し、テンプレートファイルの変換を行ってみましょう。  
 今までと異なる点としては、テンプレートファイルに特殊なタグが追加されたり、TSファイルにロジックが追加されることです。



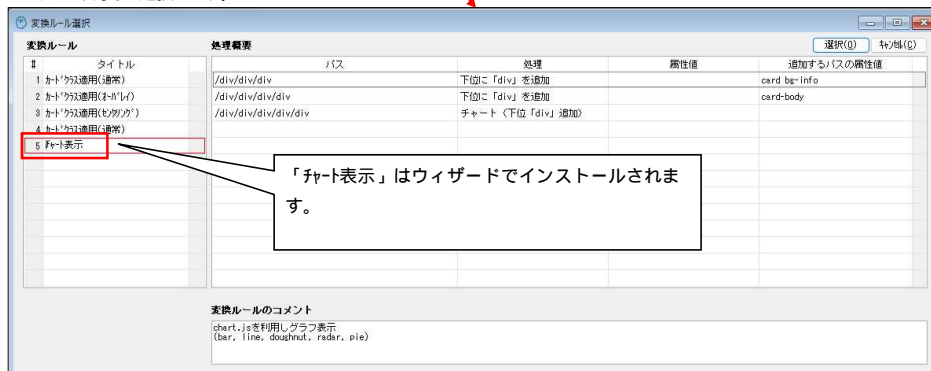
WebClientフォームから「DrawChart」プログラムを選択します。  
 「変換」ボタンを押して、テンプレート変換を起動します。



変換ルール番号でズームし、「チャート表示」を選択します。

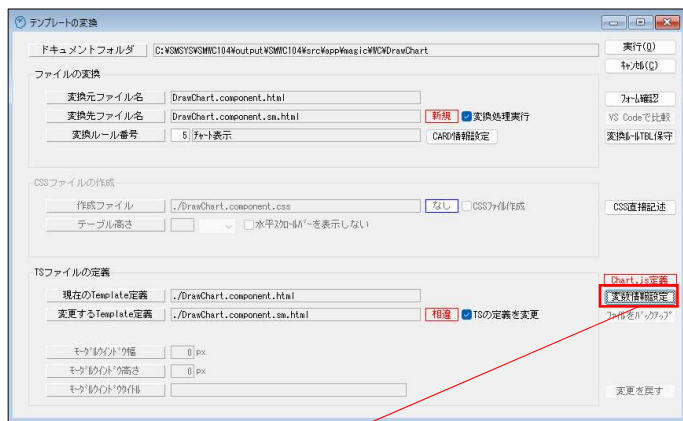


「チャート表示」を選択します。



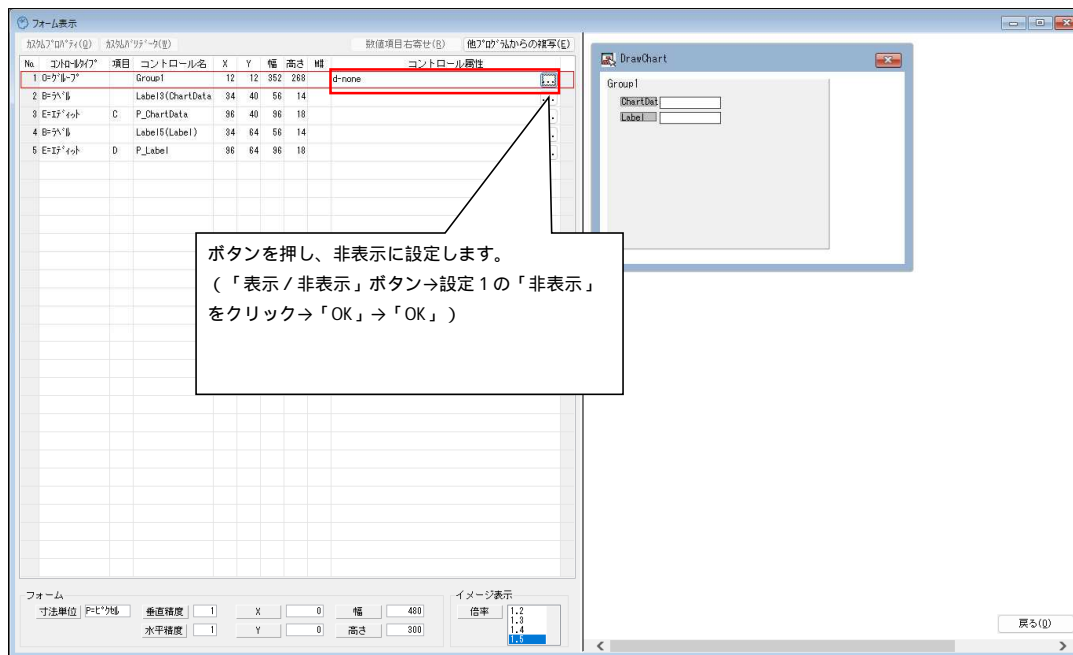


「変数情報設定」ボタンを押します。



| 名前        | タイプ      | コントロール/値    |
|-----------|----------|-------------|
| chartType | 2=値      | bar         |
| datasets  | 1=コントロール | P_ChartData |
| labels    | 1=コントロール | P_Label     |

「フォーム確認」ボタンを押します。  
グループコントロールの属性を開き、非表示に設定します。



「実行」ボタンを押して、テンプレート及びTSファイルを変換します。

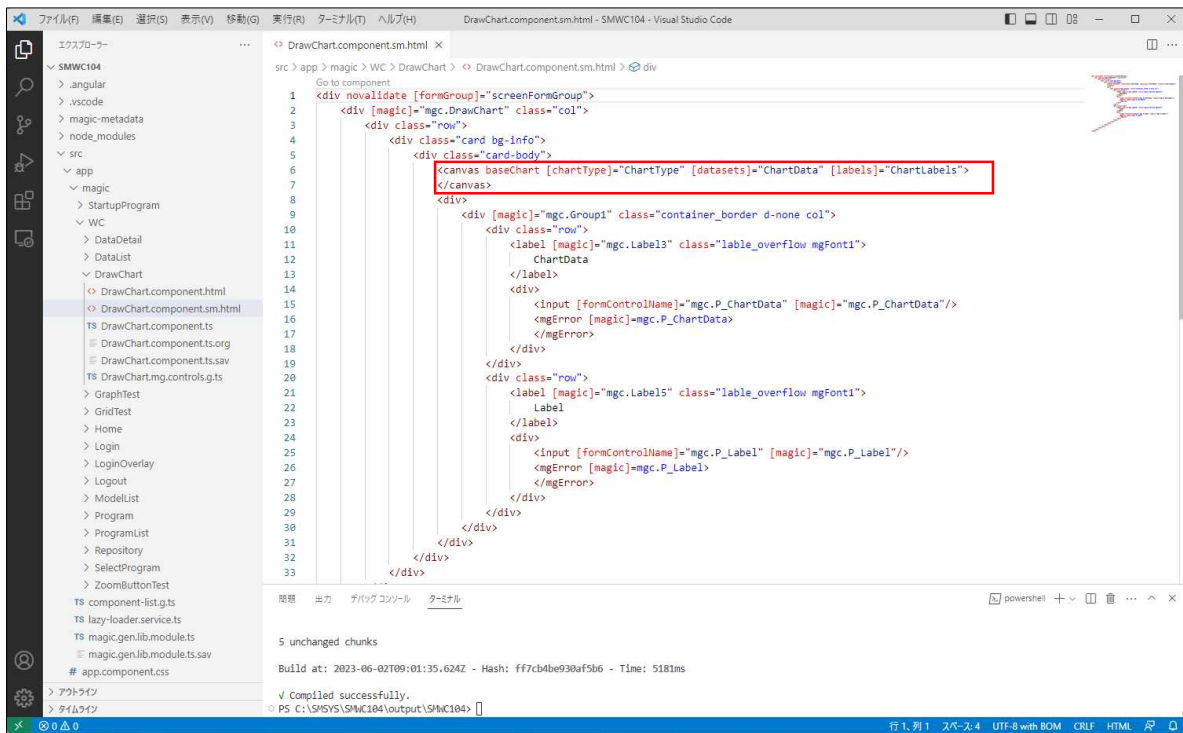
#### 4. VS Codeによる確認と一部手修正

この状態で実行可能となればいいのですが、一部のTSファイルの編集が必要になります。以下に編集手順を示します。

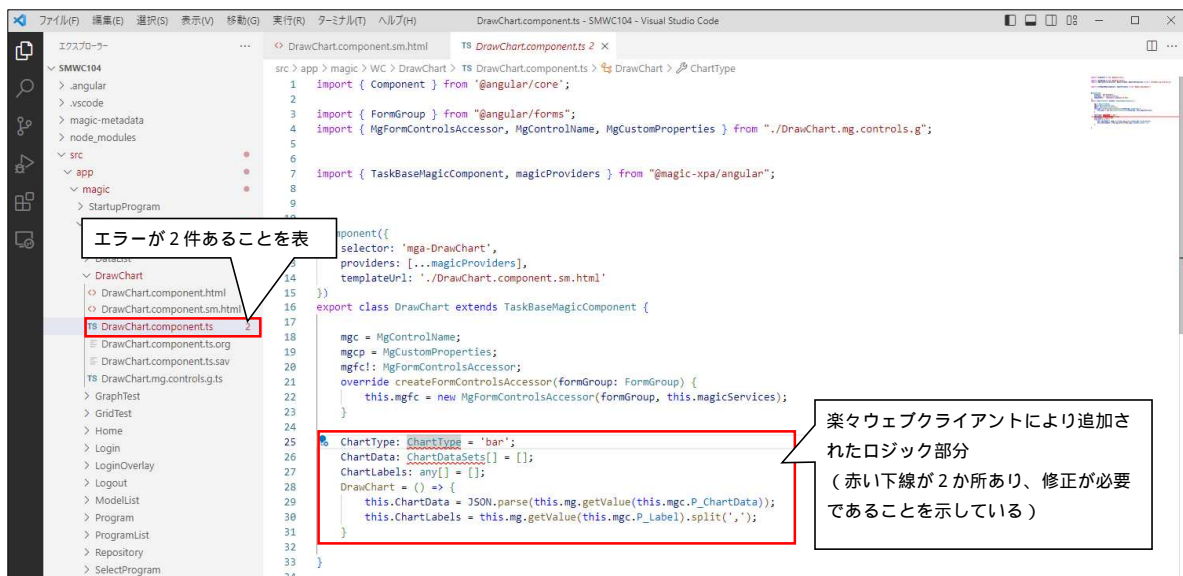
「VS Codeで開く」ボタンを押して、テンプレートファイルを開きます。



「canvas」タグが挿入されていることを確認します。



ツリービューから「DrawChart.component.ts」ファイルを選択します。  
「DrawChart」クラスが追加されているようですが、2か所、変数名に赤い下線がついています。



### 'ChartType'修正方法

'ChartType'の最後の文字'e'の後ろにカーソルを移動し、1文字BackSpaceで削除した後、「e」を入力します。  
変数名の候補に'ChartType'が表示されるので選択します。  
正しく認識されると赤い下線は消えます。

```

25 ChartType: ChartType = 'bar';
26 ChartData: ChartData = [];
27 ChartLabels: any[] = [];
28 DrawChart = () => {
29     this.ChartData = JSON.parse(this.mg.getValue(this.mgc.P_ChartData));
30     this.ChartLabels = this.mg.getValue(this.mgc.P_Label).split(',');
31 }

```

### 'ChartDataSets'修正方法

'ChartDataSets'の最後の文字's'の後ろにカーソルを移動し、1文字BackSpaceで削除した後、「s」を入力します。  
変数名の候補に'ChartDataSets'が表示されるので選択します。  
正しく認識されると赤い下線は消えます。

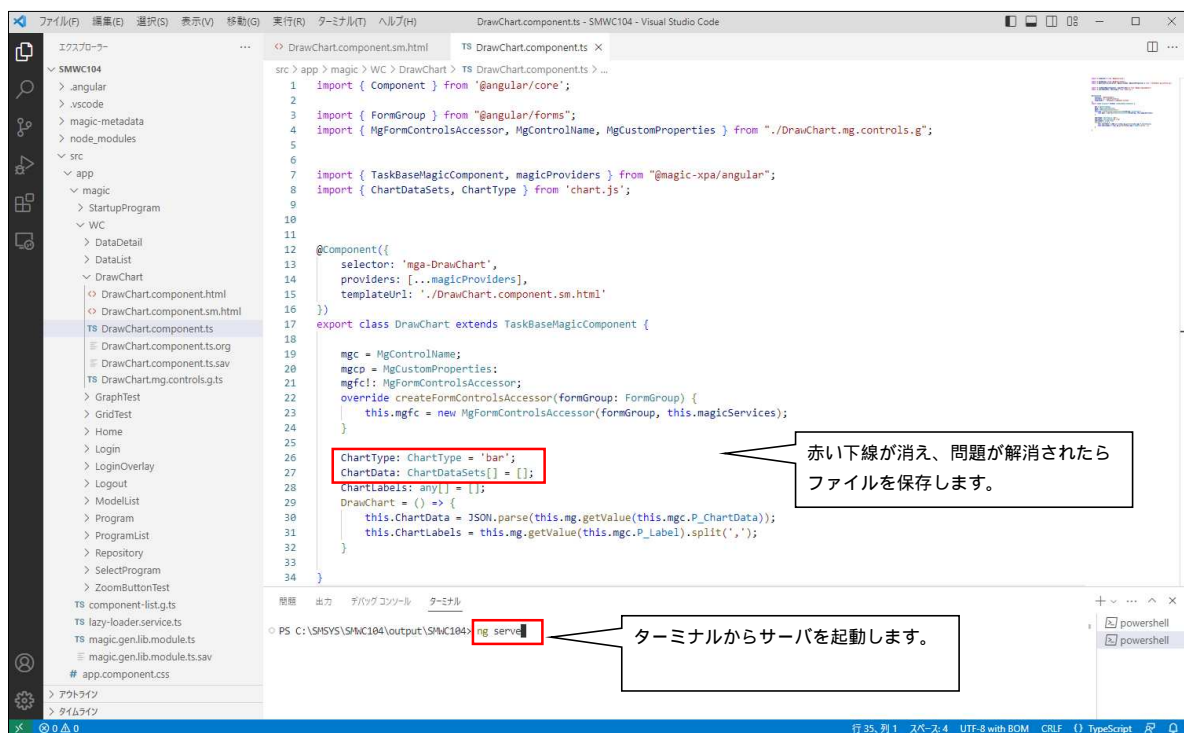
```

26 ChartType: ChartType = 'bar';
27 ChartData: ChartData = [];
28 ChartLabels: any[] = [];
29 DrawChart = () => {
30     this.ChartData = JSON.parse(this.mg.getValue(this.mgc.P_ChartData));
31     this.ChartLabels = this.mg.getValue(this.mgc.P_Label).split(',');
32 }

```

修正後、ファイルを上書き保存します。

ターミナルから'ng serve'を実行します。



## 5. 実行画面

グラフが表示されました。

コンボボックスを変更することにより、編集するデータを切り替えます。ラジオボタンでデータの表示 / 非表示を変更します。

値をクリックし、数値を入力することで編集が可能です。

データに変更があると自動的に再描画します。

| ラベル  | 値      |
|------|--------|
| 2020 | 111.23 |
| 2021 | 123.45 |
| 2022 | 136.20 |
| 2023 | 141.29 |

## 6. グラフ種類の変更

変数情報設定画面でChartTypeを「line」に変更してみます。

「chartType」の値「line」をコンボボックスから選択します。（「bar」「line」以外にも「doughnut」「radar」「pye」の選択が可能です）

| # | 名前        | タイプ      | コントロール/値の指定 | 無効                       | OK    |
|---|-----------|----------|-------------|--------------------------|-------|
| 1 | chartType | 2=値      | line        | <input type="checkbox"/> | 転写/削除 |
| 2 | datasets  | 1=コントロール | P_ChartData | <input type="checkbox"/> |       |
| 3 | labels    | 1=コントロール | P_Label     | <input type="checkbox"/> |       |

グラフ表示が変わりました。

| ラベル  | 値      |
|------|--------|
| 2020 | 111.23 |
| 2021 | 123.45 |
| 2022 | 136.20 |
| 2023 | 141.29 |

補足) サンプルプログラムの説明から抜粋

11) DrawChart

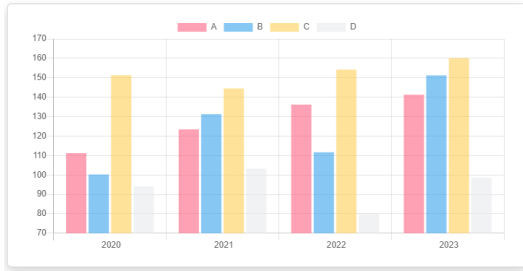


図2.5.3-11 チャート描画面面のデザイン例

変換ルール番号 5:チャート表示  
 変数情報設定 chartType 2=値 bar  
 datasets 1=コントロール P\_ChartData  
 labels 1=コントロール P\_Label

コントロール属性

| コントロールタイプ | 項目 | コントロール名     | x  | y  | 幅   | 高さ  | 設定値    |
|-----------|----|-------------|----|----|-----|-----|--------|
| 1 O=グループ  |    | Group1      | 12 | 12 | 352 | 268 | d:none |
| 2 B=ラベル   |    | Label3      | 34 | 40 | 56  | 14  |        |
| 3 E=エディット | C  | P_ChartData | 96 | 40 | 96  | 18  |        |
| 4 B=ラベル   |    | Label5      | 34 | 64 | 56  | 14  |        |
| 5 E=エディット | D  | P_Label     | 96 | 64 | 96  | 18  |        |

モジュール設定ウィザードにより「Chart.js」のインストールが必要になります。

12) GraphTest

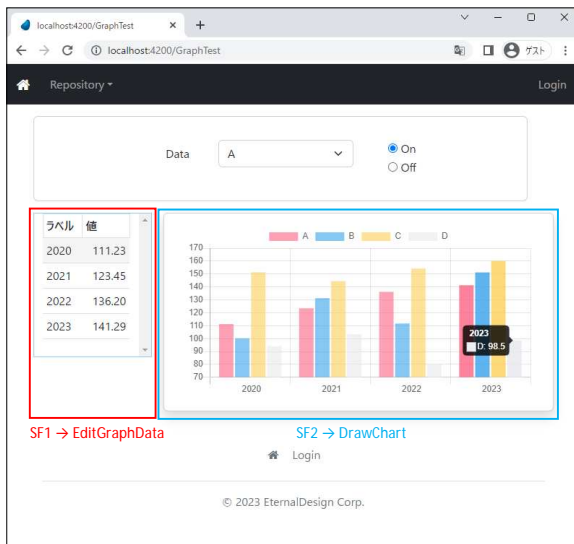


図2.5.3-12 グラフテスト画面のデザイン例

GraphTest

変換ルール番号 1:カードクラス適用(通常)  
 エリア別ルール設定 エリア2を除外

コントロール属性

| コントロールタイプ      | 項目 | コントロール名      | x   | y   | 幅   | 高さ  | 設定値                                   |
|----------------|----|--------------|-----|-----|-----|-----|---------------------------------------|
| 1 O=グループ       |    | Group18      | 12  | 12  | 627 | 111 | mb-3/del_border                       |
| 2 B=ラベル        |    | Label1(Data) | 36  | 42  | 62  | 20  | col-form-label col text-end me-3      |
| 3 C=コントロールボックス | C  | DataNo       | 116 | 42  | 268 | 18  | form-select me-3/col/size:1           |
| 4 D=ラジオボタン     | G  | V Radio      | 407 | 42  | 61  | 62  | col/form-check form-check-inline/me-3 |
| 5 U=サブフォーム     |    | SF1          | 12  | 162 | 474 | 228 | me-3 d-block col mb-3                 |
| 6 U=サブフォーム     |    | SF2          | 507 | 162 | 454 | 228 | d-block col                           |

EditGraphData

変換ルール番号 1:サンプル TABLE変換  
 テーブル高さ 200px  
 水平スクロールバー 表示しない

コントロール属性

| コントロールタイプ     | 項目 | コントロール名   | x   | y  | 幅   | 高さ  | 設定値                         |
|---------------|----|-----------|-----|----|-----|-----|-----------------------------|
| 1 T=テーブル      |    | Table1    | 12  | 12 | 318 | 244 |                             |
| 2 M=コラム       |    | Column1   | 0   | 0  | 50  | 0   | d:none                      |
| 3 M=コラム       |    | Column2   | 0   | 0  | 44  | 0   | d:none                      |
| 4 M=コラム       |    | Column4   | 0   | 0  | 74  | 0   |                             |
| 5 M=コラム       |    | Column6   | 0   | 0  | 98  | 0   |                             |
| 6 P=アクションボタン  |    | PB Edit   | 16  | 34 | 36  | 21  | btn btn-secondary/fa-pencil |
| 7 E=エディット     | O  | XNo       | 66  | 34 | 36  | 19  | pull-right                  |
| 8 E=エディット     | R  | LabelX    | 110 | 34 | 66  | 19  |                             |
| 9 E=エディット     | Q  | Val       | 184 | 34 | 90  | 19  | pull-right                  |
| 10 P=アクションボタン |    | PB Update | 16  | 59 | 36  | 21  | btn btn-danger/fa-check     |
| 11 P=アクションボタン |    | PB Cancel | 16  | 84 | 36  | 21  | btn btn-warning/fa-undo     |