# First Steps with GeneXus<sup>®</sup>

Create your first Application without knowing how to code.





ADE NON Despension 2000/00/00 de 00.0010/w de 00.0010/w de 00.0010/w de 00.0010/w

# **El Table of Contents**

INTRODUCTION	3
GETTING STARTED WITH GENEXUS	3
CREATING A NEW GENEXUS APPLICATION	4
DEFINING THE FIRST OBJECTS	6
GENERATING AND RUNNING THE APPLICATION FOR THE FIRST TIME	17
THE APPLICATION GROWS UP	25
ADDING BUSINESS RULES	29
DEFINING CALCULATIONS: FORMULAS	32
USING PATTERNS (FOR WEB AND FOR MOBILE DEVICES)	35
GENERATING WHAT YOU HAVE DEFINED SO FAR IN ANOTHER LANGUAGE AND/OR FOR A DIFFERENT DATABASE	57
WHAT ELSE DOES GENEXUS OFFER?	59
NEXT STEPS	62



#### INTRODUCTION

GeneXus is a Low-Code Development Suite that enables the quick generation of software applications in multiple languages and platforms. GeneXus offers several advantages: It's easy to learn, highly productive, cross-platform and future-proof, in a way that both protects your digital assets and simplifies new technology adoption.

For example, GeneXus generates applications for the Web and/or Mobile devices (from a watch, cell phone, tablet or TV), for the target platform selected by the developer (certain language, database, environment, platform, with web responsive design, etc.).

This document is a beginners' guide for developing applications with GeneXus.

#### GETTING STARTED WITH GENEXUS

Upon opening GeneXus, you will see an interface known as IDE (Integrated Development Environment), similar to the one below. This interface is easy to use and may be parameterized by each developer.

It consists of different windows:

GeneXus 18 File Edit View Layout Window Tools Test H	lelp 3	Release	IAPSHOT		- 0 ×
AP Explore     2     4     X     Open: Vision or Patient	Start Page × Start Learn Samp TOP NEWS Tour Onboarc Rodolfo Andre ( U chat allows	ies Marketplace Update		<ul> <li>✓ Properties</li> <li>✓ Properties</li> <li>✓ Properties</li> <li>✓ Properties</li> <li>✓ Properties</li> </ul>	* ×
	Name Dutput	Folder	Last Opened	O Properties T Toolbox (0) Tests Explore	er † X
	Show: General	X IQ 5			Autoscrol



1. Main Window (Start Page): It dynamically displays technical information about the tool and the GeneXus community (news, as well as solutions posted by other developers). It also shows recently used projects available to be opened and it offers the chance to create a new project.

2. **KB Explorer:** Displays objects and settings of the currently open project.

3. Toolbar: Displays an easy-to-use interface for commonly used functions in GeneXus.

4. **Properties window:** Displays properties associated with the context in which the developer is positioned (like a selected object, attribute, variable, control, etc.).

5. **Output:** Displays the output of the actions performed.

#### CREATING A NEW GENEXUS APPLICATION

To start developing a new GeneXus application, you have to create a new **Knowledge Base** (a Knowledge Base is a GeneXus project).

By selecting **File > New > Knowledge Base** in the Toolbar, the following dialog box will be displayed:

#### New Knowledge Base

totyping Target User Interface Language English User Interface Language English User Interface Language English V Interface Language Interface Lan	c Advanced	
cal  English  Ck end totyping Environment NET  SQL Server  English  English  Front end Meb (.NET)  Android  Web (Angular)	ototyping Target	User Interface Language
ck end   totyping Environment   .NET   ta Source   SQL Server   Front end     Front end     Web (.NET)     Android     Web (Angular)	ocal 🗸 🗸	English ~
	I.NET ~	☐ ∰ Android ☐ ऒ Apple ☐ Ø Web (Angular)



The sample application that will be defined throughout this document is a real, but simplified application for a pharmacy. So, it makes sense to call the Knowledge Base "Pharmacy" (or "PharmacySystem", among other options). Also, the location where you want to create the Knowledge Base must be entered.

The next step consists of selecting, if you want to, the programs to be generated by GeneXus in your local prototyping machine or in the GeneXus Prototyping Cloud. Select "GeneXus Prototyping Cloud" in the **Prototyping Target** combo box.

	Phamacy	
ocation:	C:\KBs	
Basic	Advanced	
Prototy Gene	ping Target Xus Prototyping Cloud 🛛 🗸 🗸	User Interface Language
Back of Prototy	end rping Environment IET ~	Front end
Data S	Source	🗌 🔇 Web (Angular)

The **User Interface Language** combo box enables you to select the language in which you want GeneXus to generate automatic button captions, labels, messages for the users, etc. The default language is English.

The Back end box allows you to indicate:

- One of the programming languages available in the Prototyping Environment combo box. GeneXus will use the selected language to generate the back end application programs, as well as the necessary programs to create and maintain the database. By default, the selected language is .NET. Keep it.
- The DBMS over which the database will be created, accessed, and maintained. The Data Source combo box offers SQL Server by default. Keep it. Further ahead, you will have to enter the database details.



The **Front end box** allows you to indicate the programming languages you want GeneXus to generate the front end applications programs with. .NET is the language offered by default to generate front end applications for the Web. You can select others to generate Web and/or Native Mobile applications, too. In this case, .NET will be used.

By pressing the Create Button, GeneXus starts the Knowledge Base creation process.

#### DEFINING THE FIRST OBJECTS

Once a new Knowledge Base is created, the next step is to describe the users' visions . In order to do so, it is necessary to identify real-life objects (we recommend paying attention to the nouns that users mention in their descriptions, such as: products, invoices, customers, etc.) and start defining them by using GeneXus **objects**.

GeneXus developers don't work on low-level tasks such as defining tables, normalizing, designing programs, programming, etc. Instead, their work is a higher-level activity that implies describing the users' reality. After that, GeneXus analyzes the defined objects and goes on to design the database and the application programs for the selected platform in a totally **automatic manner**.

Consider the case where the pharmacy requesting the application asks to be able to record the products they have on sale.

To describe each identified real-life object, you have to create a GeneXus object of the Transaction type (not related to Database Transactions). So, let's see how to create a Transaction object to describe the Products.

By selecting **File > New > Object** in the Toolbar, the following dialog box will be displayed to allow you to select the type of object you want to create and enter a name for it. You have to select the Transaction type and you can call it: *Product*.



Select a Categor	ry:	Select a Type:		
Data Managem User Interface BPM Chatbots Resources Documentation Extensibility Deploy Reporting T Describes an ob nanipulation.	ent	API Data Provider Data Selector Data View Domain Procedure Structured Data Type	Subtype Group Transaction	UI for data
Name:	Product			
Description:	Product			
	10		~	

By clicking on the Create button, the *Product* Transaction is created and it keeps open ready for you to start defining its structure:

ama	Type	Description	Formula	Nullable
anne Dua duat	Type	Description	Formula	Nullable

Each Transaction has some sections that will gradually be explained. Specifically, the Transaction structure enables the definition of the attributes or fields that describe a real-life object.

Suppose that at the pharmacy you were told that they need to keep a record of every product's code, name, sale price, stock and its type (medicine, cosmetic, etc.). Therefore, this data that must be recorded for each product matches the attributes that have to be created for this Transaction.

Note that in the image above, the first line in the Transaction structure is created ready to enter the first attribute. Also note that an icon key is associated with this line. The reason for this is that in every Transaction, an attribute – or set of attributes – must be set with an identifier or key role.

The concept of identifier or key attribute is aimed at uniquely identifying each product (or any object). In other words, the users will not be able to enter two products with the same identifier value. Clearly, the key attribute of the Product Transaction is the product code. So, let's see how to define it.



By pressing the dot key on the keyboard, GeneXus will automatically show the Transaction name as prefix in the attribute name:

Product * ×				
Structure 🐔 Web	Layout   Rules   Event	Variables   Help   Docume	entation Patterns	
Name	Туре	Description	Formula	Nullable
Product	Product	Product		
Product				86

and you only have to type *Code* after the *Product* prefix:

Product* ×				$\sim$
Structure * 🐇 Web Layo	ut Rules Events Variab	bles Help Documentation	Patterns	
Name	Туре	Description	Formula	Nullable
Product	Product	Product		
ProductCode	Numeric(4.0)	Product Code		No

Then, you can choose the data type for this attribute. The default data type is: Numeric of 4 digits with no decimals. However the pharmacy requested that the product code always be a numeric value of up to 10 digits, so you have to change its length to 10:

Product · ×				~
Structure * 🧏 Web L	ayout   Rules   Events	Variables Help Docur	mentation Patterns	Nullable
Product	Product Numeric(10.0)	Product Product Code		No
				No

By pressing Enter, a new line is opened where you can start defining the second attribute. Again, you have to type the dot key on the keyboard and complete the attribute name with *Name*, that is, *ProductName* (of the Character type, and length of 50):

Product * X	avout   Pular   Event	- Variabler Help Dorug	antation Dattacar	
Name	Type	Description	Formula	Nullable
🖃 🥅 Product	Product	Product		
ProductCode	Numeric(10.0)	Product Code		No
ProductName	Character(50)	Product Name		No



Now you must add the *ProductPrice* attribute (of the Numeric type, with 9 digits and 2 decimals):

Product * ×				
Structure * 🐇 Web L	ayout   Rules   Events	Variables   Help   Docur	nentation   Patterns	
Vame	Туре	Description	Formula	Nullable
Product	Product	Product		
ProductCode	Numeric(10.0)	Product Code		No
ProductName	Character(50)	Product Name		No
ProductPrice	Numeric(9.2)	✓ Product Price		No

As you will probably need to create more attributes to define prices or amounts (i.e. when the pharmacy buys or sells products), it might be a good idea to create a generic definition type for all prices. To do this, you just have to write: "Price=" in the Type column, before the type you have recently selected:

Structure * 😽 Web L	ayout   Rules   Events	Variables   Help   Docum	nentation Patterns	
Name	Туре	Description	Formula	Nullable
Product	Product	Product		
- ProductCode	Numeric(10.0)	Product Code		No
ProductName	Character(50)	Product Name		No
ProductPrice	Price =Numeric(9.2)	Product Price		No

Press Enter, and you will see that the *ProductPrice* attribute has been set as *Price* type:

Structure * 🐇 Web L	ayout Rules Events	Variables   Help   Docum	nentation Patterns	
lame	Туре	Description	Formula	Nullable
Product	Product	Product		
ProductCode	Numeric(10.0)	Product Code		No
ProductName	Character(50)	Product Name		No
ProductPrice	Price	✓ Product Price		No

Your *Price* definition with Numeric type (9 digits with 2 decimals) is called **Domain**.

Domains aim at making generic definitions possible. One of the advantages the domains provide is that, if later on you need prices to be Numeric of a different length, changing the domain definition will be enough to update all the attributes based on that domain in a single step.



You can see the created domains in the Knowledge Base, by selecting **View > Domains** in the Toolbar:

Type         Module         Description           PerintStatus         Numeric(4.0)         GeneXus         Event Status           PerintStatus         LongVarChar(2M)         GeneXus         Event Status           PerintStatus         LongVarChar(2M)         GeneXus         Event Status           PerintStatus         LongVarChar(2M)         GeneXus         Event Status         Price           PerintStatus         Numeric(1.0)         GeneXus         Application State         Syndmonization Receiv, Result           PerintStatus         Numeric(1.0)         GeneXus         Application State         Region State           RegionState         Numeric(1.0)         GeneXus         BeacorProximity         Module         Root Module           RescorProximity         Numeric(1.0)         GeneXus         Http: Authentication Type         Module         Root Module           RescorProximity         Numeric(1.0)         GeneXus         Http: Authentication Type         Discore           RescoreProximity         Numeric(1.0)         GeneXus         Flex Night Content         Discore           RescoreProximity         Numeric(1.0)         GeneXus         Flex Night Content         Discore           RescoreProximity         Common Call Target         Numeric(1.0)         G	Filter:	Module:		× …	L	2 Ye Filter	
Image     Price     Price       Period     DeventStatus     Numeric(4.0)     GeneXus     EventStatus       Period     EventStatus     LongVarChar(28)     GeneXus     EventStatus       Period     EventStatus     LongVarChar(28)     GeneXus     EventStatus       Period     EventStatus     LongVarChar(28)     GeneXus     EventStatus       Period     EventStatus     Numeric(1.0)     GeneXus     Application State       Period     RegionState     Numeric(1.0)     GeneXus     Application State       ResconProximity     Numeric(1.0)     GeneXus     Region State       ResconProximity     Numeric(1.0)     GeneXus     Region State       ResconProximity     Numeric(1.0)     GeneXus     Media Frink Resson       Numeric(2.0)     GeneXus     Http Authentication Type       ResconProximity     Media Frink Resson     Object Visibility       Public     Target     SeconProximity       ResconProximity     GeneXus     BeruXus     Common Call Target       ResconProximity     GeneXus     BeruXus     Common Call Target       ResconProximity     GeneXus     Proce     Object Visibility       Public     Target     SeconProximity     Proce       ResconProximity     GeneXus		Tupa	Modula	Description		V Domain: Price	
Points     Longitacion     Database     Enclusion       Ventitiona     Longitacion     Description     Price       Ventitiona     Longitacion     Genekus     Event Data       Ventitiona     Longitacion     Genekus     Event Data       Ventitiona     Longitacion     Genekus     Event Data       Ventitiona     Application State     Multis in Forms     Empty as Null       Ventitiona     Genekus     Synchronization Receive Result     Module     Root Module       Ventitiona     Genekus     Regionistate     Numeric(1.0)     Genekus     Regionistate       Ventitiona     Genekus     Method     Character (2)     Genekus     Method       Ventitional     Common Call Target     Character (20)     Genekus     Flex Direction       Ventitional     Common Call Target     Comparetific     Genekus     Flex Direction       Ventition     Contracter (20)     Genekus     Flex Direction     Mumeric       Ventition     Genekus     Flex Direction     Mumeric       Ventralification     Genekus	- P EventStatue	Numeric(4.0)	GeneYur	Event Statur	- "	Name	Price
Production     Comported Trans     Comported Trans       Production     Comported Trans     Production       Production     Comported Trans     Comported Trans       Production     Comported Trans     Production       Production     Comported Trans     Production       Production     Comported Trans     Production       Production     Comported Trans     Production       Production	P EventData	LoogVarChar(2M)	GeneYus	Event Data		Description	Price
ApplicationState     Numeric(1.0)     Genebula     ApplicationState     MulticationState       ApplicationState     Numeric(1.0)     Genebula     ApplicationState     MulticationState       RegionState     Numeric(1.0)     Genebula     ApplicationState     MulticationState       RegionState     Numeric(1.0)     Genebula     RegionState       MeddaminResson     Numeric(1.0)     Genebula     RegionState       MeddaminResson     Numeric(1.0)     Genebula     MeddaminResson       MeddaminResson     Character(7)     Genebula     Http://webid       Hittp://webidid     Character(7)     Genebula     Http://webididididididididididididididididididid	EventErrors	LongVarChar(2M)	GeneYus	Event Errors		Mulle in Ferme	Emph. os Null
Syndromastor     Class     Attribute       RegionState     Numerc(4.0)     GeneXus     SyndromastorReceive Result       RegionState     Numerc(4.0)     GeneXus     RegionState       ModuFindAResson     Numerc(4.0)     GeneXus     BeaconProximity       MedaFindAResson     Numerc(4.0)     GeneXus     Meda FindA Resson       MittyNethod     Character(7)     GeneXus     Http Authentication Type       CommonCalTarget     CommonCalTarget     Diget Visibility     Public       RecoldProve     VarChar(40)     GeneXus     Flex More       RecoldProve     Character(20)     GeneXus     Flex More       RecoldProve     CommonCalTarget     Based on     (none)       Based On     Character(20)     GeneXus     Flex Author Content       ReckultpContent     Character(20)     GeneXus     Prexision       MotionalBorestion     Character(20)     GeneXus     Prexision       MotionalPresition     Character	- ApplicationState	Numeric(1.0)	GeneXus	Application State		Nulls in Forms	Empty as Null
Productation     Numer(1,0)     GeneXus     Region State     Module     Root Module       RegionState     Numer(1,0)     GeneXus     Region State     Qualified Name     Price       Media/michascon     Numer(2,0)     GeneXus     Media/michascon     Numer(2,0)     GeneXus     Media/michascon       Media/michascon     Character(7)     GeneXus     Media     Findh Reson     Mumer(2,0)     GeneXus       Media/michascon     Character(7)     GeneXus     Http Authentication Type     Object Visibility     Public       BiscrodeType     Nameric(30)     GeneXus     Bec Dection     Fies/Recon     Common Call Target     Based on     (none)       BiscrodeType     Varchar(20)     GeneXus     Fies/ Nation     Data Type     Based on     (none)       Pies/Wap     Compare Kind     Character(20)     GeneXus     Fies/ Mign Content     Data Type     Numeric       Pies/MignContent     Character(20)     GeneXus     Fies/ Mign Content     Length     9       Pies/MignContent     Character(20)     GeneXus     Prickal Prick     Signed     Signed       Compare/Mind     Character(20)     GeneXus     Prickal Prickalon     Signed     False       MotizontalPosition     Character(20)     GeneXus     Pertical Position     <	- SynchronizationRecei	Numeric(4.0)	GeneXus	Synchronization Receive Result		Class	Attribute
BeacoProximity         Numeric(1.0)         GeneXus         Beacon Proximity         Qualified Name         Price           MedaFinishReason         Numeric(1.0)         GeneXus         MedaFinishReason         Object Visibility         Public           HttpAuthenticationType         CommonCalTarget         Object Visibility         Public         Test Values		Numeric(1.0)	GeneXus	Region State		Module	Root Module
MedaFnickReason     Numeric(4.0)     GeneXus     Meda FnickReason     Object Visibility     Public       MedaFnickReason     Character(7)     GeneXus     Http:Authentication Type     Diget Visibility     Public       MedaFnickReason     Character(7)     GeneXus     Http:Authentication Type     Diget Visibility     Public       MedaFnickReason     Character(20)     GeneXus     Http:Authentication Type     Test Values     Test Values       MedaFnickReason     Character(20)     GeneXus     Barcode Type     Test Values     Test Values       Barcode Type     VarChar(40)     GeneXus     Flex Aufor Content     Data Type     Numeric       FlexAlgnContent     Character(20)     GeneXus     Flex Aufor Content     Data Type     Numeric       FlexAlgnContent     Character(20)     GeneXus     Flex Algn Content     Data Type     Numeric       CompareKind     Character(20)     GeneXus     Flex Algn Content     Data Type     Numeric       CompareKind     Character(20)     GeneXus     Restain     Compare Kind     Signed     False       MediaFnishenson     Character(20)     GeneXus     Peritoal Position     Collection     False       MediaFnishenson     Character(20)     GeneXus     Peritoal Position     Collection     False	BeaconProximity	Numeric(1.0)	GeneXus	Beacon Proximity		Ourlified Mana	Drive
Http://teth.do/     Character(7)     GeneXus     Http://teth.do/     Object Visibility     Public       Ittp://teth.ubinoficationType     Karekus     Common Call Target     Text Values     Text Values       Common Call Target     Character(20)     GeneXus     Barcode Type     Barcode Type     Based on     (none)       PlexUrecton     Character(20)     GeneXus     Flex Varcon     Data Type     Numeric       PlexUrecton     Character(20)     GeneXus     Flex Varcon     Data Type     Numeric       PlexUrecton     Character(20)     GeneXus     Flex Varcon     Data Type     Numeric       PlexUrecton     Character(20)     GeneXus     Flex Algn Content     Data Type     Numeric       PlexUrecton     Character(20)     GeneXus     Flex Algn Content     Decimals     2       Compare Kind     Character(20)     GeneXus     Rex Algn Content     Decimals     2       Compare Kind     Character(20)     GeneXus     Vertical Position     Signed     False       VerticalPosition     Character(20)     GeneXus     Permission     Collection     False       VerticalPosition     Character(20)     GeneXus     Permission     Collection     False       VerticalPosition     Character(20)     GeneXus     Permi		Numeric(4.0)	GeneXus	Media Finish Reason		Qualmed Name	Price
HttpAuthenticationType Numeric(1.0)     Genebus     Http Authentication Type     Test Values       CommonCall Target     CommonCall Target     Secode Type       Barcode Type     Vertical Secode Type       Barcode Type     Vertical Secode Type       PieckWrop     Character (20)     Genekus       PieckWrop Character (20)     Genekus     Pieck Agric Content       Compareldind     Character (20)     Genekus       PieckWrop Character (20)     Genekus     Pieck Agric Content       Compareldind     Character (20)     Genekus       VerticalPosition     Character (20)     Genekus       VerticalPosition     Character (20)     Genekus       VerticalPosition     Character (20)     Genekus       VerticalPosition     Character (20)     Genekus       Permission     Marchar(100)     Genekus Permission       Permission     Varchar(200)     Genekusthanimo       Discorption     Varchar(200)     Genekusthanimo       Discorption     Varchar(200)     Ge	- HttpMethod	Character(7)	GeneXus	Http Method		<b>Object Visibility</b>	Public
Common Call Farget         Character (20)         GeneXus         Common Call Farget         Type Definition           Barcode Type         Marcode Type         GeneXus         Barcode Type         Based on         (none)           PlexNarpo         Character (20)         GeneXus         Plex Decton         Data Type         Data Type           PlexNarpo         Character (20)         GeneXus         Plex Xustfy Content         Data Type         Data Type           PlexNarpo         Character (20)         GeneXus         Plex Xustfy Content         Data Type         Data Type           PlexNarpo         Character (20)         GeneXus         Plex Xustfy Content         Data Type         Data Type           Compare Kind         Character (20)         GeneXus         Plex Xustfy Content         Data Type         Data Type           Compare Kind         Character (20)         GeneXus         Plex Xustfy Content         Data Type         Data Type           Compare Kind         Character (20)         GeneXus         Plex Xustfy Content         Length         9           Compare Kind         GeneXus         Rotation         Rotation         Decimals         2           Peremission         Varchare(20)         GeneXus         Pertisolon         Collection <t< td=""><td>- HttpAuthenticationType</td><td>Numeric(4.0)</td><td>GeneXus</td><td>Http Authentication Type</td><td></td><td>Test Values</td><td></td></t<>	- HttpAuthenticationType	Numeric(4.0)	GeneXus	Http Authentication Type		Test Values	
BarcodeType VarChar(40) GeneXus BarcodeType     BarcodeType VarChar(40) GeneXus Piex VarChar(40)     GeneXus Piex VarChar(40)     GeneXus Piex VarChar(40)     GeneXus Piex VarChar(40)     GeneXus Piex Algn Tems     Character(20) GeneXus Piex Algn Tems     FiexAlgnTems     Character(20) GeneXus Piex Algn Tems     GeneXus Piex Algn Tems     GeneXus Piex Algn Tems     CompareKind     Character(20) GeneXus Piex Algn Tems     GeneXus     GeneXus Piex Algn Tems     GeneXus     GeneXus Piex Algn Tems     GeneXus	- CommonCallTarget	Character(20)	GeneXus	Common Call Target		T	
Pieco/rection     Character (20)     Gene/Lus     Piec Netcon     Based on     (none)       Pieck/inpact     Character (20)     Gene/Lus     Pieck July Toolnent     Data Type     Data Type <td>- T BarcodeType</td> <td>VarChar(40)</td> <td>GeneXus</td> <td>Barcode Type</td> <td></td> <td><ul> <li>Type Definition</li> </ul></td> <td>(</td>	- T BarcodeType	VarChar(40)	GeneXus	Barcode Type		<ul> <li>Type Definition</li> </ul>	(
PreckVrap         Character (20)         GeneXus         Flex Vrap         Data Type         Numeric           PreckJutfryContent         Character (20)         GeneXus         Flex Algn Content         Length         9           PreckJutfryContent         Character (20)         GeneXus         Flex Algn Content         Data Type         Numeric           PreckJutfryContent         Character (20)         GeneXus         Flex Algn Content         Decimals         2           CompareRind         Character (20)         GeneXus         ReckJan Content         Decimals         2           ReckJancContent         Character (20)         GeneXus         ReckJan Content         Decimals         2           VerticalPosition         Character (20)         GeneXus         Notectal Position         GeneXus         Fermission           VerticalPosition         Character (20)         GeneXus         Permission         Collection         False           VerticalPosition         Title         VarChar(100)         GeneXus/Linnimo         Decryption         Autonumber         False           VerticalPosition         GeneXus/Linnimo         Decryption         Scalar         Initial Value         Initial Value           VerticalPosition         GeneXus/Linnimo         Decryption	- P FlexDirection	Character(20)	GeneXus	Flex Direction		Based on	(none)
PlexAlgrittems     Character (20)     GeneXus     Plex Algrit Content     Length     9       PlexAlgrittems     Character (20)     GeneXus     Plex Algrit Tems     Decimals     2       PlexAlgrittems     Character (20)     GeneXus     Plex Algrit Tems     Decimals     2       Rotation     Numeric (10)     GeneXus     Rotation     Signed     False       VerticalPosition     Character (20)     GeneXus     Vertical Position     Signed     False       VerticalPosition     Character (20)     GeneXus     Vertical Position     Collection     False       Permission     VarChar(40)     GeneXus     Permission     Collection     False       Permission     VarChar(100)     GeneXus     Permission     Autonumber     False       Questry Experiments     VarChar(100)     GeneXus     Description     Autonumber     False       VarGene     VarGeneXus     GeneXus     Permission     Initial value     Initial value       Permission     GeneXus     GeneXus     Collection     False       Autonumber     False     Autonumber     False       Autonumber     False     Initial value     Initial value       Autonumber     False     Initial value     Initial value	- T FlexWrap	Character(20)	GeneXus	Flex Wrap		Data Type	Numeric
PiexAlgnZentems     Character (20)     GeneXus     Fiex Algn Content     Length     9       PiexAlgnZentems     Character (20)     GeneXus     Fiex Algn Content     Decimals     2       Compare Kind     Character (20)     GeneXus     Compare Kind     Discussion     Signed     False       VerticalPosition     Character (20)     GeneXus     Vertical Position     False     Enum Values       VerticalPosition     Character (20)     GeneXus     Permission     Collection     False       VerticalPosition     Character (20)     GeneXus     Permission     Collection     False       VerticalPosition     Character (20)     GeneXus     Permission     Collection     False       VerticalPosition     Character (20)     GeneXus Innimo     Description     Collection     False       VerticalPosition     Collection     False     Dimensions     Scalar       VerticalPosition     GeneXusInamino     Description     Scalar       VerticalPosition     GeneXusInamino     Description     Scalar       VerticalPosition     VerticalPosition     Initial Value     Initial Value       VerticalPosition     GeneXusInamino     Alert Type     VerticalPosition     VerticalPosition	- P FlexJustifyContent	Character(20)	GeneXus	Flex Justify Content		eene ope	
PiexAlgnContent     Character (20)     GeneXus     Flex Algn Content     Decimals     2       CompareVind     Character (3)     GeneXus     Compare Vind     Signed     False       Rotation     Numer(1,0)     GeneXus     Rotation     Enum Values     Collection     False       VertoalPosition     Character (20)     GeneXus     VertoalPosition     Collection     False       Permission     VarCher(40)     GeneXus     Permission     Collection     False       Title     VarChar(100)     GeneXus/hanimo     Discription     Code     Munumber     False       Ocde     VarChar(100)     GeneXus/hanimo     Decorption     Dimensions     Scalar       Operation     VarChar(100)     GeneXus/hanimo     Alert Type     Initial value	- T FlexAlignItems	Character(20)	GeneXus	Flex Align Items		Length	9
Propereising     Compare/Kind     Compare/Kind     Signed     False       Proceeding     Rotation     Numeric(1.0)     GeneXus     Rotation     False       Proceeding     Compare/Kind     Compare/Kind     False     False       Proceeding     Compare/Kind     Compare/Kind     False       Proceeding     Compare/Kind     False     False       Proceeding     Compare/Kind     Collection     False       Proceeding     VarChar(90)     GeneXus     Permission       Proceeding     VarChar(100)     GeneXus/Inamino     Title       Proceeding     VarChar(100)     GeneXus/Inamino     Description       Proceeding     VarChar(100)     GeneXus/Inamino     Description       Proceeding     VarChar(100)     GeneXus/Inamino     Description       Proceeding     VarChar(100)     GeneXus/Inamino     Dimensions       Scalar     Initial value     Initial value     Initial value	- FlexAlignContent	Character(20)	GeneXus	Flex Align Content		Decimals	2
Protectalism     Numeric(1.0)     Gene/Lus     Rotation     Junited     Value       VerticalPosition     Character(20)     Gene/Lus     VerticalPosition     Collection     False       VerticalPosition     Character(20)     Gene/Lus     VerticalPosition     Collection     False       VerticalPosition     Character(20)     Gene/Lus     Permission     Collection     False       VerticalPosition     Character(20)     Gene/Lus     Permission     Autonumber     False       Title     VarChar(100)     Gene/LusIhamimo     Description     Dimensions     Scalar       Quector     VarChar(100)     Gene/LusIhamimo     Code     Dimensions     Scalar       Quector     VarChar(100)     Gene/LusIhamimo     Alert Type     VarChar(100)     Gene/LusIhamimo       Quector     VarChar(100)     Gene/LusIhamimo     Alert Type     VarChar(100)     Gene/LusIhamimo	CompareKind	Character(4)	GeneXus	Compare Kind		Signed	Falco
Cole VarChar(10)     CeneXus     VarChar(20)     Code     VarChar(20)     CeneXus     VarChar(20)     Code     VarChar(20)     CeneXus     VarChar(20)     Code     VarChar(20)	- Rotation	Numeric(1.0)	GeneXus	Rotation		Signed	Tabe
Permission     Code     VarChar(30)     CeneXus     Permission     VarChar(40)     CeneXus     Permission     VarChar(100)     CeneXusUnamino     Code     VarChar(100)     CeneXusUnamino     Code     VarChar(100)     CeneXusUnamino     Code     VarChar(100)     CeneXusUnamino     Code     VarChar(100)     CeneXusUnamino     AertType     VarChar(100)     CeneXusUnamino     Code     Code     VarChar(100)     CeneXusUnamino     Code     VarChar(100)     CeneXusUnamino     Code     VarChar(100)     CeneXusUnamino     Code     Code     VarChar(100)     CeneXusUnamino     Code     Code     Code     Code     Code     Code     Code     Code     Cod	VerticalPosition	Character(20)	GeneXus	Vertical Position		Enum Values	
Permission         VarChar(40)         GeneXus         Permission           Title         VarChar(100)         GeneXusIhanimo         Title           Description         VarChar(200)         GeneXusIhanimo         Description           Code         VarChar(100)         GeneXusIhanimo         Code           AlertType         VarChar(100)         GeneXusIhanimo         Code           AlertType         VarChar(100)         GeneXusIhanimo         Alert Type	- HorizontalPosition	Character(20)	GeneXus	Horizontal Position		Collection	False
Title         VarCher(100)         GeneXusthanimo         Title         Autonumber         raise           2         Description         VarChar(200)         GeneXusthanimo         Description         Dimensions         Scalar           2         Code         VarChar(100)         GeneXusthanimo         Code         Initial value         Initial value           2         AlertType         VarChar(100)         GeneXusthanimo         Alert Type         VarChar(100)         GeneXusthanimo	Permission	VarChar(40)	GeneXus	Permission		Automation	E.L.
Description         VarChar(200)         GeneXusUnanimo         Description         Dimensions         Scalar           Code         VarChar(100)         GeneXusUnanimo         Code         Initial value         Initial value           Public HCType         VarChar(100)         GeneXusUnanimo         Alert Type         VarChar(100)         GeneXusUnanimo         Alert Type	Title	VarChar(100)	GeneXusUnanimo	Title		Autonumber	raise
Code         VarChar(100)         GeneXusUnanimo         Code         Initial value           -QuertType         VarChar(100)         GeneXusUnanimo         Alert Type         VarChar(100)         VarChar(100)         Alert Type         VarChar(100)         VarChar(100	- Description	VarChar(200)	GeneXusUnanimo	Description		Dimensions	Scalar
AlertType VarChar(100) GeneXusUnanimo AlertType VarChar(100)	Code	VarChar(100)	GeneXusUnanimo	Code		Initial value	
Validation	AlertType	VarChar(100)	GeneXusUnanimo	Alert Type		interes volue	
L Alert Position VarChar(100) GeneXusUnanimo Alert Position	AlertPosition	VarChar(100)	GeneXusUnanimo	Alert Position		> Validation	

As shown in the image, GeneXus automatically creates some domains. When you click on a given domain, the Properties Window is refreshed with its properties. Note that, in addition to setting the data type for a domain, you can also define other interesting properties for it.

The Properties window will be refreshed every time you select another attribute, domain, object, etc., with the corresponding properties available for configuration in each case. If it is not visible, you can press F4 to open it.

Let's go back to the *Product* Transaction, where the next attribute you have to define is *ProductStock* of the Numeric type, and length 4:

Structure * 📕 Web L	ayout   Rules   Events	Variables Help Docun	nentation Patterns	
ame	Туре	Description	Formula	Nullable
Product	Product	Product		
ProductCode	Numeric(10.0)	Product Code		No
ProductName	Character(50)	Product Name		No
ProductPrice	Price	Product Price		No
ProductStock	Numeric(4.0)	Product Stock		No



Now it is necessary to record the product type. You could create an attribute called *ProductType* as Character(50)...

Product * ×				$\sim$
Structure * 🐔 Web Layout	Rules Events Variab	les   Help   Documentation   Patter	ns	
Name	Туре	Description	Formula	Nullable
Product	Product	Product		
📍 ProductCode	Numeric(10.0)	Product Code		No
	Character(50)	Product Name		No
····   ProductPrice	Price	Product Price		No
···  ProductStock	Numeric(4.0)	Product Stock		No
<ul> <li>ProductType</li> </ul>	Character(50)	Product Type		No
	$\sum$			

But what happens if the users want to enter two products of the same type? They would have to enter the same type name twice, being careful to type it exactly the same! Later on, they might need to search for all the products of a certain type, and to get them, the name must have been typed exactly the same.

Product* ×				$\sim$
Structure * 🕺 Web Layout   Ri	ules Events Variables Help	Documentation Patterns		
Name	Туре	Description	Formula	Nullable
- Product	Product	Product		
📍 ProductCode	Numeric(10.0)	Product Code		No
	Character(50)	Product Name		No
<ul> <li>ProductPrice</li> </ul>	Price	Product Price		No
<ul> <li>ProductStock</li> </ul>	Numeric(4.0)	Product Stock		No
ProductType	Character(50) 🗸	Product Type		No
	1			



It seems more reasonable to enter the type only once, in a single location, and then, for each product, to reference the corresponding product type. So, let's delete the *ProductType* attribute from the structure and save the *Product* Transaction as shown:

Pharmacy - GeneXus 18     File Edit View Layout Insert Build Knowledge     P □ ↓ & □ □ ↑ へ ① ↓ ###	ge Manager Window Too	ls Test Help	+ <sub>∓</sub> Android 1.3-S	NAPSHOT	
WB Explorer         # X           Save	Product* × Structure * % Web L	ayout   Rules   Events	Variables Help Docur	nentation Patterns	~
Dest Madels	Name	Туре	Description	Formula	Nullable
Root Module	Product	Product	Product		
> 😭 General	ProductCode	Numeric(10.0)	Product Code		No
Fharmacy	-  ProductName	Character(50)	Product Name		No
2 Domains	ProductPrice	Price	Product Price		No
Images	ProductStock	Numeric(4.0)	Product Stock		No
> DE References					
> Y Customization					
> Documentation					

GeneXus has automatically designed a Web Layout according to the defined structure. This form will enable users to add, update and delete products in runtime:

Product ×
Structure <mark>% Web Layout</mark> Rules Events Variables Help Documentation Patterns
FormButtons Confirm Cancel Delete
✓ Ⅲ MainTable
Product
<errorviewer: errorviewer=""></errorviewer:>
<toolbar></toolbar>
Code ProductCode
Name ProductName
Price ProductPrice
Stock ProductStock
<formbuttons></formbuttons>



Now let's create another Transaction to record the product types, and after that, let's assign a product type to each product. To do so, select again **File > New > Object** and complete the following:

New Object					×
Select a Categor	y:	Select a Typ	e:		
Data Managem User Interface BPM Chatbots Resources Documentation Extensibility	ent 🔺	API Data Pro Data Sel Data Vier Data Vier Domain Procedur Structure	A Sub vider I Tra ector w e d Data Type	type Group nsaction	
Reporting					
Name: Description:	Produ	ictType ictType			
Module/Folder:	R	oot Module		× …	
				Cre	ate Cancel
ProductType * ×					~
Structure * 🕺 Web	Layout	Rules Events Vari	ables Help Documentati	on Patterns	
Name		Туре	Description	Formula	Nullable
ProductType		ProductType	Product Type		
···· <b>}</b>					NO

Let's store the code and name for each product type:

ProductType * ×	ut Rules Events Var	iables Help Documentation	Patterns	~
Name	Туре	Description	Formula	Nullable
ProductType	ProductType Numeric(4.0)	Product Type Product Type Code		No
ProductTypeName	Character(50)	Product Type Name		No



Remember the recommendation of typing the dot key on the keyboard when defining attributes, **so** that GeneXus will automatically write the Transaction name as prefix, and all you will need to do is to complete the end of the attribute names.

Naming attributes with the Transaction name as prefix not only makes defining attributes easier and faster, but is also a GeneXus community convention for an easier comprehension when reading an attribute name wherever it may be, as to know to which object it is describing.

Look at the properties of the *ProductTypeCode* attribute:

	Tupe	Description	Formula	Nullable	× Attribute: Produ	tTypeCode
ProductTupe	ProductTupe	Product Type	Formula	Nullable	Attribute. I Toda	rippecode
ProductTypeCode	Numeric(4.0)	Product Type Code		06	KBObject	Artech.Genexus.Comm
ProductTypeName	Character(50)	Product Type Name		No	Name	ProductTypeCode
					Description	Product Type Code
					Title	Product Type Code
					Column title	Product Type Code
					Contextual Title	Type Code
					Formula	
					Nulls in Forms	Empty as Null
					Class	Attribute
					Qualified Name	ProductTypeCode
					Test Values	
					✓ Type Definition	1
					Supertype	
					Based on	(none)
					Data Type	Numeric
					Length	4
					Decimals	0
					Signed	False
					Autonumber	False

The Autonumber property is set to False by default. By changing it to True, all the new product types entered by the end user will be automatically numbered in sequence. So, set the Autonumber property to True for this identifier attribute and save the *ProductType* Transaction.



As explained before, each Transaction has a Web Layout automatically designed by GeneXus according to its structure. The following image shows the *ProductType* Web Layout:

3 Confirm Cancel Delete
ible
lype
wer: ErrorViewer>
>
le ProductTypeCode
ne ProductTypeName

Now let's assign a product type to each product. To do so, go back to the *Product* Transaction and type "P" in a new line of the structure. The list of attributes existent in the Knowledge Base that begin with that letter is displayed:

Structure 🕺 Web Layout	Rules Events Varial	oles Help Documentation	Patterns	
ame	Туре	Description	Formula	Nullable
Product	Product	Product		
ProductCode	Numeric(10.0)	Product Code		No
	Character(50)	Product Name		No
ProductPrice	Price	Product Price		No
ProductStock	Numeric(4.0)	Product Stock		No
• P				No
ProductCode				
ProductName				
ProductPrice				
ProductStock				
ProductTypeCode				
ProductTypeName				



Select *ProductTypeCode* and its entire definition is displayed.

Let's also include the ProductTypeName attribute in this Transaction, as users may want to see the corresponding product type name in the form after selecting a product type code while executing this Transaction.

Structure * 🐇 Web Layou	it   Rules   Events   Vari	iables   Help   Documentation   F	Patterns	
Name	Туре	Description	Formula	Nullable
Product	Product	Product		
ProductCode	Numeric(10.0)	Product Code		No
	Character(50)	Product Name		No
ProductPrice	Price	Product Price		No
- ProductStock	Numeric(4.0)	Product Stock		No
ProductTypeCode	Numeric(4.0)	Product Type Code		No
ProductTypeName	Character(50)	Product Type Name		

Let's focus on these two attributes, included in more than one Transaction:

*ProductTypeCode* is the identifier attribute on the *ProductType* Transaction (more specifically, it is the primary key of that Transaction). So, when a primary key is included in another Transaction, GeneXus understands that the attribute has the role of a foreign key there.

Including an attribute that is a Transaction primary key in another Transaction allows you to relate both Transactions.

GeneXus establishes relations through attribute names, so when it finds attributes with the same name in different Transactions, it assumes that they refer to the same concept.

The *ProductTypeName* attribute is also present in both Transactions. However, it is not marked as the identifier of any of the defined Transactions. Therefore, GeneXus takes it as a **secondary** attribute. GeneXus will then include *ProductTypeName* in the *ProductType* physical table that it will create in the database and not in the *Product* physical table.

The concepts of Transaction and physical table are not the same. Keep in mind that Transaction is the GeneXus object that you create in the Knowledge Base to represent an object of reality. Upon considering its structure and the rest of the Transaction structures defined in the Knowledge Base (and also taking into account some properties), GeneXus will determine the physical tables that must be created in the database, as well as the attributes that must be stored in each table.

When executing the *Product* Transaction form at runtime, the user must enter for the *ProductTypeCode* attribute (which is a foreign key attribute there) a value that has been previously registered through the *ProductType* Transaction. Otherwise, an error will be displayed. After entering a valid *ProductTypeCode*, its *ProductTypeName* value will be obtained and shown on screen.



#### GENERATING AND RUNNING THE APPLICATION FOR THE FIRST TIME

If you want to generate and execute the application for the first time, you only have to press F5 and the following dialog box will be displayed:

GeneXus Account	Authentication	?	×
		A	
Enter your GeneXus	Account information.		
<u>U</u> ser name:	2		$\sim$
Password:			
	<u>R</u> emember my p	assword	
	ОК	Ca	ncel

Use your GeneXus Account credentials to complete the following fields:

- User name or email corresponding to your GeneXus Account.
- Password corresponding to your GeneXus Account.

Next, press the OK button.

GeneXus will evaluate the impact caused by the new definitions in the Knowledge Base, and it will show a report under the name **Impact Analysis**.

By clicking on each table (*ProductType* and *Product* on the left) the attributes that will be included in them will be displayed on the right.



The Database tables	will be created.	
This report describes how the I Please select Create to procee	Database tables will be created. d or Cancel.	
Create Cance	al	
Pattem:	Table Product Type specification	\$
Product	Table name: ProductType	
	ProductType is new	
	Table Structure	\$
	Attribute         Definition         Previous values           ProductTypeCode         Numeric (4), Not null, Autonumber           ProductTypeName         Character (50), Not null, NLS	Takes value from
	Indexes	\$
	Name         Definition         Composition           IPRODUCTTYPE         primary key Clustered         Image: ProductType	aCode
	Statements	\$
	CREATE TABLE [ProductType] ( [ProductTypeCode] SMALLINT NOT NULL IDENTITY [ProductTypeName] NCHAR(50) NOT NULL, PRIMARY KEY ( [ProductTymeCodel ))	(1,1),

Observe each table structure proposed by GeneXus to create in the database:

This report describes how the Please select Create to proce	Database tables will be created. ed or Cancel.	
Pattem:	Table ProductType specification	*
Product	Table name: ProductType	
	ProductType is new	
	Table Structure	*
	Attribute         Definition         Previous values Takes value           ProductTypeCode         Numeric (4), Not null, Autonumber           ProductTypeName         Character (50), Not null, NLS	fron
	Indexes	\$
	Name         Definition         Composition           IPRODUCTTYPE         primary key Clustered         Image: ProductTypeCode	
	Statements	*
	CREATE TABLE [ProductType] ( [ProductTypeCode] SMALLINT NOT NULL IDENTITY ( 1 , 1 [ProductTypeName] NCHAR(50) NOT NULL, DENTIFY ( 1 ProductTypeCodel ))	),



nis report describes how the [	Database tables will be created.		
ease select Create to procee	d or Cancel.		
Create Cance			
attem:			
Product Type	Table Product speci	fication	
Product	Table name: Product		
	Product is new		
	Table Structure		
	Attribute	Definition	Previous values
	ProductCode	Numeric (10), Not null	
	ProductName	Character (50), Not null, NLS	
	ProductPrice	Numeric (9.2), Not null	
	ProductStock	Numeric (4), Not null	
	ProductTypeCode	Numeric (4), Not null	
	Indexes		
	Name	Definition	Composition
	IPRODUCT	primary key Clustered	ProductCode
	IPRODUCT1	duplicate	ProductTypeCod

Note that, as it was previously explained, the *ProductTypeName* attribute is not included in the *Product* physical table that will be created, despite the fact that you included it in the *Product* Transaction structure (with the purpose of being shown in its layout).

If you agree with the Impact Analysis proposal, click on the Create button and GeneXus will start creating the programs needed to build the database (still inexistent), as well as the tables with their structures in that database. Next, GeneXus executes these programs and after creating the database and the tables, it will generate all the necessary lines of code - in the selected programming language - to obtain the application that will enable the users to insert, update and delete product types and products.



You are informed if the result was successful or if there were any errors or warnings. After that, you will be able to run and test the application. To do so, the Launchpad window will be opened offering a quick way to execute the defined objects:

Launchpad	×				
WEB APPS	NATIVE APPS	APIS			
Search					
MAIN OBJE	CTS				
		There a	re no main objects to display.		
RECENT OF	JECTS				
RECENT OE Name	BJECTS	Module	Туре	Description	
RECENT OE Name	B <b>JECTS</b>	Module	<b>Type</b> Transaction	<b>Description</b> Product	

#### Click on the *ProductType* link:

	Pharmacy
Product Type	
	< < > >  SELECT
Type Code	
Type Name	
	CONFIRM CANCEL DELETE
	CONTROL OFFICE

The Web browser is opened by default, and the page above is shown. It allows you to add, update and delete *product types*. Enter the first product type.



Since the *ProductTypeCode* attribute has the Autonumber property set to True, the users will not have to enter a value for the identifier because it will be numbered automatically. So, enter the product type name:

	Pharmacy
Product Type	
	I< < > >I SELECT
Type Code	0
Type Name	Cosmetics
	CONFIRM

After entering the product type name and clicking on the Confirm button, a message will be displayed to inform that the data was added successfully; meanwhile, the form is cleared and get ready to enter another product type:

	Pharmacy
Product Type	e iully added.
Type Code	< < > >  SELECT 【
Type Name	
	CONFIRM CANCEL DELETE



Enter the second product type:

	Pharmacy
Product Type	
	< < > >  SELECT
Type Code	0
Type Name	Medicines
	CONFIRM CANCEL DELETE

Click on the Confirm button and then you can browse the data to confirm that they were numbered:

	Pharmacy		
Product Type			
	<mark>()</mark> < > )	SELECT	
Type Code	1		
Type Name	Cosmetics		
		CONFIRM CANCEL DELET	E
	Pharmacy		
Product Type			
	K < 🕥 X	SELECT	
Type Code	2		
Type Name	Medicines		
		CONFIRM CANCEL DELET	re -



Now, execute the *Product* Transaction. To do so, go back to the Launchpad and execute the *Product* Transaction by clicking on its link:

Launchpad X				
WEB APPS NATIVE AF	PPS APIS			
Search				
MAIN OBJECTS				
	The	re are no main objects to display.		
RECENT OBJECTS				
Name	Module	Туре	Description	
Product		Transaction	Product	
ProductType		Transaction	Product Type	

Try adding the first product. You must indicate the product type. If you remember the product type code you can enter it. Another option is to select it from a list by clicking on the arrow.

Code	0
Name	STAR Muscular Pain
Price	20.00
Stock	120
Product Type Code	
Product Type Name	



		Pharmacy	
Product			
		< < >>  SELECT	
Code	0		
Name	STAR Muscular Pain		
Price	20.00 Selection	on List Product Type	×
Stock	120 Product T	vpe Code	
Product Type Code		0	Type Code Type Name
Product Type Name	Product Ty	ype Name	1 <u>Cosmetics</u>
		CANCEL	2 Preuktings
			Ŀ
			CONFIRM CANCEL DELETE

## Pharmacy Product $|\langle \langle \rangle \rangle|$ Select 0 Code STAR Muscular Pain Name Price 20.00 Stock 120 2 Product Type Code Product Type Name Medicines CONFIRM CANCEL DELETE



Now, try deleting the "Medicines" product type. To do so, select that product type and press the Delete button:

	Pharmacy
Product Typ Invalid delete, rela	e ated information in Product
	I< < > >I SELECT
Type Code	2
Type Name	Medicines
	CONFIRM CANCEL DELETE

A message informs you that the deletion cannot be performed because related data exists in Product (the STAR Muscular Pain medicine is a product that belongs to this product type).

#### THE APPLICATION GROWS UP

Now suppose that one of the requirements you were asked for is that you need to have a picture of each one of the products.

To do that, go back to the *Product* Transaction, and you have just to enter a new attribute called *ProductPhoto*:

Product ×				$\sim$	
Structure 🕺 Web Layout   Rules   Events   Variables   Help   Documentation   Patterns					
Name	Туре	Description	Formula	Nullable	
Product	Product	Product			
📍 ProductCode	Numeric(4.0)	Product Code		No	
···· 🖓 ProductName	Character(50)	Product Name		No	
····   ProductPrice	Price	Product Price		No	
····  ProductStock	Numeric(4.0)	Product Stock		No	
ProductTypeCode	Numeric(4.0)	Product Type Code		No	
ProductTypeName	Character(50)	Product Type Name			
ProductPhoto	Image	Product Photo		No	

The Image type enables you to store images.

The Web Layout is automatically updated, including the *ProductPhoto* attribute.



Press F5 and GeneXus will evaluate the impact caused by the new definitions in the Knowledge Base, and it will show the **Impact Analysis Report**:

organize Cancel				
	Table Product specifica	tion		
Product	Table name: Product			
	Product needs conversion			
	Warnings			
	A rgz0007 Attribute Produc	tPhoto does not allow nulls and does not have an Ini	tial Value. An empty default value	will be used.
	Information			
	<u>nfo0003</u> The reorganization The following operations Add not null attrib Add not null attrib	on for this table makes the schema not backward co are not backward compatible: ute <u>ProductPhoto</u> ute <u>ProductPhoto.Uri</u>	mpatible.	
	Table Structure			
	Attribute ProductCode ProductCode ProductPrice ProductPrice ProductStock ProductTrueCode New ProductPhoto New ProductPhoto OXI	Definition Numeric (4), Not null, Autonumber Character (50), Not null, NLS Numeric (92, Not null Numeric (4), Not null Image, Not null VarChen (2048), Not null	Previous values	Takes value from Product, ProductCode Product, ProductName Product, ProductPrice Product, ProductProduct Product, ProductProduct emptyvalue(ProductPhoto)
	Indexes			
	Name IPRODUCT	Definition primary key Clustered	Composition	2
	1000011071	duplicate	Dt. Brochust Tures	o

Remember: The Impact Analysis Report indicates the structural changes required in the database

By reading the report, you will see that the main title, in this case, informs that **"The Database needs to be reorganized"**. The term "Reorganize" implies the task of making changes to the database. In this particular case, the report indicates that the *Product* table must be updated.

By clicking on the Reorganize button, GeneXus will create and execute the programs that will change the database. Then, it will generate the necessary programs that correspond to the application itself.



MAIN OBJECTS							
		There are no m	ain objects to display.				
ECE	Name	Module	Туре	Description			
	Product		Transaction	Product			
	ProductType		Transaction	Product Type			

Note that you will be able to run the application again immediately, with the new definition included:

	Pharmacy
Product	
	< < > >  SELECT
Code	0
Name	
Price	0.00
Stock	0
Product Type Code	
Product Type Name	
Photo	CHANGE
	CONFIRM CANCEL DELETE



If you remember the product code you may enter it, or another option is to select it from a list by clicking on the SELECT button. From there, you can retrieve the "STAR muscular pain medicine" and upload its photo:

	Pharmacy
Product A Data has been successfu	illy updated.
	< < > >  SELECT
Code	
Name	STAR Muscular Pain
Price	20.00
Stock	120
Product Type Code	2
Product Type Name	Medicines
Photo	CHANGE
	CONFIRM CANCEL DELETE



#### ADDING BUSINESS RULES

In addition to all the automatic controls included by GeneXus in the applications it generates, sometimes users request some specific controls. In Transactions, the rules that must be complied with, or the controls that you are asked to validate, are defined in the **Rules** section.

If, for example, a requirement is not to allow storing of products without a name, GeneXus offers a rule called **Error** that will enable you to avoid that:



Press F5 and GeneXus will save and evaluate the new definitions included in the Knowledge Base. In this case, it will infer that it is not necessary to modify the database, so it will not show an Impact Analysis Report. GeneXus will generate the necessary code and after that, it will execute the application updated with the new definitions.



Run the *Product* Transaction. Note that if the product name is left blank, the rule you have defined is executed:

	Pharmacy
Product	
	I< < → →I SELECT
Code	0
Name	The product name     cannot be empty
Price	0.00
Stock	0
Product Type Code	
Product Type Name	
Photo	CHANGE
	CONFIRM CANCEL DELETE

There is another rule whose syntax is very similar to the Error rule. It is called **Msg** and the only difference between them is that if the condition is met, in this case the message is displayed as a notice or warning, and the user can continue working.

If, for example, you want to inform that the product's price has been left blank without forcing the user to enter it, you can add the following rule in the *Product* Transaction:





	Pharmacy
Product	
Code .	I< < >>I SELECT
Name	• • • • • • • • • • • • • • • • • • •
Price	0.00 A The product price is empty
Stock	0
Product Type Code	
Product Type Name	
Photo	CHANGE
	CONFIRM CANCEL DELETE

This set of rules could be written in any other order and the result at runtime would be exactly the same, as GeneXus decides when each one of the defined rules should be triggered (when the user leaves each involved field, if the condition is true, etc.).

Of course, GeneXus offers more available rules to define different kinds of validations and actions.

Each Transaction may need to have its own behavior rules defined.



#### DEFINING CALCULATIONS: FORMULAS

Applications are often required to make calculations that involve the values of specific attributes, constants and/or functions. For all these cases, GeneXus offers **Formulas**.

There are different possible ways to define formulas.

Let's start by learning what a **Global Formula** is. A global formula is a calculation you define associated with an attribute. Note that the Transaction structures contain a column labeled **Formula**:

ProductType ×				$\sim$
Structure 🕺 Web La	ayout Rules Events	Variables Help Docume	ntation Patterns	
Name	Туре	Description	Formula	Nullable
ProductType	ProductType	Product Type		
ProductTypeCo	de Numeric(4.0)	Product Type Code		No
ProductTypeNa	me Character(50)	Product Type Name		No

When a calculation is defined for an attribute in this column, this means that the attribute is virtual. In other words, it will not be physically created as a field in a table, because the value of the attribute will be obtained by doing the calculation every time it is needed.

Let's see this with an example. Suppose the pharmacy needs to know how many registered products there are of each product type at all times. So, let's define a new attribute in the *ProductType* Transaction with the purpose of defining it as a global formula:

ProductType* X				`
Structure * 📕 Web Layout   Rule	s Events Variables Help Doc	umentation Patterns		
Name	Туре	Description	Formula	Nullable
∃- 🔲 ProductType	ProductType	Product Type	·	
ProductTypeCode	Numeric(4.0)	Product Type Code		No
ProductTypeName	Character(50)	Product Type Name		No
ProductTypeProductQuantity	Numeric(9.0)	Product Type Product Quantity		No

Let's now define the calculation associated with the *ProductTypeProductQuantity* attribute.

GeneXus offers a formula called Count to calculate the pharmacy's need (there are many others, like Sum, Average, etc.).

ProductType* X						
Structure * 🐔 Web Layout   Rules   Events   Variables   Help   Documentation   Patterns						
Name	Туре	Description	Formula	Nullable		
ProductType	ProductType	Product Type				
ProductTypeCode	Numeric(4.0)	Product Type Code		No		
ProductTypeName	Character(50)	Product Type Name		No		
ProductTypeProductQuantity	Numeric(9.0)	Product Type Product Quantity	count(ProductName)			



The attribute referenced inside the parentheses of the formula provides GeneXus with the information of the table to be navigated to do the calculation (in the definition above, GeneXus knows that it has to count in the Product table). Then, if GeneXus detects a relation between the table it will navigate (Product) and the context where the formula attribute is defined (ProductType), it will only consider the related records for the calculation. In this example, *ProductTypeCode* is present in both contexts: where the formula is defined and in the table to be navigated for doing the calculation of the formula. So, only the products of each product type are counted and not all the products recorded in the navigated table, will be considered. If no relation is found, then GeneXus will do the calculation considering all records in the navigated table.

Press F5. You can see that no physical changes will be made to the database. GeneXus will only generate the necessary programs and you will get the Launchpad again:

Pro	ductType ×	Launchpad ×				\ \
WEB	APPS	NATIVE APPS	APIS			
Q Se	arch					
MAI	N OBJECTS					^
			There are no	o main objects to display.		
RECI	емт овјест	S				^
	Name	Мо	dule	Туре	Description	
	Product			Transaction	Product	
	ProductTyp	e <b>(</b>		Transaction	Product Type	



Execute the *ProductType* Transaction in order to see how the product quantity is always calculated for each product type at the time:

	IC C > > SELECT	
Type Code		
Type Name	Cosmetics	
Product Quantity		
		LETE
	CONFIRM CANCEL DELI	
	CONFIRM CANCEL DELI	
	Pharmacy	
reduct Tup	Pharmacy	
roduct Typ	CANCEL DEL	
roduct Typ	Pharmacy e	
roduct Typ	Pharmacy e I< < > >I SELECT 2	
<b>roduct Typ</b> Type Code Type Name	Pharmacy e I< < > >I SELECT 2 Medicines	

You can add more products in order to verify how the product quantity is always calculated at the time for each product type.



#### USING PATTERNS (FOR WEB AND FOR MOBILE DEVICES)

Patterns allow you to empower your applications even more, automatically.

Applying a pattern is really easy, and as soon as you do it, GeneXus creates objects, codes and settings to provide interesting behaviors without the need for us to program them.

Look at the Patterns section of a Transaction. For example, in the *ProductType* Transaction, select the Patterns section:



Note that two tabs are available, each one of them offering a different pattern to be applied to the same Transaction.

First of all, choose the *Work With for Web* tab, in order to see how simple applying a pattern is, and how quickly you obtain interesting results.



You only have to click on the Apply this pattern on save checkbox and save ( $\square$ ):



After that, if you look for the *ProductType* Transaction in the **KB Explorer**, you can see that several objects are located below the Transaction:



They were created by GeneXus when the Work With for Web pattern was applied.



Press F5 to view the results in runtime:

Search				
AIN OBJECTS				
	There ar	e no main objects to display.		
	incre di	e no man objecto to disploy.		
CENT OBJECTS				
Name	Module	Туре	Description	
		Transaction	Product	
Product				

Look at the last link. You're offered to "work with Product Type" and the *ProductType* Transaction will be called from there. Click on that link.

Pharmacy						
Product Types	INSERT	Q  Type Name				
Type Code	Type Name		Product Quantity	UDDATE		
2	Cosmetics Medicines		0	UPDATE	DELETE	

You can see that a page showing all the product types is opened. This page lets the users work with the product types with a wider range of features.

For example, click UPDATE for the first line:

Pharmacy					
Product Types	INSERT	<b>्रि</b>   Type Name			
Type Code	Type Name		Product Quantity		
1	Cosmetics		0	UPDATE	DELETE
2	<u>Medicines</u>		1	UPDATE	DELETE



	Pharmacy
Product Type	
Type Code	1
Type Name	Cosmetics
Product Quantity	0
	CONFIRM

You can see that the *ProductType* Transaction is opened offering to edit the details of the product type in that line. Let's edit the type name and confirm:

	Pharmacy	
Product Type	9	
Type Code	1	
Type Name	Cosmetics for Teens	
Product Quantity	0	
		CONFIRM



After the edition and confirmation, the application returns to the *Work With Product Types* page and you can see the change:

Pharmacy						
Product Types	INSERT Q Type Nan	ne		)		
Type Code	Type Name	Product Quantity	UDDATE	DELETE		
2	Medicines	1	UPDATE	DELETE		

The DELETE link offers the users to delete the product type in the line.

Meanwhile, the INSERT button located outside the grid, allows the users to add new product types.

	Pharmacy			
Product Types	INSERT ame			
Type Code	Type Name	Product Quantity		
1	Cosmetics for Teens	0	UPDATE	DELETE
2	Medicines	1	UPDATE	DELETE



By clicking on the INSERT button, the *ProductType* Transaction is opened, ready for adding a new product type. Press it in order to enter a new product type (remember in this case it's only necessary to enter the product type name because you have set the key attribute's Autonumber property = True):

	Pharmacy	
Product Type	2	
Type Code	0	
Type Name	Baby Care	
Product Quantity	0	
	CONFIRM	CANCEL

After the insertion, the application returns once again to the *Work With Product Types* page:

Pharmacy						
Product Types	INSERT Q Type Nam	1e				
Type Code	Type Name	Product Quantity				
3	Baby Care	0	UPDATE	DELETE		
1	Cosmetics for Teens	0	UPDATE	DELETE		
2	Medicines	1	UPDATE	DELETE		



Note that each product type name has a link. Click on the product type: *Medicines*.

Pharmacy					
Product Types	INSERT Q Type Name				
Type Code	Type Name	Product Quantity			
3	Baby Care	0	UPDATE	DELETE	
1	Cosmetics for Teens	0	UPDATE	DELETE	
2	Medicines	1	UPDATE	DELETE	

As you can see below, all the details of the selected product type are displayed in a first tab (General tab):

	Pharmacy	
PRODUCT TYPES		
Medicines		
General Product		
		UPDATE DELETE
Type Code	2	
Type Name	Medicines	
Product Quantity	1	



Another tab (*Product tab*) shows the list of products that belong to that product type:

	Pharm	асу		
<b>&lt; PRODUCT TYPES</b>				
Medicines				
General Product				
Code	Name	Price	Stock Photo	
1	STAR Muscular Pain	20.00	120	

The *Product tab* was automatically generated because **each product type has several related products**. If each product type had also several related data of other kinds, more tabs would have been generated in order to show each list of data related to the product type.

Now, let's go back to the Work With Product Types page by clicking on the link offered in the top left corner.

	Pharmacy			
Product Types				
Type Code	Type Name	Product Quantity		
1	Cosmetics for Teens	0	UPDATE	DELETE

Note that it's possible to search by name. This means that if, for example, the user types "C", only the product types that begin with this letter will be displayed:

Go back to GeneXus again. So far, you have only selected **Apply this pattern on save** in the *Work With for Web* tab of the *ProductType* Transaction, and after saving you have seen all the features that are automatically generated.



What you may not have noticed is this configurable tree:

ProductType ×
Structure 🐆 Web Layout   Rules   Events   Variables   Help   Documentation   Patterns
Patterns usable in this object (underlined means pattern is applied)
Work With for Web
Apply this pattern on save
V 5 Work With Pattern Instance
Transaction (ProductType)
V 😫 Level (ProductType)
P DescriptionAttribute (ProductTypeName)
V selection (Product Types)
🟲 modes (Insert, Update, Delete)
V 들 Attributes
🚱 ProductTypeCode
🚱 ProductTypeName
Rain ProductTypeProductQuantity
✓ L <sup>™</sup> Orders
V D Order (Type Name)
ProductTypeName
✓ Y Filter
Attributes
Received Product I ypeName
✓ ♥ Conditions ✓ ProductTraceNerror life %ProductTraceNerror when not %ProductTraceNerror la France ()
ProductlypeName like & ProductlypeName when not & ProductlypeName.lstmpty()
BradustTupeCode
Eived Data
RenductTupeName
Tab (General)
→ Attributes
R ProductTypeCode
R ProductTypeName
ReproductQuantity
Actions
Action (Update)
Action (Delete)
V 🔂 Tab (Product)
Transaction (Product)
🟲 modes (Insert, Update, Delete)
V 🔁 Attributes
🖓 ProductCode
ProductName
R ProductPrice
8 ProductStock
Si ProductPhoto

It has configurable nodes, sub-nodes and elements, so that you can customize the behaviors to be generated (i.e., change the search criteria).



Now let's apply the *Work With for Web* pattern to the *Product* Transaction, too. As explained before, you only have to open the *Product* Transaction and in its Patterns section, you have to select the *Work With* tab. Then, check the **Apply this pattern on save** option and save:



Press F5. GeneXus proceeds to generate the necessary programs and execute the application with the changes. Click on the shown link:

Launchpad X					
WEB APPS	NATIVE APPS	APIS			
Search					
<					
AIN OBJECT	S				
		-	and the second second		
ECENT OBJE	CTS	There are r	no main objects to display.		
RECENT OBJE	CTS	There are r Module	no main objects to display. Type	Description	
ECENT OBJE Name	CTS	There are r Module	no main objects to display. <b>Type</b> Web Panel	<b>Description</b> Products	

You can see the same query features that you already saw for the *Work With Product Types* page:

		P	harmac	:y				
Products				INSER	QName			
Code	Name	Price	Stock	Product Type Code	Product Type Name	Photo		
1	STAR Muscular Pain	20.00	120	2	Medicines		UPDATE	DELETE



### Let's insert a new product:

	Pharmacy
Product	
Code	0
Name	X Eyeshadow Palette
Price	25.00
Stock	30
Product Type Code	
Product Type Name	Cosmetics for Teens
Photo	
	CONFIRM

After the confirmation, the application returns to the *Work With Product* page:

Pharmacy								
Products				INSER	T Q Name			)
Code	Name	Price	Stock Pr	oduct Type Code	Product Type Name	Photo		
1	STAR Muscular Pain	20.00	120	2	<u>Medicines</u>		UPDATE	DELETE
2	<u>X Eyeshadow</u> <u>Palette</u>	25.00	30	1	Cosmetics for Teens	1000	UPDATE	DELETE



Since it's irrelevant to display the Product Type Code in the grid, let's remove it from the configurable tree which is taken into account to generate this *Work With for Web*:



Press F5 and GeneXus proceeds to save, generating only the necessary programs and executing the application with the changes.

Now, insert some products in the same manner as shown before (by pressing the INSERT button that invokes the *Product* Transaction).



The Work With Products lists all the products that have been added:

		Р	harmacy				
Products				INSERT Q Name			
Code	Name	Price	Stock Prod	uct Type Code Product Type Name	Photo		
4	Asterix Headache Medicine	20.00	100	2 Medicines		UPDATE	DELETE
3	Feeding Bottle	20.00	15	3 Baby Care	-	UPDATE	DELETE
5	LOVE Lipstick #18	8.00	60	1 Cosmetics for Teens		UPDATE	DELETE
6	<u>Magic Anti-</u> inflammatory painkillers	30.00	10	2 <u>Medicines</u>	And the second	UPDATE	DELETE
1	STAR Muscular Pain	20.00	120	2 <u>Medicines</u>		UPDATE	DELETE
8	WONDER Facial Cream	900.00	20	1 Cosmetics for Teens	-	UPDATE	DELETE
2	<u>X Eyeshadow</u> <u>Palette</u>	25.00	30	1 Cosmetics for Teens	1000	UPDATE	DELETE

Now, pay attention to the *Work With* tab offered for each Transaction. Let's apply it to the *ProductType* Transaction.

ProductType ×	
Structure 🖌 Web Layout 🛛 Rules 🛛 Events	Variables Help Documentation Patterns
Patterns usable in this object (underlined me	eans pattern is applied)
South State Work With State Work With	
Apply this pattern on save	
Level (ProductType)	🔯 Layout 🔀 Rules   🏲 Events   🍸 Conditions 🗟 Variables
List Detail	Application Bar Insert
Section (General)	✓ Ⅲ MainTable
	GRID         ProductTypeName             Image: Contract of the second se



In contrast to the *Work With for Web* pattern, instead of seeing a list of attributes to be included in the grid under the node, the Layout is already shown in this case.

Now, look at the **Detail** node. You can associate the term Detail with seeing the details of a particular line in the list.

The Detail node is composed of two sections: General and Product.

Like the functionality implemented by the *Work With for Web* pattern, the **General** section displays the data associated with the selected product type and the **Product** section displays all the products that belong to the product type inside a grid.

ProductType ×	
Structure 📕 Keb Layout Rules Events	Variables Help Documentation Patterns
Patterns usable in this object (underlined me	eans pattern is applied)
State of the second sec	
Apply this pattern on save	
😫 Level (ProductType)	😺 Layout 🔀 Rules   🏲 Events   🍸 Conditions 🗟 Variables
La Detail	- Application Bar Update Delete
section (General)	
Co Section (Product)	
	Type Code ProductTypeCode
	Type Name ProductTypeName
	Product Quantity ProductTypeProductQuantity
	🗌 Any Platform, View, Default Orientations 👻 🛟 Add Layout 😢 Delete Layout



ProductType ×				
Structure 📕 Web Layout 🛛 Rules 🛛 Events	s Variables Help Documentation	Patterns		
Patterns usable in this object (underlined m	eans pattern is applied)			
South State of the second				
Apply this pattern on save				
Level (ProductType)	😺 Layout 🔀 Rules 🟲 Events 🍸 Conditions & Variables			
📑 Detail	+ Application Bar			
Section (General)	✓ Ⅲ MainTable			
	GRID	ProductName		
🗌 Any Platform, View, Default Orientations 👻 🔂 Add Layout 😢 Delete Layout				

After applying this pattern and saving, note that there is a new object called *WorkWithProductType* under the *ProductType* Transaction in the **KB Explorer**:





If you look at both objects generated under the *ProductType* Transaction, you can notice that the *WorkWithProductType* doesn't have other objects under it (because it includes different sections to define the entire implementation inside it).

On the other hand, the *WorkWithWebProductType*, is a configurable instance; so, you can set and save that instance object and GeneXus generates other objects under it to provide the useful behaviors you have already seen.

Soon, you will see the *WorkWithProductType* in action.

The proposal now is to create a **Menu** object, which will be the first one the application will execute. It will show an icon, so that when the user taps on it, the object *WorkWithProductType* is executed.

As explained before, to create an object you only have to select **File > New > Object**. The following window is then opened and you must choose the category User Interface, so that the objects that GeneXus offers which belong in this category are shown:

lew Object	)
Select a Catego Data Manageme User Interface BPM Chatbots Resources Documentation Extensibility Deploy Reporting Test ALL	Select a <u>Iype:</u> Image: Web Component         Image: Design System       Image: Web Component         Image: Master Panel       Image: Web Panel         Image: Menu (mage: Web Panel)       Image: Web Panel         Image: Panel       Image: Web Theme         Image: Stencil       Image: Web Theme         Image: Web Component       Image: Web Theme         Image: Web Theme       Image: Web Theme         Image: Web Component       Image: Web Theme         Image: Web Theme       Image: Web Theme         Image: Web Component       Image: Web Theme         Image: Web Theme       Image: Web Theme         Image: Web Component       Image: Web Theme         Image: Web Theme       Image: Web Theme
Displays a set o	PharmacyMenu
Description:	Pharmacy Menu
Module/ <u>F</u> older:	Root Module × ···
	Create Cancel

By default, every **Menu** is created with its property **Main program = True,** so that this object becomes an executable object (that is to say, it is compilable and executable on its own).



Now, drag and drop the WorkWithProductType object from the KB Explorer to the Items node:



The shown Action is added and the following Event associated with the Action is automatically created:





To understand the code line inside the Event, let's review the node tree contained inside the *WorkWithProductType* object:



The object has a main node, *ProductType*, and under it you can find the nodes List and Detail respectively, so to call the List node, the complete syntax is:

PharmacyMenu* ×	
Menu * Events * Variables Documentation	
'WorkWithProductType'	
<pre>1 = Event 'WorkWithProductType' 2 WorkWithProductType.ProductType.List()</pre>	
3 EndEvent	

Change the Description property of the Action node in the Menu to "List of Product Types":



Now everything is defined and ready to run the Mobile application.



Since the *PharmacyMenu* object has its Main Program property set to True, you can execute it independently from the KB Explorer, by right-clicking the object and selecting **Run**:





The execution may be performed either in a Mobile device connected to your computer or in an emulator which will open directly in the computer in which you are working:

Android Emulator - GeneX	us-API31-X86:5554	
7:12 🌣		●▲ ■
Pharmacy Menu	I	
List of Product Types		
•	•	



When you tap the image, the list of product types the pharmacy offers is shown:



Note that the insert button on the top-right corner can be easily removed, as this application is for end-users, and they should only be able to view the different product types, neither edit them nor insert new products.



When you tap each product type (for example on "Cosmetics for teens"), the **Detail** is shown, along with its two sections GENERAL and PRODUCT:



Obviously, this is just a very simple demonstration and you can achieve much more sophisticated applications.



#### GENERATING WHAT YOU HAVE DEFINED SO FAR IN ANOTHER LANGUAGE AND/OR FOR A DIFFERENT DATABASE

As it has been mentioned before, one of GeneXus' great advantages is that it allows you to generate the same application for different platforms, generating code in different programming languages and/or storing the application data in different databases. All this information is defined in an **Environment**.

An **Environment** allows you to configure and store all the information **related to a specific implementation of your application** (the generators that you want to use to generate the Back-end of your application, the generators that will be used to generate the Front-end, the information of the database, etc.).

Select the Preferences window by clicking on the tab next to the KB Explorer tab:

HI Preferences	×
V 🎯 Pharmacy	
V T Pharmacy	
V 🖧 NETSQLServer	
✓ march Back end	
🔤 🔅 Default (.NET)	
🗸 🧮 Data Stores	
🖳 🕸 Default (SQL Server)	
E Services	
V 🔲 Front end	
Web (.NET)	
💮 Android	
Y Deployment	
> Patterns	
> 2 Workflow	
ig KB Explorer	

There is only one Environment defined **(.NET Environment)**. It was created automatically at Knowledge Base creation time when you chose .NET as the generation language. After that, it was completed when you pressed F5 for the first time and you entered the database name to be created and the database server. The sub nodes of the Environment have configurable properties.



It is possible to create more than one execution Environment for the same Knowledge Base. For example, it is common to create an Environment for development, where you connect to a database with test data and another Environment for production, where you define the server and database that you will use for the finished system.

You may also want to create, in the same Knowledge Base, a new Environment to generate everything for a different platform (for example, you may want to generate the code in Java and use another DBMS such as Oracle, MySQL, PostgreSQL, DB2 or others. For the generation of the Front end you could use Apple or Angular).

As the following image shows, to create a new Environment, you have to right-click on your node Environment, choose **New Environment**, and then choose the language, database, and configure the necessary properties.



To work with the new environment once it was created, you have to right-click on it and choose **Set As Current Environment**. It is possible to identify the active environment by the PLAY symbol.

Learn more about how to create a new Environment.



#### - Accessing External Databases

• You may need to access external databases from GeneXus applications. For example, in order to make an initial load, you may need to get data from an external database to the tables of the database associated with the Knowledge Base. After that, you may not need to stay connected to that external database, or, you may need to connect and always stay connected to a certain table or tables of one or more external databases (not just to read them, but also to access and change the data in them). GeneXus offers a "reverse engineering process" to connect to tables of external databases in order to achieve the needs described above.

#### - Collaborative Development Support

• GeneXus Server offers the option to upload a Knowledge Base to a server. After that, new developers can create a local copy from the Knowledge Base in the server, from any geographical point, when the need arises. They can work (always locally) and upload their changes to the server. Of course, there is a mechanism for conflict resolution. This solution offers a lot of advantages, including version control of the models in the Knowledge Base.

#### - Consuming and defining web services

- It is possible to consume web services developed by third parties from a GeneXus application, as well as to develop your own web services with GeneXus.
- Defining massive updates to the database and freely defining other types of processes.
- Defining interactive and personalized panels for both Web apps and Mobile apps.
- Designing & fine tuning UIs (User Interfaces)
  - As the user experience is extremely important, GeneXus offers the power to customize the user interface through specific generators for native apps, apps with web responsive design, web mobile, etc. It also offers a Cross-Platform Live Editing feature, which simplifies the process of applying Design to your application and Live Prototyping it.



#### - Deploying your app to production in Local Servers or Cloud Providers

• You can deploy your app to production just by clicking on one button.

#### - Documenting within the Knowledge Base

• GeneXus provides a Wiki-style Documentation editor, so that you can easily describe the Knowledge Base's purpose (in an object of the Documentation type - called Main - which every knowledge base has automatically created).

Moreover, all GeneXus objects have a Documentation tab, where you can describe the object's purpose as well.

When writing the documentation, you may include texts, images, links to attributes, objects, etc. Files can also be stored in the Knowledge Base as part of your documentation.

#### - Artificial Intelligence

- GeneXus provides capabilities for easily integrate Artificial Intelligence (IA).
- Chatbot generator
  - GeneXus includes a <u>Chatbot generator</u> to automatically build and deploy a chatbot to any of the supported Chatbot providers.

#### - Extensibility

- GeneXus allows the creation of specific extensions that let developers take advantage of different platform languages to create specific solutions and extend the capabilities of the GeneXus core.
  - https://training.genexus.com/en/learning/courses/genexus/v18/core
- Integrating external Systems and Data Sources into a GeneXus application
  - GeneXus ERP Connector for SAP makes the development of applications integrated to the SAP ERP possible, allowing you to complement the functionalities it offers.
- Managing Security
  - GeneXus offers a fully integrated security module called GeneXus Access Manager (GAM). By just enabling it, it solves authentication and authorization functionalities for both Web apps and Mobile apps.



#### - Modeling and automating business processes

• GeneXus has a suite of tools that allow modeling and automation of business processes, as well as an execution environment to manage them. The modeling tool GeneXus Business Process Modeler is based on the BPMN 2.0 standard, and it is aimed at those users whose objective is to model business processes. These diagrams can be integrated or created in the GeneXus developing environment to implement the automation stage in which, by using GeneXus, we associate the different objects in each task modeled in the processes. GXflow offers the execution, management, and monitoring tools for the end users. In this way, GeneXus offers what we know as GeneXus BPM Suite, which is the set of tools which enable the development of systems based on Business Process Management; that is to say, business process-oriented systems.

#### - Reporting

- Defining static reports (typical reports which can be printed, saved or just viewed on screen).
- Defining visual and dynamic queries.
  - You can create queries to the database, group data according to one or several criteria, make calculations, and finally show the result in different types of graphs, Pivot tables, and tables. To carry out these kinds of queries, GeneXus offers the Query object and the Query Viewer control.
  - Moreover, the <u>GXquery product</u> allows end users to dynamically carry out queries, based on the same data model of the Knowledge Base. This tool focuses on enabling data access and analysis on the system's actual operational database, and gives the user an intuitive interface from which he can create his own queries and later see them through the web interface and the mobile application, or integrated into Microsoft Office Excel.

#### - Sharing Development and Marketplace

• <u>GeneXus Marketplace</u> allows developers to share their User Controls, Extensions, Patterns, External Tools and External Objects created for and with GeneXus.

#### - Testing applications with GXtest

• When new functionalities or variations are implemented, it's necessary to check that what already worked (before the changes) continues to behave properly. This kind of task can become very tedious if the application grows a lot, as the number of things to test will increase each time, etc. GeneXus helps to automatize these tests through its **GXtest** software, which allows you to save sequences of operations to test. Then, the tests are reproduced automatically, verifying that the system still works properly.



#### NEXT STEPS

You have come so far knowing GeneXus, so the natural question is "What's Next?"

- First things first:
  - Access the following online course in order to continue learning:
    - If you haven't tried GeneXus yet, you can do that for free, following this link: <u>http://genexus.com/trial</u>
- **Dig Deeper:** GeneXus is a very comprehensive development platform, and there is so much for you to read and learn. You could start digging deeper both in:
  - The GeneXus Training site: <u>http://training.genexus.com/</u>
  - The GeneXus Wiki: <u>http://wiki.genexus.com/</u>
- Get GeneXus! We live to provide the best tool that simplifies software development, so we thank every new client as the first one. Please, get in contact with us through info@genexus.com or check the <a href="http://genexus.com/plans">http://genexus.com/plans</a> to see which one fits you the best.
- Be part of our Community: Once you are ready, you can join our ever-growing community through a great array of possibilities
  - Publish your work in our marketplace: <u>http://marketplace.genexus.com/</u>
  - Jobs in GeneXus: http://genexus.com/company/work-with-us?en
  - Opportunities in our Partners: <u>http://genexus.com/jobs/Opportunities?en</u>
  - Be part of the GeneXus Alliance: <u>http://genexus.com/partners</u>
  - Come to the next GeneXus Meeting near you: <u>http://genexus.com/meetings</u>



We really hope we hear from you soon!!

The Gene Kur Veam

Copyright © GeneXus S.A. 1988-2022.

All rights reserved. This document may not be reproduced by any means without the express permission of GeneXus S.A.The information contained herein is intended for personal use only.

Registered Trademarks: GeneXus is trademark or registered trademark of GeneXus S.A. All other trademarks mentioned herein are the property of their respective owners.



MONTEVIDEO - URUGUAY CIUDAD DE MÉXICO - MÉXICO MIAMI - USA SÃO PAULO - BRASIL TOKYO - JAPAN Av. Italia 6201- Edif. Los Pinos, P1 Hegel N° 221, Piso 2, Polanco V Secc. 8950 SW 74th Ct., Suite 1406 Rua Samuel Morse 120 Conj. 141 2-27-3, Nishi-Gotanda Shinagawa-ku, Tokyo, 141-0031