



GeneXus Trial: Tutorial

Copyright © Artech Consultores S. R. L. 1988-2012.

All rights reserved. This document may not be reproduced by any means without the express consent of Artech Consultores S.R.L. The information contained herein is intended for personal use only.

Registered Trademarks

Artech and GeneXus are trademarks or registered trademarks of Artech Consultores S.R.L. All other trademarks mentioned herein are the property of their respective owners.

Índice

Introduction	3
GeneXus Trial	3
Functional Restrictions	3
License Restrictions	3
Prototyping Server Restrictions	3
License Agreement	4
Technical Support	4
Installation and Authorization	5
Installation Requirements	5
.NET Generator Requirements	5
Smart Devices Generator Requirements	6
Authorization	7
GETTING STARTED: Step-by-Step Tutorial	9
Symbols Used	9
PART 1: Discovering GeneXus	10
What Is GeneXus?	10
PART 2: Your First Application	10
Designing Knowledge-based Applications	10
Step 0: Case Study	10
Step 1: Development Environment	11
Step 2: Creating a Knowledge Base and Defining the Environment	12
Step 3: Creating a Transaction Object	14
Step 4: Describing the Transaction Structure	15
Step 5: Defining Calculated Fields ⇔ Formulas	17
Step 6: Viewing the Data Model Inferred by GeneXus	18
Step 7: Viewing the Transaction Object Forms	20
Step 8: Running your Application	21
Step 9: Testing your Application	23
Step 10: Adding Business Rules ⇔ Rules	24
Step 11: Creating the Customer Transaction Object	26
Step 12: Reviewing the Changes Made in your Data Model	29
Step 13: Viewing the Specification Report	31
PART 3: Generating applications from development patterns	33
Using Patterns during Development	33
Step 14: Pattern for web applications	33
Step 15: Pattern for smart devices applications	35
Summary	43
Contacts and Resources	44
GeneXus Community	44
Support	44
How to buy	44
Recommended Links	44

INTRODUCTION

The objective of this document is to help you discover the potential of the knowledge-based development methodology proposed by GeneXus as you experience its main features:

- Automatic design of the data model.
- Automatic code generation
- Automatic maintenance of the database and code
- Multi-platform development and deployment

GENEXUS TRIAL

FUNCTIONAL RESTRICTIONS

The GeneXus Trial Version is completely functional and the available generator (.NET and Smart Devices) is authorized with a single Site Key (which expires 60 days after activation). However, some restrictions apply to the maximum number of GeneXus objects and attributes that may be created for a given Knowledge Base:

- 90 attributes
- 140 objects.

LICENSE RESTRICTIONS

It can be locally installed for a single user.

PROTOTYPING SERVER RESTRICTIONS

The generated applications have server-side components that in this version can only be run on web servers (Cloud Computing) provided by GeneXus International. For them to be run locally or on corporate platforms (on-premise), the Full version is required. The GeneXus Trial will expire 60 days after it has been requested, and the same applies to the application(s) and database(s) created by the user and residing in the prototyping server

LICENSE AGREEMENT

Conditions of Use of the GeneXus Trial Intellectual Property License

1. This agreement governs the intellectual property license of copies of **GeneXus Trial**, a knowledge-based intelligent tool which automatically designs, generates and maintains databases and applications.
2. **Artech Consultores S.R.L. (hereinafter Artech)** declares and the **USER** agrees that the intellectual creation of **GeneXus** and the **GeneXus**, **GXflow**, **GXplorer**, **GXportal**, **GXquery** and **Artech** names and logos, as well as any other trademark that Artech may launch in connection with **GeneXus**, regardless of whether or not they are registered, are the property of **Artech**. The current agreement does not imply, either directly or indirectly, any transfer of ownership nor does it entitle the **USER** to transfer the licenses, which are the subject matter hereof.
3. The **USER** agrees to use the **GeneXus Trial** without disclosing or using for his or her own benefit, any of the ideas and techniques on which **GeneXus** is based. In particular, the user agrees not to reverse engineer it in order to interpret its code, nor to enable others to do so.
4. In no event shall the **USER** be allowed to duplicate or disable the protection mechanisms against non-authorized use of **GeneXus**. **Artech** reserves the right to modify these mechanisms and/or add new ones at any time.
5. In compliance with the general terms, **Artech** grants the **USER** a license for the **GeneXus Trial** tool under the following conditions:
 - a) Each licensed copy will be able to work on one microcomputer. The **USER** agrees not to use any of them in more than one microcomputer at the same time.
 - b) This license is non-exclusive and non-transferable.
 - c) The **USER** will use the **GeneXus** programs and documentation exclusively for personal use, for evaluation purposes, and agrees not to provide copies of any of them to third parties.
 - d) In no event shall Artech be liable, either implicitly or explicitly, for any incidental damages to users relating directly or indirectly to the use of the **GeneXus Trial**.
 - e) The **USER** agrees to use the license in compliance with the instructions and specifications indicated in the materials associated with it.
 - f) The **GeneXus Trial** version will be active for a maximum of 60 days. Said version only includes the .NET and Smart Device generators. In addition, SQL Server is the only database supported in this version and it will allow creating a maximum of 90 attributes and 140 objects.
 - g) The **USER** agrees to use the **GeneXus Trial** to develop applications that DO NOT infringe any law or regulation at the international or local level in any territory.
 - h) In no event shall the **USER** be allowed to develop an application with the GeneXus Trial and deploy it to an application server other than the one provided by the tool, which is indicated in the 'Deploy Server URL' property of the IDE.
6. By installing this licensed software product you accept all the terms and conditions of this agreement.

TECHNICAL SUPPORT

If you need assistance to install and authorize your trial version, contact: gxtrial@genexus.com

If you are in the United States or Canada you may send support requests to: gxtrial.usa@genexus.com

To learn about the various GeneXus technical support and training services and resources, visit: genexus.com/support and genexus.com/training

For additional information contact your local distributor genexus.com/distributors or contact us at info@genexus.com

INSTALLATION AND AUTHORIZATION

INSTALLATION REQUIREMENTS

The GeneXus Trial Version includes the following products:

- **GeneXus Modeler**
It is an Integrated Development Environment (IDE) for designing, developing and maintaining business applications regardless of the production platform used.
- **GeneXus Generators**
GeneXus generates native code for the market's leading platforms. For a complete list of GeneXus Generators, visit: genexus.com/technologies. The generators provided with the GeneXus Trial Version are the .NET Generator and the Smart Devices Generator (iOS, Android, Blackberry).

Below you will find a list of the hardware and software necessary to run GeneXus and the applications generated with GeneXus

Hardware Requirements	Processor: 1 GHz minimum (multi-core recommended)
	Memory: 1 GB RAM minimum (2 GB recommended)
	Hard disk: Minimum of 300 MB of disk space for the installation. To create GeneXus applications, you will need additional space or a shared disk for the Knowledge Bases and generated code.
Software Requirements	Microsoft Windows 2000, XP SP2 or higher
	Microsoft .NET Framework 3.5 SP1 or higher
	Microsoft Internet Explorer 6.0 SP1 or higher
	Microsoft SQL Server 2005 or higher (Express, Standard or any other Edition) ¹

¹ Microsoft .NET Framework 3.5 SP1 and Microsoft Internet Explorer 8.0 will be automatically installed upon installing GeneXus Trial.

² Microsoft SQL Server Express Edition is the free, redistributable version. If Microsoft SQL Server is not installed on your machine when you install GeneXus Trial, you will be given the option to install it (the "sa" user will be created with the password "genexustrial").

.NET GENERATOR REQUIREMENTS

[Microsoft .NET Framework 3.5 or higher](http://www.microsoft.com/net/framework)

SMART DEVICES GENERATOR REQUIREMENTS

Android

- [Java JDK version 6](#)
- [Android SDK](#) (Google API 7 and Google API 17)

More information: [Android Prerequisites](#)

Blackberry

- [Java JDK version 6](#)
- [Blackberry JDE 5.0](#)
- [Blackberry Simulator](#)

More information: [Blackberry Prerequisites](#)

Apple (iOS)

- Knowledge Base Navigator (iOS Device):

If you don't have a Mac to prototype the generated applications, you will be able to prototype directly on the device (iPhone, iPad, iPod) using the [Knowledge Base Navigator](#) available in the [Apple Store](#).

- Xcode (Mac):

If you have a Mac, you can prototype the generated applications with the iPhone / iPad simulator or directly on the device.

More information: [iOS Prerequisites](#)

AUTHORIZATION

1. Run the setup file of the GeneXus Trial Version (GenexusTrial.exe), either from the Windows Start menu or by selecting the corresponding option on your Trial Version CD.

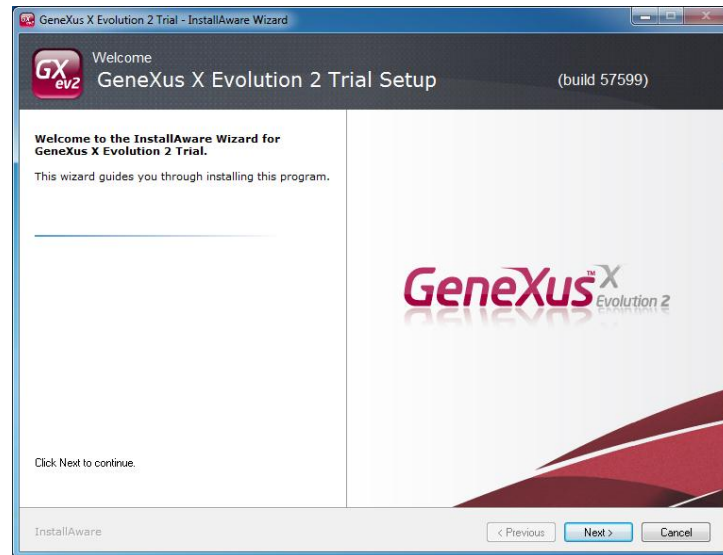


Figure: GeneXus Trial Version Installation Wizard

2. Follow the installation setup instructions.
3. The first time that you use the GeneXus Trial you will have to authorize the product to be able to run it. Authorization can be performed Online or By Mail. Online authorization is recommended if you have Internet access.

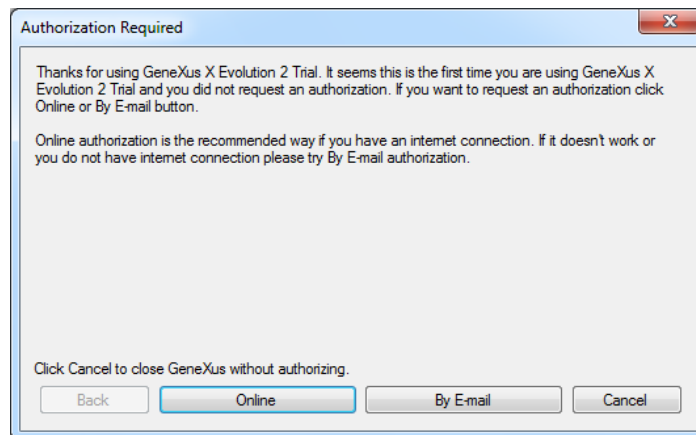


Figure: GeneXus Trial registration

4. To authorize it, you must have a [GXTechnical](#) user account, by the “Create User” button you can create your user:

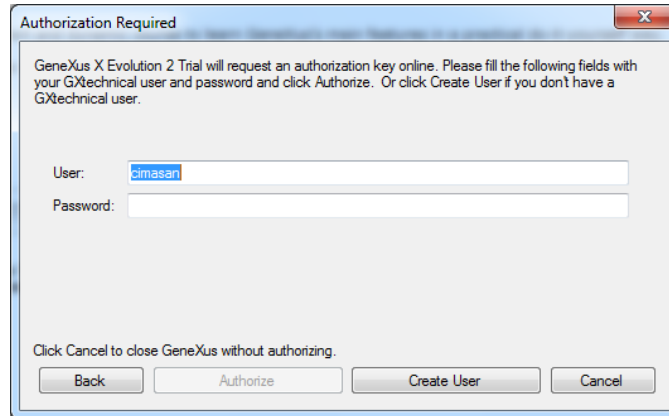


Figure: Enter username and password

5. Once you select the **Online** option and complete all the required data, the GeneXus Trial Version will be immediately activated for a period of 60 days.
6. If you have selected the option labeled **By Mail**, an email will be created with the corresponding Site Code to request your license. You will receive an email containing your **Site Key**. To do this, you will have to use the email address associated with your [GXTechnical](#) account.
7. The activation Site Key will be sent to the same email address. Copy and paste your **Site Key** in the Site Key field of the Authorization Required window and click **Continue**.
8. You are now ready to start using your GeneXus Trial!
9. If you need assistance to activate your GeneXus Trial, contact gxtrial@genexus.com

GETTING STARTED: STEP-BY-STEP TUTORIAL

The objective of this tutorial is to provide you with your first hands-on experience with GeneXus, as you learn the fundamentals of the GeneXus Methodology. Eventually, you will be able to create complex business applications faster than you ever thought.

In this step-by-step exercise you will create a simple business application and install it on one of the platforms supported by the GeneXus Trial: .NET. In addition, you will be able to generate the application for smart device platforms such as Android, iOS and Blackberry. For a complete list of platforms supported by GeneXus, visit: <http://www.genexus.com/technologies/>

As you work on this tutorial, you will experience the key features of GeneXus:

- Knowledge-based application design
- Intelligent database generation
- Automatic code generation
- Fully functional applications
- Incremental development and automatic maintenance of the application
- Multi-platform development

SYMBOLS USED



This symbol introduces a concept that is key to following this tutorial.



This symbol introduces a NOTE.

PART 1: DISCOVERING GENEXUS

WHAT IS GENEXUS?

Basically, GeneXus is a program that makes programs.

It is a tool that starts from users' views and encapsulates knowledge in what we call a Knowledge Base. GeneXus systematizes that knowledge and then automatically designs, builds and maintains the database and programs.



In sum, GeneXus is a **knowledge-based** tool that automatically **designs, generates** and **maintains** the programs and database to achieve rapid development of critical applications on multiple platforms.

PART 2: YOUR FIRST APPLICATION

DESIGNING KNOWLEDGE-BASED APPLICATIONS

In this section you will learn to create a GeneXus Knowledge Base and design an application based on the users' requirements. In this particular case, we will first design an invoice and other components required in a Sales Management System.



BUSINESS ANALYSTS VS. DEVELOPERS

Understanding the end user's needs is one of the few software development tasks that can't be automated. This is why we refer to GeneXus developers as **Business Analysts** rather than programmers, coders or developers.



PROGRAMMING VS. DECLARING THE APPLICATION

The GeneXus Methodology is based on **describing** the end users' entities (both tangible and intangible real-life objects) that your application deals with. This is done by **describing** the end users' views about these entities, with a high abstraction level. For this reason, we will use **declarative programming**. Thus, **Business Analysts** will describe the reality to have GeneXus create the data model in a specific database and build the application programs needed to provide the required functionalities. When this reality changes, Business Analysts will simply have to describe the new reality and GeneXus will make the necessary changes to the data model and programs to represent this new reality.

STEP 0: CASE STUDY

Let's suppose that a company needs an application to manage its billing system. Currently they don't have one and billing is done manually, using large catalogs of product codes. We will be working with invoices, customers and products as we introduce the basics of the GeneXus methodology.

Over the course of this tutorial, you will be able to describe the reality presented in GeneXus to generate a typical billing system on a Web platform, using the .NET Generator and a SQL database. In addition, the corresponding Smart Device application will be generated for you to query or change your clients' details.

STEP 1: DEVELOPMENT ENVIRONMENT

Upon opening GeneXus, you will see an interface similar to the figure below. This interface is called IDE (Integrated Development Environment) and it is intuitive, easy to use and parameterizable by each developer.

It is divided into windows:

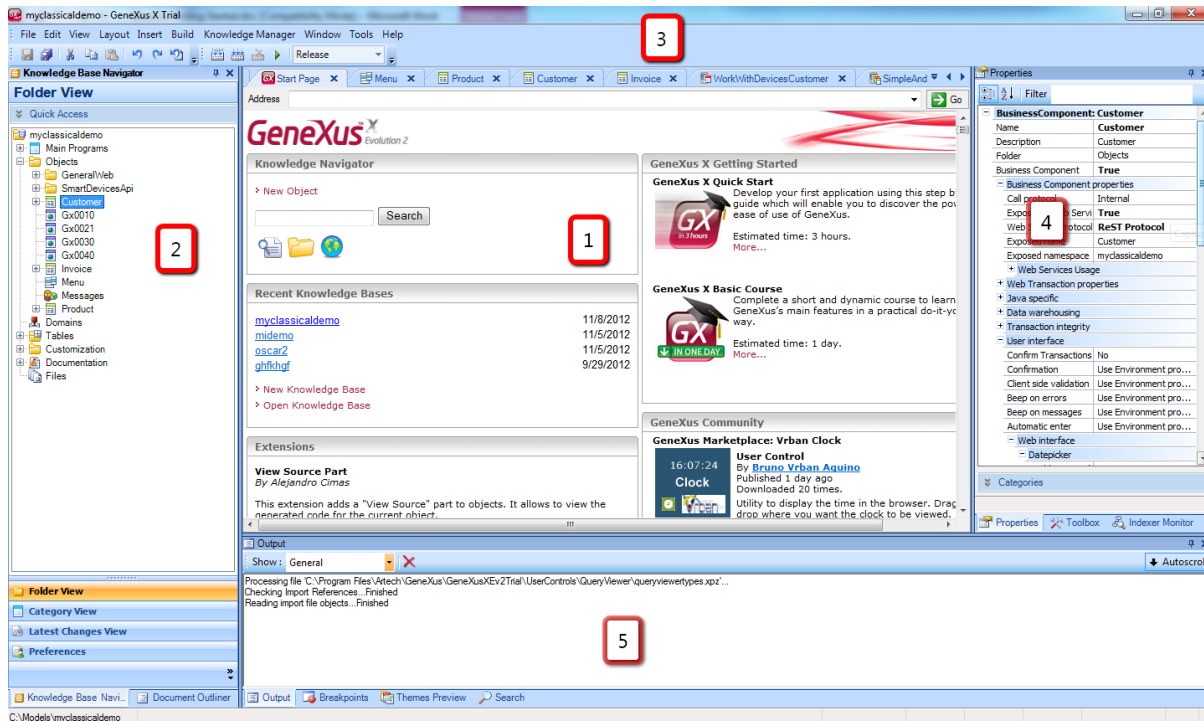


Figure: Development Environment (IDE)

1. **Main Window (Start Page):** It is made up of the Start Page, which consumes RSS and dynamically displays technical information about the tool and user community. To obtain this information you need an Internet connection. If your machine is not connected to the Internet, you will not be able to read the news published on the GeneXus Community.
2. **Knowledge Base Navigator**
3. **Toolbar**
4. **Properties window**
5. **Output**

STEP 2: CREATING A KNOWLEDGE BASE AND DEFINING THE ENVIRONMENT

The first step to create an application using GeneXus is to create a Knowledge Base and define the work environment.



KNOWLEDGE BASE (KB)

It is a repository that contains all the information needed to generate an application on multiple platforms. In other words, it is a *repository of the entire description of reality*.



ENVIRONMENT

To generate and maintain a working application on a specific software platform, we have to define an **Environment**, which integrates everything related to the execution platform (Generator, database access, user interface and other properties of this platform). To this end, we specify a DBMS, a target language and some additional parameters for each Environment. GeneXus will generate and maintain the database schema and all programs on the selected platform. Thus, the GeneXus analyst doesn't need a deep knowledge of the target platform.

If this is the first time that you open the GeneXus Trial and have never before run a commercial version of the product, you will be notified that GeneXus will automatically create a Knowledge Base. The first time that you open the trial version, a Knowledge Base will be automatically created on a .NET environment with SQL.

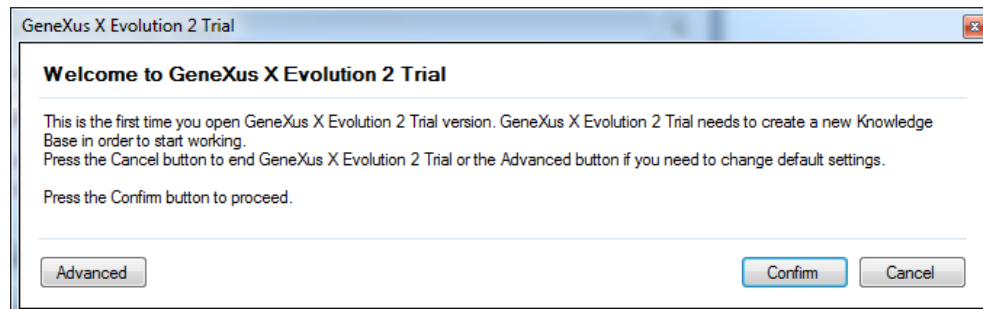


Figura: Welcome to GeneXus X Evolution 2 Trial

Pressing the **Advanced** button you can specify the Knowledge Base name and directory:

Welcome to GeneXus X Evolution 2 Trial

This is the first time you open GeneXus X Evolution 2 Trial version. GeneXus X Evolution 2 Trial needs to create a new Knowledge Base in order to start working.
Press the Cancel button to end GeneXus X Evolution 2 Trial.

Press the Confirm button to proceed.

Name:
Taller

Directory:
C:\Models

Language:
English

Knowledge Base Storage

Server name:
CIMASAN-PC\SQLEXPRESS

Database name:
GX_KB_Taller

Collation:
SQL_Latin1_General_CP1_CI_AS

☒ Use Windows NT Integrated security
☐ Use a specific user id and password

User id:
Password:

☒ Save Password
☒ Create datafiles in Knowledgebase folder

Knowledge Base will be created at

Folder: Knowledge Base already exists.
Server: CIMASAN-PC\SQLEXPRESS
Database: GX_KB_Taller

Basic Confirm Cancel

Figura: Create new Knowledge Base - Advanced

STEP 3: CREATING A TRANSACTION OBJECT

Our objective will be to define our users' views in GeneXus objects.



TRANSACTION OBJECT

It represents the real-life objects that your application deals with. The defined transactions are used to infer the application's data model (3rd normal form). GeneXus also uses the transaction object to generate the application program that will allow the end user to interactively insert, delete and update records on the physical database.

Once the Knowledge Base has been created, the next step will be to create the first transaction which represents an invoice. To do so, follow the steps below:

1. Right-click on **Objects/New/Object** or open the **FILE/NEW OBJECT** menu option. You can also create a new object by using the **CTRL+N** shortcut keys.
2. Select the type of object that you want to create:
Transaction
3. Enter a name for the Object: *Invoice*.
4. Click on **CREATE**.

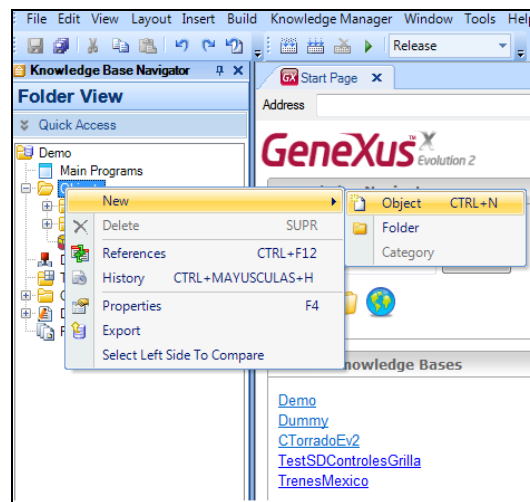


Figure: Creating a new object

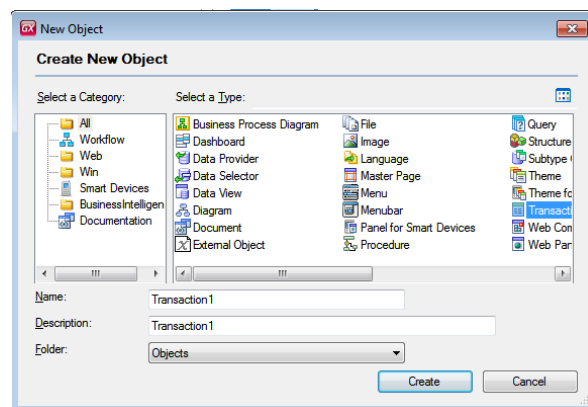


Figure: New Object dialog box

STEP 4: DESCRIBING THE TRANSACTION STRUCTURE

The next step will be to describe the invoice object by defining which attributes integrate it and how they are related.

1. Enter the attributes name, data type and description in the Structure tab of the Invoice transaction, as shown on the table below. Use the TAB key to move between the attribute name, data type and description. Use the ENTER key to add a new attribute.

ATTRIBUTE	TYPE	DESCRIPTION
InvoiceId	Numeric(4.0)	Invoice ID
InvoiceDate	Date	Invoice Date
CustomerId	Numeric(4.0)	Customer ID
CustomerName	Character(20)	Customer Name
So far, we have entered the invoice header fields. We will now enter the lines. To do so, press CTRL + Right Arrow to add a New level to the data structure.		
ProductId	Numeric(4.0)	Product ID
ProductName	Character(20)	Product Name
ProductPrice	Numeric(8.2)	Product Price
InvoiceProductQuantity	Numeric(4.0)	Product Quantity
InvoiceProductTotal	Numeric(8.2)	Product Total
Press ENTER and CTRL + Left Arrow to return to the header level and enter the footer details.		
InvoiceSubtotal	Numeric(8.2)	Invoice Subtotal
InvoiceTax	Numeric(8.2)	Invoice Tax
InvoiceTotal	Numeric(8.2)	Invoice Total

By default, the first attribute of each level is defined as primary key of that level, but this can be changed by right-clicking on the attribute and selecting the Toggle Key option (CTRL + K). Primary Key attributes are identified with a little key to the left of their names (). In this example, InvoiceID is the first level identifier, and the ProductID attribute is the second level identifier. This means that for a given invoice number (InvoiceId), the ProductID attribute value won't be repeated in different lines.

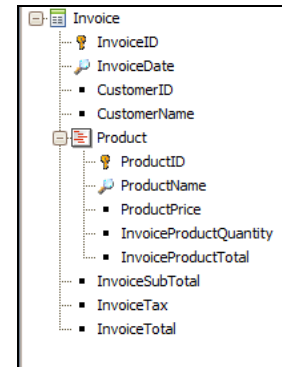
Name	Type	Description
Invoice	Invoice	Invoice
InvoiceID	Numeric(4,0)	Invoice ID
InvoiceDate	Date	Invoice Date
CustomerID	Numeric(4,0)	Customer ID
CustomerName	Character(20)	Customer Name
Product	Product	Product
ProductID	Numeric(4,0)	Product ID
ProductName	Character(20)	Product Name
ProductPrice	Numeric(8,2)	Product Price
InvoiceProductQuantity	Numeric(4,0)	Invoice Product Quantity
InvoiceProductTotal	Numeric(8,2)	Invoice Product Total
InvoiceSubTotal	Numeric(8,2)	Invoice Sub Total
InvoiceTax	Numeric(8,2)	Invoice Tax
InvoiceTotal	Numeric(8,2)	Invoice Total

Figure: Invoice structure

2. Save the new transaction structure by pressing the **Save** () button on the Standard toolbar. If you cannot view this bar, display it by right-clicking on the Menu bar (Edit option).

You have just created the structure of an Invoice transaction composed of two levels:

- A basic level (Invoice), where we specify all the information needed for the Invoice Header.
- A nested level, where we specify the information that will be repeated in every invoice line.



UNIVERSAL RELATIONSHIP ASSUMPTION (URA)

A key element of the GeneXus Methodology is the assumption that attributes with the same name are the same attribute. This is called Universal Relationship Assumption (URA), which says that:



- Everything that is conceptually equal should have the same name.
- Different concepts **MUST NOT** have the same name.

This will enable us to use the same attribute in other GeneXus objects (other transactions, procedures, reports, and so on) just by referring to its name. GeneXus establishes the relationships (foreign keys) between the tables of the data model based on the attributes' names.

STEP 5: DEFINING CALCULATED FIELDS ⇒ FORMULAS

FORMULAS

Formulas are attributes that can be inferred from the value of other attributes. A formula attribute is defined in the same way as a “normal” attribute, i.e., it has a name, data type, and description, plus a formula that defines how it is calculated.



- Unless otherwise specified, attributes defined as formulas will not be stored in the database (they are **virtual attributes**).
- Formulas are **global**; they are valid across the entire knowledge base and not just within the transaction where they are defined. This means that the formula is calculated each time that the attribute is invoked from a transaction or from any other GeneXus object (Reports, Work Panels, etc.)
- **User-Defined Variables.** These variables cannot be involved in a formula because they are local with respect to the objects where they are defined, and they have no value outside them.

You will now define the following formula attributes:

InvoiceProductTotal=ProductPrice * InvoiceProductQuantity

InvoiceSubTotal = SUM(InvoiceProductTotal)

InvoiceTax= InvoiceSubTotal * 0.11 (or the corresponding number)

InvoiceTotal = InvoiceSubtotal + InvoiceTax

1. Double-click on the **Formula** field of the InvoiceProductTotal attribute (to the right of the attribute description).
2. Type the following expression: “*ProductPrice * InvoiceProductQuantity*”. You can also right-click on the formula field to open the Formula Editor.
3. Repeat Steps 1 and 2 for the rest of the formulas listed at the beginning of this section.
4. Click **Save** to save the new formulas.

When you finish, you should have an invoice description that looks as follows:

Name	Type	Description	Formula
Invoice	Invoice	Invoice	
InvoiceID	Numeric(4.0)	Invoice ID	
InvoiceDate	Date	Invoice Date	
CustomerID	Numeric(4.0)	Customer ID	
CustomerName	Character(20)	Customer Name	
Product	Product	Product	
ProductID	Numeric(4.0)	Product ID	
ProductName	Character(20)	Product Name	
ProductPrice	Numeric(8.2)	Product Price	
InvoiceProductQuantity	Numeric(4.0)	Invoice Product Quantity	
f InvoiceProductTotal	Numeric(8.2)	Invoice Product Total	ProductPrice * InvoiceProductQuantity
f InvoiceSubTotal	Numeric(8.2)	Invoice Sub Total	sum(InvoiceProductTotal)
f InvoiceTax	Numeric(8.2)	Invoice Tax	InvoiceSubTotal * 0.11
f InvoiceTotal	Numeric(8.2)	Invoice Total	InvoiceSubTotal + InvoiceTax

Figure: Invoice structure with formulas

STEP 6: VIEWING THE DATA MODEL INFERRED BY GENEXUS

You can view the data model inferred by GeneXus and even modify it.



INTELLIGENT DATA MODEL GENERATION

Whenever you click on the Save button, GeneXus infers the optimal data model (in 3rd normal form, with no redundancies) that supports the end user entities represented by your GeneXus transaction objects. Based on this data model, GeneXus will generate a physical database when you define a target DBMS for a model.



INTELLIGENT DATA MODEL GENERATION

The structure of the transaction objects determines the tables and indexes to be created:

- The names of the tables and indexes are automatically assigned by GeneXus with the transaction name, but you can modify them if needed.
- GeneXus infers a data model in 3rd normal form, with no redundancies. However, you can define redundancies that will be automatically managed by GeneXus.
- The primary key of a table corresponding to an N level transaction is obtained by concatenating identifiers of the previous N-1 nested levels with the identifier of the N level.

1. In the left-hand menu, select **Tables**.
2. GeneXus will display the structure of data inferred from the transaction.

Name	Type	Description	Formula
Invoice Structure		Invoice	
InvoiceID	Numeric(4,0)	Invoice ID	
InvoiceDate	Date	Invoice Date	
CustomerID	Numeric(4,0)	Customer ID	
CustomerName	Character(20)	Customer Name	
Logical Attributes			
f InvoiceSubTotal	Numeric(8,2)	Invoice Sub Total	sum(InvoiceProductTotal)
f InvoiceTax	Numeric(8,2)	Invoice Tax	InvoiceSubTotal * 0.11
f InvoiceTotal	Numeric(8,2)	Invoice Total	InvoiceSubTotal + InvoiceTax

Figure: Invoice Table

Name	Type	Description	Formula
InvoiceProduct Structure		Product	
InvoiceID	Numeric(4,0)	Invoice ID	
ProductID	Numeric(4,0)	Product ID	
ProductName	Character(20)	Product Name	
ProductPrice	Numeric(8,2)	Product Price	
InvoiceProductQuantity	Numeric(4,0)	Invoice Product Quantity	
Logical Attributes			
f InvoiceProductTotal	Numeric(8,2)	Invoice Product Total	ProductPrice * InvoiceProductQuantity

Figure: InvoiceProduct Table

In the Database Listing above you can see that GeneXus automatically inferred a normalized data model, creating two tables to support the Invoice transaction object, Invoice (the invoice header) and InvoiceProduct (the invoice lines), with the following structure:

<u>Invoice</u>	<u>InvoiceProduct</u>
InvoiceID	InvoiceID
InvoiceDate	ProductID
CustomerID	ProductName
CustomerName	ProductPrice
	InvoiceProductQuantity

Note that:

- The primary key of the InvoiceProduct table is formed by two attributes: InvoiceID and ProductID (the concatenation of the first level identifier, InvoiceID, with the second level identifier, ProductID).
- GeneXus automatically eliminated from the tables the attributes that had been defined as formulas and converted them to global formulas so that they can be accessed from anywhere within the knowledge base).
- In the Invoice table:
 - No two invoices can have the same InvoiceID.
 - For each InvoiceID there is **only one** value for InvoiceDate, CustomerID and CustomerName.
- In the InvoiceProduct table:
 - No two invoice lines can have the same InvoiceID and ProductID.
 - For each pair of InvoiceID and ProductID there is **only one** value for ProductName, ProductPrice and InvoiceProductQuantity.

STEP 7: VIEWING THE TRANSACTION OBJECT FORMS

Look at the default Web form that has been automatically generated by GeneXus for the transaction object you just created (to do this, you have to be positioned inside the Invoice object).



WEB FORM

After saving a new Transaction Object, GeneXus automatically creates a predetermined Web Form to specify how end users will access the data in the application. These forms can later be customized by the business analyst.

To view the Web form, follow the steps below:

1. Select the **WebForm** tab in the Invoice transaction.

The screenshot shows a web form titled "Invoice" within a design environment. The form contains several input fields and a table. At the top, there's a red error message: "Errorviewer: ctlError". Below this, there are fields for "ID" (labeled "InvoId"), "Date" (labeled "InvoiceDe"), "ID" (labeled "Custo"), and "Name" (labeled "CustomerName"). A section titled "Product" contains a table with columns: "ID", "Name", "Price", "Product Quantity", and "Product Total". The table has a header row with values "Product", "ProductName", "ProductPrice", "InvoiceQ", and "InvoiceProd", followed by three empty rows. Below the table, there are fields for "Sub Total" (labeled "InvoiceSu"), "Tax" (labeled "InvoiceTe"), and "Total" (labeled "InvoiceTc"). At the bottom, there are three buttons: "Confirm", "Cancel", and "Delete". The design environment's toolbar at the bottom shows tabs for "Structure", "Web Form", "Win Form", "Rules", "Events", "Variables", "Help", and "Doc".

Figure: Invoice web form

"Error Viewer: ctlError" is the default control where error messages are displayed. You can place it anywhere in the form and configure properties for it. The possible messages are those that are displayed in the Msg and Error rules, and GeneXus' automatic controls (i.e., referential integrity, data type errors, etc.).

These forms will enable end users to enter new invoices that will be inserted as new records in the corresponding tables. Users will also be able to update or delete existing invoices, provided they are authorized to do that.

The GeneXus analyst does not need to program any of these actions because they are implicit in the transaction logic. GeneXus will automatically generate the corresponding native code in the selected language.

Remember that when you define transactions in GeneXus you are:

- Explicitly: Describing the user interface for displaying and capturing data.
- Implicitly: Designing the application's data model (tables, indexes, etc.)

STEP 8: RUNNING YOUR APPLICATION

GENERATING THE DB IN THIRD NORMAL FORM



- GeneXus generates the executable programs required to create its database in the selected DBMS based on the inferred data model.
- When you're **updating** your data structure, GeneXus generates the executable programs required to reorganize the database; that is, it creates a new schema and converts the data of the old schema to the new one.
- In these cases, a **Database Creation Report** or an **Impact Analysis Report**, respectively, will be displayed, showing you what GeneXus will do.

In this step we will run the application in .NET and SQL Server will be used as DBMS.



PROTOTYPING IN THE CLOUD (DEPLOY TO CLOUD)

- GeneXus allows us to prototype our applications in a server located in the cloud, so that we can automatically access them from any device, any time. For more information, read [Easy prototyping: Deploy to cloud](#)

The application server (Internet Information Services) and the database server (SQL Server) will reside in the application server of the prototyping server in the cloud assigned to the GeneXus Trial. Upon running the application, the necessary programs to run it locally are generated and will be automatically uploaded to the cloud. The **commercial version allows you to use any of the DBMS supported by GeneXus, and to prototype your applications in your own application and database servers.** You can read the complete list of generators and DBMS supported by GeneXus here: <http://www.genexus.com/technologies>

Database Creation Report: This is the report that describes the database schema that GeneXus will generate in the selected DBMS. It contains all the information concerning the inferred data model and the database schema proposed for generation. The information about each table is divided into five sections:



- **Header:** Contains the name of the table, the actions to be performed on it, warnings, and errors. If the data model contains errors, the Reorganization button will be disabled.
- **Table Structure:** Shows the attributes of the table, their relationships, and the actions to be performed on them.
- **Indexes:** Describes the table indexes that GeneXus uses to maintain the referential integrity of your database as well as to access the tables efficiently.
- **Foreign Key Restrictions:** Describes the table integrity restrictions.
- **Statements:** Describes the orders that will be executed.

1. Press F5 or go to the BUILD /RUN DEVELOPER MENU option on the menu.
2. Enter the same username and password that you used to authorize GeneXus Trial.

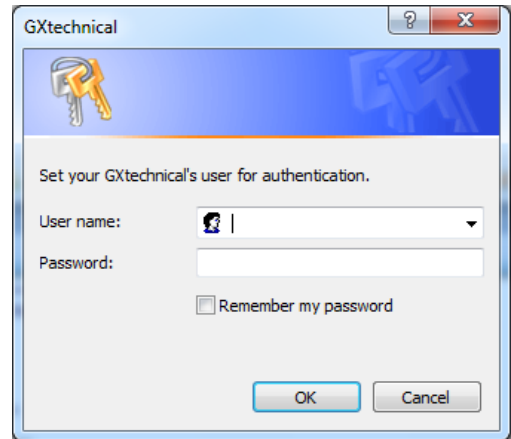


Figure: User name / password

3. GeneXus will show the database creation report.

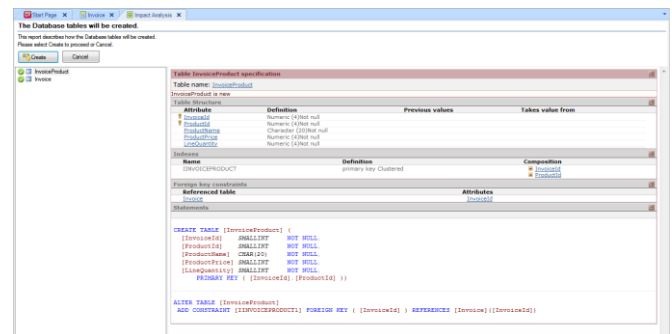


Figure: Database creation report

4. Click on **CREATE**. GeneXus will write the code for creating the necessary tables and programs in the selected language to access said DB.

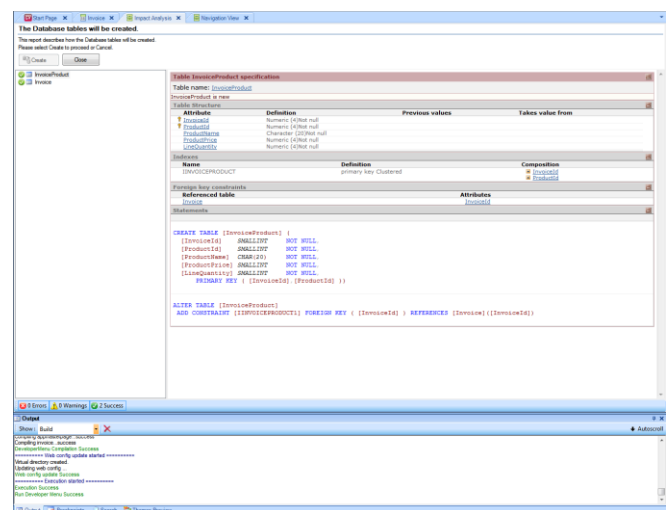


Figure: Create database

STEP 9: TESTING YOUR APPLICATION

1. The Developer Menu is an XML file that features all your executable objects. It is an auxiliary menu for prototyping your application. Click on the **Invoice** option.
2. Enter a few invoice instances. Thanks to AJAX technology, the formulas are calculated automatically, without having to reload the whole page.
3. Once you have finished click on the **Close** button.

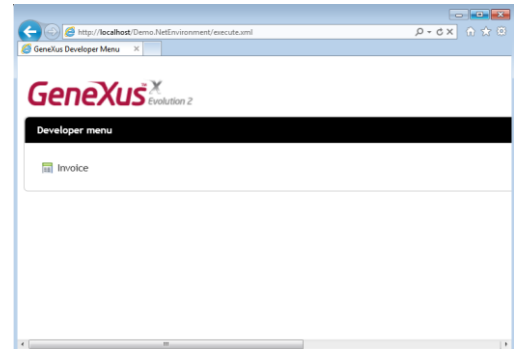


Figure: Developer menu

 A screenshot of the GeneXus application interface. At the top is the 'Application Header' with navigation tabs: 'First Option', 'Second Option', 'Third Option', and 'Fourth Option'. Below the header, the 'Invoice' form is displayed. It includes fields for 'ID' (value: 1), 'Date' (value: 01/22/13), and 'Name' (value: John Smith). Below these is a 'Product' table with columns: ID, Name, Price, Product Quantity, and Product Total. The table contains one row for 'Orange Juice' with a price of 10.00 and a quantity of 1. Below the table, there are summary fields: 'Sub Total' (10.00), 'Tax' (1.10), and 'Total' (11.10). At the bottom of the form are three buttons: 'Confirm', 'Cancel', and 'Delete'. The footer of the application shows 'Footer info'.

Figure: Testing the Invoice transaction

STEP 10: ADDING BUSINESS RULES ⇒ RULES

Add some basic business logic to your application.

GENEXUS RULES

GeneXus **Rules** are the means you use to define the business logic associated with each object. They are written in a declarative way, and GeneXus intelligently decides which rule to apply and when.

These **Rules** play a very important role in transaction objects, as they allow you to program their behavior (for example: assigning default values, defining data controls, etc.).



- They may involve attributes defined in the transaction structure, as well as variables and functions.
- Rules are programmed in a **declarative** way, which means the order in which they are written is not necessarily the order in which they will be executed. The proper execution order is determined automatically by GeneXus.

They are only valid in the transaction in which they are defined. That is why we say they are **local**.

We will now add a simple rule that sets the current date as the default Invoice Date:

1. Select the **Rules** tab in the Invoice transaction.
2. We will use the **Default** rule that assigns a default value to an attribute or variable.
3. Complete the formula as follows: `Default(InvoiceDate, &today);` which indicates that the default value for the Invoice Date will be the current date.¹
4. Now let's see another simple rule that sets an error message to be displayed when the amount of products entered is null: add the following rule: `Error("The product quantity cannot be empty") if InvoiceProductQuantity.IsEmpty();`
5. Click on the **Save** button.

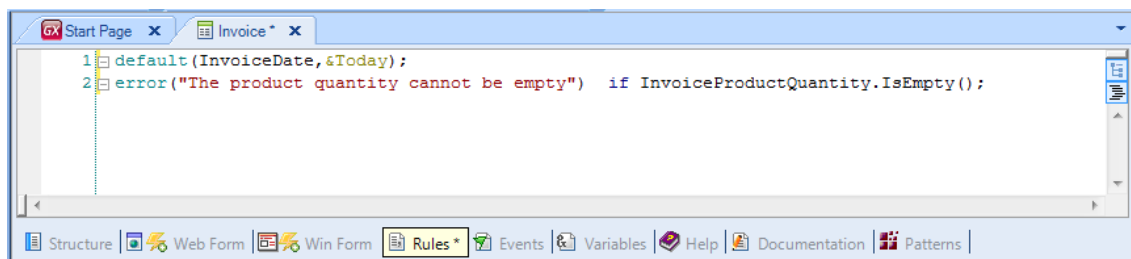


Figure: Invoice rules

- Run the application (press F5) to try out the rules you entered.

Application Header English | Español | Português

First Option Second Option Third Option Fourth Option

Recents: Invoice |

Invoice

ID:

Date:

ID:

Name:

Product

ID	Name	Price	Product Quantity	Product Total
1	Orange Juice	10.00	<input type="text"/>	The product quantity cannot be empty
0		0.00	0	0.00
0		0.00	0	0.00
0		0.00	0	0.00
0		0.00	0	0.00
[New row]				

Sub Total: 10.00

Tax: 1.10

Total: 11.10

Confirm **Cancel** **Delete**

Footer Info

Figure: Testing Invoice rules

STEP 11: CREATING THE CUSTOMER TRANSACTION OBJECT

Customers are concepts that correspond to independent entities of the invoice. Therefore, they must be defined as a transaction in themselves.



SEMANTIC DOMAINS

- GeneXus provides [semantic domains](#) (Phone, Email, Address, etc.) in order to add behavior to the attributes in our transactions. For example: if we create an Email attribute, every time we use this attribute in our objects it will behave as an email, and it will allow us to send an email message by tapping on it in smart devices or web objects.

1. Create the Customer Transaction object
2. Add the following attributes to the Customer Structure:

ATTRIBUTE	TYPE	DESCRIPTION
CustomerID	-----	-----
CustomerName	-----	-----
CustomerAddress	Address	Customer Address
CustomerEmail	Email	Customer Email

Note that as you start writing the CustomerID and CustomerName attributes, GeneXus prompts you with the full name of the attribute and its type and description. That happens because these attributes have already been defined in your database.



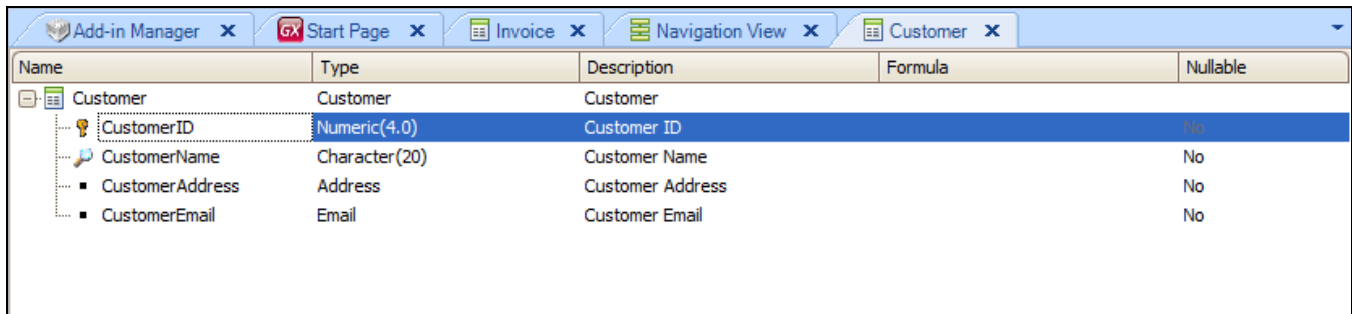
SEMANTIC DOMAINS

GeneXus provides [semantic domains](#) (Phone, Email, Address, etc.) in order to add behavior to the attributes in our transactions. For example: if we create an Email attribute, every time we use this attribute in our objects it will behave as an email, and it will allow us to send an email message by tapping on it in smart devices or web objects.

3. Right-click on the CustomerID attribute and select **Properties**.
4. Inside Type Definition, set the CustomerID **Autonumber** property to **True**. In this way, the programs generated will automatically assign a CustomerID value to each new Customer instance.
5. Activate the Suggest for the customer code, by positioning yourself on the Customer ID properties, and under the InputType option, defining Description and adding as ItemDescription: CustomerName.

This way, instead of entering the ID for a customer to identify the customer, we can enter the customer's name and the application will automatically infer the customer's ID. The Suggest property will suggest all the customer names that match the name entered by the user. These properties are part of the **AJAX** implementation that GeneXus performs automatically.

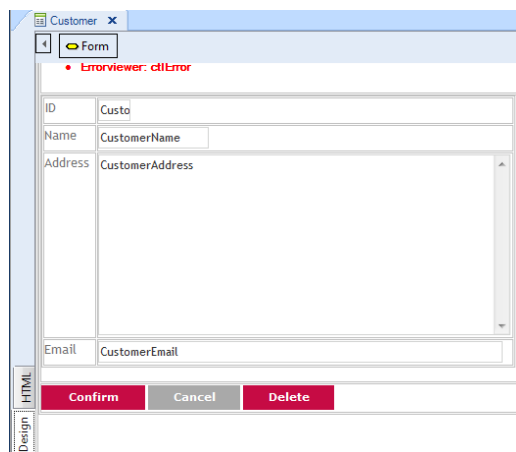
The Structure for the Customer Transaction will look like the one shown in the figure below



Name	Type	Description	Formula	Nullable
Customer	Customer	Customer		
CustomerID	Numeric(4.0)	Customer ID		No
CustomerName	Character(20)	Customer Name		No
CustomerAddress	Address	Customer Address		No
CustomerEmail	Email	Customer Email		No

Figure: Customer Transaction Structure

The Web Form for the Customer Transaction will look like the one shown in the figure below.



Customer

Form

Errorviewer: dltError

ID: Custo

Name: CustomerName

Address: CustomerAddress

Email: CustomerEmail

Confirm Cancel Delete

Figure: Customer Transaction Web Form

Note (below) that the Invoice Web Form also changed, reflecting the changes in the properties of the CustomerID attribute: the CustomerName attribute is now the description of the CustomerID attribute. As we will soon see, this greatly enhances the usability of the application!

The screenshot shows the GeneXus IDE with the 'Invoice' web form open. The form is titled 'Invoice' and includes an error message: 'Errorviewer: ctrlError'. The form contains the following fields and sections:

- ID**: A text field with the value 'Invoic'.
- Date**: A text field with the value 'InvoiceDe'.
- Customer Name**: A text field with the value 'CustomerID'.
- Product**: A section containing a table with the following columns: ID, Name, Price, Product Quantity, and Product Total. The table has 5 rows, with the first row containing the values: ProductI, ProductName, ProductPrice, InvoiceF, and InvoiceProd.
- Sub Total**: A text field with the value 'InvoiceSu'.
- Tax**: A text field with the value 'InvoiceTz'.
- Total**: A text field with the value 'InvoiceTc'.
- Buttons**: Three buttons labeled 'Confirm', 'Cancel', and 'Delete'.

The IDE interface shows the 'Form' tab selected, and the 'Design' view is active. The bottom status bar indicates the current view is 'Web Form'.

Figure: Invoice Transaction Web Form

STEP 12: REVIEWING THE CHANGES MADE IN YOUR DATA MODEL

Press the F5 key. This will generate an Impact Analysis Report.



IMPACT ANALYSIS

This describes the physical changes that must be made to update the new data model inferred by GeneXus.

You will find that GeneXus has automatically normalized your data model, after including the new transaction.

It shows us that the Invoice table will be affected, eliminating the CustomerName attribute from the Invoice table, as this attribute can be inferred through the CustomerID.

Database needs to be reorganized.

This report describes Database changes and how they will be handled by reorganization programs.
Please select Reorganize to proceed or Cancel.

Table Invoice specification

Table name: [Invoice](#)

Invoice needs conversion

Table Structure

Attribute	Definition	Previous values	Take
InvoiceID	Numeric (4)Not null Autonumber		InvoiceID
InvoiceDate	Date Not null		InvoiceDate
CustomerID	Numeric (4)Not null		CustomerID
Del CustomerName	Character (20)Not null		CustomerName

Indexes

Name	Definition	Composition
IINVOICE	primary key Clustered	InvoiceID
New IINVOICE1	duplicate	CustomerID

Foreign key constraints

Referenced table	Attributes
New Customer	CustomerID

Statements

```
CREATE NONCLUSTERED INDEX [IINVOICE1] ON [Invoice] (
    [CustomerID])
```

0 Errors 1 Warnings 1 Success

Figure: Database impact report

A new table appears, the Customer table:



NOTE

In the specification warning, GeneXus indicates that there may be duplicate values for customers in the Invoice table and that in the process of normalizing them, they could be affected. Another warning is displayed, indicating that the records that will be created have a null value in the tables.

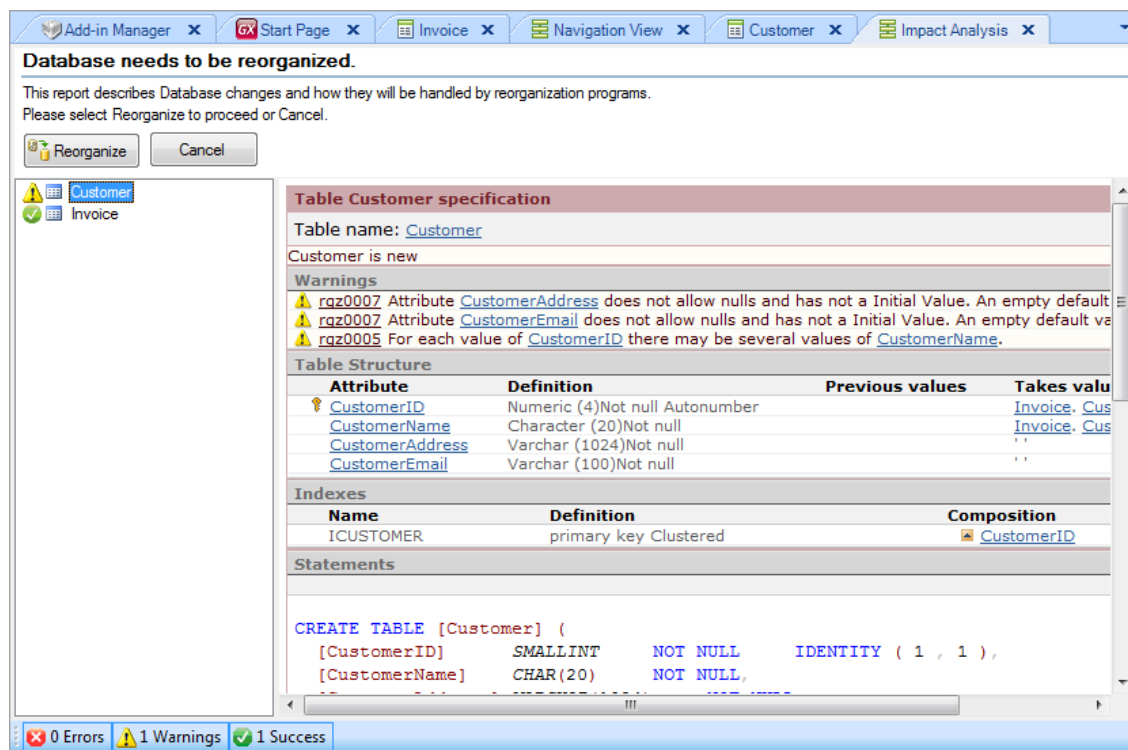


Figure: Database impact report

Note that:

- GeneXus added a new table: the Customer table (associated to the Customer transaction).
- The CustomerName attribute has been eliminated from the Invoice table (this attribute is stored in the Customer table only), leaving the database normalized.
- In the Customer table:
 - No two customers can have the same CustomerID.
 - For each CustomerID there is **only one** value for CustomerName, CustomerAddress and CustomerEmail.
- In the Invoice table:
 - No two invoices can have the same InvoiceID.
 - The CustomerID attribute in the Invoice table is a foreign key taken from the Customer table. Thus, there is a one-to-many relation between Customers and Invoices:
 - For each InvoiceID there is **only one** CustomerID.
 - Each CustomerID can have **many** Invoices (InvoiceID).

STEP 13: VIEWING THE SPECIFICATION REPORT

You are now ready to go ahead with the reorganization of the DB. Select the REORGANIZE option. The reorganization programs create a new database schema in the physical database of the target model and transport the data from the old schema to the new one.

When the reorganization is executed, GeneXus displays a **Specification Report** consisting of a series of **Navigation Reports**, one for each program that it will generate.



SPECIFICATION REPORT

Describes how the program will be executed, which tables it will access (and how), and what operations it will perform.

“Referential Integrity controls on delete” means that when you delete a customer from the Customer Transaction, the program will check that there are no invoices for that customer. To perform this search efficiently, the foreign index “CustomerID” of the Invoice table is used.

Transaction Customer Navigation Report

Name	Customer	Environment	.net Default (C#)
Description	Customer	Spec. Version	10_2_2-64721
		Form Class	HTML
		Program Name	Customer
		Parameters	

Levels

Level Customer

=Customer(CustomerID)

Insert into Customer (CustomerName, CustomerAddress, CustomerEmail)

Update Customer (CustomerName, CustomerAddress, CustomerEmail)

Delete from Customer

Referential integrity controls on delete:

- Invoice(CustomerID)

Prompts

Table	Program	In Parameters	Out Parameters
Customer			CustomerID

Figure: Customer Transaction Navigation Report

Referential Integrity in the Invoice Transaction:

The CustomerID foreign key at the INVOICE level means that when you insert or update an invoice using the INVOICE Transaction, GeneXus will automatically check that the value entered in the CustomerID foreign key exists as the primary key of a record in the Customer table. To perform this search efficiently, the CustomerID primary index of the Customer table is used.

The referential integrity check validates that an attribute entered (e.g. CustomerID in an Invoice) is valid, but it provides no information on what the valid values are. In order to facilitate the search for the valid values, GeneXus creates **Selection List** objects (prompts) that show you the complete set of valid values for you to choose from.

Transaction Invoice Navigation Report

Name	Invoice	Environment	.NET Default (C#)
Description	Invoice	Spec. Version	10_2_2-64721
		Form Class	HTML
		Program Name	Invoice
		Parameters	

Levels

Level Invoice

=Invoice(InvoiceID)

~InvoiceSubTotal navigation

=InvoiceProduct(InvoiceID)

Insert into Invoice (InvoiceID, InvoiceDate, CustomerID, CustomerName)

Update Invoice (InvoiceDate, CustomerID, CustomerName)

Delete from Invoice

Level InvoiceProduct

=InvoiceProduct(InvoiceID, ProductID)

Insert into InvoiceProduct (InvoiceID, ProductID, ProductName, ProductPrice, InvoiceProductQuantity)

Update InvoiceProduct (ProductName, ProductPrice, InvoiceProductQuantity)

Delete from InvoiceProduct

Prompts

Table	Program	In Parameters	Out Parameters
InvoiceProduct	Gx0021	InvoiceID	ProductID
Invoice	Gx0010		InvoiceID

Figure: Invoice Transaction Navigation Report

PART 3: GENERATING APPLICATIONS FROM DEVELOPMENT PATTERNS

USING PATTERNS DURING DEVELOPMENT

Upon testing our application, we could see the data entry through transactions. But it would be convenient to have a more general view of the data, so as to not only enter, delete and modify Customers, Invoices or Products, but also to filter them according to specific criteria, sort them, etc. Patterns are what we resort to in order to quickly and easily implement this functionality.



PATTERNS

Patterns are similar actions applied to different elements. Patterns allow the automatic creation of all objects necessary for fulfilling a specific functionality, while avoiding the manual method.

STEP 14: PATTERN FOR WEB APPLICATIONS

We will apply the “Work With” pattern to the transactions created.

1. Through the **View** menu, select the **Work with Objects** option.
2. Select Transaction from Type combo box menu on Work With Objects panel.
3. Select the Customer and Invoice transactions.

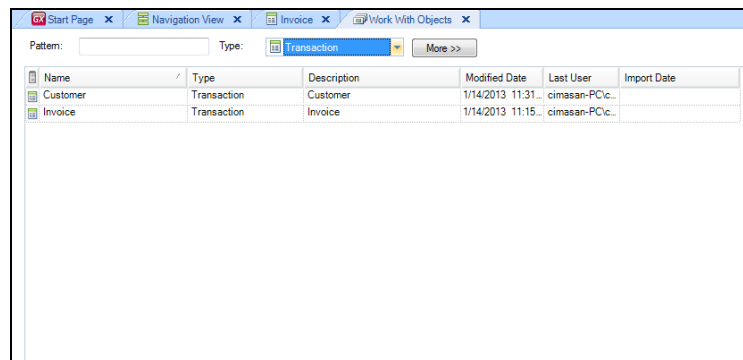


Figure: Work With Objects

4. Right-click on the selected transactions, and select the Apply Pattern, Work With option.
5. Save changes.
6. Press F5 to execute the application. We can see there is no more access to the transactions. We do have access to our “Work with” Customer and Invoice.

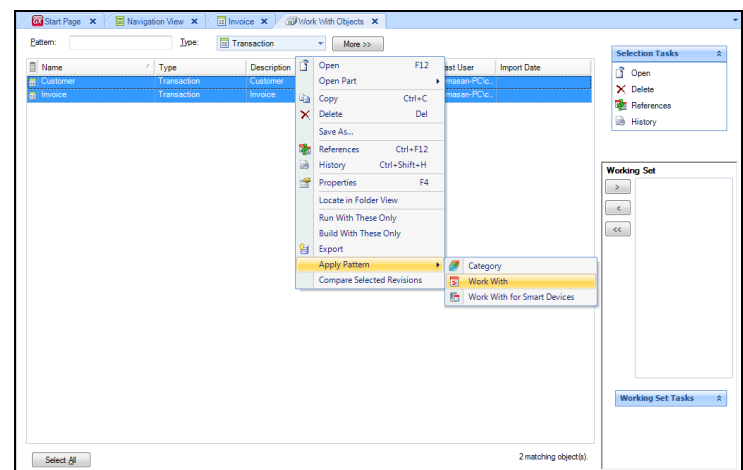


Figure: Apply Pattern

7. Execute the application

The pattern added the following functionality:

- Filter by date, without the need to refresh the application.
- It may be sorted by columns by clicking the column headers.
- The grid can be paged.
- By clicking on a link, we view an object showing the related data.
- Saving of the historical file with links visited.

Application Header

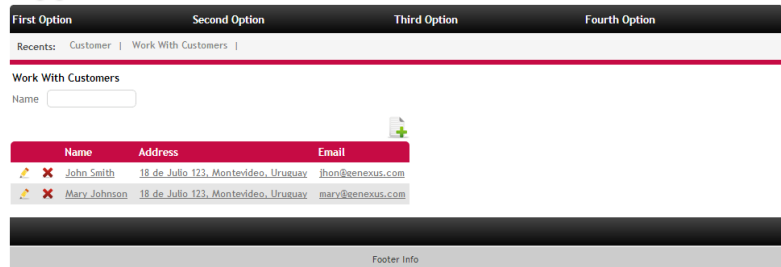


Figure: Running the application



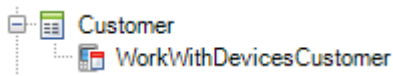
SEMANTIC DOMAINS

- Open the Customers Work With and look at how the application behaves regarding the attributes that we previously set as semantic domains.

STEP 15: PATTERN FOR SMART DEVICES APPLICATIONS

We will apply the “Work With for Smart Devices” pattern to the Customer transaction.

Apply the “Work With for Smart Devices” pattern to the Customer transaction. To do so:

- Go to the Patterns section of the transaction
- Select “Apply this pattern on save”
- Familiarize yourself with the nodes labeled List, Detail and Section (General) in the instance, and take a look at the screen displayed in each case.
- Save and note what happens with the following elements:
 - Transaction properties:
 - **Business Component** = True
 - **Expose as Web Service** = True
 - **Web Services Protocol** = ReST Protocol
 - Folder View / Customer: a sub-node is displayed:
 
 - Smart Device generator as secondary generator (take a careful look at its properties, especially: **Generate Android** = True) in Preferences / .Net Environment / Generators
- What else do you need to test the application? If you press F5 now, since there is no main object for the smart devices section of the application, there will be nothing to execute. So, prior to pressing F5:
 - Create a **Dashboard** (Select File Menu -> New -> Object -> Dashboard)

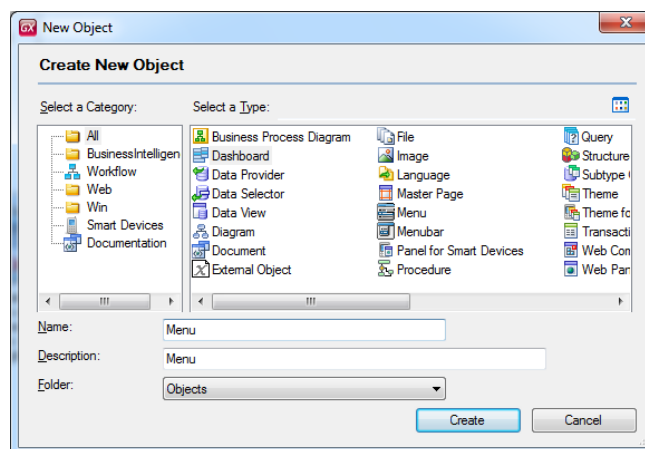


Figure: Creating a Dashboard

- Add an item to invoke the **“WorkWithDevicesCustomer”**:

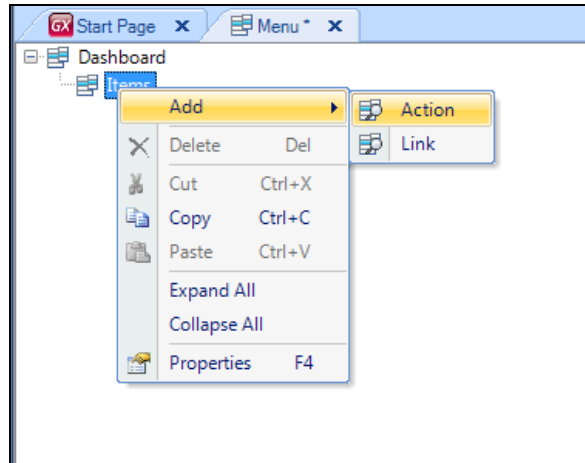


Figure: Adding an Action

- Associate an image to the option:

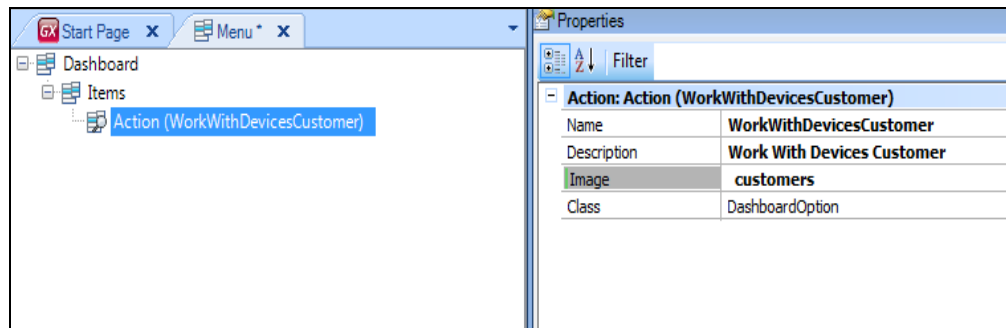


Figure: Adding actions to the Dashboard

- Note the event associated to this option (right-clicking on the action):

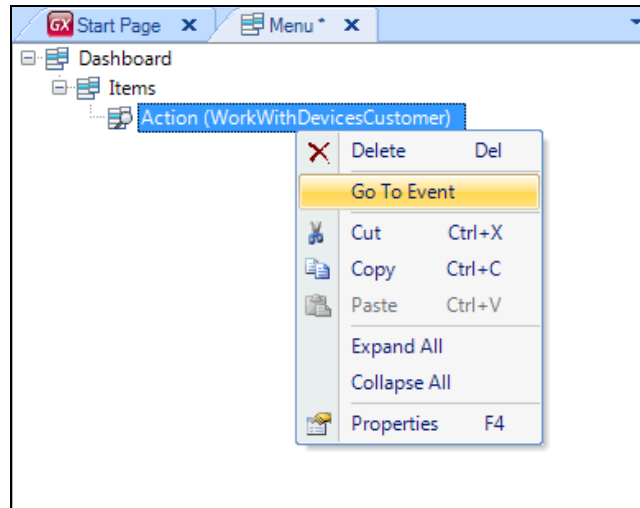


Figure: “Go To Event” action

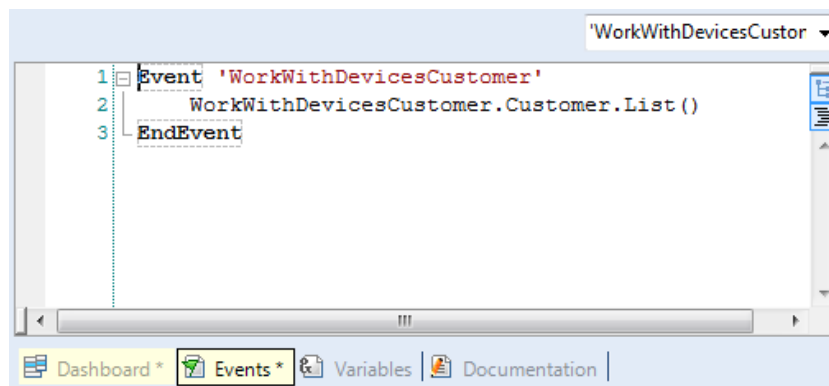


Figure: Dashboard events

- Save and note that the **Main Program** property has “True” value.

- Then press **F5** to generate and execute the application on the emulator.

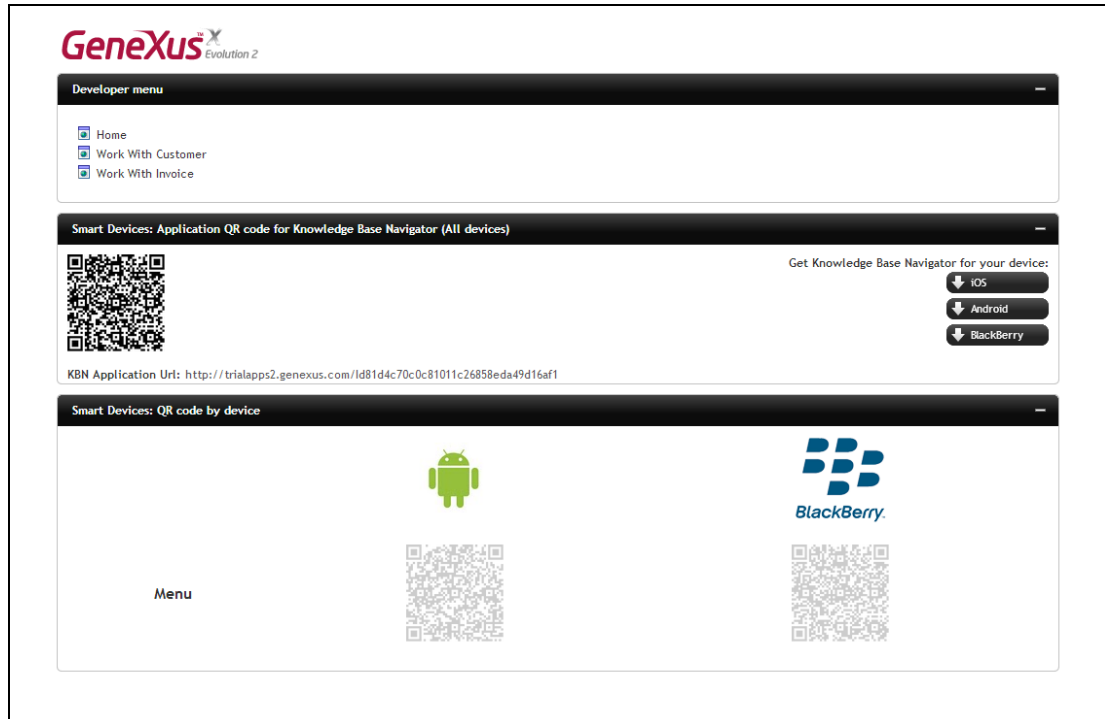


Figure: Developer Menu with QR codes

Notes:

- Because we have not specified any “Startup Object”, GeneXus will open both the web Developer Menu on the predefined browser, and the Android emulator, so we can try the .NET web application and also the smart devices application: Android.
- In our case, the Android emulator is opening the KBN (Knowledge Base Navigator) that shows the URL to execute the application’s dashboard (the only main object that we have so far).
- In addition to the links for executing the objects of the web application, the web Developer Menu will also include QR codes: one that encapsulates the URL for downloading the KBN according to the device, for installation in it, and the others which will contain the compiled application for smart devices, for both Android and Blackberry. If you set any smart device object as “Startup object” a compiled file will be created and the related QR code will be available on the web Developer Menu.
- For more on this go to [Executing From QR Codes](#)

- Tapping on the URL will display the layout corresponding to the Dashboard

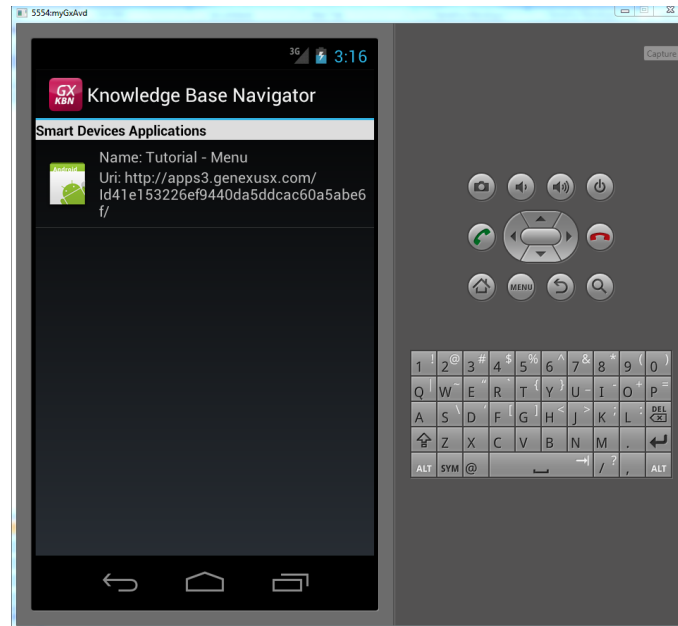


Figure: Android emulator

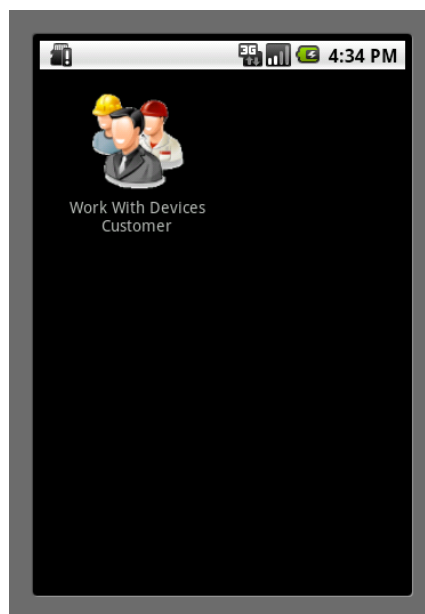


Figure: Dashboard

- Tap on “**Work With Devices Customers**”

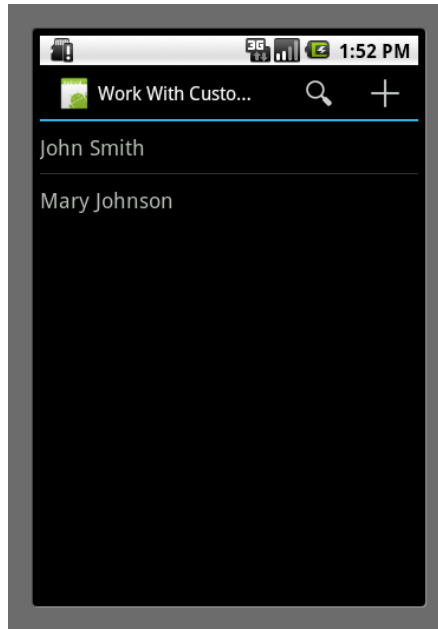


Figure: Work With Device Customers

- Take a look at the attributes being displayed for each customer. Go to GeneXus and in the pattern look for the layout of this **List**, and verify the match between them.
- Tap on any of the customers listed to see its details displayed:

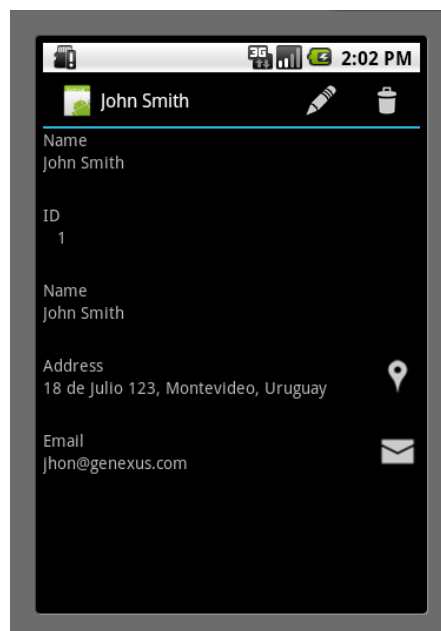


Figure: Customer Detail

- See what happens when you tap on the icon displayed in the address field (remember semantic domains):



NOTE

To set an email account in your emulator press menu -> system setting -> accounts & sync.

- Edit or delete some customer, by pressing the related option on the top of the screen:



Figure: Edit Customer

- To insert new customers or search press back and use the related options:

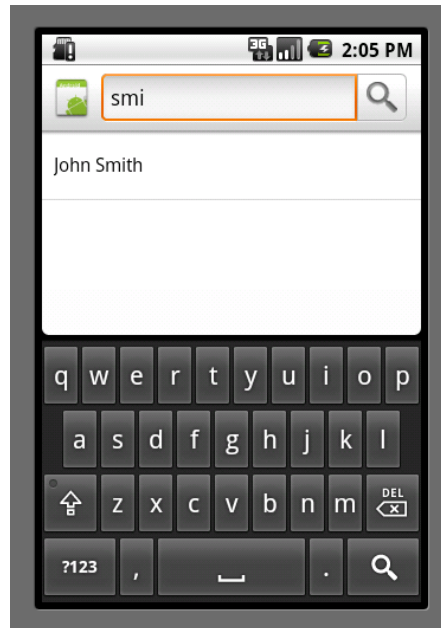


Figure: Search

CONGRATULATIONS!

You have successfully created your first application with GeneXus.

SUMMARY

We hope that this tutorial has allowed you to experience GeneXus' key functionalities and benefits:

Knowledge-based Design of Applications

We start by creating a Knowledge Base, and describing the data requirements of our application as a series of business components called Transaction Objects in the Design Model of the Knowledge Base. GeneXus used this information to infer the optimum data model (3rd normal form) required to support all its Transactions. We also added certain business rules that GeneXus used as a base to generate the code of the application's business logic.

Automatic Generation of the Database

We selected a specific execution platform (Web, programming language, DBMS, etc.) and created a Model where GeneXus automatically generated a physical database with the data model of the Design Model.

Automatic Code Generation and Fully Functional Prototype

We generated the source code for our application programs, and tested the application in the prototype environment.

Maintenance of the application

We saw the simplicity of maintaining/expanding an application by editing the existing GeneXus objects and/or adding new ones for GeneXus to update the database and automatically regenerate the application programs.

Multi-platform Development

We finally described how to easily migrate the GeneXus application from one environment to another.

CONTACTS AND RESOURCES

GENEXUS COMMUNITY

The GeneXus Community provides various ways for obtaining answers to your questions and solutions for your problems, as well as the possibility to share your own experiences with others. You will find a full list of the Community's resources available at genexus.com/community

SUPPORT

Artech offers a wide variety of support resources and services:

- **Online Self-serve Support**
These resources are available online for everyone. However, the data that may be accessed depends on the Access Level to GXtechnical (Registered User or Customer).
- **Interactive Support Services**
Interact with other members of the Community or with the Support Team.

Visit genexus.com/support

If you are a U.S. or Canadian resident you may forward your questions to gxtrial.usa@genexus.com

HOW TO BUY

GeneXus Technologies are sold through its worldwide network of distributors.

Look up your nearest distributor at genexus.com/distributors

or contact sales@genexus.com

RECOMMENDED LINKS

Website: genexus.com

GeneXus X Training Area: training.genexus.com

GeneXus Community: genexus.com/community