

インタラクティブ画面

Web Panel オブジェクト

GeneXus[™]

Web Panel オブジェクト

- 柔軟性に優れた Web 画面を実装
- 例:

EnterAttractionFilter X

Web Form Rules Events Conditions Variables Help Documenta

<選択されているアクショングループはありません>

MainTable

国番号 &CountryI

観光名所ふりがな &FromAttractionFurigana

観光名所ふりがな &ToAttractionFurigana

国の観光名所リスト

ふりがなフィルタつき観光名所リスト

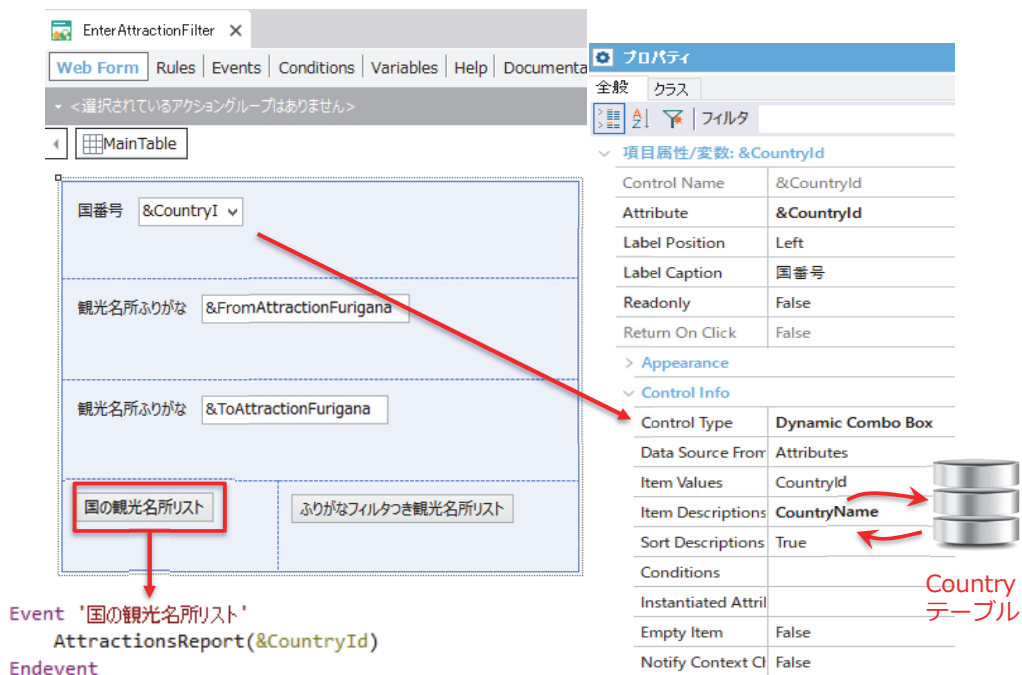
変数: 入力コントロール
(読み取り専用ではない)

Web パネルは GeneXus が提供する最も多目的に利用できるオブジェクトです。

いくつかの例で見てきたように、Web パネルには、Web ページのデザインを可能にする Web フォームがあり、さまざまな機能を提供できるようになっています。

例では、Web フォームに含める変数にユーザーが値を割り当てられることを示しています。これは、変数が入力コントロールであり、読み取り専用ではないという意味です。

例: Web パネル内の変数



EnterAttractionFilter X

Web Form Rules Events Conditions Variables Help Documenta

<選択されているアクショングループはありません>

MainTable

国番号 &CountryId

観光名所ふりがな &FromAttractionFurigana

観光名所ふりがな &ToAttractionFurigana

国の観光名所リスト

ふりがなフィルタつき観光名所リスト

Event '国の観光名所リスト'
AttractionsReport(&CountryId)
Endevent

プロパティ

全般 クラス

項目属性/変数: &CountryId

Control Name	&CountryId
Attribute	&CountryId
Label Position	Left
Label Caption	国番号
Readonly	False
Return On Click	False

> Appearance

> Control Info

Control Type	Dynamic Combo Box
Data Source From	Attributes
Item Values	CountryId
Item Descriptions	CountryName
Sort Descriptions	True
Conditions	
Instantiated Attril	
Empty Item	False
Notify Context Cl	False

Country データベース

具体的には、ダイナミック コンボ ボックス タイプの上図の変数では、コンボボックスにロードされる選択肢の中から国を 1 つ選択することがユーザーに求められます。[国別の観光名所リスト] ボタンをクリックすると、関連付けられているイベントが実行され、その国の観光名所リストを含む PDF ファイルが呼び出されます。

ここで、データベースは、コンボボックスに値をロードするためだけにアクセスされます。

例: Web パネル内の変数

The screenshot displays the GeneXus IDE interface. On the left, a web panel design is shown with the following elements:

- A dropdown menu labeled "国番号" with the value "&CountryI".
- Two input fields labeled "観光名所ふりがな" with values "&FromAttractionFurigana" and "&ToAttractionFurigana".
- Two buttons: "国の観光名所リスト" and "ふりがなフィルタつき観光名所リスト".

On the right, the "Rules" window for the "AttractionsByname *" object is shown. It contains the following code:

```
1 Parm(&FuriganaFrom,&FuriganaTo);
```

Below the code, the event definition is shown:

```
Event 'callAttractionsByName'  
AttractionsByname(&FromAttractionFurigana,&ToAttractionFurigana)  
Endevent
```

Red arrows indicate the mapping from the input fields in the web panel to the parameters in the rule code.

これらの変数では、ユーザーが観光名所名の範囲を入力し、もう 1 つのボタンをクリックすると、パラメーターで指定された範囲内の観光名所リストを示す PDF ファイルが呼び出されます。

例:

プロシージャ
オブジェクトではなく、
Web パネルで、
この PDF リストを
実装できるか?

The screenshot shows the GeneXus IDE with a web panel layout on the left and source code on the right. The layout includes a title bar, a column titles section, and a table of attractions. The source code is as follows:

```

1 print Title
2 print columnTitles
3 For each Attraction order AttractionName
4   where AttractionFurigana >= &FuriganaFrom
5   where AttractionFurigana <= &FuriganaTo
6   print attractions
7 Endfor

```

[Layout] について思い出してください。[Source] では、データベースへの問い合わせに For each を使用し、名前（ふりがな）でフィルタリングしました。

これらの問い合わせは、なぜユーザーがフィルタのデータを入力する画面で直接定義するのではなく、PDF リストを通じて定義しているのでしょうか。

データベースに問い合わせできるように Web パネルを変更する

The screenshot shows the GeneXus Web Form editor interface. At the top, there's a tab labeled 'EnterAttractionFilter' and a menu with 'Web Form', 'Rules', 'Events', 'Conditions', 'Variables', 'Help', and 'Documenta'. Below the menu, a message says '<選択されているアクショングループはありません>'. The main area shows a form with the following fields:

- 国番号: &CountryI (dropdown)
- 観光名所ふりがな: &FromAttractionFurigana
- 観光名所ふりがな: &ToAttractionFurigana

At the bottom of the form, there are two buttons: '国の観光名所リスト' and 'ふりがなで検索する観光名所リスト'. A red arrow points to the 'ふりがなで検索する観光名所リスト' button, with the label '観光名所のグリッド' next to it. To the right of the form, there is a database icon and a table structure for 'Attractions'.

AttractionName	CountryName	Attrac	CategoryName

ここに、ボタンではなく観光名所を示すグリッドを追加したらどうでしょうか。

グリッドの明細行は、各観光名所を示すプリントブロックのようにします。

Web パネル: データベースに対する対話形式の問い合わせ

Web パネル内の
項目属性は
出力項目属性:
読み取り専用



Web パネルでは、ボタンにプログラミングされているアクションで使用する変数を定義できるだけでなく、データベースに対する**対話形式**の問い合わせを実装できます。むしろ、これが Web パネルの主な目的です。

「**対話形式 (インタラクティブ)**」とは、ユーザーが Web ページで多くの異なる値を**変数**に入力し、その入力された値と一致するデータをデータベースに照会し、それらをフィルタとして使用できることを意味します。詳しくは後述します。

ここで、この Web パネルを別名で保存します。WorkWith パターンの Selection に似たものを実装するので、名前を WWAttractionsFromScratch にします。

ボタンと、対応するイベントは不要になったので削除します。変数の下に、グリッドタイプのコントロールを挿入します。

ダイアログボックスが開き、グリッドの列として使用する項目属性や変数の選択を求められます。PDF リストと同じ内容を表示するので、AttractionId、AttractionName、CountryName、AttractionPhotoの各項目属性を選択します。その後、[OK] をクリックします。これらの列を含むグリッドが作成されます。列見出しを変更するには、グリッドの列を構成する各項目属性のプロパティを編集します。

Web パネルフォーム内の**項目属性**は、既定で**出力項目属性**になります。つまり、項目属性は**読み取り専用**です。このため、GeneXus では、項目属性の値をデータベースで検索し、ユーザーに表示する必要があることを理解します。

Web パネル: データベースに対する対話形式の問い合わせ

F5

国番号

アメリカ合衆国 ▼

観光名所ふりがな

観光名所ふりがな

観光名所番号	観光名所名	国名	観光名所写真
2	ルーブル美術館	フランス	
3	万里の長城	中国	
4	エッフェル塔	フランス	
5	コルコバードのキリスト像	ブラジル	
10	スミソニアン博物館	アメリカ合衆国	
11	東京スカイツリー	日本	


DEMO

F5 キーを押して現時点の Web パネルを実行します。

すべての観光名所について、指定したデータ (番号、名前、国、写真) が出力されます。
また、AttractionId の順序になっていることも分かります。

項目属性を含むグリッドは For each コマンドと同等

The screenshot illustrates the relationship between a web form's grid and a 'For each' command in GeneXus. The web form on the left includes a 'CountryID' dropdown and a grid with columns: 'AttractionID', 'AttractionName', 'CountryName', and 'AttractionPhoto'. The code editor in the center shows a 'For each' loop with the following code:

```

1 Print Title
2 Print ColumnTitles
3 For each Attraction order CountryName
4   Where AttractionFurigana = &NameFrom
5   Where AttractionFurigana = &NameTo
6   Print Attractions
7 Endfor

```

A red box highlights the 'Base Trn' property in the grid's properties panel, which is set to 'Attraction'. This indicates that the grid is configured to use the 'Attraction' table as its base table, making it equivalent to the 'For each' command.

標準グリッドに指定された項目属性に基づき、GeneXus は各観光名所の国情報を取得するために、Attraction テーブルにナビゲートした上で、Country にアクセスすることが必要なことを理解します。

この問い合わせは、For each コマンドから Order 節と Where 節を取り除いたものと同じです。

グリッドのプロパティの 1 つに **[Base Trn]** があります。このプロパティは For each コマンドのベーストランザクションに似ています。グリッドにおいて Attraction がベーステーブルとして選択されることを確実にするには、開発者は For each コマンドのようにベーストランザクションを指定する必要があります。

項目属性を含むグリッドは For each コマンドと同等

The screenshot displays the GeneXus IDE interface. On the left, a web form is shown with a dropdown menu for country selection and a grid of attraction data. The grid columns are labeled: 観光名所番号 (AttractionID), 観光名所名 (AttractionName), 国名 (CountryName), and 観光名所写真 (AttractionImage). The source code in the center shows a 'For each' loop over 'Attraction' ordered by 'CountryName'. The 'Order' property of the grid is highlighted in red. A dialog box on the right shows the 'Order' property of the grid set to 'CountryName'.

Order

また、グリッドには **[Order]** プロパティがあります。このプロパティは For each コマンドの Order 節に相当します。

たとえば国名順にする場合、[Order] プロパティに CountryName 項目属性を入力します。

項目属性を含むグリッドは For each コマンドと同等

国番号

観光名所ふりがな

観光名所ふりがな

観光名所番号	観光名所名	国名	観光名
10	スミソニアン博物館	アメリカ合衆国	
5	コルコバードのキリスト像	ブラジル	
4	エッフェル塔	フランス	
2	ルーブル美術館	フランス	
3	万里の長城	中国	
11	東京スカイツリー	日本	

Web Panel WWAttractionsFromScratch Navigation Report

Name: WWAttractionsFromScratch
Description: WWAttractions From Scratch
Environment: C# Default (C#)
Spec. Version: 15_0_6-116888
Form Class: HTML
Program Name: WWAttractionsFromScratch
Parameters:

Warnings

spc0038 There is no index for order CountryName; poor performance may be noticed in grid 'Grid1'.

FILL &CountryId with CountryId, CountryName in

=Country (CountryId) INTO CountryId CountryName Order CountryName

ダイナミックコンボをロードするナビゲーション

Event Load

Order: CountryName
! No index

Navigation filters: Start from: FirstRecord
Loop: NotEndOfTable
while:

Join location: Server

=Attraction (AttractionId)
=Country (CountryId)

グリッドのベーステーブル

F5 キーを押すと、グリッドが国名順になっていることが分かります。

この Web パネルのナビゲーションリストは、&CountryId 変数のコンボボックスをロードするために行う必要があるナビゲーションを示しています。これは今まで使用してこなかったものです。

また、ここでは、グリッドのロードに必要なナビゲーションも示しています。このナビゲーションは、For each コマンドと同じです。Attraction テーブルがベーステーブルに選択され、CountryName ([Order] プロパティの項目属性) 順に処理します。この問い合わせでは、テーブル全体を処理し、Attraction テーブルの各レコードはロードされ、観光名所の CountryName を表示するために Country テーブルにアクセスします。

グリッドデータのフィルタリング

The screenshot shows the GeneXus IDE interface. On the left, a web form is visible with a dropdown menu for 'CountryID' and a grid below it. The grid has three columns: 'AttractionID', 'AttractionName', and 'CountryName'. In the center, a code editor shows a loop structure: 'For each Attraction order CountryName where CountryId = &countryId Print Attractions endfor'. On the right, the 'Properties' window for the 'Grid' control is open. The 'Conditions' property is highlighted with a red box and contains the text 'CountryId = &CountryId;'. A red arrow points from this property to the grid in the web form. Below the screenshot, the text 'フィルタの条件' (Filter Condition) is written in red. In the bottom right corner, there is a small box containing the text 'F5'.

フィルタの条件

F5

ここまでで、変数については何も行っていません。しかし、変数は、グリッドに表示するデータを国別または観光名所名別にフィルタリングするために使用するつもりでした。

AttractionsList では国別にフィルタリングしました。このフィルタをグリッドに指定するにはどうすればいいでしょうか。[Conditions] プロパティを使用します。

グリッドデータのフィルタリング

国番号 アメリカ合衆国 ▼

観光名所ふりがな

観光名所ふりがな

観光名所番号	観光名所名	国名	観光名所写真
10	スミソニアン博物館	アメリカ合衆国	

初期表示

操作後

国番号 フランス ▼

観光名所ふりがな

観光名所ふりがな

観光名所番号	観光名所名	国名	観光名所写真
2	ルーブル美術館	フランス	
4	エッフェル塔	フランス	

コンボボックスの初期値は「アメリカ」で、グリッドにはアメリカ国内の観光名所のみが表示されます。

「フランス」を選択すると、画面が再表示され、グリッドが再びロードされ、今度はフランス国内の観光名所が表示されます。

グリッドデータの条件付きフィルタリング

国番号 &CountryID

観光名所ふりがな &AttractionFuriganaFrom

観光名所ふりがな &AttractionFuriganaTo

GRID

観光名所番号	観光名所名	国名	観光名所写真
AttractionID	AttractionName	CountryName	

プロパティ

全般 クラス

フィルタ

Control Info

Control Type	Dynamic Combo Box
Data Source From	Attributes
Item Values	CountryID
Item Descriptions	CountryName
Sort Descriptions	True
Conditions	

Instantiated Attributes

Empty Item	True
Empty Item Text	CX_EmptyItemText
Notify Context Change	False

Behavior

値: 「(なし)」

コンボボックスは、最初は値が選択されていない状態で開き、すべての国の観光名所を表示するとします。

そのためには、コンボボックスのプロパティを編集し、[Empty Item] プロパティを [True] に設定します。このようにすると、コンボボックスに「(なし)」オプションが追加されます。このオプションは空の値に対応します。

グリッドデータの条件付きフィルタリング

国番号 &CountryID

観光名所ふりがな &AttractionFuriganaFrom

観光名所ふりがな &AttractionFuriganaTo

GRID

観光名所番号 AttractionID

観光名所名 AttractionName

国名 CountryName

観光名所写真

Grid1のConditions

CountryID = &CountryID
when not &CountryID.IsEmpty();

条件付き条件

F5

プロパティ

全般 クラス

フィルタ

グリッド: Grid1

Control Name	Grid1
Collection	
Base Trn	Attraction
Order	countryName
Conditions	CountryID = &Co
Data Selector	(none)
Appearance	
Class	Grid
Custom Bonds	

[Conditions] プロパティを開き、コンボボックスの値が空ではないときにのみこの条件を適用するように指定します。空の場合は条件を適用しません。

グリッドデータの条件付きフィルタリング

国番号 (なし)

観光名所ふりがな

観光名所ふりがな

観光名所番号	観光名所名	国名	観光名所写真
10	スミソニアン博物館	アメリカ合衆国	
5	コルコバードのキリスト像	ブラジル	
4	エッフェル塔	フランス	
2	ルーブル美術館	フランス	
3	万里の長城	中国	
11	東京スカイツリー	日本	

国番号 フランス

観光名所ふりがな

観光名所ふりがな

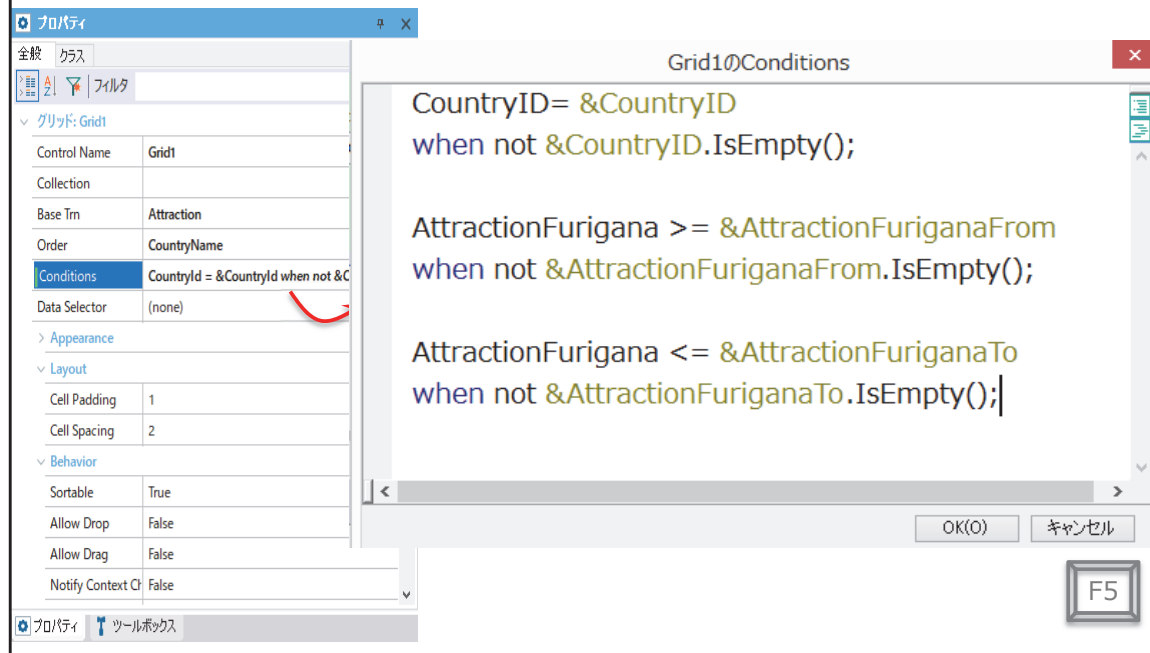
観光名所番号	観光名所名	国名	観光名所写真
2	ルーブル美術館	フランス	
4	エッフェル塔	フランス	

実行します。

コンボボックスに「(なし)」の値が表示されます。この場合、観光名所にフィルタが適用されず、すべてが表示されます。

ここで、たとえばフランスを選択すると、変数の値が空ではないため、フィルタが適用され、フランス国内の観光名所が表示されます。

複数の条件でグリッドデータをフィルタリング



このフィルタに、観光名所名（ふりがな）のフィルタも追加する必要があります。For each コマンドで 2 つの Where 節を使用していたフィルタは、グリッドでは、[Conditions] プロパティを使って追加します。

AttractionFurigana は、ユーザーがフォームの &AttractionNameFrom 変数に入力する値以上である必要があります。このフィルタも、ユーザーが変数に値を入力していない場合には適用しません。そこで When 節を使用します。なお、この節は、Grid の [Conditions] プロパティに限らず、For each コマンドの Where 節でも同じように使用できます。

もう 1 つ &AttractionNameTo のフィルタも追加します。

ソート条件による検索の最適化

Grid1のOrder

CountryID,AttractionFurigana when not &CountryID.IsEmpty()
AttractionFurigana

国番号 (なし)

観光名所番号 観光名所名 国名 観光名所写真

観光名所番号	観光名所名	国名	観光名所写真
10	スミソニアン博物館	アメリカ合衆国	
5	コルコバードのキリスト像	ブラジル	
4	エッフェル塔	フランス	
2	ルーブル美術館	フランス	
3	万里の長城	中国	
11	東京スカイツリー	日本	

条件付き順序

国番号 フランス

観光名所名 国名 観光名所画像

観光名所名	国名	観光名所画像
エッフェル塔	フランス	
ルーブル美術館	フランス	

ユーザーが国を選択した場合、CountryId 順で並べ替えた上で第 2 ソートキーに AttractionFurigana 順で並べ替えるよう Web パネルに指示します。ユーザーが国を未選択の場合は AttractionFurigana で並べ替えます。これは、テーブルレコード検索の最適化を意図しています。

この実装を行うには、グリッドの [Order] プロパティを編集して、コンボボックスでユーザーが国を選択しているかどうかを条件とし、国番号、観光名所名の順で指定します。国を選択している場合、データはその国によってフィルタリングされ、その国内の観光名所が観光名所名順で表示されます。ユーザーがコンボボックスを空 (「(なし)」値) のままにした場合、AttractionFurigana が、ソート順に利用されるようになります。

ここでは詳細は省きます。ここでこの説明をした目的は、グリッドのソート順にも条件を指定できることを示すことでした。For each コマンドの場合でも同様です。

実行します。AttractionFurigana 順になりました。フランスを選択すると、フランスの国番号順のインデックスが利用されフィルタリングされた上で、AttractionFurigana 順になります。

まとめると、観光名所は常に五十音順で表示されます。

フランスの観光名所の中で あ ～ そ のものを見る場合、[Conditions] プロパティに追加した 3 つの条件によるフィルタリングが行われてグリッドにロードされます。

まとめ

Web Form | Rules | Events | Conditions | Variables | Help | Documentation | Patterns

<選択されているアクショングループはありません>

MainTable CountryID

国番号 &Country1

観光名所ふりがな &AttractionFuriganaFrom

観光名所ふりがな &AttractionFuriganaTo

GRID

観光名所番号	観光名所名	国名	観光名所写真
AttractionID	AttractionName	CountryName	

プロパティ

Grid1	
Control Name	Grid1
Collection	
Base Trn	Attraction
Order	CountryId, AttractionName when not &Count...
Conditions	CountryId = &CountryId when not &CountryId...
Data Selector	(none)
Appearance	
Layout	
Cell Padding	1
Cell Spacing	2
Behavior	
Sortable	True
Allow Drop	False
Allow Drag	False
Notify Context Cl	False

プロパティ ツールボックス

ベーステーブル

For each Order Where

拡張テーブル

Web パネルを実装しました。ユーザーが値を入力するいくつかの変数に加え、項目属性を含む Grid コントロールを挿入しました。

項目属性はデータベース内の情報に対応するので、GeneXus は、検索が必要であることを理解します。項目属性を含むグリッドは For each に似ており、For each の場合と同様に、検索するテーブルのトランザクションレベルを指定する [Base Trn] プロパティがあります。このテーブルを**グリッドのベーステーブル**と言います。For each コマンドを使用する場合と同じように指定しなかった場合、使用されている項目属性に基づいて GeneXus が推測します。ただし、このような場合についてここでは取り上げません。

For each コマンドの場合と同様に、グリッドの項目属性はすべて、ベーステーブルの拡張テーブルに属している必要があります。For each では、Order 節を使用してデータの順序を指定し、1 つまたは複数の Where 節を使用して、問い合わせによって返されるデータをフィルタリングします。グリッドで同様の処理を行うには、それぞれ [Order] プロパティと [Conditions] プロパティを使用します。

イベント

Start/Refresh/Load
コントロールイベント

グリッドのロード: Load イベント

PDF プロシージャ

Web Form | Rules | Events | Conditions | Variables | Help | Documentation | Patterns

<選択されているアクショングループはありません>

MainTable CountryID

国番号 &Country1

観光名所ふりがな &AttractionFuriganaFrom

観光名所ふりがな &AttractionFuriganaTo

GRID

観光名所番号 AttractionID	観光名所名 AttractionName	国名 CountryName	観光名所写真 AttractionImage

For each
<メインのコード>
endfor

&SupplierQty = Count(SupplierName)
Print Attractions

Attractions

Attr	AttractionName	CountryName	AttractionImage	&SupplierQty

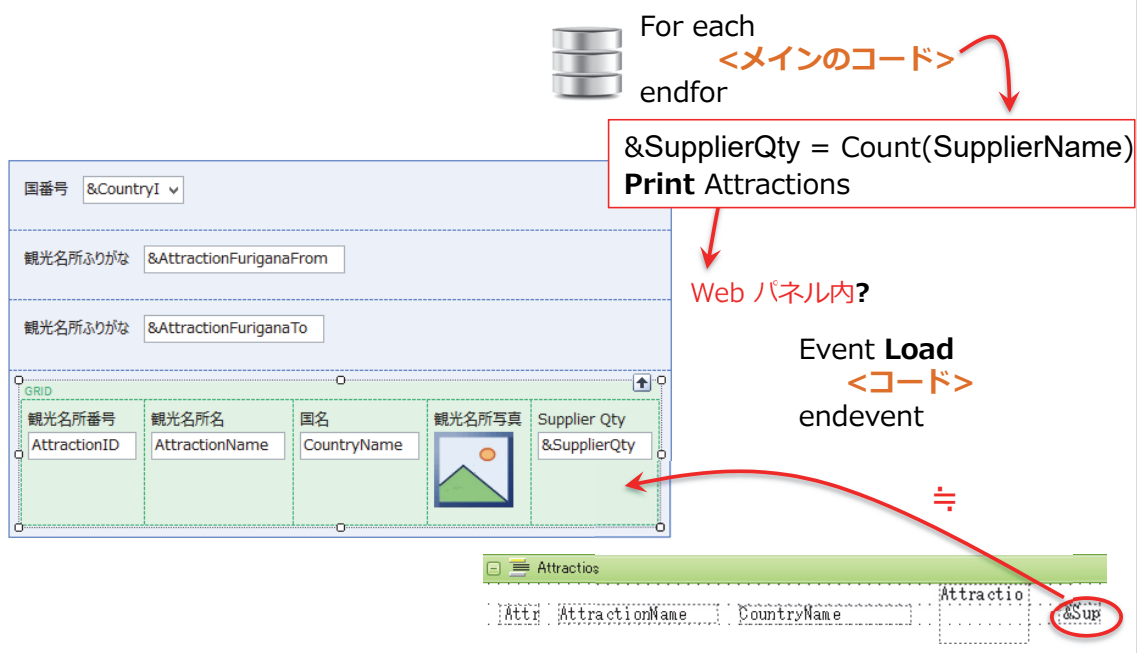
For each コマンドでは、条件を満たす各レコードの処理をメインのコードとしてプログラミングできました。
たとえば、観光名所のリストでは、Print Attractions コマンドで明細行の出力を指示します。これに対して、グリッドの場合、この処理は自動的に行われるので指定する必要がありません。

しかしながら、Grid メインのコードにプログラミングが必要になるケースがあります。
たとえば、旅行代理店と提携している提携先を記録する Supplier トランザクションがあるとしてします。

このとき、プロシージャオブジェクトとして実装する観光名所のリストでは、各観光名所に関連付けられている提携先数も表示するとします。そのためには、数値変数 &SupplierQty を定義し、For each コマンドの本体で (For each コマンドが、処理しようとしているベーステーブルのレコードに位置付けられているとき)、その観光名所を扱う提携先を数えた結果を割り当ててから、その変数をプリントブロックに含めて実装します。

グリッドのロード: Load イベント

PDF プロシージャ



一方、Web パネルで同じ処理を行うには、グリッドを右クリックし、[項目属性/変数を挿入] を選択します。タイプ Numeric(4,0) で &SupplierQty 変数を作成し、グリッド内の適切な位置まで移動します。

これは、プロシージャオブジェクトにおける変数のプリントブロックへの挿入に該当します。

ここで、計算方法はどのように指定すればいいのでしょうか。For each コマンドの場合、For each と Endfor の間の <メインのコード> で行いました。この場合はどこで指定するのでしょうか。

そのために、システムイベントの 1 つに **Load** イベントがあります。

このLoad イベントで、グリッドのベーステーブルの各明細行がグリッドにロードされる直前に実行する処理をプログラミングします。

ここでは、&SupplierQty 変数に値を割り当てます。

グリッドのベーステーブル内でフィルタリングの条件を満たすすべてのレコードについて、明細行がグリッドに追加される直前に Load イベントが自動的に実行されます。

式のベーステーブル:

```

Supplier
├── SupplierID
├── SupplierName
├── SupplierAddress
└── Attraction
    ├── AttractionID
    └── AttractionName
  
```

SupplierAttraction
SupplierId*
AttractionId*

AttractionId	AttractionName	CountryId	CityId	...
1	ルーブル美術館	2	1	
2	万里の長城	3	1	
3	エッフェル塔	2	1	

国番号 &CountryId

観光名所ふりがな &AttractionFuriganaFrom

観光名所ふりがな &AttractionFuriganaTo

GRID

観光名所番号	観光名所名	国名	観光名所写真	Supplier Qty
AttractionID	AttractionName	CountryName		&SupplierQty

Event Load
&SupplierQty = Count(SupplierName)
endevent

新しい
明細行

スライドの Load イベントの実装により、Load イベントが実行される時は、ベーステーブルとその拡張テーブルの各レコードが処理されます。ここでインライン Count(SupplierName)は (すべての提携先のレコードをカウントせずに) これからグリッドにロードしようとしている Attraction レコードの AttractionId と一致する SupplierAttraction テーブルのレコードのみをカウントします。

【補足】 SupplierName 項目属性は Supplier テーブルに属していますが、GeneXus は、式のベーステーブルとして Supplier テーブルではなく SupplierAttraction テーブルを選択します。
詳細は省きますが、SupplierName は Attraction テーブルの関連がある SupplierAttraction の拡張テーブルに含まれます。
GeneXus は、データの関連付けが可能なテーブルを探しました。

デモ



観光名所番号	観光名所名	国名	観光名所写真	提携先数
4	エッフェル塔	フランス		0
5	コルコバードのキリスト像	ブラジル		0
10	スミソニアン博物館	アメリカ合衆国		0
11	東京スカイツリー	日本		0
3	万里の長城	中国		1
2	ルーブル美術館	フランス		4

グリッド外の
変数に総数を表示

提携先総数: 5



前ページまでで説明したことを GeneXus で実装します。Supplier トランザクションは作成済みです。

Web パネルの [Events] エlementに移動します。コンボボックスに、事前定義済みのイベントが表示されます。これらは、特定の時点で実行されるシステムイベントであり、コードをプログラミングできるものです。

そのシステムイベントの 1 つが Load イベントです。ここで、Attraction テーブルの明細行をグリッドにロードする前に実行する処理をプログラミングします。実行します。

次に、グリッドに表示されている観光名所を取り扱う提携先の総数の表示を検討します。この列の値の合計の表示です。

【補足】変数&SupplierQtyのディスクリプションを「提携先数」としています。

グリッド外の変数を使用した総数の計算

AttractionId	AttractionName	CountryId	CityId	...
1	ルーブル美術館	2	1	
2	万里の長城	3	1	
3	エッフェル塔	2	1	

Event Load

```
&SupplierQty = Count(SupplierName)
&TotalSupplierQty = &TotalSupplierQty + &SupplierQty
```

Endevent

国番号 &CountryId

観光名所ふりがな &AttractionFuriganaFrom

観光名所ふりがな &AttractionFuriganaTo

GRID

観光名所番号	観光名所名	国名	観光名所写真	提携先数
AttractionID	AttractionName	CountryName		&SupplierQty

Total Supplier Qty &TotalSupplierQty

&SupplierQty 2

&TotalSupplierQty 2


変数が表示する値を計算する効率的な方法として、グリッドに明細行をロードするたびに、その行の &SupplierQty 変数の値を、その時点での &TotalSupplierQty の値に加算します。

つまり、Load イベントで &SupplierQty の値を計算後、&TotalSupplierQty に、その時点で含まれる値に &SupplierQty 変数の値を加算した値を割り当てます。

グリッド外の変数を使用した総数の計算

AttractionId	AttractionName	CountryId	CityId	...
1	ルーブル美術館	2	1	
2	万里の長城	3	1	
3	エッフェル塔	2	1	

国番号 &CountryId
観光名所ふりがな &AttractionFuriganaFrom
観光名所ふりがな &AttractionFuriganaTo
GRID

観光名所番号	観光名所名	国名	観光名所写真	提携先数
AttractionID	AttractionName	CountryName		&SupplierQty
				

Total Supplier Qty &TotalSupplierQty

Event Load
&SupplierQty = Count(SupplierName)
&TotalSupplierQty = &TotalSupplierQty + &SupplierQty
Endevent

&SupplierQty 1
&TotalSupplierQty 3

2 行目をロードするとき、&SupplierQty の値が計算されます。&TotalSupplierQty には以前の値が含まれ、この値に &SupplierQty 変数の値が加算されます。以後、同様です。

最後の明細行がロードされると、&TotalSupplierQty 変数が必要な値になります。

試行

初期表示

国番号 (なし) ▼

観光名所ふりがな

観光名所ふりがな

観光名所番号	観光名所名	国名	観光名所写真	提携先数
4	エッフェル塔	フランス		0
5	コルコバードのキリスト像	ブラジル		0
10	スミソニアン博物館	アメリカ合衆国		0
11	東京スカイツリー	日本		0
3	万里の長城	中国		1
2	ルーブル美術館	フランス		4

Total Supplier Qty 5

フランスで絞り込み

国番号 フランス ▼

観光名所ふりがな

観光名所ふりがな

観光名所番号	観光名所名	国名	観光名所写真	提携先数
4	エッフェル塔	フランス		0
2	ルーブル美術館	フランス		4

Total Supplier Qty 9

フランスで絞り込んだときに、4 ではなく 9 と表示されるのはなぜか？

実行してみます。

2 つのことが分かります。まず、総数が正しく計算されています。次に、この値は変数であるため、ユーザーが値を変更できる入力方式になっていますが、これでは意味がありません。そこで、最初にこの変数を読み取り専用を設定します。

そのためには、フォーム内で変数をクリックし、[ReadOnly] プロパティを [True] に変更します。

(&SupplierQty も変数ですが、何もしなかったのに読み取り専用として表示されたことが不思議かもしれません。どのイベントも明細行レベルでプログラミングされない場合、グリッドの変数は常に読み取り専用になります。)

ここで、たとえばフランスでフィルタリングするとどうなるかを見えます。

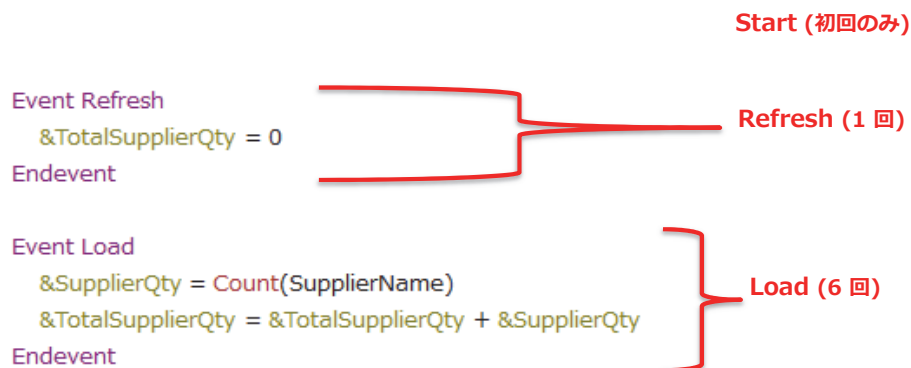
総数が 4 ではなく、9 と表示されます。これは、以前に表示されていた値 5 に、現時点で表示されているはずの 4 が加算された値です。なぜ、このようなことになったのでしょうか。

Refresh イベント

システムイベントの 1 つ

発生タイミング:

- DB で検索する直前
- **Load** イベントの前段階



フィルタ変数を 1 つ変更した後、Web パネルにグリッドが再びロードされたため、データベース内の Attraction テーブルに対して再び問い合わせが実行され、フィルタの条件を満たすすべてのレコードについて Load イベントが実行されました。

問題は、グリッドのロードを開始する前に &TotalSupplierQty 変数をゼロにリセットしていなかったことです。

これはどこで行えばいいのでしょうか。それは **Refresh イベント**です。リセットを行い F5 キーを押します。

提携先の総数は読み取り専用になっています。この Web パネルを初めて実行するとき、3 つのイベントが順番にトリガーされます。**Start イベント**は、Web パネルが初めて開くときにのみ実行されます。**Refresh イベント**は、変数をゼロに設定します。**Load イベント**は、グリッドにロードする行の数だけ実行されます。この例では、6 回でした。

Refresh イベント

- フィルタに関連する変数の値を変更すると、**Refresh** イベントと **Load** イベントが再びトリガーされる (**Start** イベントはトリガーされない)。

国番号 フランス ▼

観光名所ふりがな

観光名所ふりがな

観光名所番号	観光名所名	国名	観光名所写真	提携先数
4	エッフェル塔	フランス		0
2	ルーブル美術館	フランス		4

Total Supplier Qty 4

Refresh (1 回)
Load (2 回)

国、たとえばフランスでフィルタリングすると、提携先数が正しく計算されます。

フィルタに利用する変数の値を変更すると、Refresh イベントが再びトリガー (これに伴い &TotalSupplierQty 変数がゼロにリセット) されます。次に、データベースにアクセスしてレコードがフィルタリングされ、グリッドに再びロードされます。このグリッドにロードする過程で、フィルタされたフランス国内の観光名所ごとに **Load** イベントがトリガーされます。

パターンの Work With Attractions エlement

データに対するアクション: 観光名所の挿入、更新、削除

Attraction

```
parm(in:&Mode, in:&AttractionId);
AttractionId = &AttractionId if not &AttractionId.IsEmpty();
```

TrnMode ドメイン: Enum Values Insert, Insert (INS), Update, Update (UPD), Delete, Delete (DLT), Display, Display (DSP)

ここまでで実装した Web パネルを見ると、WorkWith パターンを Attraction トランザクションに適用して作成したオブジェクトに似ていることが分かります。

当然、このオブジェクトは Web パネルでした。

最も興味深い点として、WorkWith では、グリッドデータのフィルタリングができるだけでなく、データに対してアクションを実行できます。たとえば、観光名所の挿入、更新、削除が可能です。

このために、パターンはグリッドの明細行レベルで 2 つのコントロール (更新、削除) を挿入し、別のコントロール (挿入) をグリッド外に挿入しました。このいずれの場合も、各コントロールに対応するアクションでは、Attraction トランザクションを呼び出し、データの追加、更新、削除のために**モード**をパラメーターとしてこのトランザクションに送信します。更新と削除については、グリッドの各明細行に対応するため、観光名所の識別子を 2 番目のパラメーターとしてトランザクションに送信することで、**該当する観光名所**を更新または削除する画面を開くことができます。グリッド外の Insert コントロールについては、2 番目のパラメーターとして 0 が送信されます。INS (挿入) モードの観光名所は自動採番されます。

上記のとおりパラメーターを受け取るため、パターンで Attraction トランザクションが変更され、Parm ルールなどが追加されました。見てのとおり、2 つの変数を受け取ります。&Mode 変数は 3 文字のトランザクションの標準変数で、TrnMode 列挙型ドメインに示す 4 つの値のいずれかを受け取ります: 挿入 (INS)、更新 (UPD)、削除 (DLT)、表示 (DSP)。

4 つの値のいずれかを受け取ると、トランザクションはどのモードで開くのが分かります。

また、2 番目のパラメーターとして、更新、削除、表示する観光名所番号を &AttractionId 変数として受け取ります。

グリッド明細行の更新アクション

国番号 &CountryI

観光名所ふりがな &AttractionFuriganaFrom

観光名所ふりがな &AttractionFuriganaTo

観光名所番号	観光名所名	国名	観光名所写真	提携先数	Update
AttractionID	AttractionName	CountryName		&SupplierQty	

Total Supplier Qty &TotalSupplierQty

Event Start
 &Update.FromImage(updateIcon)
 Endevent

Event &Update.Click
 Attraction(TrnMode.Update, AttractionId)
 Endevent

画像をナレッジベースに追加

UpdateIcon.png
 (既定)
 サイズ: 25px,33px

&Update 変数 (Image タイプ)

この章で扱っていた Web パネルに、このいずれかのアクションをトランザクションに実装します。たとえば、更新の場合を考えます。データに対するアクションの例を示します。

グリッドにコントロールを挿入する必要があります。パターンの場合では、updateという文字変数を挿入し、実行時に表示する「更新」というテキストが割り当てられています。一方、この Web パネルでは画像を挿入するので、素材となる画像をまず、ナレッジベースに挿入します。名前を updateIcon にします。

画像挿入後、Web パネルに戻り、ツールボックスの [項目属性/変数コントロール] をグリッドの最後の列にドラッグしてください。新しい変数を、文字ではなく Image

タイプで &Update として定義します。[Title] プロパティを削除して列見出しとして表示されないようにし、ナレッジベースに挿入した画像を変数にロードします。

これはどこで行えばいいのでしょうか。

画像がグリッドの明細行ごとに変わる場合は、Load イベントを使用します。ただし、ここでは画像は変化せず、どの明細行でも同じであるため、Web パネルが開くときに 1 回だけ実行される **Start イベント**での実行が適切です。

この画像にイベントを関連付け、ユーザーが画像をクリックしたらイベントがトリガーされ、コードが実行されるようにします。このコードで Attraction トランザクションを呼び出します。

これには、いくつかの方法があります。1 つの方法として、&Update 変数コントロールの Click イベントを使用します。

この場合、ユーザーがある明細行の画像をクリックすると、このイベントに入力したコードが実行されます。ここで Attraction トランザクションを呼び出すことができます。Update モード、つまり TrnMode 列挙型ドメインの Update 値と、クリックされたグリッドの明細行の AttractionId の値を渡します。

デモ

Application Name

Reacts With Attractions From... — 観光名所

観光名所番号	観光名所名	国名	観光名所写真	優先度	更新
4	エッフェル塔	フランス		0	
5	コルコバードのキリスト像	ブラジル		0	
10	スミソニアン博物館	アメリカ合衆国		0	
11	東京スカイツリー	日本		0	
3	万里の長城	中国		1	
2	ルーブル美術館	フランス		4	
Total Supplier Qty				5	

観光名所

観光名所番号 6

観光名所名 スカイツリー

観光名所ふりがな ぽかいっりー

国番号 5

国名 日本

都市番号 1

都市名 東京

Start Refresh Load

実行してみます。

最初に、&SupplierQty 変数の列が編集可能になっていることが分かります。これまでの実行時に既に読み取り専用になっていたため、明示的に読み取り専用として定義していませんでした。前述のように、グリッド変数は最初は読み取り専用として追加されます。

ただし例外として、イベントがこのように明細行レベルで定義されている場合や、その他の一定の条件を満たした場合を除きます。ここでは詳細は省きます。次に実行したときのために [Read only] プロパティを True に設定します。

国、たとえばフランスを選択します。エッフェル塔の隣の更新アイコンをクリックします。更新 (UPD) モードでトランザクションが表示されます。変更を加えます。たとえば、エッフェル塔をスカイツリーに変更します。保存します。別の章で適用済の Work With パターンによって、呼び出し元に戻る Return コマンドがトランザクションに追加されているため、Web パネルに戻ります。この Return コマンドの実行時は、初めて Web パネルを呼び出すときと同様に、Start イベントが実行されてから、続けて Refresh イベントと Load イベントが、ロードされるレコード数に応じた回数だけ実行されます。

非表示化：グリッドに列を含めることができない場合

Event `&Update.Click`
`Attraction(TrnMode.Update, AttractionId)`
 Endevent

国番号 `&CountryI` ▼

観光名所ふりがな `&AttractionFuriganaFrom`

観光名所ふりがな `&AttractionFuriganaTo`

GRID

観光名所番号	観光名所名	国名	観光名所写真	提携先数	Update
AttractionID	AttractionName	CountryName		&Supp	

Total Supplier Qty `&TotalSupplierQty`

Appearance

Auto Resize	True
Format	Text
Visible	False
Tooltip Text	
Invite Message	

グリッドに含まれる必要がある。表示しない場合は非表示にできる。

AttractionId 項目属性をグリッドに追加していなかったら、どうなるでしょうか。更新する画像をクリックするとき、パラメーターとしてトランザクションに送信される AttractionId は何でしょうか。送信できる値がありません。



AttractionId は、明細行のロード後にトリガーされる明細行レベルのイベントに使用されるため、グリッドから削除することはできません。この時点ではもうデータベースを参照していません。Load イベントによるロードによって、グリッドにはすべての列の値のみが保存されます。その後のイベントでは、グリッドにロードされているデータに対してのみ機能します。この列をグリッドに表示する必要がない場合は、非表示にできます。非表示でも存在はします。そのために、**[Visible] プロパティの値を [False] に設定**します。

明細行レベルの別のアクション: 明細行の再表示

国番号

観光名所名

観光名所名

観光名所番号	観光名所名	国名	観光名所写真	提携先数	Update	New Supplier
AttractionID	AttractionName	CountryName		&SupplierQty		&NewSupplier
						

Total Supplier Qty

明細行の観光名所を含む新しい提携先をデータベースに作成

Character(10)

```

| Event Start
  &Update.FromImage(UpdateIcon)
  &NewSupplier = "提携先追加"
- EndEvent

Event &NewSupplier.Click
/* SupplierAttractionに1レコード追加するプロシーダを呼び出す*/
&SupplierQty = NewSupplier(AttractionID)
EndEvent

```

明細行レベルで別のアクションを追加します。ただし、このアクションは、Attraction トランザクションを呼び出す場合のように、インターフェースを持つ別のオブジェクトを呼び出しません。
今回は、クリックした明細行の観光名所に基づき、その観光名所を特定の提携先で取り扱うようにレコードを作成することにします。

まず、10 文字の新しい変数 newSupplier をグリッドに追加します。

これを読み取り専用に変更します。この変数のラベルとする「新しい提携先」のテキストは、明細行によって変わらないので、Start イベントで割り当てます。次に、この変数の Click イベントを定義します。

ユーザーが [提携先追加] をクリックしたときにどのような処理を実行するかを考えます。

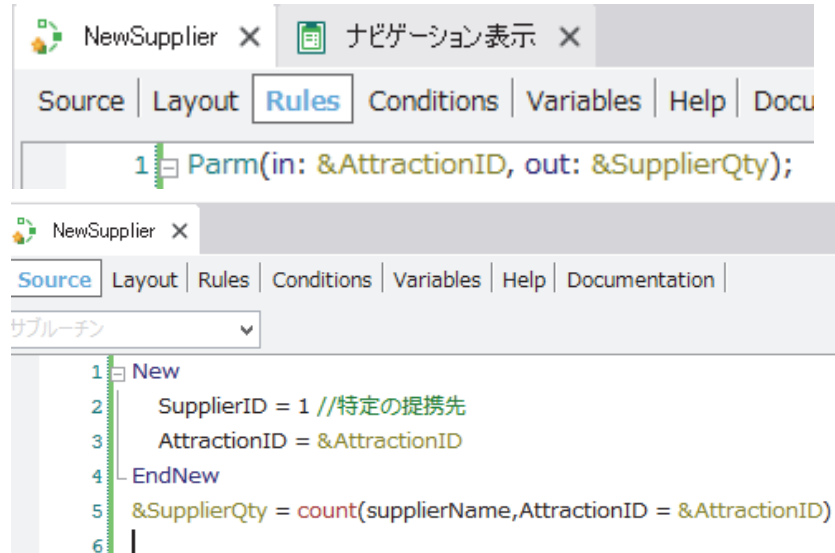
明細行レベルの別のアクション: 明細行の再表示

Event &NewSupplier.Click

/* SupplierAttractionに1レコード追加するプロシージャを呼び出す*/

&SupplierQty = NewSupplier(AttractionID)

EndEvent



例として、ここでは、プロシージャーを呼び出してその明細行の観光名所番号を渡し、その観光名所を含む提携先を作成します。

実装には **new** コマンドを使用して、Supplier テーブルと SupplierAttraction テーブルにそれぞれレコードを直接作成しています。このソリューションは各コマンドの使用法を示しています。

ただし、より推奨される方法として、Supplier のビジネスコンポーネントを挿入に使用することもできます (このコースでは、2 レベルのビジネスコンポーネントによるデータ更新を扱っていませんが、これもシンプルに実装することができます)

Supplier のヘッダーと明細行の挿入後に、このプロシージャーオブジェクトの 10 行目では、この観光名所を含む提携先数を計算します。

インライン式は、For each コマンドの外側で (=メインのコードの内側ではないタイミングで)トリガーされるため、**関連する**観光名所を扱う提携先をフィルタリングする条件を明示的に指定する必要があります。

このインライン式の計算結果は、プロシージャーの戻り値とします。&newSupplier コントロール (変数) の Click イベントから、このプロシージャーを呼び出し、クリックされた明細行の AttractionId を渡します。

返されるパラメーターを &SupplierQty 列に表示するために、変数に直接割り当てます。

論理的には、ユーザーが [提携先追加] をクリックすると、[提携先] 列の値が以前の値に 1 を加算したものになる必要があります。つまり、明細行を再表示する必要があります。

デモ

観光名所名	国名	観光名所写真	提携先数 Update New Supplier
エッフェル塔	フランス		0  提携先追加

`&SupplierQty = NewSupplier(AttractionID)`



クリックした結果、
この明細行の提携先数は
再表示された。

観光名所名	国名	観光名所写真	提携先数 Update New Supplier
エッフェル塔	フランス		1  提携先追加

Total Supplier Qty

6



総数は更新されていない。

実行してみます。

たとえば、フランスでフィルタリングし、現時点ではどの提携先も扱っていないエッフェル塔に注目します。[提携先追加] をクリックします。明細行が自動的に再表示されることが分かります。

そして、エッフェル塔を扱う提携先数が表示されます: 1 です。

しかし、総数は再表示されません。2 であるはずですが、値が変わりません。なぜでしょうか。

&newSupplier 変数に関連付けられている Click イベントのクリック時には、そのコードだけが実行されました。

その中でグリッドの変数に値が割り当てられ、明細行が再表示されましたが、**その明細行**だけでした。

Load イベントを含むその他のイベントは実行されませんでした。

Refresh コマンド

- オプション 1

```

| Event &NewSupplier.Click
  &SupplierQty = NewSupplier(AttractionID)
  &TotalSupplierQty = &TotalSupplierQty + 1
- EndEvent

```

- オプション 2

```

Event &NewSupplier.Click
  &SupplierQty = NewSupplier(AttractionID)
  Refresh
EndEvent

```

Refresh イベントと
Load イベントのトリガー

```

Event Refresh
  &TotalSupplierQty = 0
Endevent

Event Load
  &SupplierQty = Count(SupplierName)
  &TotalSupplierQty = &TotalSupplierQty
Endevent

```



フォーム全体が再表示される

このイベントの実行時に、グリッドにロードされている提携先の総数を最新の状態に維持するには、2 通りの方法があります。

1 つ目は、&TotalSupplierQty に 1 を加算する方法です。
プロシージャーはこの観光名所への提携先を 1 つだけ追加したからです。

しかし、将来的にさらに提携先を追加できるようにプロシージャーが変更された場合、このイベントに合わせて変更を反映する必要があります。

より適切なソリューションは、Refresh イベントと Load イベントを再び実行するように Web パネルにリクエストする方法です。そのためには、Refresh コマンドを使用します。この場合、グリッドが再びデータベースからロードされます。

まとめ

- Web パネル内のグリッド (項目属性を含む)

システムイベント

Start

Refresh

Load (n 回)



ベーステーブル

For each
Order
Where

項目属性を含むグリッドを持った Web パネルの場合、GeneXus は、グリッドに関連付けられたベーステーブル (つまり、グリッドの明細行をロードするためにナビゲートすべきテーブル) があることを理解します。ベーステーブル内のレコードごとに明細行がロードされます。フィルタリングを行うには、グリッドの [Conditions] プロパティ、並べ替えるには [Order] プロパティを使用します。ベーステーブルを持つグリッドは For each と同様です。

Web パネルで実行されるシステムイベントではコードをプログラミングできます。Web パネルが開いたとき、つまり最初の実行時に必ずトリガーされるイベントを 3 つ見てきました。

- **Start** は 1 回だけトリガーされます。ここでは、たとえば変数を初期化できます。
- **Refresh** は画面データの再表示の際にトリガーされます。このイベントに続いて、データベースへのアクセスが行われ、ベーステーブルとその拡張テーブルのデータが取得されます。
- **Load** イベントは、グリッドにロードするベーステーブルのレコードごとに発生します。このため、明細行がグリッドに実際にロードされる前に実行する処理はすべてここでプログラミングする必要があります。グリッドにロードされるデータは、グリッドに含まれる表示/非表示の列のデータのみです。

これらのイベントのあと、クライアントサイドの画面には DB から取得された情報がロードされ、DB との接続は解除されます。

まとめ

- Web パネル内の**グリッド** (項目属性を持つ)

国番号 &Country1

観光名所ふりがな &AttractionFuriganaFrom

観光名所ふりがな &AttractionFuriganaTo

GRID

観光名所番号	観光名所名	国名	観光名所写真	提携先数	Update	New Supplier
AttractionID	AttractionName	CountryName		&SupplierQty		&NewSupplier

Total Supplier Qty &TotalSupplierQty

システムイベント

Start

Refresh

Load (n 回)

ユーザーイベント/コントロールイベント
Refresh コマンド

Refresh

Load (n 回)

(重要) RefreshコマンドはRefreshイベントとLoadイベントが発生

一方Web パネルでは、最初の実行時にトリガーされるシステムイベント以外のイベントも定義できます。これらのイベントは、Web パネルのロード後に、必ずユーザー操作の結果としてトリガーされます。以前の章で定義した Web パネルでも、ボタンにユーザーイベントを関連付けました。

今回のケースでは、画像 (&update) でプログラミングしている Click イベントや、[新しい提携先] のクリックなどがあります。

これらのイベントを**ユーザーイベント**または**コントロールイベント** (前述の Click イベント) と呼びます。ユーザーがイベントをトリガーするとき、そのコードだけが実行され、画面は再表示されません。唯一の例外が、イベントがグリッドの明細行レベルで発生する場合です。&newSupplier では、同じグリッド内の変数 (この例では &SupplierQty) がコード内で割り当てられます。この場合、変数の値は画面上で再表示され、更新された値を示します。

再び Refresh を実行し、グリッドの明細行をデータベースからロードする必要性が生じたら (たとえば合計行数の更新)、イベント内に **Refresh コマンド** を記述できます。

補足：グリッドがない場合

- フォーム内に**グリッド**を含まないが**項目属性を持つ** Web パネル

Parm(in: AttractionId);

1 レコードのみがロードされる

AttractionId	AttractionName	CountryId	CityId	...
1	ルーブル美術館	2	1	
2	万里の長城	3	1	
3	エッフェル塔	2	1	

ベーステーブル

項目属性を持つグリッドを含む Web パネルの例を見てきました。ここで、フォーム内にグリッドがなく、項目属性がある Web パネルの場合を考えます。

Web パネル WWAttractionsFromScratch で観光名所の名前をクリックすると、その観光名所に関する全データを表示する Web パネルを呼び出すとします。

呼び出し先の Web パネルのフォームに、表示する観光名所の項目属性を挿入しています。

また、パラメーターを受け取るための parm ルールを定めています。ここでは、変数ではなく、AttractionId 項目属性で受け取ることにしました。

この Web パネルが、呼び出し元の Web パネルの AttractionName の Click イベントから呼び出され、明細行の観光名所番号が渡されると、GeneXus は自動的に Attraction テーブルにアクセスしてその観光名所番号の観光名所を取得し、その観光名所について、フォーム内の項目属性に関する情報を表示するように実装します。

まとめると、フォーム内にグリッドがなく項目属性を含む Web パネルにもベーステーブルがあります。ベーストランザクションがない場合、GeneXus ではどのようにこのことを判断するのでしょうか。これについてはこのコースでは説明しませんが、ベーストランザクションを指定しない For each の場合に似ています。

ただし、ここでは、グリッドがないため、ベーステーブルとその拡張テーブルから 1 レコードだけがロードされます。そのレコードが返されるようにするためのフィルタはどこで指定するのでしょうか。

この例では、AttractionId の値を項目属性で受け取ります。ここで、ベーステーブルから取得するレコードを指定する自動フィルタを指定しています。

補足：グリッドがない場合

- フォーム内に**グリッド**を含まないが**項目属性を持つ** Web パネル

Parm(in: &AttractionId);

AttractionId	AttractionName	CountryId	CityId	..
1	ルーブル美術館	2	1	
2	万里の長城	3	1	
3	エッフェル塔	2	1	

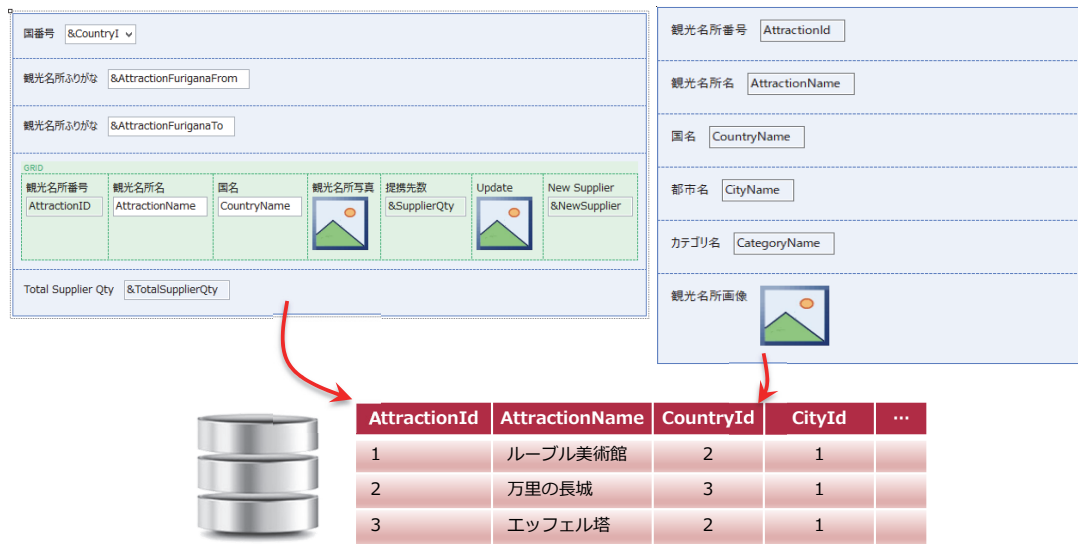
ベーステーブル

等価演算子 (=) 以外の条件を設定する必要がある場合や、項目属性ではなく変数で受け取った場合は、[Conditions] エレメントでフィルタを設定できます。

ベーステーブルを持つ/持たない Webパネル

これまで ベーステーブルを持つ Web パネル

項目属性が利用されている
⇒ 暗黙的なFor each(データベース参照)



ここまでで、ベーステーブルを持つ Web パネルを 2 種類見てきました。

- 1 つ目のケースは 1 つのグリッドを使用したものでした。この場合、グリッドのベーステーブルは Web パネルのベーステーブルです。このタイプの Web パネルの場合、グリッドも、その外側の項目も、一緒にナビゲートされ同じベーステーブルが自動的に判断されます。
- 2 つ目のケースはグリッドを含まず、項目属性を含むものでした。この場合、Web パネルのベーステーブルは項目属性だけで判断されます。

ここから ベーステーブルを持たない Web パネル

項目属性が利用されていない。例：変数のみ
⇒暗黙的For eachなし

ここで、ベーステーブルを持たない Web パネル、つまりデータベースへの問い合わせが**自動的**にプログラミングされない Web パネルを見ていきます。

最も分かりやすい例が、データベースに対する問い合わせがまったくない場合です。たとえば、スライドに示した Web パネルは、ユーザーにデータを求め、ほかのオブジェクトを呼び出したただけでした。

このようなとき自動的なデータベース参照はありません。

データベースに対する特殊な問い合わせや、CSVファイルを読み込むなど、開発者がプログラミングによりデータを画面への表示する必要がある場合があります。

そのようなとき「ベーステーブルなし」のWEBパネルで実現することができます。
(CSV、エクセルファイル、WEBサービスの内容をSDTに割り当てた上で、WEBパネルにSDTを配置するというやり方もありますが、このコースでは取り扱いません。)

ここから ベーステーブルを持たない Web パネル

項目属性が利用されていない。例：変数のみ
⇒暗黙的For eachなし



ここで、ベーステーブルを持たない Web パネル、つまりデータベースへの問い合わせが**自動的**にプログラミングされない Web パネルを見ていきます。

最も分かりやすい例が、データベースに対する問い合わせがまったくない場合です。たとえば、スライドに示した Web パネルは、ユーザーにデータを求め、ほかのオブジェクトを呼び出したただけでした。

このようなとき自動的なデータベース参照はありません。

データベースに対する特殊な問い合わせや、CSVファイルを読み込むなど、開発者がプログラミングによりデータを画面への表示する必要がある場合があります。

そのようなとき「ベーステーブルなし」のWEBパネルで実現することができます。
(CSV、エクセルファイル、WEBサービスの内容をSDTに割り当てた上で、WEBパネルにSDTを配置するというやり方もありますが、このコースでは取り扱いません。)

例：ベーステーブルを持たない Web パネル への変更

グリッドの項目属性⇒変数

観光名所番号 &AttractionID	観光名所名 &AttractionName	国名 &CountryName	観光名所写真 	提携先数 &SupplierQty	Update 	New Supplier &NewSupplier
-------------------------	--------------------------	--------------------	---	----------------------	---	------------------------------

イベントの項目属性⇒変数

```

| Event &NewSupplier.Click
    &SupplierQty = NewSupplier(&u>&AttractionID)
    &TotalSupplierQty = &TotalSupplierQty + 1
- EndEvent

| Event &update.Click
    Attraction(TrnMode.Update, &AttractionID)
- EndEvent

```

プロパティの項目属性⇒変数

プロパティ

全般 クラス

フィルタ

▼ グリッド: Grid1

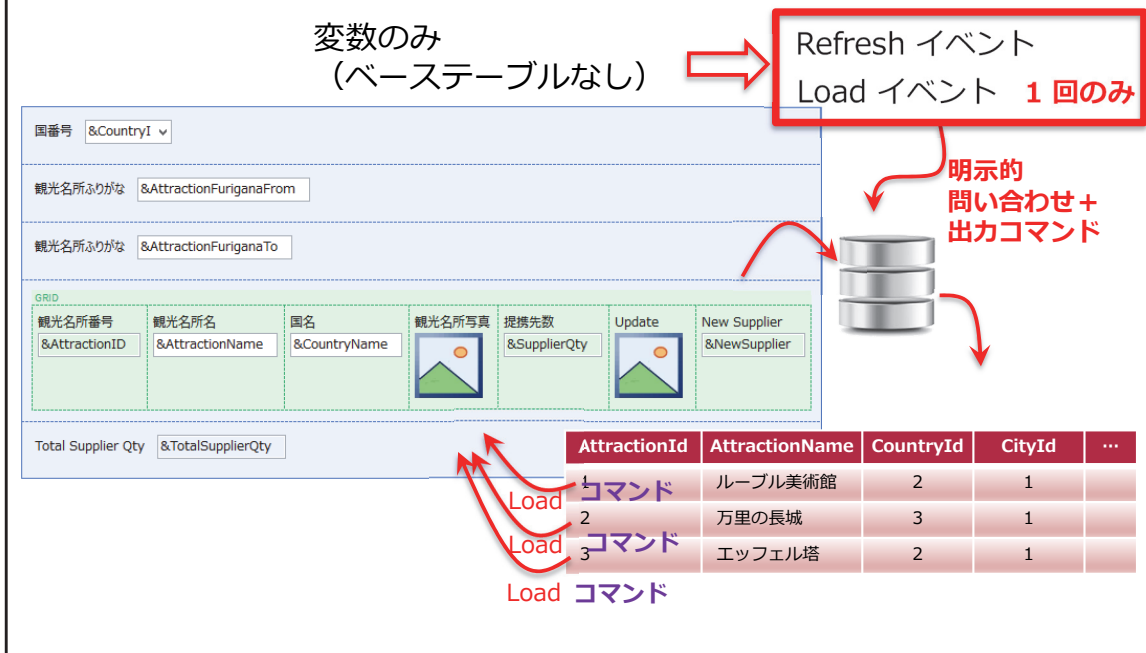
Control Name	Grid1
Collection	
Base Trn	
Order	
Conditions	
Data Selector	既定を使用(D)

前項で見てきたベーステーブルを持つ Web パネルは、「ベーステーブルなし」で実装することも可能です。

観光名所の一覧をグリッドに配置していた項目属性を変数に変更します。

イベント内で、AttractionId 項目属性を渡す呼び出しを &AttractionId 変数を渡す呼び出しに変更します。

例：ベーステーブルを持たない Web パネル への変更



変数のみがある場合、Attraction テーブルとその拡張テーブルをナビゲートし、レコードごとにグリッドに明細行をロードする必要があることを GeneXus はどのように判断するのでしょうか。

実際には、GeneXus にその判断はできません。このグリッドのロードは、項目属性を使用した場合のように自動的には行われません。

つまり、各レコードが処理されるような Load イベントは実行されないということです。問い合わせ処理がないので、もはや問い合わせ結果すらないからです。

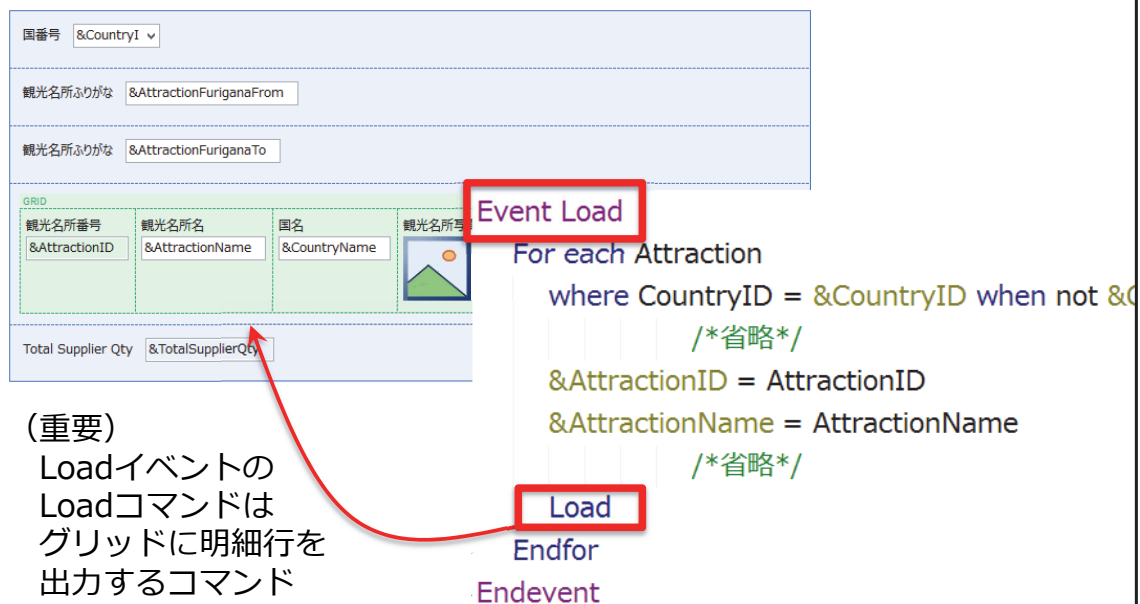
Load イベントはトリガーされますが、Refresh の後に 1 回のみで、データベースへのアクセスはありません。

この Load イベントのトリガー時にグリッドがロードされるように手動でプログラミングを行う必要があります。

データベース (この例では Attraction テーブル) へのアクセスを明示的にリクエストする必要があります。毎回、明細行がロードされることを指示する必要があります。

グリッド上に新しい明細行を出力するには、**Load コマンド**を使います (Load イベントではありません。コマンドです)。明細行の各カラムには、このコマンドが行われた時点で変数に格納されている値が挿入されます。この **Load コマンドは Load イベントでのみ記載が可能**です。Load イベントで、Load コマンドが実行される都度、グリッドに行が挿入されます。

例：ベーステーブルを持たない Web パネル への変更



国番号

観光名所ふりがな

観光名所ふりがな

GRID

観光名所番号	観光名所名	国名	観光名所写真
<input type="text" value="&AttractionID"/>	<input type="text" value="&AttractionName"/>	<input type="text" value="&CountryName"/>	

Total Supplier Qty

Event Load






```

For each Attraction
  where CountryID = &CountryID when not &CountryID
    /*省略*/
    &AttractionID = AttractionID
    &AttractionName = AttractionName
    /*省略*/
  Load
Endfor
Endevent
  
```

(重要)
Loadイベントの
Loadコマンドは
グリッドに明細行を
出力するコマンド

Load イベント内で、For each を使用して Attraction テーブルへのアクセスをプログラミングします。ここで Order 節を指定します。データが満たす必要がある条件は Where 節で指定します。フィルタに適合する各レコードについて、グリッド変数にその観光名所の値をロードします。すべての変数がロードされたら、**Load コマンド**を使用して、これらの値でグリッドに明細行を追加します。

例：ベーステーブルを持たない Web パネル への変更

Web Panel WWAtractionsFromScrachWithoutBasetable Navigation Report	
Name	 WWAtractionsFromScrachWithoutBasetable
Description	WWAtractions From Scrach Without Basetable
Environment	 Default (C#)
Spec. Version	 15_0_8-119728
Form Class	HTML
Program Name	WWAtractionsFrom
Parameters	
FILL &CountryID with CountryID , CountryName in	
 =Country (CountryID) INTO CountryID CountryName Order CountryName	
Event Load	
For Each Attraction (Line: 6)	
Order:	AttractionID
	Index: IATTRACTION
Navigation	Start from: FirstRecord
filters:	Loop while: NotEndOfTable
Constraints:	CountryID = &CountryID WHEN not &CountryID. isempty()
	 =Attraction (AttractionID)

ベーステーブルを持たない Web パネルのナビゲーションのリストを確認すると、Load イベントにナビゲーションを示す For each があることが分かります。

まとめ

- Web パネル内のグリッド (ベーステーブルを持つ)

The screenshot shows a web panel with the following elements:

- 国番号: &CountryI (dropdown)
- 観光名所ふりがな: &AttractionFuriganaFrom (text input)
- 観光名所ふりがな: &AttractionFuriganaTo (text input)
- GRID table with columns:
 - 観光名所番号 (AttractionID)
 - 観光名所名 (AttractionName)
 - 国名 (CountryName)
 - 観光名所写真 (AttractionPhoto)
 - 提携先数 (&SupplierQty)
 - Update (button with image)
 - New Supplier (button with image)
- Total Supplier Qty: &TotalSupplierQty (summary row)

Start
Refresh
Load (n 回)



ベーステーブル ≡ For each
Order
Where

ベーステーブルを持つ Web パネルの場合、Web パネルを開くと Start イベントが実行され、その直後に Refresh イベントが続き、その後、ベーステーブルへのアクセスが自動的に行われました。グリッドの [Conditions] プロパティに従ってフィルタリングが行われ、グリッドの [Orders] プロパティの項目属性に従って並べ替えが行われました。これは、暗黙の For each と考えることができます。開発者が何もしなくても GeneXus が内部で配置するものです。レコードが明細行としてグリッドにロードされるときは、ロード前に Load イベントが実行されます。これが、このスライドの例では**グリッドにロードされる明細行の数だけ Load イベントがトリガーされる理由**です。



まとめ

- Web パネル内のグリッド (ベーステーブルを持たない)

国番号 &CountryI ▼

観光名所ふりがな &AttractionFuriganaFrom

観光名所ふりがな &AttractionFuriganaTo

観光名所番号	観光名所名	国名	観光名所写真	提携先数	Update	New Supplier
&AttractionID	&AttractionName	&CountryName		&SupplierQty		&NewSupplier
						

Total Supplier Qty &TotalSupplierQty

Start Refresh Load (1 回のみ)

For each Order Where

Load コマンド

これに対して、Web パネルにベーステーブルがない場合、グリッドには変数のみがあり、暗黙の For each はありません。

Web パネルを開くと、Start イベントと Refresh イベントが実行されるのは同じですが、Load は 1 回のみ実行されます。データベースにアクセスして情報を取得する必要がある場合は、このイベント内で明示的に行う必要があります。つまり、ここで For each を記述し、Order 節と Where 節を使用します。各明細行のロードには、1 行ずつ明示的に **Load コマンド** を使用します。

レスポンスィブWEBデザイン

Application Name

by Application Name

Recents View Attraction Fr... WWAttractions From... 観光名所

SHOW FILTERS 観光名所s 観光名所名

観光名所番号	観光名所名	観光名所所在地	国名	都市名	カテゴリ名	観光名所画像		
3	エッフェル塔	エッフェル塔	フランス	パリ	モニュメント		更新	削除
4	コルコバードのキリスト像	コルコバードのキリスト像	ブラジル	リオデジャネイロ	モニュメント		更新	削除
6	スカイツリー	スカイツリー	日本	東京	有名なランドマーク		更新	削除
5	スミソニアン博物館	スミソニアン博物館	アメリカ合衆国	ワシントン	美術館		更新	削除
1	ルーブル美術館	ルーブル美術館	フランス	パリ	美術館		更新	削除
2	万里の長城	万里の長城	中国	北京	モニュメント		更新	削除

Application Name

Recents View Attraction Fr... WWAttractions From... 観光名所

SHOW FILTERS 観光名所s 観光名所名

観光名所名

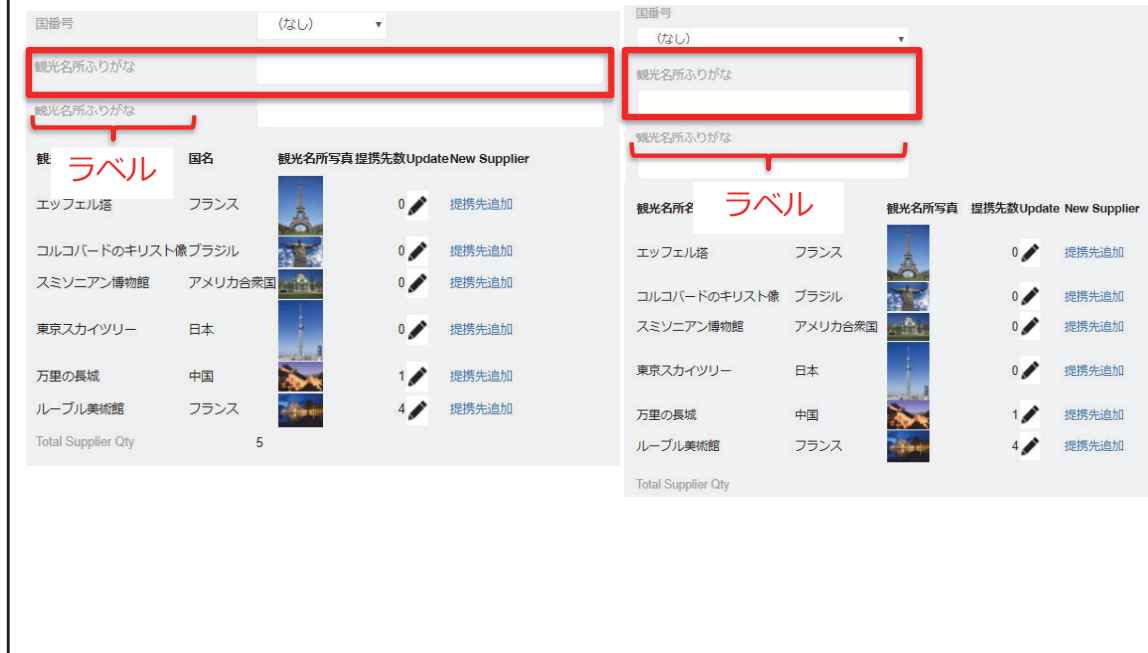
エッフェル塔	更新	削除
コルコバードのキリスト像	更新	削除
スカイツリー	更新	削除
スミソニアン博物館	更新	削除
ルーブル美術館	更新	削除
万里の長城	更新	削除

レスポンシブ Web デザイン

最後に、GeneXus で設計したアプリケーションでは、画面のサイズに応じて、画面上の情報が自動的に調整されることに触れておきます。たとえば、Work With パターンによって作成された Work With Attractions を、ノート PC の画面と一致するフルサイズのブラウザで実行しているとします。

ここで、ブラウザのウインドウサイズを縮小したらどうなるでしょうか。右側から縮小した場合、ある時点から、観光名所とアクションのみがグリッドに表示されるようになります。さらに縮小すると、左側の列が表示されなくなります。モバイル端末でブラウザからアプリケーションを実行した場合はこのような表示になります。

レスポンス Web デザイン



WWAttractionsFromScratch オブジェクトを確認すると、何もしなくても自動的に調整されていることが分かります。

画面上部の変数を見ると、左側にラベルがあり、対応する変数コントロールまで一定の幅があります。

画面を縮小すると、ラベルが上になり、変数コントロールの幅は、セルの全体を占めるようになります。

レスポンス Web デザイン

The screenshot displays the GeneXus IDE interface. On the left, a web form is shown with various controls. A red arrow points to a table control labeled 'MainTable' in the top toolbar. The form contains a dropdown for '&CountryI', two text boxes for '&AttractionFuriganaFrom' and '&AttractionFuriganaTo', a grid with three columns for '&AttractionID', '&AttractionName', and '&CountryName', and a text box for '&TotalSupplierQty'. On the right, the 'プロパティ' (Properties) window is open, showing the 'レスポンステーブル: MainTable' (Responsive Table: MainTable) section. The 'Responsive Sizes' property is highlighted with a red box and shows an empty array '[]'. Other properties like 'Row Heights' (set to '****') and 'Form Class' (set to 'Form') are also visible.

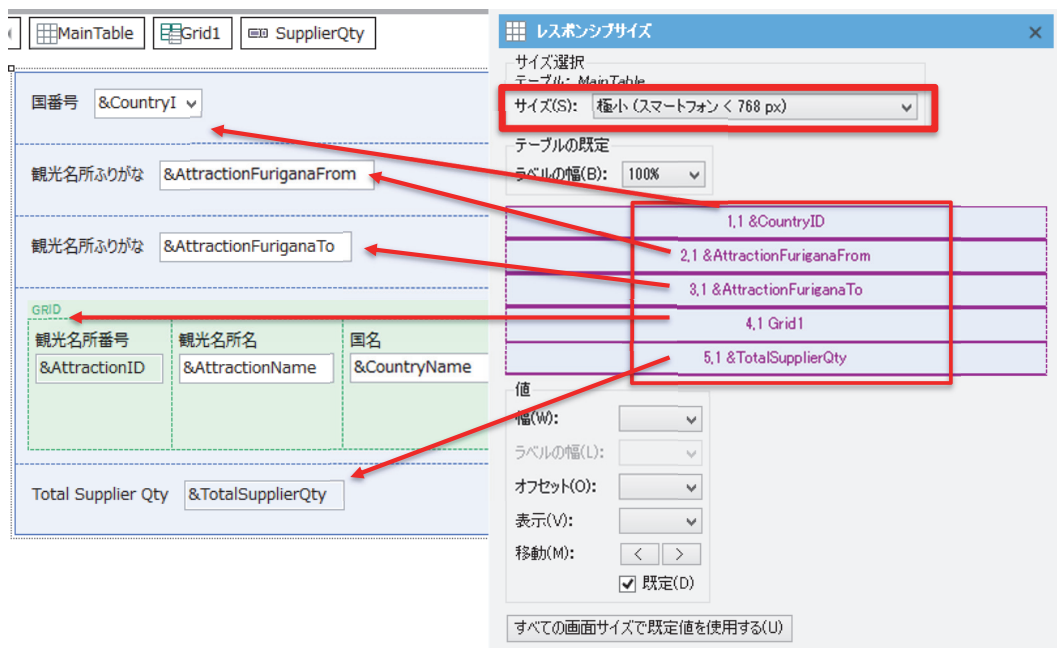
Control Name	MainTable
Responsive Sizes	[]
Row Heights	****
Tooltip Text	

Appearance	
Class	Table

Form	
Form Class	Form

この動作を実現するには、Web Form でレスポンステーブルを使用します。このトピックについてはこのコースでは詳細を省きますが、Web パネルでフォーム内の [&CountryId] コントロールを選択すると、その上 (スライド赤い枠) にテーブルコントロールがあることが分かります。これが [MainTable] です。クリックすると、このテーブルのプロパティに、レスポンスサイズに関連する [**Responsive Sizes**] プロパティが含まれることが分かります。ここをクリックします。

レスポンス Web デザイン



[レスポンスサイズ] ウィンドウ上にメインテーブルに挿入されているコントロールが表示されます。その一部、2.1 のテーブル (&AttractionNameFrom 変数と &AttractionNameTo 変数があるもの) や 3.1 のグリッドには、その内側にもコントロールがあります。

画面サイズを選択できるコンボボックスもあります。現在は [中] サイズになっています。[小] または [極小] を選択します。

レスポンシブ Web デザイン

レスポンシブサイズ

サイズ選択
テーブル: MainTable
サイズ(S): 極小 (スマートフォン < 768 px)

テーブルの既定
ラベルの幅(B): 100%

1,1 &CountryId
2,1 Table1
3,1 Grid1
4,1 &TotalTrips


値
幅(W): 100%
ラベルの幅(L): 100%
オフセット(O): 0%
表示(V): True
移動(M): < >
☒ 既定(D)

すべての画面サイズで既定値を使用する(U)

国番号
(なし)

観光名所ふりがな

観光名所ふりがな

観光名所名	国名	観光名所写真	提携先数	Update	New Supplier
エッフェル塔	フランス		0		提携先追加
コロンバードのキリスト像	ブラジル		0		提携先追加
スミソニアン博物館	アメリカ合衆国		0		提携先追加
東京スカイツリー	日本		0		提携先追加
万里の長城	中国		1		提携先追加
ルーブル美術館	フランス		4		提携先追加

Total Supplier Qty

値が変わります。

[極小] の画面では、&CountryId のラベルが幅全体を占め、変数コントロール自体も幅の 100% を占めます。実行時にラベルが変数の上に表示されたのはこのためです。

レスポンス Web デザイン

レスポンスサイズ

サイズ選択
テーブル: MainTable

サイズ(S): 中 (デスクトップ) >= 992 px 小から継承

テーブルの既定
ラベルの幅(B): 25%

1,1 &CountryId
2,1 Table1
3,1 Grid1
4,1 &TotalTrips

幅(W): 100%
ラベルの幅(L): 25%

表示(V): True
移動(M): < >
☒ 既定(D)

すべての画面サイズで既定値を使用する(U)

国番号 (なし)

観光名所ふりがな

観光名所ふりがな

観光名所名	国名	観光名所写真	優先数	Update New Supplier
エッフェル塔	フランス		0	優先先追加
コロンビアのキリスト像	ブラジル		0	優先先追加
スミソニアン博物館	アメリカ合衆国		0	優先先追加
東京スカイツリー	日本		0	優先先追加
万里の長城	中国		1	優先先追加
ルーブル美術館	フランス		4	優先先追加

Total Supplier Qty 5

ここで [中] サイズを選択すると、ラベルが幅の 25%、変数コントロールが 100% まで残りを占めることが分かります。


これは、実行時の状況と合致しています。また、特定の画面サイズで特定のコントロールを非表示にできる [Visible] プロパティもあります。

これによって、フォームが展開される画面のサイズに応じて、表示するコントロールを実行時に自動調整できます。

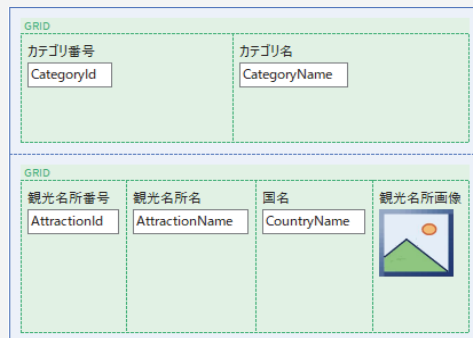
これをレスポンス Web デザインと言います。ここでは概要のみを示しました。

フリースタイルグリッド コントロール

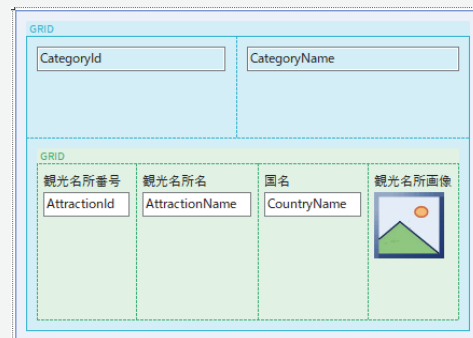
追加情報

- Web パネルのグリッドの有無
 - 自動ベーステーブル? 
- 複数のグリッドを持つ Web パネル

並列



ネスト



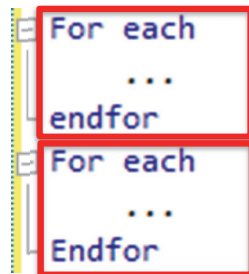
Web パネルについては、まだ学ぶことがたくさんあります。

たとえば、Web パネルにグリッドがないか、1 つだけグリッドがある場合、GeneXus は、暗黙のベーステーブルが関連付けられているかどうかを判断する項目属性をどこで見つけるのでしょうか。また、そこに項目属性を表示する場合、どのような基準を満たしている必要があるのでしょうか。For each の場合と同じで、拡張テーブルに属している必要があると推測できます。

複数のグリッドを並列またはネストさせた Web パネルの実装も可能です。For each の並列またはネストと同様です。

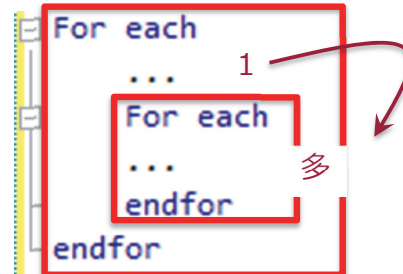
ネストされた For each の暗示的フィルタ

並列の For each



独立した
ナビゲーション

ネストされた For each



関連する
ナビゲーション?

どのようにして GeneXus は開発者の記述なしでフィルタを判断したのでしょうか？
これは、For each コマンドをどう記述するかによって決まります。

2 つの For each コマンドが順に記述されている場合 (スライド左側)、これらのコマンドは**相互に独立**しています。

一方、1 つの For each コマンドの**中**にもう 1 つ For each コマンドが記述されている場合 (スライド右側)、1 つ目のナビゲーションの各レコードに対して、2 つ目のナビゲーションのレコードセット (結果集合) を参照できます。

ネストされた For each コマンドを記述すると、GeneXus は、各 For each コマンドに対してナビゲートされるベーステーブルを決定し、**これらのテーブル間の関連を確認**します。

