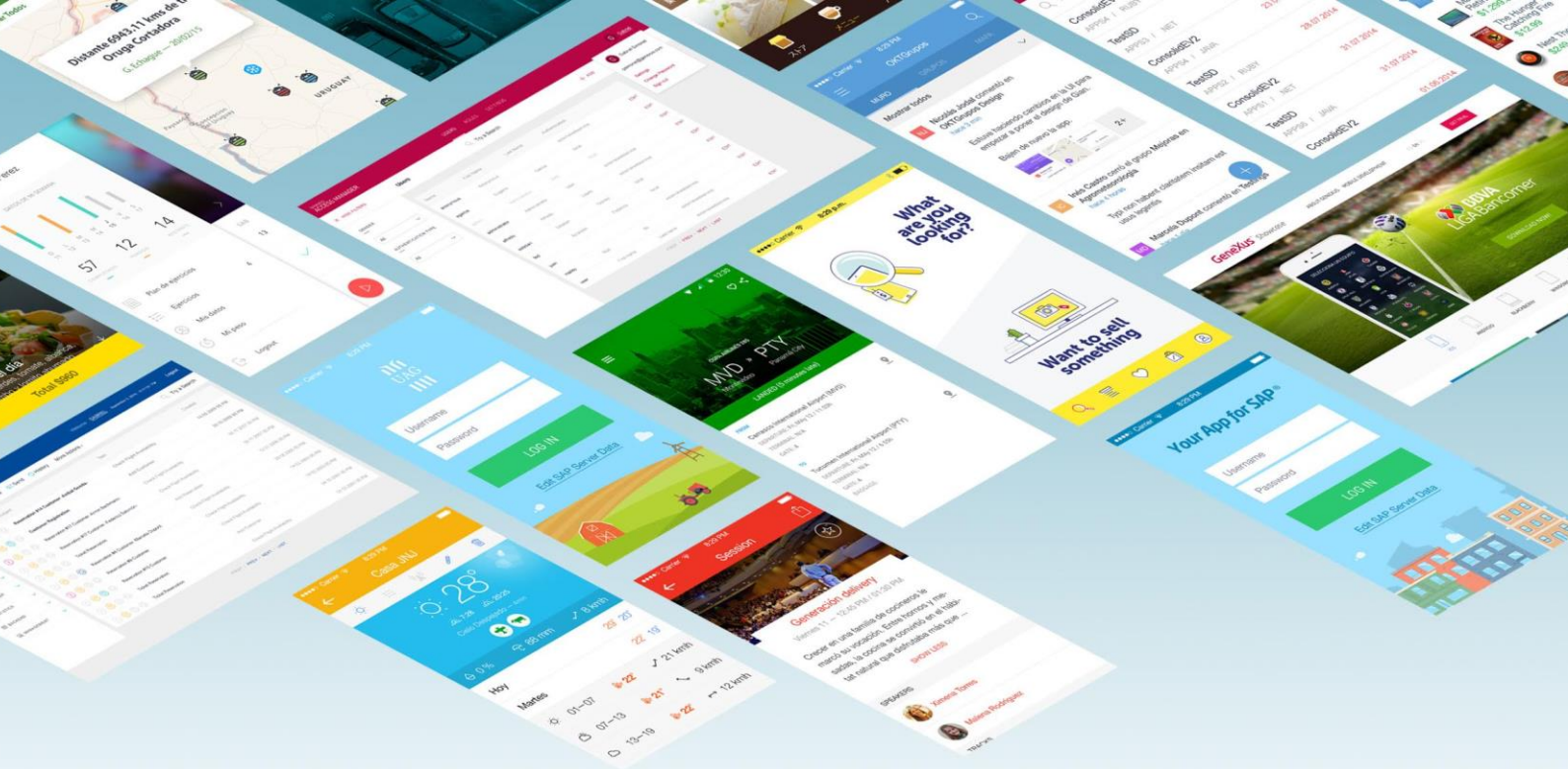


はじめての

GeneXus™

ローコードツールを利用したはじめてのアプリケーション作成

Copyright © GeneXus S.A. 1988-2021.

All rights reserved. 本書は、GeneXus S.A.の明示的許可なしにはいかなるメディアにも複写することはできません。本書の内容は個人的使用のみを目的として提供するものです。

登録商標:

GeneXus は GeneXus S.A.の商標または登録商標です。本書において取り上げているその他の商標はすべて、それぞれの所有財産です。

目次

目次 2

序論 3

前提条件 3

GENEXUS の開始 3

新しい GENEXUS アプリケーションの作成 4

最初のオブジェクトの定義 5

アプリケーションのはじめての生成と実行 20

アプリケーションの拡張 30

ビジネスルールの追加 34

計算の定義：式 36

パターンの使用（WEB 用およびモバイルデバイス用） 40

ここまで定義してきたものを別の言語、別のデータベースで生成するには 66

GENEXUS のその他の特長 69

次のステップ 72

序論

GeneXus は、複数の言語で、異なるプラットフォーム用のソフトウェアアプリケーションを迅速に生成できるローコード開発スイートです。GeneXus には次のような特長があります。習得が簡単で、生産性が高く、クロスプラットフォームで、将来にわたって使用可能であるため、デジタル資産を守ることができ、新しい技術も容易に取り入れることができます。

たとえば、GeneXus では、開発者が選択したターゲットプラットフォーム (特定の言語、データベース、環境、およびプラットフォーム、さらにレスポンシブ Web デザインの有無など) に対応した、Web 用 / モバイルおよびスマートデバイス用のアプリケーションを生成できます。

本書は、GeneXus でアプリケーションを開発するためのビギナーズガイドです。

本書に関するご質問は次のアドレスにお問い合わせください。

customersupport@genexus.jp

前提条件

GeneXus を使用するための前提条件につきましては、弊社ホームページダウンロードページ
（https://www.genexus.jp/community-and-support-jp/downloads02）の
ドキュメント→教育 / e ラーニング / 資格試験 からダウンロードできる「受講者端末準備のご案内」記載の
「A. GENEXUS 利用環境とアプリ実行環境の準備」の箇所をご確認ください。

GENEXUS の開始

GeneXus を起動すると、IDE (統合開発環境) と呼ばれる次のようなインターフェースが表示されます。このインターフェースは使い方が簡単で、各開発者が独自に設定できるようになっています。

このインターフェースは複数のウィンドウで構成されています:

1. **メインウィンドウ (開始ページ)**: ツールに関する技術情報や GeneXus コミュニティ (ニュースや、ほかの開発者が投稿したソリューション) が動的に表示されます。また、最近使用されたプロジェクトも表示されるため、ここから開くことができます。新しいプロジェクトを作成することも可能です。

2. **ナレッジ ベース ナビゲーター:** 現在開いているプロジェクトのオブジェクトと設定が表示されます。

3. **ツールバー:** GeneXus でよく使用される機能の使いやすいインターフェースが表示されます。

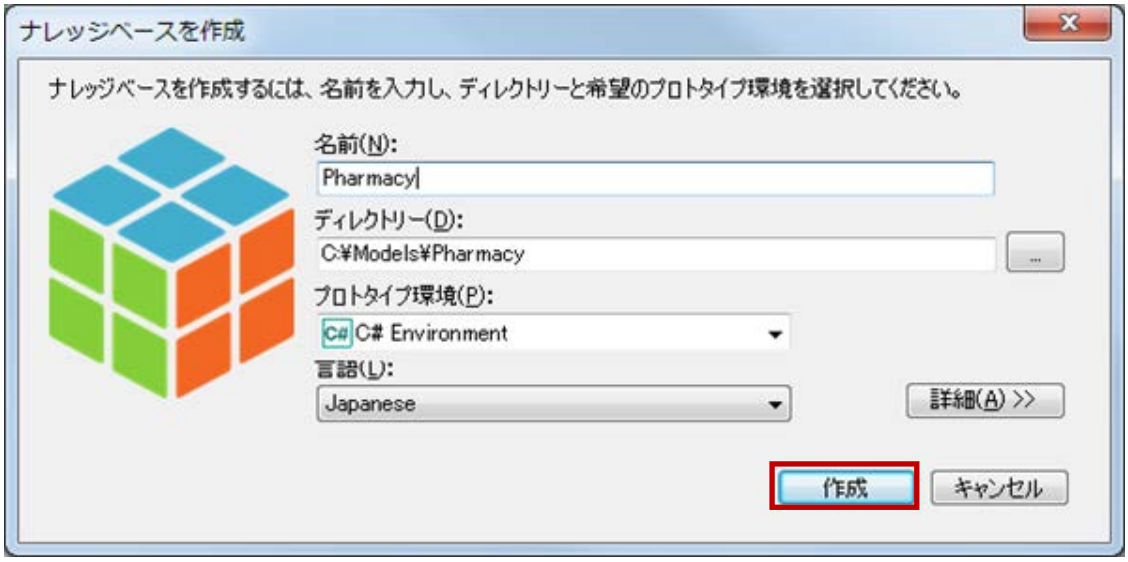
4. **[プロパティ] ウィンドウ:** 開発者の現在のコンテキスト (選択されているオブジェクト、項目属性、変数、コントロールなど) に関係するプロパティが表示されます。

5. **出力:** 実行されたアクションの出力が表示されます。

新しい GENEXUS アプリケーションの作成

新しい GeneXus アプリケーションの開発を始めるにあたって、新しい**ナレッジベース**を作成する必要があります (ナレッジベースが GeneXus プロジェクトになります)。

ツールバーで **[ファイル]** → **[新規]** → **[ナレッジベース]** を選択すると、次のようなダイアログボックスが表示されます:



本書で定義するドラッグストア用のサンプルアプリケーションは現実的なものですが、簡略化されたアプリケーションになっています。ナレッジベースの名前は「Pharmacy」(あるいは「PharmacySystem」) とします。また、ナレッジベースの作成先のパスを入力する必要があります。

次に、**[プロトタイプ環境]** コンボボックスで、いずれかのプログラミング言語を選択します。GeneXus では、選択された言語を使用してアプリケーションプログラムを生成し、またデータベースの作成 / 維持に必要なプログラムも生成します。上の画像では **[C# Environment]** が選択されています。以降の作業でデータベースの詳細を入力する必要があります。

[言語] コンボボックスでは、ボタンキャプション、ラベル、ユーザーへのメッセージなどを自動生成する言語を選択できます。既定の言語は [Japanese] です。

[作成] ボタンを押すと、ナレッジベース作成プロセスが開始します。

最初のオブジェクトの定義

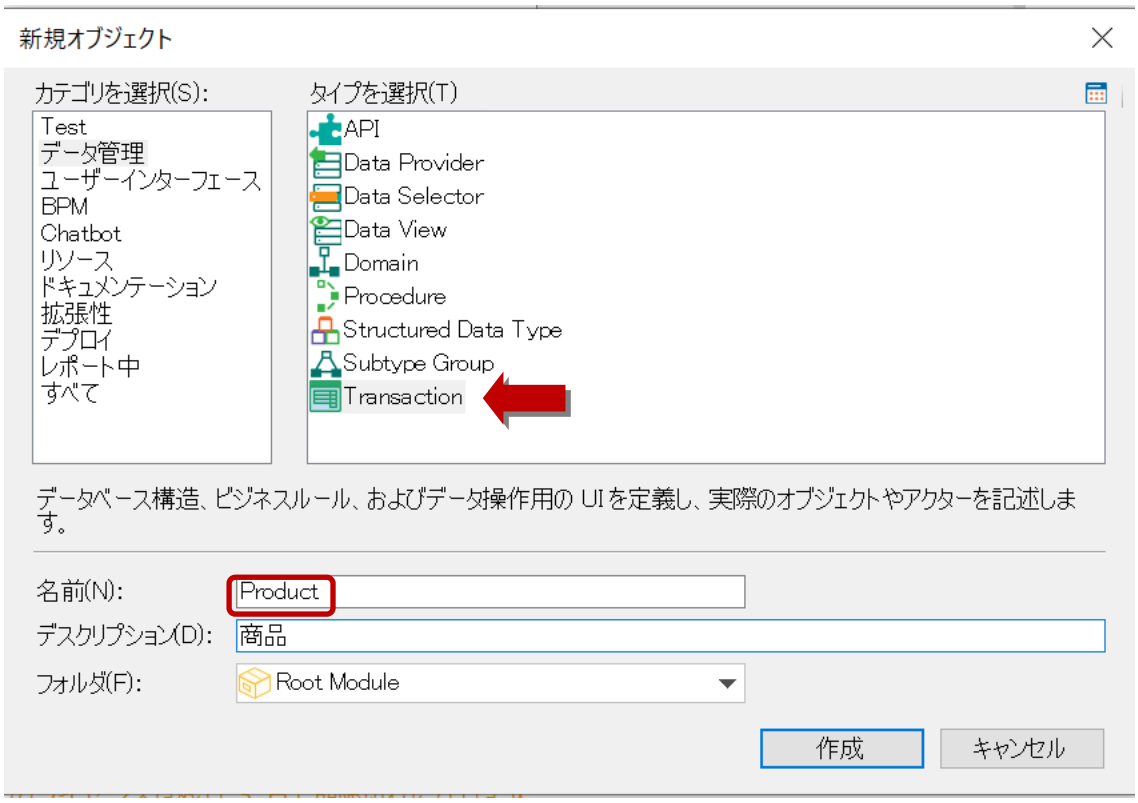
新しいナレッジベースを作成したら、次にユーザーの業務内容を記述します。このためには、現実の (実際の業務に関連する) オブジェクトを特定し (商品、請求書、顧客など、ユーザーが説明時に口にした名詞に注目することをお勧めします)、それらを GeneXus **オブジェクト**で定義する必要があります。

GeneXus の開発者は、テーブル定義、正規化、プログラム設計、プログラミングなどの低レベルの作業は行いません。代わりに、ユーザーの現実のオブジェクトを記述するという高レベルの作業を行います。その後、GeneXus が、**完全に自動化された**方法で、定義されたオブジェクトを分析し、選択されたプラットフォームに対応したデータベースとアプリケーションプログラムを設計します。

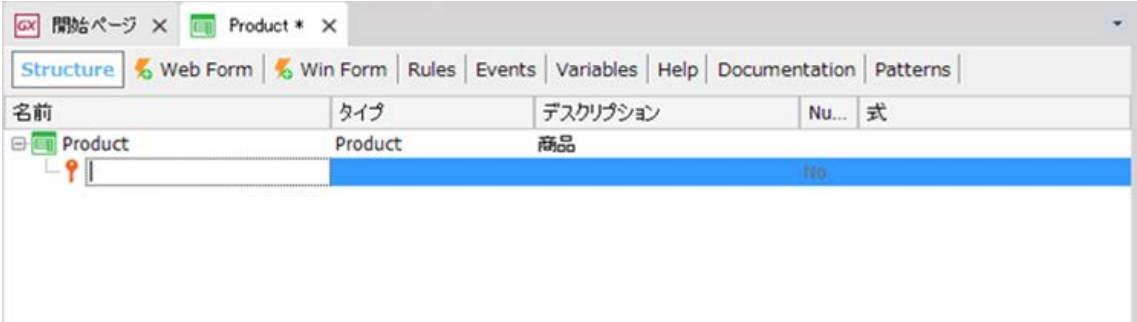
たとえば、アプリケーションを依頼しているドラッグストアが、販売中の商品を記録できるようにしたいと言っているとします。

特定されたそれぞれの現実のオブジェクトを記述するには、Transaction タイプの GeneXus オブジェクトを作成する必要があります (データベースのトランザクションとは無関係です)。それでは、商品を記述するトランザクションオブジェクトの作成方法について説明します。

ツールバーで **[ファイル]** → **[新規]** → **[オブジェクト]** を選択すると、次のようなダイアログボックスが表示されます。ここで、作成するオブジェクトのタイプを選択し、名前を入力します。タイプは Transaction を選択する必要があります。名前は次のように入力します: **Product**



[作成] ボタンをクリックすると、**Product** トランザクションが作成され、構造を定義できるように Product トランザクションが開きます:



各トランザクションにはいくつかのセクションがあります (少しずつ説明していきます)。まず、トランザクション構造では、現実のオブジェクトを記述する項目属性やフィールドを定義できます。

たとえば、ドラッグストアで、すべての商品のコード、名前、販売価格、在庫、種類 (医薬品や化粧品など) の記録を取る必要があると言われたとします。したがって、商品ごとに記録する必要があるこれらのデータと、このトランザクションに作成する項目属性を一致させます。

上の画像では、最初の項目属性を入力できるように、トランザクション構造の最初の行が作成されています。また、この行には鍵のアイコンが付いています。これは、どのトランザクションも 1 つ以上の項目属性を識別子、すなわちキーロールとして設定する必要があるためです。

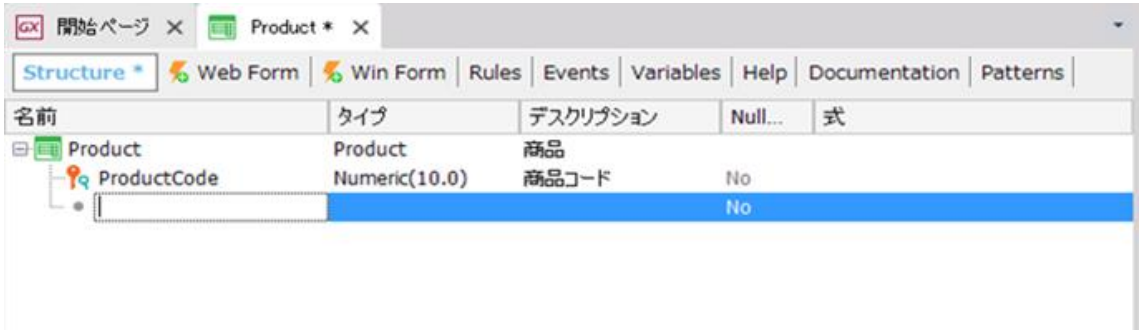
識別子すなわちキー項目属性の概念の目的は、各商品を一意に識別することです (どのオブジェクトも同様)。したがって、ユーザーは、識別子の値が同じ 2 つの商品を入力することはできません。明らかに、商品コードが Product トランザクションのキー項目属性になります。次に、その定義方法を説明します。

キーボードのドットキーを押すと、自動的に、項目属性名の接頭語としてトランザクション名が表示されます:

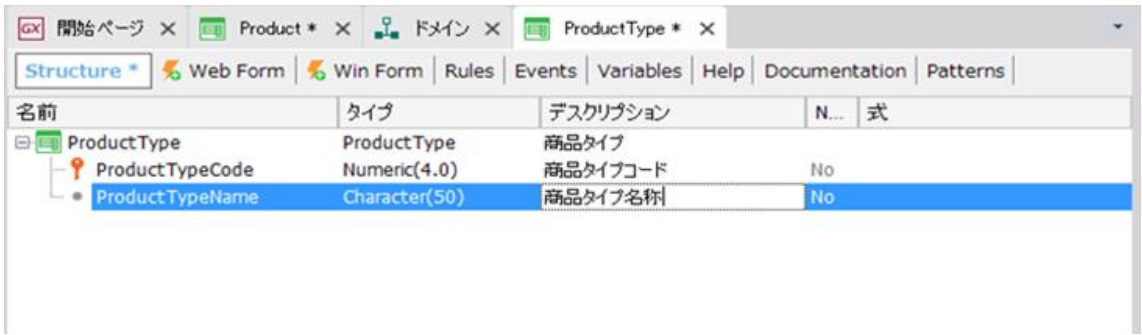
これにより、**Product** という接頭語の後に **Code** と入力するだけで済みます:

次に、Tab キーを押すと、この項目属性に格納されるデータのタイプを選択できるようになります。既定のデータタイプは次のとおりです: 小数部なしの 4 桁の数字。しかし、ドラックストアから、商品コードは常に最大 10 桁の数値となるようにしてほしいという要望があったため、長さを 10 に変更する必要があります:

Enter キーを押すと、2 つ目の項目属性を定義できる新しい行が開きます:



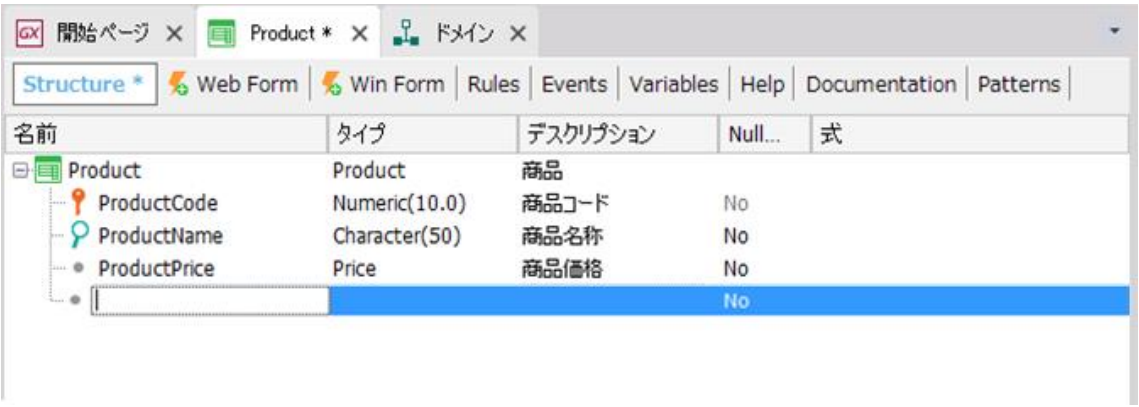
ここでもキーボードのドットキーを押し、(**ProductName** となるように) **Name** と入力して、項目属性名を完成させる必要があります (タイプは Character、長さは 50):



次に、**ProductPrice** 項目属性を追加する必要があります（タイプは Numeric、整数部 9 桁、小数部 2 桁):

価格や金額を定義する項目属性は、おそらくいくつか作成する必要があります（ドラッグストアでの商品の購入や販売のため)。したがって、すべての価格で使用できる汎用定義タイプを作成しておくのも一案です。それには、[タイプ] 列に次のように入力する必要があります：先ほど選択したタイプの前に「Price=」と入力します:

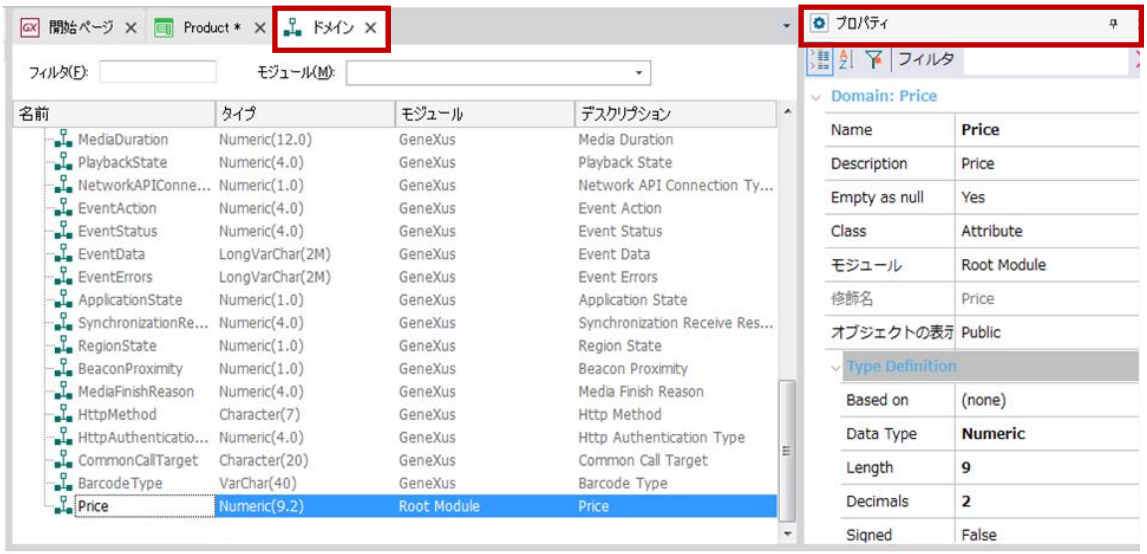
Enter キーを押すと、**ProductPrice** 項目属性が **Price** タイプで設定されたことが確認できます:



この Numeric タイプ（整数部 9 桁、小数部 2 桁）の **Price** 定義を**ドメイン**と呼びます。

ドメインにより汎用定義が可能になります。ドメインの利点の 1 つは、後になって価格を異なる長さの Numeric にする必要が出てきた場合、ドメイン定義を変更するだけで、そのドメインを使用しているすべての項目属性を更新できるという点です。

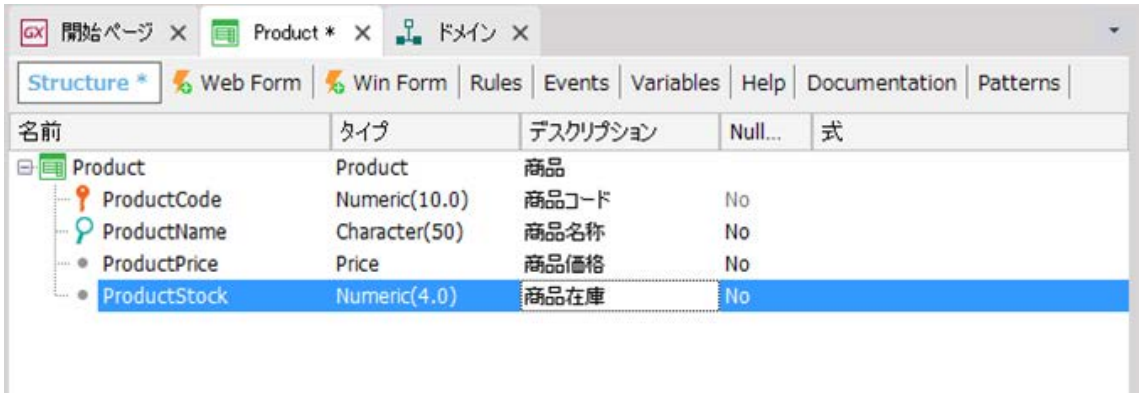
ナレッジベースの作成済みドメインを表示するには、ツールバーで **[表示]** → **[ドメイン]** を選択します:



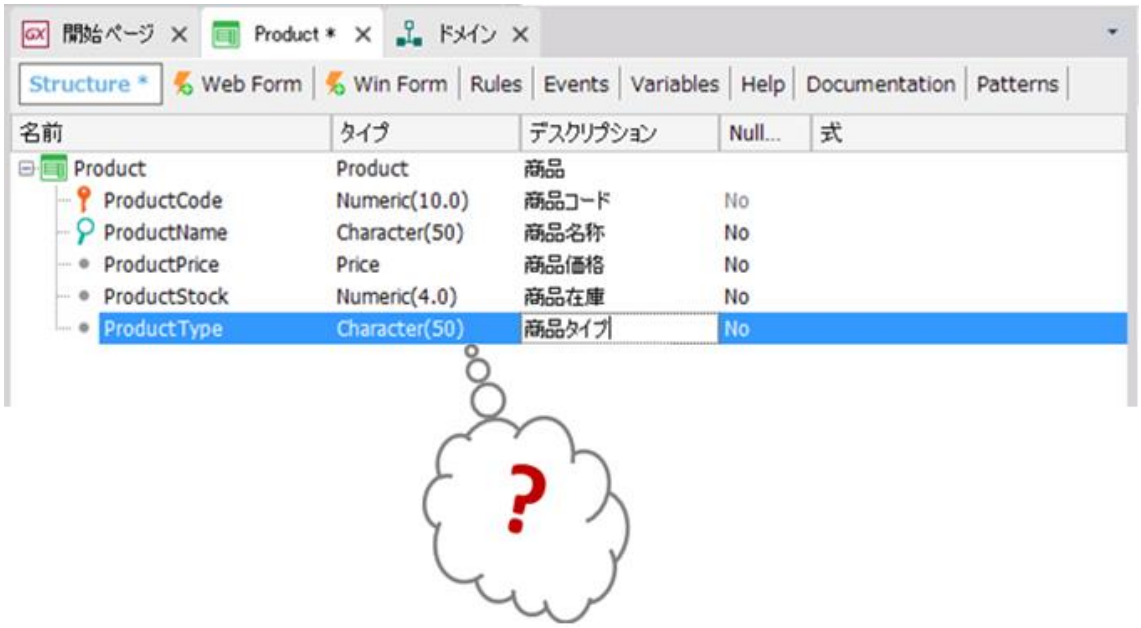
この画像のように、GeneXus では自動的にさまざまなドメインが作成されます。特定のドメインをクリックすると、[プロパティ] ウィンドウがリフレッシュされ、選択したドメインのプロパティが表示されます。ここでは、ドメインのデータタイプの設定だけでなく、ほかのプロパティも定義することができます。

[プロパティ] ウィンドウは、ほかの項目属性やドメイン、オブジェクトなどを選択するたびにリフレッシュされ、設定可能な対応するプロパティが表示されます。[プロパティ] ウィンドウが表示されない場合は、F4 キーを押すと開きます。

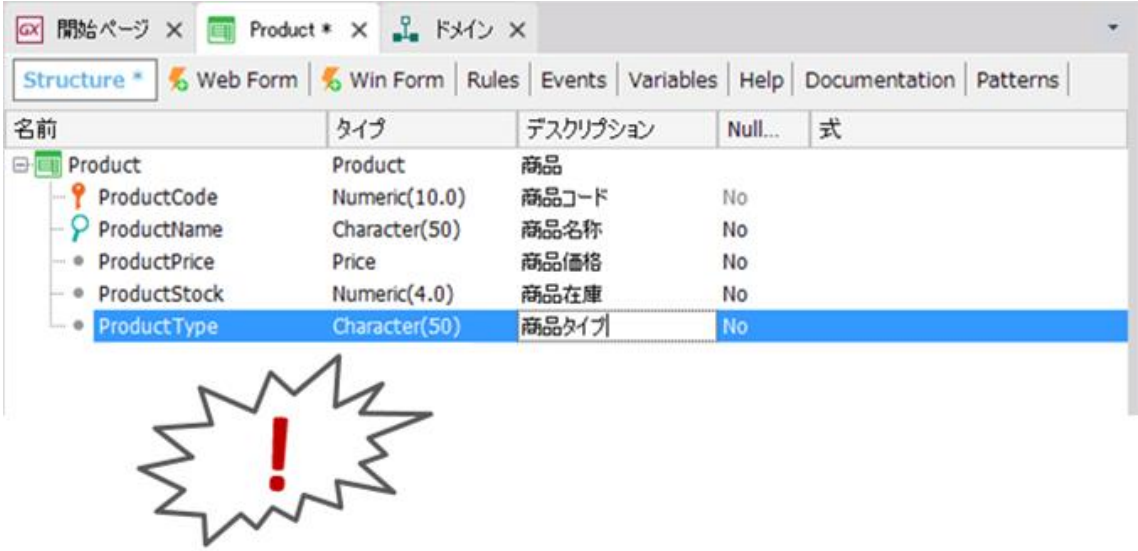
Product トランザクションに戻ります。次に定義する必要がある項目属性は **ProductStock** です。タイプを Numeric、長さを 4 にします:



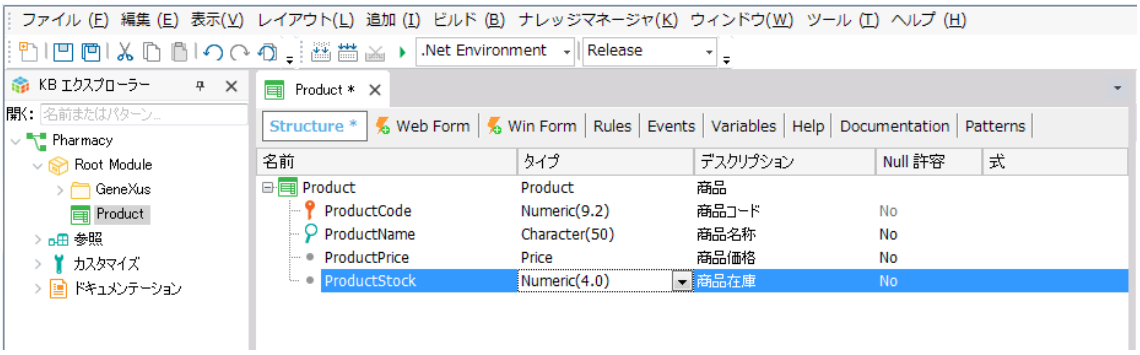
次に、商品タイプを記録する必要があります。**ProductType** という項目属性を Character(50)タイプにして作成することはできますが...



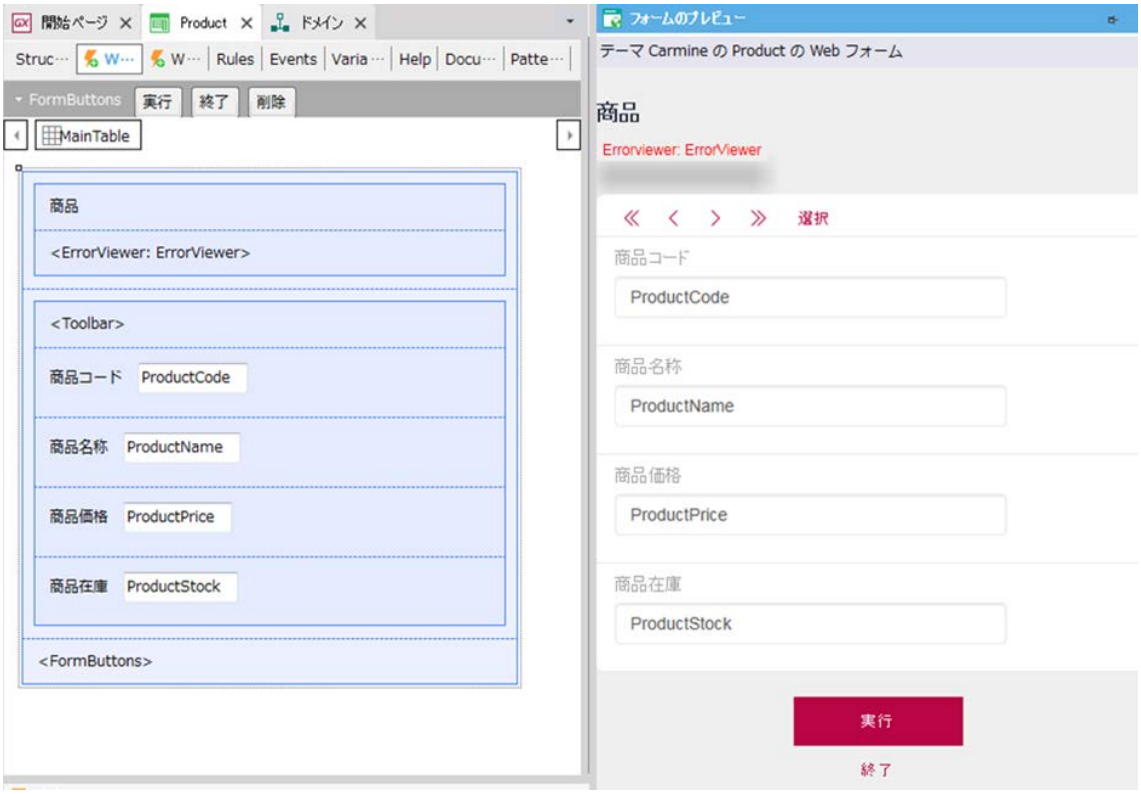
ユーザーが同じタイプの 2 つの商品を入力したい場合はどうなるでしょうか。ユーザーは同じタイプ名を 2 回入力することになります。しかも、まったく同じになるように注意しながら入力しなければなりません。また、後になって、特定のタイプの商品をすべて検索する必要が出てくるかもしれません。そのときも、まったく同じになるようにタイプ名を入力しなければなりません。



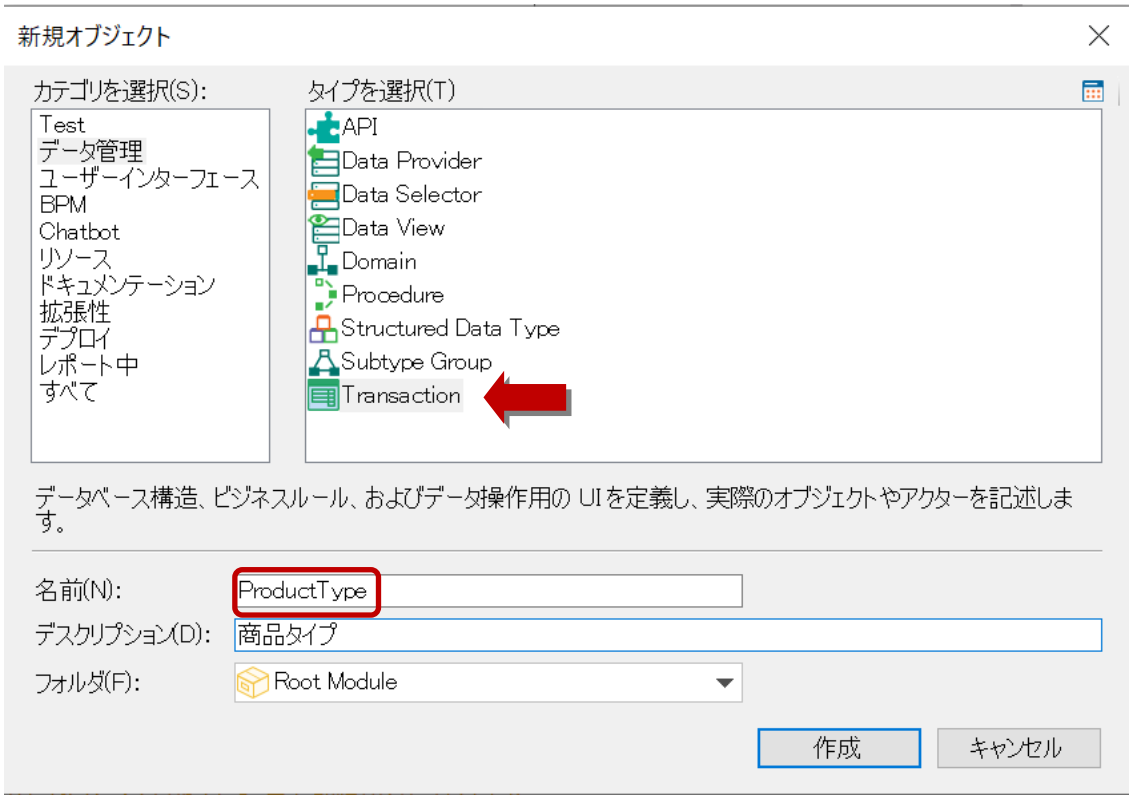
タイプは 1 か所に 1 回だけ入力し、商品ごとに、対応する商品タイプを参照するようにした方が合理的だと考えられます。そこで、トランザクション構造から **ProductType** 項目属性を削除し、次のような **Product** トランザクションを保存します:

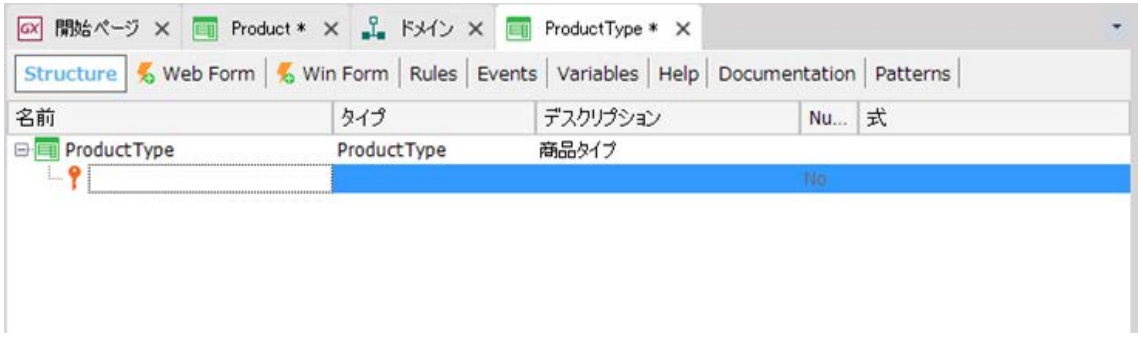


定義された構造に従って、GeneXus が自動的に Web フォームを設計します。ユーザーは、実行時にこのフォームで商品の追加、更新、削除を行うことができます:

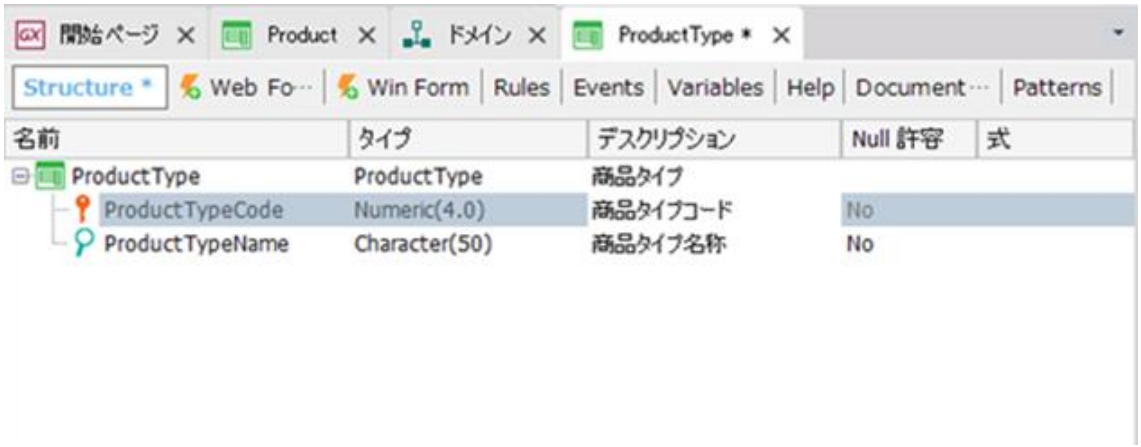


次に、商品タイプを記録する別のトランザクションを作成します。それから、各商品に商品タイプを割り当てます:





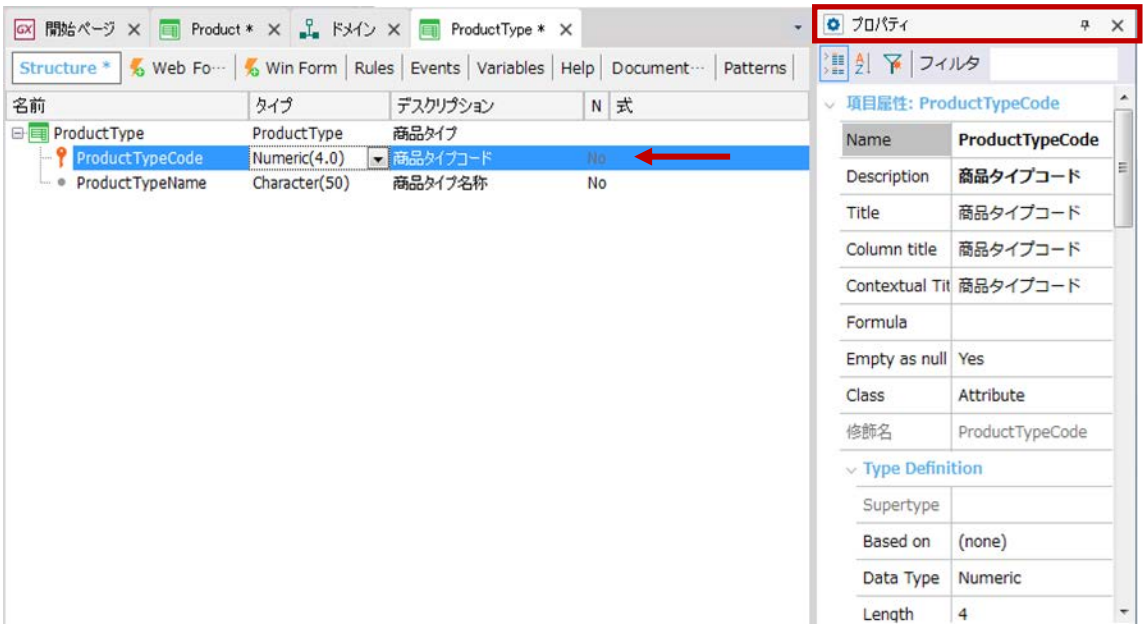
商品タイプごとにコードと名前を格納します:



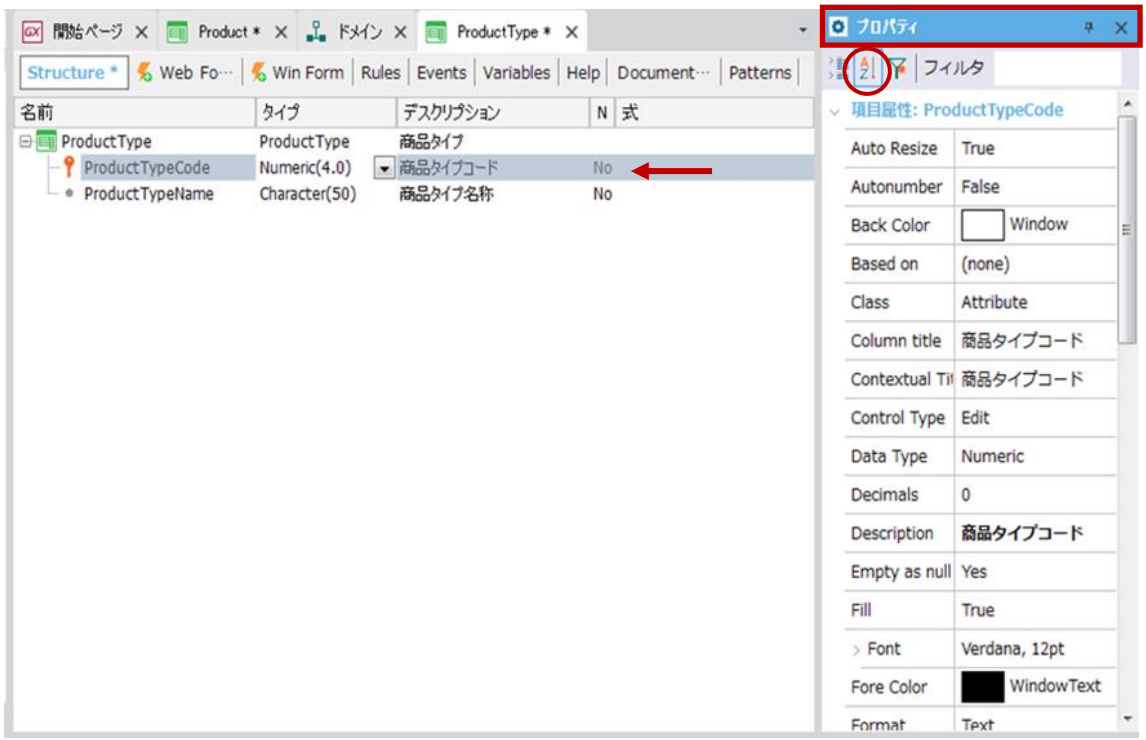
前述のように、項目属性を定義するときは、キーボードのドットキーを押します。すると自動的に、接頭語としてトランザクション名が入力されるので、続けて項目属性名の後の部分のみ入力します。

接頭語としてトランザクション名を使うという項目属性の名称設定方法は、項目属性の定義を簡便にするだけではありません。どこにあっても、記述されているオブジェクトが分かるため、項目属性名を見たときに簡単に理解できるようにするための GeneXus コミュニティの規則でもあります。

ProductTypeCode 項目属性のプロパティを見てみましょう:

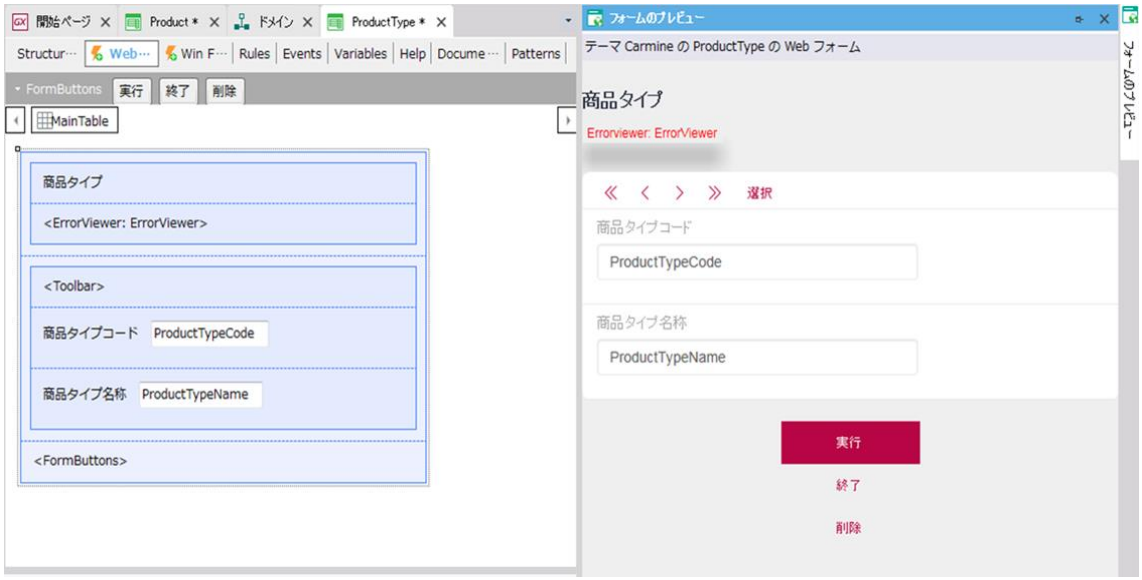


プロパティはアルファベット順に並べ替えることができます:

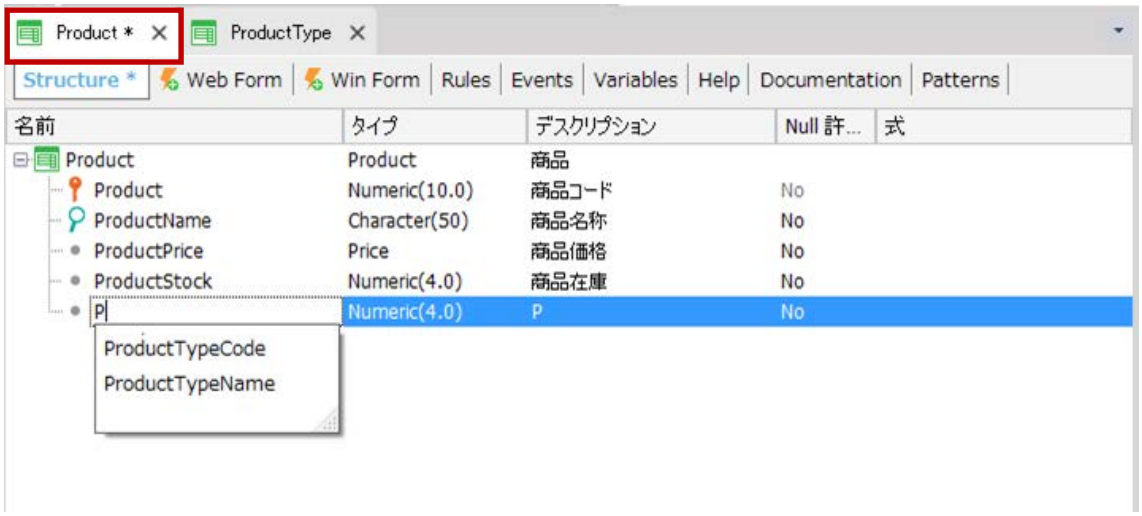


[Autonumber] プロパティについて説明します。既定では [False] に設定されています。これを [True] に変更すると、エンドユーザーが入力した新しい商品タイプはすべて、順番に自動ナンバリングされます。したがって、この識別子項目属性の [Autonumber] プロパティは [True] に設定しておきます。**ProductType** トランザクションを保存します。

前述のとおり、各トランザクションには、トランザクション構造に従って自動的に設計された Web フォームがあります。次の画像は **ProductType** の Web フォームです:

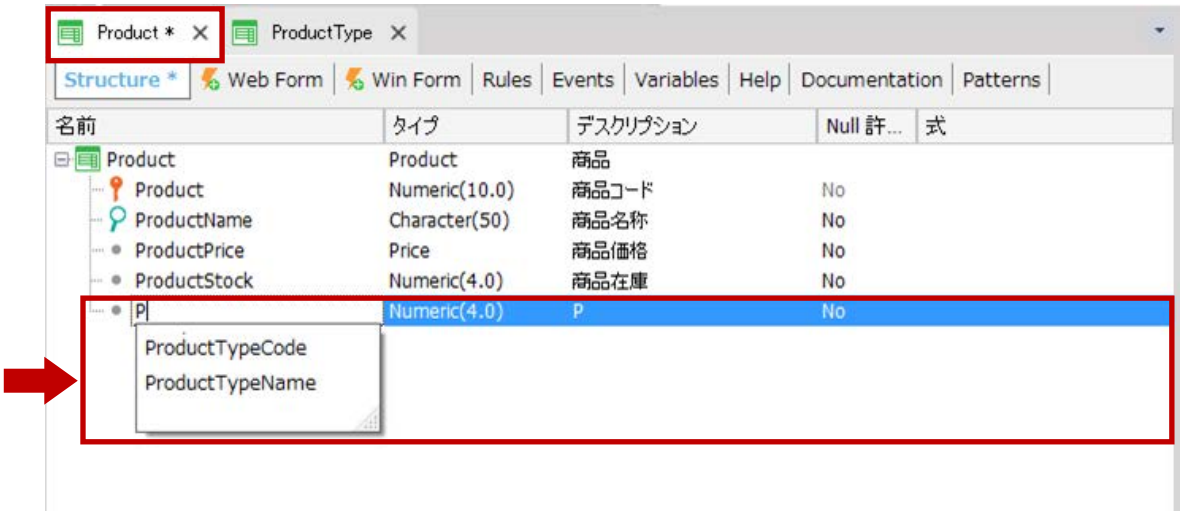


次に、各商品に商品種類を割り当てます。**Product** トランザクションに戻り、「P」と入力してトランザクション構造の新しい行を開始します。「P」と入力すると、この文字で始まるナレッジベース内の既存の項目属性のリストが表示されます:



ProductTypeCode を選択して、その定義をすべて表示します。

このトランザクションに **ProductTypeName** 項目属性も含めます。ユーザーは、このトランザクションを実行して商品タイプコードを選択する際、対応する商品タイプ名称をフォームで確認したいと思うはずだからです。ここで、複数のトランザクションに含まれるこの 2 つの項目属性について説明します:



ProductTypeCode は、**ProductType** トランザクションの識別子項目属性です（すなわち、このトランザクションの主キーです)。したがって、主キーが別のトランザクションに含まれている場合、GeneXus は、その項目属性はそこでは外部キーのロールを果たしていると判断します。

あるトランザクションの主キーである項目属性を別のトランザクションに含めることで、2 つのトランザクションを関連付けることができます。

GeneXus は、項目属性名に基づいてリレーションを確立します。したがって、異なるトランザクションで同じ名前の項目属性を見つけると、それらは同じ概念を参照していると見なします。

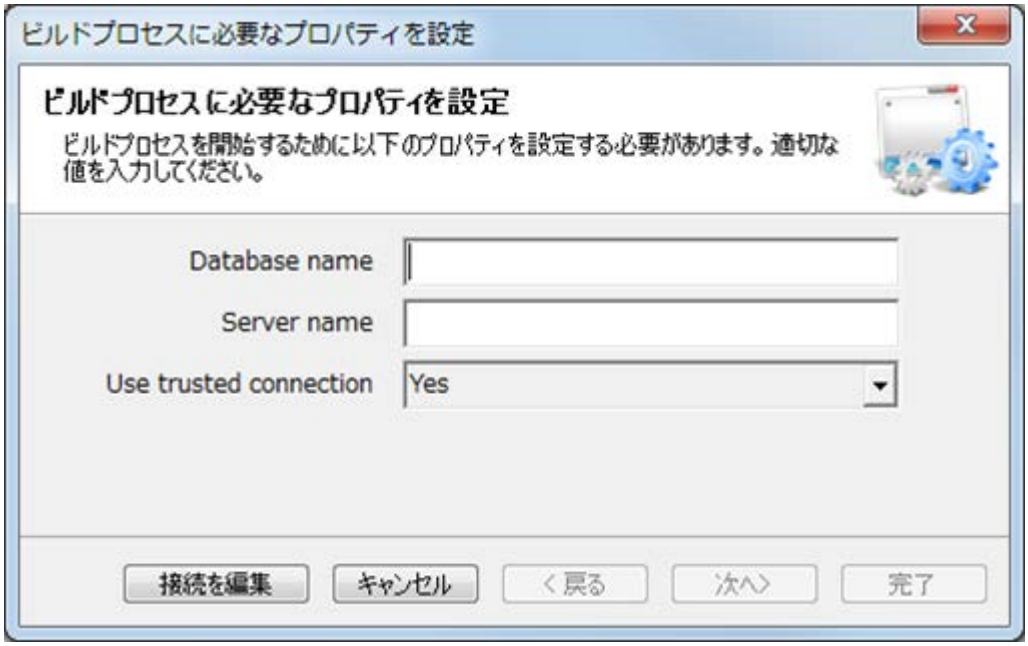
ProductTypeName 項目属性も両方のトランザクションに存在します。しかし、定義済みトランザクションの識別子として設定されていません。したがって、GeneXus は、この項目属性を**セカンダリー**項目属性と見なします。GeneXus は、データベースに作成する **ProductType** 物理テーブルには **ProductTypeName** を含めますが、**Product** 物理テーブルには含めません。

トランザクションの概念と物理テーブルの概念は同じではありません。トランザクションは、現実のオブジェクトを記述するためにナレッジベースに作成する GeneXus オブジェクトです。GeneXus は、当該トランザクション構造と、ナレッジベースに定義されているその他のトランザクション構造を考慮して（さらにいくつかのプロパティも考慮に入れて)、データベースに作成すべき物理テーブルと各テーブルに格納すべき項目属性を決定します。

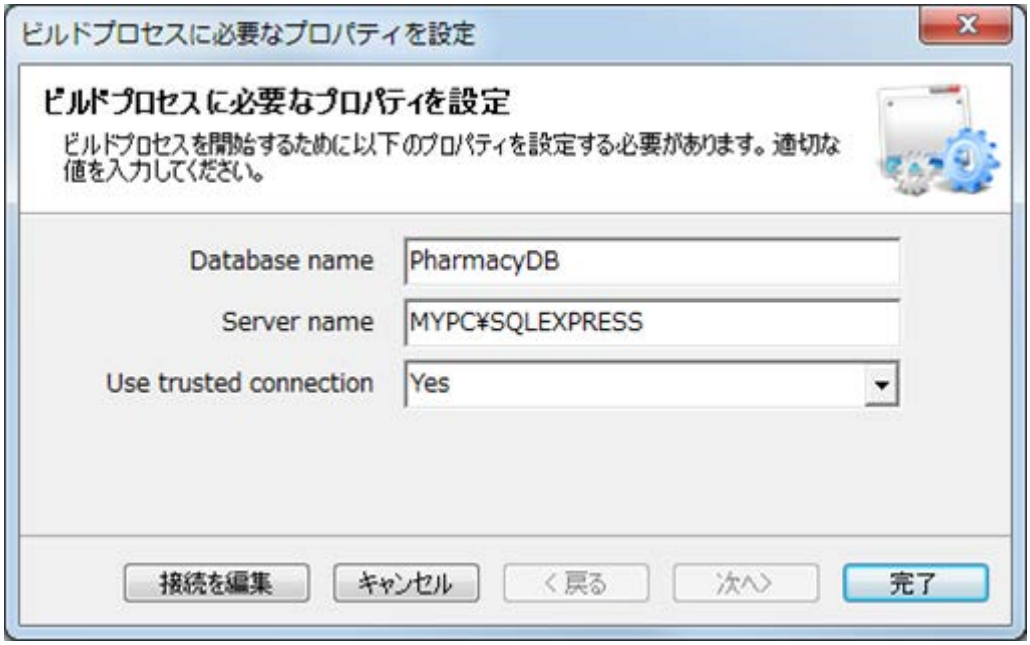
実行時、ユーザーは、**Product** トランザクションフォームを実行する場合、**ProductType** トランザクションによって事前に記録された値を **ProductTypeCode** 項目属性（ここでは外部キー項目属性）に入力する必要があります。これを行わなかった場合はエラーが表示されます。有効な **ProductTypeCode** が入力されると、その **ProductTypeName** 値が取得され、画面に表示されます。

アプリケーションのはじめての生成と実行

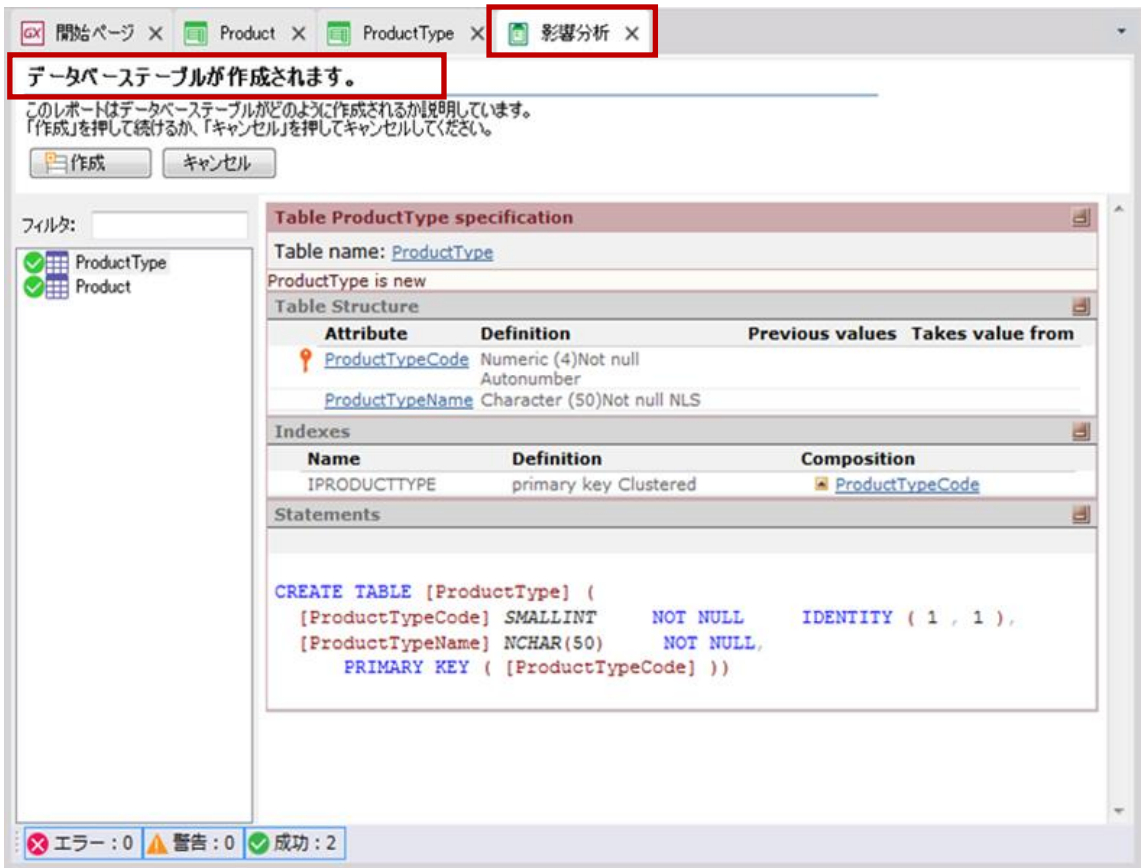
アプリケーションをはじめて生成し、実行する場合は、F5 キーを押します。すると、次のようなダイアログボックスが表示されます。



GeneXus では、アプリケーションをローカルまたはクラウドでプロトタイプすることができます。これを設定するプロパティがあり、既定ではローカルでのプロトタイプに設定されています。したがって、アプリケーションのデータベースのための名前と、インストールされているデータベースサーバーの名前を入力する必要があります:



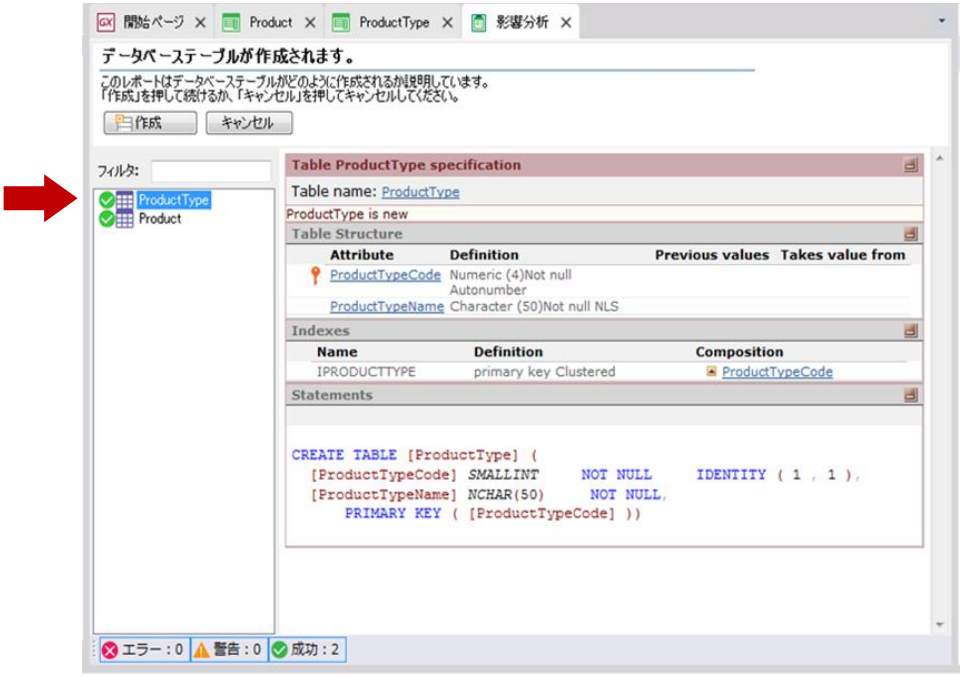
[完了] をクリックすると、ナレッジベース内の新しい定義によって生じる影響が評価され、**[影響分析]** にレポートが表示されます:

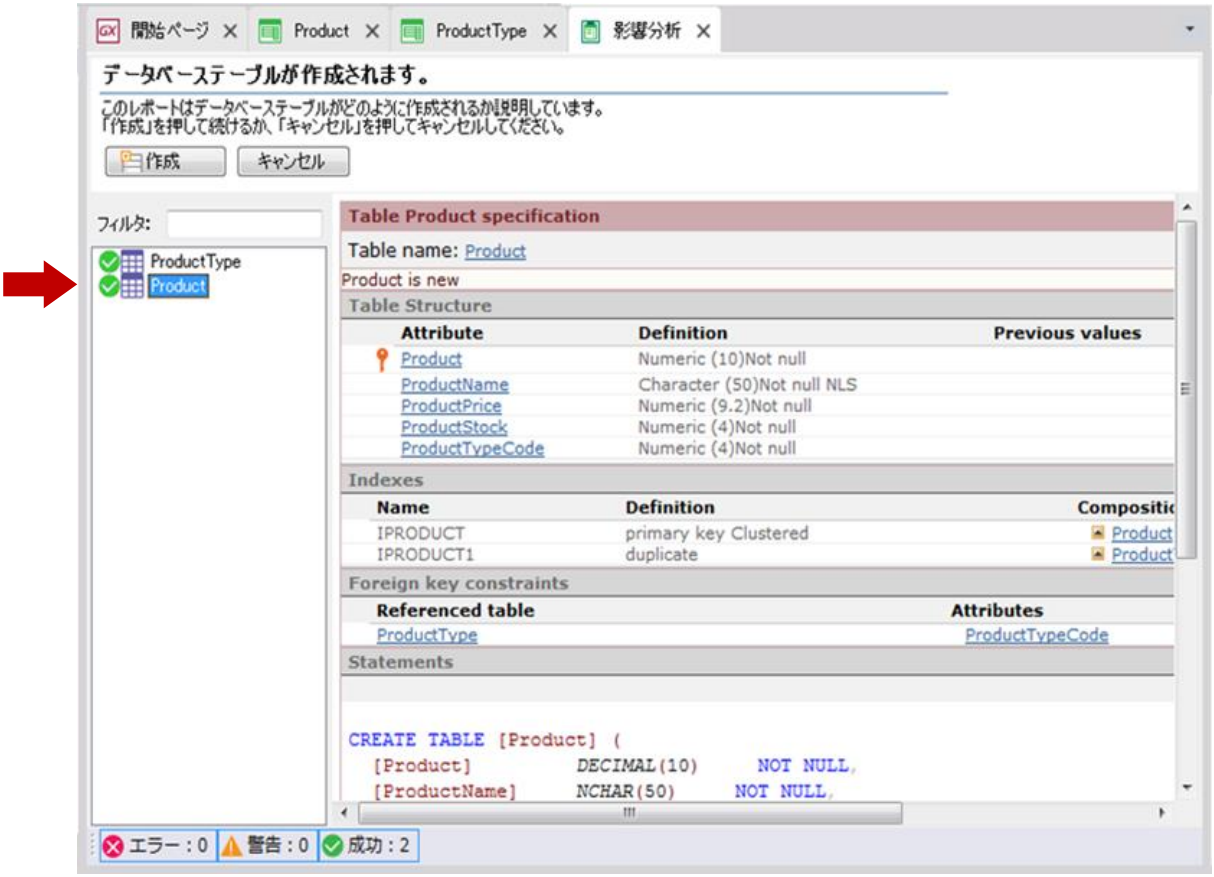


このレポートには、データベースで行う必要がある構造変更が表示されます。

このレポートのメインタイトルから、今回は**「データベーステーブルが作成される」**ことが分かります。

各テーブル（左側のウィンドウの **ProductType** と **Product**）をクリックすると、右側のウィンドウに、そのテーブルに含まれる項目属性が表示されます:

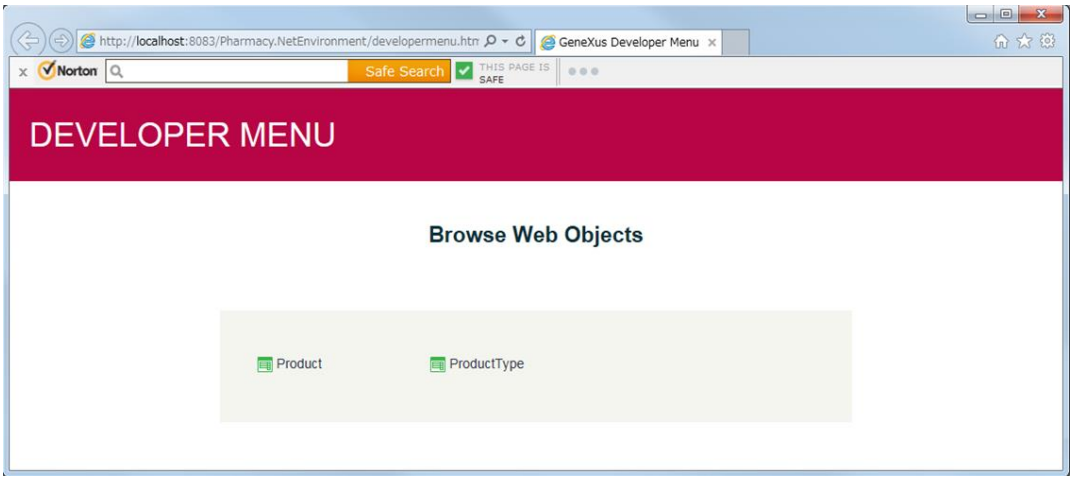




前述のとおり、(Product フォームに表示するため) **Product** トランザクション構造に ProductTypeName 項目属性を含めましたが、作成される **Product** 物理テーブルに **ProductTypeName** 項目属性は含まれません。

[影響分析] の提案で問題がなければ、[作成] ボタンをクリックします。データベースを作成し (まだ存在していません)、そのデータベースにトランザクション構造に基づいたテーブルを作成するのに必要なプログラムの作成が開始されます。次に、そのプログラムが実行されます。データベースとテーブルの作成後、選択したプログラミング言語で必要なすべてのコード行が生成され、ユーザーが商品タイプと商品の挿入 / 更新 / 削除を行うことができるアプリケーションが作成されます。

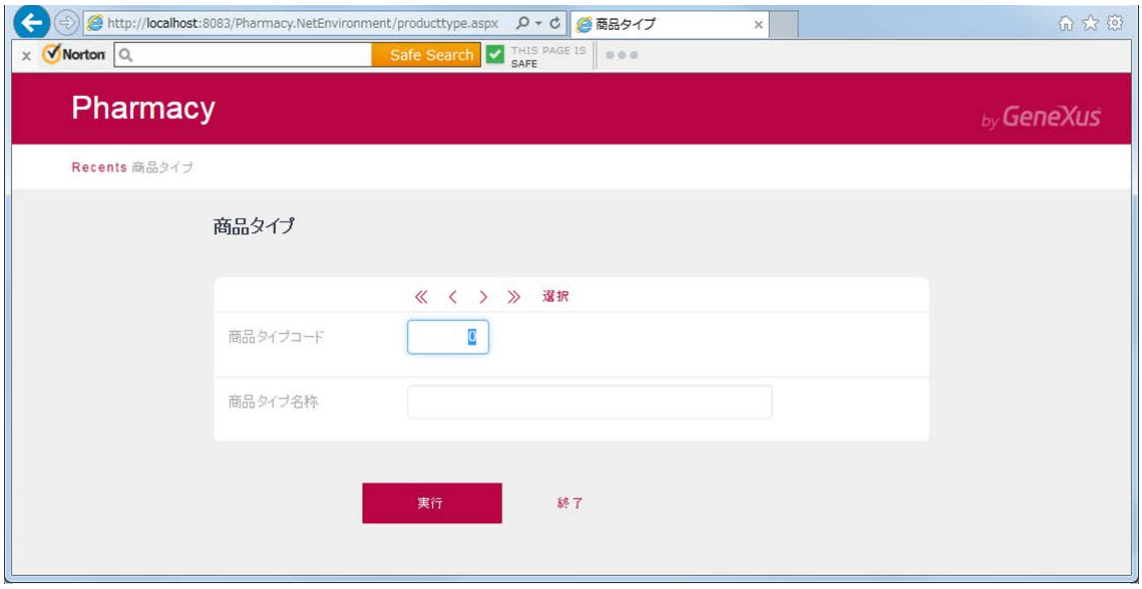
結果 (成功したか、エラーや警告があるか) が表示されます。続けてアプリケーションを実行します:



既定では Web ブラウザーが開き、定義したオブジェクトを簡単に実行できるシンプルなページが表示されます。

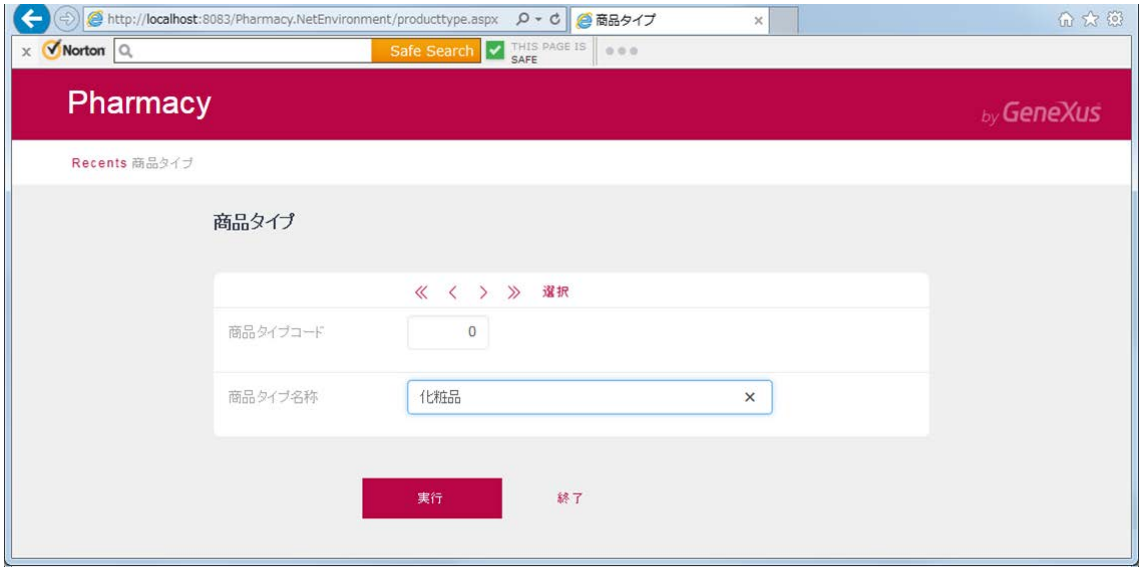
開発者メニューという名前のこのシンプルなページは、その名のとおり開発者用のページです。当然のことながら、ユーザーの画面には表示されません。

ProductType リンクを右クリックし、新しいタブで開くメニューを選択します:



上のページでは、ユーザーが**商品**タイプを追加 / 更新 / 削除できます。1 つ目の商品タイプを入力します。

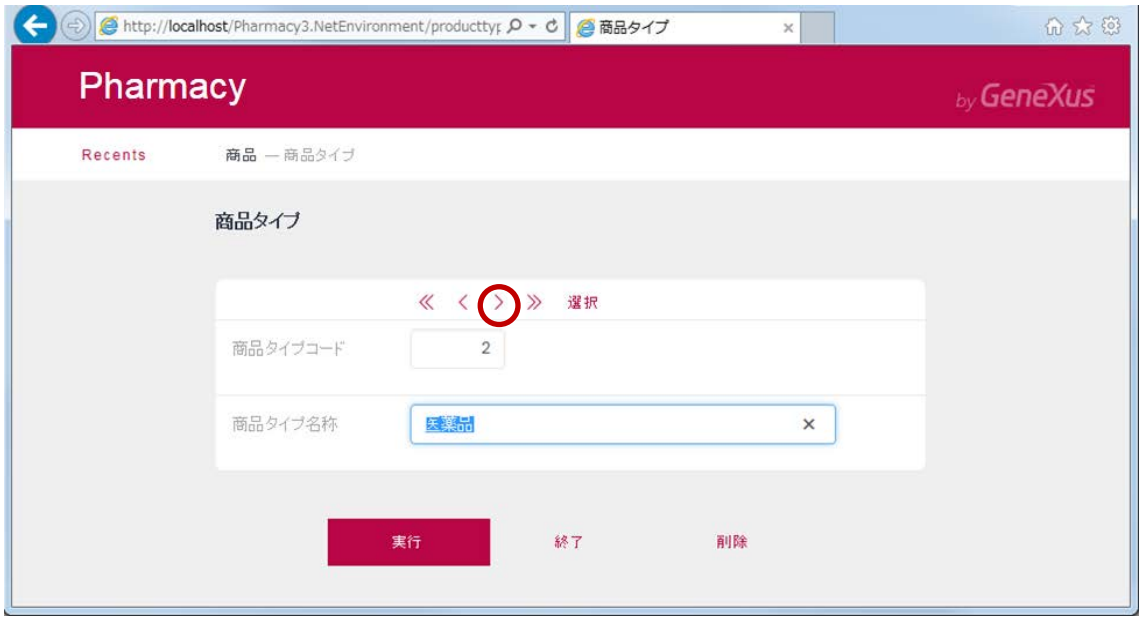
ProductTypeCode 項目属性の [Autonumber] プロパティは [True] に設定されており、自動ナンバリングされるため、ユーザーがこの識別子の値を入力する必要はありません。商品タイプ名称を入力します:



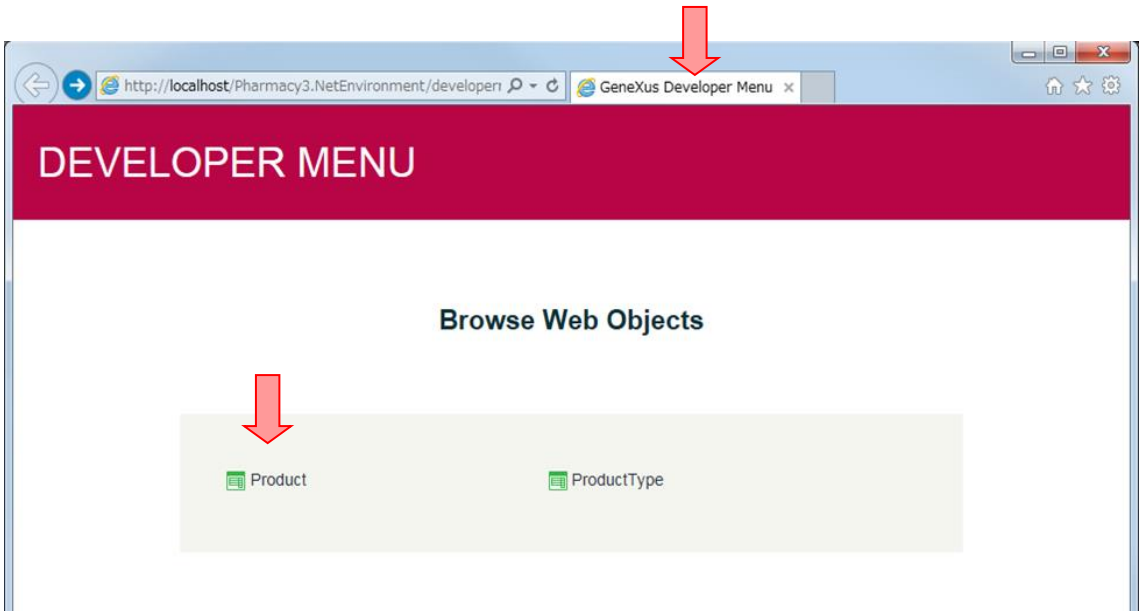
商品タイプ名称を入力し、[実行] ボタンをクリックすると、データが正常に追加されたというメッセージが表示されます。同時に、フォームがクリアされ、ほかの商品種類を入力できるようになります:

2 つ目の商品種類を入力します:

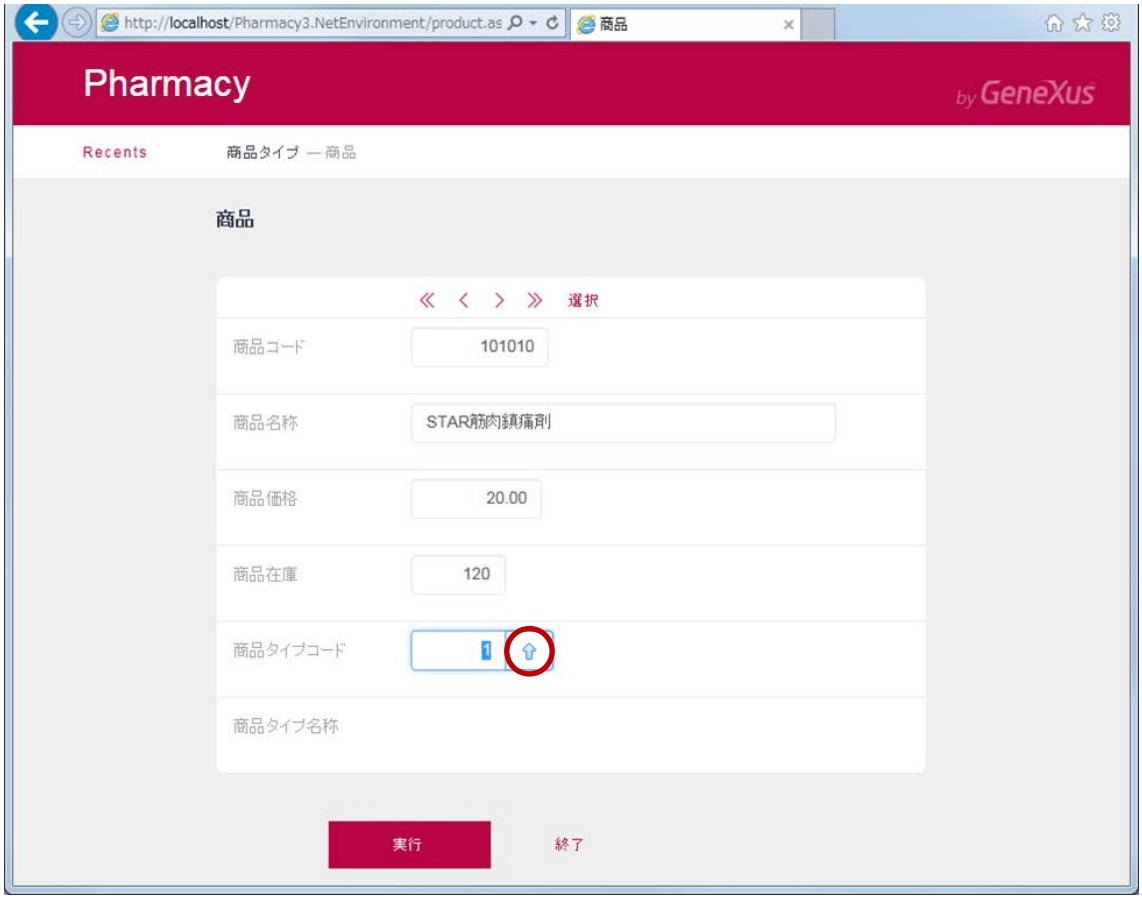
[実行] ボタンをクリックします。登録されたデータに移動し、ナンバリングされていることを確認します:



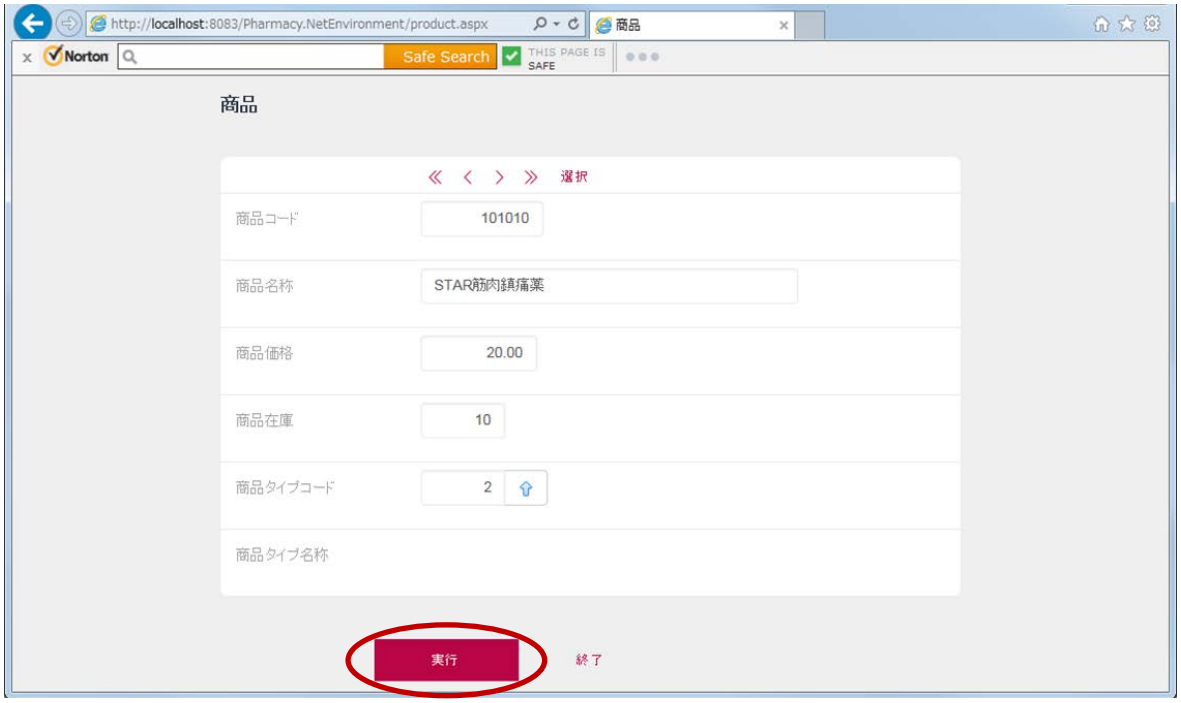
次に、**Product** トランザクションを実行します。開発者メニューページが表示されているナビゲータータブを選択し、Product トランザクションを実行します:



1 つ目の商品を追加してみます:



商品タイプを指定する必要があります。商品タイプコードを覚えている場合は、それを入力します。別の方法として、青色の矢印をクリックしてリストから選択するやり方もあります。



次に、商品タイプを削除してみます:

関連データが Product に存在するため削除は実行できないというメッセージが表示されます（この商品タイプには「STAR 筋肉鎮痛剤」という商品が属しています)。

アプリケーションの拡張

これまで見てきたように、定義した 2 つのオブジェクトを基にして、GeneXus によってさまざまなものが自動生成されました。

ここで、ドラッグストアから、商品ごとに画像を記録したいという要望が出されたとします。

これを実現するには、GeneXus に戻り、**Product** トランザクションに **ProductPhoto** という項目属性を入力する必要があります:

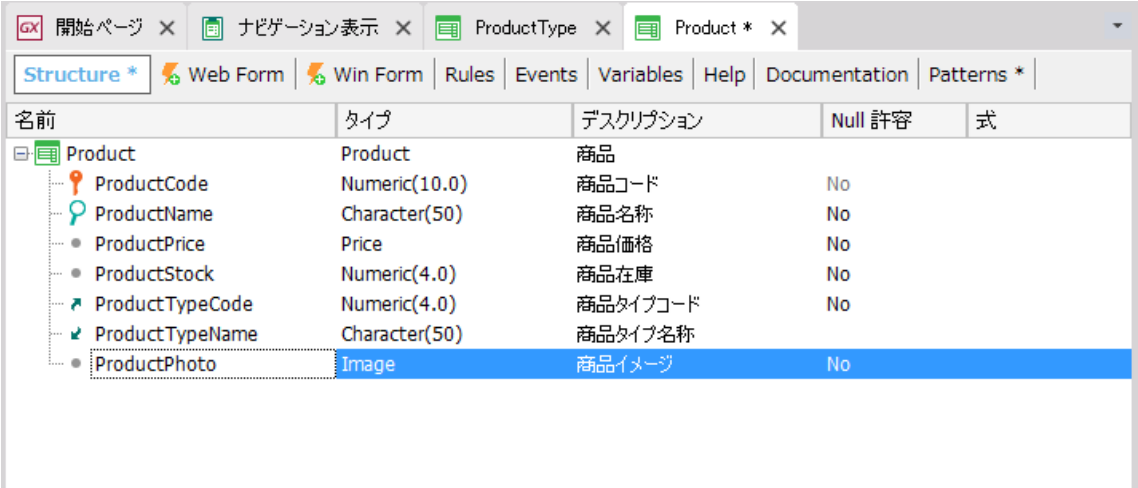
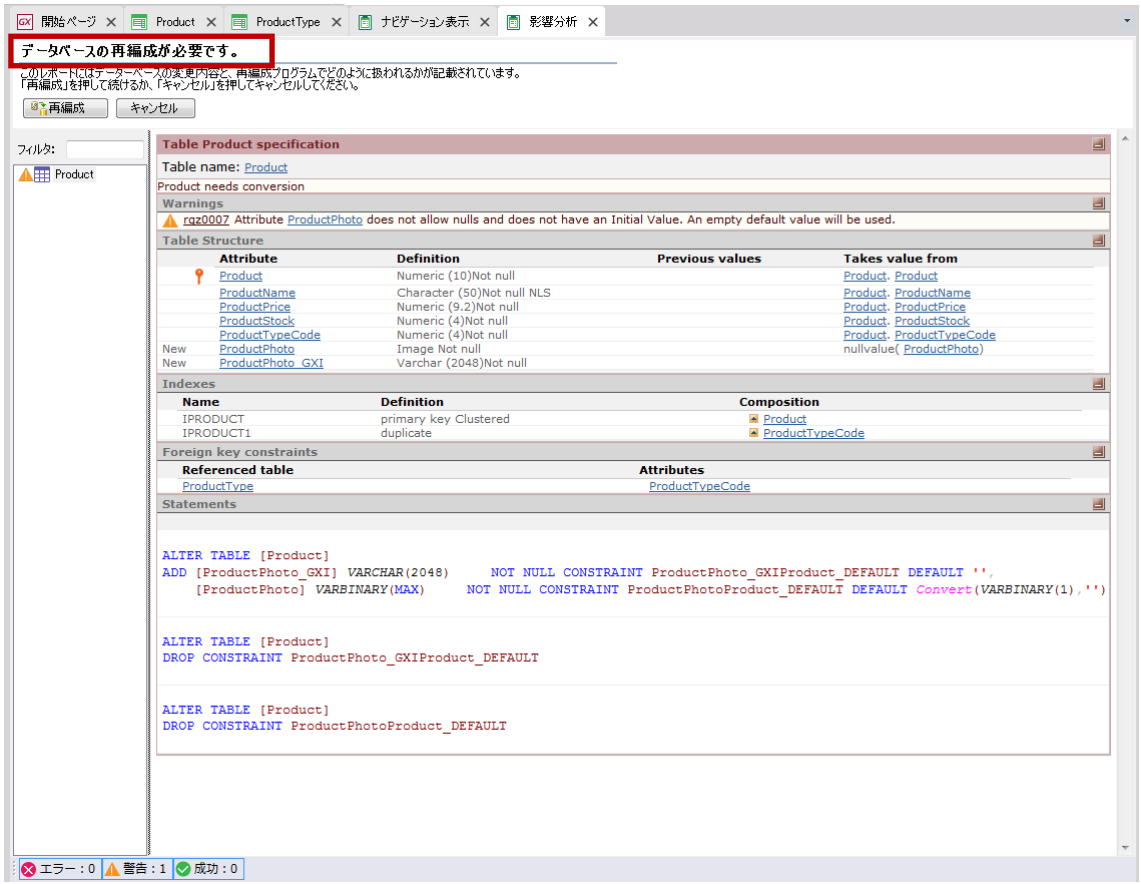


Image タイプでは画像を格納することができます。

Web フォームが自動更新され、**ProductPhoto** 項目属性が組み込まれます。

F5 キーを押すと、ナレッジベース内の新しい定義によって生じる影響が評価され、**[影響分析]** にレポートが表示されます:

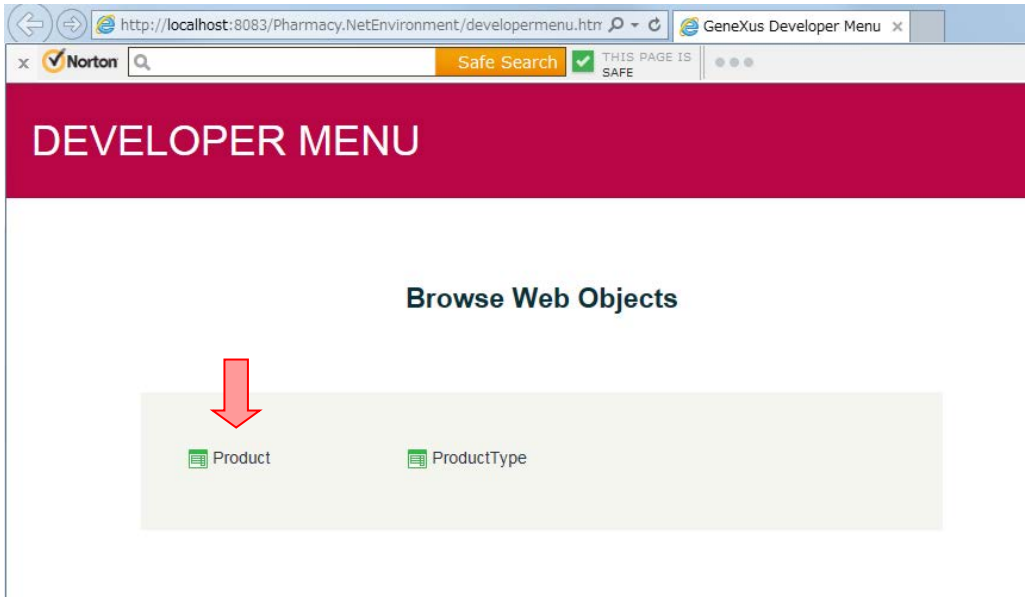


注意: 影響分析レポートには、データベースで行う必要がある構造変更が表示されます。

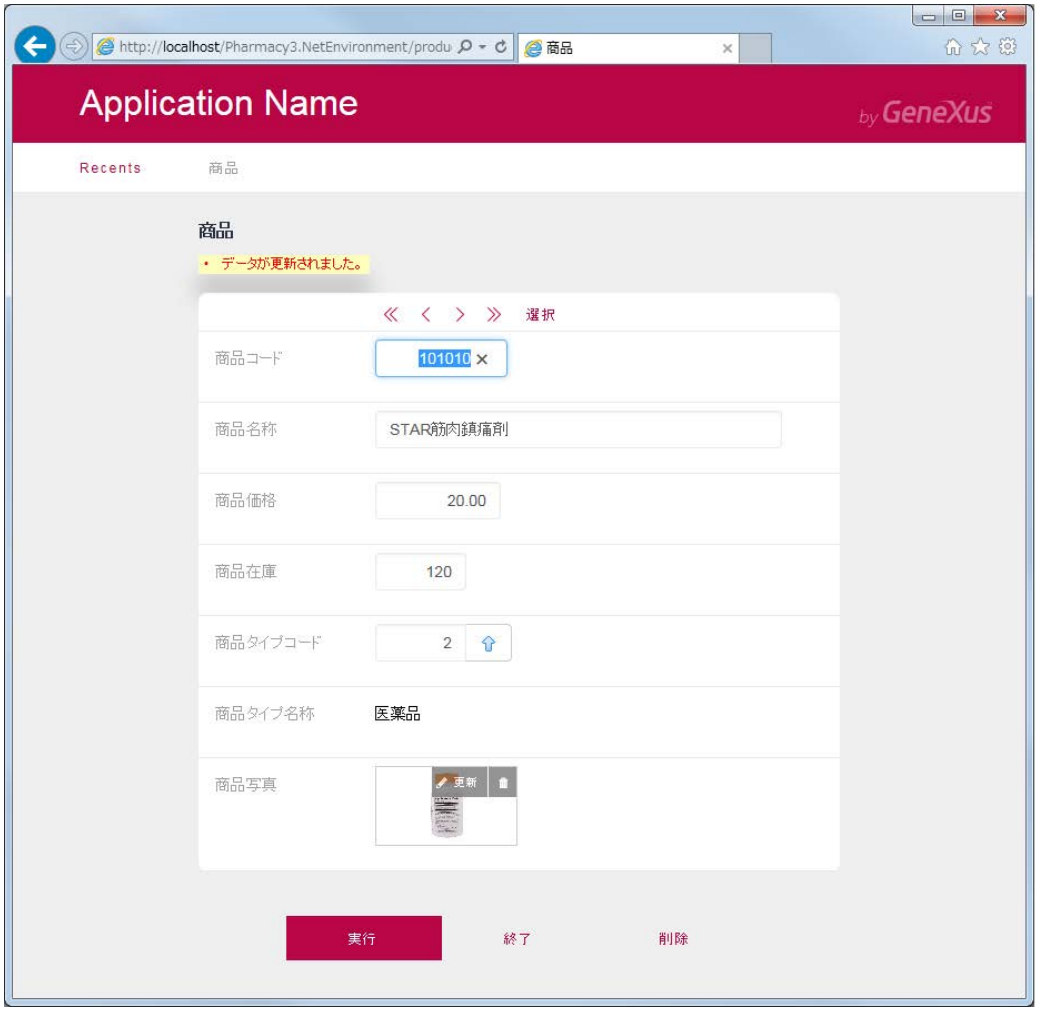
このレポートのメインタイトルから、今回は**「データベースの再編成が必要である」**ことが分かります。「再編成」という用語は、データベースに変更を加える作業を意味します。このレポートには、**Product** テーブルを更新する必要があると示されています。

[再編成] ボタンをクリックすると、データベースを変更するプログラムが作成され、実行されます。そして、アプリケーション自体に対応した必要なプログラムが生成されます。

続けて、再度 Web ブラウザーで、新しい定義が組み込まれたアプリケーションを実行します:



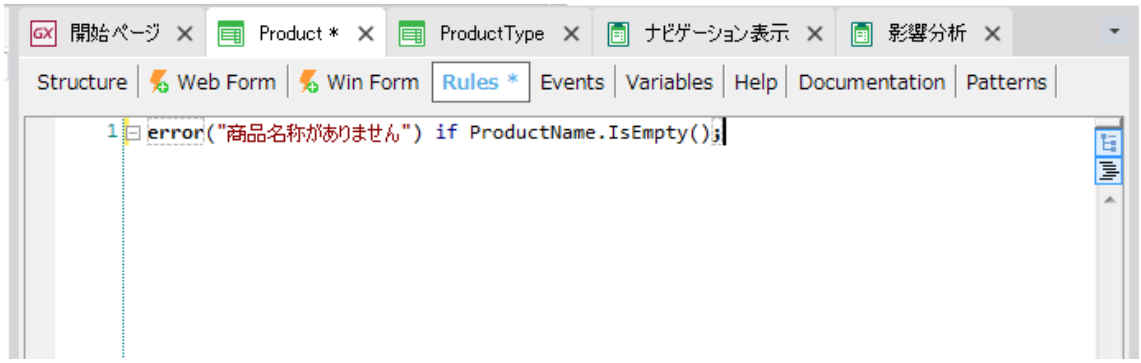
商品コードを覚えている場合は、それを入力します。別の方法として、[選択] ボタンをクリックしてリストから選択するやり方もあります。ここから「STAR 筋肉鎮痛剤」を取得し、写真をアップロードできます:



ビジネスルールの追加

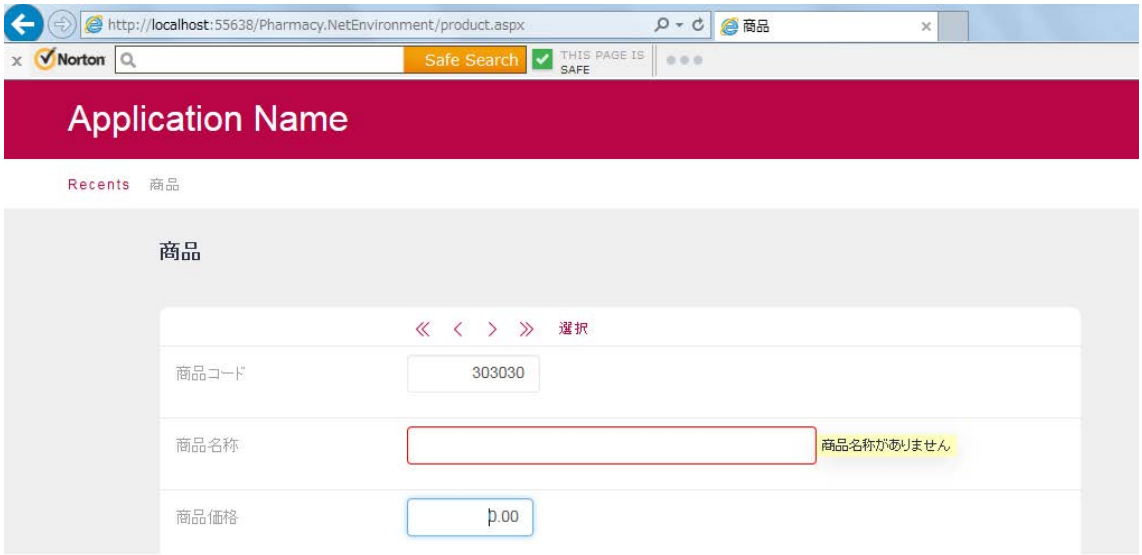
生成されるアプリケーションに GeneXus によって組み込まれる自動コントロールに加えて、ユーザーが特別なコントロールを要求することがあります。トランザクションでは、順守しなければならないルールや検証が必要なコントロールは **[ルール]** セクションで定義します。

たとえば、名前を入力せずに商品を格納できないようにするという要件がある場合は、未入力を回避できる **Error** というルールを使用します:



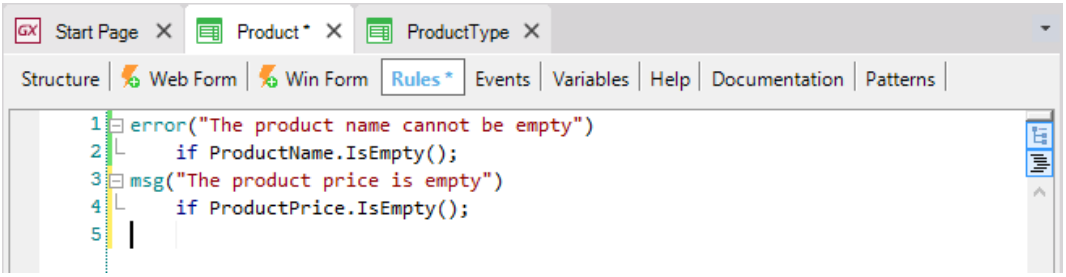
F5 キーを押すと、ナレッジベース内に新しい定義が保存され、評価されます。今回は、データベースの変更は必要ないと推論されたため、影響分析レポートは表示されません。GeneXus によって必要なコードが生成され、その後、新しい定義で更新されたアプリケーションが実行されます。

Product トランザクションを実行します。商品名称がブランクのままになっていると、定義したルールが実行されます:



Error ルールとよく似た構文の別のルールもあります。それは **Msg** というルールです。唯一の違いは、条件を満たした場合、メッセージは通知または警告として表示され、ユーザーは作業を続行できるという点です。

たとえば、商品価格がブランクのままになっていることをユーザーに通知し、入力は強制しない場合は、**Product** トランザクションに次のルールを追加します:



このルールセットは任意の順序で記述でき、実行時の結果はまったく同じになります。これは、定義された各ルールをトリガーするタイミングは GeneXus が判断するためです (たとえば、関係のあるフィールドからフォーカスが移動したときなど)。

言うまでもなく、GeneXus には、さまざまな検証やアクションを定義するための有用なルールが用意されています。

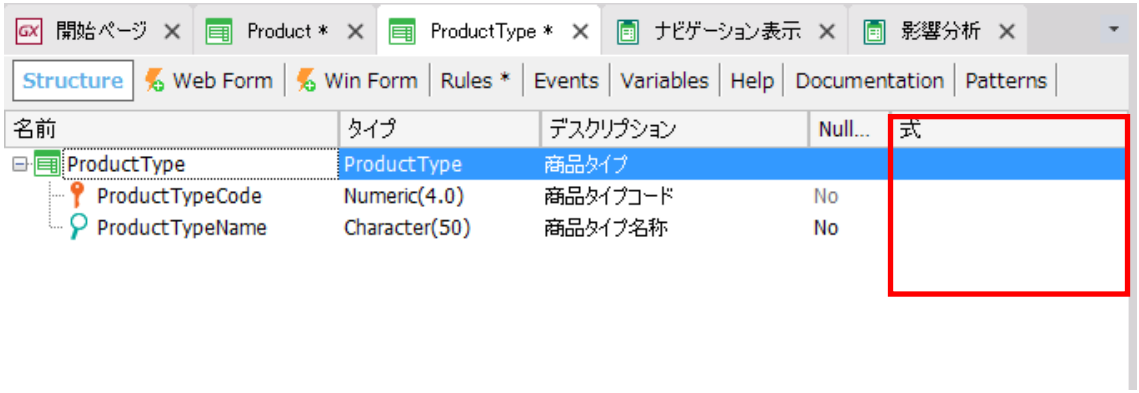
トランザクションごとに独自の動作ルールを定義する必要がある場合もあります。

計算の定義：式

アプリケーションでは、特定の項目属性、定数、あるいは関数の値が関与する計算が必要になることが多々あります。これに対応するために、GeneXus には**式**が用意されています。

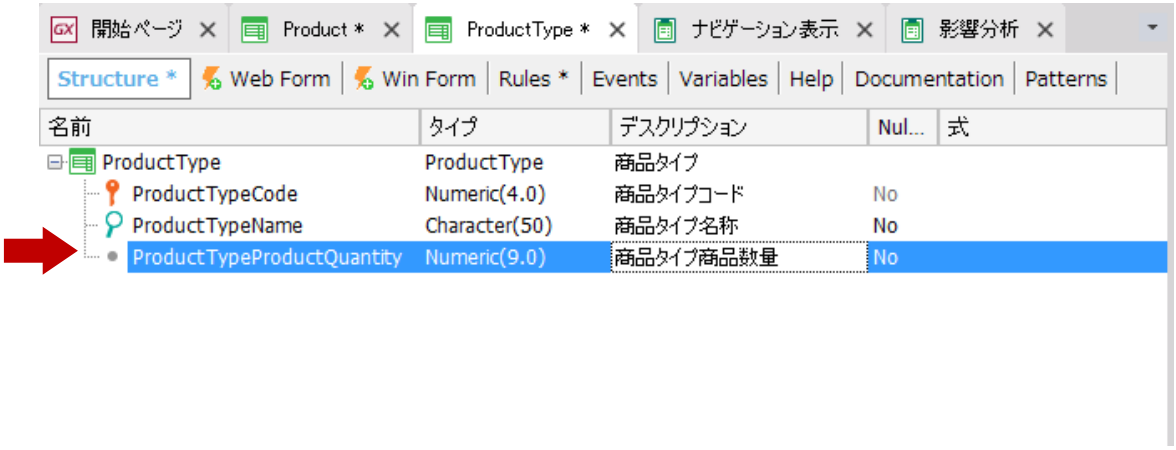
式は、さまざまな方法で定義できます。

まず、**グローバル式**について説明します。グローバル式とは、開発者が項目属性に関連付けて定義する計算式です。トランザクション構造には **[式]** というラベルの付いた列があります:



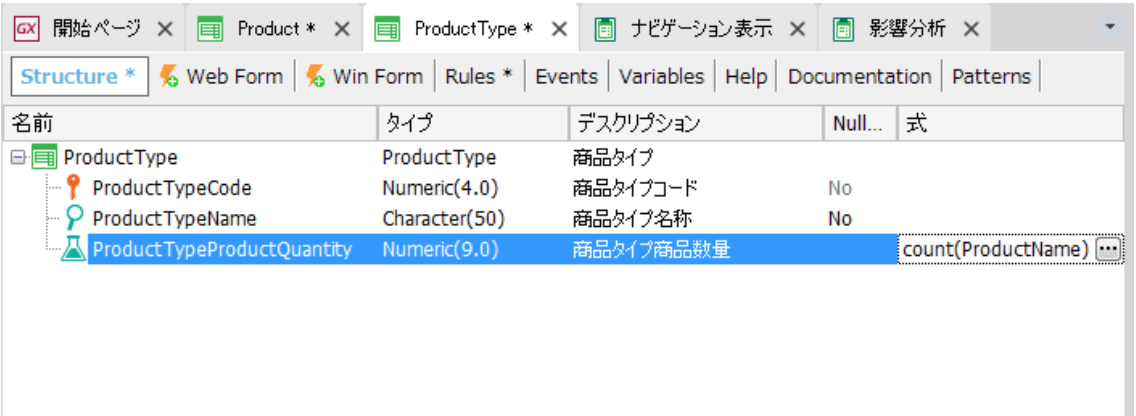
項目属性のこの列に計算式が定義されている場合、その項目属性は仮想項目属性になります。つまり、テーブルのフィールドとして物理的に作成されないということです。これは、この項目属性の値が、計算で必要になるたびに取得されるためです。

例を挙げて説明します。ドラッグストアでは、常に、各商品タイプに登録されている商品の数を把握する必要があるとします。そこで、**ProductType** トランザクションに新しい項目属性を定義し、グローバル式を定義します:



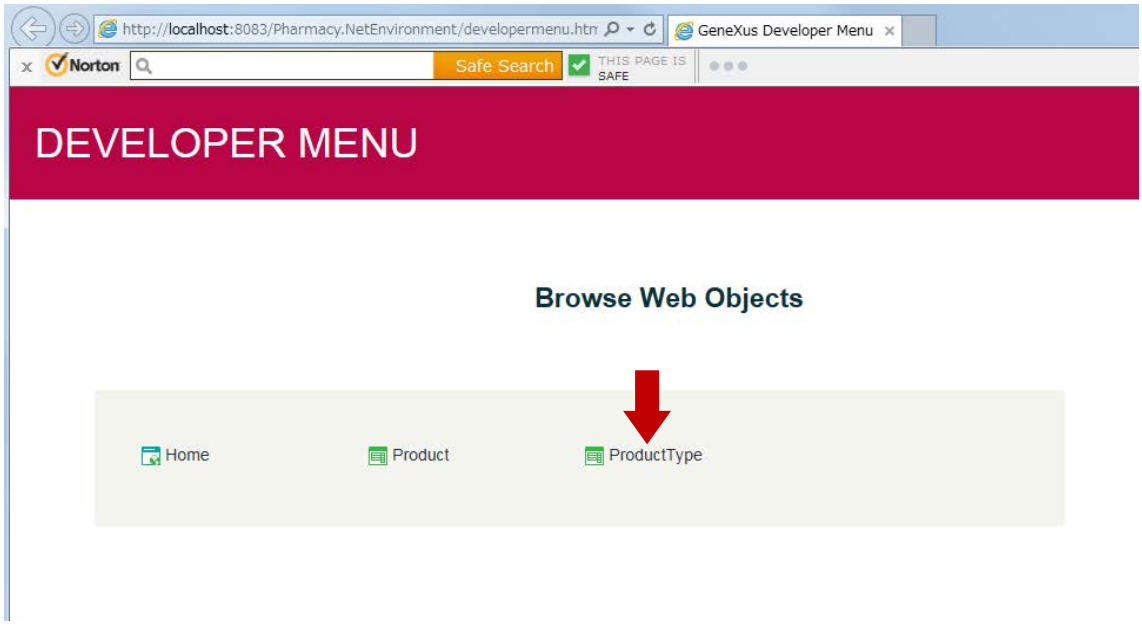
ProductTypeProductQuantity 項目属性に関連付ける計算式を定義します。

GeneXus には、ドラッグストアの要望に対応できる Count という式があります（Sum や Average などほかにも数多くの式があります)。



式の括弧内で参照される項目属性により、計算のために参照すべきテーブルが分かります（上記の定義では、Product テーブルでカウントする必要があることが分かります)。参照先のテーブル（Product）と式の項目属性が定義されているコンテキスト（ProductType）の間にリレーションが検出された場合は、計算に関係のあるレコードのみ考慮されます。この例では、**ProductTypeId** が両方のコンテキストに存在します。すなわち、式が定義されているコンテキストと、式の計算のために参照されるテーブルです。したがって、参照先テーブルに記録されているすべての商品が考慮されるのではなく、各商品タイプの商品のみカウントされます。リレーションが見つからない場合は、参照先テーブルのすべてのレコードを考慮して計算が実行されます。

F5 キーを押します。データベースが物理的に変更されないことを確認できます。GeneXus でいくつかのプログラムのみ生成され、再度開発者メニューが実行されます:



ProductType トランザクションを実行し、その時点での各商品タイプの商品の数量がどのように計算されるか確認します:

商品をさらに追加し、その時点での各商品タイプの商品の数量がどのように計算されるか検証します。

パターンの使用 (WEB 用およびモバイルデバイス用)

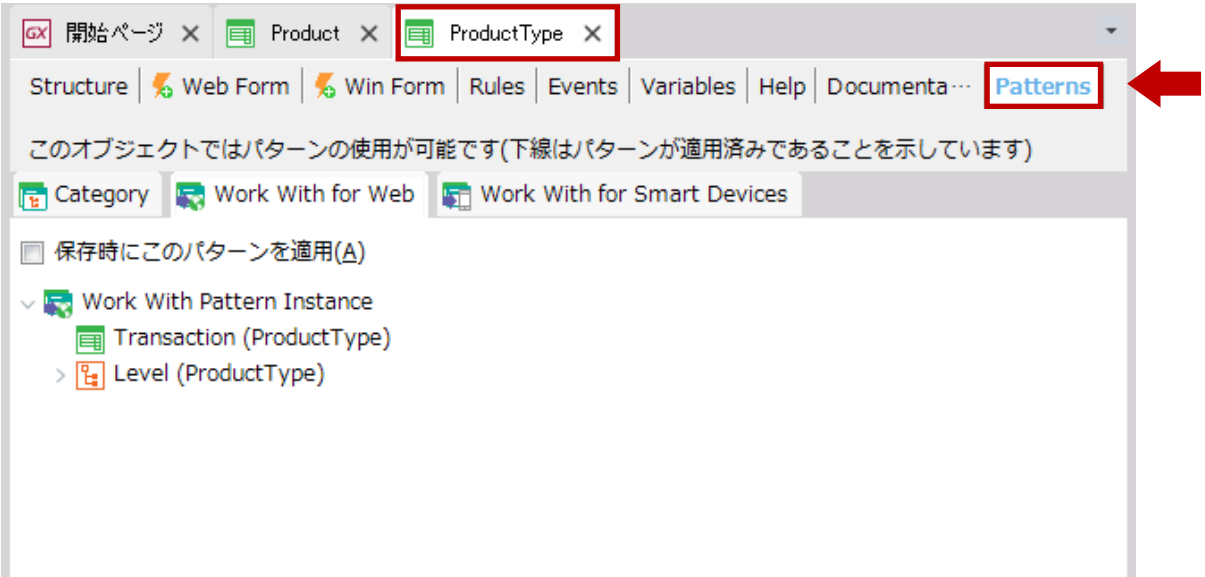
モバイルデバイスを使用する場合は、事前に次のページをお読みください。

http://wiki.genexus.jp/hwikibypageid.aspx?14449

パターンを使用することで、アプリケーションを自動的にさらに強化できます。

パターンの適用はきわめて簡単です。パターンを適用すると、GeneXus によって直ちにオブジェクト、コード、設定が作成され、開発者がプログラミングすることなく便利な動作を実現できます。

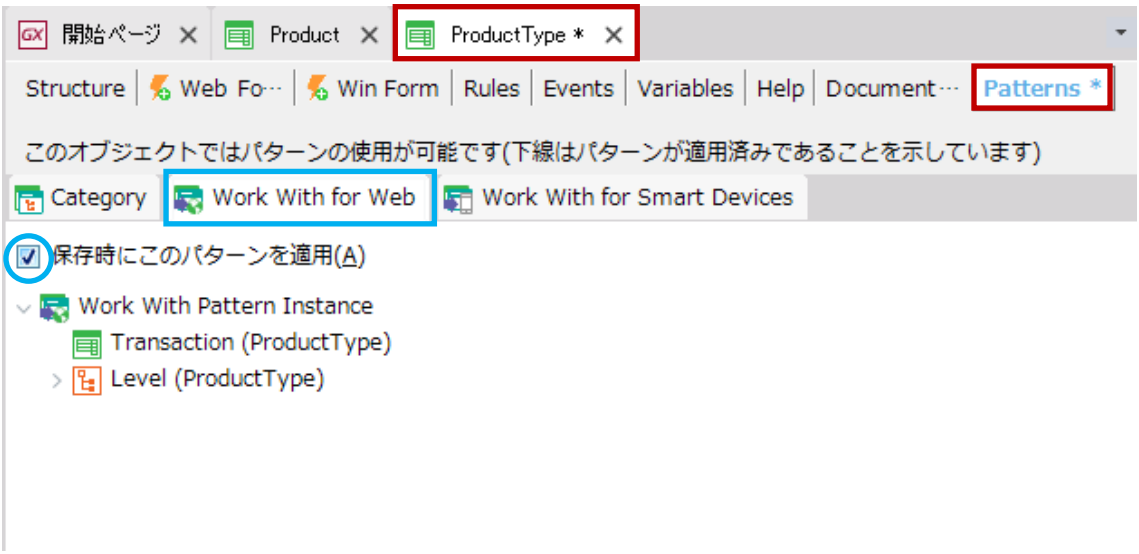
トランザクションの [パターン] セクションについて説明します。例として、**ProductType** トランザクションの [パターン] セクションを選択してみます:



3 つのタブがあり、各タブでは同じトランザクションに適用できる異なるパターンが提供されています。

パターンの適用によって、簡単で役立つ結果をすばやく得られることを確認するために、まず **[Work With for Web]** タブを選択します。

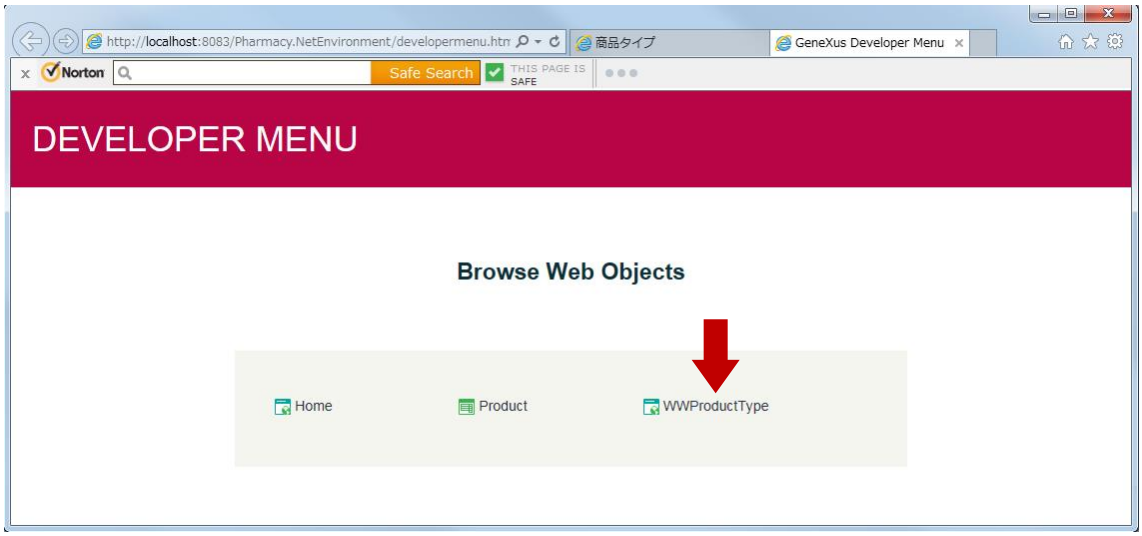
[保存時にこのパターンを適用] チェックボックスをオンにして保存 ( ) するだけです:



[KB エクスプローラー] で **ProductType** トランザクションを確認すると、このトランザクションの下にいくつかのオブジェクトがあることが分かります:

これらのオブジェクトは、**Work With for Web** パターンが適用されたときに GeneXus によって作成されたものです。

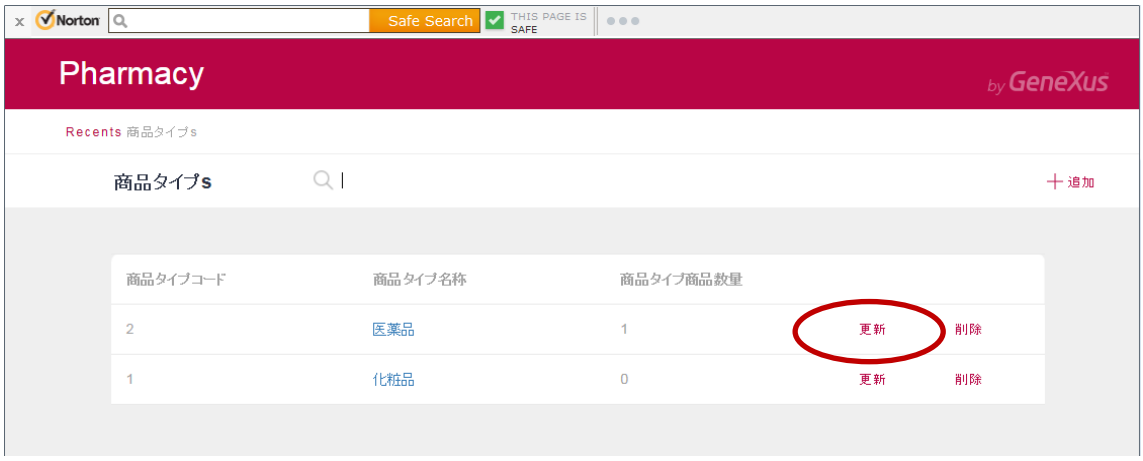
F5 キーを押し、実行時の結果を表示します:

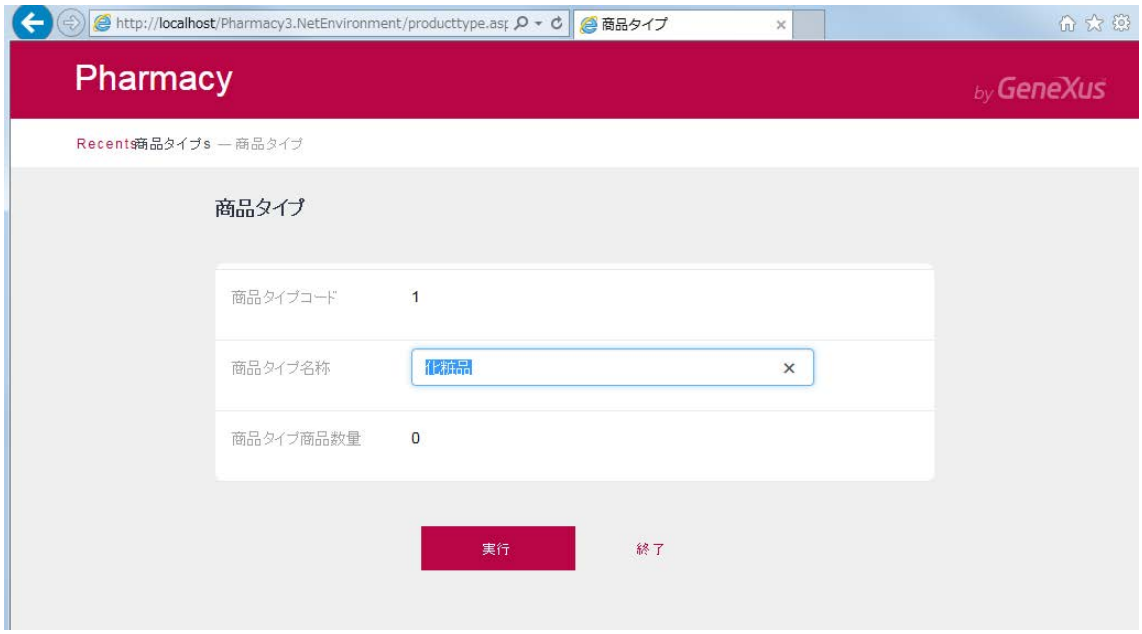


表示されている最後のリンクを見てください。「WWProductType」が提供されています。ここから **ProductType** トランザクションが呼び出されます。このリンクをクリックします。

ページが開き、格納されている商品タイプがすべて表示されます。ユーザーは、このページで、さまざまな機能を使用して商品タイプの処理を行うことができます。

たとえば、最初の行の [更新] をクリックしてみます:





ProductType トランザクションが開き、この行の商品タイプの詳細を編集できるようになります。タイプ名称を編集して実行します:

編集および実行後、アプリケーションは **WWProductType** ページに戻ります:

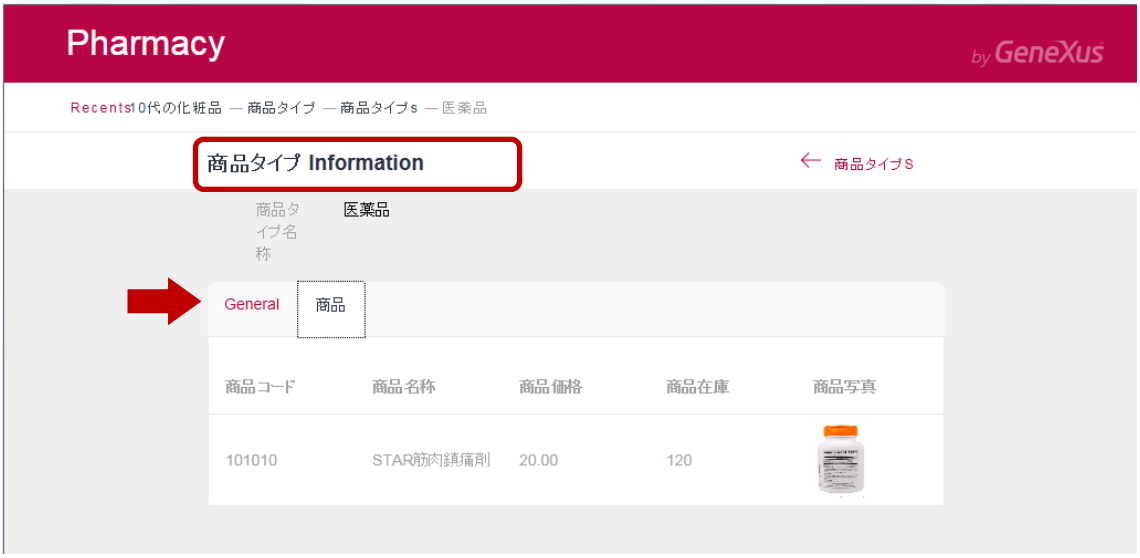
[削除] リンクでは、ユーザーがその行の商品タイプを削除できます。

グリッドの外側にある [追加] ボタンでは、ユーザーが新しい商品タイプを追加できます。このボタンをクリックすると、**ProductType** トランザクションが開き、新しい商品タイプを追加できるようになります。このボタンをクリックして新しい商品タイプを入力します (前述のとおり、キー項目属性の [Autonumber] プロパティを [True] に設定してあるため、入力するのは商品タイプ名称だけです):

挿入後もアプリケーションは **WWProductType** ページに戻ります:

各商品タイプ名称にはリンクが設定されています。次の商品タイプをクリックします: 医薬品。

次のように、最初のタブには、選択した商品タイプのすべての詳細が表示され、次のタブには、その商品タイプに属している商品のリストが表示されます。



各商品タイプには関連する商品がいくつか存在するため、**[商品] タブ**が自動的に生成されました。各商品タイプにほかの種類の関連データも存在する場合は、商品タイプの各関連データのリストが表示されるように、さらにタブが生成されます。

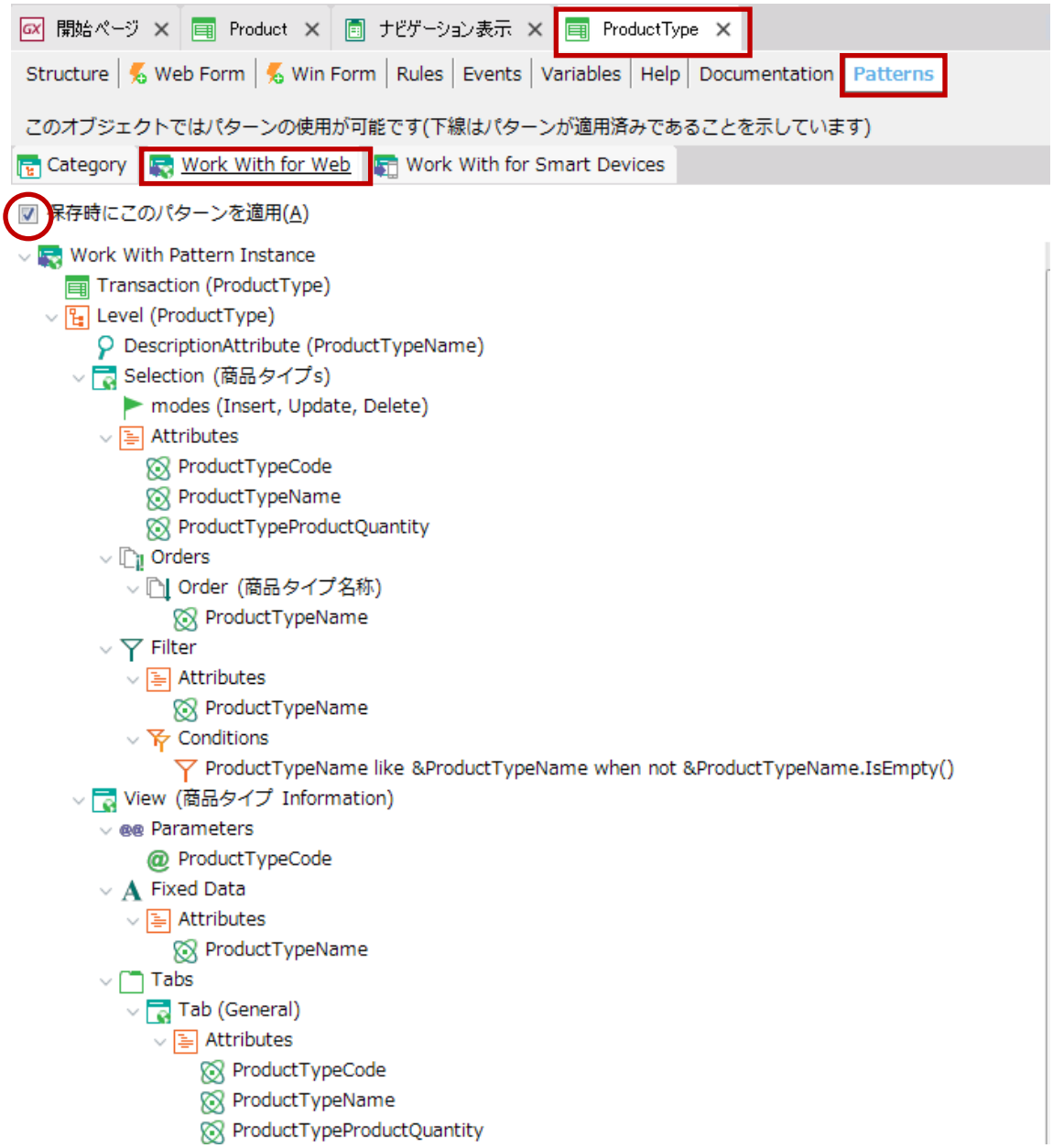
ページの左上にある **[Recents]** セクションのリンクをクリックして **WWProductType** ページに戻ります。

名前で検索することもできます。たとえば、「C」と入力すると、この文字で始まる商品タイプのみ表示されます:

GeneXus に戻ります。

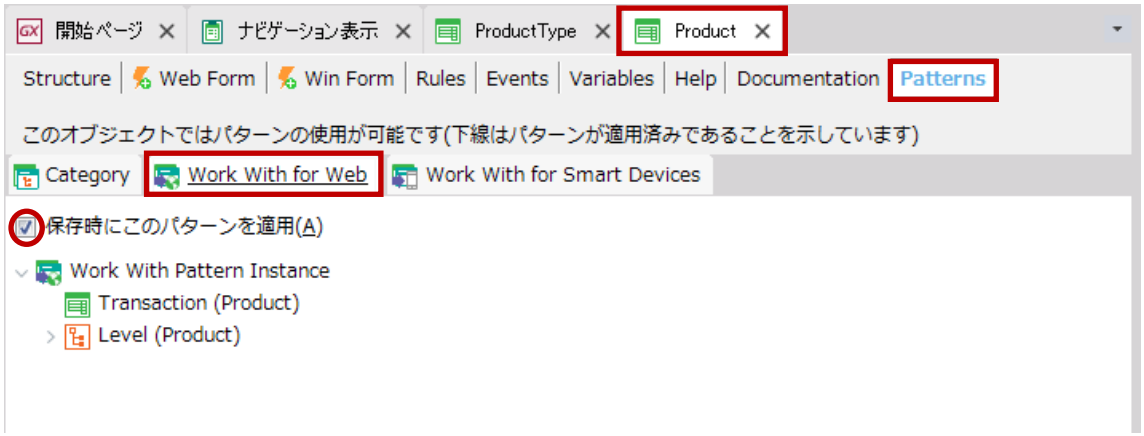
これまでの手順では、**ProductType** トランザクションの **[Work With for Web]** タブにある **[保存時にこのパターンを適用]** だけを選択しました。保存後は、自動的に生成された機能がすべて表示されました。

このツリーで設定ができることに気付かなかったかもしれません:



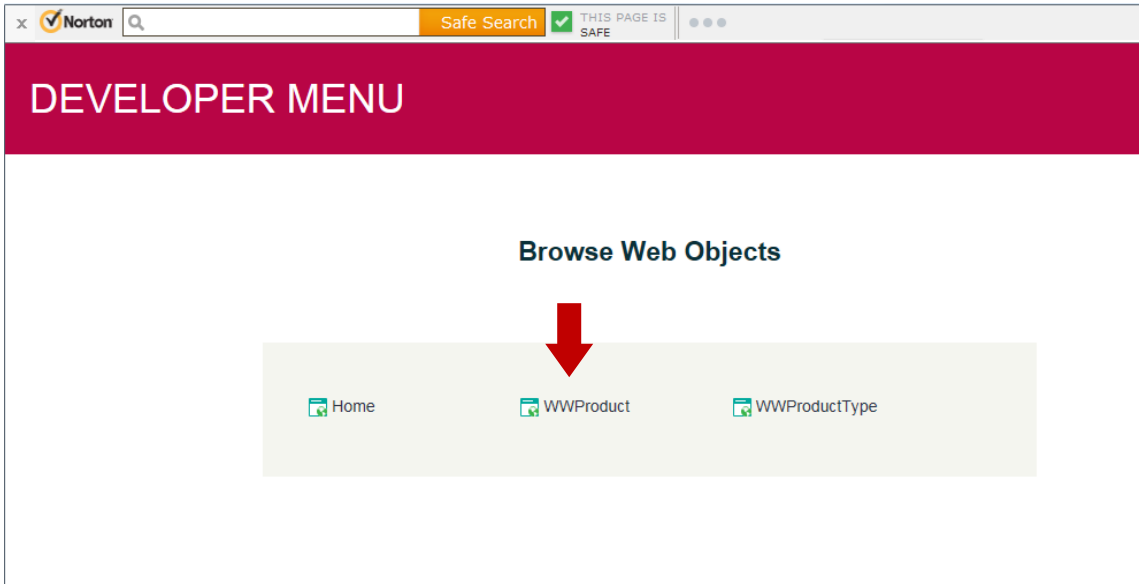
ここには設定可能なノード、サブノード、エレメントが含まれており、生成する動作をカスタマイズできるようになっています (たとえば検索条件の変更など)。

Product トランザクションにも **Work With for Web** パターンを適用します。前述のとおり、**Product** トランザクションを開き、その [パターン] セクションで **[Work With for Web]** タブを選択します。そして、**[保存時にこのパターンを適用]** オプションをオンにして保存します:



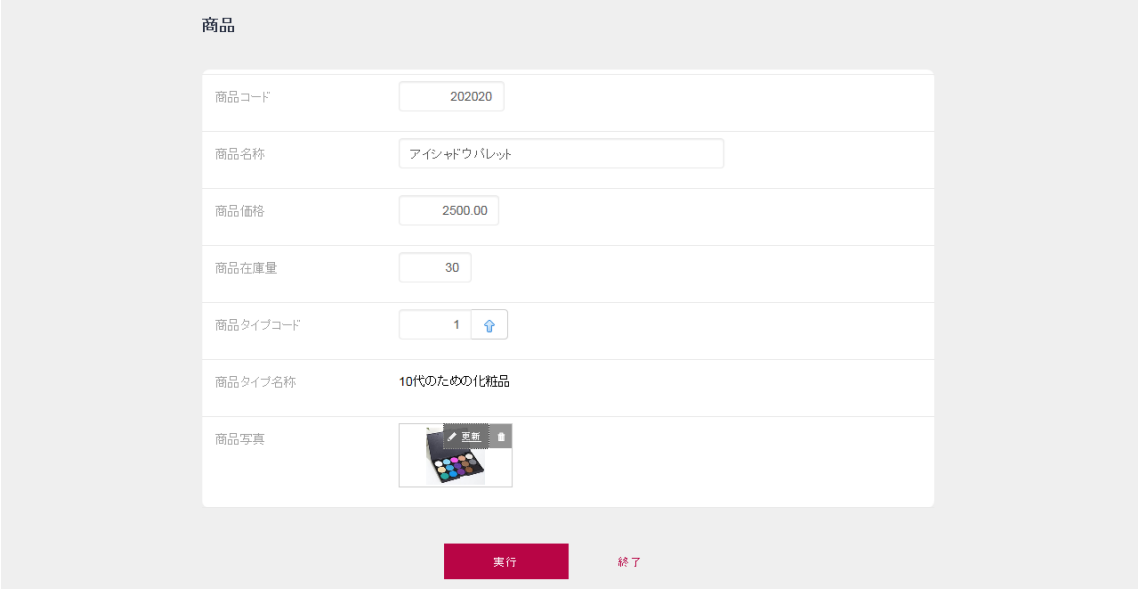
F5 キーを押します。GeneXus によって、必要なプログラムが生成され、変更されたアプリケーションが実行されます。

WWProduct ページを実行します:



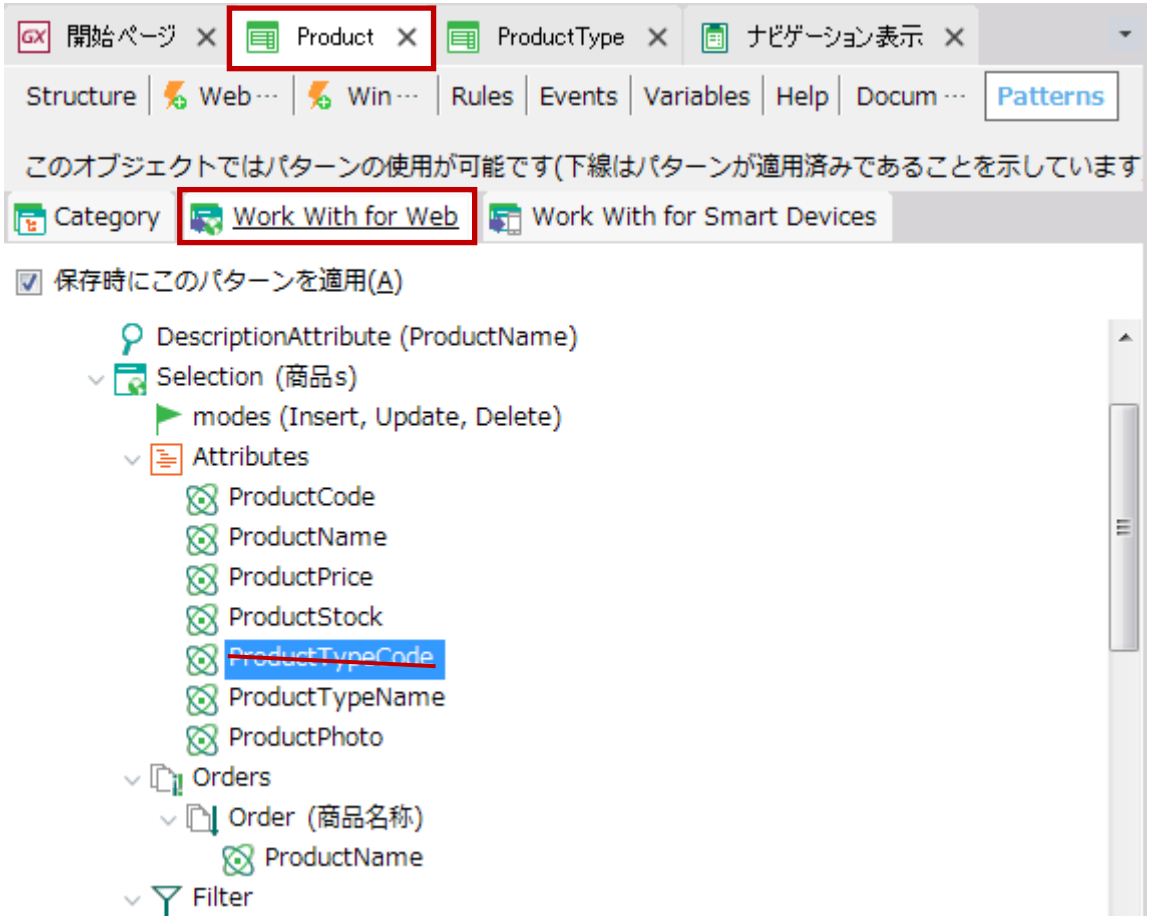
WWProductType ページで確認したものと同じクエリ機能が表示されます:

新しい商品を挿入します:



実行後、アプリケーションは **WWProduct** ページに戻ります:

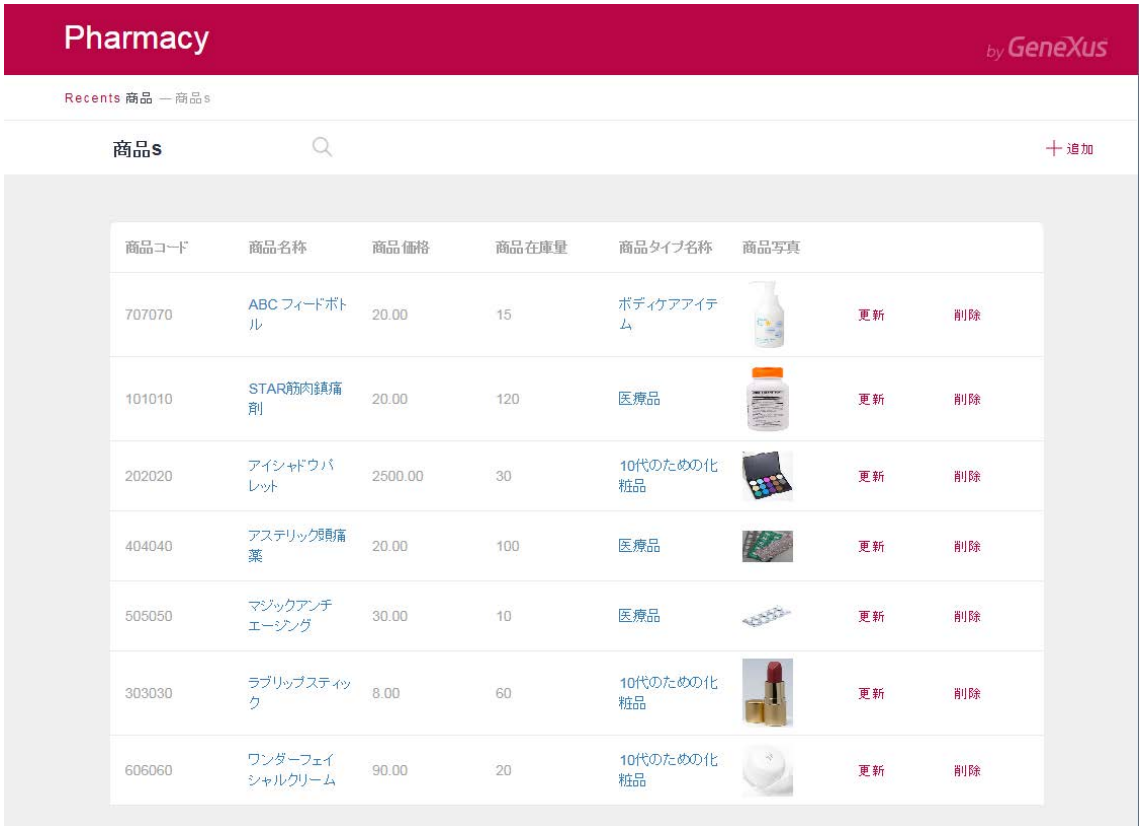
グリッドに商品タイプコードを表示する必要はないため、この **Work With for Web** の生成時に考慮される設定ツリーから削除します:



F5 キーを押します。GeneXus によって保存され、必要なプログラムのみ生成されて、変更されたアプリケーションが実行されます:

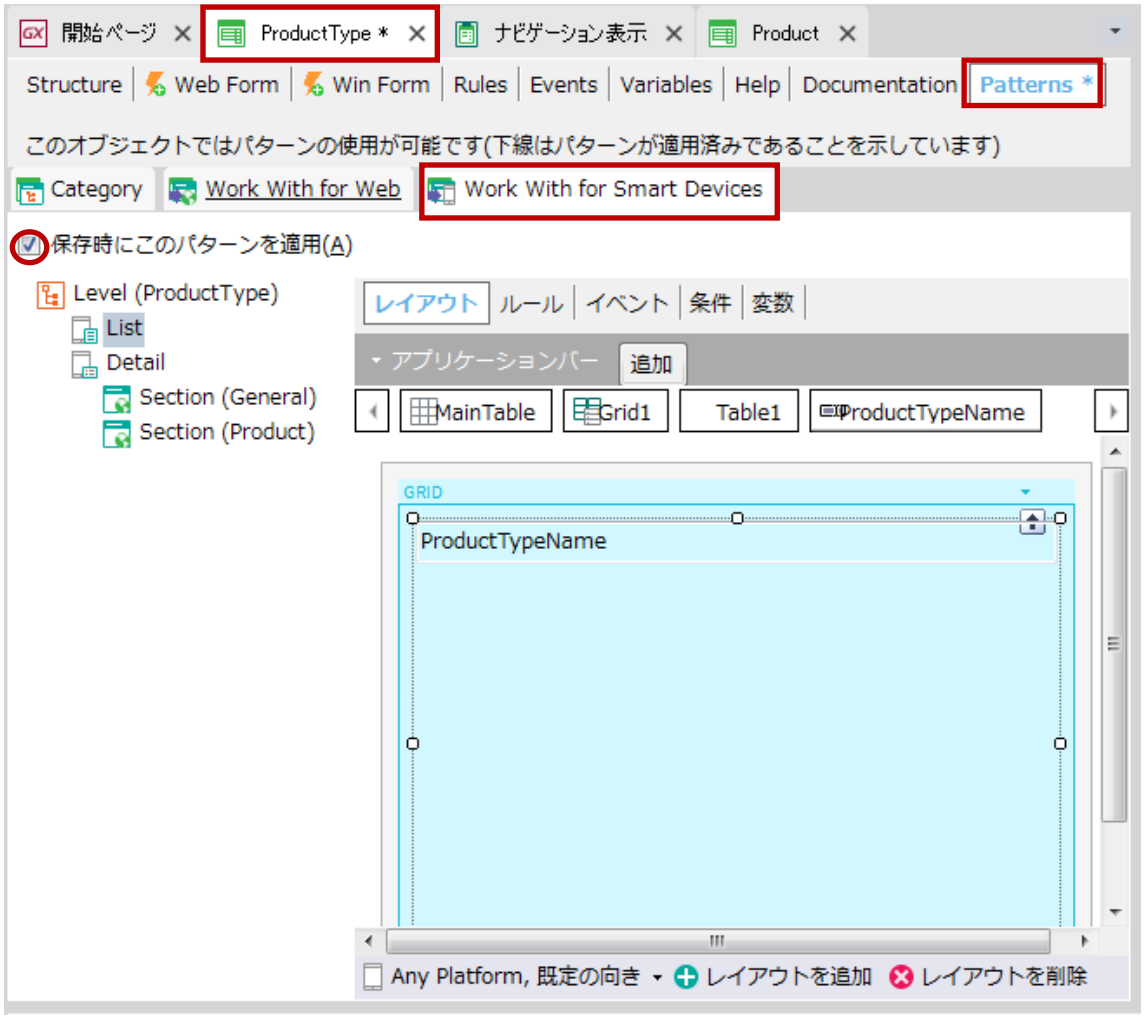
前述と同じ方法で、連続していくつかの商品を挿入します（[追加] ボタンを押して Product トランザクションを呼び出します)。

次の画像は、追加したすべての商品が一覧表示された商品 Work With です:



次に、各トランザクションに用意されている **[Work With for Smart Devices]** タブについて説明します。

このパターンを **ProductType** トランザクションに適用します:



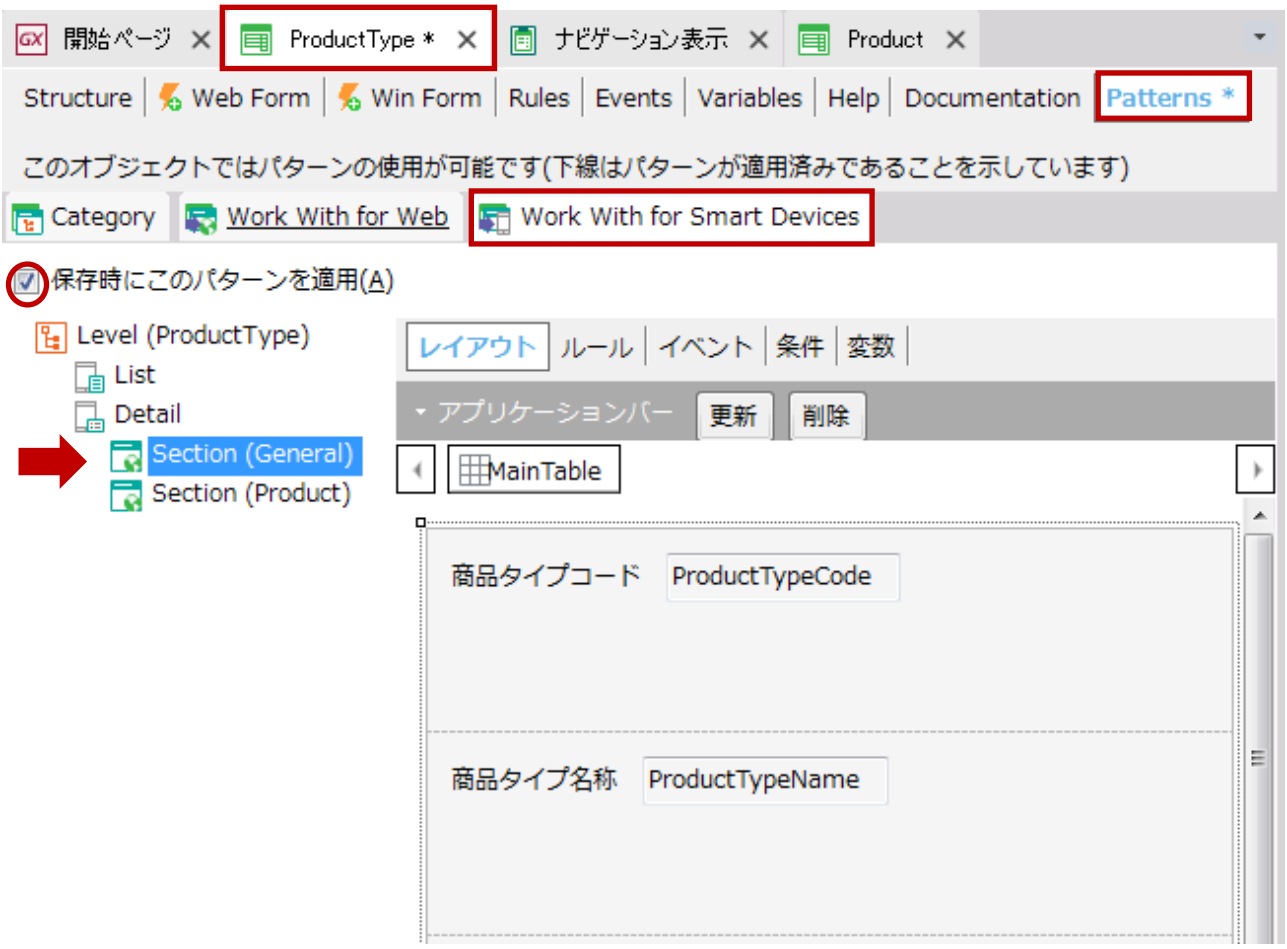
メインノード (**ProductType**) の下に **[List]** ノードがあります。これをクリックすると、右側のウィンドウに、**ProductTypeName** 項目属性が挿入されたグリッドが表示されます。

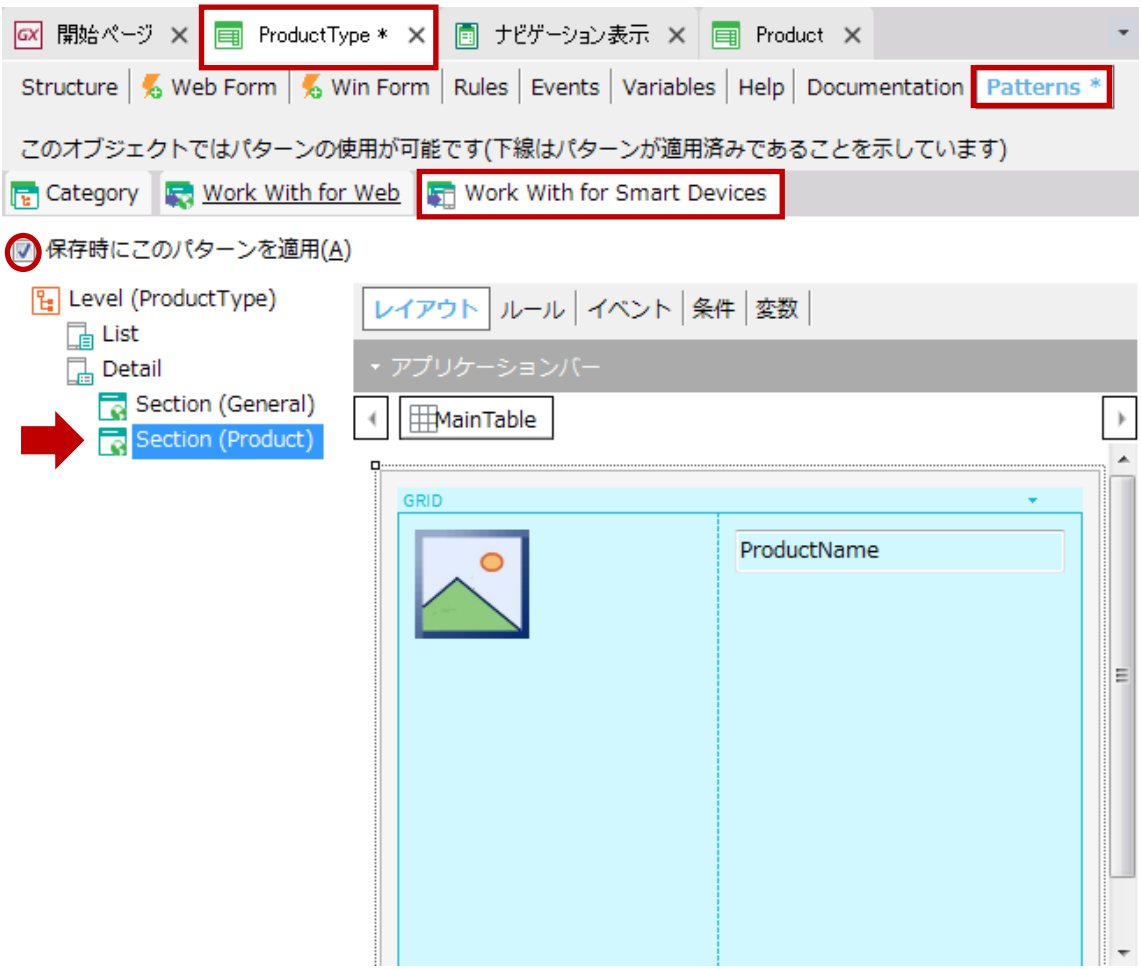
Work With for Web パターンとは異なり、グリッドに含まれる項目属性のリストがノードの下に表示されるのではなく、レイアウトそのものが表示されます。

[Detail] ノードについて説明します。Detail という言葉から、リストの特定の行の詳細を表示することだと想像できます。

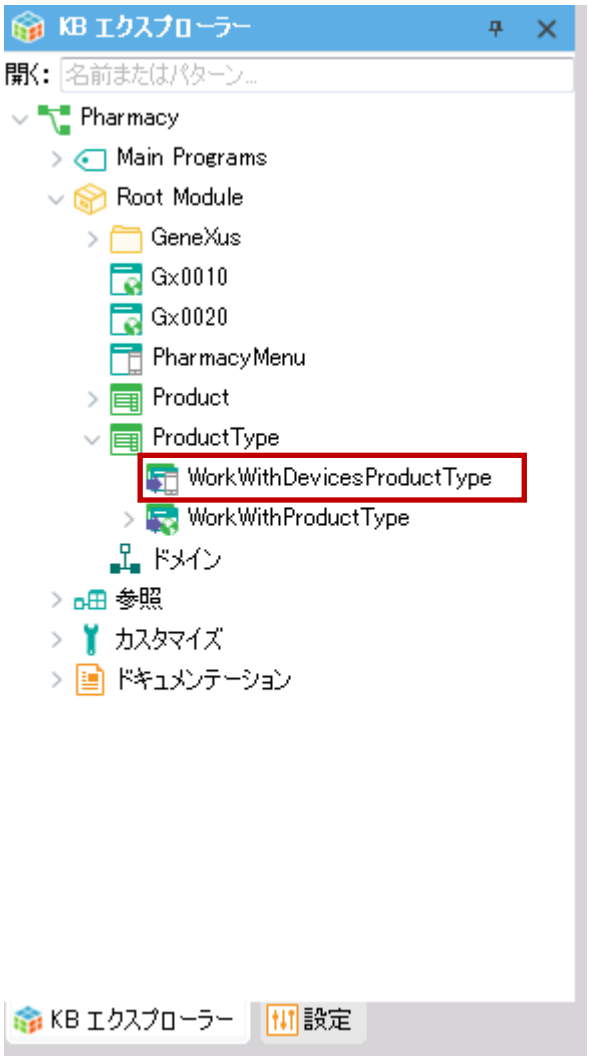
[Detail] ノードは次の 2 つのセクションで構成されています: **General** と **Product**。

Work With for Web パターンで実装された機能と同じように、**General** セクションには、選択された商品タイプに関するデータが表示され、**Product** セクションでは、その商品タイプに属するすべての商品がグリッド内に表示されます。





このパターンを適用して保存してから、**[KB エクスプローラー]** の **ProductType** トランザクションを見ると、**ProductType** トランザクションの下に **WorkWithDevicesProductType** という新しいオブジェクトが生成されているのがわかります:



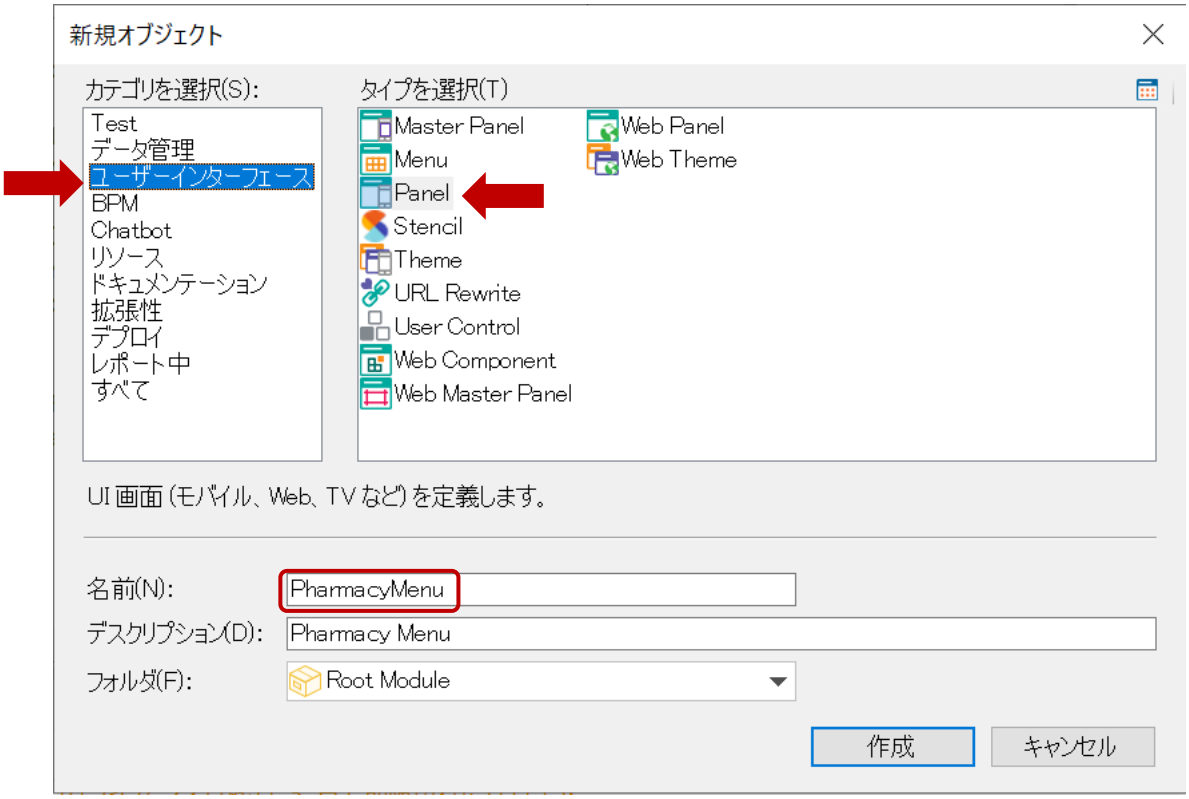
ProductType トランザクションの下に生成されている２つのオブジェクトを見ると、**WorkWithDevicesProductType** の下にはほかのオブジェクトがないことに気が付きます（これは、さまざまなセクションを組み込んで実装全体を定義しているためです)。

これに対して、**WorkWithProductType** は前述のとおり設定可能なインスタンスです。したがって、開発者がこのインスタンスオブジェクトを設定して保存でき、GeneXus がその下にほかのオブジェクトを生成して便利な動作を実現します。

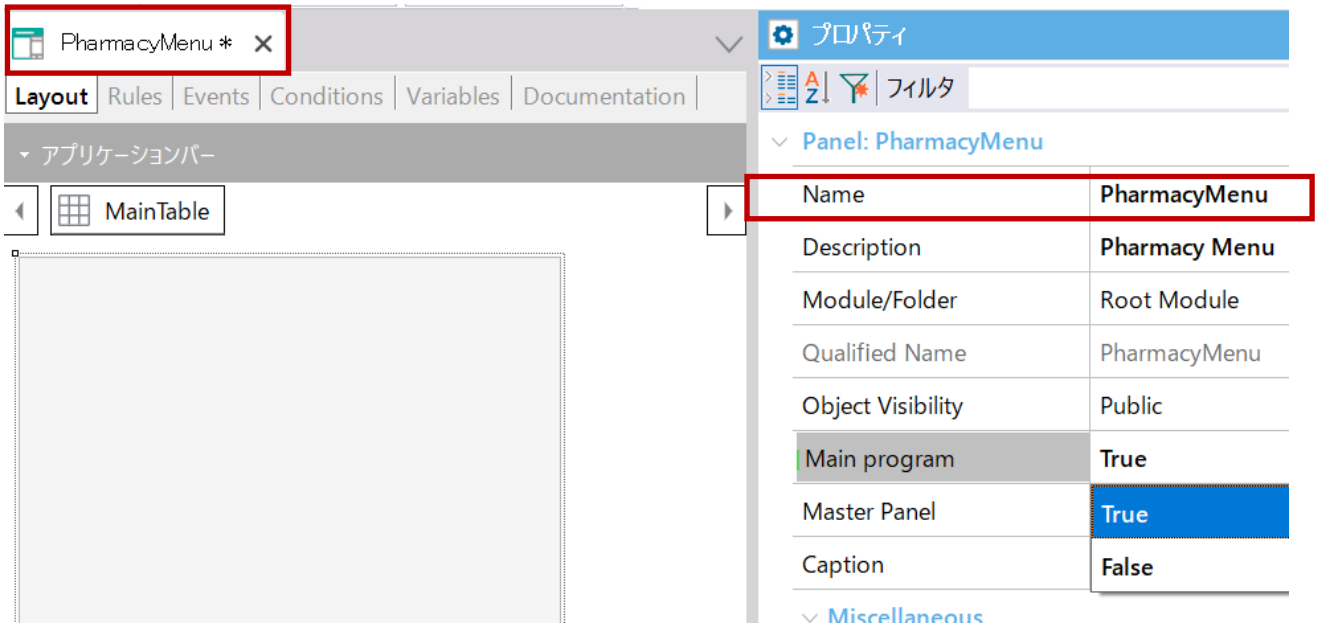
WorkWithDevicesProductType の実際の動作はすぐに確認できます。

次に、アプリケーションの最初の実行オブジェクトとなるように、**Panel** オブジェクトを作成します。このオブジェクトでは画像が表示されます。ユーザーがその画像をタップすると、**WorkWithDevicesProductType** オブジェクトが実行されます。

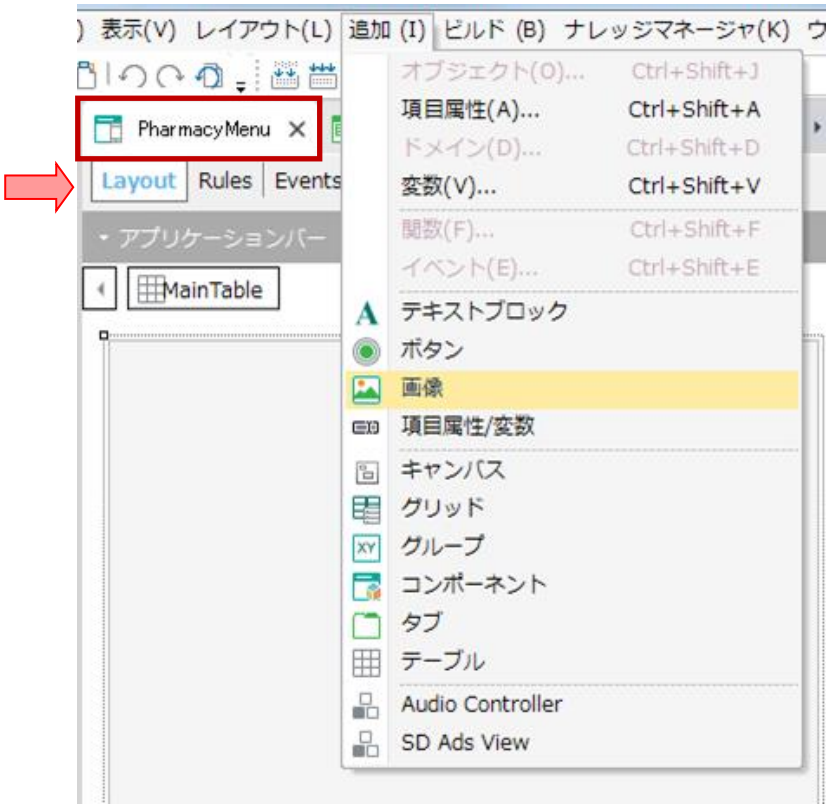
前述のとおり、オブジェクトを作成するには、ツールバーで **[ファイル]** → **[新規]** → **[オブジェクト]** を選択します。次のようなウィンドウが開くので、カテゴリとして [ユーザーインターフェース] を選択する必要があります。GeneXus に用意されている、このカテゴリに属するオブジェクトが表示されます:



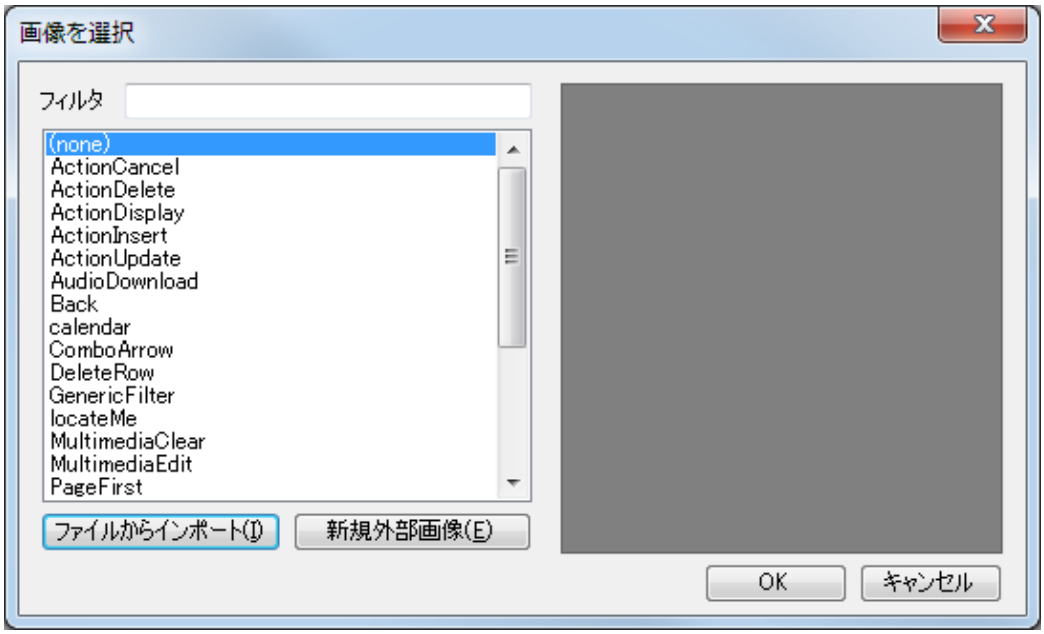
Panel オブジェクト（この例での名前は **PharmacyMenu**）を作成したら、**[Main program] プロパティ = [True]** と設定する必要があります。これにより、このオブジェクトが開発者メニューから独立した実行可能オブジェクトになります（つまり、単独でコンパイル可能かつ実行可能となります)。



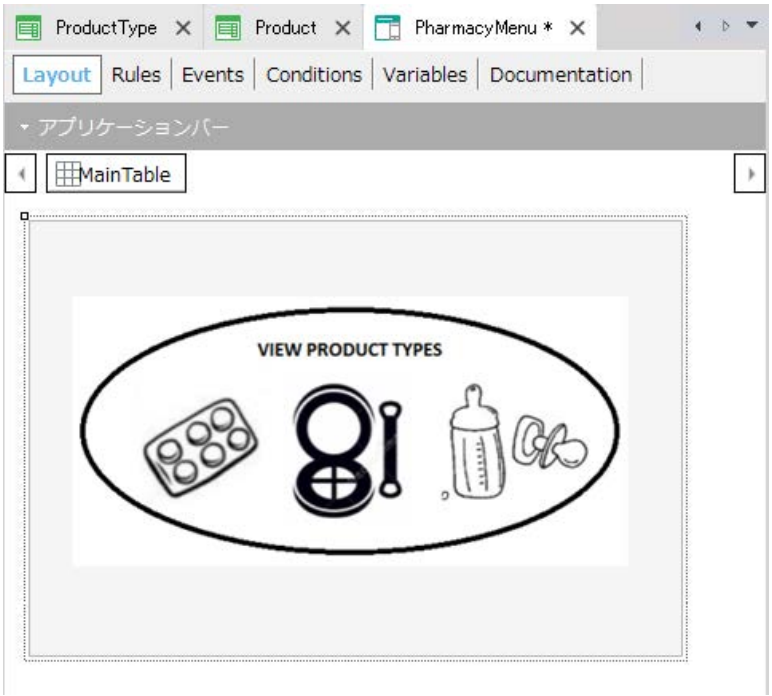
[レイアウト] 内に画像コントロールを挿入します:



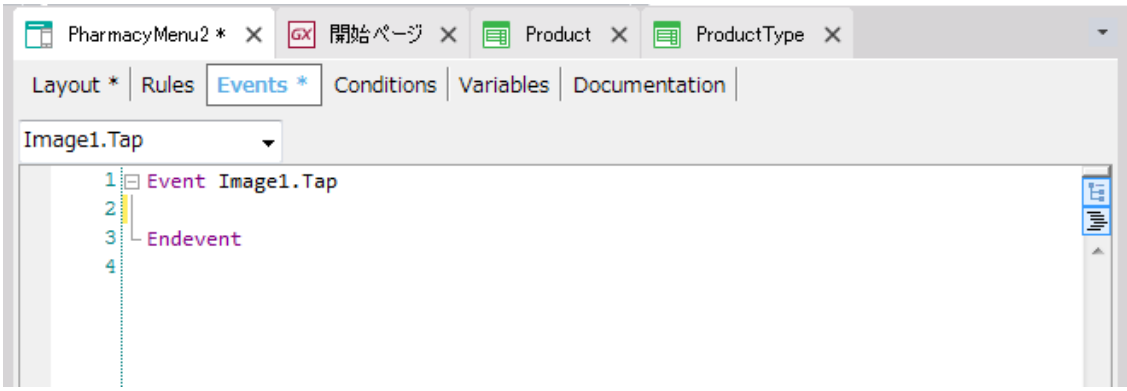
[追加] → **[画像]** を選択すると、次のようなウィンドウが表示されます。写真がファイルに格納されている場合は、**[ファイルからインポート]** ボタンを押す必要があります。



画像をナレッジベースにインポートし、それを **PharmacyMenu** の **Panel** の［レイアウト］に挿入すると、レイアウトは次のようになります:



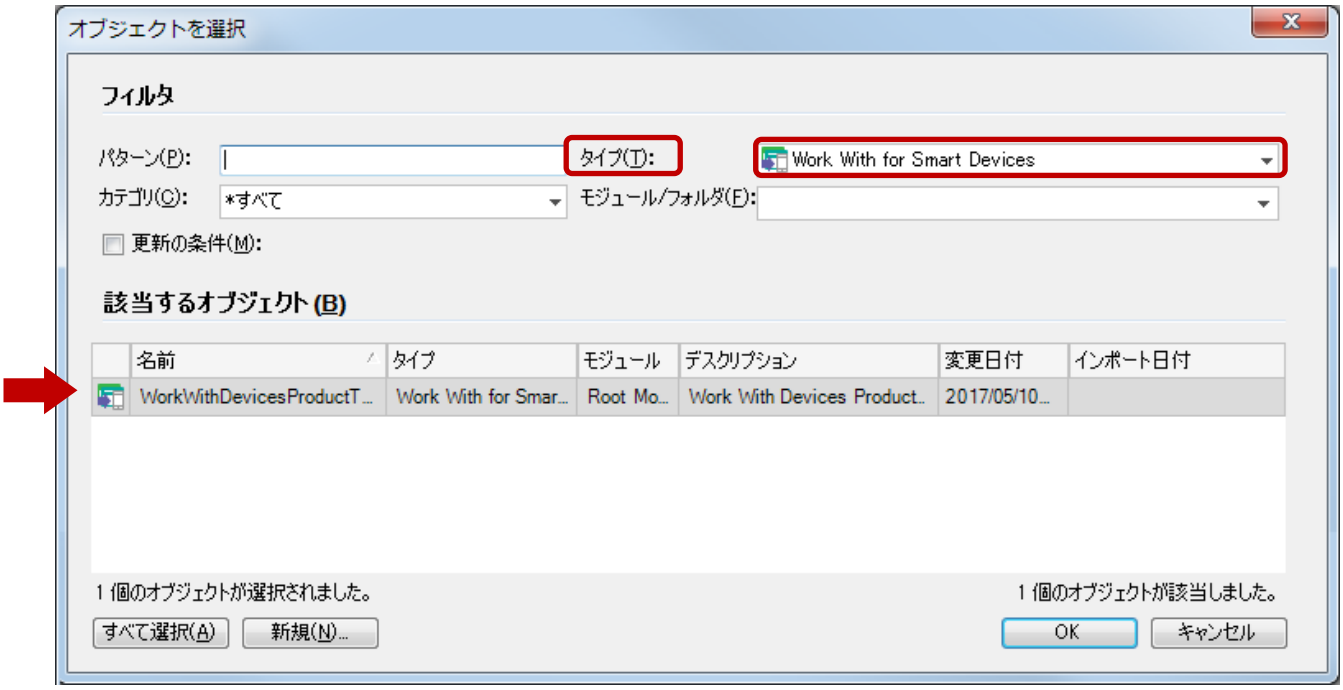
この画像コントロールをダブルクリックすると、関連付けられた既定のイベントが表示されます:



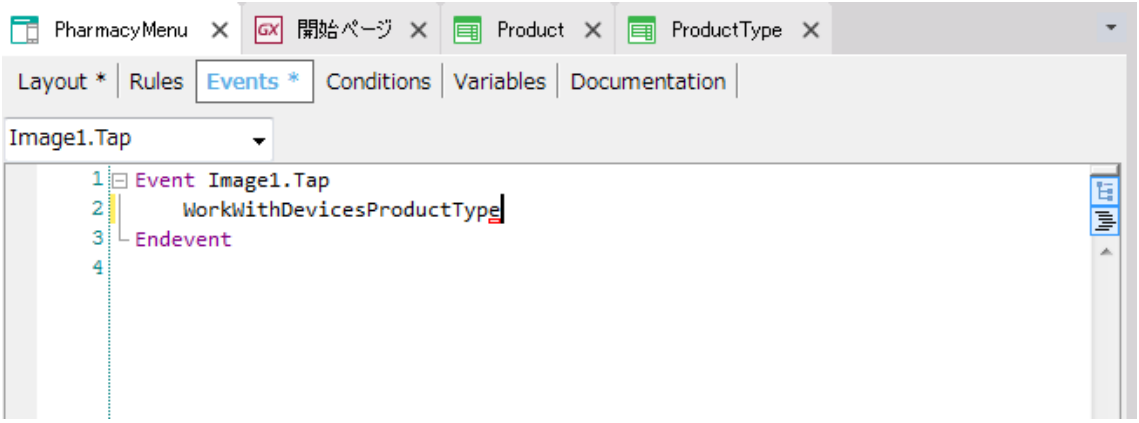
画像に関連付けられているTapイベント内で**WorkWithDevicesProductType**オブジェクトを呼び出す必要があります。しかし、このオブジェクトは多くのパートから構成されています（ノードを思い出してみてください）。ここでの目的は、ドラッグストアで提供される商品タイプがすべて含まれたリストを表示することであるため、今回はオブジェクトの**[List]** ノードを呼び出す必要があります。

これを行うには、イベント内にカーソルを置いてから **[追加]** → **[オブジェクト]** を選択します。

次のようなダイアログが表示されます:

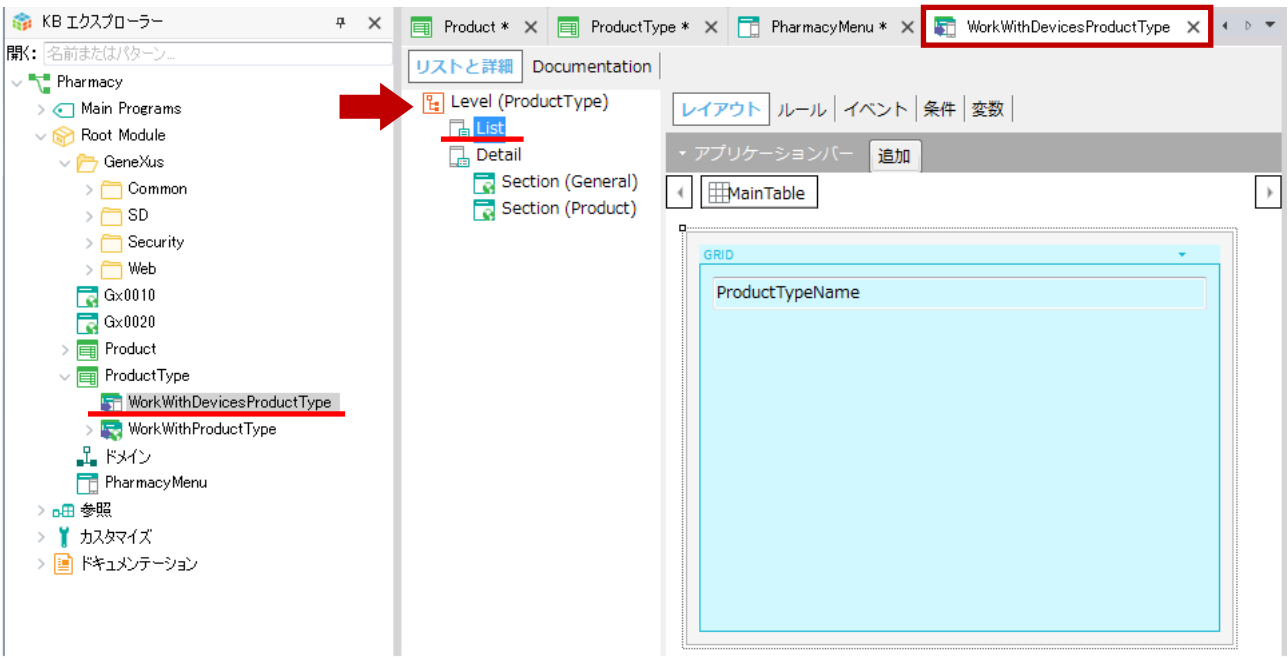


上の画像のようにフィルタリングし、**WorkWithDevicesProductType** オブジェクトを選択します:

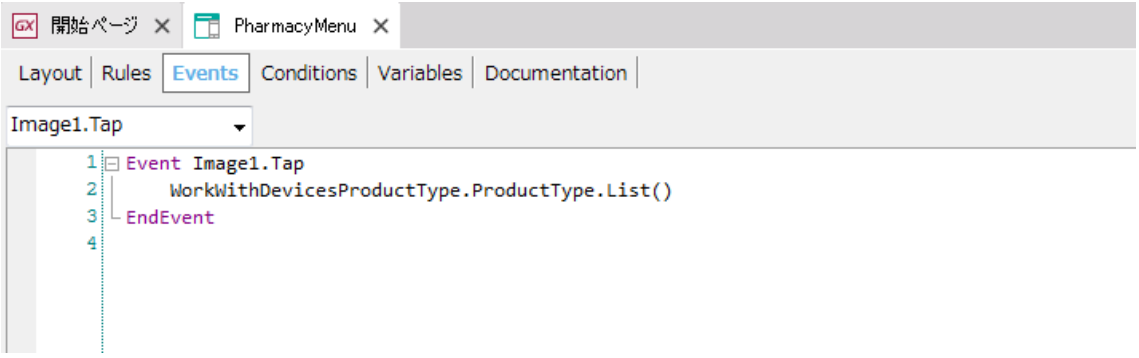


オブジェクトの名前が挿入されるので、実行する必要があるオブジェクトコンポーネントを入力します。

コード行に入力すべき内容を確認するには、**WorkWithDevicesProductType** オブジェクト内に含まれているノードツリーを表示します:

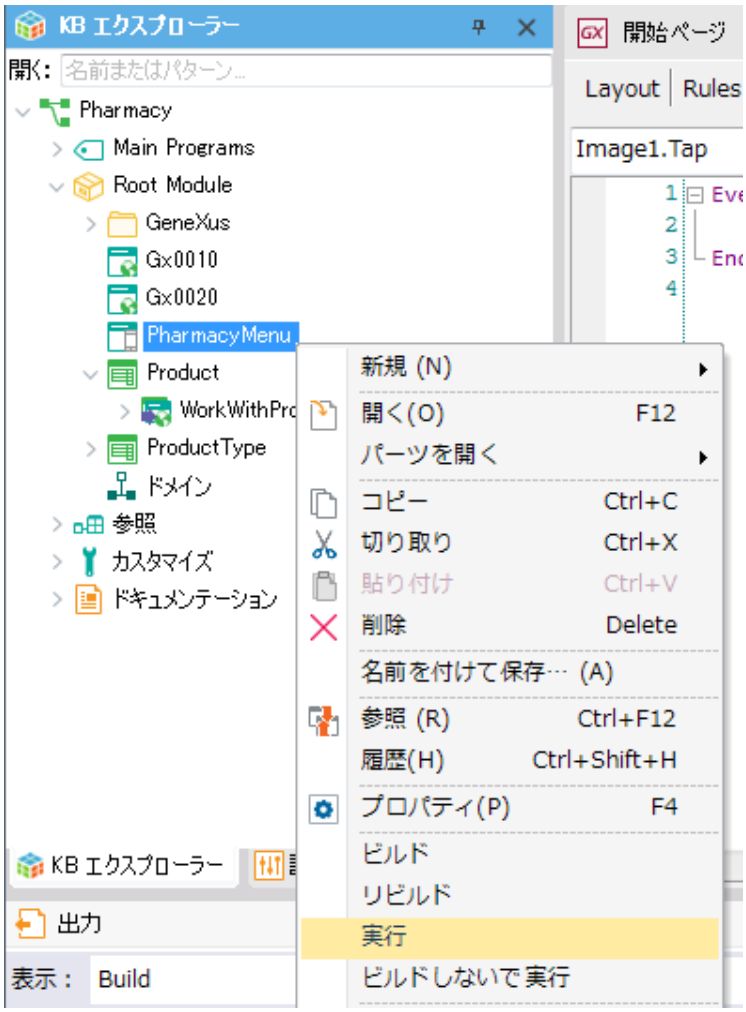


このオブジェクトには**ProductType** というメインノードがあり、その下に [List] ノードと [Detail] ノードがあります。[List] ノードを呼び出すための完全な構文は次のとおりです:



これですべてが定義され、モバイルおよびスマート デバイス アプリケーションを実行する準備が整いました。

PharmacyMenu オブジェクトは、[Main program] プロパティが [True] に設定されているため、[KB エクスプローラー] とは関係なく実行できます (オブジェクトを右クリックして **[実行]** を選択します):



このオブジェクトは、作業しているコンピューター上に開いたモバイルおよびスマート デバイス エミュレーターで実行されます。また、作業しているコンピューターにスマートデバイスを接続すれば、そのスマートデバイスで直接実行できます:

画像をタップすると、ドラッグストアで提供されている商品タイプのリストが表示されます:

右上の追加ボタンは簡単に削除できます。このアプリケーションはエンドユーザー用であり、エンドユーザーが新しい商品を挿入する必要はありません。エンドユーザーに必要なのは、さまざまな商品タイプの表示機能だけであり、編集機能は不要です。

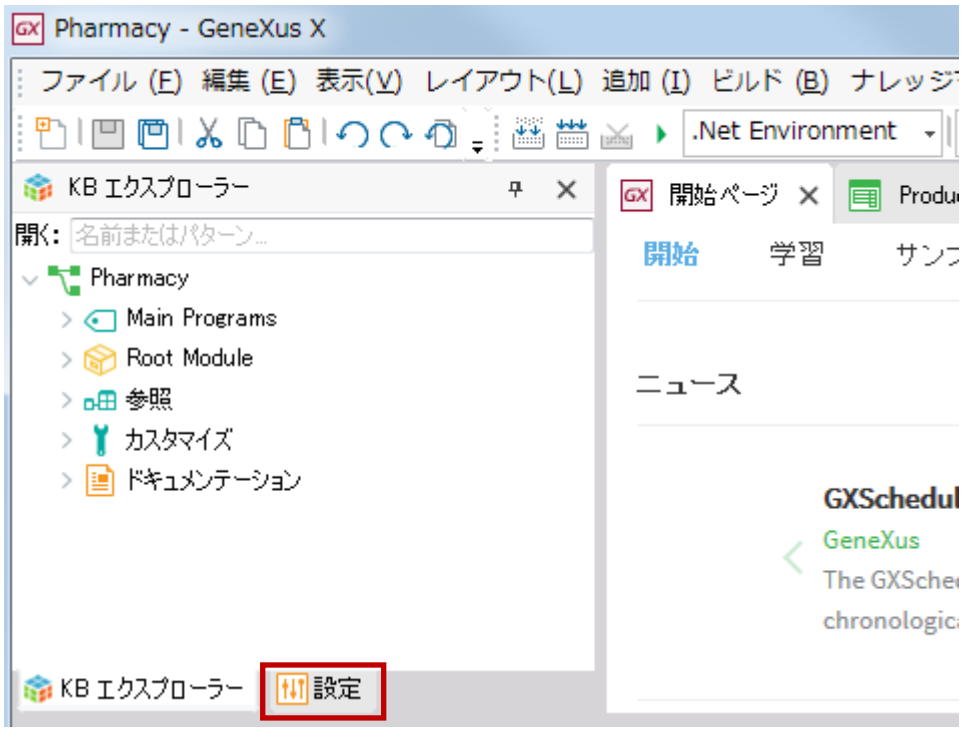
各商品タイプ（たとえば「十代の化粧品」）をタップすると、GENERAL と商品という 2 つのセクションから成る**詳細**が表示されます:

言うまでもなく、これはきわめてシンプルなデモにすぎません。はるかに高度なアプリケーションも実現可能です。

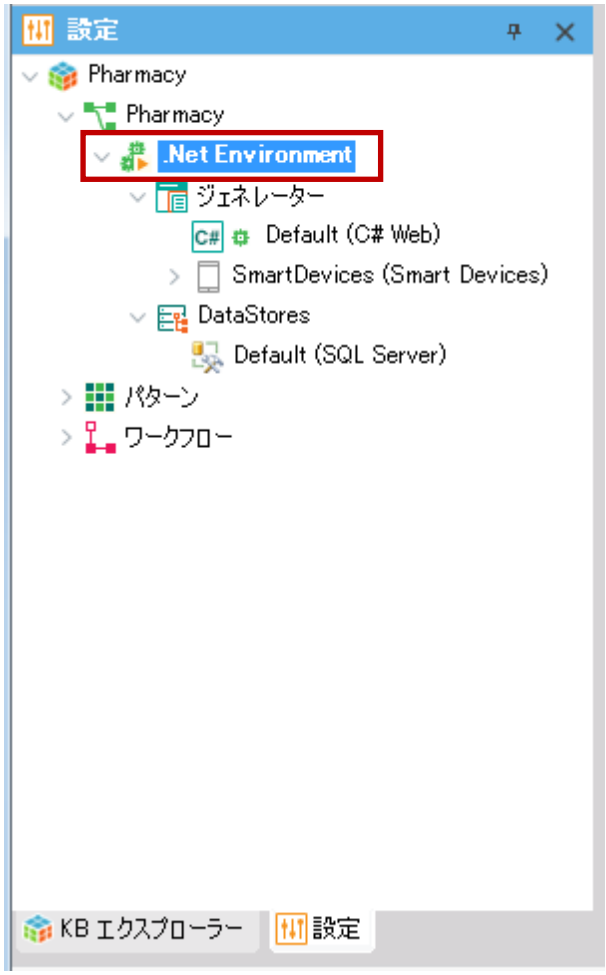
ここまで定義してきたものを別の言語、別のデータベースで生成するには

前述のとおり、GeneXus の大きな特長の 1 つは、異なるプラットフォームに対応した同じアプリケーションの生成、異なるプログラミング言語でのコードの生成、異なるデータベースでのアプリケーションデータの格納が可能であるという点です。これらの情報はすべて**環境**に定義します。

[設定] ウィンドウにアクセスします:



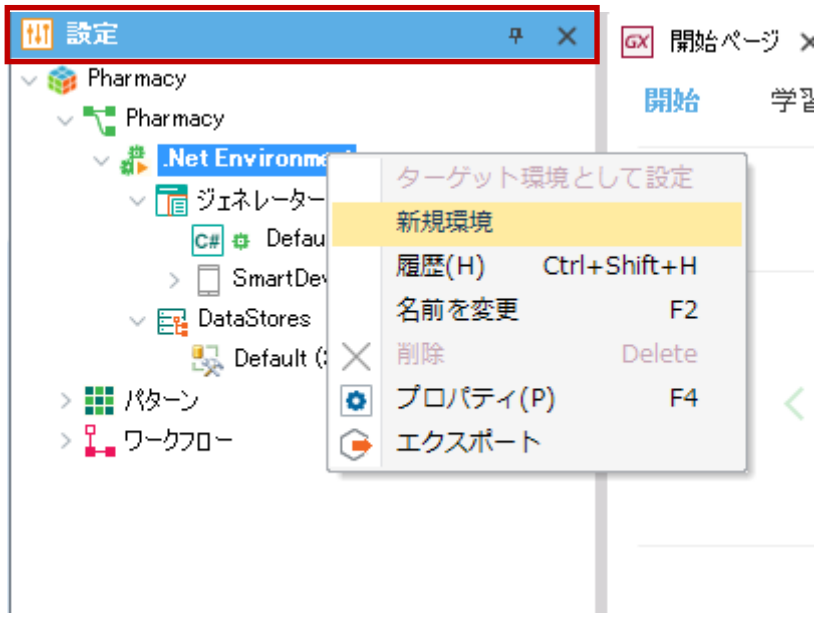
F5 キーをはじめて押したときに作成された 1 つの環境の情報が表示されます (ノードとサブノードには設定可能なプロパティがあります):



同じナレッジベースで複数の実行環境を作成できます。たとえば、一般的には、テストデータが格納されたデータベースに接続されている開発環境と、完成したシステムで使用されるサーバーとデータベースが定義された本番環境を作成します。

また、すべてのものを別のプラットフォーム用に生成する場合は、同じナレッジベースに新しい環境を作成することをお勧めします。

新しい環境を作成するには、次の画像のように、現在の環境ノードを右クリックし、**[新規環境]** を選択します。言語とデータベースを選択し、必要なプロパティを設定します。



新しい環境が作成されたら、アクティブにするためにその環境を右クリックして **[ターゲット環境として設定]** を選択する必要があります。

アクティブな環境は再生マークで特定できます。

GENEXUS のその他の特長

- 外部データベースへのアクセス
    - GeneXus アプリケーションから外部データベースにアクセスしなければならないこともあります。たとえば、場合によっては、最初のロードのときに、外部データベースからナレッジベースに関連付けられたデータベースのテーブルにデータをロードする必要があります。その後は、外部データベースへの接続を維持する必要がない場合もあります。また、1 つ以上の外部データベースの特定のテーブルに接続し、その接続を常に維持する必要がある場合もあります (データを読み取るだけでなく、データにアクセスして変更する必要もあるかもしれません)。上記のニーズに対応できるように、GeneXus は、外部データベースのテーブルに接続するための「リバース エンジニアリング プロセス」を備えています。

- 協同的な開発サポート
    - **GeneXus Server** では、ナレッジベースをサーバーにアップロードすることもできます。その後、新しい開発者は、どのような場所からでも必要に応じて、サーバーのナレッジベースからローカルコピーを作成できます。
      開発者は (常にローカルで) 作業を行い、変更したものをサーバーにアップロードできます。言うまでもなく、競合を解決するためのメカニズムも備えています。このソリューションには、ナレッジベースのモデルのバージョン管理など数多くのメリットがあります。

- Web サービスの使用および定義
    - サードパーティが開発した Web サービスを GeneXus アプリケーションから使用できます。また、GeneXus で独自の Web サービスを開発することもできます。

- データベースの大規模更新の定義、およびほかのプロセスタイプの自由定義

- Web アプリケーションとモバイルおよびスマート デバイス アプリケーション両方のパーソナライズされたインタラクティブなパネルの定義

- UI (ユーザーインターフェース) の設計および微調整
    - GeneXus では、ユーザーインターフェースをカスタマイズできます。ユーザーエクスペリエンスはきわめて重要であるため、ネイティブアプリケーション、レスポンシブ Web デザインのアプリケー

ション、Web モバイルなどのための特別なジェネレーターが用意されています。また、クロスプラットフォームのライブ編集機能も備えています。これにより、設計をアプリケーションに適用してライブプロトタイピングするプロセスが簡略化されます。

- ローカルサーバーまたはクラウドプロバイダーの本番環境へのアプリケーションのデプロイ
  - 1 つのボタンをクリックするだけで、アプリケーションを本番環境にデプロイすることができます。

- ナレッジベース内のドキュメンテーション
  - GeneXus には、Wiki スタイルのドキュメンテーションエディターがあります。そのため、(すべてのナレッジベースで自動的に作成される Main という名前の Documentation タイプのオブジェクトで) ナレッジベースの目的を容易に記述することができます。
    また、どの GeneXus オブジェクトにも [ドキュメンテーション] タブがあります。ここでオブジェクトの目的を記述できます。
    ドキュメンテーションを書き込むときは、テキスト、画像、項目属性へのリンク、オブジェクトなどを含めることができます。ファイルもドキュメンテーションの一部としてナレッジベースに格納できます。

- 拡張性
  - GeneXus では独自のエクステンションを作成できます。これにより、開発者は異なるプラットフォーム言語を使用して独自のソリューションを作成し、GeneXus のコア機能を拡張することができます。

- GeneXus アプリケーションへの外部システムおよびデータソースの統合
  - GeneXus ERP Connector for SAP® ERP により、SAP ERP に統合されたアプリケーションの開発が可能になり、SAP ERP で提供される機能を補完できるようになります。

- セキュリティの管理
  - GeneXus には、GeneXus Access Manager (GAM) という (GeneXus に完全に統合された) セキュリティモジュールがあります。これを有効にすることによって、Web アプリケーションとモバイルおよびスマート デバイス アプリケーション両方の認証機能とオーソライズ機能を解決することができます。

- ビジネスプロセスのモデル化および自動化
  - GeneXus は、ビジネスプロセスのモデル化と自動化を可能にするツールセットと、ビジネスプロセスを管理するための実行環境を備えています。モデリングツールの GeneXus Business Process Modeler は BPMN 2.0 標準に基づいており、ビジネスプロセスのモデル化を目指しているユーザー向けのツール

です。このダイアグラムを GeneXus 開発環境で統合または作成して自動化工程を実装できます。この工程では、GeneXus を使用して、プロセスのモデル化された各タスクのさまざまなオブジェクトを関連付けます。GXflow には、エンドユーザー用の実行ツール、管理ツール、モニターツールがあります。このように、GeneXus では、Business Process Management に基づいたシステム（つまりビジネスプロセス指向システム）の開発を可能にするツールセットである GeneXus BPM Suite が提供されます。

- レポート
  - 静的レポートの定義（印刷、保存、画面表示が可能な一般的なレポート）
  - ビジュアルクエリおよび動的クエリの定義
    - データベースのクエリの作成、1 つ以上の条件に基づいたデータのグループ化、計算、さまざまなグラフ、ピボットテーブル、およびテーブルでの最終的な結果の表示を行うことができます。このようなクエリを実行できるように、GeneXus には Query オブジェクトと Query Viewer コントロールが用意されています。
    - また、GXquery により、エンドユーザーはナレッジベースの同じデータモデルに基づいてクエリを動的に実行できます。このツールは、システムの実際の運用データベースのデータアクセスおよび分析を可能にすることに焦点を当てています。ユーザーは直観的なインターフェースを使用して、独自のクエリを作成し、後で Web インターフェースやモバイルアプリケーションを使用したり、Microsoft Office Excel に統合したりして結果を確認することができます。

- 開発の共有および Marketplace
  - 開発者は、GeneXus Marketplace で、GeneXus 用に作成した、または GeneXus で作成したユーザーコントロール、エクステンション、パターン、外部ツール、外部オブジェクトを共有できます。

- GXtest でのアプリケーションのテスト
  - 新しい機能やバリエーションを実装したら、(変更前の) 既存の機能が引き続き適切に動作するか確認する必要があります。アプリケーションが大きく成長すると、テストすべきものが増えるため、このような確認タスクは非常に長い時間がかかる場合があります。GeneXus では、**GXtest** ソフトウェアによってこの種のテストを自動化できるため、開発者の一連のテスト作業が不要になります。テストは自動的に再生され、システムが適切に動作しているか検証されます。

次のステップ

- GeneXus はきわめて広範な開発プラットフォームであり、学ぶべきことが山ほどあります。

  次の場所で深く掘り下げて学ぶことができます:

  - GeneXus トレーニングコース:

    https://www.genexus.jp/training

  - GeneXus Wiki:

    http://wiki.genexus.jp/hwikibypageid.aspx?46071

開発に役立つサイト

https://www.genexus.jp/community-and-support-jp/support-overview#link05