



WORKWITHPLUS

WorkWithPlus for Web

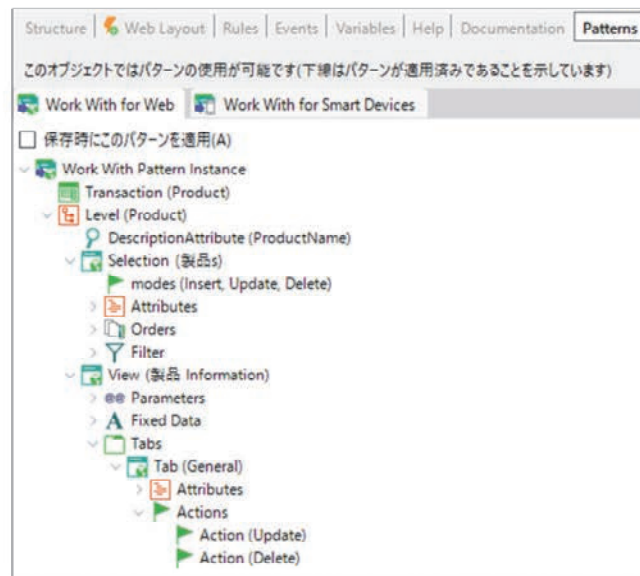
トレーニング

目次

01	柔軟性	04	Web パネル
02	トランザクションオブジェクトへの適用	05	PDF レポート
2.1	Transaction	06	セキュリティ
2.2	List	07	監査
2.3	View	08	まとめ
03	WorkWithPlus for Web の設定		
3.1	既定値の設定		
3.2	テンプレート		
3.3	インスタンスの更新 / 適用		

1. 柔軟性	P. 3
2. トランザクションオブジェクトへの適用	P. 24
2.1. Transaction	P. 24
2.2. List	P. 37
2.3. View	P. 62
3. WorkWithPlus for Web の設定	P. 71
3.1. 既定値の設定	P. 73
3.2. テンプレート	P. 81
3.3. インスタンスの更新 / 適用	P.102
4. Web パネル	P.106
5. PDF レポート	P.114
6. セキュリティ	P.122
7. 監査	P.130
8. まとめ	P.135

はじめに: パターンとは？



はじめに、GeneXus がもつパターンについてです。

パターンとは、GeneXus があらかじめ持つ拡張性の1つとなり、生産性を高めるために特定の動作を備えた一連のオブジェクトを自動生成する機能です。

GeneXus はあらかじめ2つのパターンを保有し、それぞれ Web と Native Mobile に向けたパターンです。

特定のオブジェクトにパターンを適用することで、スライドのように「パターンインスタンス」が生成されます。

自動生成されるオブジェクトは必要に応じて開発者がカスタマイズできるように柔軟性を含んでいます。

このカスタマイズを容易にするためにパターンインスタンスは階層構造による定義が可能となっています。



WORKWITHPLUS
FOR WEB

1. 柔軟性 (WorkWithPlus for Web の特徴)



柔軟性: WorkWithPlus for Webによる生成

The screenshot displays the WorkWithPlus for Web interface. On the left, a table lists companies with columns for company number, name, and description. The table includes entries for '3M', 'Overlog', 'Genetix', 'SABIC', 'BASF', 'TCS', and 'Thomson'. The 'Overlog' entry is highlighted. To the right, two overlapping windows show the detailed view for 'Overlog'. The top window is a '詳細情報' (Detailed Information) form with fields for company name, description, website, and various dates. The bottom window is a '会社概要' (Company Overview) form with fields for company name, description, website, and various dates. Both windows have '実行' (Execute) and '終了' (End) buttons.

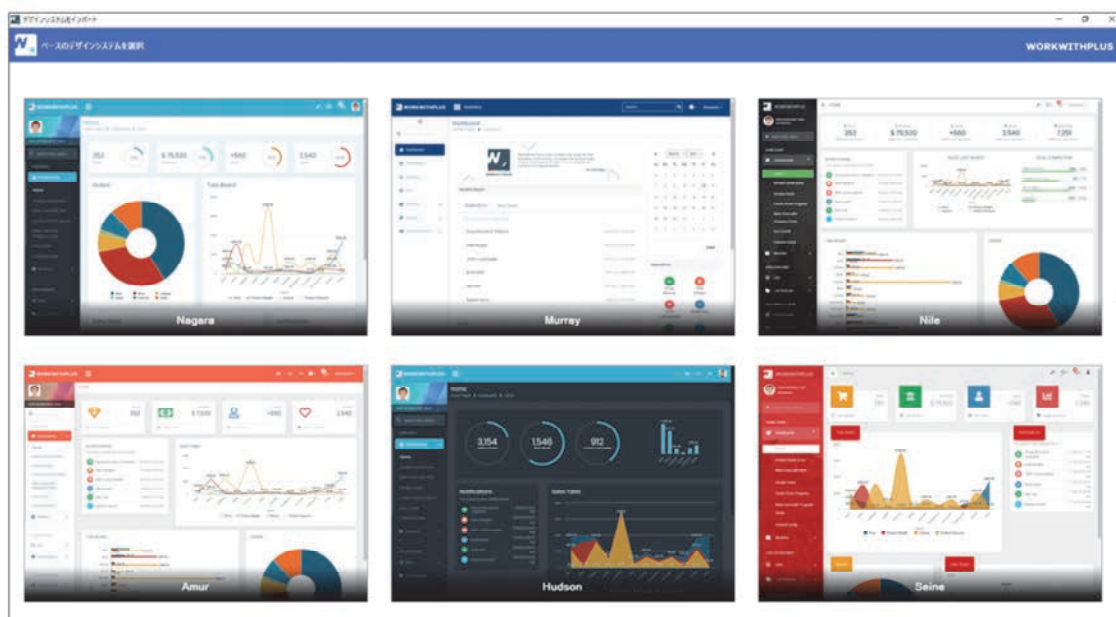
And more...

WorkWithPlus for Web を利用すると一覧画面や参照用画面を自動生成できます。さらにトランザクションオブジェクトの入力画面をパターンからカスタマイズすることが可能です。またこれら意外にもユーザービリティを向上するオブジェクトや機能の生成が可能です。

WorkWithPlus for Web はトランザクションオブジェクト以外にも Web パネルオブジェクトや、Procedure オブジェクトへも適用できます。

本コースではこれらの生成、利用について修得いただけます。

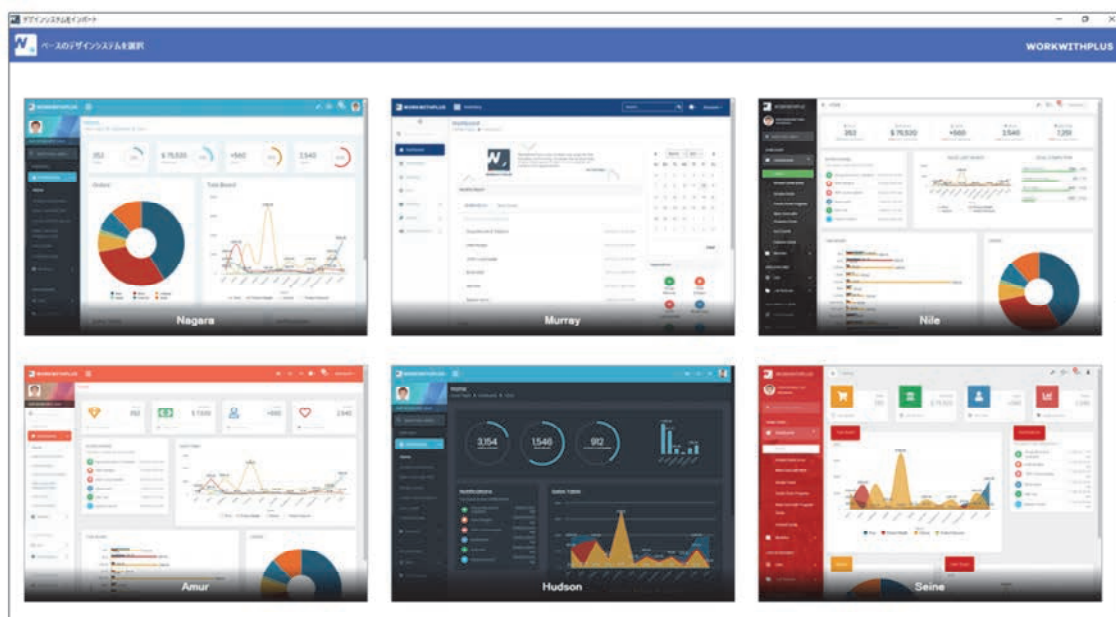
柔軟性: デザイン システム ウィザード



最初に説明する項目は、オブジェクト生成時にパターンが備えている柔軟性についてです。WorkWithPlus for Web を利用し、作業を開始するときは、プロジェクトのテーマと機能をカスタマイズできるデザイン システム ウィザードを使用します。

このデザイン システム ウィザードでは、それぞれのプロジェクトの外観、操作感、および設定をすべて変えることができるため、柔軟性に富んだカスタマイズが可能です。

柔軟性: デザイン システム ウィザード ステップ1



ステップ1 「ベースのデザインシステムを選択」

WorkWithPlus for Web があらかじめ定義しているデザインからベースとなるデザインを選択します。

表示されているデザインは以降のステップで自由に選択できるため、生成するアプリケーションに一番近いイメージを選択することでデザインの変更を最小限に抑えられます。

柔軟性: デザイン システム ウィザード ステップ2



ステップ2「基本設定」

ステップ1で選択したデザインから基本的な設定のみを変更します。

この変更でデザインの指定が十分な場合、「デザインシステムをインポート」ボタンをクリックすることで WorkWithPlus for Web の初期設定を完了し、アプリケーションの開発を開始できます。

もし、設定が十分でない場合、「詳細設定」リンクをクリックし、以降のステップで設定を行います。

柔軟性: デザイン システム ウィザード ステップ3

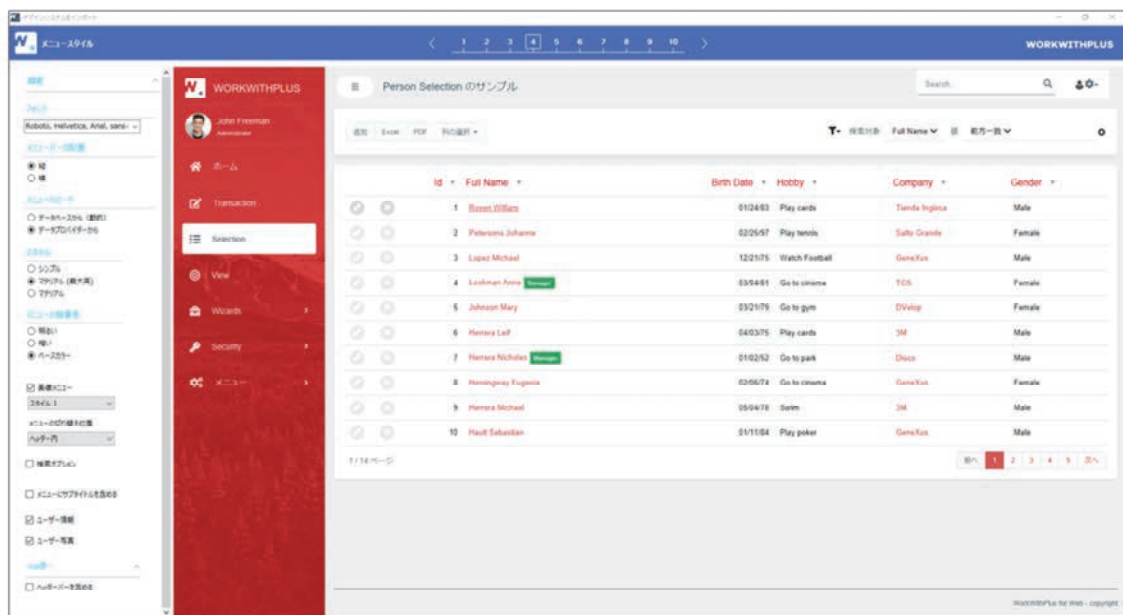


ステップ3「セキュリティ」

セキュリティ機能の有効無効を設定します。

セキュリティ機能を有効にする場合、付随するいくつかの機能についても指定します。

柔軟性: デザイン システム ウィザード ステップ4



ステップ4「メニュースタイル」
メニューに関連したいくつかの設定を行います。
メニュー自身のスタイルや、表示内容など設定が可能です。

柔軟性: デザイン システム ウィザード ステップ5

The screenshot displays the 'WorkWithPlus' design system wizard at Step 5. The interface is divided into three main sections:

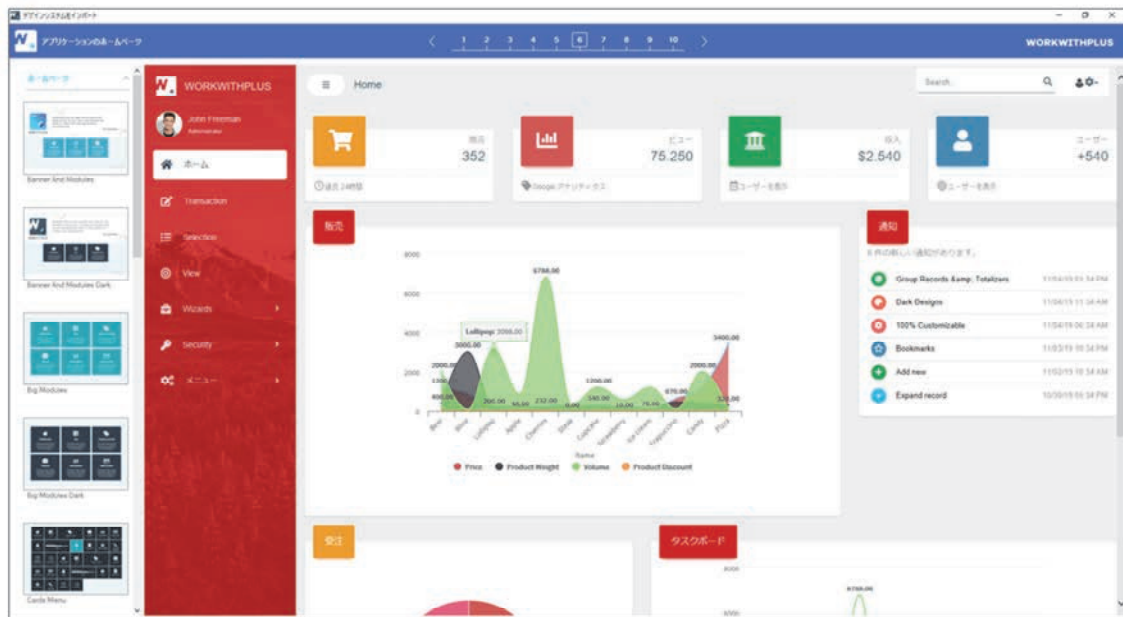
- Left Sidebar:** Contains navigation links and a list of design system components. The 'Transaction' link is highlighted.
- Main Content Area:** Displays a form titled 'Person Transaction のサンプル'. The form includes fields for 'First Name' (Mary), 'Last Name' (Johnson), 'Nick Name' (Male), 'Home Number' (295585), 'Mobile Number' (98727272), 'Hobby' (Go to gym), 'Music Band' (The Beatles), and 'Gender' (Female). A 'Birth Date' field is set to '05/09/87'.
- Bottom Section:** Contains a table with two columns: 'Email' and 'Description'. The 'Email' column lists 'maryjohnson@mail.com' and 'mary@mail.com'. The 'Description' column lists 'Professional' and 'Personal'.

The bottom of the screen shows a 'Copyright' notice: 'WorkWithPlus for Web - copyright'.

ステップ5 「マスターページ」

すべての画面で共通して表示されるパーツであるマスターページのデザインや表示内容を設定します。

柔軟性: デザイン システム ウィザード ステップ6



ステップ6「アプリケーションのホームページ」
アプリケーションのホームとして表示する画面のデザインを設定します。
想定するデザインに一番近いものを選択します。

どのデザインを選択した場合も、デザインシステムウィザードによるインポートが完了した後に、選択しなかったデザインへ変更することが可能です。

柔軟性: デザイン システム ウィザード ステップ7

The screenshot displays the WORKWITHPLUS design system wizard, Step 7: Item Attributes and Background Style. The interface is divided into three main sections:

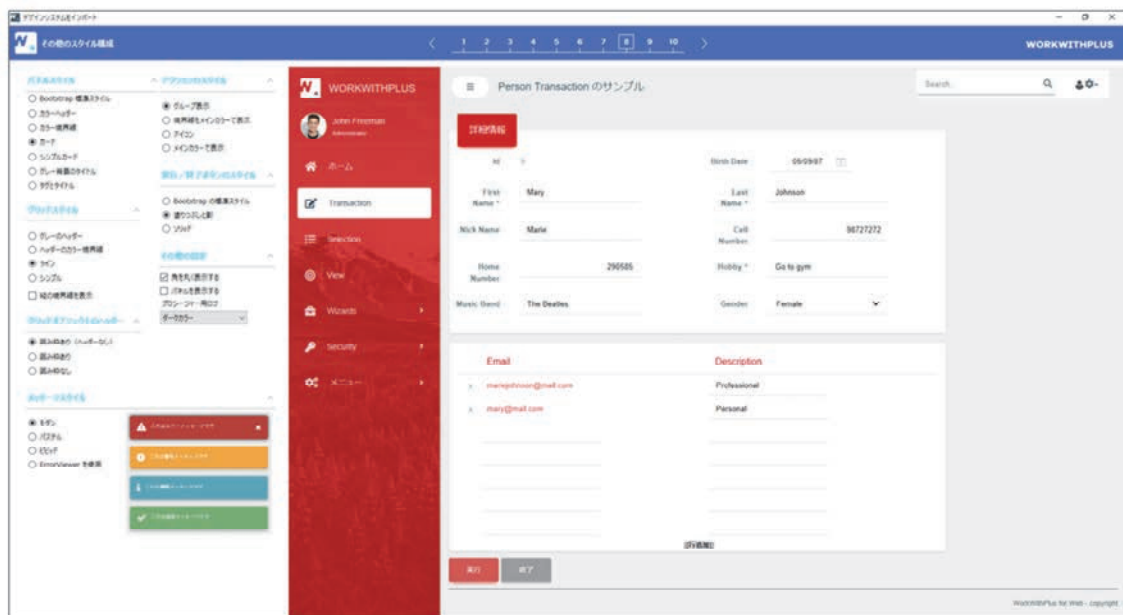
- Left Sidebar:** Contains navigation options for various design elements, including colors, fonts, and backgrounds. The 'Background' section is currently selected.
- Main Content Area:** Displays a form for 'Person Transaction のサンプル'. The form includes fields for personal information (Name, Birth Date, First Name, Last Name, Nick Name, Home Number, Music Band, Cell Number, Hobby, Gender) and a table for 'Email' and 'Description'.
- Right Sidebar:** Contains a search bar and a user profile section with a search bar and a list of items.

The form fields are as follows:

Person Transaction のサンプル	
登録済	
ID	Birth Date
First Name	Last Name
Nick Name	Cell Number
Home Number	Hobby
Music Band	Gender
Email	
Description	

ステップ7「項目属性および背景のスタイル」
入力欄のスタイルや画面の背景スタイルを設定します。

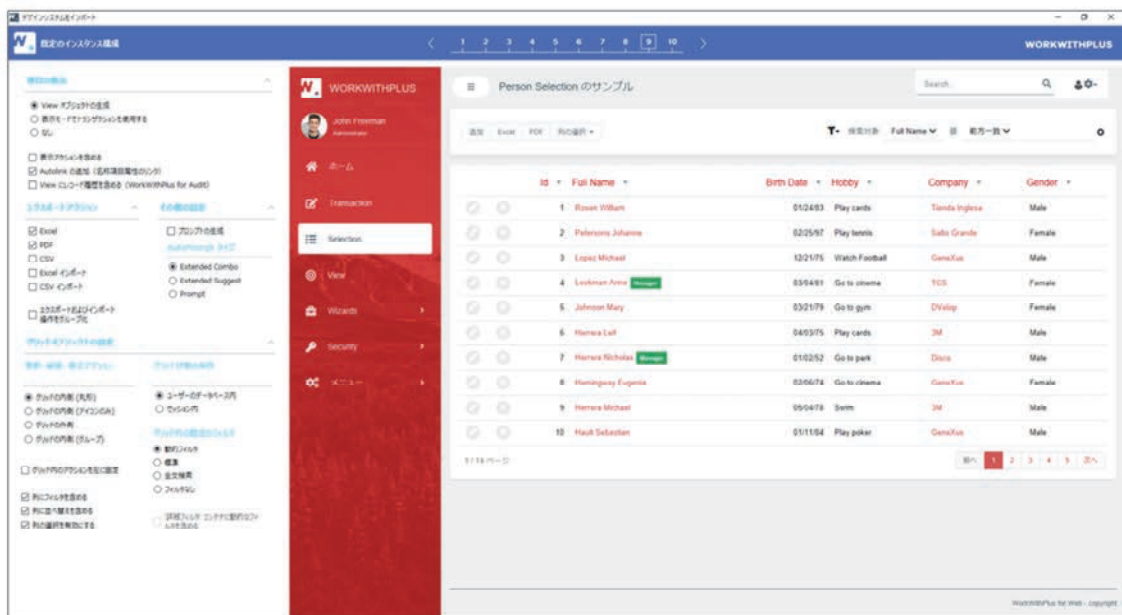
柔軟性: デザイン システム ウィザード ステップ8



ステップ8「その他のスタイル構成」

ここまでのステップで対象となっていなかったスタイルを設定します。
グリッドやボタン、メッセージのスタイルなどが対象となります。

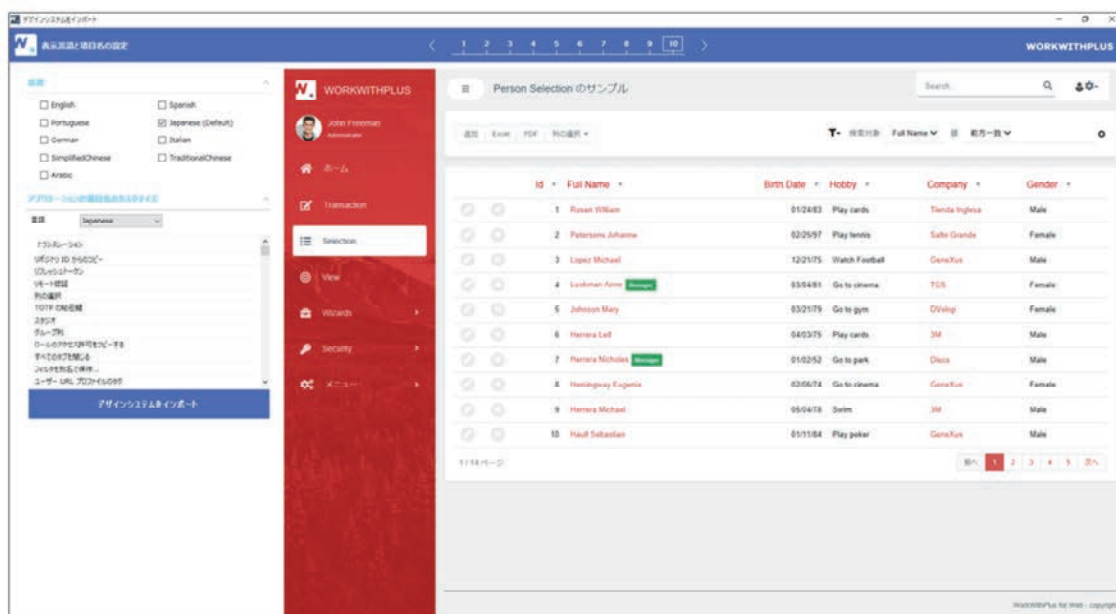
柔軟性: デザイン システム ウィザード ステップ9



ステップ9「既定のインスタンス構成」

WorkWithPlus for Web パターンを適用する際に含める機能について設定します。データの出力機能や、外部参照データの選択方法などがあります。

柔軟性: デザイン システム ウィザード ステップ10



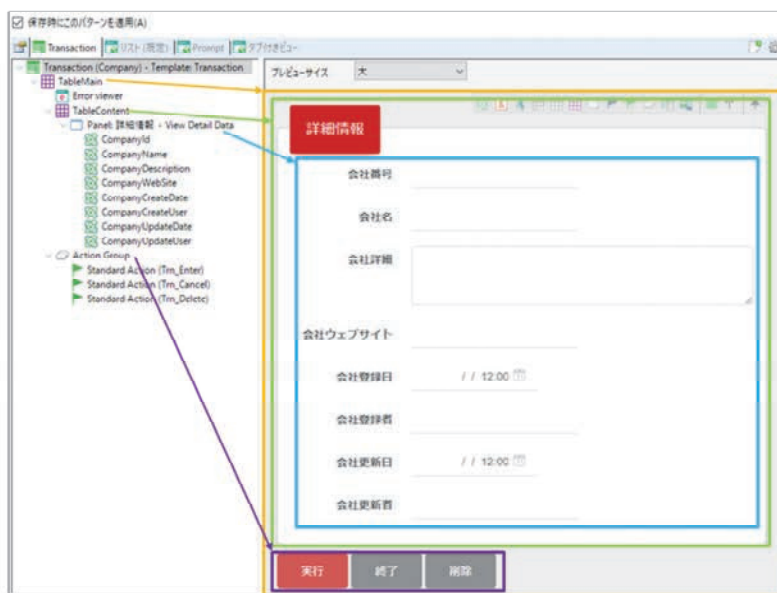
ステップ 10 「表示言語と項目名の設定」

GeneXus ではマルチランゲージに対応したアプリケーションを生成できます。このステップではマルチランゲージで生成するか（生成時に含める言語の種類追加）、各言語でどのようなテキストを表示するかを設定します。

以上でデザインシステムウィザードにおける詳細設定が完了となります。

「デザインシステムをインポート」をクリックすることで WorkWithPlus for Web の初期設定が完了し、アプリケーションの開発を開始できます。

柔軟性: 階層構造



WorkWithPlus for Web のパターンを適用することで、画面のデザインを階層構造で定義することができます。

WorkWithPlus for Web を利用する場合、必ず [TableMain] ノードが含まれ、すべてのコントロールを含むテーブルコントロールが生成されます。

この階層構造に基づき、1つのノードは1つのコントロールとして画面が生成されます。

生成される画面イメージはリアルタイムプレビューで確認することができます。

同一階層内でノードを移動したい場合、ドラッグアンドドロップや、Ctrl + 上（下）矢印キーで表示順を変更できます。

別の階層へ移動したい場合、切り取り + 貼り付けで変更できます。

柔軟性: 階層構造

TableMain



Table: Table	
Table name	TableMain
Type	Responsive
Form	
Theme class	TableMain
Is Group	False
Is Section	False
Include in panel	False
Table	
Number Of Columns	0
Num. Of Cols. (Extra Small)	1
Num. Of Cols. (Small)	1
Num. Of Cols. (Medium)	1
Num. Of Cols. (Large)	1
Width	
Height	
Visibility	
Visible Condition	
Location in father container	
Column span	1
Row span	1
Horizontal alignment	<default>
Vertical alignment	Middle
Start Col/1st Avail	0
Other GeneXus Properties	
Other GeneXus Properties	

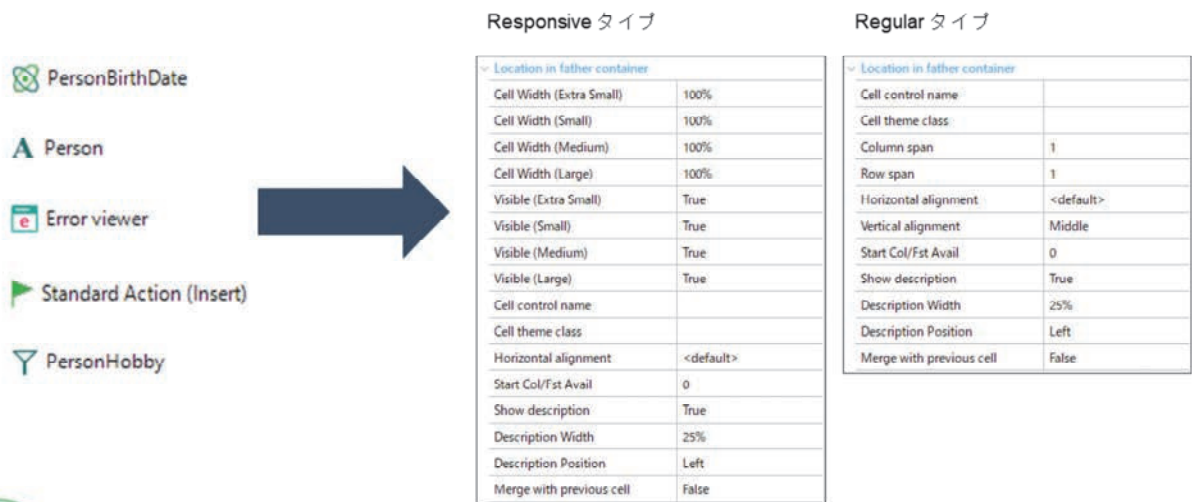
階層構造による画面デザインに加え、柔軟性を持たせるためには各ノードのプロパティも利用する必要があります。

まず、画面のデザインに大きく影響するノードとして、他のノードを追加し、画面構成を行う [Table] タイプ ([Table] ノードや [Panel] ノード等) があります。このノードが持つプロパティ群のうち、[Table] プロパティ群、[Location in father container] プロパティ群に含まれるプロパティが重要です。

[Table] プロパティ群に含まれる [Number Of Columns] プロパティは、生成されるテーブルコントロールで1行あたり何列表示するか指定します。この設定により、[Table] タイプのノードに含まれるノードを1列ではなく、指定された列数で左から右、上から下へと配置した画面が生成できます。

[Location in father container] プロパティ群についてはほとんどのノードに含まれるため、次のスライドでフォーカスします。

柔軟性: 階層構造



[Table] タイプのノードに追加できるノードはいくつかの種類があり、主に GeneXus の [Web Layout] エLEMENTで利用できるコントロールに基づきます。これらほぼすべてのノードが [Location in father container] プロパティ群を含んでいます。

このプロパティ群は、名前の通りノードの親にあたる [Table] タイプのノードにおける「セル」にかかわる設定を行うことができ、これらのプロパティを設定することでより柔軟な画面デザインを行えます。

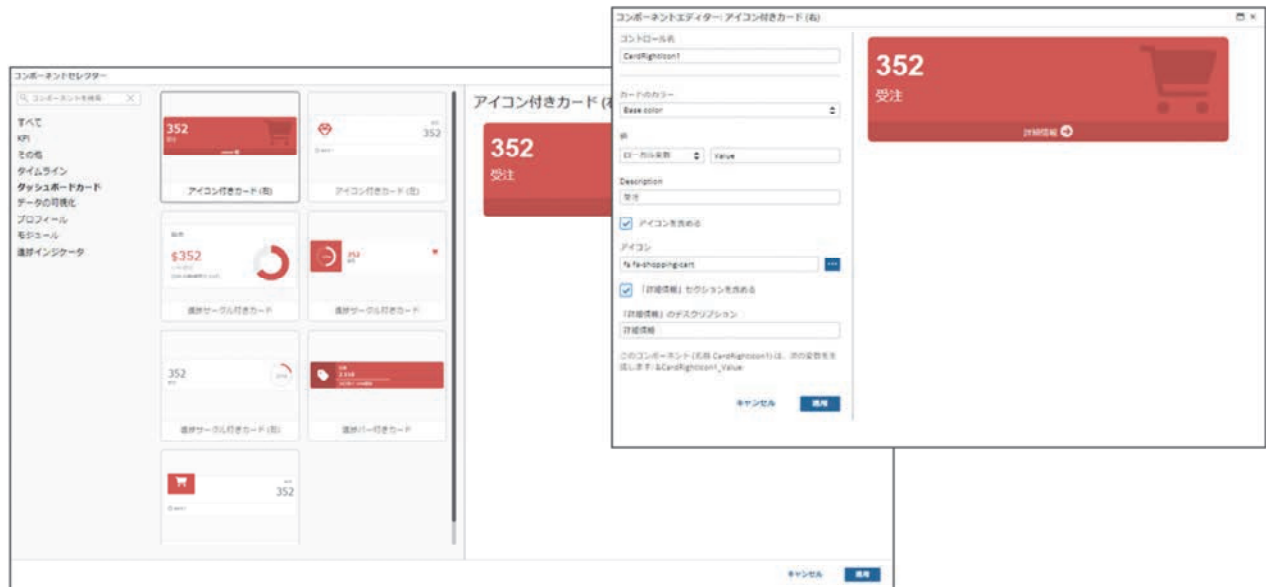
一部のプロパティはノードによって表示されるもの、表示されないものがあります。

また、親ノードが Responsive Web Design に対応した Responsive タイプか、対応していない Regular タイプかも表示されるプロパティに影響します。

- [Cell Width (**)] / [Column span] (**部分はサイズ名標記を省略)
1行のうち、このノードが全体の何%（または列数）を占有するか指定します。
- [Cell theme class]
セルのパディングや背景画像を指定するため等の目的でクラスを指定します。
- [Merge with previous cell]
プロパティの対象ノードを格納するセルを直前のノードを格納するセルと結合（1つのセルに複数のコントロールを含める）したい場合、Trueに変更します。

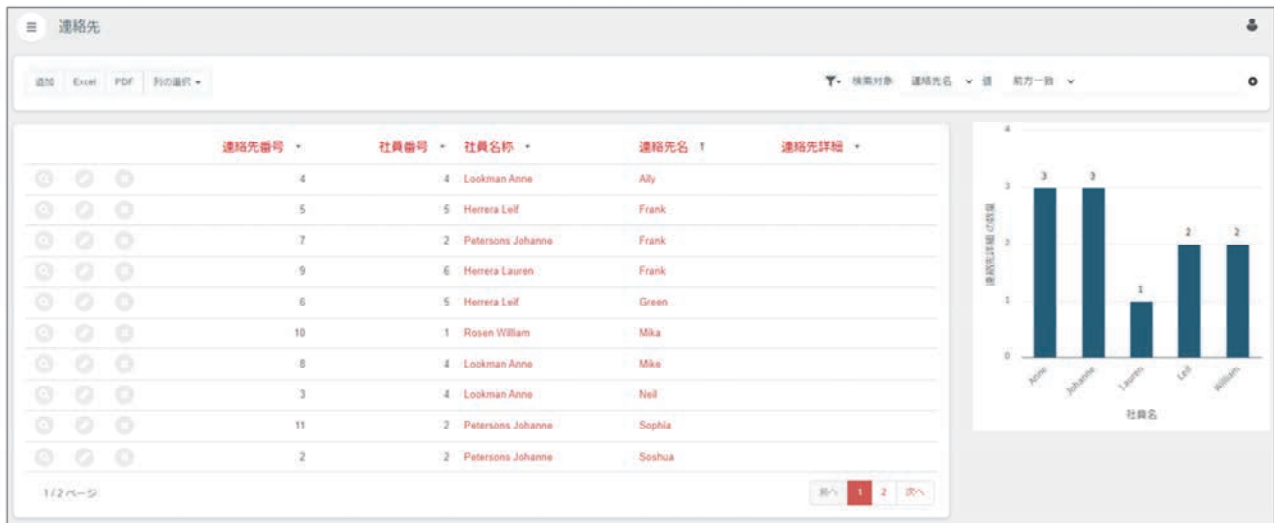
このスライドでフォーカスしているプロパティ群外となりますが、[Theme class] プロパティにクラスを指定することで、ノードから生成されるコントロールの色、境界線、および外観と操作感を設定することができます。

柔軟性: コンポーネント



追加できるノードの1つに [components] ノードがあります。
このノードはより生産性を高めるために WorkWithPlus for Web で事前に定義されたインターフェース、変数、イベントを追加します。
[components] ノードを追加すると、実際にどのコンポーネントを利用するか選択するための画面が表示され、追加するコンポーネントを選択すると、コンポーネントの詳細設定を行う画面が表示されます。

柔軟性: ユーザーコントロール



開発者は、オブジェクトの階層構造によって、オブジェクトの Web フォームを生成する方法を決定できます。

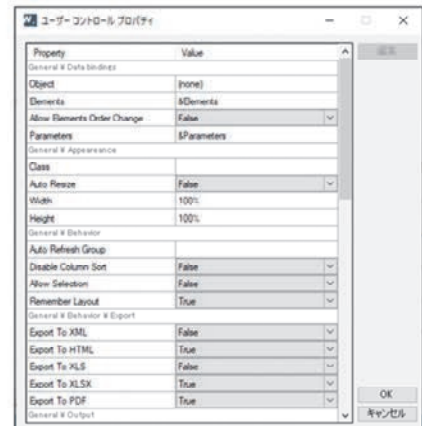
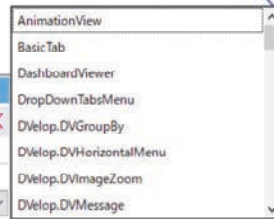
しかし、WorkWithPlus for Web で追加できるノードでは生成できないコントロールを配置したい場合があります。

たとえば、このスライドのようにグリッドの横に QueryViewer コントロールを表示したいという要望がありました。

WorkWithPlus for Web を使用する場合、このような「ユーザーコントロール」と呼ばれる特別な動作を備えたコントロールを配置する場合、[UserControl] ノードを利用できます。

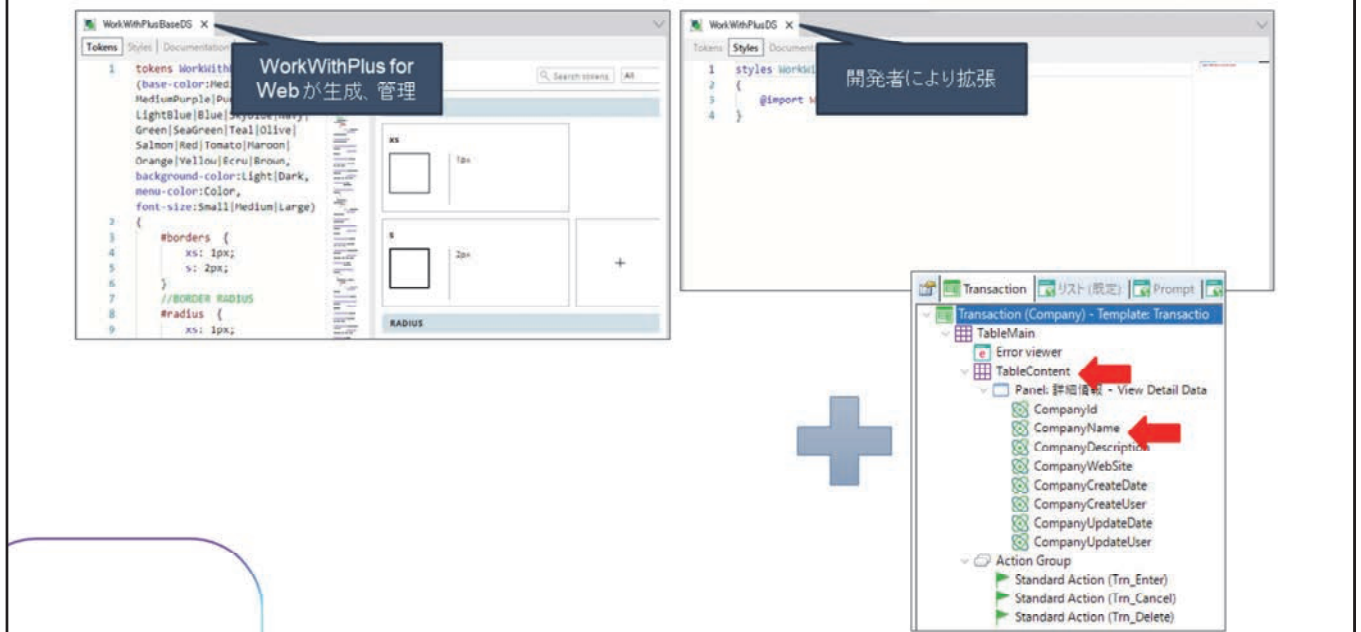
柔軟性: ユーザーコントロール

User Control: UserControl1



[UserControl] ノードでは、[User Control] プロパティのコンボボックスから GeneXus にインストールされているユーザーコントロールを選択できます。選択したユーザーコントロールに関するプロパティは [User Control Properties] プロパティの 3 点ボタンから起動する「ユーザーコントロールプロパティ」ダイアログ内で指定できます。

柔軟性: 外観と操作感の変更



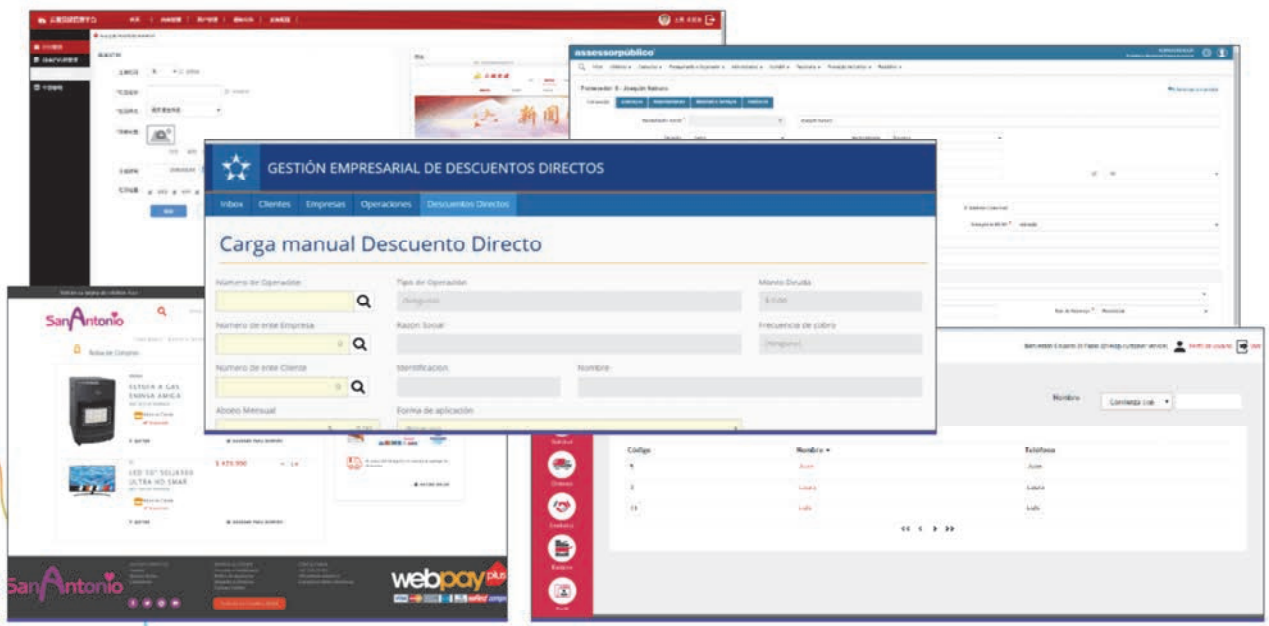
アプリケーションの外観や操作感を変更する場合、WorkWithPlus for Web によって定義されたデザインシステムオブジェクトに含まれるスタイルを階層構造の各ノードが持つクラスに利用します。

WorkWithPlus for Web が生成、管理するデザインシステムオブジェクトとして WorkWithPlusBaseDS があり、これをインポートする形で実際に生成されるアプリケーションが利用する WorkWithPlusDS デザインシステムオブジェクトがあります。

ここまでに見てきたように各ノードは [Theme Class] および [Cell theme class] という名前のプロパティを持っています。これらのプロパティを変更し、対象のコントロールの外観と操作感を定義します。

したがって、デザインシステムオブジェクト内に新しいスタイルを作成し、それをコントロールまたはコントロールのセルに割り当てることができます。スタイルを使用して、余白や背景などを定義できます。

柔軟性: 様々な種類のアプリケーション



WorkWithPlus for Web を使用することで、GeneXus アプリケーションの開発時間を平均 60% 短縮することができます。この値は内部のメトリクスおよびユーザーの声によって得られたもので、開発するアプリケーションのタイプによって異なります。

ここに表示されているのは、WorkWithPlus for Web を使用して生成されたさまざまな海外のアプリケーションです。

画像に表示されているアプリケーションの一部は従来のトランザクションのアプリケーションとはまったく異なります。またすべてのケースにおいて、パターンが有用であり、開発の生産性が向上しています。



WORKWITHPLUS
FOR WEB

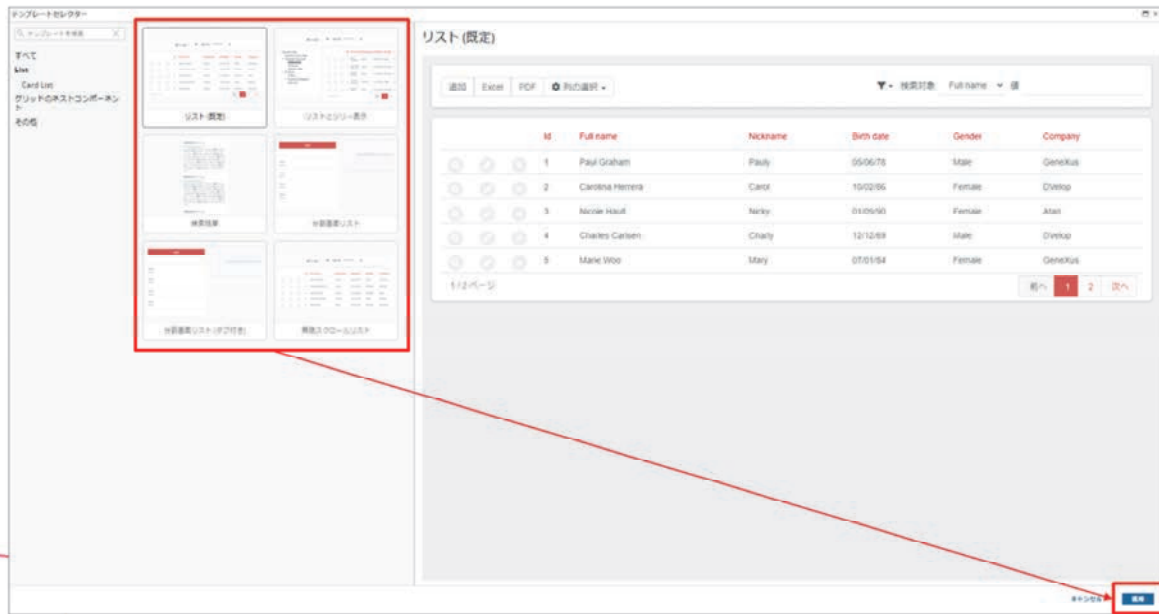
2. トランザクションオブジェクトへの適用

2.1. Transaction



ここからは WorkWithPlus for Web をオブジェクトに適用し、どのような機能が利用できるか説明していきます。
まずはトランザクションオブジェクトへの適用です。

Transaction: 適用



トランザクションオブジェクトの Patterns エlementを開き、[WorkWithPlus] タブを選択した場合、適用するテンプレートを選択する「テンプレートセクター」ダイアログが表示されます。

この画面では、リスト（次の章で紹介）を基準としたテンプレートの選択となります。

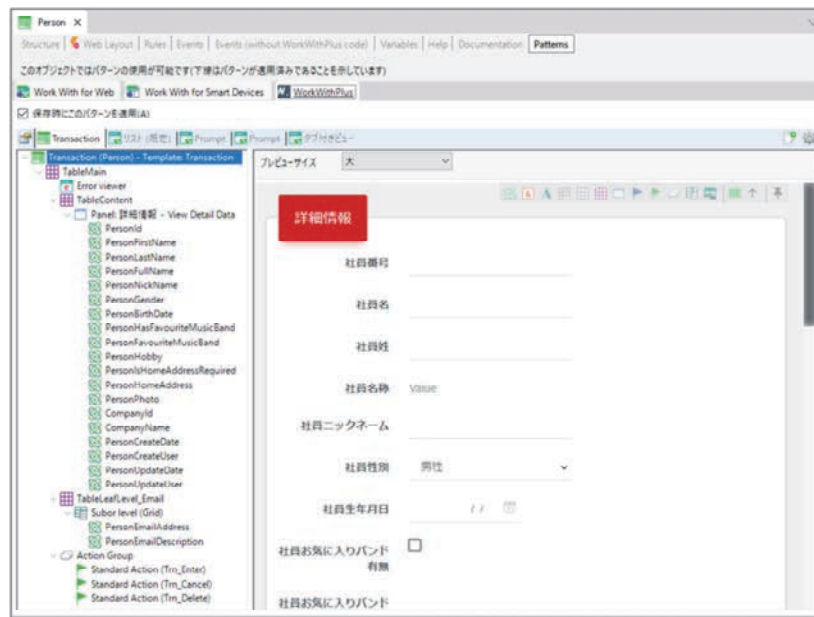
テンプレートを選択し、「適用」ボタンをクリックすることで、テンプレートに基づく既定の項目属性や変数を指定するダイアログが表示されます。

（次スライドへ続く）

Transaction: 適用

必要な操作を完了したら「適用」ボタンをクリックし、トランザクションオブジェクトにパターンを適用します。

Transaction: 適用



すでに説明したように、WorkWithPlus for Web を利用し、トランザクションオブジェクトをカスタマイズできます。

この章では WorkWithPlus for Web によるトランザクションオブジェクトのカスタマイズにフォーカスしていきます。

「柔軟性」の章で扱った通り、WorkWithPlus for Web によってトランザクションオブジェクトの Web Layout エレメントを階層構造で定義できます。

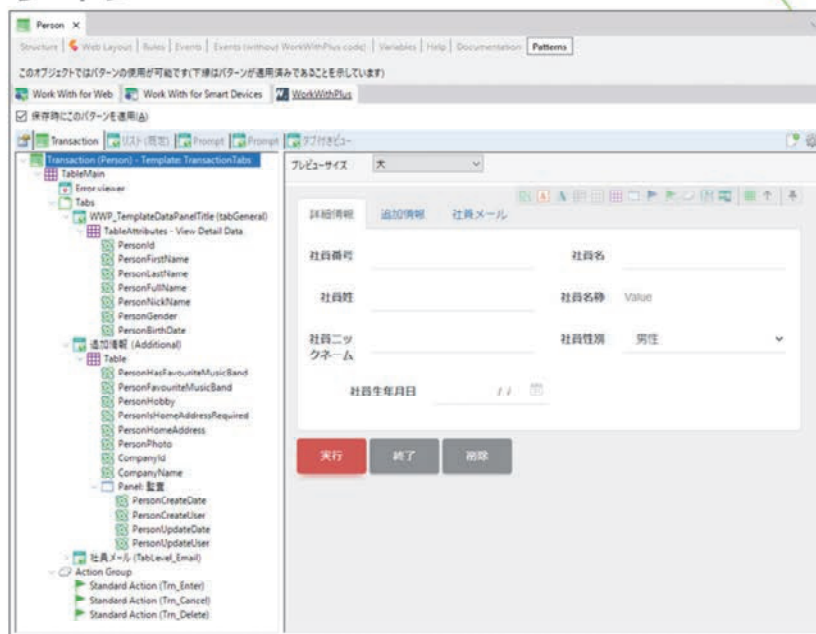
このほか、一部のルールやイベントをこのパターンインスタンスで定義することができます。

Transaction: 画面デザイン

Tabs

WWP_TemplateDataPanelTitle (tabGeneral)

Panel: 監査



パターンインスタンスにて、階層構造へ [Tabs] ノードを追加することでタブを利用した画面デザインが実装できます。

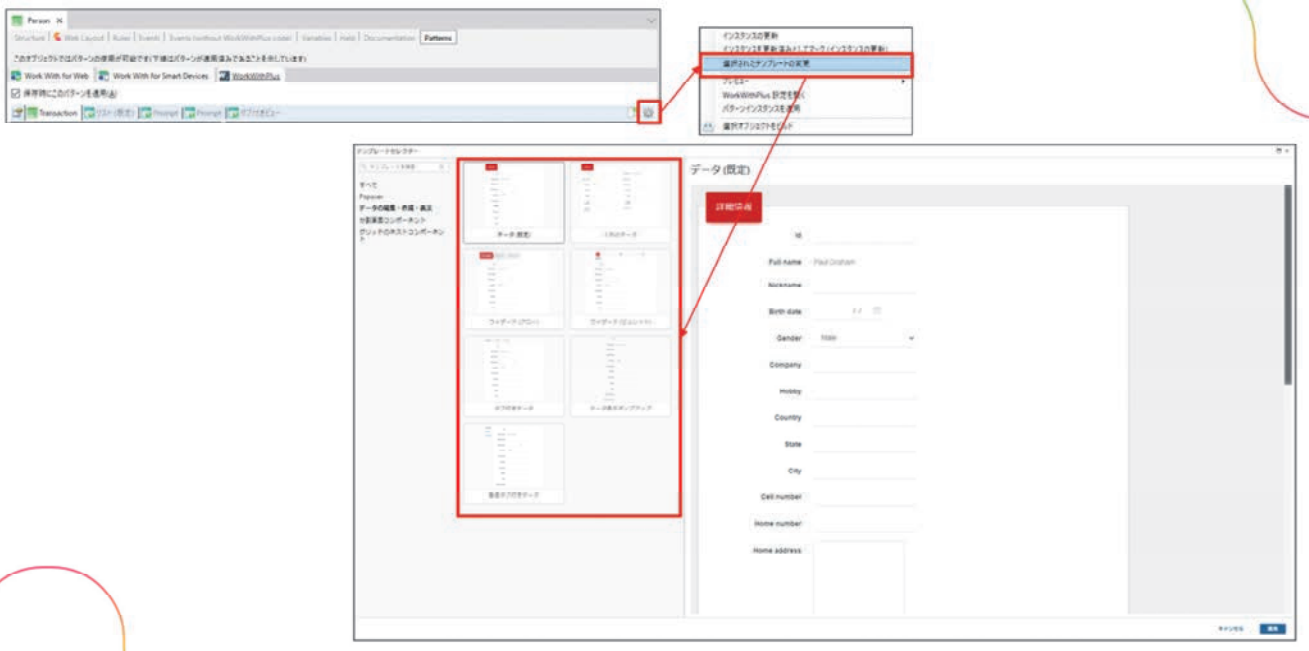
[Tabs] ノードには、定義したいタブ数分 [Tab] ノードの追加が必要となります。

また、[Panel] ノードを追加すると、タイトル付きのテーブルコントロールが追加できます。

この [Panel] ノードで生成されたテーブルはタイトル部分をクリックすることで折りたたむ機能が既定で有効になっています。

これはプロパティの設定で無効にすることも可能です。

Transaction: 画面デザイン



前のスライドでは、階層構造で [Tabs] ノードを追加し、タブを利用したレイアウトをデザインしていますが、[Tabs] ノードを 1 からすべてデザインせずにベースを構築する方法があります。

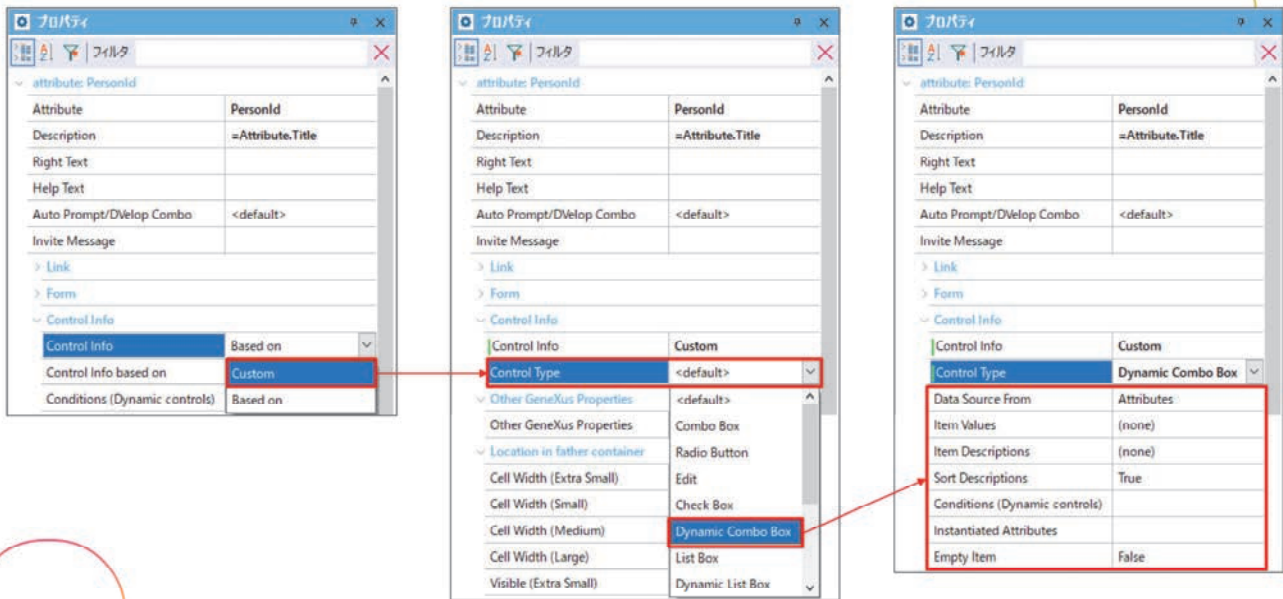
この方法は、「テンプレートの変更」という方法です。パターンを適用する際にも出てきた「テンプレート」ですが、後半に改めてご紹介します。

Transaction が参照するテンプレートを変更したい場合、画面右端の「歯車」のアイコンをクリックし、表示されるメニューから「選択されたテンプレートの変更」をクリックします。

そして、表示される「テンプレートセクター」で任意のテンプレートを選択します。

この場合は「タブ付きデータ」テンプレートです。

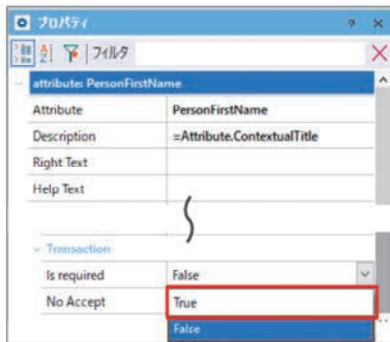
Transaction: 画面デザイン



[Attribute] タイプノードがコントロールとして生成される場合、トランザクションオブジェクトの [Structure] エレメントにおいて、項目属性の [Control Type] プロパティに指定されたタイプが適用されます。ただし、これを特定の画面固有で変更したい場合があります。この場合、[Attribute] タイプノードの [Control info] プロパティを [Custom] に変更し、表示される [Control Type] プロパティを変更することで対応できます。

Transaction: 機能 (プロパティ)

[Is required] プロパティ



変更点1

PersonFirstName



PersonFirstName

変更点2

社員名



社員名

変更点3

```
Error(Format('wP_RequiredAttribute', '社員名')) if PersonFirstName.IsEmpty();
```

Transaction の階層構造内に追加された項目属性ノードの [Is required] プロパティを利用することで、開発者が項目属性の入力を必須とするか指定することができます。

つまり、このプロパティの設定によってユーザーが該当項目を未入力のまま登録することができるか指定することができます。

[Is required] プロパティを True に設定した場合、GeneXus IDE 上で 3 つの変化を確認できます。

1. 階層構造内のアイコンの変化
2. リアルタイムプレビューの表示の変化
3. Error ルールの追加

既定で追加される Error ルールのメッセージ内容については、

WorkWithPlus for Web によって既定値が管理されています。

この既定値については、「テンプレート」についての章で改めて取り上げます。

既定のエラーメッセージでは、下記ドキュメントで説明している GeneXus の機能が利用されています。

- ・ Format 関数

<http://wiki.genexus.jp/hwikibypageid.aspx?8406>

- ・ アプリケーションのローカライズ（ドキュメント内「静的翻訳」）

<http://wiki.genexus.jp/hwikibypageid.aspx?6330>

Transaction: 機能 (プロパティ)

[Is required] プロパティ

- [Is required extra condition] プロパティ
 - [Is required initial appearance] プロパティ

Is required	True
Is required extra condition	PersonIsHomeAddressRequired = True
Is required error message	<default>
Is required initial appearance	Not required

社員住所必須 ☐

社員住所

社員住所必須 ☒

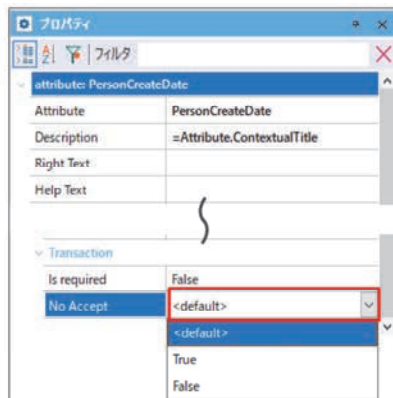
社員住所

入力を必須とするために追加の条件を定義することができます。
これは、[Is required] プロパティを [True] に変更した際に表示される
[Is required extra condition] プロパティを利用します。
このプロパティに記載された条件を満たした場合、未入力の場合にエラーが発生するようになります。

[Is required extra condition] プロパティに値を設定した場合、さらに
[Is required Initial appearance] プロパティが表示されます。
このプロパティによって既定の挙動（入力必須か必須でないか）を指定できます。

Transaction: 機能 (プロパティ)

[No Accept] プロパティ



変更点1

社員登録日

// 12:00



社員登録日

Value

変更点2

```
NoAccept(PersonCreateDate);
```

Transaction の階層構造内に追加された [Attribute] ノードの [No Accept] プロパティを利用することで、開発者が項目属性の入力を許可するか指定することができます。

つまり、このプロパティの設定によって Transaction オブジェクトの NoAccept ルールの追加について定義できます。

プロパティの値は [<default>]、[True]、[False] から選択できます。それぞれ下記のような実装となります。

- <default>

このプロパティの既定値です。

WorkWithPlus for Web による既定の設定が適用されます。

「既定の設定」とは、対象の項目属性が「外部参照のキー項目」の場合に、特定の条件下で有効となる NoAccept ルールが追加されます。

- True

[True] に変更した場合、GeneXus IDE 上で2つの変化を確認できます。

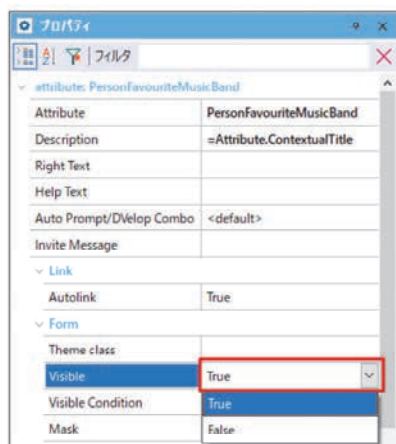
1. リアルタイムプレビューの表示の変化（リアルタイムプレビューにおいて読み取り専用項目は、入力欄ではなく、「Value」という表示になります。）
2. No Accept ルールの追加

- False

この項目属性に対する NoAccept ルールは追加されません。

Transaction: 機能 (プロパティ)

[Visible] プロパティ



Attribute	PersonFavouriteMusicBand
Description	=Attribute.ContextualTitle
Right Text	
Help Text	
Auto Prompt/Develop Combo	<default>
Invite Message	
Link	
Autolink	True
Form	
Theme class	
Visible	True
Visible Condition	True
Mask	False

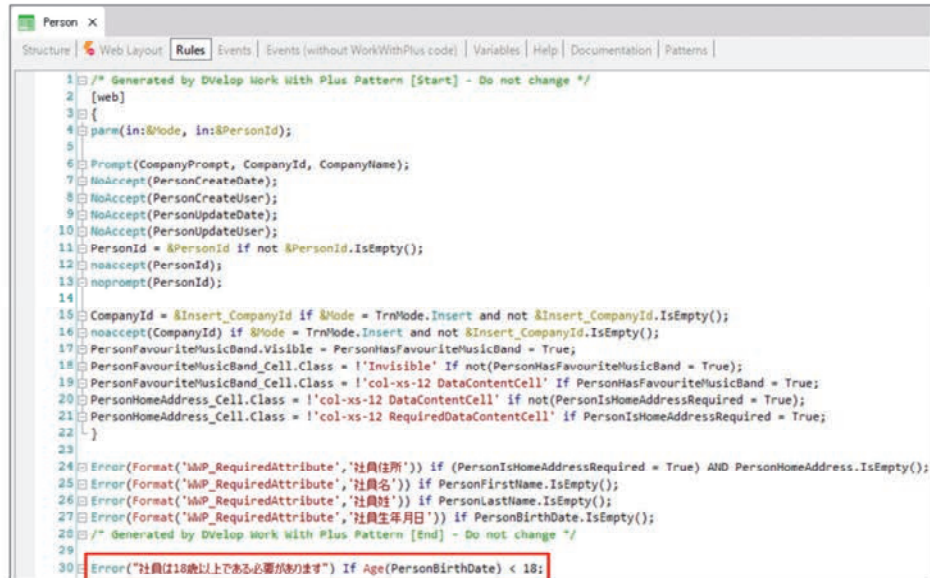
- [Visible Condition] プロパティ
- [Is visible initial appearance] プロパティ

Visible	True
Visible Condition	PersonHasFavouriteMusicBand = True
Is visible initial appearance	Not visible

Transaction の階層構造内に追加された [Attribute] ノードの [Visible] プロパティを利用することで、開発者が項目属性の表示/非表示を指定することができます。もちろん非表示として画面に含める必要がない場合は、ノードを削除することで表示させないように生成することも可能です。

[Visible] プロパティの値が [True] の場合、生成されたアプリケーションで表示されるコントロールとなりますが、特定の条件を満たす場合に表示/非表示を切り替えるために [Visible Condition] プロパティが用意されています。また、[Visible Condition] プロパティに値を設定した場合、さらに [Is visible initial appearance] プロパティが表示されます。このプロパティによって既定の挙動（表示/非表示）を指定できます。

Transaction: 機能 (Rules)



```
1 /* Generated by Develop Work With Plus Pattern [Start] - Do not change */
2 {web}
3 {
4   param(in:&Mode, in:&PersonId);
5
6   Prompt(CompanyPrompt, CompanyId, CompanyName);
7   NoAccept(PersonCreateDate);
8   NoAccept(PersonCreateUser);
9   NoAccept(PersonUpdateDate);
10  NoAccept(PersonUpdateUser);
11  PersonId = &PersonId if not &PersonId.IsEmpty();
12  noaccept(PersonId);
13  noprompt(PersonId);
14
15  CompanyId = &Insert_CompanyId if &Mode = TrnMode.Insert and not &Insert_CompanyId.IsEmpty();
16  noaccept(CompanyId) if &Mode = TrnMode.Insert and not &Insert_CompanyId.IsEmpty();
17  PersonFavouriteMusicBand.Visible = PersonHasFavouriteMusicBand = True;
18  PersonFavouriteMusicBand_Cell.Class = '!Invisible' if not(PersonHasFavouriteMusicBand = True);
19  PersonFavouriteMusicBand_Cell.Class = '!col-xs-12 DataContentCell' if PersonHasFavouriteMusicBand = True;
20  PersonHomeAddress_Cell.Class = '!col-xs-12 DataContentCell' if not(PersonIsHomeAddressRequired = True);
21  PersonHomeAddress_Cell.Class = '!col-xs-12 RequiredDataContentCell' if PersonIsHomeAddressRequired = True;
22 }
23
24 Error(Format('WMP_RequiredAttribute', '社員住所')) if (PersonIsHomeAddressRequired = True) AND PersonHomeAddress.IsEmpty();
25 Error(Format('WMP_RequiredAttribute', '社員名')) if PersonFirstName.IsEmpty();
26 Error(Format('WMP_RequiredAttribute', '社員姓')) if PersonLastName.IsEmpty();
27 Error(Format('WMP_RequiredAttribute', '社員生年月日')) if PersonBirthDate.IsEmpty();
28 /* Generated by Develop Work With Plus Pattern [End] - Do not change */
29
30 Error("社員は18歳以上である必要があります") If Age(PersonBirthDate) < 18;
```

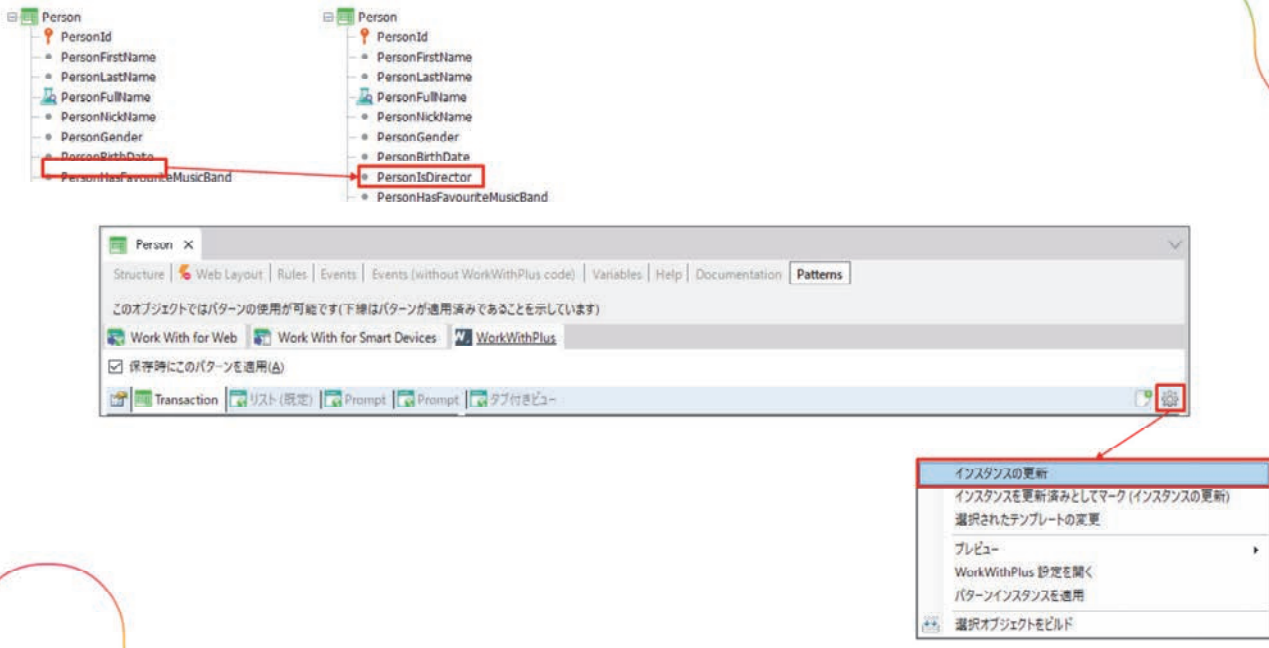
もし、WorkWithPlus for Web が提供するプロパティでは生成できないような
ルールの定義が要件として必要な場合、どうすればいいでしょうか？

Rules エlementに直接ルールを記述することができます。

この場合、WorkWithPlus for Web によって自動生成される範囲の外側で定義を行
う必要があります。

WorkWithPlus for Web が自動生成するコードは、常にその「開始」と「終了」の
行にコメント行が含まれています。

Transaction: インスタンスの更新



トランザクションオブジェクトの Structure エlement で項目属性が追加された場合、開発者はこの項目を階層構造へ追加する必要があります。前述の通り階層構造のカスタマイズにより、[Attribute] ノードを追加する方法で対応できますが、追加した項目属性分の作業が必要となります。また、この章で扱った Transaction 以外にも WorkWithPlus for Web の適用で生成されるオブジェクトはあり、すべてに追加しなければならない場合があります。

このような状況に対応するために「インスタンスの更新」という機能が用意されています。

画面右端の「歯車」のアイコンをクリックし、表示されるメニューから「インスタンスの更新」をクリックします。

これで自動的に Structure Element で追加したすべての項目属性を階層構造へ追加することができます。



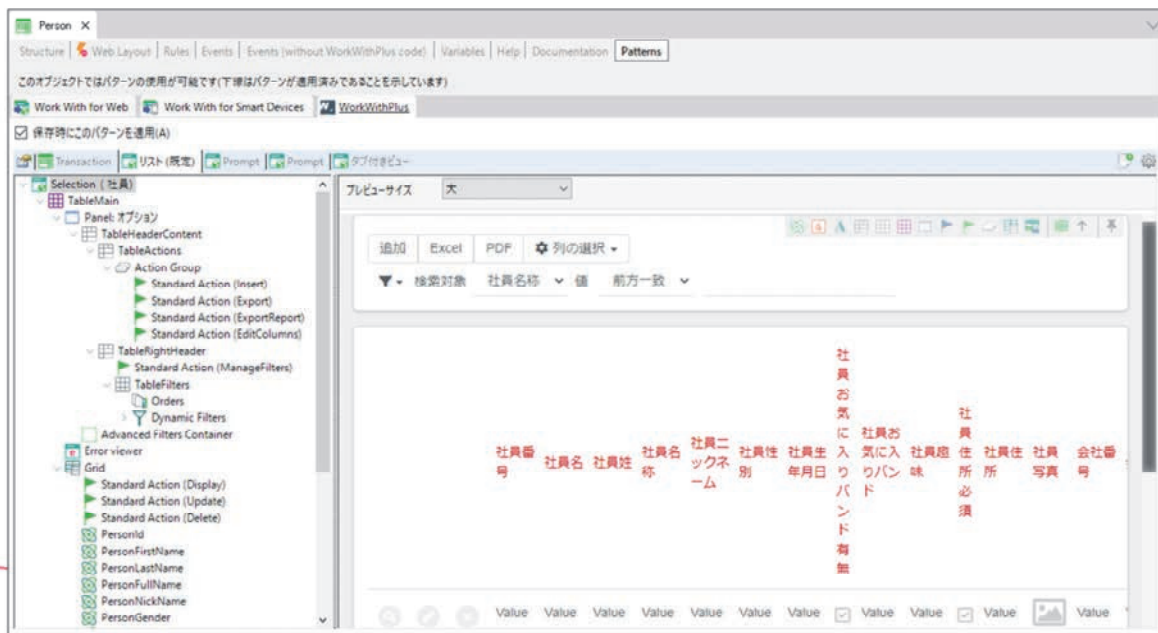
WORKWITHPLUS
FOR WEB

2. トランザクションオブジェクトへの適用

2.2. List



List



この章では WorkWithPlus for Web によって生成される List（GeneXus 上では「リスト（既定）」と表示されるタブ）のカスタマイズにフォーカスしていきます。

List の目的は、レコードを管理することです。たとえば、新しいレコードの追加、既存のレコードの変更または削除、レコードに関するすべての情報の表示などです。

この目的を満たすために List を構成するコントロールの中心として Grid コントロールがあります。

Grid コントロールに表示するレコードの項目属性や、変更などのアクションが含まれています。

また、レコードを管理するためには、対象のレコードを検索するためにフィルタを適用して検索を行える必要があります。

その他、表示されるレコードの順序をどの列を基準とするかも指定する必要があります。

このような機能をこのパターンインスタンスで定義することができます。

List: 機能 (Grid の拡張)

Grid

社員番号 ↑ 社員名 ▼ 社員姓 ▼ 社員名称 ▼ 社員ニックネーム ▼ 社員性別 ▼ 社員生年月日 ▼							
1	William	Rosen	Rosen William	Willy	男性	22/11/11	
2	Johanne	Petersons	Petersons Johanne	Johannie	女性	22/11/12	
3	Michael	Lopez	Lopez Michael	Mikey	男性	22/11/13	
4	Anne	Lookman	Lookman Anne	Annie	女性	22/11/14	
5	Leif	Herrera	Herrera Leif	Leif	男性	22/12/01	
6	Lauren	Herrera	Herrera Lauren	Laurie	女性	22/10/26	

1 / 1 ページ

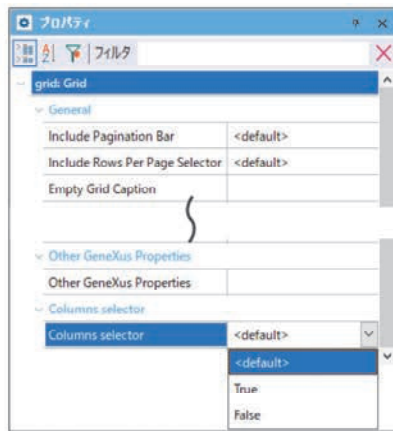
前へ 1 次へ

WorkWithPlus for Web では、List に標準の Grid コントロールと FreeStyleGrid コントロールどちらも追加することができます。
本コースでは標準の Grid コントロールを中心に List の機能説明を行います。

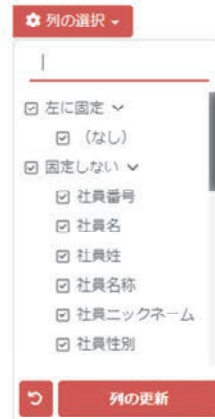
「標準」とお伝えしましたが、WorkWithPlus for Web によるカスタマイズを含む形となっているため、GeneXus 単体で生成される標準の Grid にはない機能が含まれています。

List: 機能（Grid の拡張）

[Columns Selector] プロパティ



Standard Action (EditColumns)



WorkWithPlus for Web によって生成される List の [Grid] ノードには、[Columns Selector] プロパティがあります。
このプロパティが [True] の場合、実行したアプリケーションではユーザーによって表示する列の選択、順序の変更ができます。
WorkWithPlus for Web ではこのプロパティの既定値は [True] となるため、パターンを適用した際には自動的に有効になります。

また、この機能を利用するためには、表示する列の選択、順序の変更を行うための [Standard Action (EditColumns)] ノードが必要となります。

（[Standard Action] ノードについては後で扱います）
このノードに基づき、実行画面に「列の選択」というボタンが表示され、クリックすることで表示する列の選択、順序の変更が可能となります。

List: 機能（Grid の拡張）

[Columns Selector] : 表示する列の選択、順序の変更

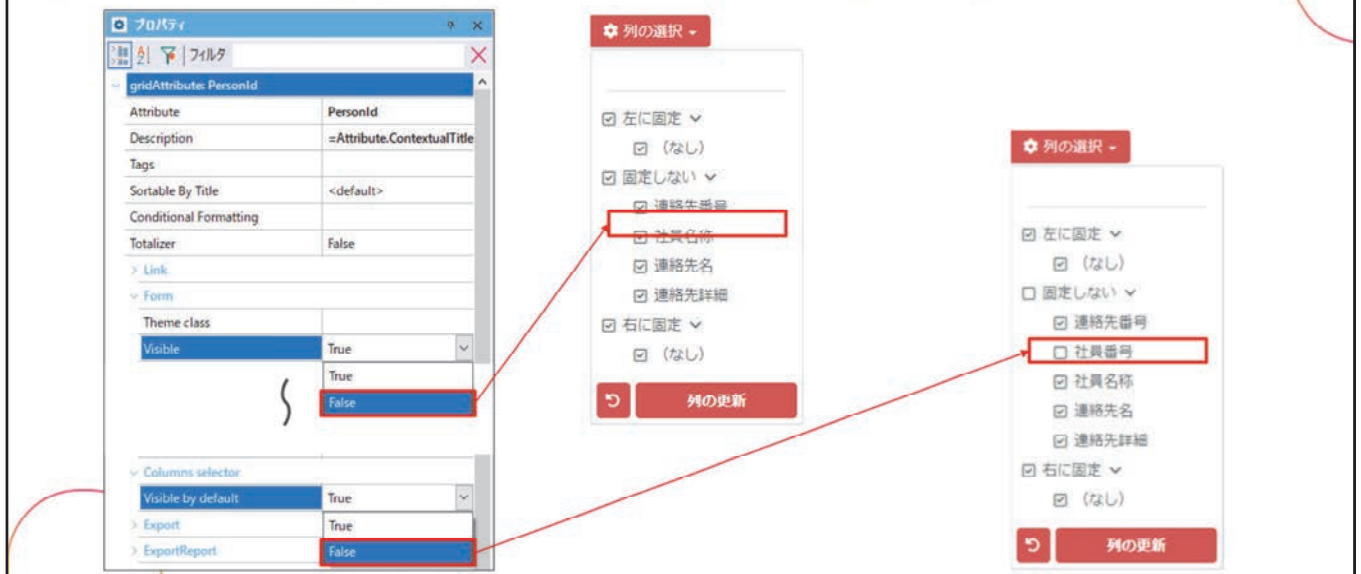


「列の選択」ボタンをクリックし、表示されるドロップダウンウィンドウでは、チェックボックスのオン/オフで表示する列を選択することができます。
表示されている項目属性のディスクリプションのドラッグアンドドロップで表示する順序を変更することができます。
操作が完了したら「列の更新」ボタンをクリックすることで変更が確定され、Grid の表示に反映されます。

「左に固定」、「右に固定」グループヘドドラッグアンドドロップすることで、任意の列をスクロールしても固定して表示されるように設定できます。

List: 機能 (Grid の拡張)

[Columns Selector] : [Visible] プロパティと [Visible by default] プロパティ



Columns Selector が有効な Grid では、列を非表示にする方法を 2 種類用意しています。

1 つ目は Transaction の項目属性ノードにおいても利用可能だった [Visible] プロパティです。

このプロパティを [False] に設定した場合、Grid 上に表示されず、Columns Selector による列の選択にも表示されません。

2 つ目は [Visible by default] プロパティです。

Columns Selector が有効な Grid において画面の初期表示時に列を表示するかどうか指定することができます。

List: 機能 (ページング)

[Page] プロパティ

プロパティ

selection: Selection (社員)

Caption expression

Description 社員

Page <default>

Paging Type <default>

Load Data <unlimited>

Is Main false

その他の指定方法

Page 10

Page Page.Rows

Page PrcPages()

Page &PageRow

社員番号	社員姓	社員名呼	社員ニックネーム	社員性別	社員生年月日	社員お気に入り!
1	Rosen	Rusan William	Willy	男性	22/11/11	<input type="checkbox"/>
2	Petersons	Petersons Johanne	Johanne	女性	22/11/12	<input type="checkbox"/>
3	Lopez	Lopez Michael	Mikey	男性	22/11/13	<input type="checkbox"/>
4	Lockman	Lockman Anne	Anne	女性	22/11/14	<input type="checkbox"/>
5	Hemera	Hemera Leif	Leif	男性	22/12/01	<input type="checkbox"/>
6	Hemera	Hemera Lauren	Laure	女性	22/10/26	<input type="checkbox"/>

1 / 1 ページ

前へ 1 次へ

社員番号	社員姓	社員名呼	社員ニックネーム	社員性別	社員生年月日	社員お気に入り!
1	Rosen	Rusan William	Willy	男性	22/11/11	<input type="checkbox"/>
2	Petersons	Petersons Johanne	Johanne	女性	22/11/12	<input type="checkbox"/>
3	Lopez	Lopez Michael	Mikey	男性	22/11/13	<input type="checkbox"/>
4	Lockman	Lockman Anne	Anne	女性	22/11/14	<input type="checkbox"/>
5	Hemera	Hemera Leif	Leif	男性	22/12/01	<input type="checkbox"/>
6	Hemera	Hemera Lauren	Laure	女性	22/10/26	<input type="checkbox"/>

Grid コントロールで 1 ページあたりに表示されるレコード数は、階層構造の一番親となる [selection] ノードが持つ [Page] プロパティで設定することができます。このプロパティでは、[<default>] と [<unlimited>] という値が選択できます。[<default>] の場合、WorkWithPlus for Web の既定値が利用されます。[<unlimited>] の場合、ページングは行われず、すべてのレコードが 1 ページで表示されます。そのため、ページング用のコントロールは生成されません。

このプロパティは選択可能な 2 つの値以外に手入力で指定することができます。入力できる値としては、実際に表示したいレコード数を数値で入力することや、列挙型ドメインの値、プロシーチャー呼び出しの結果、オブジェクト内の任意の変数が利用可能です。

List: 機能（フィルタ）

標準フィルタ

PersonFullName (<default>)

社員名称

動的な演算子が含まれたフィルタ

PersonFullName - Dyn. Operators (<default>)

StartsWith

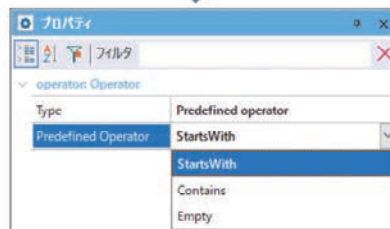
Contains

社員名称

前方一致

前方一致

次を含む



続いて List で定義可能なフィルタの機能について説明していきます。
WorkWithPlus for Web を利用し、実現できるフィルタは様々なものが用意されています。
ユーザーのニーズに合わせてご利用いただけます。

初めに紹介するフィルタは、「標準フィルタ」です。このフィルタは、WorkWithPlus for Web の既定の設定で定義された演算子（[=] や [>]、[Like] など）による条件が指定できます。

追加したい場合、任意の [Table] タイプのノードに対し [FilterAttribute] ノードを追加し、対象の項目属性を選択します。

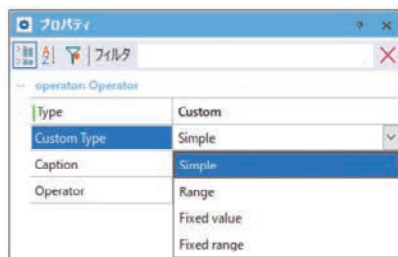
現在、この方法で追加した場合でも、多くの場合に次に紹介する「動的な演算子が含まれたフィルタ」が既定の状態となります。

そのため、利用する場合には、ノードを右クリックし、「標準フィルタ」に変換」を実行する必要があります。

続いて紹介するフィルタは、「動的な演算子が含まれたフィルタ」です。
このフィルタは、アプリケーションの実行時にユーザーによって演算子を動的に変更することができます。
そのため、ユーザーは決められた特定の演算子によるフィルタではなく、ニーズに合わせた演算子を利用することができます。
追加したい場合、上記の通り任意の [Table] タイプのノードに対し [FilterAttribute] ノードを追加し、対象の項目属性を選択することで多くの場合は追加できます。
もし、別のフィルタで追加された場合、ノードを右クリックし、「動的な演算子が含まれたフィルタ」に変換」を実行する必要があります。
追加できる演算子ノードには [Type] プロパティがあり、選択肢としては [Predefined operator] と [Custom] です。
既定で追加される演算子は [Type] プロパティが [Predefined operator] となり、WorkWithPlus for Web によって事前に定義された演算子です。
この場合、[Predefined Operator] プロパティで事前に定義された別の演算子に変更することができます。

List: 機能（フィルタ）

「動的な演算子」の種類



Type	Custom
Custom Type	Simple
Caption	Simple
Operator	Range
Fixed value	
Fixed range	



operator: Operator	
Type	Custom
Custom Type	Simple
Caption	
Operator	

operator: Operator	
Type	Custom
Custom Type	Range
Caption	
Operator	
Operator To	
Middle Text	

「動的な演算子が含まれたフィルタ」において追加することができる演算子には4つのタイプが用意されています。

事前に定義された演算子もこの4つのタイプのいずれかに該当します。

これらのタイプについて補足していきます。

はじめに、すべてのタイプに共通するプロパティとして [Caption] プロパティがあります。

このプロパティでは、実行時に演算子を選択するコンボボックスに表示する文字列を指定します。

Simple タイプの演算子では、1つの値を条件として入力でき、この条件を満たすデータをフィルタリングできます。

そのため、実行時にユーザーが値を入力できるフィールドが1つ追加され、演算子を [Operator] プロパティで指定します。

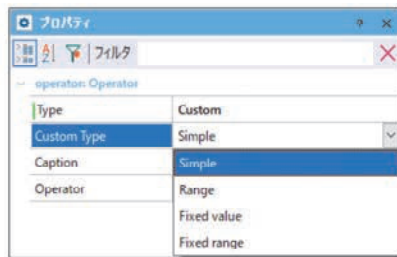
Range タイプの演算子では、2つの値を条件として入力でき、2つの条件を満たすデータをフィルタリングできます。

そのため、実行時にユーザーが値を入力できるフィールドが2つ追加され、1つ目のフィールドに対する演算子を [Operator] プロパティで指定し、2つ目のフィールドに対する演算子を [Operator To] プロパティで指定します。

また、2つのフィールドの間に表示する文字列を [Middle Text] プロパティで指定します。

List: 機能（フィルタ）

「動的な演算子」の種類



operator: Operator	
Type	Custom
Custom Type	Simple
Caption	Simple
Operator	Range
Fixed value	
Fixed range	

Fixed Value

社員生年月日 明日

operator: Operator	
Type	Custom
Custom Type	Fixed value
Caption	
Operator	
Fixed Value	

Fixed Range

社員生年月日 先週

operator: Operator	
Type	Custom
Custom Type	Fixed range
Caption	
Operator	
Operator To	
Fixed Value	
Fixed Value To	

Fixed value タイプの演算子では、事前に定義された固定値でデータをフィルタリングできます。

そのため、実行時にユーザーが値を入力できるフィールドは追加されません。演算子を [Operator] プロパティで指定し、フィルタリングに利用する値を [Fixed Value] プロパティで指定します。

Fixed range タイプの演算子では、事前に定義された2つの固定値でデータをフィルタリングできます。

つまり、2つの条件を満たすデータのみを表示します。

そのため、実行時にユーザーが値を入力できるフィールドは追加されません。

1つ目の演算子を [Operator] プロパティで指定し、1つ目の値を [Fixed Value] プロパティで指定します。

2つ目の演算子を [Operator To] プロパティで指定し、2つ目の値を [Fixed Value To] プロパティで指定します。

List: 機能（フィルタ）

範囲フィルタ

PersonBirthDate - Range (<default>)

分割された 2 つのフィールド

Picker Separate dates

社員生年月日

11 12 ~ 11 12

< 1月 2023 >

日	月	火	水	木	金	土
25	26	27	28	29	30	31
1	2	3	4	5	6	7
8	9	10	11	12	13	14
15	16	17	18	19	20	21
22	23	24	25	26	27	28
29	30	31	1	2	3	4

今日 消去

「範囲ピッカー」による 1 つのフィールド

Picker Range

社員生年月日

11 12 ~ 11 12

< 1月 2023 > 2月 2023 >

日	月	火	水	木	金	土	日	月	火	水	木	金	土
25	26	27	28	29	30	31	29	30	31	1	2	3	4
1	2	3	4	5	6	7	5	6	7	8	9	10	11
8	9	10	11	12	13	14	12	13	14	15	16	17	18
15	16	17	18	19	20	21	19	20	21	22	23	24	25
22	23	24	25	26	27	28	26	27	28	1	2	3	4
29	30	31	1	2	3	4	5	6	7	8	9	10	11

今日 消去

3 つめのフィルタとしては、「範囲フィルタ」です。

このタイプでは、2 つの値を条件として入力でき、2 つの条件を満たすデータをフィルタリングできます。

これは、「動的な演算子が含まれたフィルタ」において追加することができる演算子の 2 つめと同様の機能を持ったフィルタとなります。

追加したい場合、任意の [Table] タイプのノードに対し [FilterAttributeRange] ノードを追加し、対象の項目属性を選択することで追加できます。

もし、別のタイプで追加されたフィルタを範囲フィルタとして利用したい場合、ノードを右クリックし、「範囲フィルタ」に変換」を実行する必要があります。

範囲フィルタでは、対象の項目属性が日付型（Date 型、DateTime 型）の場合、[Picker] プロパティの設定によって異なる 2 種類の入力方法が提供されます。

・ 分割された 2 つのフィールド

通常の GeneXus によって生成される日付型項目の入力と同じように 1 つの入力欄には 1 つの日付を入力します。

そのため、画面上には日付を入力するための 2 つのフィールドが生成されます。

・ 「範囲ピッカー」による 1 つのフィールド

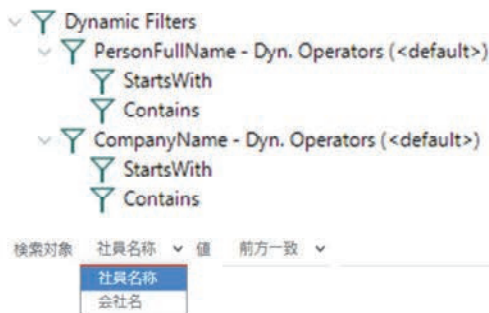
WorkWithPlus for Web によって提供される拡張機能となり、1 つの入力欄に -（ハイフン）を区切り文字として 2 つの日付を入力します。

そのため、画面上には 1 つのフィールドが生成されます。

この時、日付を選択するためのピッカーは、開始日をクリックし、つづいて終了日をクリックします。

List: 機能（フィルタ）

動的フィルタ



4 つめに紹介するフィルタは「動的フィルタ」です。

このフィルタは、WorkWithPlus for Web で利用可能なフィルタを実行時にユーザーが選択することができます。

例えば、ここまでにご紹介してきた「標準フィルタ」、

「動的な演算子が含まれたフィルタ」、「範囲フィルタ」すべて含めることが可能なフィルタとなり、実行時にユーザーに選択させることができます。

追加したい場合、任意の [Table] タイプのノードに対し [Dynamic Filters] ノードを追加し、追加されたノードにさらに任意のフィルタノードを追加します。

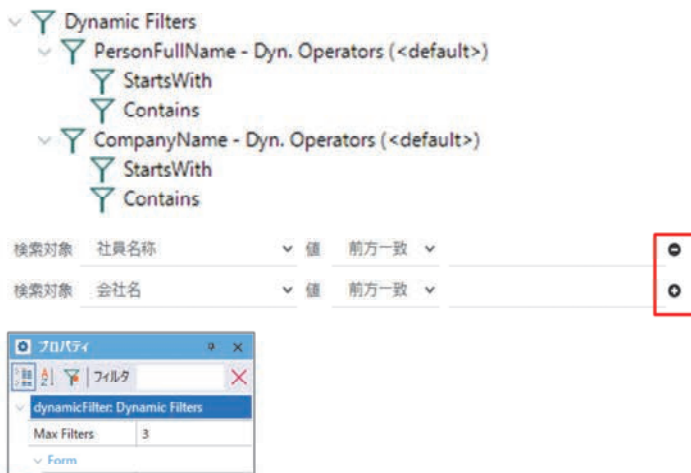
追加されたフィルタノードについてはここまでに説明してきたものと同一のため、動的フィルタを利用するうえで追加の設定はありません。

アプリケーションを実行すると、追加したフィルタノードは「検索対象」という形でコンボボックスから選択できます。

選択したフィルタに基づき表示が切り替わり、任意のフィルタを利用することができます。

List: 機能（フィルタ）

動的フィルタ



動的フィルタを利用する場合、追加されているフィルタノードに基づく複数の検索対象を指定することができます。

フィルタの右側に表示される「+」（プラス）アイコンをクリックすることでフィルタを追加することができます。

追加したフィルタが不要となった場合は、検索対象を変更するか、右側に表示される「-」（マイナス）アイコンをクリックすることでフィルタを削除することができます。

同時に利用できるフィルタ数は [Dynamic Filters] ノードの [Max Filters] プロパティで指定します。

このプロパティは既定値が 3 となっています。

プロパティの値を任意の値に変更することが可能ですが、大きくすればするほど GeneXus 上で自動生成される条件が多くなり、オブジェクトの保存や、ビルドに掛かる時間に影響を与えるものとなります。

開発元としての推奨値が既定値の 3 となります。

List: 機能（フィルタ）

動的フィルタ

固定フィルタ



検索対象 社員名称 値 前方一致 ▾

検索対象 会社名 ▾ 値 前方一致 ▾

動的フィルタで利用できる特別なフィルタ「固定フィルタ」を紹介します。
このフィルタは、動的フィルタと同じ外観と操作感で必ず表示させたい
フィルタがある場合に活用できます。
追加したい場合、[Dynamic Filters] ノードに対し [Fixed Filters] ノードを追加し、
追加されたノードにさらに任意のフィルタノードを追加します。
追加できるフィルタは、動的フィルタと同様です。

List: 機能（フィルタ）

タイトルフィルタ

社員番号 ↑	社員名称 ↓	社員趣味 ↓	社員住所必須 ↓
1	Rosen William	Play Cards	
2	Petersons Johanne	Play tennis	
3	Lopez Michael	Watch Football	
4	Lookman Anne	Go to cinema	
5	Herrera Leif	Go to park	
6	Herrera Lauren	Go to cinema	

↑ ↓ 昇順並び替え
↓ ↑ 降順並び替え
左に固定
右に固定

Go 🔍
Go to cinema (2)
Go to park (1)

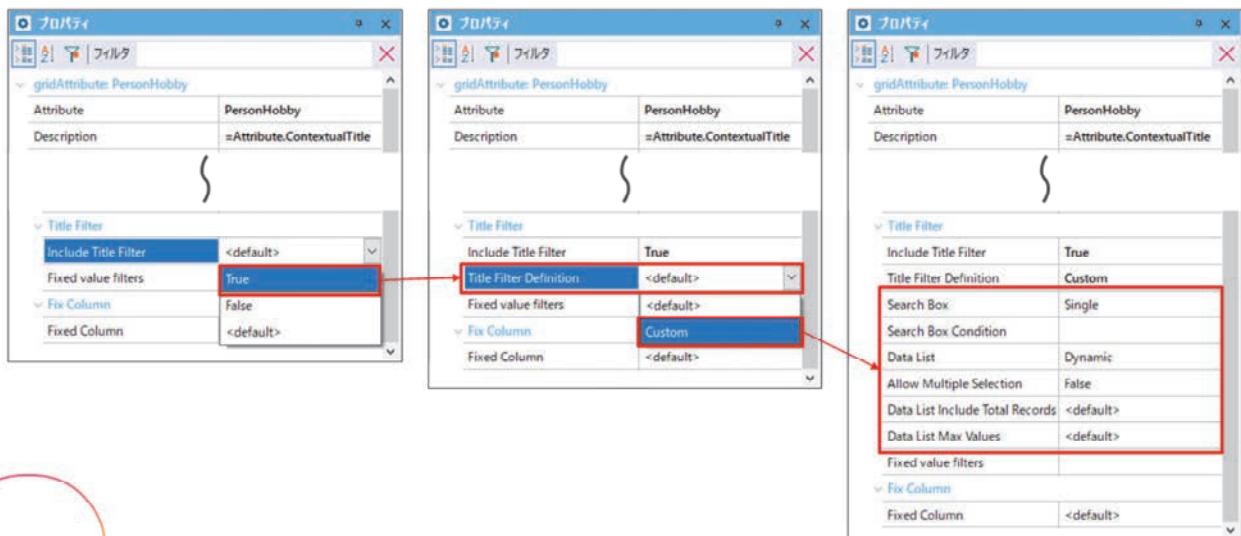
ここまで紹介してきたフィルタノードの追加によるフィルタ以外にも WorkWithPlus for Web がユーザーへフィルタを提供する機能があります。この機能は「タイトルフィルタ」と呼ばれ、グリッドの任意の列を対象にしてフィルタの適用ができます。また、データの並び替えを指定できます。タイトルフィルタという名前の通り、機能はグリッドに表示される列のタイトル部分に表示されます。タイトルの隣に表示される下向きの三角形（または上向き／下向きの矢印）をクリックすることで利用できます。

この機能では、あらかじめ GeneXus 上で実施した定義に基づき機能を利用できるようになっています。

WorkWithPlus for Web では、データタイプごとの既定値が設定されています。スライドのように文字列型の項目の場合、データの検索欄と登録されたデータの一覧が表示され、一覧から特定のレコードを選択した場合、このデータを持つレコードを対象としたフィルタが適用されます。この一覧に表示されるデータはサーバーサイドで処理が行われるため、現在表示されているページ内のレコードだけではなく、テーブルに登録されているすべてのレコードが対象となります。

List: 機能（フィルタ）

タイトルフィルタ



前述の通り、タイトルフィルタには既定の設定があります。
機能は既定で有効であり、データタイプごとに既定の設定が定義されています。

もし、既定と異なる設定を行う必要がある場合、[Grid] ノード内に追加されている項目属性ノードがもつ [Include Title Filter] プロパティを [True] に変更し、新たに表示される [Title Filter Definition] プロパティを [Custom] に変更します。この操作で表示されたプロパティを変更することで個別にタイトルフィルタをカスタマイズすることができます。

一部のプロパティをピックアップして紹介します。

Search Box :

タイトルフィルタ内で検索ボックスを含めるか、含める場合には単一か範囲かを指定できます。

Data List Include Total Records :

タイトルフィルタに表示するレコード一覧に同じ値の件数を表示するか指定することができます。

List: 機能（フィルタ）

フィルタの管理

Standard Action (ManageFilters)



実行したアプリケーションで活用できるフィルタに関連する挙動を持つ

[Standard Action (ManageFilters)] ノードがあります。

（[Standard Action] ノードについては後で扱います）

このノードに基づき、実行画面に漏斗のアイコンが表示されたボタンが表示されます。

このボタンをクリックすることで2種類の機能を利用できます。

まず1つめは「フィルタを消去」という機能です。

このオプションを利用することで、現在指定したフィルタを破棄することができます。

検索条件を初期化する際に活用できる機能です。

もう1つは「フィルタを別名で保存」という機能です。

この機能を利用することで、現在指定したフィルタを任意の名称で保存しておくことができます。

いくつかの条件を手動で設定する必要がなくなるため、作業効率の向上が見込めます。

保存したフィルタは必要に応じて名前の変更や、表示順の変更、不要な保存したフィルタを削除を行うことができます。

List: 機能（アクション）

Standard Action

Standard Action ()

Caption	
GXObject	(none)
Condition	
Popup	False
Control type	<default>
Class	
Tooltip	
Name	Update
Include action	Update
Confirmation	Delete
Confirm	Display
Image	CopyRecord
Image Type	Insert
Image	Export
Theme Class	ExportReport
Font Icon	ExportCSV

追加 Excel PDF 列の選択



続いては WorkWithPlus for Web において挙動を追加するための「アクション」という機能をご紹介します。
まず初めに、[Standard Action] というノードについてをご紹介します。

このノードを追加することによって WorkWithPlus for Web があらかじめ定義した挙動に基づくコードが生成されるオブジェクトに定義されます。
追加したい場合、任意の [Table] タイプのノードに対し [Standard Action] ノードを追加し、[Name] プロパティにて Standard Action を選択します。
各 Standard Action には、画面上で表示される既定のデザインが設定されているため、別途設定する必要はありません。
また、一部の Standard Action は、[Grid] タイプのノードで利用することができます。

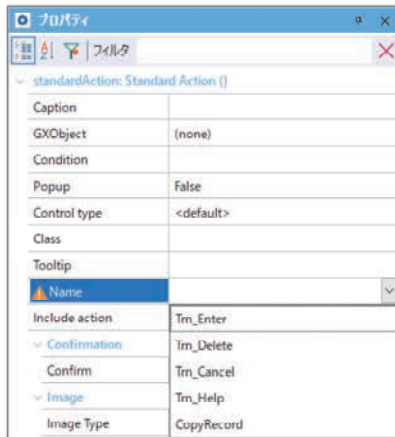
ここまでの内容で取り扱いのあった [Standard Action (EditColumns)] や、[Standard Action (ManageFilters)] もこの Standard Action の一種となります。
List では、上記 2 種の Standard Action に加え、Insert、Update、Delete、Export、ExportReport の 5 つが既定で追加されます。

デザインシステムウィザードの設定によって Display も既定で追加することができます。

List: 機能（アクション）補足

Standard Action

Standard Action ()



standardAction: Standard Action ()	
Caption	
GXObject	(none)
Condition	
Popup	False
Control type	<default>
Class	
Tooltip	
Name	
Include action	Tm_Enter
Confirmation	Tm_Delete
Confirm	Tm_Cancel
Image	Tm_Help
Image Type	CopyRecord

実行

終了

削除

既に取り扱った Transaction も [Standard Action] ノードを追加することができます。
ただし、List から生成される画面と Transaction から生成される画面では役割が異なるため、
利用可能な Standard Action が異なる点に注意が必要です。

List: 機能（アクション）

User Action

オブジェクトの呼び出し

User Action (UDelete)

プロパティ	
userAction: User Action (UDelete)	
Control type	Button
Name	UDelete
GXObject	DeleteCompany
General	

User Action (UDelete)

Parameters

<parameter>

プロパティ	
parameter: CompanyId	
Name	CompanyId

続いては [User Action] というノードについてご紹介します。

このノードを追加することによってアプリケーション固有の挙動を実装するコードを定義できます。

追加したい場合、任意の [Table] タイプのノードに対し [User Action] ノードを追加します。

追加した User Action のデザインは、各プロパティで指定することができます。また、User Action も、[Grid] タイプのノードに追加することができます。

挙動を実装するためには2つの実装方法を利用することができます。

1つめの方法は、「オブジェクトの呼び出し」です。

この場合、User Action の [GXObject] プロパティに実際に呼び出したいオブジェクトを指定します。

もし、呼び出すオブジェクトにパラメーターが必要な場合（オブジェクトに Parm ルールが定義されている場合）、[User Action] ノードに対し [Parameters] ノードを追加し、追加された [Parameters] ノードにさらに [Parameter] ノードを追加します。

[Parameter] ノードは、[Name] プロパティにてパラメーターとして渡す値を持つ項目属性や変数、列挙型ドメイン等を指定します。

List: 機能 (アクション)

User Action

イベントの定義

User Action (CreateData)

プロパティ	
userAction: User Action (CreateData)	
Control type	Button
Name	CreateData
GXObject	(none)
General	

```
Event 'DoCreateData'  
LoadCompany()  
LoadCountries()  
LoadPerson()  
Grid.Refresh()  
EndEvent
```

2つめの方法は、「イベントの定義」です。

この場合、User Action の [Name] プロパティに指定した値の頭に「Do」が足されたイベントで、実装したい処理を定義します。

そのため、この方法を利用する場合には、[User Action] ノードを追加し、[Name] プロパティを設定し、

一度オブジェクトを保存する必要があります。

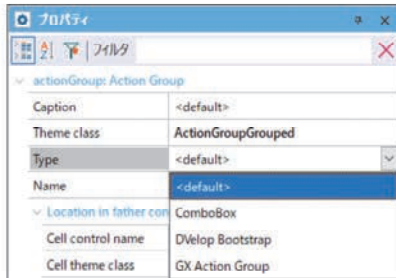
保存後、[User Action] ノードを右クリックし、「イベントへ移動」オプションをクリックすることで該当のイベントを簡単に表示することができます。

もし、[User Action] ノードが [Grid] タイプのノードに追加された場合、イベントは [Name] プロパティに指定した文字列を利用した変数のクリックイベントとなります。

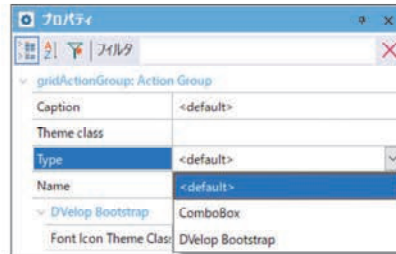
List: 機能（アクション）

Action Group

▼ Action Group



actionGroup: Action Group	
Caption	<default>
Theme class	ActionGroupGrouped
Type	<default>
Name	<default>
Location in father container	ComboBox
Cell control name	Dvelop Bootstrap
Cell theme class	GX Action Group



gridActionGroup: Action Group	
Caption	<default>
Theme class	<default>
Type	<default>
Name	<default>
Dvelop Bootstrap	ComboBox
Font Icon Theme Class	Dvelop Bootstrap

[Action Group] ノードを利用することで、[Standard Action] ノードや [User Action] ノードをグループ化することができます。
この [Action Group] ノードはグリッドの内外どちらへも追加することができますが、グリッドの内部の場合、[Type] プロパティで選択可能なタイプが1つ未対応となるため、表示されません。

追加したい場合、任意の [Table] タイプまたは [Grid] タイプのノードに対し [Action Group] ノードを追加します。
追加した [Action Group] ノードに任意の [Standard Action] ノードや [User Action] ノードを追加します。

利用可能な各タイプについては以下の通りです。

ComboBox :

一般的なコンボボックス形式で内包するアクションを表示します。
各アクションは必ず [Caption] プロパティを設定する必要があります。

Dvelop Bootstrap :

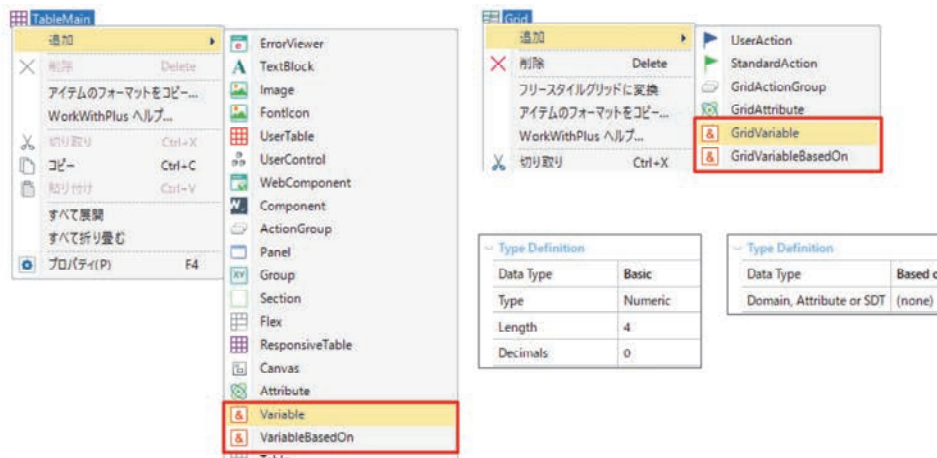
ドロップダウンオプション形式で内包するアクションを表示します。
各アクションは必ず [Caption] プロパティを設定する必要があり、さらに画像を設定することができます。

GX Action Group :

GeneXus が標準で持つ Action Group コントロールを利用してアクションのグループを生成します。
内包するアクションをすべて横に並べて表示します。
この時、アクションはボタン形式でも画像形式でも表示することができます。

List: 機能 (変数)

変数の追加



GeneXus では、DB に格納された値ではない可変の値を扱うため、変数を定義することができます。

WorkWithPlus for Web を利用し、生成される画面でも変数を追加することができます。

変数を追加したい場合、任意の [Table] タイプのノードまたは [Grid] タイプのノードに [Variable] ノードを追加します。

[Variable] ノードを追加する場合、2 種類のオプションが用意されています。

1 つは [Variable] というオプションです。この場合、[Data Type] プロパティが [Basic] として追加され、GeneXus で利用可能な基本データタイプを [Type] プロパティで指定できます。

この [Type] プロパティに指定したデータタイプに応じて、追加で設定できるプロパティが異なります。

別のオプションは [VariableBasedOn] です。この場合、追加時にダイアログが表示され、ナレッジベース内に定義されている「項目属性」、「ドメイン」から追加する変数のタイプを選択できます。

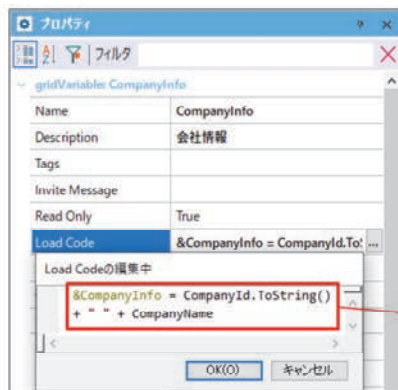
そのため、[Data Type] プロパティが [Based on] として追加されます。

また、[Grid] ノードに追加する場合、上記オプションの名前に [Grid] が付いています。

List: 機能 (変数)

Grid に表示するデータの取得

[Load Code] プロパティ



```
Event Grid.Load

/* Generated by Dvelop Work With Plus Pattern [Start] - Do not change */

&Display = '<i class="fa fa-search"></i>'
If (False)
    &Display.Link = PersonView.Link(PersonId, '')
    &Display.Class = '!Attribute'
Else
    &Display.Link = ""
    &Display.Class = '!Invisible'
EndIf
&Update = '<i class="fa fa-pen"></i>'
&Update.Link = BaseObjects.Person.Link(TrnMode.Update, PersonId)
&Delete = '<i class="fa fa-times"></i>'
&Delete.Link = BaseObjects.Person.Link(TrnMode.Delete, PersonId)
PersonFullName.Link = BaseObjects.PersonView.Link(PersonId, '')
CompanyName.Link = BaseObjects.CompanyView.Link(CompanyId, '')

&Selected = false

&CompanyInfo = CompanyId.ToString() + " " + CompanyName
&PersonIdOfInfo = PersonId
Do "GetIndexofSelectedRow"
If &i > 0
    &Selected = True
EndIf

/* Generated by Dvelop Work With Plus Pattern [End] - Do not change */

EndEvent
```

[Grid] タイプのノードに追加した [Variable] ノードに基づき生成される変数を利用し、表示したいデータがある場合、[Load Code] プロパティを利用する必要があります。

このプロパティでは、変数に対しデータを取得するためのコードを記述します。記述したコードは、List の [Grid.Load] イベントに追加されます。

このプロパティを利用することで、グリッドにデータが表示されるほか、WorkWithPlus for Web によって提供される様々な機能（例：Columns selector）で、この変数を常に意図した値で表示することができます。



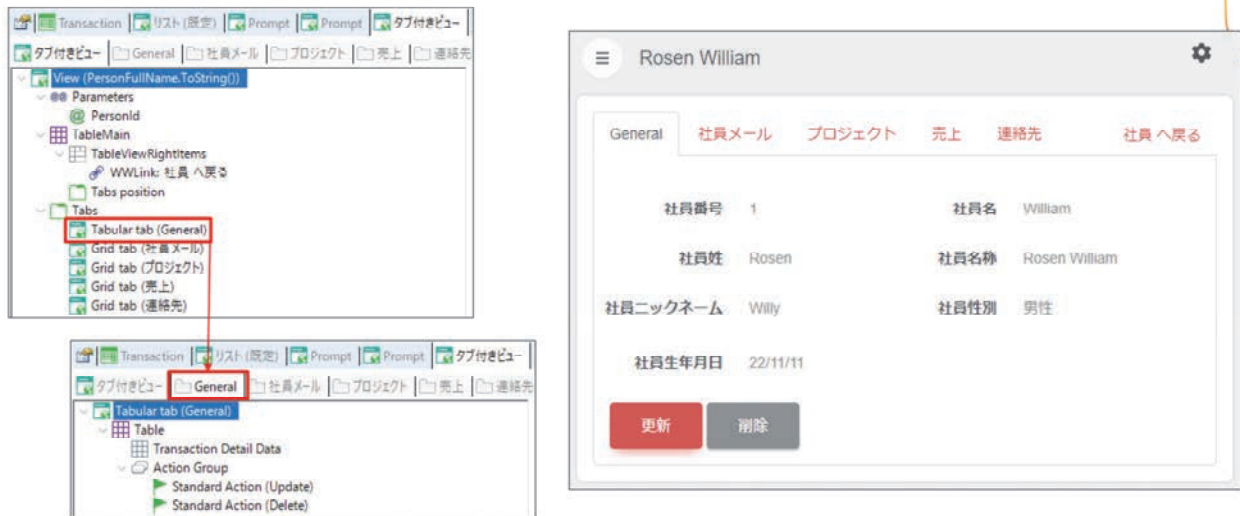
WORKWITHPLUS
FOR WEB

2. トランザクションオブジェクトへの適用

2.3. View



View



この章では WorkWithPlus for Web によって生成される View（GeneXus 上では「タブ付きビュー」と表示されるタブ）のカスタマイズにフォーカスしていきます。

View の目的は、トランザクションを利用し、登録されたレコードや、関連するレコードといった情報を表示することです。

関連するレコードは、選択されたレコードに対し、「1 対 1」や「1 対 多」の関係を持つトランザクションから取得されます。

この目的を満たすために View では、各情報を「タブ」形式で表示するように実装されます。

View の階層構造には必ず [Tabs] ノードが含まれていますが、生成された画面上での表示位置を決めるためには、[Tabs position] ノードを利用し、決定します。

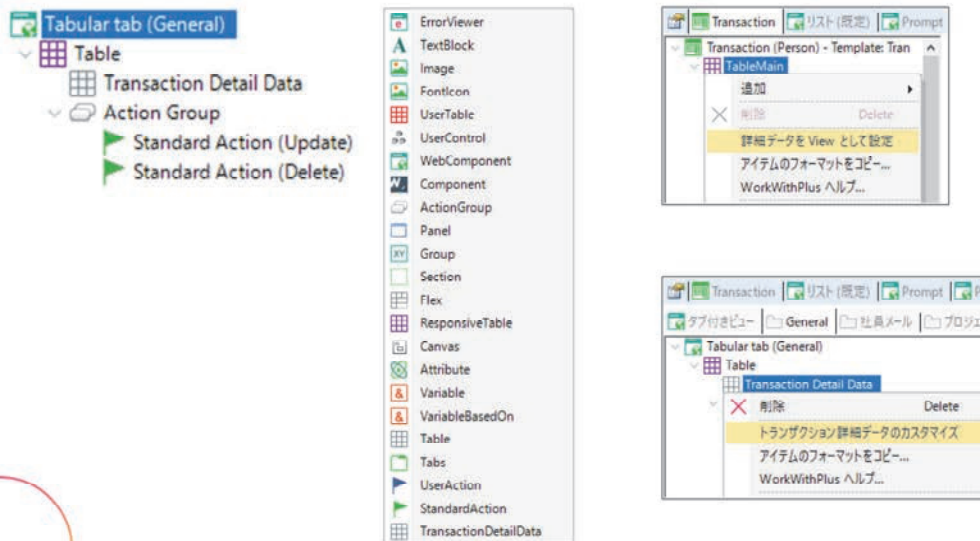
この理由は、View において「タブ」の定義が重要となり、画面上での表示につながるためです。

スライドに表示されているように [Tabs] ノードに追加された各タブが、「タブ付きビュー」タブ内のタブを形成しています。

View における [Tabs] ノードには追加できるタブが 3 種類あります。次のページからは各タブについて説明していきます。

View: タブの種類

Tabular tab



[Tabular tab] ノードを [Tabs] ノードに追加した場合、[Table] タイプのノードを利用し、単一のレコードを表示するタブを追加することができます。つまり、[Grid] タイプのノードを追加し、複数のレコードを表示するタブとしては利用できません。

View には、既定でパターンを適用しているトランザクションの情報を表示するための [Tabular tab] ノードが 1 つ含まれています。

また、データの表示以外に [Standard Action] が含まれ、データの更新や削除を呼び出す実装が含まれています。

レコードを表示するレイアウトの実装は、[Transaction Detail Data] ノードによって管理されます。

このノードは、Transaction の定義にて指定されてノード以下の内容を自動で生成する機能を持っています。

既定で対象となるノードは、[Name] プロパティに [TableAttributes] という値が設定された [Table] タイプのノードです。

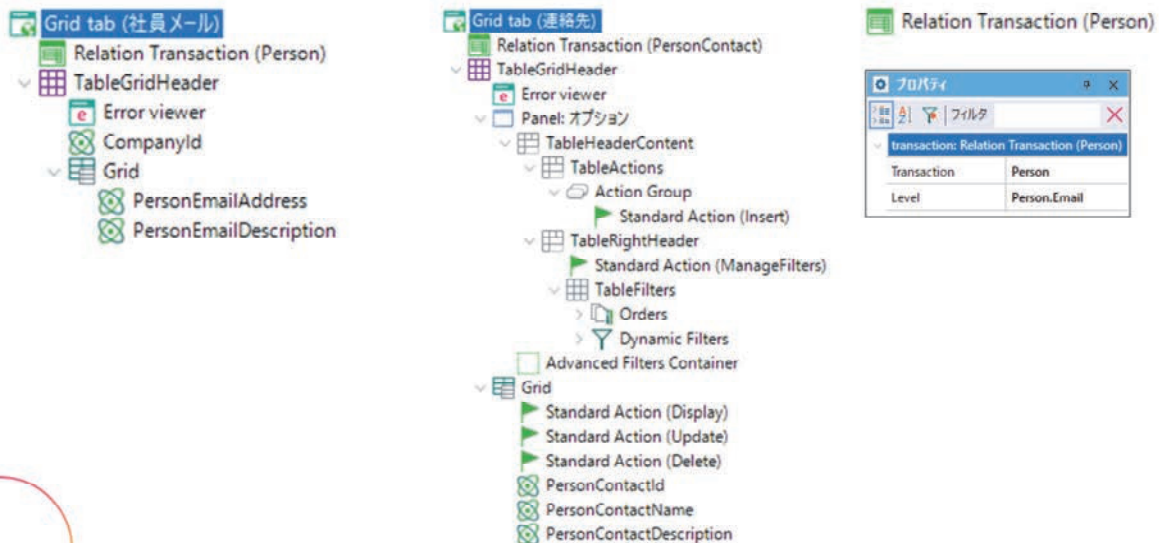
[Transaction Detail Data] ノードを利用することで、Transaction のレイアウトと自動で同期を取り、変更内容を共有することができます。

この対象ノードは、Transaction にて任意の [Table] タイプのノードを右クリックし、「詳細データを View として設定」オプションをクリックすることで変更できます。

また、Transaction と同期せず、独自の実装としたい場合、[Transaction Detail Data] ノードを右クリックし、「トランザクション詳細データのカスタマイズ」オプションをクリックすることで、独自のカスタマイズを行うことができます。

View: タブの種類

Grid tab



[Grid tab] ノードを [Tabs] ノードに追加した場合、[Grid] タイプのノードを利用し、選択されたレコードに関連のある複数のレコードを表示するタブを追加することができます。

この [Grid tab] タイプの階層構造は、List の定義と同様の機能を利用することができます。

View には、既定でパターンを適用しているトランザクションの第 2 レベルや、関連するトランザクションのレコードを表示するための [Grid tab] ノードが含まれています。

対象が 1 つもない場合、既定で追加される [Grid tab] ノードはありません。

また、パターン適用時に自動生成される [Grid tab] タイプの階層構造には、[Relation Transaction] ノードが含まれています。

このノードは、生成されるタブのベーステーブルを指定するために 2 つのプロパティが用意されています。

View: タブの種類

Grid tab

Transaction の [No Accept] プロパティが <default> の場合

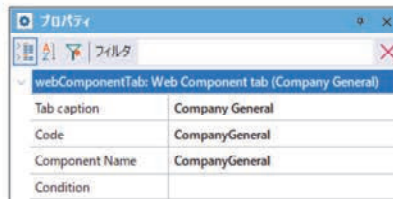


Transaction の説明にて対象の項目属性が「外部参照のキー項目」であり、[No Accept] プロパティが <default> の場合に、特定の条件下で有効となる NoAccept ルールが追加されることを説明していました。

この「特定条件」が View において関連するレコードを表示する [Grid tab] にて [Standard Action (Insert)]（画面では [追加] ボタン）のクリックです。View の機能にて表示している特定のレコードが呼び出された関連レコードの登録画面にて既定値として割り当てられ、NoAccept ルールによって読み取り専用となることが確認できます。

View: タブの種類

Web Component tab



[Web Component tab] ノードを [Tabs] ノードに追加した場合、ナレッジベース内ですでに作成された Web コンポーネントオブジェクトをタブとして追加することができます。

[Web Component tab] ノードを追加すると、Web コンポーネントオブジェクトの選択ダイアログが表示されるため、任意のオブジェクトを選択します。

選択した場合、自動的に対象の Web コンポーネントにて定義されている Parm ルールに基づき、[Parameters] ノードの追加とその配下に [Parameter] ノードが追加されます。

必要に応じてプロパティより引数で受け渡す項目属性や変数を指定します。

もし、Web コンポーネントオブジェクトを選択しなかった

（「キャンセル」をクリックした）場合、Web コンポーネントは紐づけられず、[Parameters] ノードの自動追加も行われません。

そのため、手動で [Component Name] プロパティより対象の Web コンポーネントオブジェクトを選択し、引数が必要な場合には、手動で [Parameters] ノードを追加する必要があります。

View: 画面遷移

WWLink: 社員 へ戻る

The screenshot displays the WWLink application interface. The top navigation bar includes tabs for 'General', '社員メール', 'プロジェクト', '売上', '連絡先', and '社員 へ戻る'. The '社員 へ戻る' tab is highlighted with a red box. Below the navigation bar, the 'View' page for 'Rosen William' is shown, with fields for '社員番号' (1), '社員姓' (Rosen), '社員ニックネーム' (Willy), and '社員生年月日' (22/11/11). A red arrow points from the '社員 へ戻る' button to the 'List' view, which is a table of employees.

社員番号	社員名	社員姓	社員名称	社員ニックネーム	社員性別	社員生年月日
1	William	Rosen	Rosen William	Willy	男性	22/11/11
2	Johanne	Petersons	Petersons Johanne	Johannie	女性	22/11/12
3	Michael	Lopez	Lopez Michael	Mikey	男性	22/11/13
4	Anne	Lookman	Lookman Anne	Annie	女性	22/11/14
5	Leif	Herrera	Herrera Leif	Leif	男性	22/12/01
6	Lauren	Herrera	Herrera Lauren	Laurie	女性	22/10/26

1 / 1 ページ

前へ 1 次へ

View にのみ追加することができるノードとして、[WorkWithLink] ノードがあります。

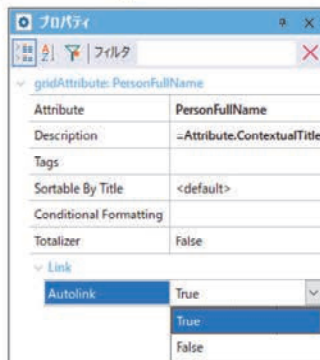
このノードに基づき生成される機能は、View からレコードを一覧表示する List へ遷移するためのリンクです。

View がタブ切り替え形式で表示しているため、いくつかのタブ切り替えを行った場合、ブラウザバックでは List へすぐに遷移できません。

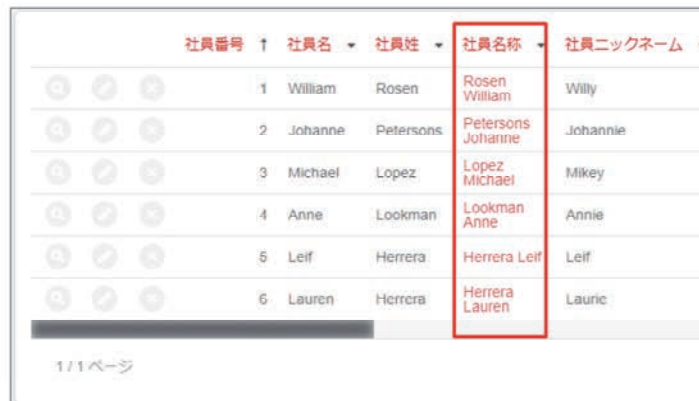
このような場合に活用できる機能です。

View: 画面遷移

[Autolink] プロパティ



Attribute	PersonFullName
Description	=Attribute.ContextualTitle
Tags	
Sortable By Title	<default>
Conditional Formatting	
Totalizer	False
Link	
Autolink	True
	True
	False



社員番号	社員名	社員姓	社員名称	社員ニックネーム
1	William	Rosen	Rosen William	Willy
2	Johanne	Petersons	Petersons Johanne	Johannie
3	Michael	Lopez	Lopez Michael	Mikey
4	Anne	Lookman	Lookman Anne	Annie
5	Leif	Herrera	Herrera Leif	Leif
6	Lauren	Herrera	Herrera Lauren	Laurie



会社番号	会社名
1	Dvelop
3	3M
4	Disco
6	Salto Grande
1	Dvelop
1	Dvelop

ここまで扱ってきた View へ遷移する場合、List や View で表示されるリンクになっている項目属性をクリックする必要があります。

このリンクとして表示される項目属性は、[Autolink] プロパティが [True] に設定された「名称項目属性 (DescriptionAttribute)」が対象となります。

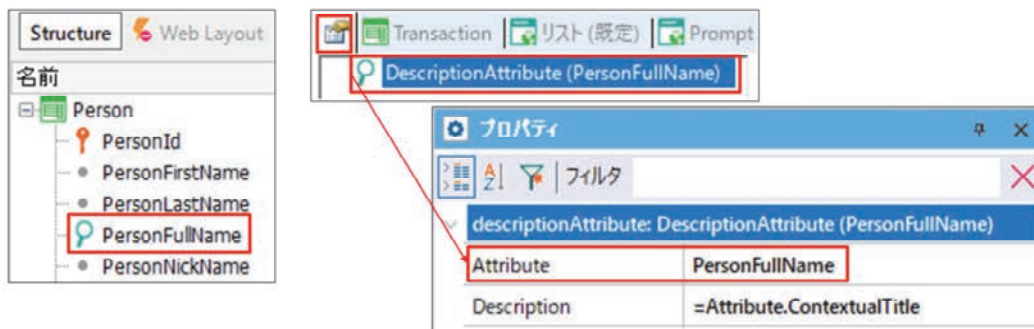
[Autolink] プロパティは既定で [True] に設定されているため、もし View へのリンクを生成しない場合、対象となる画面を生成する階層構造内で、対象の項目属性の [Autolink] プロパティを [False] に変更します。

リンクが自動的に設定される条件である

「名称項目属性 (DescriptionAttribute)」について次のページで詳しく説明しています。

View: 画面遷移

名称項目属性 (DescriptionAttribute)



「名称項目属性 (DescriptionAttribute)」は、GeneXus が持つ機能の 1 つです。トランザクションオブジェクトにて Structure の定義において虫眼鏡のアイコンが付いている項目が「名称項目属性」と呼ばれます。この項目属性は、トランザクションオブジェクトが生成する画面で登録されたレコードを呼称する際に利用できる値を保有していると想定されます。そのため、WorkWithPlus for Web では、View へのリンクを生成する対象としています。

ただし、WorkWithPlus for Web としての「名称項目属性」の既定値は Structure で虫眼鏡のアイコンが付いた項目属性ですが、必要に応じて変更することができます。

この場合、パターンインスタンスにおいて、Transaction の左に表示される Instance タブ (IDE 上ではアイコンの表示のみ) を選択し、[DescriptionAttribute] ノードの [Attribute] プロパティで変更することができます。

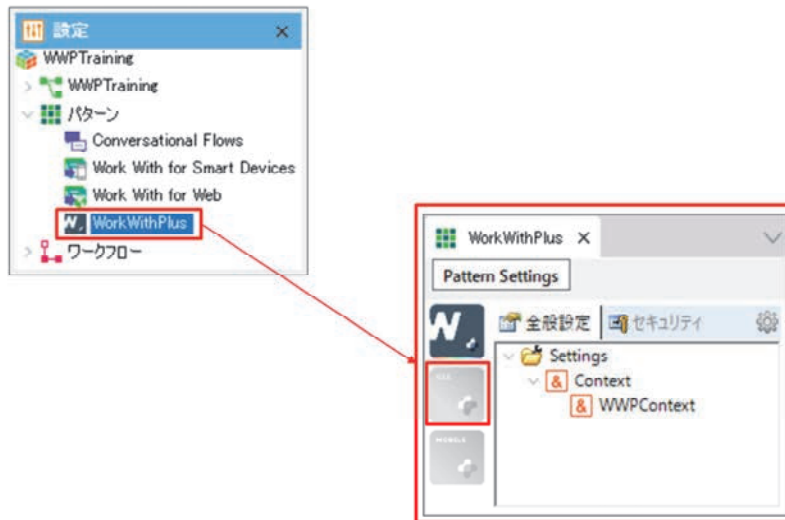


WORKWITHPLUS
FOR WEB

3. WorkWithPlus for Web の設定



WorkWithPlus for Web の設定



WorkWithPlus for Web には、ナレッジベース内で使用するすべてのパターンインスタンスが参照する WorkWithPlus for Web の設定があります。

この設定は、「設定」ウィンドウ内の [パターン] ノードに含まれる [WorkWithPlus] ノードをダブルクリックすることで表示することができます。この表示された画面は「WorkWithPlus 設定」と呼ばれます。

WorkWithPlus for Web のインストーラーは、WorkWithPlus for Native Mobile と統合されているため、この設定も統合されています。

本コースでの対象となるものは、「for Web」となるため、表示される 3 つのアイコンのうち、2 つめとなります。

では、具体的にどのような「設定」が含まれているか見ていきましょう。



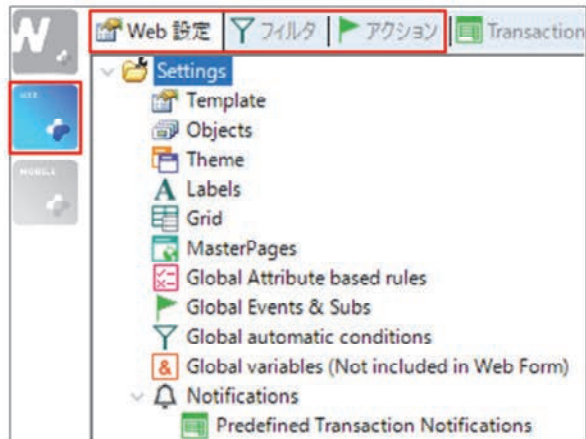
WORKWITHPLUS
FOR WEB

3. WorkWithPlus for Web の設定

3.1. 既定値の設定



既定値の設定

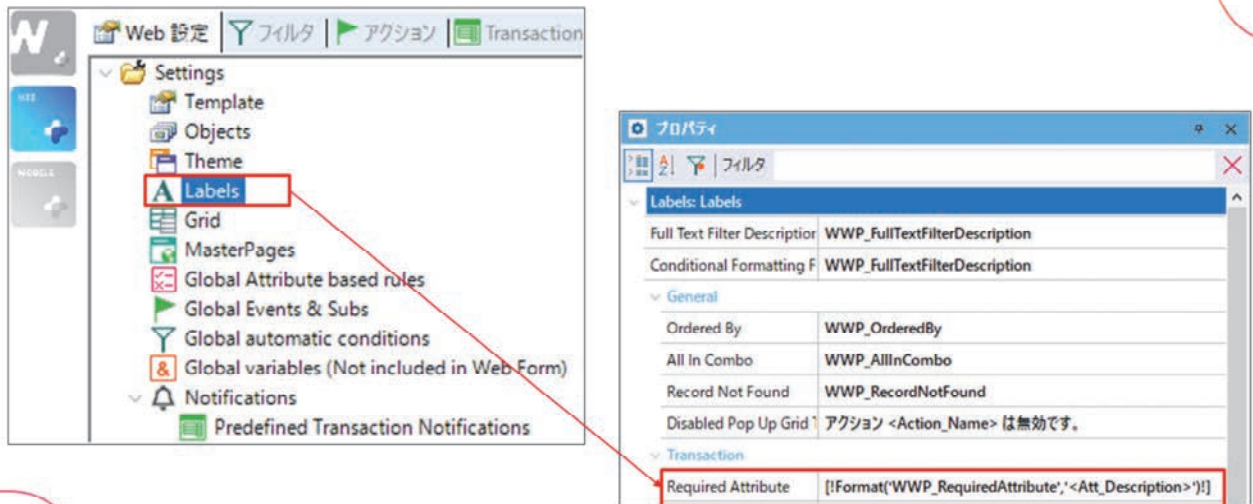


WorkWithPlus for Web において、既定値の設定は大きく 3 つにカテゴライズされています。
各タブ内には階層構造があり、各ノードのプロパティを変更し、既定値の設定を行います。

それぞれのタブごとに内容を確認していきます。

既定値の設定: Web 設定

[Labels] ノード



Web 設定 タブでは、WorkWithPlus for Web によって生成されるパターンインスタンスの様々な既定値を設定できます。

いくつかのノードで設定内容はカテゴリズされています。

この中でこれまでに扱った範囲に関連するいくつかのノード、そのプロパティにフォーカスします。

[Labels] ノードには、「ラベル」のように実行したアプリケーションで、文字列を表示する挙動にかかわる既定値を設定するプロパティがあります。

例えば、[Required Attribute] プロパティです。

Transaction において、[Is Required] プロパティについて学習しましたが、その際に自動で生成された Error ルールのメッセージがありましたが、そのエラーメッセージの既定値が定義されている場所がこのプロパティです。

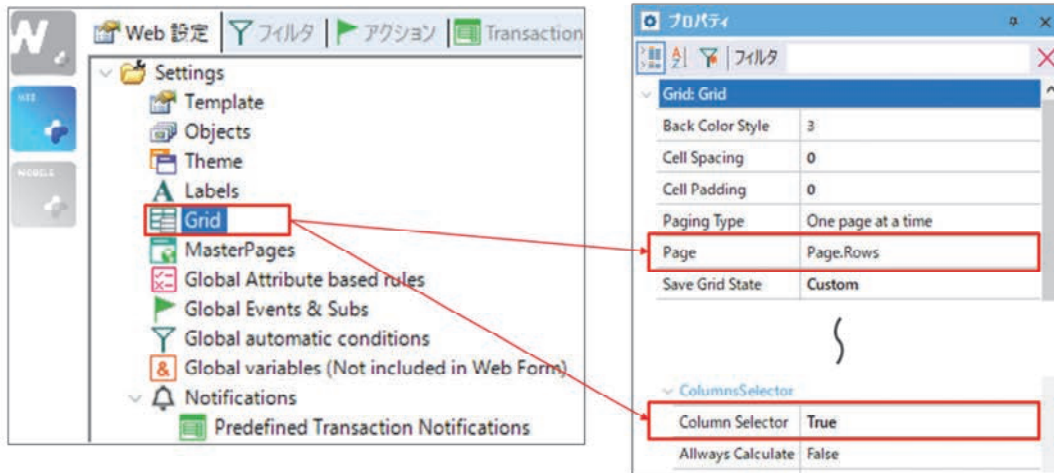
そのため、[Required Attribute] プロパティの内容を変更することで、ナレッジベース内の [Is Required] プロパティによって生成された Error ルールすべての表示メッセージが変更されます。

また、このプロパティに記載された内容を確認すると、不等号で囲まれた文字列（この場合、<Att_Description>）が含まれていることが確認できます。

このようなフォーマットをとった文字列は、WorkWithPlus for Web における予約語であり、実際に各オブジェクトへの適応時やプログラムの自動生成時に決められた値へ自動置換が発生するものです。

既定値の設定: Web 設定

[Grid] ノード



[Grid] ノードには、グリッドコントロールを生成する際の既定値を設定するプロパティがあります。

例えば、[Page] プロパティです。

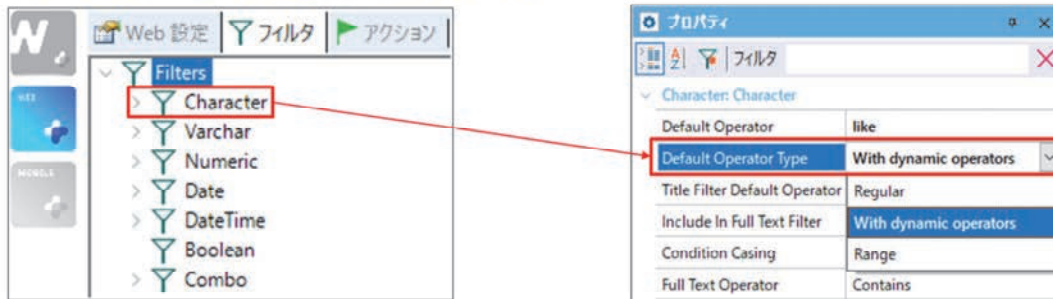
List の [Selection] ノードが持つ [Page] プロパティを [<default>] に設定した場合に参照されるプロパティです。

同様に、[Column Selector] プロパティは、List の [Grid] タイプのノードが持つ [Columns selector] プロパティを [<default>] に設定した場合の既定値です。

これらがデザインシステムウィザードの実行によって既定値として設定されたため、パターンを適用し、実行した際にすでに挙動が決まっていた理由です。

既定値の設定: フィルタ

データタイプごとの既定のフィルタタイプ



フィルタ タブでは、WorkWithPlus for Web で追加できるフィルタ機能に関連する既定値を設定できます。

このタブに含まれるノードは、GeneXus で利用可能な基本のデータタイプ 6 種とコントロールタイプがコンボボックス (Combo) の場合を対象としたノードです。

これは、データタイプごとに追加されるフィルタの既定値を指定するためです。コンボボックスも 1 つのノードとして含まれていますが、この場合は、どのようなデータタイプであっても適用できるため、既定値を設定するために用意されています。

任意のノードの [Default Operator Type] プロパティを確認します。

このプロパティで選択できる値は、List にてご紹介したフィルタの 3 種類となります。

[Boolean] ノードと [Combo] ノードのみが [Default Operator Type] プロパティの既定値が [Regular] となり、その他すべてがスライドのように

[With dynamic operators] となります。

そのため、追加するフィルタは既定で「動的な演算子を含むフィルタ」として追加されます。

既定値の設定: フィルタ

既定の「演算子」

- ▼ Date
 - ▼ LowerThan (Simple)
 - ▼ Equals (Simple)
 - ▼ HigherThan (Simple)
 - ▼ Past (Fixed value)
 - ▼ Yesterday (Fixed value)
 - ▼ Today (Fixed value)
 - ▼ Tomorrow (Fixed value)
 - ▼ InTheFuture (Fixed value)
 - ▼ LastWeek (Fixed range)
 - ▼ LastMonth (Fixed range)
 - ▼ LastYear (Fixed range)
 - ▼ ThisWeek (Fixed range)
 - ▼ ThisMonth (Fixed range)
 - ▼ ThisYear (Fixed range)
 - ▼ NextWeek (Fixed range)
 - ▼ NextMonth (Fixed range)
 - ▼ NextYear (Fixed range)
 - ▼ SpecificDate (Simple)
 - ▼ Range (Range)
 - ▼ RangePicker (Range)

プロパティ	
フィルタ	
operator: Past (Fixed value)	
Name	Past
Include Operator By Default	True
Type	Fixed value
Caption	WWP_FilterPast
Operator	<
Fixed Value	&Today

フィルタ タブの各データタイプノードには、子ノードとして [Operator] ノードを追加することができます。

これは、List の「動的な演算子を含むフィルタ」の説明時、演算子を追加した際に [Type] プロパティを [Predefined operator] と設定した場合に [Predefined operator] プロパティで選択できる既定の「演算子」の定義となります。

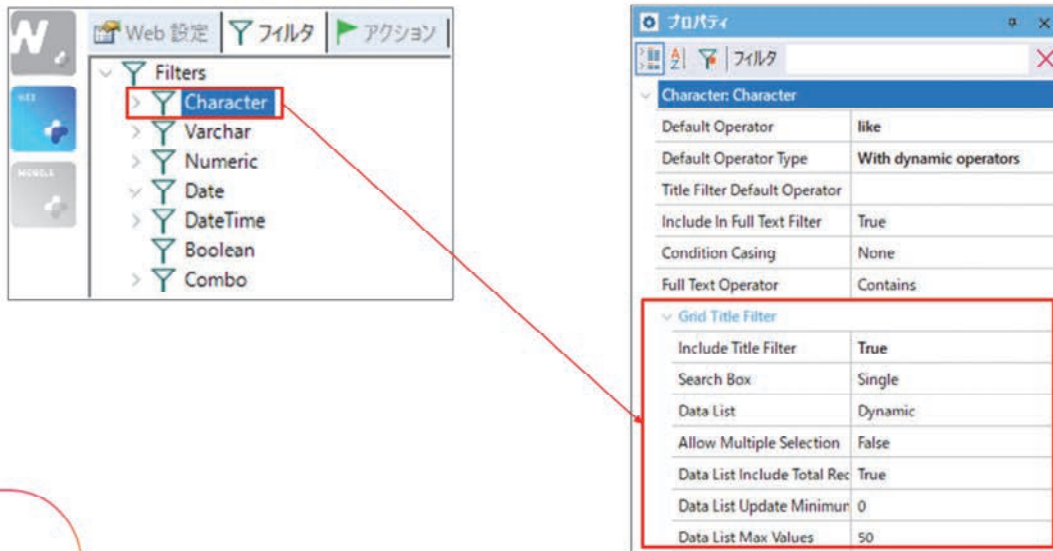
追加することができる演算子のタイプ ([Type] プロパティ) は、

「動的な演算子を含むフィルタ」にてご紹介した 4 種類です。

また、定義したものがすべて含まれることはなく、各 [Operator] ノードの [Include Operator By Default] プロパティが [True] のもののみが既定で含まれています。

既定値の設定: フィルタ

「タイトルフィルタ」の既定値



フィルタ タブの各データタイプノードには、「タイトルフィルタ」の既定値を設定するためのプロパティも用意されています。

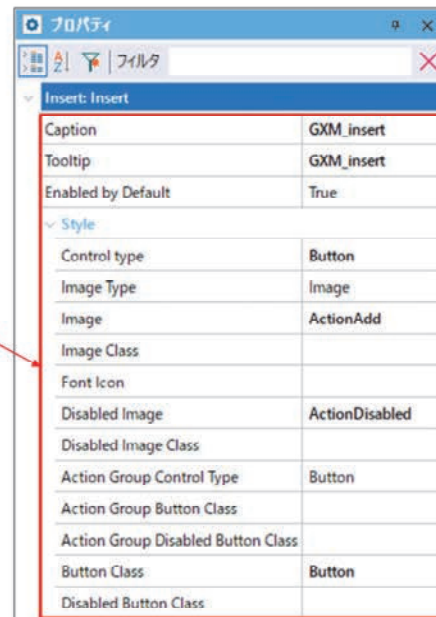
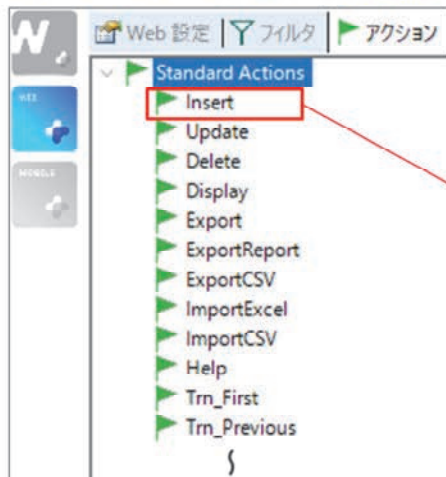
List などで [Grid] タイプのノードに含まれる項目属性を持つ

[Title Filter Definition] プロパティが [<default>] の場合に参照される定義です。

WorkWithPlus for Web が適用された場合、[Title Filter Definition] プロパティの既定値はこの [<default>] のため、既定で利用する場合は、必ずこの定義が参照されます。

既定値の設定: アクション

スタンダードアクションの既定値



アクションタブでは、WorkWithPlus for Web で追加できるスタンダードアクションの既定値を設定できます。

このタブに含まれるノードは、実際のパターンに [Standard Action] ノードを追加した際に、[Name] プロパティで選択できる値となるものが定義されています。

機能ごとに利用できるスタンダードアクションは異なりますが、既定値はここで一元管理されています。

一部のスタンダードアクションでは、固有のプロパティも用意されています。必要に応じてカスタマイズを実施します。



WORKWITHPLUS
FOR WEB

3. WorkWithPlus for Web の設定 3.2. テンプレート



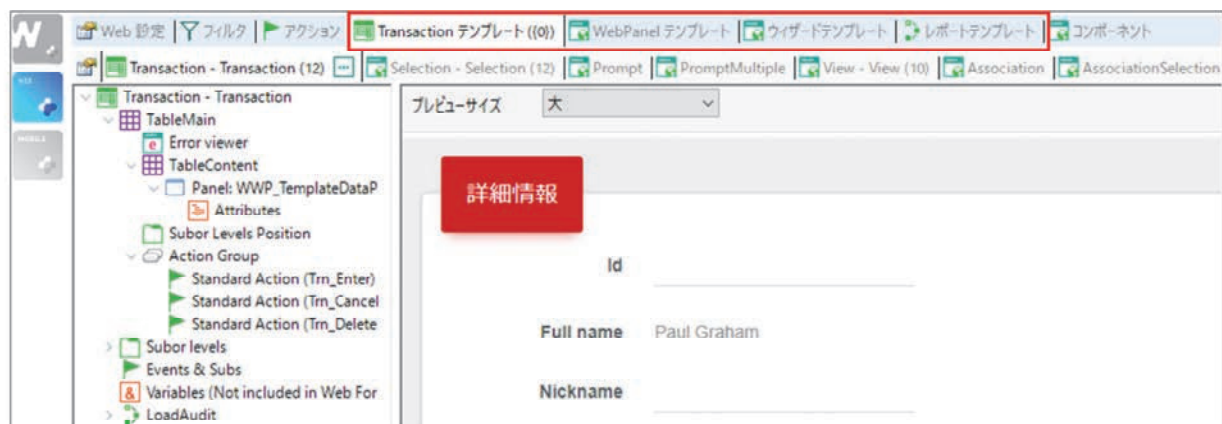
テンプレート

WorkWithPlus for Web を適用する場合、常にテンプレートを選択していました。このテンプレートを選択することにより、常に一定のデザインに基づき画面を生成することができました。では、統一したデザインで生成することだけがテンプレートを利用する理由でしょうか？

いいえ。テンプレートの利用はオブジェクトへの適用のタイミングだけではありません。

例えば、開発したアプリケーションですべての画面に統一したデザインの変更が必要となった場合、テンプレートに変更を加えることで、WorkWithPlus for Web では、このテンプレートを利用したすべてのオブジェクトを対象にパターンインスタンスの更新を行うことができます。このため、ナレッジベース内のすべてのオブジェクトを一つ一つ変更する必要がなく、テンプレートを変更するだけですべてを変更することができます。

テンプレート

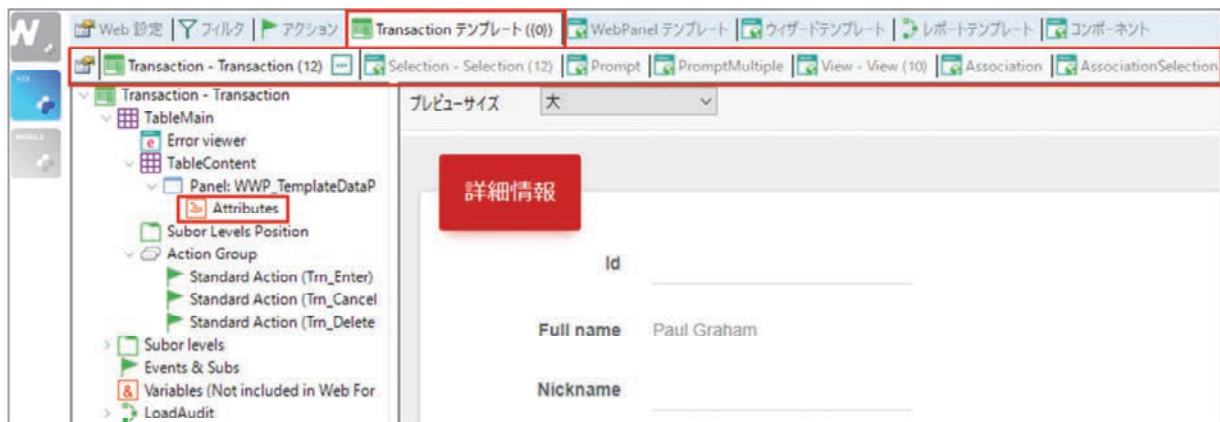


WorkWithPlus for Web では、利用できるテンプレートとして、Transaction テンプレート、WebPanel テンプレート、ウィザードテンプレート、レポートテンプレートの4つのタイプが用意されています。

各テンプレートが1つのタブとして表示されているので、各タブについて詳しく見ていきます。

テンプレート: Transaction テンプレート

Transaction タブ



Transaction テンプレート タブでは、トランザクションオブジェクトに適用可能なテンプレートを定義します。

ここまでに適用してきた通り、トランザクションオブジェクトに WorkWithPlus for Web を適用した場合、いくつかのオブジェクトが自動生成されていましたが、これらのオブジェクトすべてにテンプレートが用意されています。

（「Selection – Selection (12)」が List のテンプレートとなります）

そのため、Transaction テンプレート内にさらにタブが表示されています。表示されるタブは、トランザクションオブジェクトに適用した際に既定で表示されるものと、既定で表示されないものが含まれています。

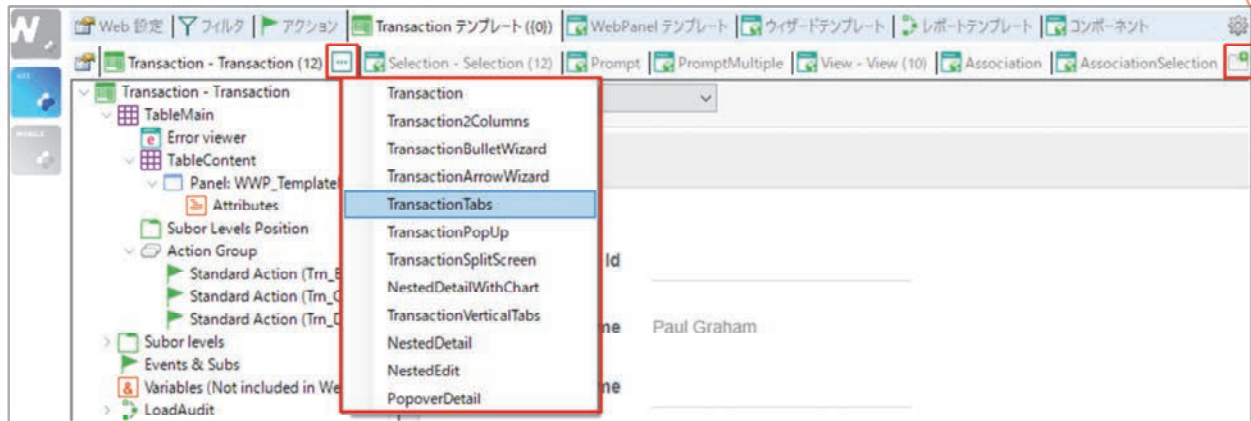
Transaction テンプレート タブ内の「Transaction – Transaction (12)」タブをクリックすると、スライドのようにトランザクションオブジェクト自身のデータ入力画面のテンプレートが階層構造で表示されます。

テンプレートのみで定義可能なノードがいくつか存在しています。
その中でも重要なノードが [Attributes] ノードです。
このノードは、WorkWithPlus for Web が適用された際に、そのトランザクションオブジェクトで定義されている項目属性を追加する位置を指定するためのノードです。
そのため、この [Attributes] ノードの親となる [Table] タイプのノードと同じ [Name] プロパティの値を持つ [Table] タイプのノードに [Attribute] タイプのノードが追加されます。
また、すべての [Attribute] タイプのノードで共通するプロパティ（例えば [Location in father container] プロパティ群）を設定することができます。

同じようにトランザクションオブジェクトにレベルがある場合、レベルを追加する位置を [Subor Levels Position] ノードで指定します。
実際に追加されるレベルに対するグリッドの定義は、[Subor levels] ノード内で定義します。

テンプレート: Transaction テンプレート

Transaction タブ



「Transaction – Transaction (12)」タブの「(12)」という表示は、定義されているテンプレート数を表示しています。

また、「-」の右側に表示されている文字列が選択したテンプレート名となります。

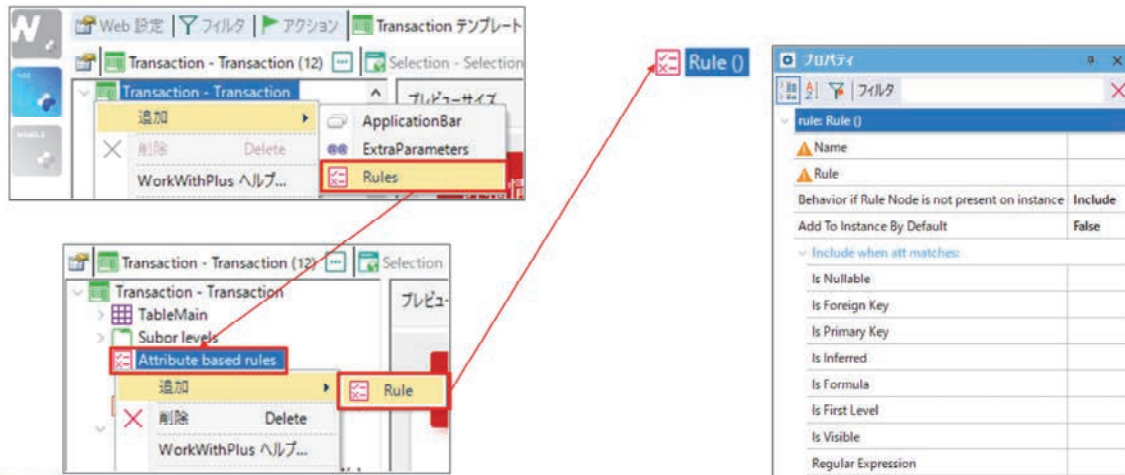
そのため、三点リーダーをクリックした場合、その他のテンプレートを選択するオプションが表示され、編集するテンプレートを切り替えることができます。例えば、Transaction の章にて画面をタブ形式で実装する簡単な方法として利用していた [TransactionTabs] テンプレートが含まれていることが確認できます。

もし、既存のテンプレートの編集ではプロジェクトの要件を満たせないという場合、画面右端の新規作成アイコンをクリックし、

「データの編集テンプレート (Web Panel)」をクリックすることで追加できます。追加する際には、テンプレートの名前等の指定が必要となりますが、魅力的な機能として、既存のテンプレートをベースに新規作成できる点です。

テンプレート: Transaction テンプレート

Transaction タブ - 自動ルール ([Attribute based rules] ノード)



WorkWithPlus for Web では、特定の正規表現または特定の条件を満たす項目属性がある場合、トランザクションオブジェクトにルールを自動で追加する機能があります。

この機能を利用することで、ナレッジベース内で条件に一致する項目属性へ個別にルールを定義する必要がなくなります。

また、現在定義されている範囲だけでなく、今後の改修で追加された項目属性が条件に該当する場合も対象となります。

この機能を利用する場合、Transaction テンプレートの「Transaction」タブにて、一番親の [Transaction] ノードを右クリックし、[Rules] を追加します。

「Transaction」タブ内に [Attribute based rules] ノードが追加され、このノードが自動ルールを定義するためのベースになります。

さらにこのノードを右クリックし、[Rule] を追加することで、実際にルールを定義するための [Rule] ノードが追加されます。

次のプロパティを設定していくことで、自動ルール of 定義を行います。

Name :

定義する自動ルール of 名前を指定します。この名前はテンプレート of 定義内で一意 of 名前である必要があります。

Rule :

実際に生成されるルールを記載します。ルール of 記載となるため、文末には必ず「;」が必要となります。

また、記載には WorkWithPlus for Web によって自動置換されるタグ

(例: <ATT_NAME>、<ATT_DESCRIPTION>) を利用可能です。

記入例: Default(<ATT_NAME>,&Today);

Behavior if Rule Node is not present on instance :

実際のパターンインスタンス (トランザクションオブジェクトへパターンを適用した際の階層構造) に [Rules] ノードが含まれていない場合 of 挙動を指定します。

既定では [Include] となるため、この自動ルールを追加する場合に [Rules] ノードがない場合、自動で追加されます。

Add To Instance By Default :

実際のパターンインスタンス

(トランザクションオブジェクトへパターンを適用した際の階層構造) にこの自動ルールを含めるかを指定します。

既定値は [False] となり、自動では含まれません。要件に合わせて [True] に切り替える必要があります。

Is Nullable :

「Null 許容」 of 設定内容を自動ルール of 条件に利用する場合に指定します。既定値 of [] の場合には、条件に利用されず、[True] の場合、「Null 許容」が [Yes]、[False] の場合、「Null 許容」が [No] of 項目属性を対象とします。

Is Foreign key :

外部キーであるかどうかを自動ルール of 条件に利用する場合に指定します。

既定値 of [] の場合には、条件に利用されず、[True] の場合、外部キー、[False] の場合、外部キー以外 of 項目属性を対象とします。

Is Primary Key :

主キーであるかどうかを自動ルール of 条件に利用する場合に指定します。

既定値 of [] の場合には、条件に利用されず、[True] の場合、主キー、[False] の場合、主キー以外 of 項目属性を対象とします。

Is inferred :

推測項目属性（外部キーによって値が決定する項目）であるかどうかを自動ルールの条件に利用する場合に指定します。

既定値の [] の場合には、条件に利用されず、[True] の場合、推測項目属性、[False] の場合、推測項目属性以外の項目属性を対象とします。

Is Formula :

グローバル式であるかどうかを自動ルールの条件に利用する場合に指定します。

既定値の [] の場合には、条件に利用されず、[True] の場合、グローバル式、[False] の場合、グローバル式以外の項目属性を対象とします。

Is First Level :

第 1 レベルに属するかどうかを自動ルールの条件に利用する場合に指定します。

既定値の [] の場合には、条件に利用されず、[True] の場合、第 1 レベル、[False] の場合、第 1 レベル以外に属する項目属性を対象とします。

Is Visible :

パターンインスタンスにおいて [Visible] プロパティの値を自動ルールの条件に利用する場合に指定します。

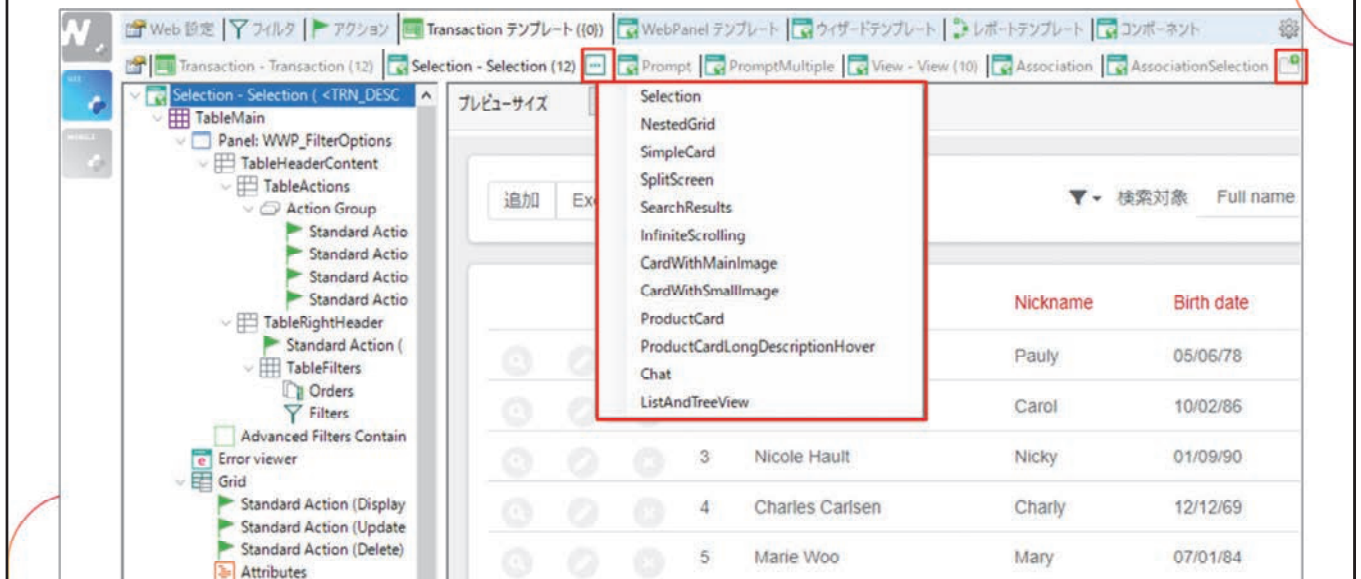
既定値の [] の場合には、条件に利用されず、[True] の場合、[Visible] プロパティが [True]、[False] の場合、[Visible] プロパティが [False] の項目属性を対象とします。

Regular Expression :

指定された正規表現の文字列に一致する項目属性を対象とします。

テンプレート: Transaction テンプレート

Selection タブ

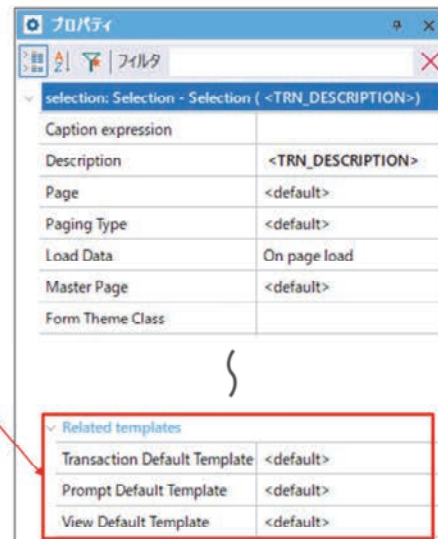
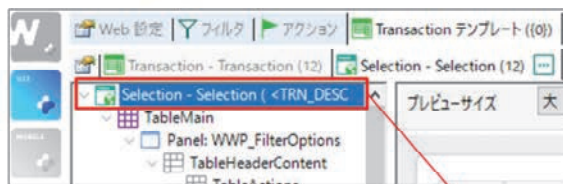


「Transaction – Transaction (12)」タブ以外のタブについても同様に
トランザクションオブジェクトに WorkWithPlus for Web を適用した際の
各オブジェクトのテンプレートの定義です。
また、WorkWithPlus for Web を適用する際に指定している「テンプレート」は、
「Selection – Selection (12)」タブで表示される List のテンプレートでした。
その理由は、選択した List のテンプレートに合わせてそれ以外のオブジェクトに
対するテンプレートが決定するためです。（設定方法は次のスライド）

また、「Transaction – Transaction (12)」タブと同様にほかのタブでも複数の
テンプレートが定義されている場合、三点リーダーをクリックし、
別のテンプレートに切り替えることができます。
利用するテンプレートを新規に作成しなければならない場合、新規作成
アイコンをクリックし、新規テンプレートを追加することができます。

テンプレート: Transaction テンプレート

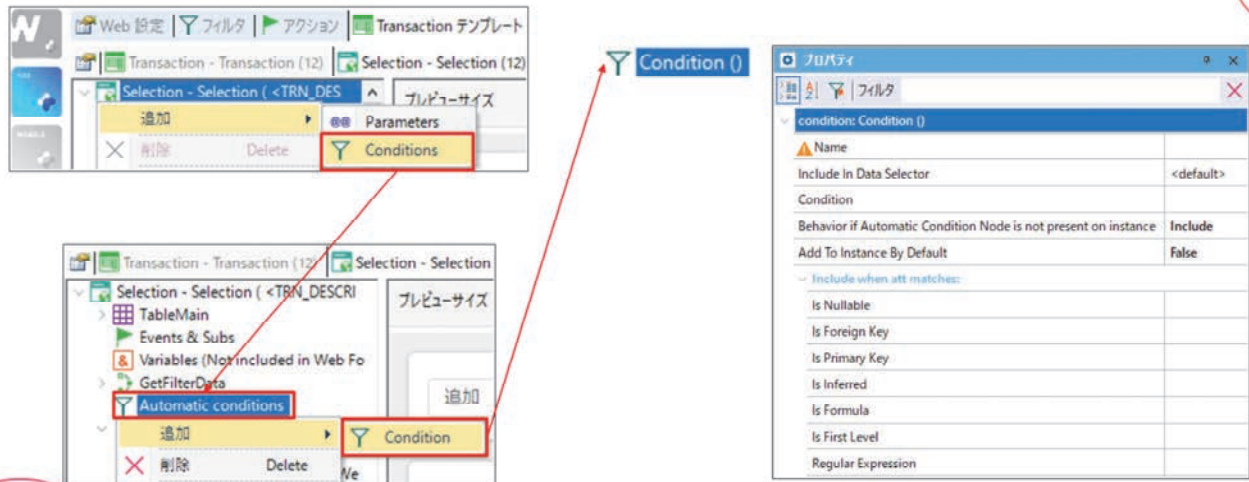
Selection タブ



「Selection – Selection (12)」タブで表示される一番親である [Selection] ノードのプロパティにある [Related templates] プロパティ群に各オブジェクトを生成する際に既定で適用されるテンプレートを指定できます。
既定値は [<default>] となり、各タブ名と同名のテンプレートが利用されます。
このように WorkWithPlus for Web を適用した際に関連するテンプレートが決定するため、トランザクションオブジェクトに適用する際に List に対するテンプレートのみを選択しています。

テンプレート: Transaction テンプレート

Selection タブ - 自動条件 ([Automatic conditions] ノード)



WorkWithPlus for Web では、特定の正規表現または特定の条件を満たす項目属性がある場合、List におけるデータフィルタリング条件を自動で追加する機能があります。

このデータフィルタリング条件は、「フィルタ」にてご説明したようにユーザーが実行時に値を指定するものではなく、あらかじめフィルタの値が決められたものの追加です。

この機能を利用することで、ナレッジベース内で条件に一致する項目属性に対するデータフィルタリング条件を個別に定義する必要がなくなります。

また、現在定義されている範囲だけでなく、今後の改修で追加された項目属性が条件に該当する場合も対象となります。

この機能を利用する場合、Transaction テンプレートの「Selection」タブにて、一番親の [Selection] ノードを右クリックし、[Conditions] を追加します。

「Selection」タブ内に [Automatic conditions] ノードが追加され、このノードが自動条件を定義するためのベースになります。

さらにこのノードを右クリックし、[Condition] を追加することで、実際にデータフィルタリング条件を定義するための [Condition] ノードが追加されます。

多くのプロパティは「自動ルール」と同名であり、設定結果についても同様となり、異なる点のみ以下へ記載します。

Include in Data Selector :

定義したデータフィルタリング条件を WorkWithPlus for Web が自動生成するデータセレクトオブジェクトにも含めるかを指定します。

[<default>] の場合、[True] を意味し、含まれます。含めたくない場合、[False] を選択します。

Condition :

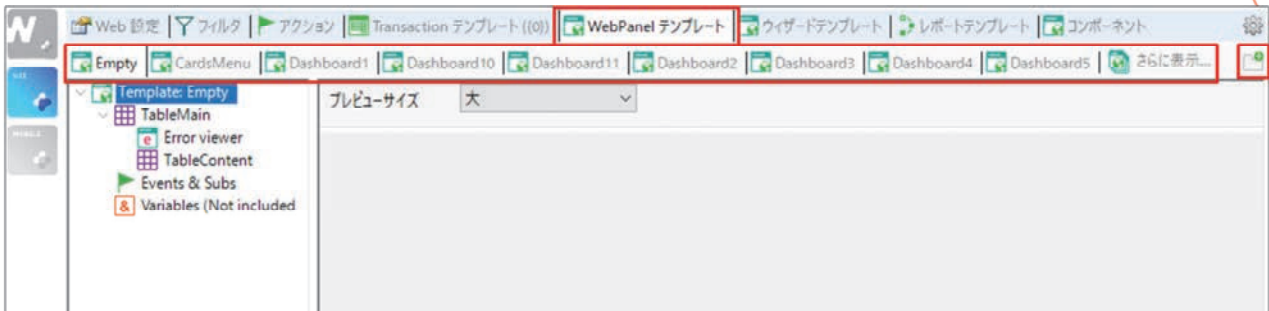
実際に生成されるデータフィルタリング条件を記載します。文末には必ず「;」が必要となります。

また、記載には WorkWithPlus for Web によって自動置換されるタグ

（例：<ATT_NAME>、<ATT_DESCRIPTION>）を利用可能です。

記入例：<ATT_NAME> >= &Today;

テンプレート: WebPanel テンプレート



WorkWithPlus for Web を適用可能なオブジェクトはトランザクションオブジェクトではありません。

トランザクションオブジェクト以外に適用可能なオブジェクトとして、Web パネルオブジェクトとプロシージャオブジェクトがあります。実際に適用した場合については、この後取り扱いますが、この章でまずはそれぞれのテンプレートについてご紹介します。

WebPanel テンプレート タブでは、Web パネルオブジェクトに適用可能なテンプレートを定義します。

あらかじめ定義されているテンプレートは、「Empty」という空のテンプレートと、デザインシステムウィザードにて、選択可能だったホーム画面のテンプレートです。

ただし、Web パネルオブジェクトに適用できるテンプレートは先ほどまでご紹介していた Transaction テンプレート タブ内で定義されていた各タブのテンプレートも適用することができます。

そのため、Web パネルオブジェクトでデータの登録画面を定義する場合、Transaction テンプレートを利用することができます。

また、この後紹介する ウィザードテンプレート タブで定義されたテンプレートも利用できます。

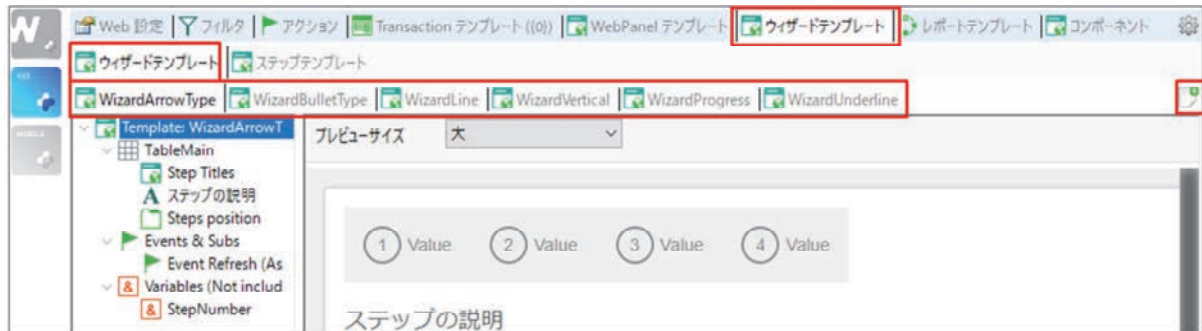
もし、既存のテンプレートの編集ではプロジェクトの要件を満たせないという場合、画面右端の新規作成アイコンをクリックし、

「WebPanel テンプレート」をクリックすることで追加できます。

追加する際には、テンプレートの名前等の指定が必要となりますが、魅力的な機能として、既存のテンプレートをベースに新規作成できる点です。

テンプレート: ウィザードテンプレート

ウィザードテンプレート タブ



ウィザードテンプレートタブでは、決められた内容を順番に表示し、ユーザーの操作性を高めた画面を自動生成するためのテンプレートを定義できます。ウィザードテンプレートタブには、さらに2つのタブが用意され、1つは同名のウィザードテンプレートタブ、もう1つはステップテンプレートタブです。それぞれのタブでどのような定義を行っているか確認していきます。

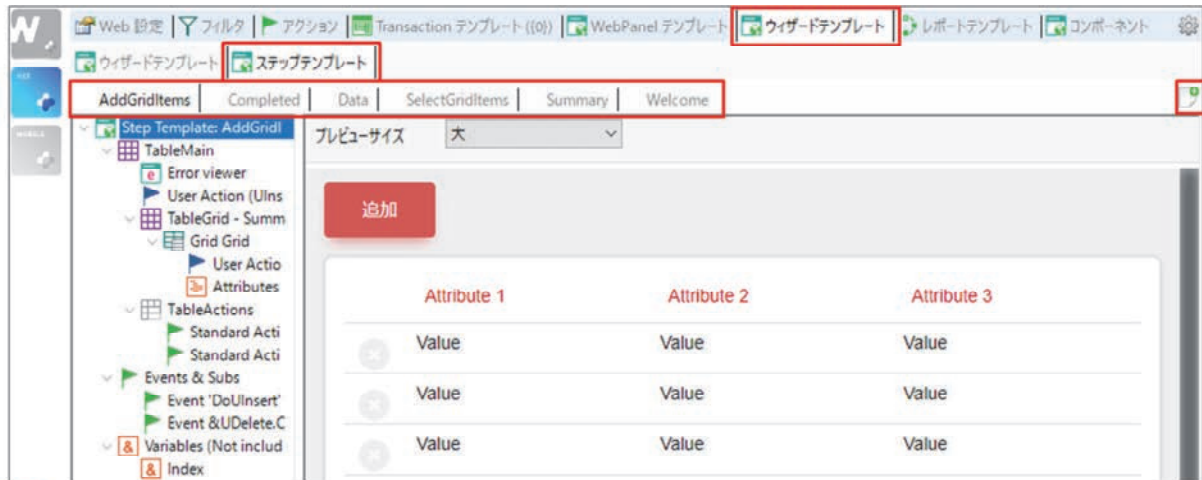
まずは、ウィザードテンプレートタブをフォーカスします。このウィザードテンプレートタブは、いくつかの「ステップ」と呼ばれるコンポーネントを切り替えて表示する画面のテンプレートを定義しています。実際の内容を表示する「ステップ」については、次のスライドでフォーカスします。

ウィザードテンプレートには、必ず [Step Titles] ノードと [Steps position] ノードが含まれます。
[Step Titles] ノードは、表示している「ステップ」のタイトルを表示する位置を指定します。
[Steps position] ノードは、「ステップ」の内容を表示する位置を指定します。

もし、既存のテンプレートの編集ではプロジェクトの要件を満たせないという場合、画面右端の新規作成アイコンをクリックし、「ウィザードテンプレート」をクリックすることで追加できます。追加する際には、テンプレートの名前等の指定が必要となりますが、魅力的な機能として、既存のテンプレートをベースに新規作成できる点です。

テンプレート: ウィザードテンプレート

ステップテンプレート タブ



ステップテンプレート タブにフォーカスしています。

このタブで定義したテンプレートは、ウィザードテンプレートが適用されたオブジェクトにおいて [step] ノードとして追加することができます。

このステップでは、前または次のステップに移動できる必要があるため、必ず [Standard Action (WizardPrevious)] と [Standard Action (WizardNext)] が含まれます。

また、既定で定義されたステップテンプレートは次の用途を想定しています。
(説明は実際に利用する順番に近い形で紹介しています)

[Welcome] ステップテンプレート :

ウィザードの紹介に使用できます。ウィザードでこのステップテンプレートを
使用するときは、ウィザードの目的の説明を入力してテキストブロックを変更
する必要があります。

[Data] ステップテンプレート :

情報を表示したり、ユーザーに情報の入力を求めたりするのに使用できます。

[Select Grid Items] ステップテンプレート :

グリッドにデータを表示し、ユーザーがグリッド内のデータを選択する必要が
ある場合に使用できます。

[Add Grid Items] ステップテンプレート :

ユーザーが1つのステップで複数のデータを1つずつ追加し、追加した内容すべ
てを表示する必要がある場合に使用できます。

[Summary] ステップテンプレート :

ここまでのステップでユーザーが入力したすべてのデータの要約を表示します。
ここでも各データの編集が可能です。

[Completed] ステップテンプレート :

ウィザードの最後のステップとして使用し、プロセスや関連アクション
などに関する情報を含めることができます。

テンプレート: レポートテンプレート



レポートテンプレートタブでは、PDF を出力するためのプロシージャオブジェクトに適用可能なテンプレートを定義します。
あらかじめ定義されているテンプレートは、「ReportEmpty」という空のテンプレートと、請求書を想定した3つのテンプレートです。

もし、既存のテンプレートの編集ではプロジェクトの要件を満たせないという場合、画面右端の新規作成アイコンをクリックし、「レポートテンプレート」をクリックすることで追加できます。
追加する際には、テンプレートの名前等の指定が必要となりますが、魅力的な機能として、既存のテンプレートをベースに新規作成できる点です。

テンプレート: 共通のイベント、サブルーチン

Events & Subs
Event ()
Sub ()

すべてのテンプレートには、[Events & Subs] ノードが含まれています。（※）この [Events & Subs] ノードに子ノードとして [eventBlock] ノードを追加することで、テンプレートを適用して生成されるオブジェクトに共通のイベント（またはサブルーチン）を追加することができます。

スライドにはイベントを追加する場合のプロパティウィンドウが表示されていますが、トランザクションテンプレートに追加された場合となり、他のテンプレートの場合、表示されないものもあります。

次のプロパティを設定していくことで、追加するコードの定義を行います。

Block Name :

定義するイベント（またはサブルーチン）の名前を指定します。この名前はテンプレートの定義内で一意の名前である必要があります。

Include : (トランザクションテンプレートのみ)

このノードにより一度だけコードを追加する（値が [Once]）か、条件に一致した項目属性ごとにコードを追加する（[Once per matching attribute]）か指定できます。

[Once per matching attribute] の場合、[Include when att matches] プロパティ群が追加されます。

このプロパティ群は、自動ルールにて表示されるものと同一です。

Wrapping Tag : (トランザクションテンプレートの Transaction タブのみ)

追加するコードに修飾子を付与するかどうかを指定します。

既定値の [] の場合には、修飾子は追加されず、[[web]] の場合、トランザクションのインターフェースを利用する場合のみのコードとなり、[[bc]] の場合、ビジネスコンポーネントとして利用する場合のみのコードとなります。

Behavior if Block is not present on instance :

実際のパターンインスタンスに [Events & Subs] ノードが含まれていない場合の挙動を指定します。

既定では [Include] となるため、コードを追加する場合に [Events & Subs] ノードがない場合、自動で追加されます。

Add To Instance By Default :

実際のパターンインスタンスに [Events & Subs] ノードを含めるかを指定します。

既定値は [False] となり、自動では含まれません。要件に合わせて [True] に切り替える必要があります。

Type :

コードを「イベント」として追加するか、「サブルーチン」として追加するかを指定します。

このプロパティの値に基づき、実際に表示されるノードが Event と表示されるか、Sub と表示されるかが決まります。

既定値は [Event] となり、サブルーチンとして追加する場合、[Sub] に変更する必要があります。

Event or Sub Name :

コードを追加するイベント（またはサブルーチン）名を指定します。

イベントの場合、テンプレートが適用されるオブジェクトタイプで定義可能なサーバーサイドイベントは選択肢として表示されます。

ユーザーイベントの場合には、このプロパティで任意の名前を指定します。

Code :

実際に生成されるコードを記載します。

また、記載には WorkWithPlus for Web によって自動置換されるタグ

（例：<ATT_NAME>、<ATT_DESCRIPTION>）を利用可能です。

記入例：Form.Caption += "(" + &Mode + ")"

Position :

コードを追加する位置を指定します。

[Event or Sub Name] プロパティで指定したイベント（またはサブルーチン）がすでに定義されている場合、その中でコードを最下部に追加する場合、[Bottom] を選択し、最上部に追加する場合、[Top] を選択します。

※ レポートテンプレートのみ [Subs] ノードとなり、追加できる子ノードは [Sub] ノードのみです。これはレポートテンプレートを適用するプロシージャオブジェクトに [Events] エLEMENTがないためです。

テンプレート: 画面に配置しない変数

& Variables (Not included in Web Form)

& <variable>

プロパティ	
variable: <variable>	
Name	
Type Definition	
Data Type	Based on
Domain, Attribute or SDT (none)	

プロパティ	
variable: <variable>	
Name	
Type Definition	
Data Type	Basic
Type	Numeric
Length	4
Decimals	0

[Events & Subs] ノードで自動生成するコード内で画面に表示されない変数を利用したい場合、[Variables (Not included in Web Form)] ノードに [Variable] ノードを追加し、定義することができます。

[Variable] ノードは、プロパティを指定し、変数名やデータタイプを指定できます。

List にて追加することができた変数と同様にナレッジベース内の項目属性や SDT 等をベースにした変数か、基本のデータタイプに基づく変数を追加できます。

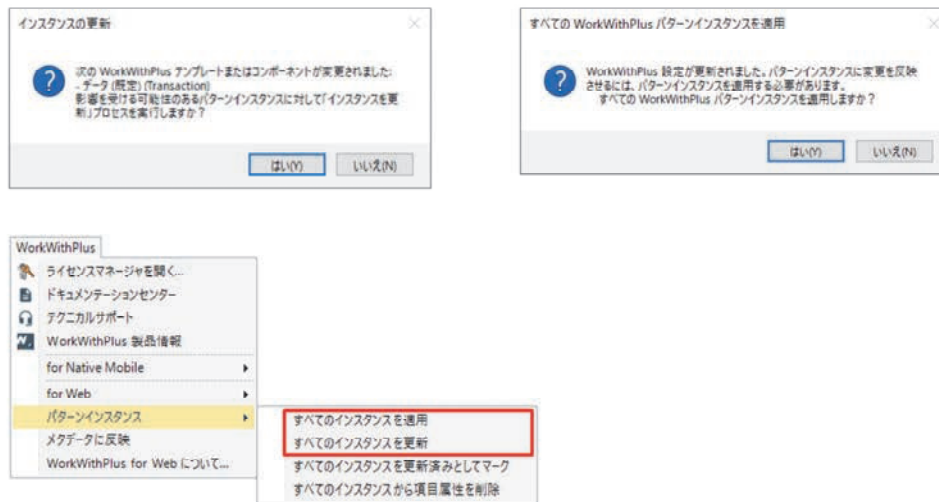


WORKWITHPLUS
FOR WEB

3. WorkWithPlus for Web の設定 3.3. インスタンスの更新 / 適用



インスタンスの更新 / 適用



「WorkWithPlus 設定」において変更を行った場合、保存時にナレッジベース内全体へ変更内容の更新または適用が行われます。

「更新」が行われるか、「適用」が行われるかは、実施した変更に基づきます。

例えば、Transaction テンプレートの変更を行った場合は、

「インスタンスの更新」が行われ、全体のプロパティ タブの変更を行った場合は、「すべての WorkWithPlus for Web パターンインスタンスの適用」が行われます。

この「インスタンスの更新」という処理と、「パターンインスタンスの適用」（「すべての WorkWithPlus パターンインスタンスを適用」）という処理の違いについて説明します。

「インスタンスの更新」

各オブジェクトの「パターンインスタンス」の階層構造に対し、外的要因（例：テンプレートに対する変更）に基づく構造の修正を行います。

「WorkWithPlus 設定」にて行われた変更に基づく対象範囲が決定し、すべての対象に対し更新が行われます。

特定のオブジェクトで実施するケースについては、既に Transaction の説明において項目属性を Structure に追加する場合に利用する機能としてご紹介しました。

「パターンインスタンスを適用」

各オブジェクトに適用している WorkWithPlus for Web のパターンインスタンスに基づきオブジェクトを再生成します。

「WorkWithPlus 設定」にて 全体のプロパティ タブでプロパティの変更のみが行われた場合に行われます。

これは、[保存時にこのパターンの適用] のチェックマークを外し、再度チェックし、オブジェクトを保存する場合や、各オブジェクトの

「パターンインスタンス」において階層構造に変更を加え、保存する場合と同じです。

この処理が「すべての WorkWithPlus パターンインスタンスを適用」という記載の場合には、ナレッジベース内でパターンが適用されているすべてのオブジェクトを対象とします。

インスタンスの更新および適用については、任意のタイミングでナレッジベース全体へ実施することができます。

この操作はメニューバーの WorkWithPlus →パターンインスタンスより選択可能です。

「すべてのインスタンスを適用」については、上記

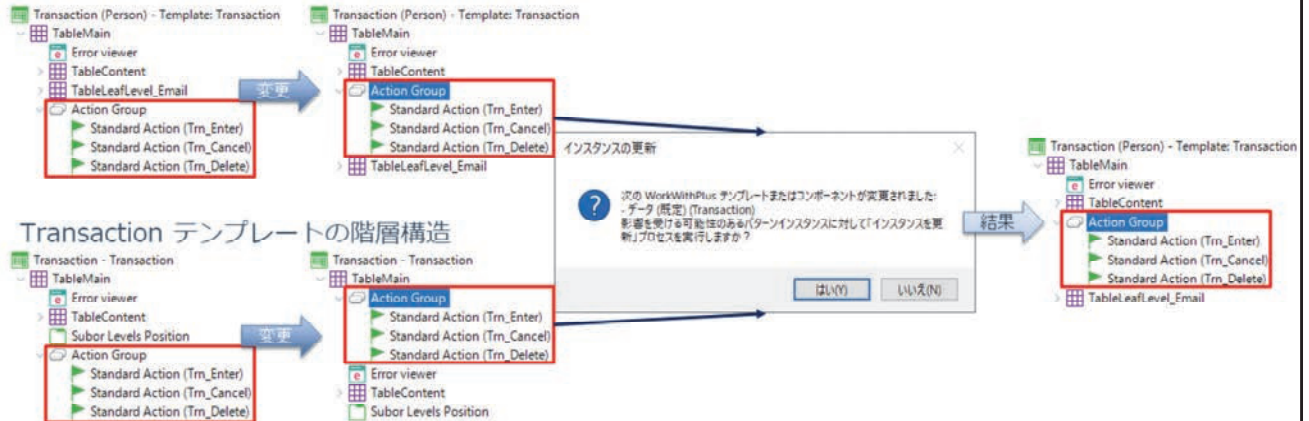
「すべての WorkWithPlus パターンインスタンスを適用」と同様です。

「すべてのインスタンスを更新」については、実施内容は上記の通り階層構造の修正となりますが、前述の Structure に項目属性を追加した場合、このトランザクションオブジェクトを View のタブとして含んでいるオブジェクトに対しても追加することができます。

インスタンスの更新 / 適用

「知的な」インスタンスの更新

Person トランザクションの階層構造



WorkWithPlus for Web によるインスタンスの更新は「知的な」更新を行います。これは、「WorkWithPlus 設定」にて行われた変更と、実際のオブジェクトにて行われていた変更が競合した場合、実際のオブジェクトにおける変更が優先されるものとなります。

つまり、独自にカスタマイズが行われているオブジェクトはその変更が維持できます。



WORKWITHPLUS
FOR WEB

4. Web パネル



Web パネル: 適用可能なテンプレート



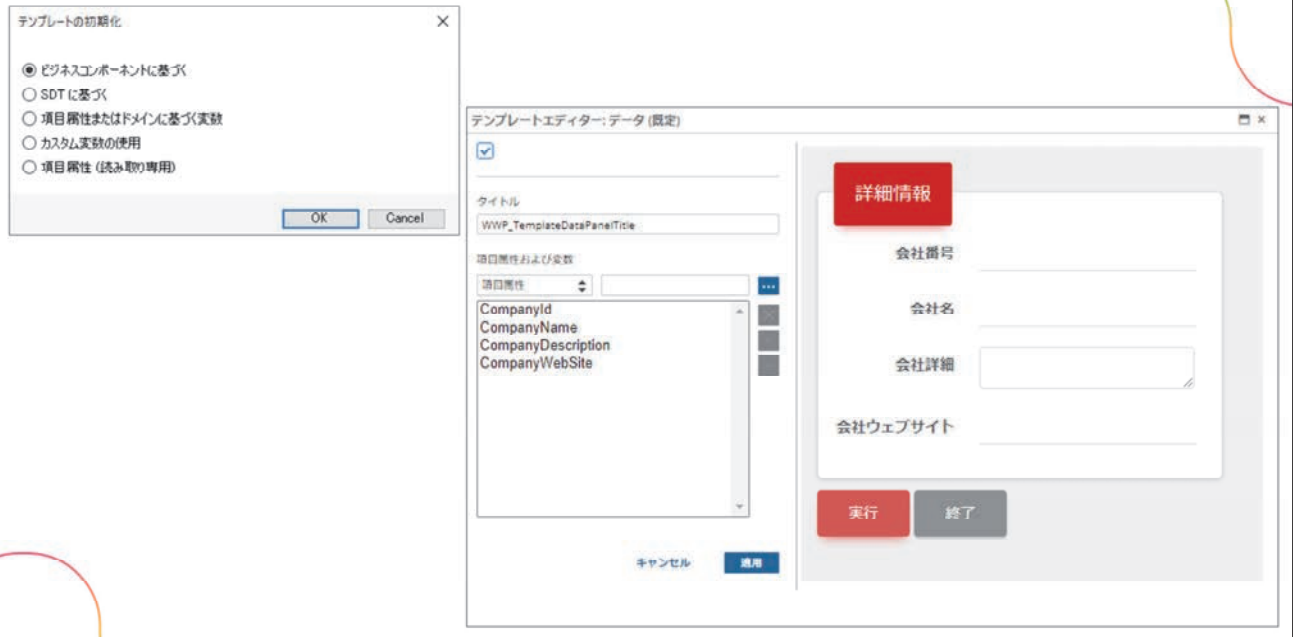
Web パネルタイプのオブジェクトを作成し、Patterns エlementを表示すると、トランザクションオブジェクトと同様に WorkWithPlus for Web パターンを適用するためにテンプレートを選択する画面が表示されます。

「WorkWithPlus 設定」の説明の中で触れていましたが、適用可能なテンプレートは、「WebPanel テンプレート」として定義されているものの他に、「Transaction テンプレート」として定義されている Transaction や Selection を適用することや、「ウィザードテンプレート」を適用することができます。

そのため、トランザクションオブジェクトにパターンを適用した際に生成される画面を単独の Web パネルオブジェクトとして構成することができます。

また、画面としても項目属性ベースではなく、変数ベースで構成することもできるため、より自由度の高い画面を生成することができます。

Web パネル: データの編集・作成・表示



「テンプレートセレクトター」ダイアログにて「データの編集・作成・表示」カテゴリを選択した場合、トランザクションオブジェクトに適用される Transaction テンプレートがベースとなります。

そのため、画面をどのように構成するかを「テンプレートの初期化」ダイアログにて指定します。

このダイアログで選択できるオプションは以下の通りです。

- ・ ビジネスコンポーネントに基づく

次のダイアログにてナレッジベース内のビジネスコンポーネントが有効に

なったトランザクションオブジェクトを選択します。

その後、さらにトランザクションオブジェクトで定義されている項目属性

の中から生成される画面に表示する項目属性を選択します。

- ・ SDT に基づく

次のダイアログにてナレッジベース内の SDT オブジェクトを選択します。その後、さらに SDT オブジェクトで定義されているメンバーの中から生成

される画面に表示するメンバーを選択します。

- ・ 項目属性またはドメインに基づく変数

次のダイアログにてナレッジベース内に定義されている項目属性またはドメインを複数選択します。

選択した項目属性またはドメインをデータタイプとする変数が生成される

画面に表示されます。

- ・ カスタム変数の使用

ナレッジベース内の定義からの選択ダイアログは表示されず、下記で説明するテンプレートエディターが表示されます。

- ・ 項目属性（読み取り専用）

次のダイアログにてナレッジベース内に定義されている項目属性を選択します。

選択した項目属性が画面に配置され、読み取り専用画面として生成が行われます。

「テンプレートの初期化」における各選択肢に基づく操作ののち、「テンプレートエディター」というダイアログが表示され、画面構成のための最後の調整として、選択漏れの追加、誤って含めてしまったものの削除、表示順の並び替えが行えます。

適用ボタンをクリックすることでこの設定に基づきパターンが適用されますので、オブジェクトを保存することで最終的に階層構造に沿った Web Layout

エレメントや Rules エレメント、Events エレメントの内容が自動生成されます。

Web パネル: データの編集・作成・表示

「ビジネスコンポーネントに基づく」場合の追加機能

Based on template: Transaction

プロパティ	
フィルタ	
WPRoot: Based on template: Transaction	
Master Page/Panel	<default>
Form Theme Class	
Show Master Page When Pop Up	<Do not update>
Save Behaviour	
Generate Save Behaviour	True
Business Component	Company
Default Perm	Mode & Key
Modes	Display, Insert, Update, Delete
Modes Security	Transaction permissions
Support Develop Combo Insertion	False

Rules

```
/* Generated by Develop work with Plus Pattern [Start] : Do not change */
pare(incNode, incCompanyId);
/* Generated by Develop work with Plus Pattern [End] : Do not change */
```

Events

```
Event Start
/* Generated by Develop work with Plus Pattern [Start] : Do not change */
/* Generated by Develop work with Plus Pattern [Start] : Do not change */
if ($trNode <> $trNode.Display)
if ($trNode <> $trNode.Delete)
if ($trNode <> $trNode.Update)
if ($trNode <> $trNode.Delete)
if ($trNode <> $trNode.Insert)
    $company.Load($companyId)
    $loadSuccess = $company.Success()
    if not $loadSuccess
        $messages = $company.GetMessages()
        Do "Show Messages"
    endif
    if ($trNode <> $trNode.Display OR $trNode <> $trNode.Delete)
        $company.CompanyName.Enabled = False
        $company.CompanyDescription.Enabled = False
        $company.CompanyWebsite.Enabled = False
    endif
endif
else
    $loadSuccess = False
    Return
endif
if $loadSuccess
    if ($trNode <> $trNode.Delete)
        msg("OK_delete")
    endif
endif
/* Generated by Develop work with Plus Pattern [End] : Do not change */
```

```
Event Enter
/* Generated by Develop work with Plus Pattern [Start] : Do not change */
if ($trNode <> $trNode.Display)
if ($trNode <> $trNode.Delete)
    Do "CheckRequiredFields"
endif
if ($trNode <> $trNode.Delete OR $CheckRequiredFieldsResult)
    $company.Delete()
else
    $company.Save()
endif
if $company.Success()
    Do "after_trn"
else
    $messages = $company.GetMessages()
    Do "Show Messages"
endif
endif
/* Generated by Develop work with Plus Pattern [End] : Do not change */
EndEvent
```

Web パネルオブジェクトに「データの編集・作成・表示」カテゴリから適用するテンプレートを選択し、ビジネスコンポーネントに基づく実装とした

場合、プロパティを利用し、Rules や Events のコードを自動生成することができます。

階層構造の一番親となるノードが持つプロパティのうち、[Generate Save Behaviour] プロパティが [True] になっていることが確認できます。

この場合、後続のプロパティが表示され、ビジネスコンポーネントに基づく Web パネルにおけるデータの操作処理が自動生成されるようになります。

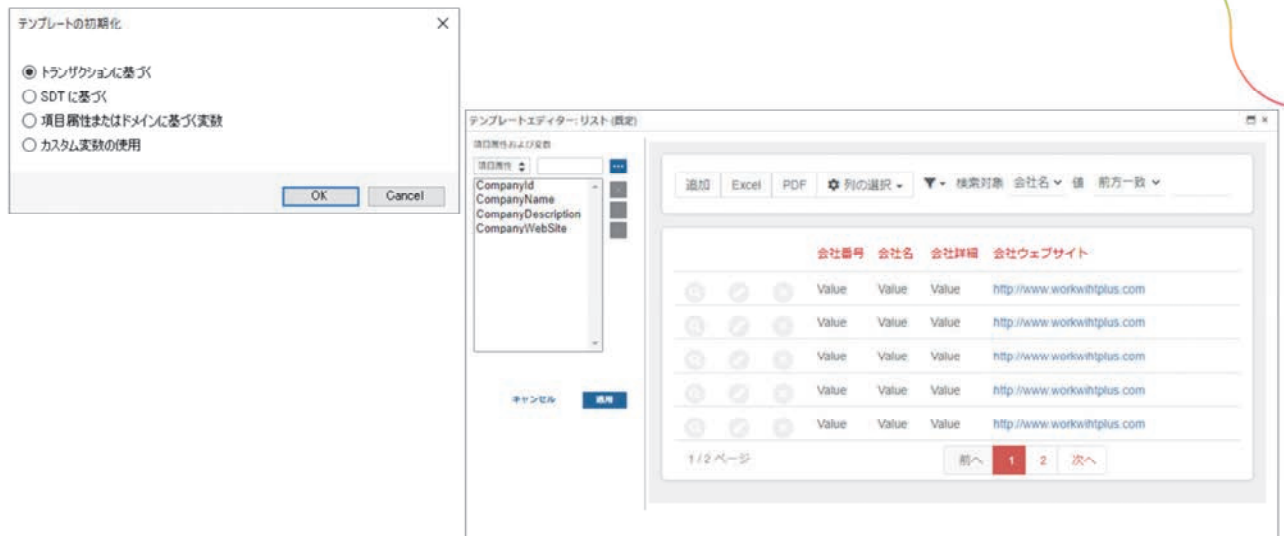
各プロパティの値をさらに変更し、自動生成されるコードをカスタマイズしたり、自動生成されたコードに対し、開発者がカスタマイズを加えることもできます。

ただし、カスタマイズを加える場合には、自動生成されたコードの最初か最後に追加する形となります。

もし、この自動生成されたコードでは、要件を満たせない場合、[Generate Save Behaviour] プロパティを [False] に設定し、登録処理を手動で

実装する必要があります。

Web パネル: List



「テンプレートセレクトター」ダイアログにて「List」カテゴリを選択した場合、トランザクションオブジェクトに適用される Selection テンプレートがベースとなります。

そのため、画面をどのように構成するかを「テンプレートの初期化」ダイアログにて指定します。

このダイアログで選択できるオプションは以下の通りです。

- ・ トランザクションに基づく

次のダイアログにてナレッジベース内のトランザクションオブジェクトを

選択します。

選択したトランザクションに含まれるすべての項目属性が画面に表示されます。

- ・ SDT に基づく

次のダイアログにてナレッジベース内の SDT オブジェクトを選択します。

その後、さらに SDT オブジェクトで定義されているメンバーの中から生成される画面に表示するメンバーを選択します。

- ・ 項目属性またはドメインに基づく変数

次のダイアログにてナレッジベース内に定義されている項目属性またはドメインを複数選択します。

選択した項目属性またはドメインをデータタイプとする変数が生成される

画面に表示されます。

- ・ カスタム変数の使用

ナレッジベース内の定義からの選択ダイアログは表示されず、下記で説明するテンプレートエディターが表示されます。

「テンプレートの初期化」における各選択肢に基づく操作ののち、「テンプレートエディター」というダイアログが表示され、画面構成のための最後の調整として、選択漏れの追加、誤って含めてしまったものの削除、表示順の並び替えが行えます。

適用ボタンをクリックすることでこの設定に基づきパターンが適用されますので、オブジェクトを保存することで最終的に階層構造に沿った Web Layout

エレメントが自動生成されます。

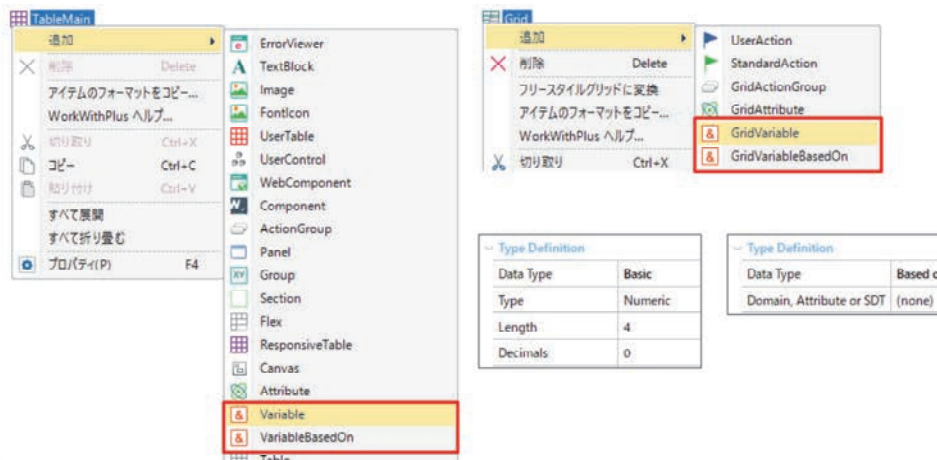
「テンプレートの初期化」において「トランザクションに基づく」を選択した場合は、データの読み込みや各アクションに対する挙動が自動生成されます。

これは、トランザクションオブジェクトに適用された際に自動生成される List と同様となります。

それ以外の選択肢の場合、これらのコードは自動生成されないため、開発者自身が Rules エlement や Events Element へコードを追記する必要があります。

Web パネル: 変数

変数の追加



Web パネルに WorkWithPlus for Web パターンを適用した場合も画面には変数を配置することができます。

変数の追加については、List にて詳細を説明した通りとなります。

レベルを持つビジネスコンポーネントや SDT に基づく変数を追加する場合、第 1 レベル (= コレクションではない) に属する項目属性 / メンバーは [テーブル] タイプのノードに追加することができますが、第 2 レベル (= コレクション) に属する項目属性 / メンバーは [グリッド] タイプのノードに追加する必要があります。



WORKWITHPLUS
FOR WEB

5. PDF レポート



PDF レポート: 適用可能なテンプレート



プロシージャータイプのオブジェクトを作成し、Patterns エlementを表示

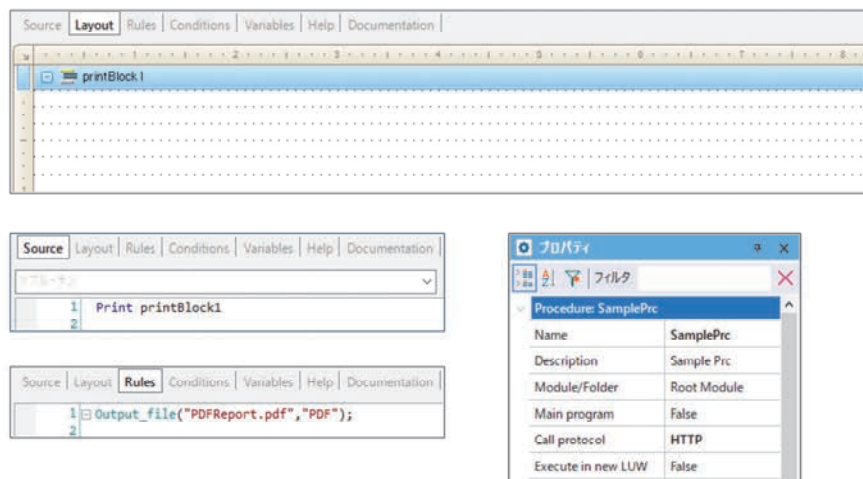
すると、トランザクションオブジェクトと同様に WorkWithPlus for Web パターンを適用するためにテンプレートを選択する画面が表示されます。

「WorkWithPlus 設定」の説明の中で触れていましたが、適用可能なテンプレートは、「レポートテンプレート」として定義されていました。

既定では「請求書」の作成を想定したテンプレートが3種類と空白のもののみとなります。

本章で説明する内容に基づいてテンプレートのカスタマイズも行えるため、プロジェクトの要件にあわせてテンプレートを追加できます。

PDF レポート: GeneXusにおける PDF レポート



はじめに、GeneXus における PDF レポート機能の作成についておさらいします。

PDF を出力するためには、次の 4 点を必ず行う必要があります。

1. Layout エlement での出力レイアウトの定義
2. Source エlement での Print コマンドを利用したプリントブロックの出力
3. Rules エlement での Output_file ルールの定義
4. プロシージャオブジェクトの [Call protocol] プロパティを [HTTP] に設定

また、Layout エlement では、「プリントブロック」という出力単位ごとにレイアウトをデザインしました。

この点を踏まえ、WorkWithPlus for Web パターン適用による PDF レポートの作成を見てみましょう。

PDF レポート: 階層構造



プロシージャオブジェクトへ WorkWithPlus for Web を適用した場合、出力される PDF のレイアウトを決定する Layout エLEMENT の定義を階層構造で行えるようになります。

階層構造の一番親となるノードは、[Based on template: <テンプレート名>] というノードとなります。（以降 [Based on template] ノードと記載）このノードのプロパティでは、Layout エLEMENT 選択時に指定できるプロパティのうち、一部を指定できます。その他のプロパティについては、このノード以下の各ノードで設定が設けられています。また、[Based on template] ノードに追加できるノードのうち、注目すべきノードが [PrintBlock] ノードと [GridPrintBlock] ノードです。

前述の通りプロシージャークオブジェクトの Layout エlement は「プリントブロック」という出力単位でレイアウトをデザインするため、階層構造でもこの単位で定義を行っていきます。この「プリントブロック」を定義するために用意されているのが前述の [PrintBlock] ノードと [GridPrintBlock] ノードです。

[PrintBlock] ノード :

GeneXus の機能のみで PDF 出力を行うために追加する「プリントブロック」を定義するためのノードです。

つまり、このノードにより追加される「プリントブロック」は単一です。

[GridPrintBlock] ノード :

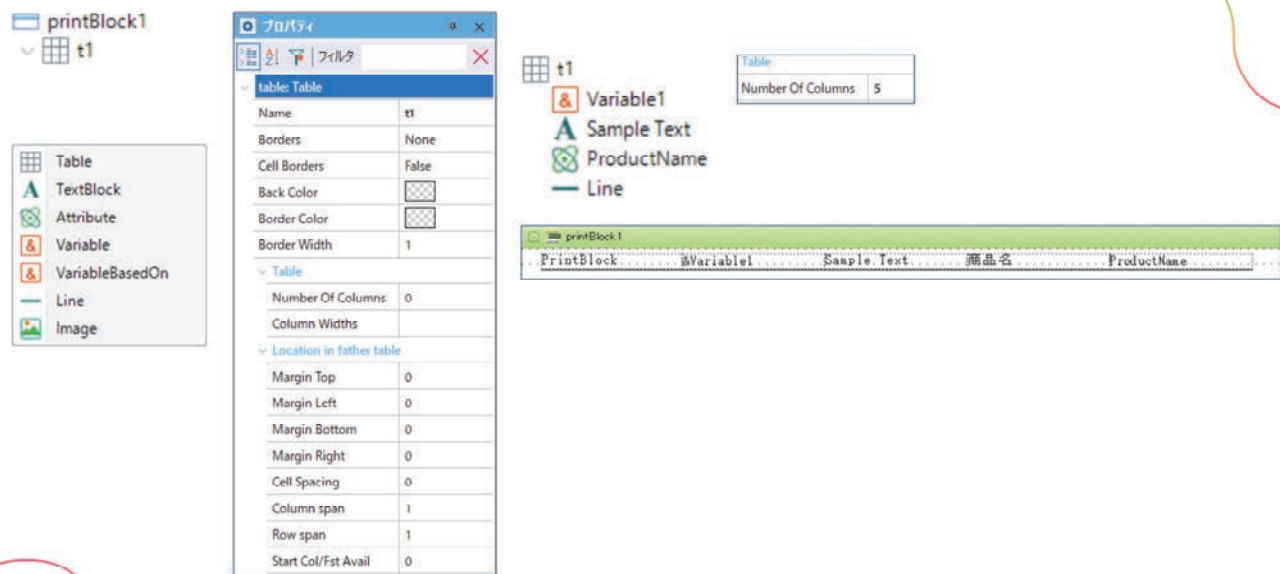
「プリントブロック」を追加するノードとなりますが、複数の「プリントブロック」を追加します。

これは、複数のレコードを繰り返し出力することを想定したノードとなるためです。

例えば既定では、タイトルを表示する「プリントブロック」とデータを表示する「プリントブロック」、横線を表示する「プリントブロック」の 3 つが追加されます。

もちろん複数回出力するレイアウトをデザインする際に [PrintBlock] ノードを利用することも可能です。

PDF レポート: [PrintBlock] ノード



[PrintBlock] ノードを追加した場合、必ず [Table] ノードが含まれます。この [Table] ノードに対し必要に応じて子ノードを追加していきます。追加することができるノードは、プロシージャオブジェクトの Layout エレメントで追加することができるコントロールすべてです。

ただ、Layout エレメントで追加することができるコントロールに

「Table」

というコントロールはありません。

これは、WorkWithPlus for Web によってアプリケーションに必要な見栄えの

良い PDF をデザインするために、階層構造のみで利用できるものとなります。

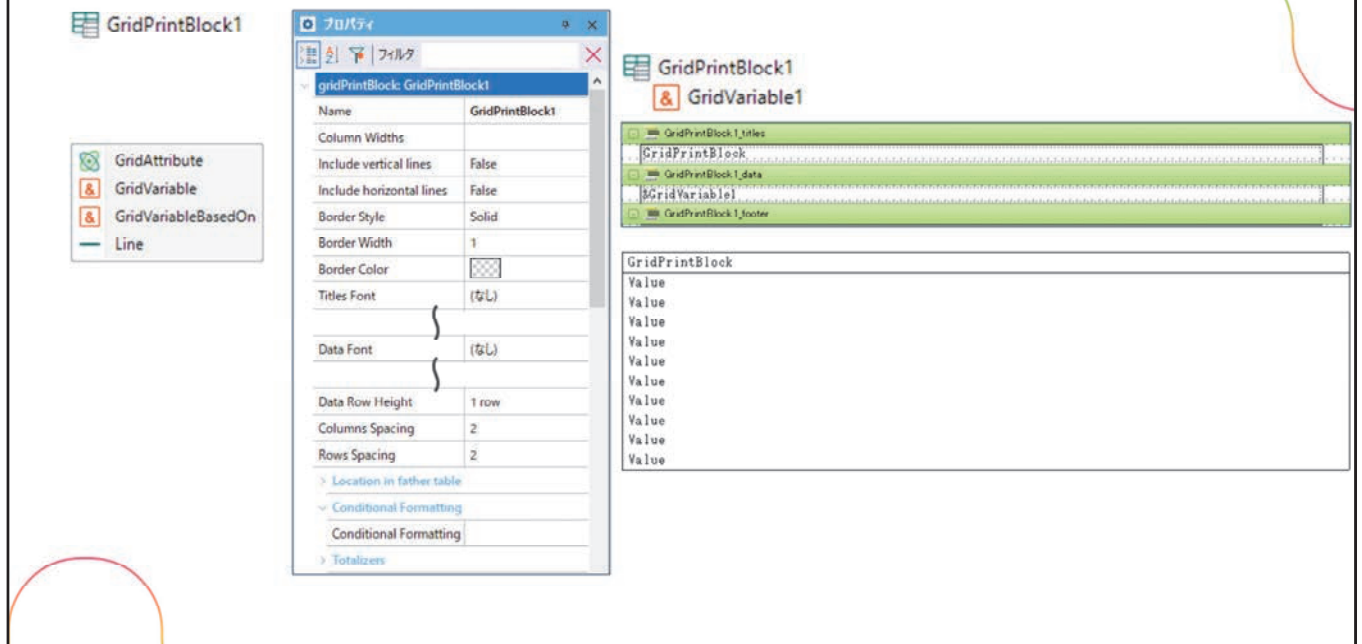
そのため、実際に生成される「プリントブロック」にテーブルは存在していません。

[Table] ノードを利用することで、ここまでのオブジェクトでも出てきたように [Number Of Columns] プロパティがあり、1 行にいくつのコントロールを配置するか指定することができます。

(ただし、他オブジェクトと異なり、項目属性や変数のデスクリプションも

1 列として数えられます)

PDF レポート: [GridPrintBlock] ノード



[GridPrintBlock] ノードを追加した場合、[PrintBlock] ノードのように [Table] ノードは含まれません。

これは、[GridPrintBlock] ノードの場合、繰り返し表示したい項目属性または

変数のみを追加することが想定されているためです。

また、[GridPrintBlock] ノードのプロパティを設定し、罫線の追加や、タイトル、データの表示領域に関する見栄えを設定することができます。

このプロパティの設定に依存しますが、[GridPrintBlock] ノードによってプリントブロックは 1 つ以上作成が行われます。

例えば、[GridPrintBlock] ノードのプロパティは既定値のまま変数を 1 つだけ

含めた場合、3 つのプリントブロックが追加されます。

PDF レポート: 出力の実装

```
Source | Source (without WorkWithPlus code) | Layout | Rules | Conditions | Variables | Help | Documentation | Patterns |
マイメニュー
1  @Variable1 = "Sample"
2  Print printBlock1
3
4  Print GridPrintBlock1_titles
5  For @Num = 1 To 5
6      @GridVariable1 = @Num.ToString()
7      Print GridPrintBlock1_data
8  EndFor
9  Print GridPrintBlock1_footer
10
```

プロパティ	
フィルタ	
▼ Procedure: SamplePrc	
Name	SamplePrc
Description	Sample Prc
Module/Folder	Root Module
Main program	False
Call protocol	HTTP
Execute in new LUW	False

```
Source | Source (without WorkWithPlus code) | Layout | Rules | Conditions | Variables | Help | Documentation | Patterns |
1 Output_file("PDFReport.pdf", "PDF");
2
```

前述の通りあくまでも WorkWithPlus for Web によって生成される範囲は Layout エlementのみとなります。

そのため、PDF を出力するための Source Element の記述、Rules の記述、プロパティの設定は手動で行う必要があります。



WORKWITHPLUS
FOR WEB

6. セキュリティ



セキュリティ

GeneXus™
Access Manager



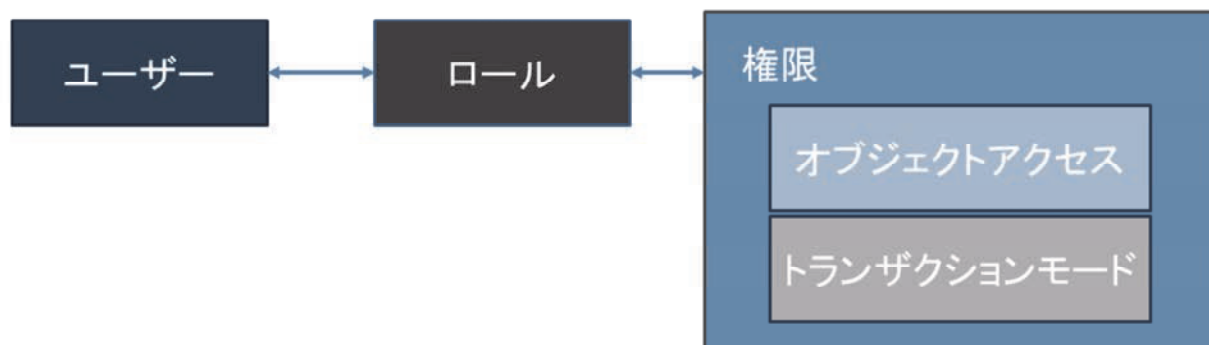
WorkWithPlus for Web を利用し、セキュリティの機能を実装する場合、いくつかのタイプを選択することができます。

その中で現在 WorkWithPlus for Web として利用を推奨しているタイプが GeneXus のビルドインセキュリティモジュールである GeneXus Access Manager (GAM) と統合したものとなります。

これにより GAM が持つ機能を再利用し、WorkWithPlus for Web による更なる追加のセキュリティオプションを利用することができます。

どのような機能が利用できるか代表的な機能について本章でご紹介します。

セキュリティ: GAM に基づく機能



GAM では、ロールベースアクセス制御（RBAC）を実装します。

そのため、ログインするためのユーザーがあり、ユーザーには複数のロールが紐づけられています。

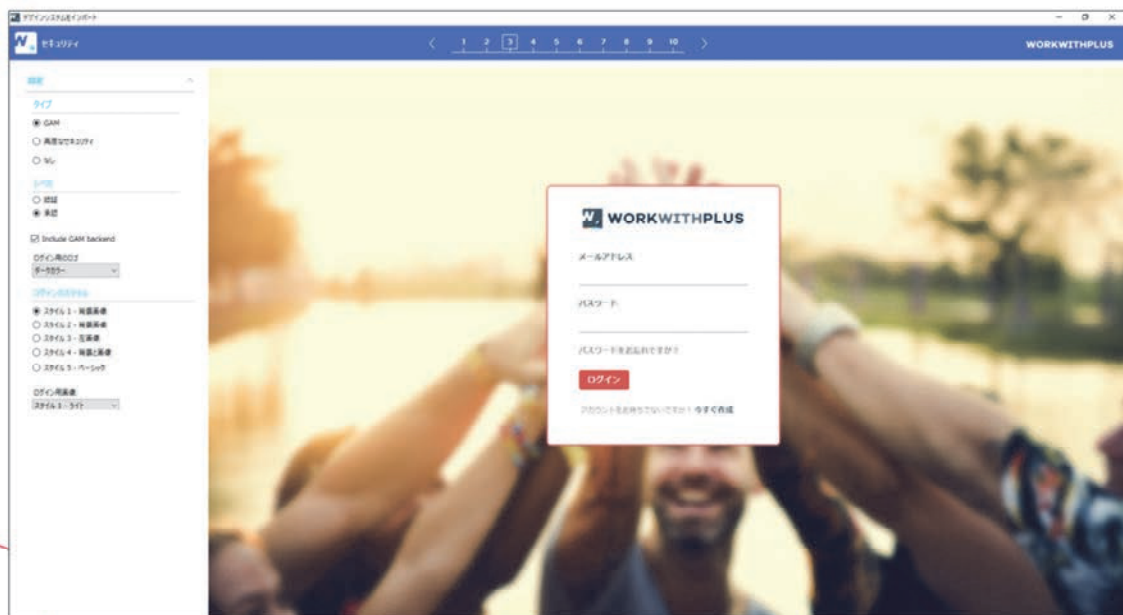
ロールには複数の権限が紐づけられています。

「権限」としてあらかじめ選択できるものはナレッジベース内のオブジェクトへの「アクセス権限」です。

また、オブジェクトが「トランザクション」の場合には、登録や更新、削除といった「モード」も「権限」として選択できます。

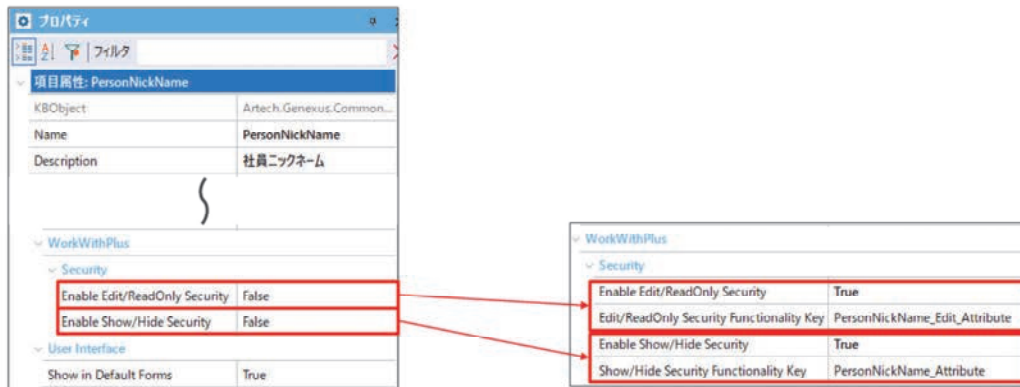
この点においては WorkWithPlus for Web による機能拡張はありませんので、詳しい機能については、GeneXus の GAM をご確認ください。

セキュリティ: ログイン画面のデザイン



既に説明の通り、デザインシステムウィザードでセキュリティを有効にするステップ3で、ログイン画面のデザインを選択することができます。

セキュリティ: 項目属性の権限



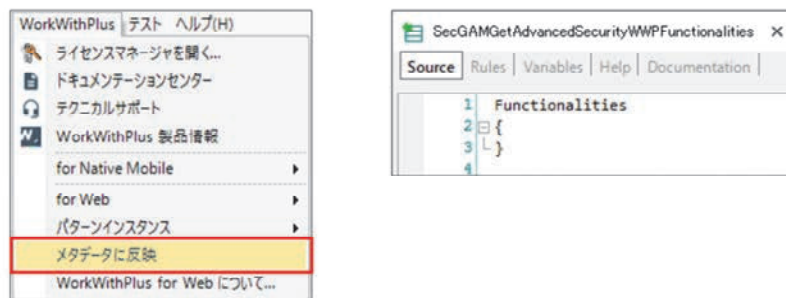
WorkWithPlus for Web によるセキュリティ機能として、項目属性の「編集 / 読取専用」、「表示 / 非表示」を権限としてロールに紐づけることが可能です。

この権限を追加するためには、トランザクションオブジェクトの Structure エlement で、対象とする項目属性のプロパティを表示します。

項目属性に対して「編集 / 読取専用」の権限を設定したい場合、[Enable Edit/ReadOnly Security] プロパティを [True] に設定します。この時、[Edit/ReadOnly Security Functionality Key] プロパティが新たに追加されます。このプロパティで、ロールに紐づける権限の名前を指定することができます。

項目属性に対して「表示 / 非表示」の権限を設定したい場合、[Enable Show/Hide Security] プロパティを [True] に設定します。この時、[Show/Hide Security Functionality Key] プロパティが新たに追加されます。このプロパティで、ロールに紐づける権限の名前を指定することができます。

セキュリティ: メタデータに反映



項目属性の権限を利用する場合、前述のプロパティ設定に加え、「メタデータに反映」という操作が必要です。

この操作を行うことで、ナレッジベース内に作成されている SecGAMGetAdvancedSecurityWWPFunctionalities データプロバイダーに自動的に項目属性の権限を登録するためのデータが生成されます。その後、このデータを登録する処理が実行されます。この結果、アプリケーションを実行した際にロールに紐づけられる権限にこれらが追加されます。

セキュリティ: パネル、タブ、アクションの表示



WorkWithPlus for Web では、設定された権限に基づき表示 / 非表示をコントロールする機能が搭載されています。

例えば、表示することができる項目属性が 1 つもない「パネル」や「タブ」は自動的に非表示となります。

権限が付与されていないオブジェクトの呼び出しや、権限が付与されていないトランザクションモードの呼び出しが紐づけられたアクションがある場合、これも自動的に非表示となります。

セキュリティ: メニューの制御



パネルやタブ、アクションの非表示だけでなく、既定で生成されるメニューについてもセキュリティを考慮した表示を行う機能が有効になります。そのため、ユーザーは権限のある機能のみにアクセスすることができます。

もちろん、権限のない画面に無理やりアクセスした場合、権限がないことを示す画面が表示されます。

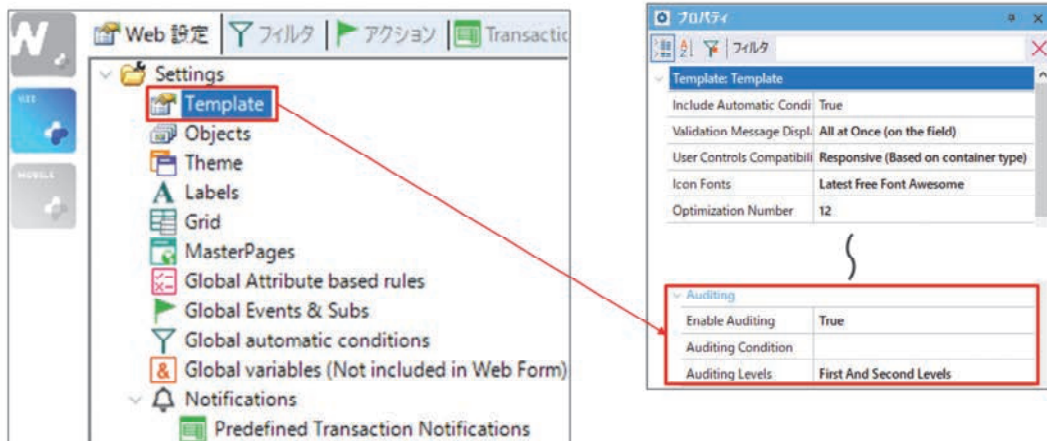


WORKWITHPLUS
FOR WEB

7. 監査



監査: 機能の有効化



WorkWithPlus for Web では、監査用に利用できるデータを残すための機能が用意されています。

この機能を利用することで、トランザクションオブジェクトを介して行われたデータの登録、更新、削除を記録することができます。

この機能を有効にするためには、「WorkWithPlus 設定」内の Web 設定 タブの [Template] ノードが持つ [Enable Auditing] プロパティを [True] に変更します。

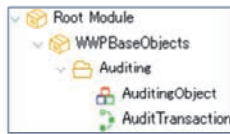
「WorkWithPlus 設定」に対する変更となるため、変更を保存する際には、「すべての WorkWithPlus パターンインスタンスを適用」を行う必要がある

旨ダイアログが表示されますので、適用を実行します。

ただし、これだけでは監査用データの記録は行われません。

続けてプロシーチャーオブジェクトの定義を行う必要があります。

監査: AuditTransaction プロシージャ



```
Source | Source (without WorkWithPlus code) | Layout | Rules | Conditions | Variables | Help | Documentation | Patterns |
▼
1
2
3 //This is a procedure in which you has to define the behaviour of auditing after a record has
4 //being inserted, modified or deleted (when the transaction has auditing enabled).
5 //There is an example below of a way to audit the actions of record, saving them in DataBase
6 //(but this can be modified to do whatever you want):
7
8 //You should create a transaction with the following structure:
9
10 //Transaction:: Audit
11 //      * AuditId          Id (Numeric) - Autounumber
12 //      AuditDate          DateTime
13 //      AuditTableName     Name (Varchar(40))
14 //      AuditDescription   LongVarChar(20)
15 //      AuditShortDescription Varchar(400)
16 //      SecUserId          -> Only when using Advanced Security
17 //      SecUserName        -> Only when using Advanced Security
18 //      GWMUserGUID        GWMGUID -> only when using GWM
19 //      AuditAction        Varchar(10)
20
21 //Set this transaction as bussiness component
22
23 //Then uncomment the code below and define the variables:
24 //
25 //      &Audit:: Audit (Business Component)
26 //      &AuditDescription:: AuditDescription
27 //      &AuditShortDescription:: AuditShortDescription
28 //      &AuditPrimaryKey:: AuditShortDescription
29 //      &WAPContext:: WAPContext
30 //      &FirstRecord:: Boolean
31 //      &ActualMode:: Character(3)
```

先ほど [Enable Auditing] プロパティを [True] に設定しましたが、この変更により WorkWithPlus for Web が適用されたトランザクションオブジェクトでは、AuditTransaction プロシージャを呼び出す処理が追加されます。このオブジェクトはデザイン システム ウィザードにて「デザインシステムをインポート」を実行することで取り込まれたオブジェクトの 1 つです。

このプロシージャオブジェクトの Source エlement に記載された内容に沿ってトランザクションオブジェクトの定義と変数の追加、説明行以降にある実際のコードのコメント解除を実行することで、監査用データの記録が実行されるようになります。

監査: 記録されるデータ

監査テーブル名 Person

監査詳細 Record with key '社員番号 = 9 - 社員名称 = 監査 テスト' was inserted.
Attributes:
社員名 = テスト
社員姓 = 監査
社員名称 = 監査 テスト
社員性別 = 1
社員生年月日 = 23/02/17
社員お気に入り/バンド有無 = false
社員住所必須 = false
会社番号 = 4
会社名 = Disco
会社管理職 = false
社員登録日 = 23/02/17 17:42
社員登録者 = testuser
社員更新日 = // 00:00

監査テーブル名 Company

監査詳細 Record with key '会社番号 = 1 - 会社名 = Develop' was updated.
Modified attributes:
会社名 = Develop! (Old value = Dvelop)
会社ウェブサイト = http://www.dvelop.com.uyu (Old value = http://www.dvelop.com.uv)
会社更新日 = 23/02/17 17:41 (Old value = // 00:00)
会社更新者 = testuser (Old value =)

監査テーブル名 PersonContact

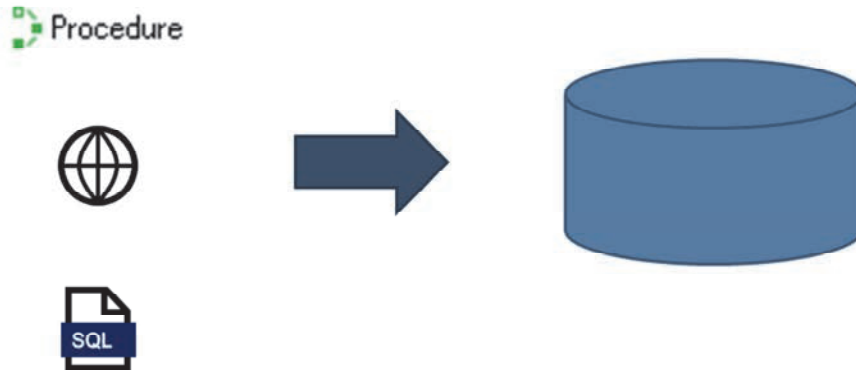
監査詳細 Record with key '連絡先番号 = 11 - 連絡先名 = Sophia' was deleted.
Attributes:
社員番号 = 2
社員名称 = Petersons Johanne
連絡先名 = Sophia

監査機能によってトランザクションに記録されたデータの一例を表示しています。

データの新規登録、更新、削除のケースを提示しています。

唯一更新の場合のみ更新前後のデータが記録されることがわかります。

監査: 記録されないケース



前述の通りこの監査機能はトランザクションオブジェクトにプロシージャオブジェクトを呼び出す処理が自動的に追加されるものとなります。

これは、監査用データを記録できるケースはトランザクションオブジェクトの画面を利用した場合と、ビジネスコンポーネントとして利用した場合であることを意味します。

そのため、下記のようなケースについてはこの機能では記録できないことを注意してください。

- ・ プロシージャオブジェクトの機能によるデータの新規登録、更新、削除
- ・ 別アプリケーション（サービス）からのテーブルへの新規登録、更新、削除
- ・ テーブルへ直接 SQL 文を発行しての新規登録、更新、削除

8. まとめ

WorkWithPlus for Web の性質

必須操作

完全な柔軟性

テンプレートに基づくオブジェクト

完全なオブジェクトの自動生成

テンプレートの変更



インスタンスの更新



項目属性の追加

WorkWithPlus for Web をご利用いただくうえで必ず覚えておいていただきたい点を改めてご紹介します。

WorkWithPlus for Web の性質:

オブジェクトの生成における完全な柔軟性により、WorkWithPlus for Web で任意の種類のアプリケーションを必要に応じて生成できます。

WorkWithPlus for Web によって生成されたすべてのオブジェクトは、設計、動作、および構造の一元化を行うためにテンプレートに基づいています。

WorkWithPlus for Web を使用してアプリケーションのすべてのインターフェースオブジェクトを自動生成することができます。

そのためには、トランザクションオブジェクト、Web パネルオブジェクト、プロシージャオブジェクトに対してWorkWithPlus for Web を適用します。

必須操作:

下記 2 つの変更を行った場合、必ず [インスタンスの更新] を実行します。

- ・ WorkWithPlus 設定 においてテンプレートを変更
- ・ トランザクションオブジェクトの Structure エLEMENT で、項目属性を追加



WORKWITHPLUS
FOR WEB

問い合わせ

ご質問またはご提案がありましたら、ご連絡ください。

東京都品川区西五反田2丁目27番3号 

03-6303-9381 

www.genexus.jp 

info@genexus.jp 