# PROGRESS EXCHANGE

**Progress Developer Studio for OpenEdge** 



#### Copyright © 2013 Progress Software Corporation

Progress<sup>®</sup> software products are copyrighted and all rights are reserved by Progress Software Corporation. This manual is also copyrighted and all rights are reserved. This manual may not, in whole or in part, be copied, photocopied, translated, or reduced to any electronic medium or machine-readable form without prior consent, in writing, from Progress Software Corporation.

The information in this manual is subject to change without notice and Progress Software Corporation assumes no responsibility for any errors that may appear in this document.

The references in this manual to specific platforms supported are subject to change.

OpenEdge® is a registered trademark of Progress Software Corporation.

All company and product names are the trademarks or registered trademarks of their respective companies.

Printed in the USA

October, 2013 Version 1.0



### WORKSHOP OVERVIEW

Welcome to Exchange and to this workshop on Progress Developer Studio for OpenEdge (PDS OE). The goal of this workshop is to introduce you to Progress Developer Studio for OpenEdge and to provide you with the knowledge needed to gain the most from this Integrated Development Environment (IDE).

This workshop is composed of a number of labs that show you how to use the primary features of Progress Developer Studio for OpenEdge. The labs explain the features of Progress Developer Studio for OpenEdge at length, including the application development using AppBuilder, GUI for .NET, WebSpeed, and REST. The discussions in this workshop have been designed based on what you need to know to get started and on frequently asked questions by the users of Progress Developer Studio for OpenEdge.

When you complete this workshop, you should be able to:

- Use Progress Developer Studio for OpenEdge as your primary tool for ABL development
- Use the OpenEdge editor to write application code
- Use AppBuilder and GUI for .NET for developing ABL desktop applications
- Use WebSpeed and REST for Web and Cloud based applications development

Enjoy the workshop!



#### CONNECTING TO YOUR MACHINE

- 1. Open the browser (Google Chrome, Mozilla Firefox, or Apple Safari).
- 2. Depending on your session, select the appropriate link and paste it in your Browser, and hit **Enter**.
  - Morning workshop, use this URL <u>http://23.23.210.136:8980/WorkshopApp</u>
  - Afternoon workshop, use this URL <u>http://54.225.237.144:8980/WorkshopApp</u>
- 3. Enter your email address.
- 4. Click Get Arcade Instance.

It gives you a DNS for a running Arcade instance, which you can use to connect to an RDP session.

Instance assignment ×		
← → C	×	/ [ ] =
💌 OE Mobile Instances 🦳 Motorfiets 🦳 Traxxas 🥥 Login - Progress Sav 🔞 JCI - Demo - Source	39	Andere bladwijzers
Instance assignment		
Enter email address		
Get Arcade Instance		
		6



#### WORKSHOP LABS

This workshop is composed of several short presentations and labs on various developer studio capabilities. Completing all of the labs is not the goal of the workshop. The goal of the workshop is to get you to the point where you can go back to your office and be ready to start using Progress Developer Studio for OpenEdge.

The labs contain Tips , that offer suggestions related to the task at hand, and Notes , that provide additional details related to the current task. Tips and Notes provide additional information on a topic, but are not mandatory to complete the labs.

Also, pay attention to the Cautions  $\checkmark$  to a

to avoid possible problems.



#### CONTENTS

1	La	b 01: Configuring your workspace and customizing the project	1-8
	1.1	Overview	1-8
	1.2	Prerequisites	1-8
	1.3	Starting Progress Developer Studio for OpenEdge and creating a workspace	1-8
	1.4	Creating a Database connection	1-11
	1.5	Creating an OpenEdge Project	1-16
	1.6	Resetting a perspective	1-20
	1.7	Adding a directory to a project	1-21
	1.8	Setting project properties	1-22
	1.9	Closing and Opening projects (Optional)	1-23
	1.10	Accessing Online Help (Optional)	1-24
2	La	b 02: Application development using Progress Developer Studio for OpenEdge – AppBuilder	2-27
	2.1	Overview	2-27
	2.2	Prerequisites	2-28
	2.3	Opening the OpenEdge AppBuilder perspective	2-28
	2.4	Creating a SmartWindow	2-29
	2.5	Designing the SmartWindow	2-32
	2.6	Working with SmartObjects	2-34
	2.6	6.1 Creating a SmartDataObject	2-35
	2.6	6.2 Creating a SmartDataBrowser	2-37
	2.6	Add SmartFilter to filter customer records (Optional)	2-41
	2.7	Designing the order entry form (Take-home)	2-47
	2.8	Linking Get Customer and Order Entry buttons to the respective frames (Take-home)	2-52
3	La	b 03: Application development using Progress Developer Studio for OpenEdge – WebSpeed	3-61
	3.1	Overview	3-61
	3.3	Creating a WebSpeed project	3-62
	3.4	Creating the SpeedScript file	3-63
	3.5	Configuring the WebSpeed server and associating the WebSpeed module	3-65
	3.6	Running the SpeedScript file on a WebSpeed server	3-68
	3.7	Adding an AppServer facet to the WebSpeed project	3-69
	3.8	Working with the SpeedScript editor	3-72
4	La	b 04: Application development using Progress Developer Studio for OpenEdge – REST	4-79
		* PROG	KESS

	4.1	Overview	4-79
	4.2	Prerequisites	4-79
	4.3	Adding a REST facet to the WebSpeed project	4-79
	4.4	Working with the REST Editor – Mapping ABL operations with HTTP verbs	4-81
	4.5	Publish and invoke the REST service	4-85
	4.6	Working with other HTTP verbs (Take-Home)	4-88
	4.7	Invoke the REST service for remaining HTTP verbs (Take-home)	4-90
5	Lal	b 05: Application development using Progress Developer Studio for OpenEdge – GUI for .NET	5-94
	5.1	Overview	5-94
	5.2	Prerequisite	5-94
	5.3	Creating Custom Project	5-95
	5.4	Assigning a database connection to a project	5-101
	5.5	Creating and designing a form	5-102
	5.6	Creating Inherited controls (Take-home)	5-109
	5.7	Adding controls to Toolbox	5-111
	5.8	Creating User control (Take-home)	5-113
	5.9	Designing the SalesReport form to display outlets data	5-115
	5.10	Working with the source editor - Adding methods and events	5-120
	5.11	Working with OpenEdge Debugger (Optional)	5-128
	5.12	Working with UltraChart (Take-home)	5-131
6	Lal	b 06: Embedding an ABL window into the MDI form (Take-home)	6-134
	6.1	Overview	6-134
	6.2	Prerequisites	6-134
	6.3	Creating an MDI form	6-134
	6.4	Embedding the ABL windows into an MDI form	6-138
7	Со	onfiguring Apache WebServer (not required for the Workshop)	7-142



### 1 LAB 01: Configuring your workspace and customizing the project

#### 1.1 Overview

This lab covers setting up the workshop environment and creating database connections and starting database servers. It includes the following:

- Creating a workspace and a project
- Configuring and customizing your project
- Setting up the project properties and PROPATH
- Closing and re-opening the project

#### 1.2 Prerequisites

The virtual machine that you will use has been set up for PDS OE with Workgroup DB and OE Ultra Controls .NET installed in the C:\Progress\OpenEdge directory.

The database required for this workshop, workshop.db is created with its schema and is placed in the C:\OpenEdge\WRK\ExchangeDB\workshop.db directory.

The files used for this workshop are placed in the C:OpenEdgeWRKPDSOEWorkshopFiles directory.

#### 1.3 Starting Progress Developer Studio for OpenEdge and creating a workspace

Before you create a project, you must start Progress Developer Studio for OpenEdge.

1. Select Start->All Programs->Progress->OpenEdge 11.3 ->Developer Studio - Clean.

The Workspace Launcher dialog box appears.

2. Set the **Workspace** to C:\OpenEdge\WRK\Exchange\_PDSOE. A new directory **Exchange\_PDSOE** is created that sets up the workspace environment.

**Tip:** Workspaces can be physically located anywhere on your system or on a network drive. However, a workspace can be used by only one developer at a time even if it is on the network drive. So, we suggest that you set up your workspaces on your local computer.



📔 Workspac	e Launcher		>
Select a wo	orkspace		
Progress Der Choose a wo	veloper Studio stores your projects in a folder called a workspace. rkspace folder to use for this session.		
Workspace:	C:\OpenEdge\WRK\Exchange_PDSOE	▼	Browse
Lise this a	s the default and do not ask again		
j ose ans a		_	Cancel
			Cancer

**Tip:** As shown in the above image, you can set up the currently configured workspace as the default workspace by selecting the **Use this as the default and do not ask again** check box. It stops developer studio from prompting for a workspace every time you open the tool and automatically opens the workspace that is set as default. You can use workspaces other than default at any time by selecting **File**  $\rightarrow$  **Switch Workspace**. If you frequently work with multiple workspaces, you can leave this check box cleared so that every time you start developer studio, you have the flexibility of starting the required workspace. This option, along with many other workspace options, can be modified using the Workspace preferences, which will be covered later in the labs.

3. Click **OK**. As Progress Developer Studio for OpenEdge opens, the splash screen shows the plug-in information as they are being loaded.

The **Welcome** page appears. It provides links to the resources of Progress Developer Studio for OpenEdge. Each link opens up related links to that particular resource.



POper File Ed	n <mark>Edge Edit</mark> o lit Navigate	- Progr Search	ess Devel Project	loper S Run	Studio OpenEdge	Window	Help			
🚳 Welc	come 🛛								 	
W TO	/ELCOM	<b>E</b> develoi	PER STU	D10						
OPI	ENEDGE*11.3							Overview		
						<		What's New	Workbench	
						Ŷ		Samples		
								Tutorials		
								Documentation		
						Z		Web Resources		
P	ROGRE	SS ware								
Cop	yright © 1984-201	3 Progress S	offivare Corp	oration a	nd/orits subsidiar	ies or affiliates	. All rights re	eserved.		

**Tip: Welcome** page gives links to all product related help. Samples and Tutorials lists items group by topic. If you select a topic and click **Open**, it takes you to the Progress communities' link of that topic. Documentation has link to the product documentation page, where latest release documents can be found.

4. Select Workbench **(**). The **OpenEdge Editor** perspective opens.



Tip: If you want to move an existing Workspace to a different physical directory, copy the workspace and move it to the new location, start PDS OE, and enter the new path for the workspace in the Workspace Launcher dialog box. You also need to change any physical paths that you have set such as the PROPATH. These changes can be minimized by using Configuration Variables (see the online help for information on Configuration Variables). You also need to change the working directory, go to the OpenEdge project properties -> Progress OpenEdge and select the **Browse** button.

#### 1.4 Creating a Database connection

You can define database connection profiles for the workspace and make connections from your projects in Progress Developer Studio for OpenEdge to OpenEdge databases.

The **Database Connections** wizard allows adding both ABL and SQL database connections. This wizard creates a database connection profile. The next section of the lab shows how projects can connect to databases by selecting database connection profiles defined for the workspace.

- 1. From the Workbench menu bar, select **Window→Preferences**. The **Preferences** dialog box appears.
- 2. Expand the **Progress OpenEdge** node and select the **Database Connections** node. The **Database Connections Preference** page appears.

Preferences						_ 🗆 ×
type filter text	Database Conne	ctions				
. Ant		-				
	Connection Name	Group	Physical Name	Other Parameters	Des	New
						Edit
						Сору
						Remove
🗉 Plug-in Development						1151115115
Progress Customer Support						Transmith
Progress Databases						Import
Progress OpenEdge						Export
AppBuilder						Import DE
						import Dr
Business Rules						Export DF
<ul> <li>Database Connections</li> </ul>						
Debug						
Editor						
Meta Catalog						
Mobile App Builder						
Server						
Shared AVM						
Startup						
Tools for Business Logic						
··· Views						
···· Visual Designer						
⊞ Run/Debug						
Server						
⊡ Team	•					
Validation	Connection string:					
⊞ Web	Connection suring.				÷.	
Web Services						
⊞∘XML					Ψ.	
0						Canaal
Ø				OK		Cancel



3. Select New.... The Add Connection Profile dialog box appears. This first page of the wizard allows you to enter the connection information for an ABL connection to the OpenEdge RDBMS.

P Add Connection F	Profile				<u>_     ×</u>
Add OpenEdge D	atabase Co	onnection			
Enter a unique name	for the databa	ase connectior	ı		
Connection name:					
Physical name:					Browse
Optional					
Description:					<b></b>
					-
Logical name:					
Host name:			Service/Port:		
User ID:			Password:		
Aliases:			Group:		-
Other parameters:					
	4				▼ ▶
Test Connection					
?		< <u>B</u> ack	Next >	Einish	Cancel
$\sim$			_		

- 4. Specify Exchange\_db in Connection name.
- 5. Click **Browse...** next to **Physical name**. Browse to the **C:\OpenEdge\WRK\ExchangeDB** folder and select the **workshop** file. Click **Save**.

The **Physical name** field should now display the **C:\OpenEdge\WRK\ExchangeDB\workshop.db** path.

6. Specify the other values as follows:

Host name:	localhost
Service/Port:	6210
User ID:	pdsoe
Password:	pdsoe

**Note:** The Service/Port field accepts either a service number or a port number. In case you you use a service name, make sure that it is defined in the services file. For Windows, the services file is located in the C:\WINDOWS\system32\drivers\etc\services directory.



P Add Connection I	Profile			<u>_ 🗆 ×</u>
Add OpenEdge D	atabase Connection			
If a User ID was spec	ified, enter the valid passw	ord for the user I	ID.	
Connection name:	Exchange_db			
Physical name:	C:\OpenEdge\WRK\Excha	ngeDB\workshop	p.db	Browse
Optional				
Description:				
				-
Logical name:				
Host name:	localhost	Service/Port:	6210	
User ID:	pdsoe	Password:	••••	
Aliases:		Group:		•
Other parameters:				<b>A</b>
	1			
Test Connection				
			_	
?	< <u>B</u> ack	<u>N</u> ext >	Einish	Cancel

7. Click Next. The Define a SQL connection dialog box opens.



**Caution:** If you do not need a SQL connection, clear the **Define SQL Connection** check box. However, we recommend that you configure a SQL connection since it used by a number of the tools in PDS OE and they might not work properly without a SQL connection.

8. Confirm that the **Add new SQL connection** option button is selected and click **Next**. The **Add SQL Connection Profile** dialog box opens; here you define the SQL connection.



P Add Connection P	rofile	_ 🗆 🗙
Add SQL Connect	on Profile	
Enter a unique name f	or the SQL connection	
Connection name:	change_ob	
Driver:	gress OpenEdge JDBC Driver (DataDirect)	<b>_</b>
URL: jdl	oc:datadirect:openedge://localhost:6210;databaseName=worksl	hop.db
User ID: pd	soe	
Password:		
	Open on Eclipse startup	
<b>v</b>	Auto-Login on connect	
URL details		
URL details		
Host name:	localhost	
🔿 Servic 🖲 Port:	6210	
Database name:	workshop.db	
SQL parameters:		
Test Commention 1		
Lest Connection		
?	< <u>B</u> ack <u>N</u> ext > ⊟nish	Cancel
	()	

You will notice that all the information that you entered for the ABL connection is automatically filled in for the SQL connection.

**Note:** When working with SQL, the user who creates the database has access to the database. By default, the database is created with windows logon ID. If you had used a different user ID to create the database then that user ID should be entered in the User ID field. here, instead of the windows logon ID. For example, the SQL connection to workshop.db needs to use the User ID of the user that created the database (in this case, PDS OE).

When using PDS OE, you may choose to not use a password for the database. It is acceptable depending on your requirement specially if are using PDS OE as only a development tool. For more information on SQL database security, including the creation and usage of user IDs and passwords, see the OpenEdge SQL documentation.

9. Click **Next**. The **Define Database Server Configuration** dialog box opens. It allows a database broker to be started for the connection profile. Continue with the defaults.



Add Conne	tion Profile				_ 🗆
Define Data	oase Server Configur	ation			
Specify either t	he service name or port of	the databas	se Server		
Auto-start	atabase server			 	
Physical name:	C:\OpenEdge\WRK\Exch	iangeDB\wo	orkshop.db		Browse.,
Service/Port:	6210				
Parameters:					4
					_
	Auto-shutdown databa	ase server			
		,			

Note: The options on this page are to configure auto-start and auto-shutdown of a database server. Ignore these options in cases when a database broker is already started for a database.

Tip: If you do not want the database broker to be started using this option, clear the **Open on** Eclipse startup check box in the Add SQL Connection Profile dialog box, so that the broker does not run when PDS OE initially starts up.

Open on Eclipse startup

- 10. Select the Auto-shutdown database server check box.
- 11. Click **Finish**. The **Database Connections** preference page appears again displaying the newly created database connection profile in the table grid.



P Preferences						_ 🗆 ×
type filter text	Database Conne	ctions				$\Leftrightarrow \bullet \bullet \bullet \bullet$
🖽 Java EE 📃 🔺						
⊡ JavaScript	Connection Name	Group	Physical Name	Other Parameters	S Des	New
Plug-In Developmen     Progress Customer	<pre>&amp; Exchange_db</pre>		C:\OpenEd			Editu
Progress Databases						
Progress OpenEdge						Copy
Advanced						Remove
AppBuilder     PDM						
Business Rules						Import
- Database Conner						Export
Debug						
						Import DF
Meta Catalog						
E Server						Export DF
Shared AVM						
Startup						
Tools for Busines	Connection string:		aban a a D.D.\arl	abon dh. U		
- Views	localhost -S 6210 -	U pdsoe	-P ****	Shop.ab -H		
I Visual Designer ▼	]				~	
?				0	к	Cancel

12. Click **OK** to close the **Preferences** dialog box.

#### 1.5 Creating an OpenEdge Project

This section shows you how to create a new project.

- 1. From the Workbench menu, select  $File \rightarrow New \rightarrow Other$ . The New dialog box appears.
- 2. Double-click the **Progress OpenEdge** node and select **OpenEdge Project**.

P New	
Select a wizard Create a new OpenEdge project	
<u>W</u> izards:	
type filter text	
Progress OpenEdge     CopenEdge Project     CopenEdge Project     CopenEdge Project     Bry     CopenEdge Project     CopenEdge	_ 
Image: Second	Enish



**Tip:** A common feature in the PDS OE user interface is a fill-in at the top of a selection that allows you to filter the list to be displayed. For example in the above screen, you can enter text in the Wizards empty text box. If you enter O, it displays the list of wizards that start with O. This filter helps to locate the selection that you are searching for.

Tip: You can directly open the **OpenEdge Project** dialog box by selecting **File**→**New**→**OpenEdge Project** or selecting the drop-down arrow button next to **New** and selecting **OpenEdge Project** as shown below:



3. Click Next. The New OpenEdge Project dialog box appears.



4. Enter OrderApp in Project name.



**Tip:** The directory path for the new project is shown in the **Location**. This is the location where the project and its associated files are physically stored. By default, the framework creates a subdirectory with the same name as the project in the workspace directory. You can choose to create the project in a different directory by clearing the **Use default location** check box and editing the **Location**.



5. Open Edge is selected as the **Project type configuration.** Continue with the defaults.

P New OpenEdge Project	_ 🗆 ×
Create an OpenEdge Project	
Enter a name for the project.	
Project name: OrderApp  Use default location	Browns
Location: C: (Openeuge/WRK/Exchange_PDSOE(OrderApp	DIOMSe
OpenEdge <custom>       AppServer       ChUI       Dynamics       GUI for .NET       Mobile       OpenEdge       REST       Static Web       WebSpeed</custom>	
< Back     Next >     Finish	Cancel

**Tip:** Depending on the application to be developed, you should select the project type configuration. It adds the related facets, and the next pages in this wizard will depend on the configurations needed for the selected project type.

6. Click Next until the Select database connections dialog box appears.



elect database co	nnecti	ons	
Select the database co	nnection	is to be used by the project.	
		Configure database	connectio
○ Show selected ●	Show all		
Connection Name	Group	Physical Name	
Exchange_db		C:\OpenEdge\WRK\ExchangeDB\work	shop.db
•			
Connection string			-
,			
Select All Deselect A	J		
2	< Bac	k Next > Finish	Cancel
	- <u>D</u> aci		Curreet

**Caution:** Add only the database connections that are needed. If you have a large number of projects that are always active and they connect to a number of databases then the time to start these connections will increase the time to start PDS OE. Also consider that there are limitations to the number of allowed database connections depending on the OpenEdge products that are installed and configured.

7. Select the **Exchange\_db** check box and click **Finish**. The project is created. The **Powered by Progress** splash screen appears showing that a Progress session is started for the project.



Once you go back to the editor, you will see the project listed in the **Project Explorer** view:



P OpenEdge Editor - Progress Deve	loper Studio	_ 🗆 🗙
File Edit Navigate Search Project	Run OpenEdge Window Help	
■ ▼ 開 橋 色   2 ▼ 祖 ▼ ← ▼ → ▼	」 ७ ୭ ▼ 0 ▼ 0 ▼ 1 / / × ] ⊔	Edge
Project Explorer 🛛 🗖 🖬		
☐ ∰ OrderApp		
An outline is not available.		
	🖳 Console 🛛 🔝 Problems 🖉 Tasks	<u>•</u>
	ABL Console	
	OpenEdge AVM started successfully for 'OrderApp'. Starting database server 'C:/OpenEdge/WRK/ExchangeDB/workshop.db' (-S 6210) Project 'OrderApp' runtime connection to 'Exchange_db' : OK	<u> </u>
		▼
📔 🍽 🖆 OrderApp		

The console view shows a message that the OrderApp project successfully connected to the AVM.

**Note:** The Workbench contains a number of views. On the left are the **Project Explorer** and the **Outline** views. The tab for the **DB Structure** view can also be seen. The bottom center shows views that can be selected by the folder tabs, for **Console**, **Problems**, and **Tasks**. The middle center is empty and is where the editor area is located.

**Tip:** It can be useful at times to clear the messages in the **Console** view to focus just on the latest messages. Click on the **Console** view to clear all the messages.

#### 1.6 Resetting a perspective

Sometimes, the views get closed or moved around the workspace. A fast way to put a perspective back to its default layout is to reset the perspective.

Each view has its own Close button on its label tab. When you move your cursor over it, the X changes color to show it has focus. The pop-up text shows the function of X as seen below. Close the Project Explorer view and the Outline view.





2. From the Workbench menu, select **Window→Reset Perspective**. The following message is displayed:



3. Click **Yes**. The selected perspective is reset to the default configuration. In this case, the **Project Explorer** view and **Outline** view are visible again on the left side of the page.

#### 1.7 Adding a directory to a project

A common task is to add a directory to a project. Directories can be used to organize resources within a project.

1. In the **OpenEdge Editor** perspective, from the workbench menu bar, select the drop-down list next to **New** 



- 2. Select Folder from the list. The New Folder dialog box is displayed.
- 3. Select OrderApp as the parent folder and enter SmartObjects in Folder name.



P New Folder	
Folder	
Create a new folder resource.	
Enter or select the parent folder:	
OrderApp	
🗈 🔔 OrderApp	
Folder name: SmartObjects	
Advanced >>	
(?)	Finish Cancel

4. Click Finish. The SmartObjects folder is created in the OrderApp project directory.

#### 1.8 Setting project properties

You can view the properties of a project and modify the PROPATH so that files from other folders can access files inside this folder.

- 1. In the **Project Explorer** view, select the **OrderApp** project.
- 2. From the Workbench menu, select **Project**→**Properties**. The **Properties for OrderApp** dialog box appears. Since project properties are specific to a project, the project name appears in the title of the dialog box.
- 3. Double-click **Progress OpenEdge** and select **PROPATH**, click **Add Workspace Directory...** The **Select Propath Directory** dialog box is displayed.
- 4. Select **SmartObjects** and click **OK**. By adding SmartObjects folder to PROPATH, all other files in the project can directly access the files in SmartObjects folder without providing fully qualified name or full path. Whenever any operation is being done, AVM first searches the PROPATH. (This is similar to JAVA CLASSPATH).
- 5. Use Move Up and Move Down to locate the newly added @{ROOT}\SmartObjects entry in PROPATH after the @{WORK} entry.



6. Double-click **Progress OpenEdge** on the left section of the page again to display the OpenEdge project settings for the selected project.

The OpenEdge project settings allow a number of OpenEdge options to be set including the working directory, the temporary directory, and the AVM startup parameters.





**Note:** The default startup parameters for projects are inherited from the workspace preferences. This saves time since most developers use a common set of startup options for each of their projects. These inherited values for the parameters can be changed or removed by clearing the **Add default parameters** check box.

**Caution:** The **--ide** startup parameter lets the AVM know that it is being used in conjunction with PDS OE. This parameter must be included for PDS OE to function properly.

**Caution:** While you could connect to a database with startup parameters on this page, we strongly discourage it. It is difficult to diagnose connection problems if they occur, as opposed to using the Database Connections properties which are designed for defining connections to databases. Details on creating database connections are covered in the *Creating a Database Connection* section.

7. Click **OK** to apply the changes. The **Project Properties** dialog box closes.

#### 1.9 Closing and Opening projects (Optional)

Workspaces use a varied amount of system resources depending upon how they are configured. You can reduce the amount of resources used by a workspace by closing projects that you are not currently using. Potential benefits of closing a project include freeing up resources used by the project such as database connections. Closing unused projects will also improve performance of some tasks. For example, searches will complete more quickly if you close projects that you do not want to search.

1. Select the OrderApp project in the Project Explorer view.



2. Right-click the **OrderApp** project and select **Close Project** from the context menu. The project and all its nodes are closed. If there were database connections for this project, they would also be closed.



The **OrderApp** project still appears in the view but is grayed out and has a different icon as shown in the screen shot above. The nodes under a closed project cannot be expanded or viewed when the project is closed.

3. Right-click the **OrderApp** project and select **Open Project** from the context menu. The project is opened again.

#### 1.10 Accessing Online Help (Optional)

Progress Developer Studio for OpenEdge comes with an extensive online help. It includes help on Eclipse Framework (Workbench User Guide) and Progress Developer Studio for OpenEdge specific help. If you add additional functionality through the plug-ins, then additional topics appear under the help contents. You can specify the topics you want help on. It speeds up the searching and reduces the number of unwanted matches. From the Workbench menu, select **Help-**>**Help Contents** to open the Progress Developer Studio for OpenEdge Help page.

P OpenEdge Editor - Progress Developer Studio		
File Edit Navigate Search Project Run OpenEdge Win	dow Help	
📬 🕶 📄 👘 🖕 📄 🖓 🍫 🔹 🚱 🕶 💁		🔍 🔊 🗿 📑 🗳 OpenEdge
] 설 ▼ 祠 ▼ ← ← ▼ → ▼	? Help Contents	
Project Explorer 🛛 🖓 🖓	Keyword Help	
□ 🔄 🏹	Service ABL Messages	
	Search	
	Dynamic Help	
	Key Assist Ctrl+Shift+L	
	Tips and Tricks	
	Cheat Sheets	
	💖 Samples	
	🖾 Tutorials	
	💷 Report Technical Issue	
	Check for Updates	
	Install New Software	
	About Progress Developer Studio	
🗄 Out 🛛 📅 DB 🗖 Pro 🖓 🗖		
An outline is not available.		

The Progress Developer Studio for OpenEdge help page opens:





You can set up the scope of the search for the helps that you would frequently need to search for.

- 1. Select Scope. The Select Scope dialog box opens.
- 2. Click New. The New Scope dialog box opens.
- 3. Set the **List name** to PDSOE.
- 4. Expand the **Progress Developer Studio for OpenEdge Guide** node. Select the check boxes that you would frequently search for. You can select multiple nodes to be included in the search. You can also select the **Progress Developer Studio for OpenEdge Guide** check box to search across the PDS OE documentation.

P New Scope
List name:
PDSOE
Topics:
Topics:      Progress DB Navigator Guide     Progress Developer Studio for OpenEdge     Getting Started     OpenEdge Projects     AppServer     OpenEdge REST     OpenEdge Mobile     OpenEdge Business Rules     WebSpeed     ABL Editor     Visual Designer (Windows only)     Class Browser     Running and Debugging ABL Programs     GUI Designer (Windows only)     Meta Catalog     Tools for Business Logic     Customization     ABL Language Reference     Copyright
Hogress recrimical issue kepoliting duide
OK Cancel

5. Click **OK**. It takes you back to the **Select Scope** dialog box.



P Select Scope	×
Show all topics Show only the following topics:	
PDSOE	
New Edit Remove	
	OK Cancel

6. Confirm that **Show only the following topics** check box is selected and that **PDSOE** is highlighted, and click **OK**. The help window now shows the scope set to **Progress Developer Studio for OpenEdge Guide**.

Scope: PDSOE

**Note:** The search scope is retained, so the next time you start this workspace and open help, the search scope is still set to PDS OE.

7. Enter a keyword such as *tools* in **Search**, and click **GO**. The first time you access help, indexing is performed.

P Help - Progress Developer Studio		
Search:	tools	Go
Search	Results	8 🔳 🛙
	Indexing	
	37% complete	
Please v is index	vait while the online inf ed. This will happen or	ormation

After the indexing is complete, the search results are displayed. The next time, you search a keyword, it completes much faster.

8. Close the **Help** page.



## LAB 02: Application development using Progress Developer Studio for OpenEdge AppBuilder

#### 2.1 Overview

The sections of this lab show you how to design, develop, and run an ABL application. You will develop an Order Entry application using AppBuilder.

Here is how your application will finally look:

💷 OrderApp				
Get Customer	Phone: 617	Begins	• Ap	ply Filter Blank Reset
	Customer ID Customer Name	Phone	Address ^	
	1 Andrews	617 555-1111	10 Smith St. Be	
	2 Justine Smith	617 333-3334	1342 Atlantic Av	
	7 Andy Davis	617 828-2222	884 Pine Dr, Bc	
Urder Entry	10 Adams Marcy	617 989-8988	23212 Winthrop	
	15 Dawsen Larry	617 383-3939	232 Westport A	
	17 Hall Mark	617 393-3331	3233 Echo Driv	
			v	



💷 OrderApp		
	Customer: Kelly Bradford Item Name: Veggie Special   Quantity: <sup>2</sup>	
	Category: Veg Price: 27.20 Add	
Get Customer	Extended Price: 54-40 Outlet Name: Atlantic Ave	
	Order Number Item Name Price Extended Price Quantity  11/Veggie Special 27.20 54.40 2	
Order Entry		elete
	Amount 54.40	
	Total Paid: 0.00	
	Cash Card Exact	

#### 2.2 Prerequisites

Complete Lab 01: Configuring your workspace and customizing the Project prior to working on this lab.

#### 2.3 Opening the OpenEdge AppBuilder perspective

Open the AppBuilder perspective in Progress Developer Studio for OpenEdge.

1. From workbench menu, select **Window→Open Perspective→Other...** The **Open Perspective** dialog box appears and displays a list of all available perspectives.





- 2. Select **OpenEdge AppBuilder** and click **OK**.
- 3. Observe that the **AppBuilder** perspective is opened. Two new views; **ABL Messages** and **ABL Cue Cards** are displayed. We will discuss these in the coming sections.

Note: Working in a related perspective has many advantages for the developers. **Project Explorer** context menu displays the related file options in the list for easy selection. Related views for the development work are provided. Views are arranged in such a way that the right editor is provided as required.

#### 2.4 Creating a SmartWindow

A SmartWindow is a blank container where you can add your SmartObjects and establish the appropriate SmartLinks to connect them. In this section, we will create a SmartWindow and edit its properties so that it could contain your SmartObject instances.

- 1. Select the **OrderApp** project from the **Project Explorer** view. Right-click and select **New→ABL UI Design** from the context menu. The **New Window** dialog box appears.
- 2. Select Container -> SmartWindow in Object Type. Enter OrderApp.w in File name.



P New Wind	ow	<u>_     ×</u>
Create an A	ABL GUI procedure	
Specify a nan	ne for the Window procedure.	
<u>C</u> ontainer:	\OrderApp	Browse
Object Type:	Container  SmartWindow  SmartFrame SmartDialog Simple SmartContainer Uindow Dialog TTY Window TTY Dialog	Template
Template:	C:\Progress\OpenEdge\src\adm2\template\cntnrwin	n.w
<u>D</u> escription:	ADM2 SmartWindow Object Template Use this template to create a new window which sup SmartObjects. Draw your SmartObjects on this con	oports
<u>F</u> ile name:	OrderApp.w	
?	< Back Next > Finish	Cancel

3. Click **Finish** to create a SmartWindow as shown in the image below:



Note: The ABL Cue Card view displays help on a specific SmartObject type. When you create a new SmartObject master, the Cue Card associated with the SmartObject type is also opened. The Cue Card provides a basic definition of the SmartObject type, along with the information on creating and using the object.



This view lists the details of all the active Cue Cards in your current AppBuilder session. The ABL Cue Cards is the default view and is available only with the AppBuilder perspective.

To retrieve the ABL Cue Cards view that you close, choose Help>ABL Cue Cards from the main menu.

- Select **Property sheet** option from **Toolbar**. 4.
- 5. Enter OrderApp in Title.
- 6. Change the values for Width and Height properties to 125 and 22 respectively. Confirm by opening the Properties Sheet window again.

Property Sheet -	wWin	×
Object: WWin		
Title: OrderAp	p	
	con Image Small Icon Image	Aa
Context-Sensitive Help		
Conte	xt Help	
Help File:	Browse	
Geometry		
Column: 2.43	Width: 125.00 Virtual Width: 146.14	
Row: 1.38	Height 22.00 Virtual Height 28.81	
Other Settings		
<b>▼</b> 3-D	Max-Button 📝 Show-in-Taskbar	
🔽 Control-Box	🥅 Message-Area 👘 Small-Title	
🔲 Drop-Target	📝 Min-Button 📃 Status-Area	
Explicit Position	Resize Suppress Window	
📝 Hidden	🔲 Scroll-Bars 🛛 📝 View	
📃 Keep-Frame-Z-(	Order 🛛 🗹 Sensitive	
		L
ОК	Cancel <u>A</u> dvanced	<u>H</u> elp

Note: The **Property Sheet** lets you customize the various properties. If you have many frames and controls designed in a window, it is difficult to select an item in a window. The Outline view shows an outline of all the controls.

- 7. Click **OK** to close the **Properties Sheet** dialog box.
- 8. Click **File** $\rightarrow$ **Save** to save the changes.



#### 2.5 Designing the SmartWindow

This section shows you how to design your user interface.

1. From the **Widgets** on the right section, select **Frame**.

😳 Palette	Þ
Rointer	
🗁 Widgets	∞
DB Fields	
<b>₀</b> Q Query	
I Browse	
🔸 🛅 Frame	
→ 🔁 Rectangle	
🐻 Image	
▶ 8 Radio Set	
🛛 Toggle Box	
∩⊐ Slider 👻	

- 2. Click anywhere on the **OrderApp** window.
- 3. Repeat Step 1 to add another frame to **OrderApp** window.
- 4. Click **Pointer** in **Palette**.
- 5. Drag frames as shown in image below. You have now divided the window into two different frames; you will now design each frame.

OrderApp - OrderApp.w		×		
Frame A	Frame B			
		:		
[				
		· · · · ·		
		:		
		· · · · ·		
[]				
		•		
		· · · · ·		
·····				
		-		
		:		
		· : · · · ·		
		:		
		:		
		· : · · · ·		

- 6. Double-click the Frame A area. The Property Sheet-FRAME-A dialog box appears.
- 7. Clear the **Title Bar** check box in **Other Settings**.



- 8. Specify frameMenu in Object. Click OK to close the Property Sheet dialog box.
- 9. Repeat for Frame B. Specify frameCustomer in Object.
- 10. Click **File** $\rightarrow$ **Save** to save the changes.
- 11. Select Button in the Widgets category from Palette.
- 12. Click frameMenu area. Button 1 gets added to Frame A.
- 13. Double click **Button 1** to open the **Property Sheet** dialog box.
- 14. Specify 25.0 in Width and 1.50 in Height.
- 15. Select to change the font of Button 1. The **Choose Font** dialog box appears.

P Choose Font	
	Sample
▲ ▲ AaBbYyZz AaBbYyZz	AaBbYyZ AaBbYyZz AaBbYyZz
OK Cancel	Edit Save Font Settings Help

- 16. Select and select the font with bold letters. Font choices may vary in different systems. Select the choice with bold letters.
- 17. Click **OK** to close the **Choose Font** dialog box.
- 18. Click **OK** to close the **Property Sheet** dialog box for **Button 1**.
- 19. Right click **Button 1** and select **Duplicate** from the context menu.
- 20. Arrange the buttons one after the other in the **frameMenu** area.
- 21. Double-click each button to open the respective **Property Sheet** dialog box and modify the values as provided in table below. Click **OK**.

	Object	Label	
Button 1	butCustomer	Get Customer	
Button 2	butOrdEntry	Order Entry	



#### Your screen will look like this:

I OrderApp - OrderApp.w				
				:
		· · · · · · · · · · · · · · · · · · ·		 
				:
Get Customer				
· · · · · · · · · · · · · · · · · · ·		:		 ·····
Order Entry				
		:	:	 
		: : : : :		
<u>}</u>	<u></u>	· · · · · · · · · · · · · · · · · · ·	• • • • • • • • • • • • • • • • • • • •	 · · · · · · · · · · · · · · · · · · ·

22. Click **File** $\rightarrow$ **Save** to save the changes. You have completed designing the frameMenu and it appears on the left side of the main frame.

#### 2.6 Working with SmartObjects

This section shows you how to create SmartObjects. SmartObjects are external procedures that encapsulate standard UI and functionality. The set of SmartObjects provided with the Application Development Model (ADM) are useful for typical database applications. For example, most database applications view data, so the ADM provides a SmartViewer object. The main purpose of a SmartViewer is to display data.

Each of these SmartObjects represents a part of a component of a typical database application.

Each SmartObject knows how to interact with other SmartObjects. For example, the SmartDataObject and SmartBrowser work together in predefined ways. The SmartDataObject tells the browser what records to display in the browser. In the SmartDataObject, we define the query to be processed and the linked SmartBrowser will display the resultant data.

**Note:** SmartObjects need a database connection. If there is no database connection assigned to the current project, you will be asked to connect to a database.



#### 2.6.1 Creating a SmartDataObject

- 1. Select the SmartObjects folder from the Project Explorer view. Right-click and select New→ABL UI Design.
- 2. Select SmartDataObject in Object type. Enter dGetCust.w in File name.
- 3. Click **Next** until you reach the following screen where you have to define a query for the customer table.

P New Window	
SmartDataObject Wizard	
4GL Query:	You need to define the query that will be used in this SmartDataObject. (If the query involves temp- tables, you need to define them first.) Define Temp-Tables Define Query Help on Queries
? < <u>B</u> ack Next >	<u>Finish</u> Cancel

- 4. Click **Define Query**. The **Query Builder** dialog box opens and it lists the tables from the database that you are connected to.
- 5. Select **Customer** from the **Available Tables** section and click **Add**. The query is defined for the customer table and shown in the **Query** section.



P Query Builder		_	_	X
O Table ○ Join		) <u>S</u> ort	Options	
Database: worksho	p	•		
Available Tables:		Selected Ta	ables & Joins:	
Invoice Item Order OrderLine Outlet	Add >>	Customer		•
▼ 		4	Þ	~
No-lock -	]	Switch	Join <u>P</u> artners	
FOR EACH Customer NO-I	OCK INDEXED-	REPOSITIO	N:	*
				-
			• •	
✓ Indexed-Reposition	Chec	k Syntax:	<u>N</u> ow On <u>(</u>	<u>D</u> K
OK Canc	el Fre	eform <u>Q</u> uery	<u>t</u>	lelp

Tip: The Query Builder dialog box has five modes; Table, Where, Join, Sort, and Options. The Table mode allows you to add and remove tables to the selected query, The Where mode allows you to generate a WHERE clause to specify search criteria in ABL for the selected query. The Join mode allows you to create custom joins in ABL for the selected query. The Sort mode allows you to generate a SORT clause in ABL for the selected query. The Options mode allows you to modify the query attributes listed below and query tuning parameters for the selected query.

- 6. Click **OK** to close the **Query Builder** dialog box. The **4GL Query** section displays your new query.
- 7. Click Next.
- 8. Click Add fields. The Column Editor dialog box appears.
- 9. Click Add. The Multi-Field Selector dialog box appears.
- 10. Select an entry from the **Available Fields** and click **Add**. Similarly, move all the entries to the **Selected Fields** section.


P Multi-Field Selector		X
Available Fields:	Add >> << <u>Remove</u> Move <u>Up</u> Move <u>Down</u>	Selected Fields: CustNum Name Phone Address
	•	← →
OK Car	ncel	Help

- 11. Click **OK** to close the **Multi-Field Selector** dialog box. All the customer fields selected are listed for **Fields in SmartDataObject**.
- 12. Click **OK** to close the **Column Editor** dialog box.
- 13. Click **Next** and click **Finish**. You have now successfully created a SmartDataObject for the customer table to display all fields. The SmartDataObject is created and opened in Editor.



14. Expand the **SmartObjects** folder in **Project Explorer** view and observe that the dGetCust.w SmartDataObject file is created.

#### 2.6.2 Creating a SmartDataBrowser

Now, you will create a SmartDataBrowser which uses the above SmartDataObject that you created.

- 1. Select the SmartObjects folder from the Project Explorer view. Right-click and select New→ABL UI Design.
- 2. Select **SmartDataBrowser** in **Object type**. Make sure **Container** displays \OrderApp\SmartObjects.
- 3. Enter **bGetCust.w** in the **File name**.



4. Click Next until you reach the Data definition source section as shown in the screen below:



- 5. Click Browse.... The Open SmartObject dialog box appears.
- 6. Click **Browse...** and browse for **dGetCust.w SmartDataObject** created in C:\OpenEdge\WRK\Exchange\_PDSOE\OrderApp\SmartObjects\dGetCust.w and click **Open**.
- 7. Click Next. Click Add Fields next to the Fields to display section.
- 8. Select an entry from the **Available Fields** and click **Add**. Similarly, move all the entries to the **Selected Fields** section.
- 9. Click OK. All the fields are added to the Fields to display section.

Fields to display:
CustNum
Name
Phone
Address

10. Click Next and click Finish to create the SmartDataBrowser.

bGetCust.w (AppBuilder)	
SmartDataBrowser	oGetCust.w
CustNum Name	Phone Address

11. Open OrderApp.w file from the Project Explorer view. Select the OrderApp.w (AppBuilder) tab in the editor.

🏷 OrderApp.w (AppBuilder) 🖂

12. Double-click SmartDataObject in Palette. The Open SmartObject dialog box appears.



- 13. Click **Browse...** and select **dGetCust.w** file from the C:\OpenEdge\WRK\Exchange\_PDSOE\OrderApp\SmartObjects path. Click **Open**.
- 14. Click anywhere in **frameCustomer** (frame B). The SmartDataObject is added to the frame.



15. Repeat Step 12 to 14 to add **bGetCust.w** to **frameCustomer**. The **PROGRESS Advisor** dialog-box appears.



16. Click OK.



17. Click **OK** to add. A table is added to the editor.





- 18. Click **File** $\rightarrow$ **Save** to save the changes.
- 19. Right-click anywhere on the **OrderApp** window and select **Run**. The OrderApp dialog box is displayed. All customer records that you added are displayed in the table.

derApp			
Get Customer			
	Customer ID Customer Name	Phone	Address 🔺
	1 Andrews	617 555-1111	10 Smith St, Be 💻
	2 Justine Smith	617 333-3334	1342 Atlantic Av
Orden Fater	3 Kelly Bradford	212 444-4433	344 Brook St ,N
Order Entry	4 Frank Petersen	703 292-2222	123 Apple StAj
	5 Katherine Weiss	215 393-3309	323 South St. P
	6 Mark Wilson	708 838-3333	3233 Bradford {
	7 Andy Davis	617 828-2222	884 Pine Dr, Bc
	8 Morris Jenny	714 223-2332	90 Summer Rd
	9 Sanders Luke	919 338-3333	7738 Rex Dr, Cl
	10 Adams Marcy	617 989-8988	23212 Winthrop

20. Close the **Run** dialog box.



#### 2.6.3 Add SmartFilter to filter customer records (Optional)

You will now add a SmartFilter to filter the customer records based on phone numbers and get the corresponding customer records.

1. In **Palette**, click the down arrow to go to the next items in **SmartObjects** category.



2. Select **SmartFilter** and click **frameCustomer** (frame B). Depending on the size you specified for **frameCustomer**, you may see the following message. Click **OK** and resize the filter frame manually.



- 3. Click **OK** to close the message. The PROGRESS Advisor message appears. Click **OK** to add create link to smart object.
- 4. Adjust the filter and the customer table as shown in image below:





- 5. Right-click the filter and select Instance Properties.... The SmartFilter Properties dialog box appears.
- 6. Click Edit Field List.... Select the Phone entry from the Available Fields and click Add. It moves to Selected Fields.
- 7. Click OK to close the Multi-Field Selector dialog box. The Fields section displays Phone as shown in the image below:

SmartFilter P	roperties			×
Data Target:	smartobjects\dgetcust.w	Browse	Style	-
Fields:	Phone .	Edit Field List	Explicit     Range     Inline     String Operators     BEGINS     CO	INTAINS
-Field Prop Label:	erties	V Filter Target	Operator View as Ocombo-box	dio-set
Width:		Default	Size & Position Width of Character Fields:	20.00
Help ID:			Width of Other Fields:	16.00
	View as range fields     Explicit operator		Number of Lines in Editors:	1
		[	✓ View	
ОК	Cancel			Help



9. Click **OK** to close any error message if displayed. Drag filter control and adjust as shown in below image so that all controls are displayed properly.

	Phone:	Begins 💌	Apply Filter
ŀ			Blank

- 10. Click **File** $\rightarrow$ **Save** to save the changes.
- 11. Double-click **frameCustomer** to open the properties. Clear the **View Property** check box and select the **Hidden property** check box. This hides the frame at runtime.

Property Sh	eet - frameCustom	er		×
Object:	frameCustomer			
Title:				
Query:			A Query	Aa
	4			
Widget ID:	300			(?)
widgetib.				
Geometry				1 the
Column:	31.00	Width: 95.00	Virtual Width: 95.00	
Row:	1.00	Height 21.81	Virtual Height: 21.81	
Other Setting	15			
📝 3-D	,-	🔲 No-Hide	✓ Sensitive	
🔳 Down		🔲 No-Labels	Shared	
🔲 Drop-T	arget	📝 No-Underline	🗹 Show-popup	
📝 Hidden		🔲 No-Validate	🗹 Side-Labels	
📝 Кеер-Т	ab-Order	📝 Open the Query	🔲 Size to fit	
📃 No-Auti	o-Validate	📝 Overlay	🔲 Title Bar	
📃 No-Box	¢	Remove from Layou	it 📃 Use-Dict-Exps	
🗖 No-Hel	p	Scrollable 📃	🗖 View	
ОК	Canc	el <u>A</u> dvance	±	<u>H</u> elp



Now we will add a trigger to the **Get Customer** button so that frameCustomer is shown on clicking the button.

12. Select and right-click the **Get Customer** button in the OrderApp and click **Add Trigger**. The **Add Trigger** dialog box appears.

P Add Trigger	
Add Trigger Select the obje	ct to which you want to associate the event.
Widget:	butCustomer [BUTTON]
Event Category:	Common Events 🔹
Event:	CHOOSE
?	<u>G</u> enerate Cancel

- 13. By default, the **CHOOSE** event is selected. Click **Generate**. The source code view of OrderApp.w opens. Observe that the trigger code is generated.
- 14. From the **Outline** view, expand the **Triggers** section and select **ON CHOOSE OF butCustomer** node.
- 15. Click **Toggle Section Editor** in the **Outline** view.

🗄 Outline 🛛	💀 DB Structure	
Definition	ons	
Main Bl	ock	Toggle Section Editor



16. Observe that only the trigger code appears in the OrderApp.w source code.



**Note:** The **Section Editor** mode enables you to view a block of code of an AppBuilder (.w) procedure file while navigating to specific places in the code in the **Outline** view, instead of displaying the entire code of the procedure file.

The **Outline** view is not available with the design view. The **Outline** view lists all the structural elements of a procedure file only when the file is opened with the ABL Editor.

17. Add the following line inside the **DO** loop to view the customer frame when the user clicks the **Get Customer** button.

VIEW FRAME frameCustomer.

- 18. Click Toggle Section Editor in the Outline view once more to view the complete code.
- 19. Click Ctrl+I and observe that the code is indented to the right automatically.



**Note:** The ABL Editor can help you improve readability by automatically indenting lines as you type, when you paste text, or on demand. A separate option (tabular formatting) lets you automatically left-align like elements of statements within a code block, further improving readability. After copy pasting code or typing the code, press Ctrl+I or select **Correct Indentation** from editor context menu to correct indentation in the file.

- 20. Click **File** $\rightarrow$ **Save** to save the changes.
- 21. Right-click anywhere in the editor and select **Run**. The **OrderApp** dialog box opens.
- 22. Observe that by default frameCustomer is not shown.





- 23. Click Get Customer and observe that frameCustomer is displayed.
- 24. Enter first three digits of any phone number in the **Phone** field. Select filter criteria as **Begins** and click **Apply Filter**. Observe that the list is filtered to display the customer records that match the filter criteria.



This completes development of initial screen along with the logic through which you can retrieve existing customer details. When you click **Get Customer**, all the customer records are listed and you can filter the records by phone numbers.

💷 OrderApp				
Get Customer	Phone: 617	Begins		pplyFilter Blank Reset
	Customer ID Customer Name	Phone	Address 🔺	
	1 Andrews	617 555-1111	10 Smith St, Be	
	2 Justine Smith	617 333-3334	1342 Atlantic Av	
Order Fater	7 Andy Davis	617 828-2222	884 Pine Dr, Bc	
Order Entry	10 Adams Marcy	617 989-8988	23212 Winthrop	
	15 Dawsen Larry	617 383-3939	232 Westport A	
	17 Hall Mark	617 393-3331	3233 Echo Driv	

25. Close the **Run** window. Select **File**→**Close All**. All the files opened in the editor are closed.

## 2.7 Designing the order entry form (Take-home)

This section is a take-home section for you to develop a whole application at your leisure. It is not mandatory to perform this section in this workshop.

In this section, we will design an order entry form through which you can add items to the list and perform the billing function. Similar to the previous section, you will create a SmartDataObject called dOrderLine.w and SmartDataBrowser using OrderLine to create item tables to retrieve the ordered item details.

- 1. Create an ABL UI Design file, select **SmartDataObject** in **Object Type**, and **dOrderLine.w** in **File name**.
- 2. Open the **Query Builder** dialog box and add **OrderLine** and **Item** tables. Select the **Join** option button and join the two tables with condition **Item.ItemNum = OrderLine.ItemNum**:



3. Click Add in the Column Editor dialog box and add the fields in following order:





- 4. Create an ABL UI Design file, select **SmartDataBrowser** as **Object Type**, and **bOrderLine.w** as **File name**.
- 5. Define the data definition source smartobjects\dorderline.w in SmartDataObject.
- 6. Click Add in the Column Editor dialog box and add the fields in following order:



- 7. Double-click **OrderApp.w** from the **Project Explorer** view. The file is highlighted if it is already opened in the editor, if not the file opens in the editor.
- 8. Select **Frame** in **Palette** under the **Widgets** category. Click **frameCustomer.** A new frame is added.
- 9. From the **Outline** view, right-click **Frame-C** and select **Cut**.



- 10. Right-click wWin in the Outline view and select Paste. It adds a new frame under wWin.
- 11. Resize the new frame and place it on top of **frameCustomer**.

I OrderApp - OrderApp.w									8
: :	 			Frame (	2				
[·····	 : :					:	:	:	:
: : :	 :		: :			:		:	1
	 ÷ ÷						-	-	÷
F F F	 	• • • • • • • • • •						:	

- 12. Double-click the frame added in the editor. The **Property Sheet Frame–C** dialog box appears. Specify **frameOrdEntry** in **Object**. Clear the **Title Bar** check box under **Other Settings**.
- 13. Select OK to close the Property Sheet dialog box.



14. Right-click **frameOrdEntry** and select **Duplicate**. It adds a new frame to the window with same properties. Observe **FRAME-D** listed in the **Outline** view.



- 15. Right-click **FRAME-D** in the **Outline** view and select **Properties**. The **Property Sheet Frame D** dialog box appears. Specify **frameWelcome** in **Object**.
- 16. Select **OK** to close the **Property Sheet** dialog box. Click **File** $\rightarrow$ **Save** to save the changes.
- 17. Copy the **frameOrdEnt.w** file from

C:\OpenEdge\WRK\PDSOEWorkshopFiles\WorkshopFiles folder and paste in the **OrderApp\SmartObjects** folder in **Project Explorer** view.



- 18. Copy the **Images** folder from C:\OpenEdge\WRK\PDSOEWorkshopFiles\WorkshopFiles and paste in the **OrderApp** folder in **Project Explorer** view.
- 19. Close and reopen the OrderApp.w file.
- 20. From the **Outline** view, expand the **wWin→fMain** entry, right-click **frameOrdEntry**, and select **Move-to-Top**.
- 21. Select SmartDataObject from the SmartObjects section in the Palette.
- 22. Browse for the **SmartObjects\frameOrdEnt.w** file. The **PROGRESS Advisor** dialog appears. Click **OK** to add a SmartLink to SmartDataObject.
- 23. Confirm that **frameWelcome**, **frameCustomer** and **frameOrdEntry** should be of the same size and appear overlapping and also that you have added the SmartObject to **frameOrdEntry**.

I OrderApp - OrderApp.w			
	Customer:	Customer Quantity: 0	
	Category:		
	Price:		
	Extended Price: 0.00	)	Add
	Outlet Name:	•	
	Order Number Item Name	Price Extended Price	Quantity 🔺
Order Entry			
			Delete
	Amount: 0.00		T
	Total Paid: 0.00	0	
	Cash	Card Exact	
·····			

24. When you click **Run**, you will see that the order entry SmartFrame is added to frameOrdEntry.

- 25. Click **File** $\rightarrow$ **Save** to save the changes.
- 26. From the **Outline** view, right-click **frameWelcome FRAME** and select **Move-to-Top**.
- 27. Select **Image** from the **Widgets** section in **Palette**. Click **frameWelcome**. The Image control is added to the frame.
- 28. Double-click the image, the Property Sheet IMAGE-1 dialog box opens.
- 29. Click Image. The Choose Image dialog box appears.



Property Sheet - IMAGE	-2	-	X
Object: IMAGE-2			
Image			
adeicon/blank			
Tooltip:			
Widget ID: 2			
Geometry			
Column: 23.00	Width: 9.14	Left-Align	
Row: 5.85	Height: 2.46	Right-Align	
Other Settings			
Convert-3D-Colors	Remove from Layout	🔲 Transparent	
🔽 Enable	Retain-Shape		
🔲 Hidden	Stretch-to-Fit		
			L
OK Cano	cel Advanced		Help

#### 30. Browse for **exchange.png** in the

C:\OpenEdge\WRK\Exchange\_PDSOE\OrderApp\Images folder under the OrderApp project.

P Choose Image	
Eile:	C:\OpenEdge\WRK\Exchange_PDSOE\OrderApp\Im
exchange.png	PPOCPESS
(None) button-add.jpg deleteicon.png exchange.png	
File Type:	
Directory:	✓ Preview
C:\OpenEdge\WRK\Exchange_PD{ -	Edit Path Browse
OK Cancel	Help

- 31. Click **OK** to apply the selection of image.
- 32. Enter WelcomeImage in Object.



- 33. Select the **Transparent** check box in **Other Settings**. Click **OK** to close the **Image Property Sheet** dialog box.
- 34. Resize the image to view it completely. Right-click and select **Run** to run the file.
- 35. Click Yes in the Save Resource dialog, if asked for, to save the changes.



The Welcome frame displays the following image:



36. Close the **Run** window.

# 2.8 Linking Get Customer and Order Entry buttons to the respective frames (Take-home)

This section is a take-home section for you to develop a whole application at your leisure. It is not mandatory to perform this section in this workshop.

This section shows how to edit code in the source editor. You will add triggers to the buttons and the corresponding code to be executed for each trigger.

- 1. Right-click **OrderApp** and select **View Source** to view the source code editor.
- 2. Right-click anywhere in the editor and select Quick Outline. The Quick Outline view opens.

Ъ	View Design	Shift+F9
	Open With	•
	Show In	Alt+Shift+W ►
	Quick Outline	Ctrl+0

3. Select ON CHOOSE OF butCustomer under Triggers node.



**Tip: Quick Outline** is a pop-up window that shows a tree view of the ABL code file which is currently open in the ABL Editor. You can click a node in the tree view to navigate to a particular section of the file. It similar to the **Outline** view, but it is more convenient to work with.



4. Add following lines in the trigger to hide the other two frames created. Click **Ctrl+I** to indent code.

	HIDE FRAME frameWelcome. HIDE FRAME frameOrdEntry.
ß	*OrderApp.w (AppBuilder) VrderApp.w 🖂
	&Scoped-define SELF-NAME butCustomer ⊖&ANALYZE-SUSPEND UIB-CODE-BLOCK CONTROL butCustomer wWin
	ON CHOOSE OF butCustomer IN FRAME frameMenu /* Get Customer */ DO:
	HIDE FRAME frameWelcome. HIDE FRAME frameOrdEntry.
	VIEW FRAME frameCustomer. END.

- 5. Click **File** $\rightarrow$ **Save** to save the changes.
- 6. Select **Window** $\rightarrow$  **Preferences** to open the **Preferences** dialog box.
- 7. Select **Progress OpenEdge→AppBuilder→Editor**. By default, the **Make AppBuilder generated code read-only** check box is selected.

P Preferences	
type filter text	Editor 🗢 🔹 👻
<ul> <li>&gt; Java</li> <li>&gt; Java EE</li> <li>&gt; JavaScript</li> <li>&gt; Plug-in Development</li> <li>&gt; Progress Customer Supp</li> <li>&gt; Progress Databases</li> <li>= Progress OpenEdge</li> </ul>	Vake AppBuilder generated code read-only Read-only code background color:
<ul> <li>&gt; Advanced</li> <li>AppBuilder</li> <li>Editor</li> <li>Grid Units</li> <li>Widget ID</li> <li>&gt; BPM</li> <li>Business Rules</li> </ul>	
Database Connection Debug	Restore Defaults         Apply
?	OK Cancel

- 8. Click the color box for read-only code background color to any dark color like Click **Apply** and **OK** to close the **Preferences** dialog box.
- 9. Observe that read-only code is shown in selected color.



🏷 Ord	erApp.w (AppBuilder) 🛛 🕢 OrderApp.w 🖂
294	
295	&Scoped-define FRAME-NAME frameMenu
296	&Scoped-define SELF-NAME butCustomer
297⊝	&ANALYZE-SUSPEND _UIB-CODE-BLOCK _CONTROL butCustomer
<mark>298</mark> ⊖	ON CHOOSE OF butCustomer IN FRAME frameMenu /* Get Cu
299⊝	DO:
300	HIDE FRAME frameWelcome.
301	HIDE FRAME frameOrdEntry.
302	VIEW FRAME frameCustomer.
303	
304	END.

10. Try typing in the read-only section and observe that it is not allowed.

**Note:** You can make the sections of the AppBuilder-generated code read-only and foldable in the ABL Editor. This ensures that users do not modify the AppBuilder-generated code while editing the ABL procedure (.w) file in the ABL Editor. Editing the AppBuilder-generated code might corrupt the AppBuilder procedure (.w) file, and display errors when trying to open in the ABL GUI Designer.

The Find/Replace option does not work with the read-only or auto-generated code of an AppBuilder procedure file. You cannot rename an internal procedure or function name which is a part of the auto-generated or read-only code section.

#### 11. Add the CHOOSE trigger for the Order Entry button.

12. Add the following code to hide other frames and view the Order Entry frame.

```
HIDE FRAME frameCustomer. /*Hide frameCustomer*/
gOrderNum = NEXT-VALUE (NextOrderNum). /*Generate Order Number*/
VIEW FRAME frameOrdEntry IN WINDOW wWin. /*View frameOrdEntry*/
Run viewFrame in h_frameordent. /*Initialize default values in Order
frame*/
```

13. From the **Outline** view, expand the **Includes** section and select {**src/adm2/widgetprto.i**}. It marks the section in the source code view.

**Note:** The **Outline** view shows the structure of the code in the ABL Editor buffer that currently has focus, and provides an easy way to navigate to specific places in the code. The elements inside the inactive preprocessor regions appear in gray in the **Outline** view. This is similar to include nodes but without the include decorator.

The **Outline** view supports the following navigation techniques:

- Click an element to position the cursor at the declaration of that element. Black elements are declared in the current file.
- Double-click a black include file name to open that include file.
- Double-click an element labeled in gray type to open the Include file and position the cursor at the declaration of that element. Gray elements are declared in include files.
- 14. Press enter after the {src/adm2/widgetprto.i} include statement. Copy and paste the following code.



```
HIDE FRAME frameOrdEntry.
HIDE FRAME frameCustomer.
/*Create a global shared variable*/
&IF DEFINED(gCust) <> 1 &THEN
DEFINE NEW GLOBAL SHARED VARIABLE gCust AS CHARACTER.
DEFINE NEW GLOBAL SHARED VARIABLE gCustNum AS INTEGER.
&ENDIF
&IF DEFINED(gOrder) <> 1 &THEN
DEFINE NEW GLOBAL SHARED VARIABLE gOrder AS INTEGER.
&ENDIF
&IF DEFINED(gOrderNum) <> 1 &THEN
DEFINE NEW GLOBAL SHARED VARIABLE gOrderNum AS INTEGER
                                                         NO-UNDO.
&ENDIF
&IF DEFINED(gCust) <> 1 &THEN
DEFINE NEW GLOBAL SHARED VARIABLE gCust AS CHARACTER.
DEFINE NEW GLOBAL SHARED VARIABLE gCustNum AS INTEGER.
&ENDIF
```

- 15. Save the changes.
- 16. Right-click the source code editor and select **View Design**. The **Design** view of OrderApp opens.
- 17. Double-click **bGetCust.w** under the **SmartObjects** folder in **Project Explorer** view. The design view of SmartDataBrowser opens.

🏷 OrderApp.w (App	Builder)	🛛 Order	App.w	🏷 bGetCust	.w (AppBı	uilder) 🛛 🔤
SmartDataBro	owser - bGet(	Cust.w			X	
CustNum Name		Phone	Address			
					=	

18. Right-click and select Add Trigger. The Add Trigger dialog box appears. Select br\_table [BROWSE] in Widget and VALUE-CHANGED in Event.



P Add Trigger	
Add Trigger Select the obje	ct to which you want to associate the event.
Widget:	br_table [BROWSE]
Event Category:	Common Events
Event:	VALUE-CHANGED 🔹
?	Generate Cancel

- 19. Click **Generate**. Observe that the source view opens and the cursor is placed inside trigger code.
- 20. Add the following code after the Do loop:

```
gCust = {fnarg WidgetValue 'Name'} .
gCustNum = {fnarg WidgetValue 'CustNum'} .
It appears as follows:
ON VALUE-CHANGED OF br_table IN FRAME F-Main
DO:
    {src/adm2/brschnge.i}
    gCust = {fnarg WidgetValue 'Name'} .
    gCustNum = {fnarg WidgetValue 'CustNum'} .
END.
```

- 21. Expand Includes in the Outline view and select {src/adm2/widgetprto.i}
- 22. Press enter after the {src/adm2/widgetprto.i} include statement, copy and paste the following code to define the global variables. Save the changes.

```
&IF DEFINED(gCust) <> 1 &THEN
DEFINE NEW GLOBAL SHARED VARIABLE gCust AS CHARACTER.
DEFINE NEW GLOBAL SHARED VARIABLE gCustNum AS INTEGER.
&ENDIF
```

- 23. Double-click the **bOrderLine.w** file under the **SmartObjects** folder in the **Project Explorer** view to open file in the editor.
- 24. Right-click **SmartDataBrowser** and select **Add Trigger**. Ensure that **VALUE-CHANGED** is selected.



25. Copy and paste the following code in the Do loop:

```
gOrder = {fnarg WidgetValue 'OrderlineNum'} .
It appears as follows:
ON VALUE-CHANGED OF br_table IN FRAME F-Main
DO:
    {src/adm2/brschnge.i}
    gOrder = {fnarg WidgetValue 'OrderlineNum'} .
END.
```

26. Open the **Outline** view. Select {src/adm2/widgetprto.i} under **Includes** category as below.



27. In the source view, press enter after the {src/adm2/widgetprto.i} and copy and paste the following code to define global variables:



- 28. Click **File** $\rightarrow$ **Save All** to save all changes.
- 29. From the **Outline** view, right-click **frameMenu** and select **Properties**. The **frameMenu Property Sheet** dialog box opens.

30. Click . The **Choose Color** dialog box appears.



P Choose	Color																	×
						Sa	ampl	е										
Foreground	Colors:																	
																	Edit	]
	? 0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15		
Background	Colors:																	
																	Edit	
	? 0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15		
ОК		C	ance	el			<u>S</u> ave	e Col	lor S	etting	gs						<u>H</u> elp	

- 31. Select **Background Color** as and click **OK**. Click **OK** to close the **Properties** dialog box.
- 32. Repeat to select the **Background Color** as for the **frameCustomer**, **frameOrdEntry** and **frameWelcome** frames. Also, open the **frameOrdEnt.w** file and select the same background color using the **Properties** dialog box.
- 33. Click File→Save All to save the changes. This completes the development of OrderApp.
- 34. Select **OrderApp.w** (**AppBuilder**) in the editor. From workbench menu, select **Run→Run As→Progress OpenEdge Application**. The **OrderApp Run** dialog box displays:



35. Click Get Customer and select any customer record from the list.



CrderApp				
	Phone:	Begins	<ul> <li>✓ Apply Filt</li> <li>Blank</li> <li>Reset</li> </ul>	er
Get Customer				
	Customer ID Customer Nam	e Phone	Address 🔺	
	1 Andrews	617 555-1111	10 Smith St. Be	
	2 Justine Smith	617 333-3334	1342 Atlantic Av	
Order Entry	3 Kelly Bradford	212 444-4433	344 Brook St,N	
Order Entry	4 Frank Petersen	703 292-2222	123 Apple StA	
	5 Katherine Weis	s 215 393-3309	323 South St, P	
	6 Mark Wilson	708 838-3333	3233 Bradford (	
	7 Andy Davis	617 828-2222	884 Pine Dr, Bc	
	8 Morris Jenny	714 223-2332	90 Summer Rd	
	9 Sanders Luke	919 338-3333	7738 Rex Dr, Cl	
	10 Adams Marcy	617 989-8988	23212 Winthrop	

36. Click **Order Entry**. Observe that the selected customer name is used. Select the items, enter the quantity, and click **Add**. The selected items are added to list.

OrderApp			×
	Custome	r: Kelly Bradford	
	Item Name: Veggie Special	Quantity: <sup>2</sup>	
	Category: Price:	27.20	
_	Extended Price:	54.40	Add
Get Customer	Outlet Name: Atla	ntic Ave 🔹	
	Order Number Item Name	Price Extended Price	Quantity 🔺
	11 Veggie Special	27.20 54.40	2
Order Entry			Delete
	Amount	54.40	
	Total Paid:	0.00	
	Cash	Card Exact	

- 37. Enter the amount in Total Paid, and click Exact. The Order Completed message appears.
- 38. Click **File** $\rightarrow$ **Close All** to close all files in editor.



# LAB 03: Application development using Progress Developer Studio for OpenEdge WebSpeed

## 3.1 Overview

The sections of this lab show you how to create a WebSpeed project and work with the new SpeedScript editor. You will also learn how to assign a SpeedScript file to the WebSpeed server, and run the file on the server.

You will develop a web application as shown below to view the customer records and to add records to the database.

Jump to: Submit								
		Re	esults List:					
1	Andrews	617 555-1111	10 Smith St, Bedford					
2	Justine Smith	617 333-3334	1342 Atlantic Ave, Apt 345b, Boston, MA					
3	Kelly Bradford	212 444-4433	344 Brook St ,New York					
4	Frank Petersen	703 292-2222	123 Apple St, Apt 34A, VA					
5	Katherine Weiss	215 393-3309	323 South St, Philadelphia					
		•						

mer Details
/8576
3rd floor, ch City, abad
r



#### 3.2 Prerequisites

Complete Lab 01: Configuring your workspace and customizing the project prior to working on this lab.

#### 3.3 Creating a WebSpeed project

This section shows how to create a WebSpeed project.

- 1. From the workbench menu, select **File→New→OpenEdge Project.** The **New OpenEdge Project** dialog box appears.
- 2. Specify WebReport in Project name.
- 3. Select WebSpeed from the drop-down list of Project Type configuration.
- 4. Click Next. The Select AVM and layout options dialog box appears.
- Click Next. The Define WebSpeed dynamic content module dialog box appears. The WebSpeed source folder displays WebSpeed. All the files created under this WebSpeed folder will be published to selected WebSpeed server.

P New OpenEdge Project					
Define WebSpeed dynamic content module					
Enter a name for the folder to contain WebSpeed source code.					
Module name:	WebReport				
WebSpeed source folder:	WebSpeed				
Supported servers:					
Server Name	Server Name				
🔲 🔲 🥂 wsbroker1 WebS	Speed 11.3 at nbhydsyellava				
R wsdynamics1 WebSpeed 11.3 at nbhydsyellava					

**Tip**: By default the project name appears in **Module name**. Module is a unit of files and folders which will be published to the server. Under a project, the folders that should be considered for this module can be configured from the **Modules** dialog box in the **Project** properties.

- 6. Click **Next** until the **Define PROPATH** dialog box appears. The WebSpeed source folder is added to the PROPATH.
- 7. Click Next. The Select database connections dialog box appears.
- 8. Select the check box next to Exchange\_db and click Finish to create the project.
- 9. The **Open Associated Perspective** dialog box appears. Click **Yes**. The **OpenEdge Server** perspective opens. The project is created along with the respective folders.





## 3.4 Creating the SpeedScript file

This section shows you how to create a SpeedScript file using an existing template. We will create a report that lists all the employees from the database.

- 1. Right-click anywhere in the **Project Explorer** view and select **New**→**SpeedScript**. The **New SpeedScript Wizard** dialog box appears.
- 2. Select the WebSpeed folder and enter CustomerList.html in File name.
- 3. Click **Next**. The **Select SpeedScript Template** dialog box appears. The default WebSpeed HTML templates are listed. When you choose a template, its HTML markup appears in the **Preview** pane.
- 4. Select **WebSpeed Report** from the list. The **WebSpeed Report** template has all the predefined code for generating a report.



5. Click **Finish** to create the file.

P New SpeedScript Wizard	
Select SpeedScript Template Select a template as initial content in the Speed	dScript page.
✓ <u>U</u> se WebSpeed Template <u>T</u> emplates:	
Name	Description
WebSpeed Table	WebSpeed Table Template
WebSpeed Report	WebSpeed Report Template
WebSpeed Main	WebSpeed Main Template
WebSpeed Frameset	WebSpeed Frameset Template
WebSpeed Blank	WebSpeed Blank Template
Preview:	
<pre><!DOCTYPE HTML PUBLIC "-//IETF//DTD <HTML>     <head> <title>Browse</title> <script language="SpeedScript"></script></head></pre>	

The file is created and opens in editor. The **SpeedScript** node appears in the **Outline** view. The outline of ABL code inside SpeedScript appears as follows:



6. In the source view of CustomerList.html file, find the code with SCOPED-DEFINE preprocessor, replace all &SCOPED-DEFINE statements with the following code:



```
&SCOPED-DEFINE Query-Table Customer
&SCOPED-DEFINE Query-Field Name
&SCOPED-DEFINE Query-Index CustNumIdx
&SCOPED-DEFINE Filter-Field Phone
&SCOPED-DEFINE Display-Fields " <TR><TD>" CustNum "</TD><TD>" Name
"</TD><TD>" Phone "</TD><TD>" Address "</TD></TR>"
&SCOPED-DEFINE Result-Rows 5
```

By default, the template code refers to tables in Sports2000 database which is a sample database shipped with the product. The code above points to the table, field, and index information corresponding to the database that we are using for this workshop.

7. Find and change the Body type background color to:

```
<BODY STYLE="background-color: #FFF8C6">
```

- 8. Save the changes.
- 9. Right-click anywhere on the editor and select **Check Syntax** from the context menu. You should see the following message.

P Cheo	:k Syntax	×
1	Syntax check: OK	
		ОК

10. Click **OK** to close message.

# 3.5 Configuring the WebSpeed server and associating the WebSpeed module

This section shows you how to work with the WebSpeed servers, associate module to the server, edit the server properties, start the server, and publish the modules.

1. Select the **Servers** view.

2	Console	Problems	Tasks	해 Servers 8	X	🛱 Progress OpenEdge Server Monitor
	📲 asbro	ker1 AppServ	er 11.3 at	nbhydsyella	va	[Stopped]
	📲 bpsbi	roker1 AppSe	rver 11.3 a	t nbhydsyell	av	/a [Stopped]
	🔏 esbbr	roker1 AppSe	ver 11.3 a	t nbhydsyell	av	/a [Stopped]
	🔏 icfrep	oos AppServer	11.3 at nl	ohydsyellava	[	[Stopped]
	🔏 Progr	ress Business	Process Se	rver for nbh	yd	isyellava [Stopped]
	📲 restbi	roker1 AppSe	rver 11.3 a	t nbhydsyell	av	/a [Stopped]
	况 restm	ngr1 OE Web S	Server 11.3	at nbhydsy	ell	lava [Stopped]
	🔏 wsbro	oker1 WebSpe	ed 11.3 a	t nbhydsyella	ava	a [Stopped]
	🔏 wsdyi	namics1 Web	Speed 11.	3 at nbhydsy	ell	lava [Stopped]

- 2. Double-click the **wsbroker1 WebSpeed** server. The **Server Configuration** dialog box opens in editor.
- 3. Expand Web Server URL.



		1			
General Information		Publishing			
specify the nost name an	a other common settings.	Timeouts			
Server name:	wsbroker1 WebSpeed 11.3 at nbhydsyellava				
Host name:	localhost	Publish Location			
Runtime Environment:	OpenEdge WebSpeed 11.3	- Web Server URL			
Open launch configurat	ion	Modify Web Server U	rl		
Connection		Web Server URL:	http://nbhydsyellava/Scripts/cgiip.exe/WService=wsbroker1		
Specify the information	for connection to the OpenEdge Explorer.	Web Server:		•	Add.
OpenEdge Explorer con	nostion: Explorer 1	CGIIP URL:	/Scripts/cgiip.exe		
OpenLuge Explorer con					
Broker name:	wsbroker1 - nbhydsyellava 🔻 🤣				



**Tip**: You use the Server Editor to view or modify the server properties that define the connection to OpenEdge Explorer and the broker.

The Server Editor provides information on the following:

- ${\boldsymbol{\cdot}}$  General Information  ${\boldsymbol{\cdot}}$  Provides the host name and other common settings.
- $\bullet$  Connection Specifies the information on connection to OpenEdge Explorer.
- Publishing Specifies when to publish.
- •Timeouts Specifies the time limit to complete server operations (Start and Stop).
- Publish Location Specifies the server publish directory.
- 4. Add **8080** port (on which our Apache web server is running) and remove **Scripts** and add **cgibin** to the URL that appears by default, as shown below:

Web Server URL: http://nbhydsyellava:8080/cgi-bin/cgiip.exe/WService=wsbroker1

- 5. Click File→Save.
- 6. Right-click **wsbroker1 WebSpeed** in the **Servers** view and select **Add and Remove...**. The **Add and Remove** dialog box appears.

Overview	ħ	Add and Remove Monitoring	•	
💷 Console 🔝 Problems ⁄ Tasks 🤻 Serve		Clear Compile Errors		
📲 asbroker1 AppServer 11.3 at nbhydsy	47	Trim all Agents	Ctrl+Alt+K	Г
腸 bpsbroker1 AppServer 11.3 at nbhyd	-	Trim Agents	Ctrl+Alt+T	
Result of the set of t	42	Add Agents	Ctrl+Alt+A	
🚟 icfrepos AppServer 11.3 at nbhydsyel	101	Launch WebSneed WorkShop		
🖓 Progress Business Process Server for		OpenEdge Explorer		
🚟 restbroker1 AppServer 11.3 at nbhyd:	1 1 1 1	Server Monitor		
🔚 restmgr1 OE Web Server 11.3 at nbhy		Properties	Alt+Enter	
🖓 wsbroker1 WebSpeed 11.3 at nbhyds	Rewsbroker1 WebSpeed 11.3 at nbhydsyenava [Stopped]			
Revealed with the second secon	/dsye	ellava [Stopped]		

7. Select WebReport from the Available section and click Add.



P Add and Remove	and a subscript		
Add and Remove Modify the resources that are configur	ed on the server		
Move resources to the right to configur	e them on the server		
<u>A</u> vailable:		<u>C</u> onfigured:	
	A <u>d</u> d >	🏭 WebReport	

8. Click **Finish**. The **WebReport** module is added and listed under **wsbroker1** server in the **Servers** view.



We are now done with creating an html file to display all the records from customer table. We have also associated the module to our server. The next step is to start our server.

- 9. From the workbench menu, select **Run→Run Configurations**. The **Run Configurations** dialog box appears. All the **Run configurations** grouped by category are listed.
- 10. Expand Progress OpenEdge WebSpeed and select wsbroker1 WebSpeed.

P Run Configurations		-		X
Create, manage, and run configur Create a configuration to start an Open	<b>ations</b> Edge WebSpeed broker.			
Image: Second Secon	Name: wsbroker1 Wet Server Startup + This launch configurati further options for con Server: Buntime Environment:	oSpeed 11.3 at nbhydsyellava PROPATH Databases I a so ion can be used to launch the see wsbroker1 WebSpeed 11.3 at r OpenErde WebSpeed 11.3	Security 🗆 Common erver specified below. To acce rver's editor from the Servers hbhydsyellava	ss view.
Ju JUnit Jü JUnit Plug-in Test	Host name:	localhost		
Filter matched 22 of 22 items			Appl <u>y</u> Re <u>v</u>	ert
?			<u>R</u> un Cle	ose



Note: A Run configuration defines the characteristics of the AVM instance under which the selected program runs. These characteristics include for example, startup parameters, PROPATH settings, and environment settings for the AVM session, database connections, and whether the program uses a dedicated instance of the AVM or the instance under which your OpenEdge project is currently running.

You can use the Run Configurations wizard to define all of a Run configuration's characteristics. Although this wizard contains a large number of fields on multiple tabs, defining a Run configuration need not be a complicated task. In fact, with a single click, you can create and run a configuration that uses default settings, and then edit any of these settings, if necessary.

- 11. Select the **Databases** tab. Select the **Show All** option button to show all database connections available in the workspace and you see the **Exchange\_db** connection listed.
- 12. Select the check box next to Exchange\_db. Click Apply and Run to start the server.

The server is started and the status changes to **Started** in the **Servers** view.

R wsbroker1 WebSpeed 11.3 at nbhydsyellava [Started, Republish]
 WebReport

We will now publish our code to the WebSpeed broker.

13. Right-click **wsbroker1 WebSpeed** server and select **Publish**. The server status changes to **Synchronized** which means that both our module and the server are in sync.



### 3.6 Running the SpeedScript file on a WebSpeed server

In the above section, we have seen how we can create an html file using default templates and with minimum coding to display records from a particular table.

This section shows how to run the SpeedScript file on the server.

- 1. Select CustomList.html tab in the editor.
- 2. Right-click and select **Run As→Run on Server**. The **Run on Server** dialog box appears and lists all the WebSpeed servers available.



P Run On Server	
Run On Server Select which server to use	
How do you want to select the server? Choose an existing server Manually define a new server Select the server that you want to use: type filter text For a constant of the server	
OpenEdge AVM Runtime	
⑦ < <u>B</u> ack <u>N</u> ext > <u>Finish</u>	Cancel

3. Select the wsbroker1 WebSpeed server and click Finish.

The SpeedScript file runs on wsbroker1 server and the result appears.

🏇 CustomerList.html 🛛 🕂 wsbroker	r1 WebSpeed 11.3 at nbhydsyella	iva 🔇 🚱 Browse 🖂	
🗇 🗇 🔳 🦑 http://nbhydsyellava	a:8080/cgi-bin/cgiip.exe/WServic	e=wsbroker1/CustomerList.html	▼ ▶
	Jump to:	Submit	•
	I	Results List:	
1 And	drews 617 555-111	1 10 Smith St, Bedford	
2 Just	tine Smith 617 333-333	4 1342 Atlantic Ave, Apt 345b, Boston, MA	
3 Kel	lly Bradford 212 444-443	3 344 Brook St ,New York	
4 Fra	ank Petersen 703 292-222	2 123 Apple St,Apt 34A, VA	
5 Kat	therine Weiss 215 393-330	9 323 South St, Philadelphia	
	<u> </u>		
			+

# 3.7 Adding an AppServer facet to the WebSpeed project

This section shows how to add an AppServer facet to the existing WebSpeed project.

1. Select the WebReport WebSpeed project in Project Explorer view.



- 2. Right-click and select Properties. The Properties for WebReport dialog box appears.
- 3. Click the **Project Facets** node. The OpenEdge, JavaScript, and WebSpeed facets are selected by default.

Properties for WebRep	ort _□×
type filter text	Project Facets
	Configuration: <custom> Save As Delete</custom>
Deployment Assembl B JavaScript Project Sectors Refactoring History Run/Debug Settings Server Targeted Runtimes Task Tags Validation XDoclet	Project Facet       Version       ▲         □ □ CKF 2.x Web Services       1.0         □ □ CKF 2.x Web Module       3.0       -         □ □ EB Module       3.1       -         □ □ EB Module       3.1       -         □ □ DavaScript       1.0       -         □ JavaScript       1.0       -         □ JavaScript       1.6       -         □ JAVA RGUE       1.6       -         □ JAVA RGUE       1.6       -         □ DCA Module       1.1       -         □ DCA Module       1.0       -         □ DCA Module       -       -
• • • • • • • • • • • • • • • • • • •	Revert Apply
?	OK Cancel

- 4. Select the **AppServer** check box to add the facet to project.
- 5. Click Further configuration available... link. The Define AppServer content module dialog box appears. By default, the AppServer folder appears as the AppServer source folder.



P Modify Faceted Project				
Define AppServer content module Enter a name for the folder to contain AppServer source code.				
	MahDanast		<u> </u>	
AppServer source folder:	AppServer			
Supported servers:				
Server Name				
Box         bysbroker1 AppSe           Box         esbbroker1 AppSe           Box         icfrepos AppServe           Box         restbroker1 AppSe	rver 11 rver 11 r 11.3 a erver 11			
Publish changes imme	diately			
?		< <u>B</u> ack	<u>N</u> ext >	ОК

- 6. Select the check box next to **restbroker1 AppServer** to publish the AppServer files of this project to the restbroker1 AppServer.
- 7. Click **Next**. The **Define PROPATH** dialog box appears. The AppServer source folder is added to PROPATH.



- 8. Click Next. The Select database connection dialog box appears.
- 9. Click **OK** in the **Modify faceted Project** dialog box.



10. Click **OK** to close the **Properties** page.

The AppServer folder is created under the WebReport project.



## 3.8 Working with the SpeedScript editor

This section shows you how to work with the SpeedScript editor. We will learn how to create a SpeedScript file from a blank template and run it on the server.

In Progress Developer Studio for OpenEdge, you edit the embedded SpeedScript files in the OpenEdge SpeedScript editor. Embedded SpeedScript files are HTML files that contain SpeedScript (a subset of ABL) code contained within HTML script elements. The OpenEdge SpeedScript editor supports editing HTML, JavaScript, CSS, and SpeedScript.

When you double-click a file that has an .htm or .html extension in Progress Developer Studio for OpenEdge, the file opens in the SpeedScript editor by default. The SpeedScript editor assists in writing code including features such as content assist, case correction, keyword expansion, hover help and so on.

Note that the SpeedScript editing features do not apply to HTML, JavaScript, or CSS elements in a SpeedScript file. For example, case correcting (Source > Correct case) only applies to the SpeedScript elements.

Most SpeedScript editing features are available from the main context menu, or under the source node of the context menu.

- 1. Create **CustomerEntry.html** SpeedScript file under the **WebSpeed** folder using **WebSpeed Blank Template**. The SpeedScript file opens in the editor.
- 2. Find the **SpeedScript** tag, type **Define**, add a space and press **Ctrl+Space**. Observe that content assist functions in a similar way as in ABL editor.


🏇 CustomerList	.html 🛛 🕂 wsbroker1 WebSpeed 11.3 at nbhydsyellava	Browse	🐝 *CustomerEntry.html 🔀	- 0
1 DOCTYP<br 2 <html> 3 <head> 4 <meta na<br=""/>5 <titlesm 6 <script 7 /* Creat 8 This 9 inter 10 prope 11 CREATE W 12 </script 13 14 15 &lt;800Y&gt; 16 <script 17 /* 18 File 19 Desc 20 Creat 20 Cre</script </titlesm </head></html>	<pre>E HTML PUBLIC "-//IETF//DTD HTML//EN"&gt; ME="AUTHOR" CONTENT="Your Name"&gt; lebSpeed Script LANGUAGE="SpeedScript"&gt; e an unnamed pool to store all the widgets creation and procedures will execute in this procedure's r cleanup will occur on deletion of the procedu IDGET-POOL. &gt; LANGUAGE="SpeedScript"&gt; ription: ted:</pre>	ted by this p dure's trigge storage, and re. */	procedure. ers and I that	*
22 DEFINE 23 24 25 4 SpeedScript Ge	ABSTRACT - OPTION BROWSE - OPTION BUFFER - OPTION DATA-SOURCE - OPTION DATA-SOURCE - OPTION DATASET - OPTION EVENT	Defines the c define any n members, as to interface i members m or behavior, hierarchy wh example, an method can	lass as abstract. An abstract class allows you to umber of instance property, method, or event abstract. Abstract members are prototypes, similar members, without an implementation. Abstract us to eimplemented by a derived class to provide data but they also function polymorphically in the class nerever they are still defined as abstract. So, for abstract property can be accessed or an abstract be called and the result depends on its	-

3. Right-click on the editor and observe that the **Source** menu displays the ABL options.

Source •	Cleanup Document
Properties	Format Format Active Elements
Check Syntax Compile Keyword Help Shift+F2 Show References Ctrl+Shift+G Quick Qutline	Correct Case Expand Keywords Ctrl+Shift+1 Surround With Adt Function Alt+Shift+F
Open Selection	Add Procedure Alt+Shift+P

4. Delete the whole code from the editor and add the following code:

```
<!DOCTYPE HTML PUBLIC "-//IETF//DTD HTML//EN">
<HTML>
<HEAD>
<META NAME="AUTHOR" CONTENT="Your Name">
<TITLE>WebSpeed Script</TITLE>
<SCRIPT LANGUAGE="SpeedScript">
/* Create an unnamed pool to store all the widgets created by this
procedure.
   This is a good default which assures that this procedure's triggers
and
   internal procedures will execute in this procedure's storage, and
that
   proper cleanup will occur on deletion of the procedure. */
CREATE WIDGET-POOL.
</SCRIPT>
</HEAD>
<BODY STYLE="background-color: #FFF8C6">
<form method="POST">
                                               PROGRESS
```

```
<DIV STYLE="text-align: center;">
<image src="`RootURL`/images/exchange.png">
<SCRIPT LANGUAGE="SpeedScript">
      DEFINE VARIABLE CustNum AS INTEGER NO-UNDO.
   DEFINE VARIABLE CustName AS CHARACTER NO-UNDO.
   DEFINE VARIABLE Phone AS CHARACTER NO-UNDO.
   DEFINE VARIABLE Address AS CHARACTER NO-UNDO.
   DEFINE TEMP-TABLE ttCustomer LIKE Customer.
   DEFINE VARIABLE prochandle AS HANDLE.
   DEFINE VARIABLE serverHandle AS HANDLE.
</SCRIPT>
<TABLE border=2>
<TR><TH colspan=2>Customer Details</TH></TR>
<TR><TD>Name: </TD><TD><INPUT TYPE="Text" NAME="CustName"
VALUE="`CustName`"></TD></TR>
<TR><TD>Phone: </TD><TD><INPUT TYPE="Text" NAME="Phone"
VALUE="`Phone`"></TD></TR>
<TR><TD>Address: </TD><TD><TextArea Name="Address"
rows="4">`Address`</TextArea></TD></TR>
<TR></TR>
<TR></TR>
</TABLE>
<INPUT TYPE="submit" NAME="Submit" VALUE="Add Customer" >
</DIV>
<script language="speedscript">
*/
IF get-value("Submit") EQ "Add Customer" THEN
       D0:
IF GET-VALUE ("CustName") NE "" THEN DO:
CREATE ttCustomer.
ASSIGN ttCustomer.Name = GET-VALUE ("CustName")
      ttCustomer.CustNum = NEXT-VALUE (NextCustNum)
         ttCustomer.Phone = get-Value("Phone")
 ttCustomer.Address = get-value("Address").
   CREATE SERVER serverHandle.
    DO ON ERROR UNDO, THROW:
        serverHandle:CONNECT("-AppService restbroker1 -H
localhost -sessionModel Session-free").
        RUN custom.p PERSISTENT SET prochandle ON SERVER
serverHandle.
        RUN createCustomer IN prochandle (INPUT-OUTPUT TABLE
ttCustomer).
    FINALLY:
        DELETE PROCEDURE procHandle NO-ERROR.
        serverHandle:DISCONNECT () NO-ERROR.
       DELETE OBJECT serverHandle NO-ERROR.
    END FINALLY.
    END.
 END.
                                        *PROGRESS
```

5. Save the file and click the **Preview** tab at the bottom of the SpeedScript editor. The preview of the file is shown. The **Preview** tab will let you preview your result without running the code.

🏇 CustomerList.html	🎯 Br	rowse 🥢 🏇 CustomerEn	try.html	× "1	
		×			*
		Customer Details			
	Name:	`CustName`			
	Phone:	`Phone`			
	Address:		*		
			Ŧ		
		Add Customer			
					Ŧ
SpeedScript Generate	ed ABL Prev	view			

 Copy and paste the custom.p business logic procedure file from the C:\OpenEdge\WRK\PDSOEWorkshopFiles\WorkshopFiles folder to AppServer folder.



**Note:** Only the files that exist under the AppServer folder gets published to the server. If you want other folder files also to get published to server then those need to be included as module. You can configure it from the **Project properties Modules** dialog box.

7. Open the file and observe that the procedures for Create, Delete, Read, and Update of customer records are available.



- 8. From the workbench menu, select **Run→Run Configurations**. The **Run Configurations** dialog box appears.
- 9. Expand Progress OpenEdge AppServer and select restbroker1 AppServer.

C (a) × (a) ×		Na	ame: restbroker1 App	erver 11.	at nbhydsyellava	3					
type filter text		R	Server 🔯 Startup 4	🖌 PROP	ATH 🤣 Databas	es 🖂 Security 🔲	Common				
Jtr JUnit Plug-in Test	*		Show selected	how all				<u>Configur</u>	e databas	e conn	ections
Progress OpenEdge Application     frameOrdEnt			Connection Name	Group	Physical Name	Other Parameters	Description	Logical Name	Aliases	Host	Servi
OF Graphs			🔽 🤣 Exchange_db		C:\OpenEdg					loc	6210
0E OrderApp											
OE Reports											
OE SalesReport											
OE TempTbl											
4 🔯 Progress OpenEdge AppServer		11.									
🔯 asbroker1 AppServer 11.3 at nbhydsyellava											
🔯 bpsbroker1 AppServer 11.3 at nbhydsyellava	=		•		m						F.
Session of the sessio											
icfrepos AppServer 11.3 at nbhydsyellava			onnection string:								
restbroker1 AppServer 11.3 at nbhydsyellava											*

- 10. Select the **Databases** tab. Select the **Show All** option button to show all the database connections available in workspace. The **Exchange\_db** connection is listed.
- 11. Select the check box next to Exchange\_db. Click Apply and Run to start the server.
- 12. The server is started. The state of the server appears as Started [Synchronized].

restbroker1 AppServer 11.3 at nbhydsyellava [Started, Synchronized]
WebReport [Synchronized]

**Note:** The Publishing option automatically publishes when resources change is selected by default for server. So changes will be automatically published to the server.



- 13. Select the CustomerEntry.html tab in the editor.
- 14. Right-click and select **Run As→Run on Server**. The **Run on Server** dialog box appears. All the WebSpeed servers available are listed.
- 15. Select **wsbroker1 WebSpeed** server and click **Finish**. The SpeedScript file is run on wsbroker1 server and the result appears.
- 16. Enter the details as shown below and click Add Customer.



F	PR( X(	DGRESS HANGE
		Customer Details
	Name:	Shiva
	Phone:	9844378576
	Address:	ILabs 3rd floor, A Hi-Tech City, Hyderabad
		Add Customer

The CreateCustomer procedure in the custom.p file is executed in AppServer restbroker1.

17. Open the restbroker1 server log file from C:\OpenEdge\WRK\restbroker1.server.log. The following message appears in the log file.

P-014192 T-008324 1 AS -- (Procedure: 'CreateCustomer custom.p' Line:51) Customer Created

18. To confirm that the customer record is created in database, from workbench menu, select Window→Show View→ABL Scratchpad.

Window Help				
New Window		o Syi 🔐 🍭 🗿 🗿 🧏	- 🖓 - 🏷	⇔
New Editor		eed 11.3 at nbhydsyellava	Browse	🏇 CustomerEntry
Open Perspective	•	hin/caiin avo/M/Sonvico-webr	okor1/Customor	Enterberg
Show View	•	ABL Scratchpad		Ctrl+F9

The ABL Scratchpad view opens.

Console	🖹 Problems	🙆 Tasks	워 Servers	🙀 Progress OpenEdge Sen	ver Monitor	ABL	Scratch	pad	×			
							‰ 0	•	B. (	🖉 Shared A	VM	•
I												~
4											•	-
Tab 1												-1



Tip: ABL Scratchpad is a multi-tabbed view provided by PDS OE. It allows you to write and test the ABL code without having to save it or creating a new file or project. By default, the shared AVM runtime is used for the execution of the code. However, it provides the option of selecting the runtime of any existing OpenEdge projects for execution. This saves time and avoids the need to create a project or document multiple times for testing the code in different runtimes.

19. Copy and paste the following code in Scratchpad editor:

```
FIND Customer WHERE Customer.Name = "Shiva".
DISPLAY Customer.
```

**Note:** Modify the **Customer.Name** value as per the name provided in the 16<sup>th</sup> step to add a Customer.

20. Select the OrderApp project AVM on which this code needs to run.

📮 Console	🖹 Probl	ems	🖉 Tasks	해 Servers	월 Progress OpenEdge Server	Monitor	ABL	. Scra	tchpac	1 23		- [
							C.	%	0 -		8	OrderApp
ETND Cus	tomer WH	IFRE	Customer	r Name = '	"Shiva"							Shared AVM
DISP	LAY Cust	tomer	`.	- Hunce -	SHIVE .							OrderApp VDReports WebReport
												-
												•
Tab 1												

**Caution:** If WebReport is selected, you will see a message to unhide tty window as WebSpeed project uses TTY Runtime. In the Progress OpenEdge Project Properties page, select **Hide TTY runtime console** check box to suppress the display of the project-specific runtime console by default. This setting needs to be changed to view TTY Console. This setting is changeable only if **Use shared AVM** check box is cleared and the **Use TTY for Runtime** check box is selected.

21. Click **Run** <sup>(2)</sup>. The **Run** dialog box displays the created customer record.

0E Progress		
Customer ID Customer Nam	e Address	Phone
25 Shiva	ILabs 3rd floor,Hi-	9844378576

22. Press the SPACEBAR to close the Run dialog box.



# 4 LAB 04: Application development using Progress Developer Studio for OpenEdge – REST

### 4.1 Overview

**RE**presentational State Transfer (**REST**) is not a protocol; it is a set of principles or an architectural style.

RESTful applications use HTTP requests to post data (create and/or update), read data (e.g., make queries), and delete data. Thus, REST uses HTTP for all four CRUD (Create/Read/Update/Delete) operations. It is a lightweight alternative to mechanisms like RPC (Remote Procedure Calls) and Web Services (SOAP, WSDL). Every operation in REST is operated on a resource.

The sections of this lab show you how to expose the ABL resources as REST services and work with them. We will create a REST service and invoke this service using a generic REST client. We will be passing Customer id as input from the REST client and will retrieve the record of that Customer.

### 4.2 Prerequisites

Complete Lab 01: Configuring your workspace and customizing the project and Lab 03: Application development using Progress Developer Studio for OpenEdge – WebSpeed prior to working on this lab.

## 4.3 Adding a REST facet to the WebSpeed project

This section shows you how to add a new REST facet to an existing WebSpeed project. We will use the same resources from the WebReport project that we created in the above section.

- 1. Select the WebReport WebSpeed project in the Project Explorer view.
- 2. Right-click and select Properties. The Properties for WebReport dialog box appears.
- 3. Select the **Project Facets** node.



P Properties for WebRep	oort		
type filter text	Project Facets		$\div \bullet \bullet \bullet \bullet$
	Conformations Longitude		E Court As
Builders	Configuration:   <custom></custom>		Save As Delete
Deployment Assembl	Project Facet V	ersion 🔺	Details Runtimes
	ChUI		Application Client module 6.0
Progress OpenEdge	CXF 2.x Web Services 1	.0	Application client module 6.0
Project Facets	Dynamic Web Module 3	.0 • 0.	Enables the project to be deployed as a Java EE
Project References	Dynamics		Application Client module.
Refactoring History	EAR 6	.0 -	Requires the following facet:
Run/Debug Settings	EJB Module 3	.1 •	Java 1.6 or newer
Server		.2.3 •	Conflicts with the following facets:
Targeted Puntimer		7 .	connects with the following facets.
Targeteu Kuntimes	Java 1	., .	Application Client module
Task Tags	□ □ □ JavaScript 1	.0 0 •	Dynamic Web Module
Validation	□ □ JAX-RS (REST Web Services) 1	.1 •	E EAR
XDoclet	JCA Module 1	.6 •	ICA Module
	Mobile		Static Web Module
	✓ Teme OpenEdge 1	1 •	
	□ 聶 OpenEdge BPM		Web Fragment Module
	Progress Adapters 1	.0	
	REST		
	🔲 📄 Static Web Module		
	Utility Module		
	Web Fragment Module		
	WebDoclet (XDoclet) 1	.2.3 •	
	✓ WebSpeed	-	
	1		l <u></u>
•			Revert <u>A</u> pply
0			
U			OK Cancel

4. Select the check box next to **REST** to add a facet to the project. The following error message appears:

🔕 REST requires Progress Adapters 1.0.

**Note:** REST facet is dependent on AppServer and Progress Adapters facets. As AppServer facet is already selected, the error displayed is only for Progress Adapters.

- 5. Select the check box next to **Progress Adapters**. It clears the error message.
- 6. Click **Apply** to save the changes.
- 7. Click **OK** to close **Properties** dialog box. The REST facet is added and appears in the **WebReport** project.



You will notice **WebReportService** REST service created under the **Defined Services** node. Whenever you create a **REST** project, a default REST service is created.



# 4.4 Working with the REST Editor – Mapping ABL operations with HTTP verbs

This section shows you how to map the HTTP verbs with ABL operations.

- 1. Select the Defined Services node under WebReport project in Project Explorer view.
- 2. Double-click the **WebReportService** REST. The **REST Resource URI Editor** dialog box opens.



량 WebReportService 업 양 REST Resource URI Editor						
ervice relative URI:	/WebReportServ	e	Edit			
lesources	+ ×	Verb Association				

3. Click Add Resource in the Resources category. The New REST Resource dialog box appears.

As mentioned before, all REST operations work on top of a resource, so we are here going to create a resource using which we can operate the service.

4. Enter /**customer** for the resource in **Resource URI**. The REST resources you add are identified with Uniform Resource Identifier (URI). The name must start with "/".

🔋 New REST I	Resource
Add a new F	REST Resource
Allows you t	to add a REST resource, query parameter, and path parameter.
Resource URI:	/customer
?	OK Cancel

5. Click **OK.** The **customer** resource is added.

The Verb Association table shows the available HTTP verbs.



<table-of-contents> *WebReportService</table-of-contents>	e 🛛		
📲 REST Resour	ce URI Editor		
Service relative URI:	/WebReportServic	2	Edit
Resources	🖶 🗙	Verb Association	
/customer		Verb='GET'	🗶
		Verb='PUT'	💥
		Verb='POST'	
		Verb='DELETE'	💥

6. Select next to Verb='GET' to associate GET verb to the GetCustomer internal procedure to execute the ABL logic when GET request is processed.

The **Associate Operation With Verb** dialog box appears and lists all the available ABL files in the current project.

7. Select **custom.p** resource and **GetCustomer** ABL routine. The **custom.p** file has all the CRUD operations/methods defined and we are selecting **GetCustomer** ABL routine which is a read operation to retrieve the records.

Associate Operation With Verb	- • • ×
Associate Operation With Verb Select an ABL routine to associate with th	e verb.
Resources: type filter text	ABL routines: type filter text
<ul> <li>▲ ➢ AppServer</li> <li>P custom.p</li> <li>▲ ➢ SpeedScriptGen</li> <li>▲ ➢ WebSpeed</li> <li>✓ CustomerEntry.w</li> </ul>	CreateCustomer (IN-OUT table) Contract Customer (integer, OUT table) Contract Customer (integer, OUT table) Contract Customer (IN-OUT table, in Contract Customer (IN-OUT tab
0	OK Cancel

- 8. Click **OK** to complete the association.
- Note: Open custom.p file and notice that the REST annotations were automatically added at the file level and to the **GetCustomer** routine that was selected. Close to proceed to REST Resource URI Editor.



We will fetch records based on the Customer id. So we now need to map customer id as an input parameter to an URI parameter as query strings.

9. Double-click the **WebReportService** tab to maximize and view it in full screen. The **Mapping Definitions** shows **Input** and **Output** tabs with the available request mappers. The input parameter **custId** that was defined in the procedure will be listed in the **Parameter** section.

📽 REST Resource UR	ll Editor		
Service relative URI: /WebF	ReportService		Edit
Resources	÷ X	Verb Association	
/customer		Verb='GET'         customGetCustomer           Verb='PUT'         Verb='POST'           Verb='DELETE'         Managing Definitions	
		Input Output           Request           * URL Parameters           **C Complete URL           **C Query String Parameters           * TIT Message           **C Method           -**C Headers           **C Concists           * Server Contexts           **C Serviet Request           **C Serviet Context           **C Serviet Context           **C Serviet Context	Para Interface Parameters (java:Integer) custld *t≛-

- 10. Place the mouse cursor on **Query String Parameters** and drag and drop the control to **custId**. The **Connection participants** dialog box appears.
- 11. Click OK.

Connection participants	×
Connection participants	
Enter Source And Target participant names(expressions) of the connection	
Source	
Type: Query String Parameter	-
Query String Parameter:	
custId	
Expression value:	
\${rest.queryparam['custId']}	*
(?) OK Canc	el

The input parameter mapping is complete and **custId** query string parameter is added and linked to the input parameter as displayed:



Bernet
Request
<ul> <li>URL Parameters</li> </ul>
- "T" Complete URL
*** Query String Parameters
└──œ custId
<ul> <li>HTTP Message</li> </ul>
- "T" Method
"t" Headers
[Drop a parameter here]
- "t" Cookies
<ul> <li>Server Contexts</li> </ul>
- "" Servlet Request
- "T" Servlet Response
- "" Servlet Context
- "T" Servlet Config
- "t" Servlet Response - "t" Servlet Context - "t" Servlet Config

### 12. Select the **Output** tab.

	Parameters	R
Interface Parameters	5	HTTP Message
- 📲 ttCustomer	{java:Object}	Response Code
		Servlet Response
		Headers **
		[Drop a parameter here]
		Cookies **
		Body ***
		[Drop a parameter here]

13. Select ttCustomer, drag and drop to Body in the Response section.



## 4.5 Publish and invoke the REST service

We have completed creating the REST service and mapping the parameters with REST verbs.

This section shows you how to publish the REST service to OE Web Server and invoke the REST service from the REST client.

1. Select the **Servers** view. The **restmgr1 OE Web Server** which is the default Web Server for REST applications appears in the **Servers** view.



Note: Progress Developer Studio for OpenEdge now ships Tomcat which is the default web server for REST and Mobile applications.

- 2. We now have to associate the REST service to the OE Web Server. Right-click the **restmgr1 OE Web Server** and select **Add and Remove** from the context menu.
- 3. Add the WebReportService using Add and Remove... to the restmgr1 OE Web Server.



4. Select **restmgr1 OE Web Server** and click **Start the server O**.

The server is started and the status changes to Started, Synchronized and WebReportService is published to the server:

restmgr1 OE Web Server 11.3 at nbhydsyellava [Started, Synchronized] WebReportService --> WebReport [Synchronized] [Published]

5. Open the WADL file to know the base URI for the REST service. In the Firefox browser, open the URL: http://localhost:8980/WebReportService/rest/WebReportService?\_wadl. The WADL displays Base URI, operations, and parameter mapping information.

Firefox •				
Http://localhost:898ReportService?_wadl				
🗲 🕙 localhost:8980/WebReportService/rest/WebReportSe 🏫 🔻 🍘	8 ▼ Google	₽ 🖬 -	+	<b>^</b>
Most Visited Getting Started Suggested Sites Web Slice Gal	llery			
This XML file does not appear to have any style information asso below.	ciated with it. The	document tr	ee is sh	own
- <application> <grammars></grammars> - <resources base="http://localhost:8980/WebReportService/&lt;br&gt;- &lt;resource path=" customer"=""> - <resource path="/customer"> - <resource path="/customer"> - <resource path="/customer"> - <reequest> - <request> </request>  </reequest></resource> </resource></resource></resources> </application>	rest/WebReportSe type="xs:string"/>	:rvice">		



Note: Web Application Description Language (WADL) provides complete information on the REST service in terms of resources and operations. This is similar to Web Service Description Language (WSDL) in Web Services.

6. In a new tab in Firefox, open the chrome://restclient/content/restclient.html URL. The **REST Client** opens.



Http://localhost:89eportService?_wadl × BRESTClient	x +
♦ ⇒ ⊘ chrome://restclient/content/restclient.html	מ על איז פאר פאר איז איז איז פאר איז
File - Authentication - Headers - View -	Favorite Requests ~ Setting ~ RESTClient
[-] Request	
Method GET VIRL http://www.example.com	★ ♥ SEND
Body	
Request Body	
Home   Github   Issues   Donate	Back to top

**Note:** We have already installed REST client in the system provided. In real-time, you need to install a generic REST client to test the service deployed.

7. Copy the **Base URI** <u>http://localhost:8980/WebReportService/rest/WebReportService</u> from the WADL and paste it in **URL** field in the REST client. Append /customer?custId=1 which is the resource that we are accessing and the parameter that we are passing as a query string:



8. Click **SEND**. Select the **Response Body** (**Highlight**) tab. The **GetCustomer** routine gets executed in the AppServer and the results corresponding to the Customer with id=1 is displayed in the client.

Response Hea	ders	Respon	se Body (Raw)	Response Body (Highlight)	Response Body (Preview)
1. (					
2.	"tto	ust	omer":		
3.	[				
4.		{			
5.			"CustNum"	: 1,	
6.			"Name": "	Andrews",	
7.			"Address"	: "10 Smith St,	Bedford",
8.			"Phone":	"617 555-1111"	
9.		}			
10.	]				
11. }					

## 4.6 Working with other HTTP verbs (Take-Home)

This section is a take-home section for you to work on other VERBS at your leisure. It is not mandatory to perform this section in this workshop.



1. Follow steps from above section to add the PUT, POST, DELETE verbs with ABL routines as shown in the screens below.

Verb Association		
Verb='GET'	customGetCustomer	🗙
Verb='PUT'	customUpdateCustomer	🗙
Verb='POST'	customCreateCustomer	💌
Verb='DELETE'	customDeleteCustomer	🗙

2. For PUT verb Input mapping, map Query String Parameter to custId and Body to ttCustomer.

erb='GET'	customGetCustomer			
erb='PUT'	customUpdateCustomer			
erb='POST'	customCreateCustomer			
erb='DELETE'	customDeleteCustomer			
oping Definitio	ons			
put Output				
	Request			Param
URL Pa	arameters			Interface Parameters
- "t" Cor	mplete URL		//////////////////////////////////////	ttCustomer 🔩 –
E-"" Que	ery String Parameters	/	{java:Integer}	custId 📆 🗆
	custId			
HTTP	Message			
- "" Me	thod			
*tt Hea	IDron a narameter here			
- "T. Coo	okies			
- "t" For	m Parameters			
E-"" Boo	iy			
L 🖡	[Drop a parameter here]			
<ul> <li>Server</li> </ul>	Contexts			
- "T" Sen	vlet Request			
- 📬 Sen	vlet Response			
- "T" Sen	vlet Context			
- "C Sen	vlet Config			

- 3. For **PUT** verb **Output** mapping, map **ttCustomer** to **Body**.
- 4. For **POST** verb **Input** mapping, map **Body** to **ttCustomer**.



erb='GET'	customGetCustomer			
erb='PUT'	customUpdateCustomer			
/erb='POST'	customCreateCustomer			
erb='DELETE'	customDeleteCustomer			
pping Definitio	ons			
put Output				
	Request			Paramete
<ul> <li>URL Pa</li> </ul>	rameters			Interface Parameters 🛛 🔻
- "t" Cor	nplete URL	_	{java:Object}	ttCustomer 🐚
- "t" Que	ery String Parameters	/		)
HTTP I	Message			
- "t" Met	hod			
E-"T Hea	ders			
	[Drop a parameter here]			
- "":" Coo	okies			
- "t" For	m Parameters			
E-*** Bod	ly .			
L[]	[Drop a parameter here]			
Server	Contexts			
- "T" Serv	/let Request			
- "T" Serv	/let Response			
- "T" Serv	/let Context			
- "T" Sen	/let Config			

- 5. For **POST** verb **Output** mapping, map **ttCustomer** to **Body**
- 6. For **DELETE** verb Input mapping, map **Query Stiring Parameter** to custId.

erb='GET'	customGetCustomer	
erb='PUT'	customUpdateCustomer	
/erb='POST'	customCreateCustomer	
/erb='DELETE'	customDeleteCustomer	
pping Definit	ions	
put Output		
	Request	Parame
URL P	arameters	Interface Parameters
- "T: Co	mplete URL	{iava:Integer} custid
"T Ou	ery String Parameters	
	custId	
▼ HTTP	Message	
- "T" Me	ethod	
⊡-"t" He	aders	
	[Drop a parameter here	
- """ Co	okies	
E-"t" Bo	dy	
LOB	[Drop a parameter here	
1.1.1	r Contexts	
<ul> <li>Server</li> </ul>	rylet Request	
Server		
▼ Server - """ Ser - """ Ser	vlet Response	
▼ Server - "t" Ser - "t" Ser - "t" Ser	rvlet Response rvlet Context	

7. For **DELETE** verb **Output** mapping, map **ttCustomer** to **Body**.

Click File→Save All to save all changes.

# 4.7 Invoke the REST service for remaining HTTP verbs (Take-home)



You need to complete the section 4.6 prior to working on this lab.

1. For POST verb request, the data is sent via JSON format. We need to add two custom headers to accept the JSON content. From **Headers** in the REST client, select **Custom Header**.

Headers - View -
Custom Header
Clear Favorites
Cical ravolites

The **Request Header** dialog box appears.

2. Enter **Content-Type** as header name and **application/json** as value and click **Okay**.

Request Header	×
Name	
Content-Type	
Value	
application/json	
Save to favorite	Okay Cancel

3. Similarly, repeat the above steps to add Accept as another Custom header.

Request Header	
Name	
Accept	
Value	
application/json	

The Headers section displays the Content-Type and Accept headers.

Headers			
Content-Type: application/json	×	Accept: application/json	×

4. Select the **POST** METHOD and provide <u>http://localhost:8980/WebReportService/rest/WebReportService/customer</u> URL. Copy and paste this JSON code in **Body** of the REST client:



{ "ttC	ustomer":
[	
	"CustNum": 100, "Nama": "IbanS"
	"Address": "12 Smith St, Bedford", "Phone": "617 555-1112"
}	
}	

thod POST V URL http://localhost:8980/WebReportService/rest/WebReportService/customer	★ ♥ SEND
eaders	🛗 Remove All
ontent-Type: application/json × Accept: application/json ×	
{ "ttCustomer": [ {	E

- 5. We are sending a record in JSON format which will be inserted into database. Click SEND.
- 6. It executes the **CreateCustomer** routine mapped for the **POST** method. The customer record is created. As we mapped the **POST** method **OUTPUT ttCustomer** to **Response**, the same JSON record is displayed in the REST client.

Response H	leaders	Respor	nse Body (Raw)	Response Body (High	light)	Response Body (Preview)
1. {						
2.	"ti	tCust	omer":			
3.	[					
4.		{				
5.			"CustNum"	100,		
6.			"Name": '	'JhonS",		
7.			"Address"	': "12 Smith	St, 1	Bedford",
8.			"Phone":	"617 555-111	2"	
9.		}				
10.	]					
11. }						

7. Use the **ABL Scratchpad** editor to check if the new record is created in the customer table. Execute the following code:



The result displays as:

0E Progress			
Customer ID	Customer Name	Address	Phone
100	JhonS	12 Smith St, Bedford	617 555-1112

8. Press **SPACE BAR** to close the **Run** window.



# 5 LAB 05: Application development using Progress Developer Studio for OpenEdge – GUI for .NET

### 5.1 Overview

The sections of this lab show how to design, develop, and run the ABL application using the GUI for .NET or OpenEdge Visual Designer as it is commonly called. With GUI for .NET, you can develop powerful graphical user interfaces (GUIs) for ABL applications. Visual Designer users can take advantage of the rich functionality and the look and feel of .NET controls without using any non-ABL language or leaving the Progress Developer Studio for OpenEdge environment.

In this lab, you will design a Sales report application to view Sales report in a given period. You will also design and use custom controls such as User controls and Inherited controls as part of the flow. In the process, you will understand many features that the editor provides to make development easy and productive.

Here is what you will develop in this lab.

	:			- • ×
Outlets Region	ns Items Ord	er By Date		
_				
From:	26-08-2013	To:	26-08-2013	<b>•</b>
		Get Outlet Data		
	Drag a colum	n header here <mark>to</mark> group	by that column.	
	OutletName	NoOfOrders		<b>^</b>
	Atlantic Ave	0		
	Brook St	0		Ξ
	Downtown	0		
	Frost Dr	0		
	Glen Av	0		
	Ice Ln	0		
	New St	0		
	New St Oak Park	0		
	New St Oak Park Peach Rd	0 0 0		<b>*</b>
	New St Oak Park Peach Rd	0 0 0		T
	New St Oak Park Peach Rd	0 0 0		T

### 5.2 Prerequisite

Complete Lab 01: Configuring your workspace and customizing the project prior to working on this lab.



### 5.3 Creating Custom Project

This section shows you how to create a custom project.

- 1. From the workbench menu, select **Window→Open Perspective→OpenEdge**. The **OpenEdge Editor** perspective opens.
- 2. From the workbench menu, select **OpenEdge→Tools→Customization Editor**. The **Customization Editor** opens.
- 3. The **OpenEdge Custom Projects** node allows defining custom projects. Select the **OpenEdge Custom Projects** node and click **Add**.
- 4. Enter VDCustomPrj in Name, OpenEdge Visual Designer in Perspective, GUI for .NET in Required facet.

OpenEdge Customization Options One or more changes have o to reflect the changes.     Available Extension Points     Available Extension Points     Menu / Toolbar Entries     Editor Context Menu     Menu / Toolbar Entries     Editor Context Menu     Menu / Toolbar Entries     Editor Context Menu     Editor Context Menu     Editor Templates     Editor Templates     Description: OpenEdge project with user-def	- 8
Template Overrides OpenEdge Custom Projects VDCustomPrj E Folder Layout Code-generation Templates OpenEdge visual Designer Required facet*: GUI for .NET OpenEdge runtime*:  GUI © TTY Values marked with an asterisk (*) are required.	Browse fined prope
Version*: 1.0.0	

- 5. Select Folder Layout node under the custom project VDCustomPrj. The Folder Layout and PROPATH sections are displayed on the right pane of the Customization Editor.
- 6. Click Add in the Folder Layout section. This allows creating a file or folder using the Add File/Folder dialog box.
- 7. Select File option button. Click Browse... next to Path.



8. Browse for path C:\OpenEdge\WRK\PDSOEWorkshopFiles\VDCustomPrj\_Files and select the MyUserControl.cls file.



P Add File/Fo	lder
Add File/Fol Specify the fil	der Epath.
🔘 Folder 🏾 🔘	File
Folder	
Folder nam	e:
	Add to PROPATH
🗸 Copy co	ntent from existing folder
Path:	C:\OpenEdge\WRK\PDSOEWorkshopFiles\VDCustomPrj_Files Browse
	Include sub-folders
File	
Path:	C:\OpenEdge\WRK\PDSOEWorkshopFiles\VDCustomPrj_Files\MyUserControl.cls
	Add to PROPATH
?	OK Cancel
~	

9. Click **OK**. The file is added to the Folder Layout. Add all files under the **VDCustomPrj\_Files** folder to **Folder Layout** section.

OpenEdge Customization Options	One or more	changes have o to reflect the changes. 🕅 ன 📾 🕻	er 🔈
Available Extension Points          Menu / Toolbar Entries         Editor Context Menu         New Jet Templates         Editor Templates         Template Overrides         OpenEdge Custom Projects         YDCustomPrj         Eolder Layout         Ode-generation Templates	Add Remove	Folder Layout         Changes made to the folder layo default and cannot be un	done.
Plugin Details     Version*: 1.0.0 Main			

10. Save changes and click Refresh Customization Options toolbar option to apply the changes.





11. Once a new custom project is created, workbench needs to be restarted to list the created project type in OpenEdge Project Creation window. Click **Yes** in the **Restart Required** dialog box.

📳 Resta	art Required	<b>×</b>
?	Do you want to restart Developer Studio now?	
		Yes No

The workbench is restarted and Workspace Launcher dialog appears.

- 12. Click **OK** to open workbench.
- 13. From the workbench menu, select **File**→**New**→**OpenEdge Project.** The **New OpenEdge Project** dialog box appears.
- 14. Specify VDReports in Project name.
- 15. Observe custom project is created **VDCustomPrj** and listed in **Project type configuration**. Select **VDCustomPrj** from the **Project type configuration** drop-down list.

Para New OpenEdge Project	
Create an OpenEdge Project	
Enter a name for the project.	
Project name: VDReports	
Use default location	
Location: C:\OpenEdge\WRK\Exchange_Demo\VDReports	B <u>r</u> owse
Project type configuration	
VDCustomPrj	•
OpenEdge project with user-defined properties.	

- 16. Click Next. The Select AVM and layout options dialog box appears.
- 17. Click Next. The Review VDCustomPrj project layout dialog box appears. Expand Project Root and observe the folder layout added is listed.



- 18. Click **Finish**. Select **Yes To All** in the **Overwrite** dialog box. The GUI for .NET project by default creates **assemblies.xml** file and as part of custom project folder layout, we are adding this file as the default and therefore you need to overwrite the files.
- 19. As this project is associated with the **OpenEdge Visual Designer** perspective, the **Open Associated Perspective** dialog box appears.



**Note:** The projects are associated with a specific perspective. You should open the views as suggested. Views are arranged to make it easy for development. **Project Explorer-> New** shows the file options that are related to this project development so that you can find options easily.



20. Select the **Remember my decision** check box and click **Yes**. The **Visual Designer** perspective opens, the VDReports project is created, and the folder is available in the **Project Explorer** view with all the files.



21. By default, an ABL form is created under the project. Right-click the **Form1.cls** file and select **Open With→OpenEdge Visual Designer**.



E Form1	مام				: : :
RunFo		New	÷.	[``	
		Open	F3		Order Entry
		Open With	+	2	OpenEdge ABL Editor
		Copy	Ctrl+C	<b>8</b> 3	OpenEdge Visual Designer Text Editor

The form opens in the design view. The **Visual Designer** perspective has a wide area for designing form (editor area) where toolbox is on the right side of the screen. The **Project Explorer**, **Properties**, and **Outline** views are displayed. The design workbench menu option is available where options related to forms designing are available.

P OpenEdge Visual Designer - VDRepo	orts/Form1.cls - Progress Developer Studio		
<u>Ele Edit Navigate Search Project E</u>	<u> 3</u> un <u>O</u> penEdge <u>D</u> esign <u>W</u> indow <u>H</u> elp		
] 📑 🖛 🗟 🛆 ] 🕲 🅸 🕶 O 🖛 💁 🖛	•   ৵ ▼   ⊔ ∡ ◙   ▦  ⋿ ☵ ♣ ៚ ⁵ч ఊ 옷 ↔	2 3 ] 2 × 2 × + ↓ ×	🗈 🖏 OpenEdge 🏷 OpenEdge »
🖒 Project Explorer 🛛 📄 😵 🖓 🗖 🗖	🏷 bGetCust.w (AppBu 🏷 *OrderApp.w (AppB	S Form1.cls (Design) 🕮 🎽	Properties OpenEdge Visual Designer perspectr
CrderApp		Toolbox 7	Form1 : Progress.Windows.Form
Referenced Assembles	Form1	Microsoft Controls	Properties Events
JL Procedure Libraries		OpenEdge Controls	ControlBox True
assembles.xml		OpenEdge Ultra Controls	Cursor Default
Form1.ds			DoubleBuffered False
RunForm1.p	D . C		Enabled True
	Design Canvas		E Font Microsoft Sans Serif, 7.8
	P		ForeColor ControlText
			FormBorderStyle Sizable
			HelpButton False
			El Icon Control
			Internode NoControl
			KevPreview False
			Language (Default)
	b		Localizable False
			E Location 0,0
			Locked False
			MainMenuStrip (none)
			Feet         The text associated with the control.         B: Outline S3         Image: S3         Image: S3         Image: S3         Image: S3         Image: S3         Image: S4         Image: S
J 📭 🖪	Writable		

**Note:** The **Design** menu option is visible only when the Visual Designer files are opened in editor. The options are specific to design view of the files. When any other file is opened in editor, this menu option is not visible. Observe this by selecting the OrderApp file.



**Note:** The major components of the Visual Designer are the Toolbox, the Design Canvas, and the Properties views.

#### Toolbox

The Toolbox is a list of controls that are available for use in the UI. You can select controls that you want to place on the form from this list. You can customize the set of controls available to the current project, and arrange them in logical groups that you define.

### **Design Canvas**

The Design Canvas is the editing area in which you model the appearance of the UI. It provides an accurate visualization of how the window appears to the user at run time, with the exception that it does not display the actual data. You use the mouse to size objects by dragging their borders, and to position controls on the form.



### **Properties view**

While a Visual Designer editing window has focus, the Properties view includes both Properties tab and Events tab, where you can view and edit the visual and behavioral characteristics of the object or objects that are selected on the Design Canvas.

### 5.4 Assigning a database connection to a project

This section shows you how to assign database connection to the VDReports project.

- 1. Select the **VDReports** project in the **Project Explorer** view.
- 2. Right-click VDReports and select Properties. The Properties dialog box appears.
- 3. Select **Progress OpenEdge→Database Connections**. The defined database connection for the workspace is displayed.
- 4. Select the check box next to **Exchange\_db**.

Properties for VDReports				Testing
type filter text	Database Connectio	ns		
Resource Builders Progress OpenEdge	Show selected OS	Show <u>a</u> ll		
Assemblies	Connection Name	Group	Physical Name	
Build	🔽 🌮 Exchange_db		C:\OpenEdge\WRK\ExchangeDB\w	orkshop.db
Custom				
Database Connections				

- 5. Click **OK**. The **Properties** dialog box closes and the connection is made by the AVM associated with project **VDReports** to the **workshop.db** database using the database connection profile. The splash screen displays briefly as the AVM is started.
- 6. Select Show View as a Fast View at bottom left corner of PDS OE to open the Console view.



7. Select **Other**. The **Show view** dialog box appears. The textbox displays **type filter text**. Enter the filter text as **Console**.

P Show View	
console General Console	<u></u>
ОК	Cancel



8. Select **Console** and click **OK**. This opens the **Console** view in the **Visual designer** perspective.

The **Console** view shows messages that the runtime was started and that the connection to the Exchange\_db database succeeded.

E Console X
ABL Console
OpenEdge AVM stopped for 'VDReports'. OpenEdge AVM started successfully for 'VDReports'. Project 'VDReports' runtime connection to 'Exchange_db' : OK

## 5.5 Creating and designing a form

This section shows you how to create a form, add controls, and design a form.

- 1. Select project VDReports in the Project Explorer view. Right-click and select New→ABL Form. The New ABL Form wizard for form creation appears.
- 2. Enter SalesReport in Form name.



P New ABL Fo	rm	- • ×
Create a For	m Class	
Enter a name	for the form. Do not use spaces or special characters.	
Package root:	\VDReports	Browse
Package:		Browse
Form name:	SalesReport	
Modifiers:	Final Abstract Widget pool	
Inherits:	Progress.Windows.Form	Browse
Implements:		Add
		Remove
Specify the co	de elements to generate:	
	Generate default constructor 🛛 🗹 Generate destructor	
	Generate super class constructors	
	Add routine-level error handling	
Specify the ret	urn value for generated methods:	
	Throw a Not Implemented exception	
	Return a default value	
Description:		
_		<b>T</b>
Purpose:		<u>_</u>
?	<u> </u>	Cancel



Note: The New Form wizard provides many options.

Package root: Specifies a currently open project to contain the class code and other project code. Click Browse if you want to select a project other than the current one (the default value).

Inherits: Optionally specifies another class in the current project as a super class from which the new class inherits state and behavior.

Implements: Optionally specifies one or more interfaces in the current project that the class implements. Click Add and select the desired interfaces in the Interface Selection dialog box. Use the **Remove** button to remove an interface from the list after adding it.

3. Click Finish. The New ABL Form wizard creates a blank form. The form opens in the Design View in editor and the respective SalesReport.cls file is created under the **VDReports** project.





4. Expand the OpenEdge Ultra Controls category in Toolbox and select UltraPanel control.



**Tip:** If you place cursor over the control in ToolBox, a brief help on the control class name, version, and usage of control is displayed. OpenEdge Ultra Controls are not shipped with Progress Developer Studio for OpenEdge and are sold as a separate product. You can buy it directly from Infragistics.

5. Drag and drop the **UltraPanel** control on the **SalesReport** form. The **UltraPanel** control is added to form. Select **SmartTag**, the small right-arrow as shown in screen:





**Note:** UltraPanel is a control which can contain other controls and provide automatic scrolling or sizing. This control provides almost the same functionality as the .NET Panel control in the Microsoft controls. In addition, it also allows advanced appearance customization of the scroll bars and UltraPanel itself, including the ability to use application styling.

Tip: SmartTags offer a subset of the most frequently used properties for the control. If SmartTags are available, the control has a small right-arrow button at the top right corner when the control is selected.

- 6. Select the **Dock in Parent container** link. Observe that the panel expands and adjusts in the space of the form.
- 7. Double click **UltraTabControl** in **ToolBox**. The **UltraTabControl** is added to **SalesReport** form in default area. **UltraTabControl** gives you the ability to add tabs to your window.

Ultra	TabControl	oControl oStripControl
Infra Versi	gistics.Win.UltraWinTabControl.UltraTabControl ion:13.1.20131.2015	
Prov appli	ides a means of segregating the controls which per cation via the familiar tab metaphor.	rtain to distinct aspects of a
	SalesReport 📃 🔳	×
	[Tab header area]	
	[Shared controls page]	
		Þ

**Tip:** If you double-click a control in ToolBox, it adds the control to the default area in the default size. If control needs to be added to a specific area with a specific size, hold down the primary mouse button, and drag the control to the desired position on the form.

8. Open **SmartTag** of **UltraTab** in the form and select **Fill** in the **Dock** property. Observe that Tab control is filled in form.



SalesReport		
[Tab header area]	Ultra labControl lask	S
	Name	ultraTabControl1
[Shared controls page]	Add Tab	
	Edit Tabs	
	Show Shared Controls F	oage
	Appearance	
	Tab Orientation	Default 💌
	Text Orientation	Default 💌
	View Style	Default
	Behavior	
	Close Button Location	Default <
	Style	Default
	Layout	
	Dock	None
		None

- 9. Select the title area of the **SalesReport** window in the Design View. The **Properties** view shows the properties of the selected form.
- 10. Enter 655, 525 in the Size property and press Enter to apply changes.

D	roportion Even	te	•	Properties
-	Localizable	False		
$\triangleright$	Location	0,0		
	Locked	False		
	MainMenuStrip	(none)		
	MaximizeBox	True		
$\triangleright$	MaximumSize	0,0		
	MinimizeBox	True		
$\triangleright$	MinimumSize	0, 0		
	Opacity	100%		
$\triangleright$	Padding	0, 0, 0, 0		
	PreviousState	Normal		
	RightToLeft	No		
	RightToLeftLayou	False		
	Showlcon	True		
	ShowInTaskbar	True		
$\triangleright$	Size	655, 525		Size Propert
	SizeGripStyle	Auto		

**Caution:** Once property value is modified in the **Properties** tab, the changes are viewed only when the cursor is placed outside the property value field. Press enter or click anywhere in the **Properties** view or **Design Canvas** to view the changes.



Observe that the form size has also increased. Also, sizes of UltraTab and UltraPanel are increased as they are docked to the form.

- 11. Click **File** $\rightarrow$ **Save** to save the changes.
- 12. Select Tab header area. The SmartTag of UltraTab appears.
- 13. Click SmartTag of UltraTab. Observe that the Add Tab link is available.

∎	UltraTabControl Tasks		
	Name	ultraTabControl1	
	Add Tab		
	Edit T Add a tab to the control's Tabs collect		n.

14. Click the Add Tab link. Tab1 is added. Similarly, add 3 more tabs.

SalesReport	
tab1 tab2 tab3 tab4	

15. Select **ultraTabPageControl1** in the **Outline** view. The **Property** view shows the properties of ultraTabPageControl1.

	🗄 Outline 🛛			
🖂 🏶 SalesReport - Progress.Windows.Form				
🗄 🗐 ultraPanel1 - Infragistics. Win. Misc. UltraPanel				
	🗄 🎆 ultraPanel 1. ClientArea - Infragistics. Win. Misc. UltraPanel ClientArea Un safe			
🖻 🚞 ultraTabControl1 - Infragistics.Win.UltraWinTabControl.UltraTabControl				
🎲 ultraTabSharedControlsPage1 - Infragistics.Win.UltraWinTabControl.UltraTabSharedControls				
	👹 ultraTabPageControl1 - Infragistics.Win.UltraWinTabControl.UltraTabPageControl			
	🎲 ultraTabPageControl2 - Infragistics.Win.UltraWinTabControl.UltraTabPageControl			
	🎲 ultraTabPageControl3 - Infragistics.Win.UltraWinTabControl.UltraTabPageControl			
	📽 ultraTabPageControl4 - Infragistics.Win.UltraWinTabControl.UltraTabPageControl			

**Tip:** The **Outline** view of a file that is open in the Visual Designer is called as **Document Outline** view, it shows the contents of the Design Canvas and provides an alternative means of selecting, copying, deleting, renaming, and re-parenting controls.

The **Document Outline** view shows a collapsible-expandable hierarchy of the top-level container (form or user control) and all of its controls in a tree structure. Controls appear in the **Document Outline** view in the order in which you place them on the Design Canvas. When there are many controls added and its hard to select a control in Design View, control

selection can be easily done from the **Document Outline** view.

16. From the **Properties** view, expand the **Tab** property and notice that the **Text** Property is available. This property represents the caption text for the tab.





17. Enter value for the **Text** property as specified in table below and press **enter** after each value is entered in the **Properties** view. The tabs will now show the specified text.

Tab Page Control	Text Property Value	
ultraTabPageControl1	Outlets	
ultraTabPageControl2	Regions	
ultraTabPageControl3	Items	
ultraTabPageControl4	Order By Date	

18. Select the combo-box in the **Properties** view and select **ultraTabControl1**.

🔲 Properties 🛛	🏣 🋃 💀 🛃 🎽 🗖 🗖		
ultraTabControl1 : Infragistics.Win.UltraWinTabCc 🔻			
SalesReport : Progress.Windows.Form ultraPanel1 : Infragistics.Win.Misc.UltraPanel			
ultraTabControl1: Infragistics.Win.UltraWinTabContro			
ultraTabPageControl1: Infragistics.Win.UltraWinTabC			
ultraTabPageControl3 : Infragistics.Win.UltraWinTab			
ultraTabPageControl4: Infragistics.Win.UltraWinTabC			
ultra l abSharedControlsPage1 : Infragistics.Win.Ultra			
SpaceBeforeTabs			
Style	Default 🔹		

19. Select the following values for the properties of ultraTabControl1:

Property	Value			
View Style	Office 2007	ViewStyle	Office2007	•
Style	Office 2007 Ribbon	Style	Office2007Ribbon	•
-				

20. Observe that the look of the form has changed. Save the changes.


## 5.6 Creating Inherited controls (Take-home)

We have already provided the Inherited controls that you can use in this workshop. This section shows you how to create the same Inherited controls, you can try it at your leisure. When you are doing this section, delete already created Inherited buttons or add them with a different name.

In addition to the extensive set of commercially published UI controls that the Visual Designer provides, you can also create and reuse custom controls that extend or combine the existing controls. You can create two types of custom controls:

User controls: Composite controls made up of multiple individual controls grouped in a container.

**Inherited controls:** Controls that derive properties and events from a super class (that is, a parent control).

- 1. Select the **VDReports** project in the **Project Explorer** view. Right-click **VDReports** and select **New→ABL Inherited Control.** The **New ABL Inherited Control** wizard opens.
- 2. Specify **MyButton** in **Inherited Control name**. The following error message is displayed because Inherited Control inherits all properties from the super class control.

You must specify the parent control (super class) in the Inherits field.

3. To create Inherited Control for **UltraButton** control, click **Browse** next to **Inherits**. Enter **ultra** in **Type filter text**. All the classes starting with ultra are displayed.



**Note:** If you cannot find the class you are looking for, add the same control that you are looking for; in this case it is UltraButton to a dummy form and then try this again.

4. Select UltraButton from the list and click OK. Inherits now displays the Infragistics.Win.Misc.UltraButton class.

Inherits:	Infragistics.Win.Misc.UltraButton	

5. Click **Finish**. The **MyButton(Design).cls** file opens in the editor and it also appears under the **VDReports** project.



6. From the **Properties** view, modify the following **MyButton** properties and set values as shown:

Property	Value	Displays as	
Text	Get Data	Text	Get Data
Size	130, 30	Size	130, 30

- 7. Save the changes.
- 8. Similarly, create another ABL Inherited Control with the following details:

#### Inherited Control name: MyDateTimeEditor Inherits: Infragistics.Win.UltraWinEditors.UltraDateTimeEditor

UltraDateTimeEditor - Infragistics.Win.UltraWinEditors

9. From the **Properties** view, modify the following **MyDateTimeEditor** properties and set the values as shown:

Property	Value	Displays as	
ButtonStyle	Office2007RibbonButto	ButtonStyle	Office2007RibbonButte
	n		
DisplayStyl	Office2007	DisplayStyle	Office2007
e			
MinDate	01-01-2013	MinDate	01-01-2013

10. Save the changes. We have successfully created two Inherited controls which can be added to the Toolbox and used as regular controls.



# 5.7 Adding controls to Toolbox

This section shows you how to add control to Toolbox.

1. Right-click **Toolbox** and select **Add Control Group**.



2. Enter **Custom Controls** in the text box and press **Enter**.

OpenEdge Ultra Controls	
Custom Controls	

Control group Custom Controls is added to Toolbox.

Custom Controls	
No controls in this group	

- 3. Right-click **Custom Controls** and select **Add Controls...**. The **Add Controls** dialog box appears.
- 4. Click the **ABL Controls** tab. We have already provided you with the **Inherited controls** and the **User control**. They are listed as follows:



P Add Controls		×
Control Group: Custom Controls Filter:	BL Controls	Clear
Control Name	Namespace	Directory
MyButton	MyButton	C:/OpenEdge/WRK/Exchange_De
MyDateTimeEditor	MyDateTimeEditor	C:/OpenEdge/WRK/Exchange_De
MyUserControl	MyUserControl	C:/OpenEdge/WRK/Exchange_De
		OK Cancel

5. Select the check boxes corresponding to all the controls. Make sure that **Control Group** selected is **Custom Controls**.

P Add Controls			X
Control Group:	Custom Controls	•	
Filter:			Clear
		PL Controle	
Browsed Assemblie	es Global Assemblies A	BL Controis	
Control Nam	e 🔺	Namespace	Directory
MyButton		MyButton	C:/OpenEdge/WRK/Exchange_De
MyDateTime	Editor	MyDateTimeEditor	C:/OpenEdge/WRK/Exchange_De
MyUserControl		MyUserControl	C:/OpenEdge/WRK/Exchange_De

6. Click OK. The selected controls are listed in Toolbox under the Custom Controls category.

Custom Controls
la Pointer
NyButton
201 MyDateTimeEditor
👹 MyUserControl

7. Click **File**→**Close All** to close all the files. We have now successfully added custom controls to the Toolbox. They are now ready for use.



## 5.8 Creating User control (Take-home)

This section shows you how to create a User control so that you can combine the functionality of several controls into a single reusable unit. You will design a User control with the Inherited controls that you have created in the above section and add the User control to **Toolbox**.

- 1. Right-click the VDReports project in the Project Explorer view and select New→ABL User Control. The New ABL User Control wizard opens.
- 2. Specify **MyUserControl** in **User Control name**. By default, user control inherits **Progress.Windows.UserControl**.

🔋 New ABL User Co	ontrol	- • •	
Create a User Control Class			
Enter a name for t	he user control. Do not use spaces or special characters.		
Package reet	WDReports	Browco	
Package root:		Browse	
Package:		Browse	
User Control name:	MyUserControl		
Modifiers:	Final Abstract Widget pool		
Inherits:	Progress.Windows.UserControl	Browse	

3. Click Finish. The MyUserControl(Design).cls file opens in the Design View.

🆏 MyUserControl.cls (Design) 🛛		- 8
	Toolbox	Ą
	Custom Controls	
	Microsoft Controls	
	OpenEdge Controls	
	OpenEdge Ultra Controls	
0		
_		

4. Expand the **Custom Controls** category in **Toolbox.** Drag and drop **MyDateTimeEditor** control to the **User control**.



	-

- 5. Similarly, add the MyDateTimeEditor control.
- 6. Expand the Microsoft Control category in Toolbox and select the Label control.



- 7. Drag and drop the Label control twice.
- 8. Arrange the controls as shown in the image Below:

label1	 <ul> <li>label2</li> </ul>	 -

9. For both the Label controls, modify the Font property. Click 📖 in the Font property value. A list of other Font properties appears. Select ellipsis button 🗔 and select Font style as Bold.



10. Specify the **Text** property of the two labels as shown in the image and resize the user control:

s *MyUserContr	rol.cls (Design)	×	
From:		To:	 •

- 11. Save all the files.
- 12. Add MyUserControl User control to ToolBox under the Custom Controls group.

Toolbox	7
Custom Controls	
🔓 Pointer	
R MyButton	
MyDateTimeEditor	
MyUserControl	



## 5.9 Designing the SalesReport form to display outlets data

This section shows you how to design the SalesReport form with controls created in the previous sections. You will design the **Outlets** tab to display the outlets data in a list for the selected dates.

- 1. To define the schema, create an ABL Include file. Right-click the **VDReports** project and select **New→ABL Include**. The **New ABL Include** dialog box appears.
- 2. Enter TempTbl.i in File name and click Finish.

P New ABL I	include	
Create an A Specify a na	ABL include file ame for the include file.	
<u>C</u> ontainer:	\VDReports	Browse
<u>F</u> ile name:	TempTbl.i	
Description:		* *
<u>P</u> urpose:		A 
<u>A</u> uthor:		
?	Einish	Cancel

The Include file opens in the editor and the TempTbl.i file is created under the VDReports project.

3. Copy and paste the following temp-table definitions in the include file.

DEFINE TEMP-TABLE ttOrderInfo	
FIELD ttDateOfOrder LIKE ORDER.OrderDate	
FIELD ttNoOfOrders AS INTEGER.	
DEFINE TEMP-TABLE ttTotalTickets	
FIELD ttttDateOfOrder LIKE ORDER.OrderDate	
FIELD ttTotalOrders AS INTEGER	
TNDEX idxOrder IS UNTOUE ttttDateOfOrder	
DEETNE TEMP-TABLE ttTtemOrders	
ETELD <b>f</b> T+omNum ITKE TTEM TTEMNUM	
ETELD FITCHMUM LIKE ITEM ITEMNAME	
FIELD TNOUTOFIGERS AS INTEGER	
INDEX 10XITEM IS UNIQUE FITEMNAME.	
DEETNE TEND TADLE thansachidens	
DEFINE TEMP-TABLE TTAREAURGERS	
FIELD ttArea LIKE Outlet.Region	
FIELD ttNoOfOrders AS INTEGER.	
DEFINE TEMP-TABLE ttOutletOrders	
PRUGRE	:99

- 4. Save the file.
- 5. Double-click the SalesReport.cls file in **Project Explorer** view. The file opens in the **Design View** in the editor.
- 6. Add **MyUserControl** User control and the **MyButton** Inherited control to the **Outlets** tab. The button has default properties as defined.
- 7. Specify Get Outlet Data in the Text property of MyButton1. Place controls as shown in screen below:

SalesReport			
Outlets Regions	Items Order By Date		
From:	24-08-2013 🔹	To: 24-08-2013	•
	Get O	utlet Data	

8. Expand the **OpenEdge Controls** category in **ToolBox** and select the **ProBindingSource** control.

OpenEdge Controls	
<mark>≽ P</mark> ointer	
₩ ProBindingSource	

9. Drag and drop the **ProBindingSource** control onto the form. The **ProBindingSource Designer** dialog appears.



**Note:** ProBindingSource is a non-visual control whose properties define the schema for the data to be displayed by the accompanying visual control. The ProBindingSource control includes a designer tool that helps you define this schema. The tool also gives you the option of importing the schema from an XML schema (XSD) file, or from an ABL source file (like p, cls, w, i, and html).

10. Click **OK**. The controls appear in a separate area at the bottom of the Design View.

	bindingSource1
Ë),	

**Note:** UI controls fall in one of two basic categories:

**Visual controls** appear on the application's user interface and allow user interaction, display data, or both. Examples include buttons, combo boxes, and data grids.

**Non-visual controls** do not appear on the user interface at run time; instead, they support visual controls or provide other services such as logging. Non-visual controls typically hold data, formatting, or other information needed by one or more interactive controls. An example is the ProBindingSource, which serves as an intermediary between an actual ABL data source and the control, such as a grid, that displays the data.

11. Open the **SmartTag** of **bindingSource1** control and enter **OutletsPBS** in **Name** and click the **ProBindingSource Designer** link.

u <sup>™</sup> bindingSource1	BindingSource Tasks	
·	Name	OutletsPBS
	ProBind	lingSource Designer

12. In the **ProBindingSource Designer** dialog box, select **Import From File** on toolbar as shown below:

ProBindingSource Designer			
: 🗐	1   🗙   🕇 🖡   🕵 🔤		
Avai	Schema definition Import Schema from File		

- Select TempTbl.i from the location
   C:\OpenEdge\WRK\Exchange\_PDSOE\VDReports.
- 14. Select ttOutletOrders from the schema in left section of designer and click Add.



ProBindingSource Designer			
🗄 🛍 🔀 🛉 🕂 👘 🔛			
Available schema	Schema definit	ion of "Outlets"	
Schema of "TempTbl.i"	Tables	Fields	Table : ttOutletOrders
	d ttOutletOrders	<ul> <li>Outlet</li> <li>NoOfOrders</li> </ul>	Design         Name         Recursive table
🖻 🛄 ttTotalTickets 🛛 🖌			
			<u>O</u> K <u>C</u> ancel

#### 15. Click OK.

16. Expand the **OpenEdge Ultra Controls** category in **ToolBox**. Drag and drop the **UltraGrid** control onto the form. The **UltraWinGrid Quick Start** wizard appears.

UltraWinGrid Quick Start			
IlltroWinGrid Oui	ck Start		
	CK SLAFL		
Data Echoma	Drosote	Eastures	
Snarify the schema that will be	Select a preset look and/or	Select one or more major features	
used to generate bands and colu	behavior	select one of more major reactines	
Appearance Browser	Band and Column Properties	Finish	
Customize the look	Customize settings on specific	Confirm the changes you've made	
	bands and columns		
To specify the data schema the contr	ol should use to generate bands and co	lumps. I want to	
<select an="" option=""></select>			•
<select an="" option=""></select>			
Create an UltraDataSource, design its data	a schema and bind it to the control (recomm	ended if you don't have a DataSource to assign now)	
Manually design the data schema 'on the	fly' (requires setting a DataSource with a co-	npatible data schema at runtime in code)	
Bind the control to an existing DataSource	e now		
I don't want to specify the data schema n	ow - I'll just bind to a DataSource at runtim	(Note: preview grids will display a dummy schema)	
	,		
Preferences			< Previous Next > Finish

- 17. Select the **Bind the control to an existing DataSource now** option from the combo-box.
- 18. From the **DataSource** combo-box, select **OutletsPBS**.





19. Click **Finish**. The **UltraGrid** control is added to the **Outlets** tab. Modify the **UltraGrid** control size and place the control as shown in screen below:

Outlets	sReport Regions	Items Order By Date	
	From:	27-09-2013 <b>v To</b> : 27-09-2013 <b>v</b>	
		Get Outlet Data	
		Drag a column header here to group by that column.	
		Outlet Name Text 7 Text 7	
		Start! Click 'Start!' to begin designing the UltraGrid	

20. Save all the files.

We have now completed designing our Sales data screen. We have added a User control which contains two **DateTime** fields to pick the dates, added a **ProBindingSource** control to fetch the data from database, and configured **ProBindingSource** with **Outlet** and **No of Orders** fields. We have also added **UltraGrid** control which can show flat or hierarchical data and associated this control with the **ProBindingSource** to show the data.

We will now add an event to the **Button** control and then add logic to the event to retrieve the data.



# 5.10 Working with the source editor - Adding methods and events

This section shows you how to work with the source editor, add methods and events to the form, and use Editor Actions.

1. Open the source view of the form. Right-click the form and select View Source.

Save Close Visual Designer	Ctrl+S Ctrl+F4		
View Source	F9		

The source view of the **SalesReport.cls** file opens in the editor. The source and design view will always be in sync.

🆏 Myl	JserControl.cls (Design)	🆏 SalesRe	port.cls (Design)	🖸 Sales	Report.cls 🛛		- 8
1							
2	/*						
3	File	: SalesRepor	rt				
4	Purpose	:					
5	Syntax	:					
6	Description	:					
7	Author(s)	: syellava					Ξ
8	Created	: Fri Aug 2	3 09:07:35 IST	2013			
9	Notes	:					
10						*/	
11							
120	USING Progress.La	ang.*.					
14							
15							
16			_				
1/	CLASS SalesReport	E INHERIIS I	-orm:				
18				C	C		
19	DEFINE PRIVA	LE VARIABLE	components AS	System.	componentmodel.	IContainer NO-UN	4
20	DEFINE PRIVA	LE VARIABLE	myButtoni AS	MyButton	NO-UNDU.	NDO	
21	DEFINE PRIVA	LE VARIABLE	myUserControl	I AS MYU	sercontrol NU-U	NDU.	
22	DEFINE PRIVA	LE VARIABLE	UUTIETSPBS AS	Trogres	s.Data.Bindings	ource NU-UNDU.	
23	DEFINE PRIVA	IE VARIABLE	ultrauridi AS	ontragi	AS Infrogratics	Winoria.Ultraori	L
24	DEETNE DRIVA	LE VARIABLE	ultraladPageC	ontrol4	AS Intragistics	Win.Ultrawiniat	,
25	DEFINE PRIVA	TE VANIADLE	ultra a dPaget	ontrol3	AS Intragistics	Win.UltraWinlat	-
	•					4	

**Note:** The ABL Editor displays different syntax elements in different colors to make them easily recognizable at a glance. You can use the default color scheme or go to the ABL Editor Preferences page to create your own.

2. Right-click the editor and select Source  $\rightarrow$ Add Method. The Add Method dialog box appears.

**Note:** The Source context menu has options to add code using wizards. For example, Add method, properties, procedure, and function options open a dialog. When you provide name or related information, it generates the code in editor automatically.

3. Enter getOutletData in Method name.



P Add Method	
Generate method code Enter the name of the new method.	<b>(10)</b>
Method name: getOutletData Modifiers Public Protected Private Final Abstract Static	
Return type:       VOID         Extent	<ul> <li>▼ Browse</li> </ul>
?	<u>G</u> enerate Cancel

4. Click Generate. The getOutletData method is added to source.



5. After the method statement, type **DV** and press **Ctrl+SPACEBAR**. The content assist window opens and displays the macros available for Define variables. When you select an option, its expanded code is displayed in the next box.

METHOD	PUBLIC VOID getOutletData(	):	
DV	Ū ,		
RE	🗏 DVDT - Date variable		DEFINE VARIABLE AS INTEGER NO-UNDO.
END ME	🗏 DVDTZ - Datetime-tz variable		
	🗏 DVHN - Handle variable		
METHOD	🗏 DVI6 - Int64 variable		
DESTRU	DVIN - Integer variable		
		-	<b>X PROGRESS</b>
		2	TINGONLOG

6. Double-click **DVIN** from list. The define integer variable statement is added and cursor is placed at position where variable name needs to be added. Enter **cntr** in variable name.

DEFINE VARIABLE cntr AS INTEGER NO-UNDO.

Tip: Syntax-completion assistance proposes syntax to complete the code that you are typing in the ABL Editor. When you press **CTRL+SPACEBAR**, proposals appear in the left pane of a pop-up window, with reference information about the selected item in the right pane. To insert an element at the current cursor position, select it, and double-click or press ENTER.

By default, syntax-completion assistance filters proposals based on the context (for example, showing only relevant keywords). Press **CTRL+SPACEBAR** while the pop-up window is open to toggle between context-filtered proposals and all proposals. The assistant is also data aware, when you select a keyword; it lists all the database tables.

- 7. After the define statement, press Enter. Press CTRL+SPACEBAR. The content assist dialogbox opens.
- 8. Type **FOR**. The filtered list appears and displays the options starting with **for**. Similarly add the following code with the help of content assistant. Press **SPACEBAR** after each selected option. Press **CTRL+SPACEBAR** after providing each keyword.

FOR EACH Outlet BREAK BY Outlet.OutletName

Content A	Content Assist List					
DEFINE VARIABLE cntr AS I for	NTEGER NO-UNDO.	Options applicable to FOR are only displayed.				
<ul> <li>EACH - OPTION</li> <li>FIRST - OPTION</li> <li>LAST - OPTION</li> </ul>						
DEFINE VARIABLE cntr AS for EACH	All relevant tables from the connected database are listed. Select workshop.Outlet.					
DEFINE VARIABLE cntr AS IN for EACH Outlet BREAK BY	Image: Second system         Image: Second system	As the Outlet table is already selected, after BREAK BY, field proposals of the Outlet table are listed. Select workshop,Outlet.City.				

The content assistant displays the following after each keyword:

9. Add the following code to the method statement to get the outlet data:



```
METHOD PUBLIC VOID getOutletData( ):
        DEFINE VARIABLE cntr AS INTEGER.
        FOR EACH Outlet BREAK BY Outlet.OutletName:
            cntr = 0.
            FOR EACH Order WHERE DATE(Order.OrderDate) > frmDate AND
date(Order.OrderDate) < tDate:</pre>
                IF Order.Outlet = Outlet.OutletName THEN
                D0:
                    cntr = cntr + 1.
                END.
            END.
            /*
                               Create temp table with data*/
            CREATE ttOutletOrders.
            ASSIGN
                ttOutletOrders.Outlet
                                           = Outlet.OutletName
                ttOutletOrders.NoOfOrders = cntr.
        END.
END METHOD.
```

- 10. Save the file. Error markers ② are displayed for FOR EACH Order statement, for the SalesReport.cls file in the Project Explorer view and the editor. Error marker represents compilation errors in file.
- 11. Right-click the editor and select Check Syntax or press Ctrl+Shift+C.



**Tip**: The options also provide the short cut keys.

The Check Syntax dialog box appears with the following error message:



12. Select **OK** to close the **Check Syntax** message.

The **Outlets** tab design takes **From** and **To** dates as input and if you click **Get Outlet Data**, it displays the results of Outlets data between given dates. Add the following code for **From** and **To dates** to the method.

METHOD PUBLIC VOID getOutletData(frmDate AS DATE, tDate AS DATE):

13. Save the file.



- 14. Open the source view of the SalesReport.cls file in the editor.
- 15. Above the Constructor statement, add the following code to include {TempTbl.i} file in source.



16. Save the **SalesReport.cls** file and observe that the error markers are cleared. Click **Check Syntax** again and click **OK** at the message.



17. Right-click the editor and select View Design to open the Design View.



Double-click Get Outlet Data on the Design Canvas, the source view opens. The myButton1\_Click event method is automatically generated and the cursor is placed at the method.



**Tip**: The quickest way to define an event method is to subscribe to the event by double-clicking in the **Design Canvas** or the **Events** tab. When you do this, you are taken to the automatically generated event-handling method in the ABL code.



The double-click technique minimizes typing errors, but you can also use the **Properties** view to associate any method with an event. On the **Events** tab, click the value cell to the right of the Event name. The down-arrow button that appears lets you select from a drop-down list of all methods defined in the source code whose signature matches that of the selected event. Alternatively, you can type a method name in the cell.

You can change the name of the event handler method by editing the value on the Events tab. The change is reflected in the source code.

19. Copy and paste code shown in the **myButton1\_Click** method to get From and To dates from the design view, query ttOutletOrders table with those dates to get data, and assign that query to ProBindingSource control handle to hold query results in the ProBindingSource handle.

```
METHOD PRIVATE VOID myButton1_Click( INPUT sender AS
System.Object, INPUT e AS System.EventArgs ):
   DEFINE VARIABLE gh
                           AS HANDLE
                                        NO-UNDO.
   DEFINE VARIABLE frmDate AS DATE
                                        NO-UNDO.
   DEFINE VARIABLE tDate AS DATE
                                        NO-UNDO.
   DEFINE VARIABLE gryStr AS CHARACTER NO-UNDO.
        frmDate = myUserControl1:getFromDate().
        tDate = myUserControl1:getToDate().
        getOutletData(frmDate,tDate).
        qryStr = "FOR EACH ttOutletOrders".
        CREATE QUERY qh.
        qh:SET-BUFFERS(BUFFER ttOutletOrders:HANDLE).
        qh:QUERY-PREPARE(qryStr).
        qh:QUERY-OPEN.
        OutletsPBS:Handle= qh.
    END METHOD.
```

- 20. Press Ctrl+I to indent file code. Save the file.
- 21. Click **Check Syntax** or alternately hover over the error marker. The following error message is displayed.

 Image: Walking and the second secon

22. Press **CTRL+SPACEBAR** after **getFromDate**, there are no proposals listed representing the methods are not added to User Control.



- 23. Double-click **MyUserControl.cls** in the **Project Explorer** view to open the file in Design View.
- 24. Right-click and select View Source or alternatively press F9.
- 25. Press Alt+Shift+M and add two public methods getFromDate and getToDate with return type as DATETIME.



P Add Method							
Generate method code	m						
Enter the return type for the method.							
Method name: getFromDate							
Modifiers							
Public      Protected      Private     Final							
Abstract							
Static							
Return type: DATETIME	► Browse						

26. Add the following statements to get the value of the DateTime field from UI and return the same result variable:

METHOD PUBLIC DATETIME getFromDate( ): DEFINE VARIABLE result AS DATETIME NO-UNDO. result = myDateTimeEditor1:Value. RETURN result.	
END METHOD.	
<pre>METHOD PUBLIC DATETIME getToDate( ): DEFINE VARIABLE result AS DATETIME NO-UNDO. result = myDateTimeEditor2:Value. RETURN result. END METHOD.</pre>	

27. Save all the changes.

#### 28. Open the SalesReport.cls file.

29. Right-click and select **Progress OpenEdge→Compile**. The file is compiled and you see no error markers now.

**Tip**: By default, Eclipse compiles source files when they are added to the workspace or saved. You can disable the Build automatically option from the Eclipse Workspace preferences (Window > Preferences > General > Workspace).

You may choose to disable automatic builds if the contents of your workspace are frequently updated significantly (for example, by copying files from an external source), triggering build processes that block creating or saving resources until they finish. We recommend that you should enable the automatic builds to ensure that up-to-date r-code is always available for tools and features that require it.

If you do disable automatic builds, it is recommended that you have Progress Developer Studio for OpenEdge compile ABL source files when you save them. To do so, select the **Compile on save if required** option in the **Editor Build** preferences.

At any time, you can explicitly compile your current file by selecting **Compile** from either the Source menu or the ABL Editor context (right-click) menu



30. Press CTRL+SPACEBAR after getFromDate again and the proposals are listed now.

341 frmDate = myUserControl1:getFrom	Date().
getFromDate() - MyUserControl Returns DATETIME value.	GetFromDate() DATETIME - MyUserControl

- 31. Run the **SalesReport.cls** file.
- 32. Select **From** and **To** dates and then click on **Get Outlet Data** button. The sale per outlet data appears.

🖳 Sales	Report					
Outlets	Regions	s Items Orde	er By Date			
	From	00.00.0010		Tai	00.00.0010	
	TOIL.	26-08-2013	*	10.	26-08-2013	•
			Cot Out	lot Doto		
			Get Out			
		Drag a column	booder bere tr	arous bu	that column	
		Drag a column	i neader nere u	group by	that column.	
		OutletName	NoOfOrders			<b>^</b>
		Atlantic Ave	0			
		Brook St	0			=
		Downtown	0			
		Frost Dr	0			
		Glen Av	0			
		Ice Ln	0			
		New St	0			
		Oak Park	0			
		Peach Rd	0			<b>T</b>

33. Close the **Run** dialog box. We have now completed creating an application to view Sales data by date.



## 5.11 Working with OpenEdge Debugger (Optional)

This section shows you how to work with Debugger and use the Debugger to find logic errors in ABL applications by adding a breakpoint for a specific executable line in the ABL code.

The key to debugging is the ability to run a program and suspend execution at strategic points so that you can monitor and evaluate the results. To allow you to control the program flow in this manner, Debugger includes the following features:

**Breakpoints**: You can insert breakpoints on executable statements anywhere in your source code and in the include files that might appear in many different procedures. The Debugger suspends execution at each breakpoint.

**Code-stepping**: You can discretely execute the next statement, the next statement plus any subprocedure or trigger called by that statement, or the remainder of the current procedure.

**Suspend-Resume-Terminate commands**: You can explicitly interrupt or resume execution. You can also suspend and resume an attached external AVM at any time.

- 1. Open the source view of the **SalesReport.cls** form that you created in the previous section in editor view.
- 2. From the **Outline** view, select the **getOutletData** method from the Methods section.
- 3. On the left margin to the first executable line in **getOutletData**, double-click to set the break point.

	13020	METHOD PUBLIC VOID getOutletData(frmD
	1303	
	1304	DEFINE VARIABLE cntr AS INTEGER.
Ð	1305⊖	FOR EACH Outlet BREAK BY Outlet.0
	1306	cntr = 0.
	1307⊖	FOR EACH Order WHERE DATE(Ord
	1308	IF Order.Outlet = Outlet.
	1309⊝	DO:
	4 3 4 4	

- 4. Click Run ->Debug Configurations. The Debug Configurations dialog box opens.
- 5. Double–click **Progress OpenEdge Application**. Name the configuration as **SalesReport\_Debug**.



P Debug Configurations	×
Create, manage, and run co Create a configuration to launch a	nfigurations In OpenEdge application in debug mode.
Image: Second	Name:       SalesReport_Debug         OF Main       Startup       Project:         Project:       Browse       Copy Project Settings         Startup program:       Run selected program in workspace         SalesReports.ds       Workspace       File System         Working directory:       C:\DenEdge \WRK\Exchange_PDSOE\UDReports       Variables         OpenEdge QURK\Exchange_PDSOE\UDReports       Workspace       File System       Variables         OpenEdge Project runtime       Automatically start this launch configuration       Automatically restart this launch configuration         OpenEdge version:       11.3       Image: Apply
?	Debug Close

- 6. Click **Debug** to run the **SalesReport.cls** form in Debug mode.
- 7. Select **Yes** to enable debugging in the **Enable Debugger** dialog box.



- 8. Provide From and To Dates in the Outlet tab from UI and click GetOutletData.
- 9. Select Yes to switch the perspective.





- 10. In the **Debug** perspective, from the **Variables** section, select **frmData**, **tDate**, and **ttOutletOrders**, right-click and select **Watch**. The **Expressions** tab opens.
- 11. Select **Step Into toolbar** in the **Debug** view. **Step Into** causes the Debugger to execute the current line and continue until it reaches the next executable statement, which may be in the current procedure, a subprocedure, or a trigger. That statement becomes the current line, and is not executed until you continue. You can do **Step Into** to inspect each line and the value of variables at each line.



P Debug	- Reports/SalesReports.cls - Progress Developer Studio						_ 🗆 X
Eile Edit	Source Navigate Search Project Run OpenEdge Window Help						
] 📸 • 🗒 🐘 ] 🏇 • Ø • 隆 • ] 🧶 • ] ½ → ỗ → 🕂 🗘 • → -					E	🎋 Debug	»
🏇 Debug	1 🔀 👭 Servers 🛛 🔌 🕩 11 🔳 83 🕞 👁 🕫 🤿 🍸 🖓 🗖 🗖	(×)= Variables	lo Breakpoints	👷 Expressions 🕅	Dynamic Objects		- 8
	ew_configuration [Progress OpenEdge Application]				<u>k.</u> =		× ¾ ▽
Ē.	SalesReports.cls at localhost				Value	07/01/13	A
E	OpenEdge AVM (suspended)	y "ttOutletOrde	rs"		("Atlantic Ave", 4)		
	SalesReports getOutletData( ) line: 1319	Outlet	-		"Atlantic Ave"		
	SalesReports GetOutletData_Click( )	NoOfOrde	ers		4	1	
	C:\Theja\113Testing\Exchange_WorkShop\.metadata\.plugins\com.openedge.p	y "frmDate"			07/01/13		
>	C:\Progress\OpenEdge 113\bin\prowin32.exe (Sep 11, 2013 3:47:41 PM)	<sup>y</sup> "tDate"			09/13/13	1	
		Add new exp	ression			1	
							-
•		•			Þ	4	Þ
Sales	teports.ds (Design) C SalesReports.ds 🛛	- 0	🗄 Outline 🕅			(i)	~
1310	cntr = cntr + 1.	<b>_</b>	🕀 🏶 USING 🛙	Declarations			
1311	END.		🗄 🚯 Include:	s			
1312	END.		🗄 🕜 Variable	s			
1313	/* Create terms table with data*/		🗄 🗐 TempTa	bles			
1315	CREATE ttOutletOrders.		🗄 🔞 Method	s			
1316	ASSIGN			esReports			
1317	<pre>ttOutletOrders.Outlet = Outlet.OutletName</pre>		🖳 🖳 get	AreaData (date, date	=)		
1318	ttOutletOrders.NoOfOrders = cntr.		🖳 🖳 geti	DailyData			
> 1319	END.		Get 🚆	Data_Click_1 (System	n.Object, System.Event	Args)	
1320			get get	ItemData (date, date	e)		
1322	END METHOD.		Get	ItemData_Click (Syst	em.Object, System.Ever	ntArgs)	
1323			get get	OutletData (date, da	te)		
1324	/*		Get Get	OutletData_Click (Sy	stem.Object, System.Ev	entArgs)	
1325	Purpose:		Get Get	RegionData_Click (S)	stem.Object, System.Ev	ventArgs)	
1326	NOTES:			Benorte			
1328		<b>v</b> 1		ePenorte Load (Svo	tem Object System Eve	entArce)	
	<u>ا</u>		Sqie	carcepor is_coad (595	controbject, system.Eve	anon gaj	
1 =0					Lead Designed Used 6	- Ordenter	
] []		1:	1		Load Designer Host fo	or OrderApp	

- 12. Select **Resume** for the variables to continue running.
- 13. Select **Terminate** to stop debugging the form.
- 14. Close the **SalesReport** form.



## 5.12 Working with UltraChart (Take-home)

This section shows you how to design graphs to provide data in a graphical format using the UltraChart control.

- 1. Open the **Visual Designer** perspective.
- 2. Copy and paste the **Graphs.cls** file from

C:\OpenEdge\WRK\PDSOEWorkshopFiles\WorkshopFiles folder to the VDReports project in the Project Explorer view.

Save Close Visual Designer	Ctrl+S Ctrl+F4
View Source	F9

- 3. Open the **Graphs.cls** in the Design View.
- 4. Add four **UltraDataSource** controls to the Graphs form and change the **Name** property value to **Outlet**, **Area**, **ByDate**, and **ByItem** in the **Property** tab or using the **SmartTag**.



- 5. Click **File** $\rightarrow$ **Save All** to save all files.
- 6. Open **SmartTag** for the **Outlet** UltraDataSource and select the **UltraDataSource Designer** link. The **UltraDataSource Designer** dialog box appears.
- 7. Click Add a new column toolbar option in DataColumns section.

DataBands				
物性メキャ	DataColumns			
<mark>Band 0</mark>	<b>*</b>			
	Add a new column			

8. In the Properties section, enter **Outlet** in **Key** and **System.String** in **DataType**. Similarly add another column for **NoOfOrders** and **System.Int32** in **Data Type**. Your screen appears as follows:



DataBands					
and a columns DataColumns					
····· Band 0	🛬 🄀 🕆 🕂	Properties for Column 'NoOfOrder	's'		
	Outlet	₿∎ <mark>2</mark> ↓   E			
	Nouturders	AllowDBNull	Default		
		DataType	System.Int32		
		DefaultValue	DB (DBNull)		
		Кеу	NoOfOrders		
		ReadOnly	Default		
		Tag			

- 9. Select OK to close the UltraDataSource Designer dialog box.
- 10. Similarly, add **DataColumns** for other **UltraDataSources**.

UltraDataSources	DataColumns Key	DataType
Area	Area	System.String
	NoOfOrders	System.Int32
ByDate	OrderDate	System.DateTime
	NoOfOrders	System.Int32
ByItem	Item	System.String
	Orders	System.Int32

11. Drag and drop the UltraChart control on to the Outlets tab. The Select a Chart Type to Begin dialog box appears.



12. Click Finish. The UltraChart control is added to Graphs form.

13. Select the chart on the form. In the **Properties** tab, select **Ontlet** in **DataSource**. ESS

ultra	Properties 🛛 🔤	rs.Win.UltraWinChart.	' 🗆
Pr	operties Ever	nts	
	BorderStyle	None	
	ChartType	ColumnChart	
⊳	ColorModel	CustomLinear	
⊳	ColumnChart	(Chart Type Properti	
⊳	CompositeChart	(Chart Type Prop	_
	ContextMenuStri	(none)	=
	Cursor	Default	
⊳	Data	Outlet [Infragistics.W	
	DataMember		
	DataSource	Outlet 💌	
	Dock	Area	
⊳	Effects	ByDate	
	EmptyChartText	Byltem	
	EnableCrossHair	Outlet	Ŧ
Ab	out. Chart Wizard	(none)	

14. Similarly, add UltraChart in other tabs and select the DataSource as follows:

Tab	DataSource
Regions	Area
Items	ByItem
OrderByDate	ByDate

- 15. Save all the files.
- 16. Click **Run As** and select **Run As Progress OpenEdge Application**.



17. Close the **Run** dialog box. Click **File** $\rightarrow$ **Close All** to close all files.



# 6 LAB 06: Embedding an ABL window into the MDI form (Take-home)

#### 6.1 Overview

The sections of this lab show you how to create an MDI form and embed AppBuilder and Visual designer forms into this MDI form.

ee         Reports           File         Edit         View         Tools         Windows         Help	
Order         all         >Put title here>           Order New	
Get Customer	
Order Entry	
Item Search	PROGRESS FXCHANGE
Clear	
Discounts	
Order Reports	
Status	

#### 6.2 Prerequisites

Complete all the above labs prior to working on this lab.

#### 6.3 Creating an MDI form

This section shows you how to create an ABL MDI form.

- 1. Right-click **VDReports** and select **New→ABL MDI Form**. The **New ABL MDI Form** dialog box appears.
- 2. Enter **Reports** in **MDI Form name**.
- 3. Click **Finish**. The MDI form that is created opens in editor.



Reports.cls (Design) 🛛			
Reports			_ 0 🔀
<u>File Edit V</u> iew <u>T</u> ools	<u>W</u> indows <u>H</u> elp		
Status			
🐿 toolTip 🖿 statusStrip	🖻 menuStrip 🔤 too	Strip	

Note: The **New ABL MDI form** wizard creates a form enabled for multiple-document interface functionality; you can use it as a parent form for other forms. The form includes a toolbar containing common menus (File, Edit, View, Tools, Windows, and Help) and command buttons (New, Open, Save, Print, Print Preview, and Help) with pre-coded event subscriptions and logic. It also includes a status bar at the bottom.

Tip: To view complete MDI form double-click a tab in editor, it will expands tab size and occupies complete workbench area. Similarly, clicking any view expands the view area.

- 4. Click  $\bowtie$  icon in the **Console** view to close view.
- 5. Drag and drop the UltraExplorerBar control on to the Reports MDI form. Modify the following properties in the **Properties** view and save the changes.

Property	Value	Displays as:	
Dock	Left	Dock	Left
Style	OutlookNavigationP	Style	OutlookNavigationPane
	ane		

In the **Properties** view, the following links are provided for the UltraExplorerBar control:

About, Custom Property Pages..., UltraExplorerBar Designer..., Add Group, Add Item, Remove Group, Remove Item, Load Layout..., Save Layout...



6. Click the Add Group link. A New Group section is added to UltraExplorerBar.

New Group	
c	
New Group	
×	

- 7. Click the **UltraExplorerBar Designer** link. The **UltraExplorerBar Designer** dialog box appears.
- 8. Click the Groups and Items tab.

UltraExplorerBar Designer						
UltraExplorerBar						
Control Styles Groups and Items	rgins/Spacing Load/Save Layout Import Groups/Items					
Add Group	d Item					
🖃 🛄 UltraExplorerBar Control	Dock Left					
[0] -New Group	Enabled True					
	Font Microsoft Sans Serif, 7.8p	t				
	GenerateMember True					
	Groups (Collection)					
	GroupSettings					
	GroupSpacing 15	=				
	ImageListLarge (none)					
	ImageListSmall (none)					
	ImageSizeLarge 32, 32					
	ImageSizeSmall 16, 16					
	ImageTransparentColor Transparent					
	ImeMode NoControl					
	ItemSettings					
	Location 0, 53					
	Locked False					
Margin 3, 3, 3, 3						
		<u>C</u> lose				



- 9. Select the [0] New Group node. Enter Order in the Text property.
- 10. Click Add Item. It adds a child node to Order.
- 11. Enter **Order New** in the **Text** property.

Control Styles Groups and Items	Margins/Spacing Load/Save Layout Imp	ort Groups/Items
♣ Add Group	Add Item	Remove Item
🖃 🛄 UltraExplorerBar Control	Checked	False
📄 🔚 [0] -Order	Key	
[0] - Order New	Settings	
	Tag	
	Text	Order New
	ToolTipText	
	Visible	True

12. Similarly, add another **Reports** group and **View Reports** and **View Sales By Date** child nodes.

🖃 🛄 UltraExplorerBar Control		Checked	False
🖕 🛅 [0] -Order		Кеу	
[0] - Order New	⊳	Settings	
🗄 📴 [1] -Reports		Tag	
[0] - View Reports		Text	View Sales By Date
[1] - View Sales By Date		ToolTipText	
		Visible	True

13. Click **Close**. The groups you added are listed and when you click the groups, the corresponding items are listed.



🖳 Reports					
<u>File E</u> dit <u>V</u> iew <u>T</u> ools <u>W</u> i					
Reports					
View Reports					
🔊 View Sales By Date					
Order					
Reports					
»					
Status					

14. Open SmartTag of UltraExplorerBar and select Label in Style of Items as shown in the screen below:

Order	UltraExplorerBar Ta	UltraExplorerBar Tasks				
	Name	ultraExplorerBar1				
Order New						
	UltraExplorerBar Desig	UltraExplorerBar Designer				
	Behavior	Behavior				
	Style of Control	OutlookNavigationP				
	Style of Groups	Default 💌				
	Style of Items	Label				
	Data	Default Button				
	Add Group	StateButton				
	1	Label				
	Add Item	Separator				

15. Save the changes.

#### 6.4 Embedding the ABL windows into an MDI form

This section shows you how to add a code to embed an ABL window and GUI windows designed as part of the above labs into this MDI form.

- 1. Select UltraExplorerBar bar in the editor.
- 2. Click the **Event**s tab in the **Properties** view. The events available for this control are listed.



Properties	🛃 🖩 灯 🗟 🗆 🗆				
ultraExplorerBar1 : Infragisti	ultraExplorerBar1 : Infragistics.Win.UltraWinExplorerE 🔻				
Properties Events					
GroupRemoved					
GroupRemoving					
HelpRequested					
ImeModeChanged					
ItemAdded					
ItemAdding					
ItemCheckStateChanc	=				
ItemCheckStateChanc					
ItemClick	•				
ItemDoubleClick					
ItemDragging					
ltemDragOver					
ItemDropped					
ItemEnteredEditMode					
ItemEnteringEditMode	*				

- 3. Double-click the field next to the **ItemClick** event method. The source view opens and the **ultraExplorerBar1\_ItemClick** event method is added to the code.
- 4. Copy and paste the following code to display **SalesReports** and **Graphs** as child forms when specific items are clicked and when **OrderNew** is selected, ABL window **Order.w** is opened as the embedded window. You can alternatively write a code inside the ItemClick method.

```
METHOD PRIVATE VOID ultraExplorerBar1 ItemClick( INPUT sender AS
System.Object, INPUT e AS
Infragistics.Win.UltraWinExplorerBar.ItemEventArgs ):
      DEFINE VARIABLE CKey AS CHARACTER NO-UNDO.
       DEFINE VARIABLE cls1 AS form .
       DEFINE VARIABLE cls2 AS form .
        cKey = e:ITEM:Text.
        cls1 = NEW Graphs().
        cls2 = NEW SalesReport().
        CASE cKey:
            WHEN 'Order New' THEN embedABLWin(e).
            WHEN 'View Reports' THEN ShowChildForm(cls1,e).
           WHEN 'View Sales By Date' THEN ShowChildForm(cls2,e).
        END.
END METHOD.
/*---- ShowChildForm method -----*/
   METHOD PRIVATE VOID ShowChildForm(childForm AS
Progress.Windows.Form, e AS System.EventArgs):
        /* Make it a child of this MDI form before showing it. */
        childForm:MdiParent = THIS-OBJECT.
        childForm:FormClosed:SUBSCRIBE(Form_Closed).
        childForm:Show( ).
    END.
 *---- embedABLWin method -----*/
   METHOD PUBLIC VOID embedABLWin( e AS System.EventArgs ):
```

**\***PROGRESS

```
DEF
               VAR
                        childForm AS Progress.Windows.MDIChildForm
NO-UNDO.
        DEFINE VARIABLE h
                                  AS HANDLE
NO-UNDO.
        DFF
               VAR
                        hwin
                                  AS HANDLE
NO-UNDO.
        RUN OrderApp.w PERSISTEN SET h.
        hwin = h:CURRENT-WINDOW.
        /* Create the WindowContainer, embedding the window into it.
*/
        childForm = NEW Progress.Windows.MDIChildForm( ).
        childForm:Size = New System.Drawing.Size( hWin:WIDTH-PIXELS,
hWin:HEIGHT-PIXELS ).
        childForm:Dock = System.Windows.Forms.DockStyle:Fill.
        childForm:EmbeddedWindow = hwin.
        childForm:MdiParent = THIS-OBJECT.
        childForm:FormClosed:SUBSCRIBE(Form Closed).
        childForm:Show( ).
        RUN initializeObject IN h.
    END METHOD.
```

5. Save the file. Warning **b** marker appear as follows:

▲896 Keywords are in lower case.e = New System.Drawing.Size( hWin:WIDTH-PIXELS, hWin:HEIGHT-P]

6. Press Ctrl+Shift+F and observe that the casing of New keyword is changed to All Caps.

&896 childForm:Size = NEW System.Drawing.Size( hWin:WIDTH-PIXELS, hWin:HEIGHT-P]

**Tip:** To set your keyword casing preference, update the preferences in the Editor Preferences dialog box. You can enable ABL Editor to automatically apply keyword casing as you type and when you save the file. Optionally, select a block of code to which you want to apply keyword casing, and press CTRL+SHIFT+F, or select Source > Correct Case. If no text is selected, the entire file is formatted.

- 7. To access the **OrderApp.w** AppBuilder file from the **VDReports** project, you need to change the PROPATH for the project.
- 8. Right-click the **VDReports** project and click **Properties**. The **Properties** dialog box opens.
- 9. Click PROPATH from the left section, and click Add WorkSpace Directory. Add the OrderApp directory and click OK. OrderApp appears in the PROPATH list.





- 10. Save the file and run the file as Progress OpenEdge Application.
- 11. Click Order New Item, the embedded form appears:



12. Close the **Run** dialog box.



# 7 Configuring Apache WebServer (not required for the Workshop)

For the workshop, Apache is pre-configured on the machines that are provided to you. The steps below are provided to help you in configuring Apache WebServer beyond workshop.

- Install Apache HTTP Server from following location: http://httpd.apache.org/download.cgi#apache22 Win32 Binary without crypto (no mod\_ssl) (MSI Installer): httpd-2.2.25-win32-x86no\_ssl.msi
- Change the listening port from 80 to 8080 in the Listen field; open httpd.conf file from C:\Program Files (x86)\Apache Software Foundation\Apache2.2\conf. Specify 8080. Save the file.
- Copy the cgiip.exe file from DLC\bin (DLC refers to the install location C:\Progress\OpenEdge) to C:\Program Files (x86)\Apache Software Foundation\Apache2.2\cgi-bin.
- 4. Copy the WebSpeed directory from \$DLC to C:\Program Files (x86)\Apache Software Foundation\Apache2.2\htdocs
- 5. Restart the Apache server.





Progress Developer Studio for OpenEdge gives you a robust development environment for various types of application that you build. We continue to add and enhance features in every release with continued focus on the productivity of a developer.

We hope you will give Progress Developer Studio for OpenEdge a try.

