

Getting Started

PowerServer Mobile® 2020
FOR WINDOWS & UNIX & LINUX

DOCUMENT ID: ADC50009-01-2020-01

LAST REVISED: March 25, 2020

Copyright © 2020 Appeon. All rights reserved.

This publication pertains to Appeon software and to any subsequent release until otherwise indicated in new editions or technical notes. Information in this document is subject to change without notice. The software described herein is furnished under a license agreement, and it may be used or copied only in accordance with the terms of that agreement.

No part of this publication may be reproduced, transmitted, or translated in any form or by any means, electronic, mechanical, manual, optical, or otherwise, without the prior written permission of Appeon Inc.

Appeon, the Appeon logo, Appeon PowerBuilder, Appeon PowerServer, PowerServer, PowerServer Toolkit, AEM, and PowerServer Web Component are trademarks of Appeon Inc.

SAP, Sybase, Adaptive Server Anywhere, SQL Anywhere, Adaptive Server Enterprise, iAnywhere, Sybase Central, and Sybase jConnect for JDBC are trademarks or registered trademarks of SAP and SAP affiliate company.

Java and JDBC are trademarks or registered trademarks of Sun Microsystems, Inc.

All other company and product names used herein may be trademarks or registered trademarks of their respective companies.

Use, duplication, or disclosure by the government is subject to the restrictions set forth in subparagraph (c)(1)(ii) of DFARS 52.227-7013 for the DOD and as set forth in FAR 52.227-19(a)-(d) for civilian agencies.

Appeon Inc., 1/F, Shell Industrial Building, 12 Lee Chung Street, Chai Wan District, Hong Kong.

Contents

1	Welcome to PowerBuilder & PowerServer Mobile	1
1.1	Introduction to PowerBuilder	1
1.1.1	What PowerBuilder is	1
1.1.2	The PowerBuilder environment	2
1.1.3	PowerBuilder objects	7
1.2	Introduction to PowerServer Mobile	15
1.2.1	What PowerServer Mobile is	15
1.2.2	The PowerServer Mobile n-tier architecture	16
1.3	About this tutorial	17
1.3.1	Learning to install PowerBuilder and PowerServer Mobile	17
1.3.2	Learning to build a mobile application	17
1.3.3	Learning to deploy a mobile application	18
2	Installing PowerBuilder & PowerServer	20
2.1	Setting up the environment	20
2.1.1	Prepare the environment	20
2.1.2	Disable UAC (User Account Control)	21
2.1.3	Install IIS	21
2.1.4	Configure IIS	24
2.2	Installing PowerBuilder and PowerServer	26
2.2.1	Installing PowerBuilder and PowerServer	26
2.2.2	Verify the PowerServer installation	28
2.2.3	Install Apeon Workspace (on the Android device)	28
2.2.4	Configure the network connection	28
3	Building a Mobile Application	29
3.1	Starting PowerBuilder	29
3.1.1	Create a new workspace	29
3.1.2	Create a target	32
3.2	Customizing the PowerBuilder Environment	35
3.2.1	Manipulate the System Tree window	35
3.2.2	Open an object	36
3.2.3	Manipulate views	37
3.2.3.1	Add an extra Script view	38
3.2.3.2	Display view title bars	38
3.2.3.3	Float and dock views	39
3.2.3.4	Manipulate tabbed views	39
3.2.3.5	Save a view layout scheme	40
3.2.3.6	Reset the default view layout scheme	40
3.2.4	Set up the toolbars	40
3.2.4.1	Show labels on toolbar buttons	41
3.2.4.2	Float the toolbars	41
3.2.4.3	Reposition the toolbars	43
3.3	Connecting to the Database	45
3.3.1	About database connection	45
3.3.2	Create the database profile for the Apeon Sample database	46
3.3.3	Look at table definitions in the Apeon Sample database	49
3.4	Setting Up the Menus	53

3.4.1	Create a new menu	54
3.4.2	Add items to the new menu	54
3.4.3	Add a new toolbar for the new menu item	55
3.5	Building a DataWindow Object	56
3.5.1	Create and preview a DataWindow object	56
3.5.2	Save the DataWindow object	61
3.5.3	Make cosmetic changes to the DataWindow object	61
3.6	Building a Window	63
3.6.1	Build an MDI frame window	63
3.6.1.1	Create an MDI frame window	63
3.6.1.2	Attach menu to the frame window	67
3.6.1.3	Change the size of the frame window	68
3.6.1.4	Write scripts to open the frame window	69
3.6.2	Build an MDI sheet window	72
3.6.2.1	Create an MDI sheet window	72
3.6.2.2	Change the size of the sheet window	74
3.6.2.3	Add controls to the sheet window	75
3.6.2.4	Write scripts to connect with the Appeon Sample database	83
3.6.2.5	Write menu scripts to open the sheet window	84
3.6.3	Run the PowerBuilder application	84
4	Deploying the Mobile Application	88
4.1	Configuring and deploying the application in PowerServer Toolkit	88
4.2	Running the mobile application in Appeon Workspace	96
Index	100

1 Welcome to PowerBuilder & PowerServer Mobile

This chapter is an overview of the PowerBuilder development environment and the PowerServer Mobile deployment tool & runtime environment.

1.1 Introduction to PowerBuilder

About this section

This section introduces the PowerBuilder development environment, which you use in this tutorial. It also describes the building blocks of a PowerBuilder application.

For more information

For a more detailed description of the PowerBuilder development environment, see the *PowerBuilder Users Guide*.

1.1.1 What PowerBuilder is

PowerBuilder is an enterprise development tool that allows you to build PowerBuilder applications easily and rapidly. And the PowerBuilder application can be deployed as a Client/Server app (via the PowerBuilder deployment tool), a Web app, and a Mobile app (via the PowerServer deployment tool), which is truly, develop once, deploy everywhere.

What's in a PowerBuilder application

A PowerBuilder application can contain:

- **A user interface:** Menus, windows, and window controls that users interact with to direct an application.
- **Application processing logic:** Event and function scripts in which you code business rules, validation rules, and other application processing. PowerBuilder allows you to code application processing logic as part of the user interface or in separate modules called custom class user objects.

PowerBuilder applications are event driven

In a PowerBuilder application, users control what happens by the actions they take. For example, when a user taps a button, chooses an item from a menu, or enters data into a text box, one or more events are triggered. You write scripts that specify the processing that should happen when events are triggered.

Windows, controls, and other application components you create with PowerBuilder each have a set of predefined events. For example, each button has a Clicked event associated with it and each text box has a Modified event. Most of the time, the predefined events are all you need. However, in some situations, you may want to define your own events.

PowerScript language

You write scripts using PowerScript, the PowerBuilder language. Scripts consist of PowerScript commands, functions, and statements that perform processing in response to an event.

For example, the script for a button's Clicked event might retrieve and display information from the database; the script for a text box's Modified event might evaluate the data and perform processing based on the data.

The execution of an event script can also cause other events to be triggered. For example, the script for a Clicked event in a button might open another window, triggering the Open event in that window.

PowerScript functions

PowerScript provides a rich assortment of built-in functions that can act on the various components of your application. For example, there is a function to open a window, a function to close a window, a function to enable a button, a function to update the database, and so on.

You can also build your own functions to define processing unique to your application.

Object-oriented programming with PowerBuilder

Each menu or window you create with PowerBuilder is a self-contained module called an object. The basic building blocks of a PowerBuilder application are the objects you create. Each object contains the particular characteristics and behaviors (properties, events, and functions) that are appropriate to it. By taking advantage of object-oriented programming techniques such as encapsulation, inheritance, and polymorphism, you can get the most out of each object you create, making your work more reusable, extensible, and powerful.

Database connectivity

PowerBuilder provides easy access to corporate information stored in a wide variety of databases. Data can be accessed through the PowerBuilder ODBC or JDBC interfaces, through a middle-tier data access server like the SAP DirectConnect server, or through a native or direct connection to a database.

For information on database connectivity, see *Connecting to Your Database*.

Help and documentation

PowerBuilder Help can be accessed using Help buttons and menu items, or by selecting the F1 key from anywhere in PowerBuilder. PDF-format manuals are also available on the Appeon Web site.

1.1.2 The PowerBuilder environment

Workspaces and targets

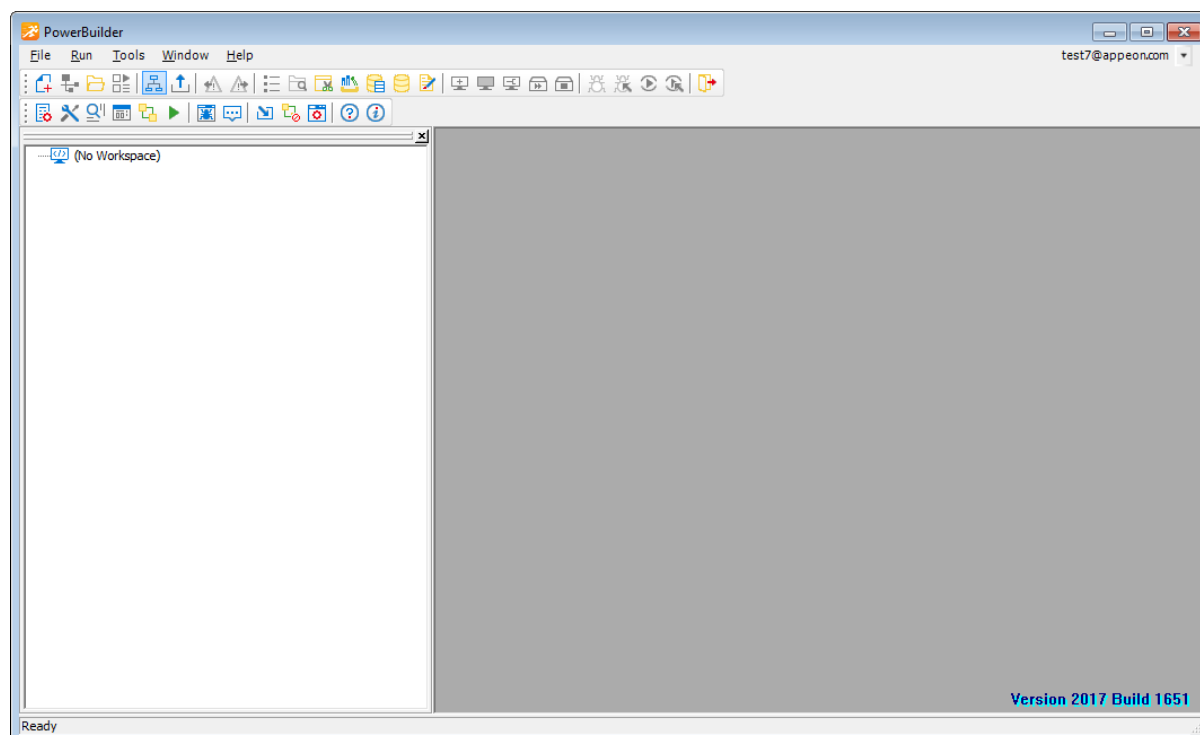
In PowerBuilder, you work with one or more targets in a workspace. You can add as many targets to the workspace as you want, open and edit objects in multiple targets, and build and deploy multiple targets at once.

A PowerBuilder application target can be a client/server, Web, or mobile executable application.

This tutorial shows you how to create a workspace and an Application target.

The development environment

When you start PowerBuilder, it opens in a window that contains a menu bar and the PowerBar at the top, and the System Tree and Clip windows on the left.

Figure 1.1: Development environment

System Tree

The System Tree window can serve as the hub of your development activities. You use it to open, run, debug, and build your targets, and for drag-and-drop programming.

Clip window

The Clip window lets you store code fragments that you use frequently.

Output window

The output of a variety of operations (migration, builds, object saves, and searches) displays in an Output window at the bottom of the main window. The Output window opens automatically when output information is generated, but you can open the Output window at any time by clicking the Output window toolbar button.

Painters

Once you have created a workspace and a PowerScript target, you build the components of the target using painters. Painters provide an assortment of tools for enhancing and fine tuning the objects in a target.

PowerBuilder provides a painter for each type of object you build. For example, you build a window in the **Window** painter. There you define the properties of the window and add controls, such as buttons and text boxes.

Wizards

Wizards simplify the creation of applications, objects, etc.

To-Do List

The To-Do List displays a list of development tasks you need to do for the current target. Entries on the To-Do list can be created automatically by most PowerBuilder wizards. You

can also type in entries or import them from a text file and then link them to a task that you want to complete.

Browser

The Browser lets you see all the objects, methods, variables, and structures that are defined for or available to your PowerScript target. Objects in the Browser can be displayed in alphabetic or hierarchical order. The Browser displays methods with their complete prototypes (signatures), which include the datatypes of all arguments and return values.

PowerBar

The PowerBar displays when you begin a PowerBuilder session. The PowerBar is the main control point for building PowerBuilder applications. You can use the New, Inherit, or Open buttons on the PowerBar to open all of the PowerBuilder painters. From the PowerBar, you can also open the Browser, debug or run the current application, and build and deploy the workspace.

PainterBar

When you open a painter or editor, PowerBuilder displays a new window that has a workspace in which you design the object you are building. PowerBuilder also displays one or more PainterBars with buttons that provide easy access to the tools available in the painter or editor. For example, here is the PainterBar for the **DataWindow** painter.

Figure 1.2: PainterBar



StyleBar

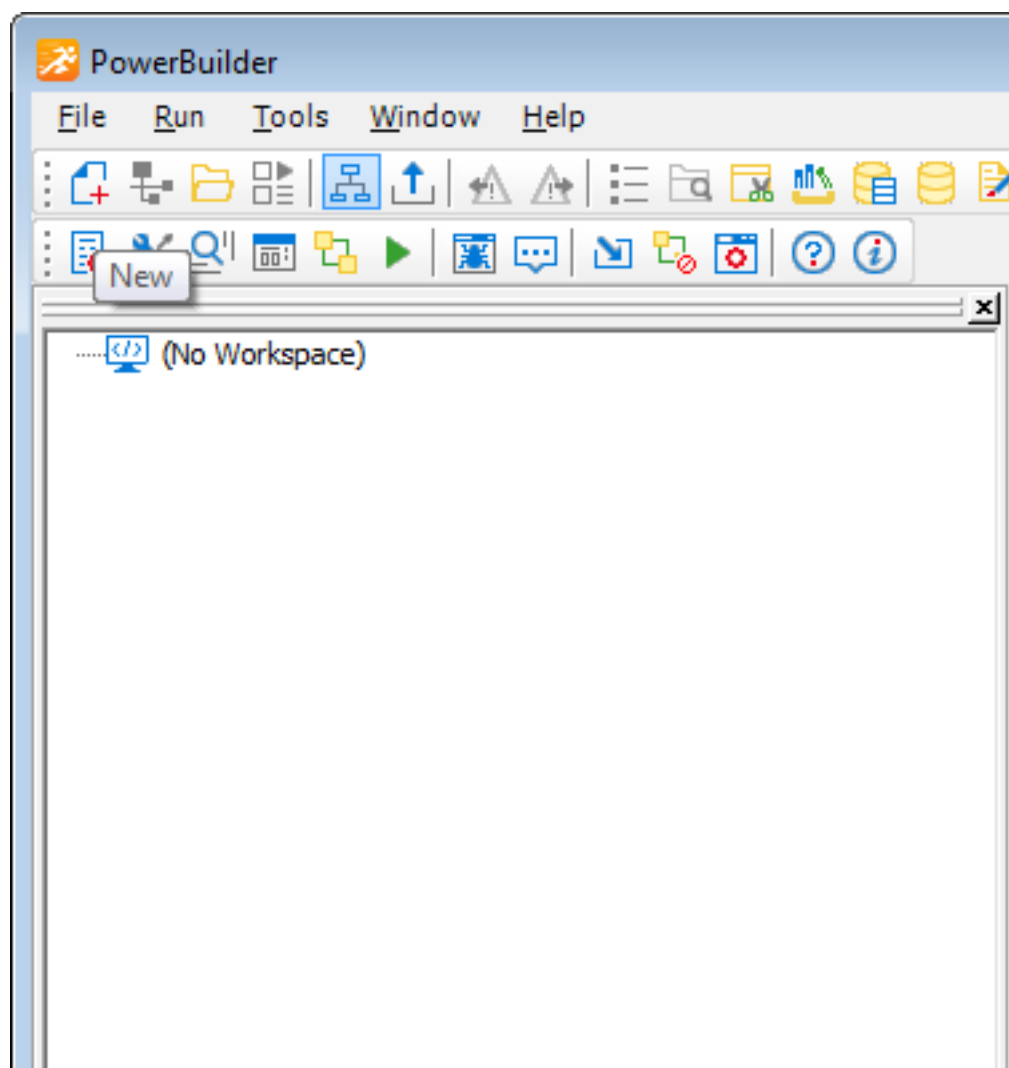
The StyleBar displays when you open any painter that can contain text controls, such as the **Window** painter. Using buttons on the StyleBar, you can modify text properties such as the font and point size.

Figure 1.3: StyleBar



PowerTips

When you leave the mouse pointer over a button for a second or two, PowerBuilder can display a brief description of the button (a PowerTip). The ability to display PowerTips is toggled on and off by selecting the Show PowerTips menu item in any toolbar pop-up menu.

Figure 1.4: PowerTips

You can also include brief descriptive texts on all toolbar buttons by selecting ShowText from any toolbar pop-up menu.

Customizing the environment

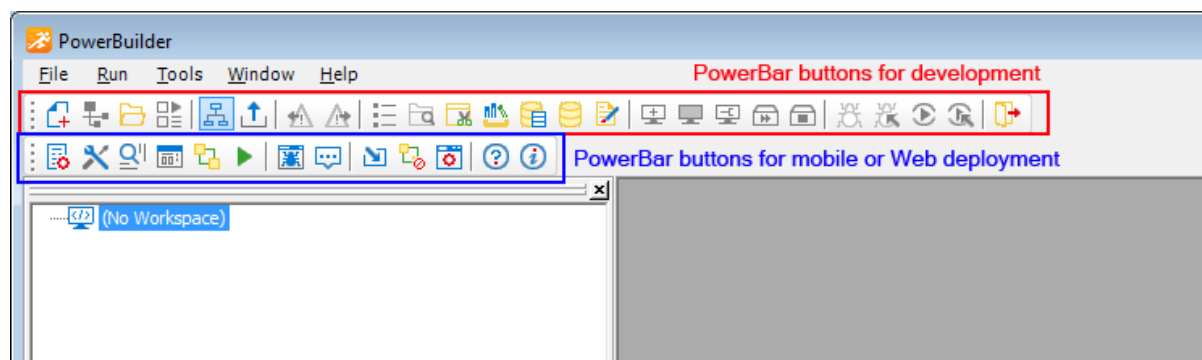
In addition to displaying text in toolbar buttons, you can move the toolbars around, add new toolbars, and customize existing ones. You can add buttons for opening painters and performing other activities.

You can also rearrange the System Tree, Clip, and Output views, set up custom layouts for each painter, choose whether PowerBuilder opens your last workspace at start-up with or without painters and editors open, customize shortcut keys, and change the colors and fonts used in scripts.

PowerBar buttons











The buttons in the PowerBar give you quick access to the most common PowerBuilder tasks.

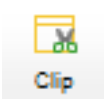

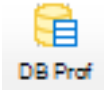
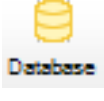

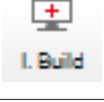
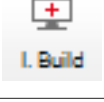
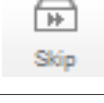
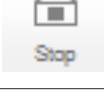
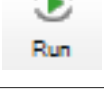
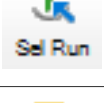

Figure 1.5: PowerTips



The following table lists only the tasks related with the mobile app development. The tasks for mobile deployment will be introduced later.

Table 1.1: Buttons in the PowerBar

Button	Use to
 New	Create new workspace, target, or other object, or open a tool.
 Inherit	Inherit from menu, user object, or window.
 Open	Open an existing application, DataWindow, function, menu, query, structure, user object, or window.
 Preview	Preview a window or DataWindow object.
 SysTree	Show or hide the System Tree window.
 Output	Show or hide the Output window.
 Next	Move to the next line in the Output window.
 Previous	Move to the previous line in the Output window.
 To-Do List	Display a list of development tasks you need to do. These can be self entered or entered automatically by PowerBuilder wizards.
 Browser	View object information (such as object properties or global variables) and copy, export, or print it.

Button	Use to
 Clip	Show or hide the Clip window.
 Library	Create and maintain libraries of PowerBuilder objects.
 DB Prof	Specify how to connect to a database.
 Database	Maintain databases, control user access to databases, and manipulate data in databases.
 Edit	Edit a file.
 I. Build	Start an incremental build of the workspace.
 I. Build	Start a full build of the workspace.
 Skip	When a series of operations is in progress, such as a full deploy of the workspace, skip to the next operation.
 Stop	Stop a build operation or series of operations.
 Run	Run the current target.
 Sel Run	Select a target and run it.
 Exit	Exit from PowerBuilder.

1.1.3 PowerBuilder objects

The basic building blocks of a PowerScript target are objects:

Table 1.2: Basic building blocks of a PowerScript target

Object	Use
Application	Entry point into an application
Window	Primary interface between the user and a PowerBuilder application

Object	Use
DataWindow	Retrieves and manipulates data from a relational database or other data source
Menu	List of commands or options that a user can select in the currently active window
Global function	Performs general-purpose processing
Query	SQL statement used repeatedly as the data source for a DataWindow object
Structure	Collection of one or more related variables grouped under a single name
User object	Reusable processing module or set of controls, either visual or nonvisual

These objects are described in more detail in the sections that follow.

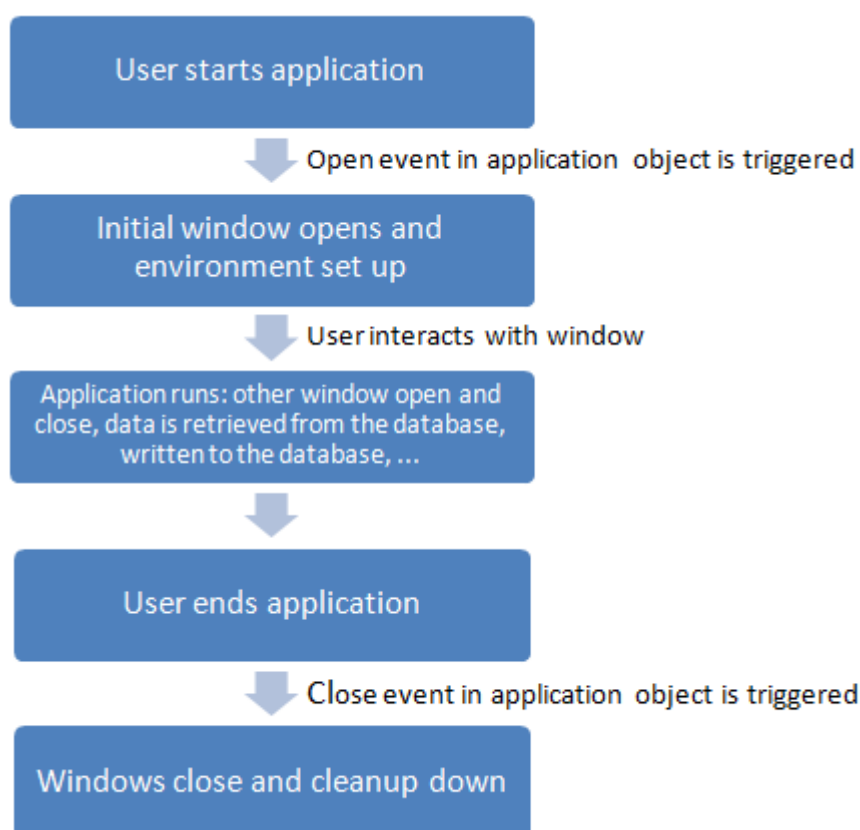
Application object

The Application object is the entry point into an application. It is a discrete object that is saved in a PowerBuilder library (PBL file), just like a window, menu, function, or DataWindow object.

The Application object defines application-level behavior, such as which fonts are used by default for text, and what processing should occur when the application begins and ends.

When a user runs the application, an Open event is triggered in the Application object. The script you write for the Open event initiates the activity in the application. When the user ends the application, the Close event in the Application object is triggered.

The script you write for the Close event typically does all the cleanup required, such as closing a database or writing to a preferences file.

Figure 1.6: Application life cycle

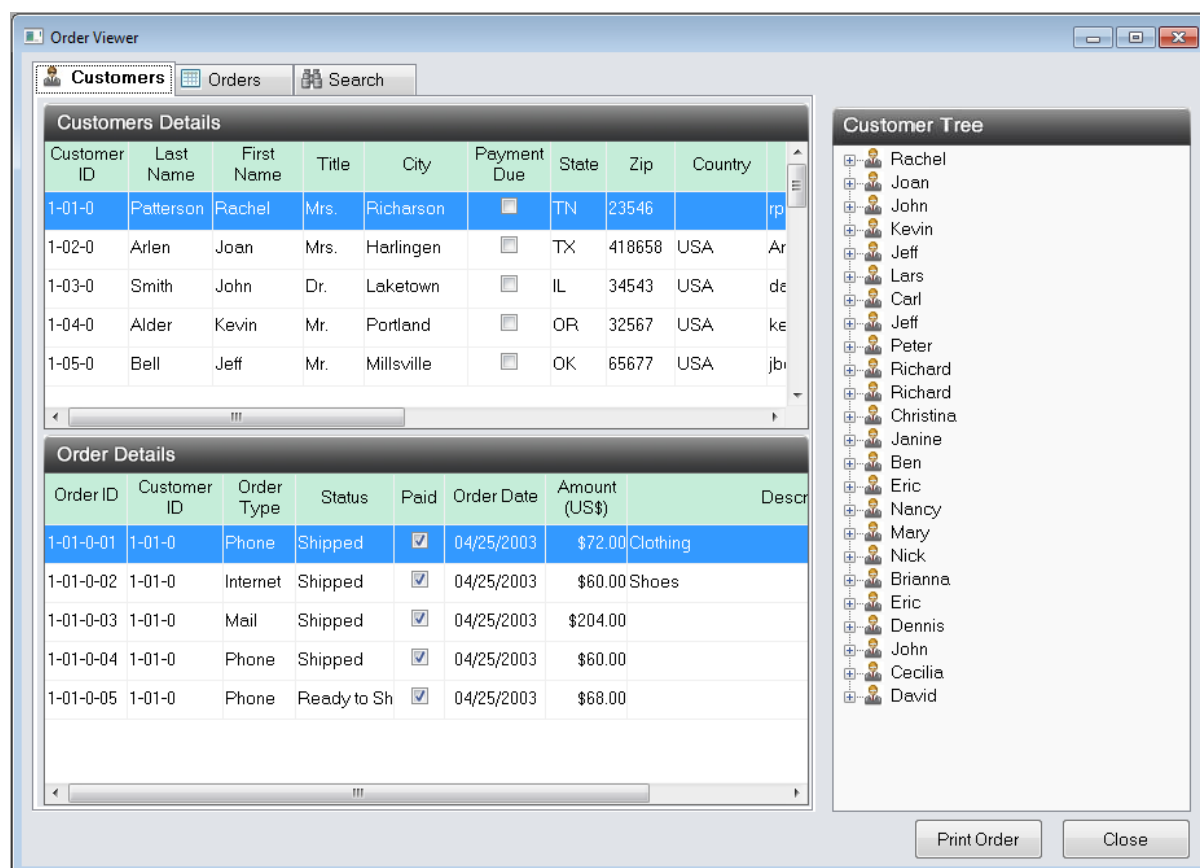
Windows

Windows are the primary interface between the user and a PowerBuilder application. Windows can display information, request information from a user, and respond to the users mouse or keyboard actions.

A window consists of:

- Properties that define the windows appearance and behavior (for example, a window might have a title bar and a Minimize box)
- Events triggered by user actions
- Controls placed in the window

Windows can have various kinds of controls, as illustrated in the following picture:

Figure 1.7: Controls placed in the window

On the left of the window is a Tab control with pictures and labels, on the Tab control is two DataWindow controls, and on the DataWindow control is check boxes. On the right is a TreeView control. Above the DataWindow control and the TreeView control is StaticText controls with black background. On the bottom right corner of the window is command buttons.

DataWindow objects

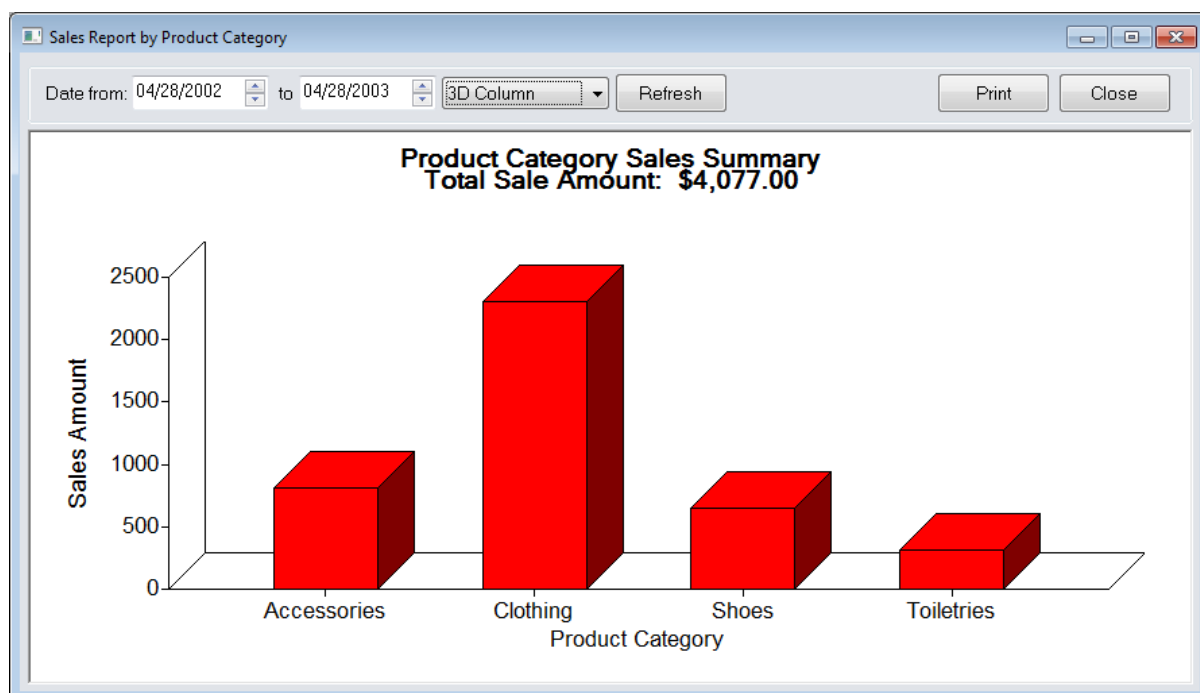
A DataWindow object is an object that you use to retrieve and manipulate data from a relational database.

1. Presentation styles

DataWindow objects also handle the way data is presented to the user. You can choose from several presentation styles. For example, you can display the data in Tabular or Freeform style.

There are many ways to enhance the presentation and manipulation of data in a DataWindow object. For example, you can include computed fields and graphs that are tied directly to the data retrieved by the DataWindow.

Figure 1.8: DataWindow with computed fields and graphs



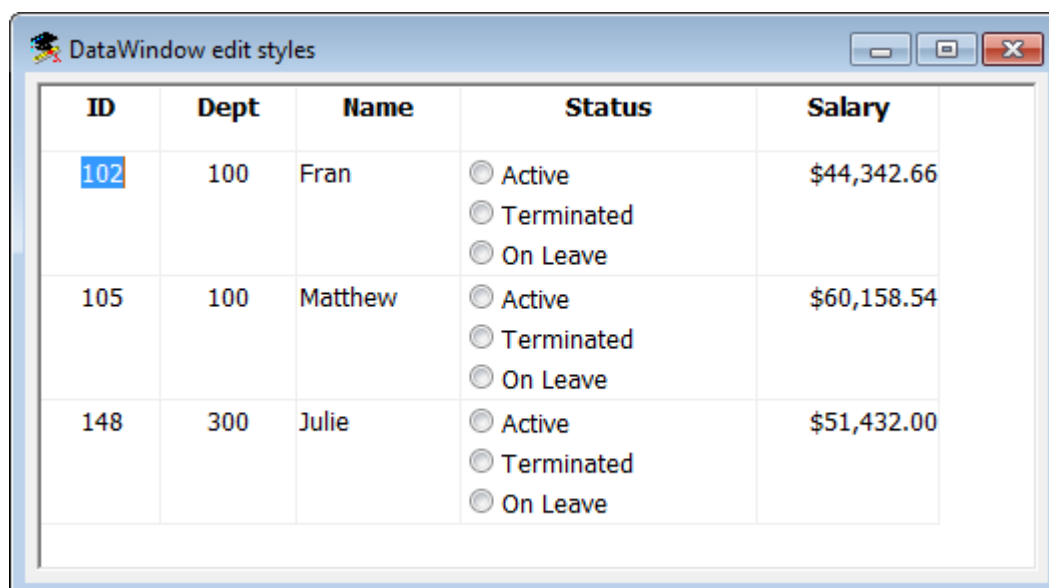
2. Display formats, edit styles, and validation

You can specify how to display the values for each column, and you can validate data entered by users in a DataWindow object. You do this by defining display formats, edit styles, and validation rules for columns.

For example:

- If a column can take only a small number of mutually exclusive values, you can have the data appear as radio buttons in a DataWindow so users know what their choices are.

Figure 1.9: Radio buttons in a DataWindow



- If the data includes phone numbers, salaries, and dates, you can format the display to suit the data.

Figure 1.10: Data format



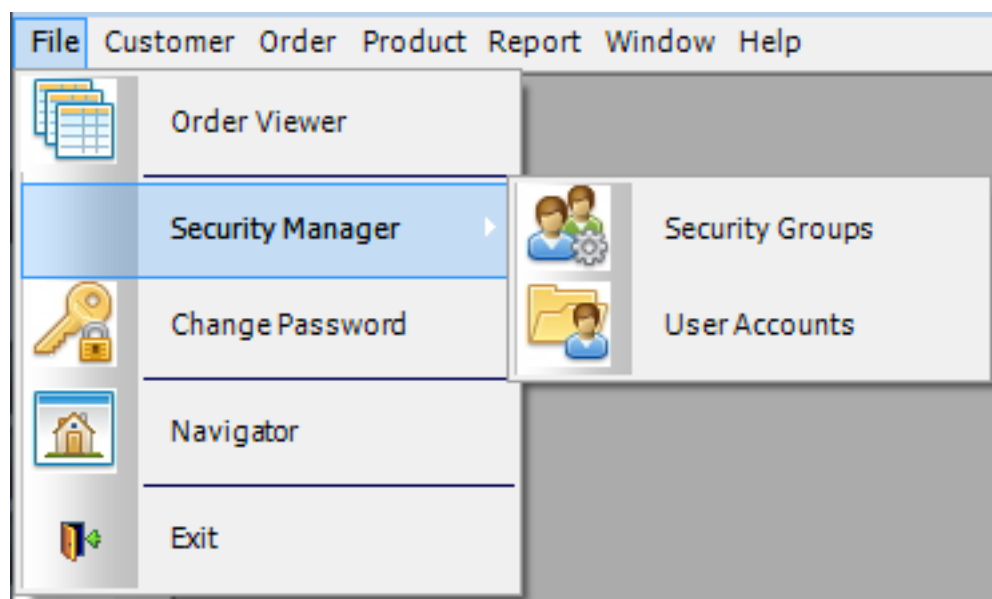
ID	Name	Phone	Zip Code	Salary	Start Date
102	Fran	(617) 555-3985	02192	\$44,342.66	02/26/1987
105	Matthew	(617) 555-3840	02154	\$60,158.54	07/02/1987
129	Philip	(404) 555-2341	30339	\$379.82	08/04/1998
148	Julie	(617) 555-7835	01890	\$51,432.00	10/04/1997
184	Melissa	(508) 555-2319	01775	\$36,490.00	04/18/1988

- If a column can take numbers only in a specific range, you can specify a simple validation rule for the data. This can spare you from writing code to make sure users enter valid data.

Menus

Menus are lists of items that a user can select from a menu bar for the active window. The items on a menu are usually related. They provide the user with commands (such as Open and Save As on the PowerBuilder File menu) or alternate ways of performing a task (for example, the items on the **Edit** menu in the **Window** painter correspond to buttons in the PainterBar).

A drop-down menu is a menu under an item in the menu bar. A cascading menu is a menu that appears to the side of an item in a drop-down menu.

Figure 1.11: Drop-down menu

Each choice in a menu is defined as a Menu object in PowerBuilder. The preceding window shows seven Menu objects on the menu bar (File, Customer, Order, Product, Report, Window and Help), five Menu objects on the drop-down Data menu (Order View, Security Manager, Change Password, Navigator, and Exit), and two Menu objects on the cascading menu beside Security Manager (Security Groups and User Accounts).

Global functions

PowerBuilder lets you define two types of functions:

- Object-level functions are defined for a particular type of window, menu, or other object type and are encapsulated within the object for which they are defined. These are further divided into system functions (functions that are always available for objects of a certain object class) and user-defined functions.
- Global functions are *not* encapsulated within another object, but instead are stored as independent objects.

Unlike object-level functions, global functions do not act on particular instances of an object. Instead, they perform general-purpose processing such as mathematical calculations or string handling.

Queries

A query is a SQL statement that is saved with a name so that it can be used repeatedly as the data source for a DataWindow object. Queries enhance developer productivity, because they can be coded once but reused as often as necessary.

Structures

A structure is a collection of one or more related variables of the same or different data types grouped under a single name. In some languages, such as Pascal and COBOL, structures are called records.

Structures allow you to refer to related entities as a unit rather than individually. For example, you can define the users ID, address, access level, and a picture (bitmap) of the employee as a structure called `user_struct`, and then refer to this collection of variables as `user_struct`.

There are two kinds of structures:

- Object-level structures are associated with a particular type of object such as a window or menu. These structures can always be used in scripts for the object itself. You can also choose to make the structures accessible from other scripts.
- Global structures are not associated with any object or type of object in an application. You can declare an instance of the structure and reference it in any script in an application.

User objects

Applications often have features in common. For example, several applications might have a **Close** button that performs a certain set of operations and then closes the window, or they might have DataWindow controls that perform the same type of error checking. Several applications might all require a standard file viewer.

If you find yourself using the same application feature repeatedly, you should define a user object. You define the user object once and use it as many times as you need.

User objects can be visual or nonvisual. They can be further divided into standard or custom user objects. Standard user objects, whether visual or nonvisual, are system objects that are always available with PowerBuilder. You can also use controls for external visual objects that were created outside PowerBuilder. The main types of user objects are:

- **Visual user objects**

These are reusable controls or sets of controls that have a consistent behavior. For example, a visual user object could consist of several buttons that function as a unit. The buttons could have scripts associated with them that perform standard processing. Once the object is defined, you can use it as often as you need.

- **Nonvisual user objects**

These are reusable processing modules that have no visual component. Standard class user objects inherit events and properties from built-in system objects. You typically use nonvisual objects to define business rules and other processing that acts as a unit.

For example, you might want to calculate commissions or perform statistical analysis in several applications. To do this, you could define a custom class user object. To use a custom class user object, you create an instance of the object in a script and call its functions.

Libraries

You save objects, such as windows and menus, in PowerBuilder libraries (PBL files). When you run an application, PowerBuilder retrieves the objects from the library. Applications can use as many libraries as you want. When you create an application, you specify which libraries it uses.

1.2 Introduction to PowerServer Mobile

About this section

This section introduces the PowerServer Mobile deployment tool & runtime environment, which you use in this tutorial.

For more information

For a more detailed description of the PowerServer Mobile deployment tool & runtime environment, see the Apeon Online Help.

1.2.1 What PowerServer Mobile is

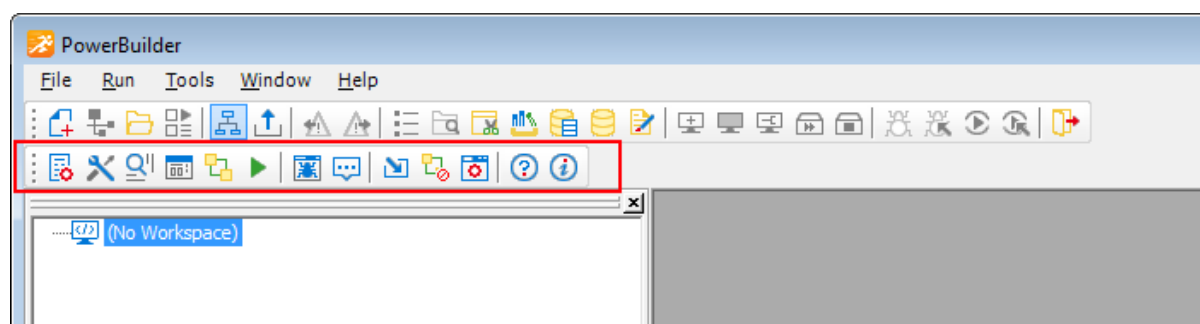
PowerServer Mobile is a deployment tool & runtime environment that allows you to deploy the PowerBuilder applications as pure native mobile applications which can run on various mobile platforms including iOS, Android etc.

PowerServer Toolkit deployment tool

PowerServer Toolkit provides a set of tools to configure and deploy the PowerBuilder application to be the native mobile application. The PowerServer Toolkit deployment tool exports and analyzes the PowerBuilder libraries (PBL files), converts PowerScript to C/C++ scripts, and uploads the script files to the PowerServer.

PowerServer Toolkit must be installed on top of PowerBuilder. The PowerServer Toolkit will be loaded into the PowerBuilder PowerBar automatically when PowerBuilder starts.

Figure 1.12: PowerServer Toolkit



For information on PowerServer Toolkit, see PowerServer Toolkit User Guide.

PowerServer runtime services

After deployed, the mobile application interacts with the PowerServer for the run-time services such as data connectivity, DataWindow support, transaction management, security, and the offline access and data sync supports. PowerServer must be installed to a .NET/IIS application server or a Java application server such as JBoss, JEUS, WebLogic, WebSphere etc.

PowerServer data sources

The deployed mobile application accesses the database thru the data source created in the PowerServer. The PowerServer data source is the counterpart to the transaction object in the

PowerBuilder application. The transaction properties in the PowerBuilder application contain database connection parameters (including database name, login ID, password etc.), which should be correspondingly configured in the PowerServer data source.

For information on PowerServer data sources, see PowerServer Configuration Guide for .NET or PowerServer Configuration Guide for J2EE.

Appeon Workspace running platform

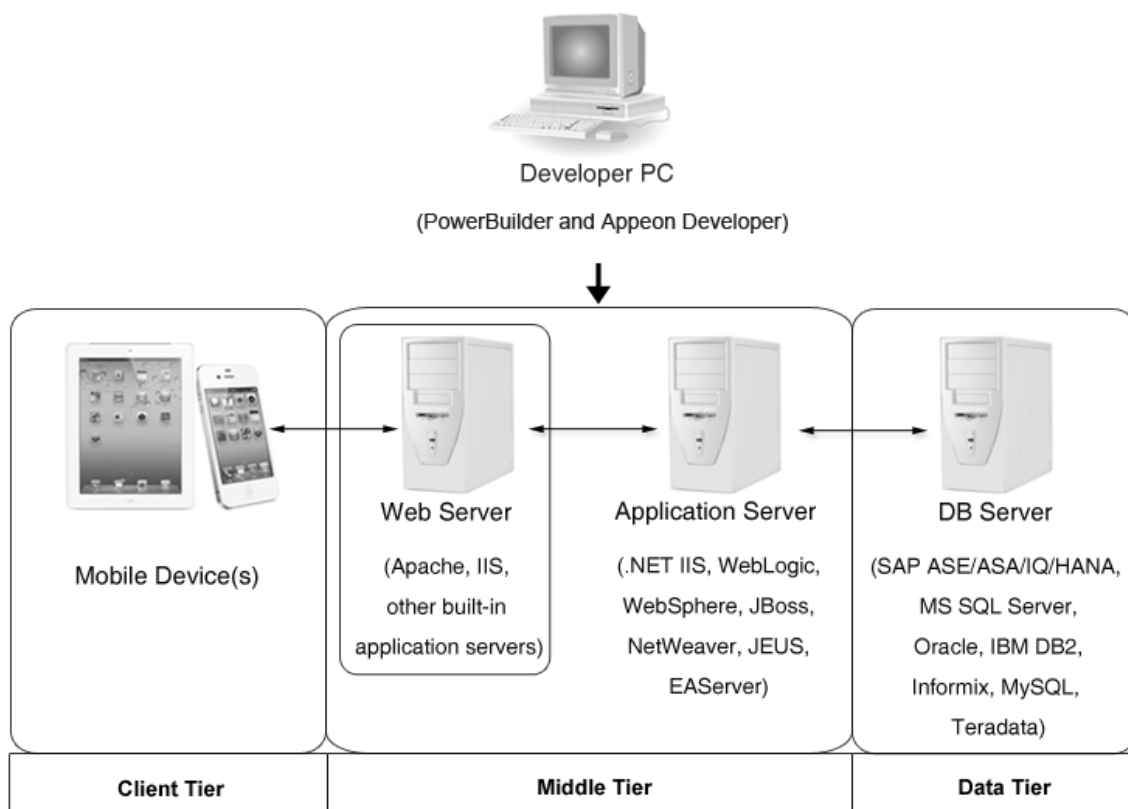
The mobile application can be immediately run in Appeon Workspace after deployed, without needing to be packaged as a standalone app first. Appeon Workspace is a running platform which consists of a set of mobile client libraries and a graphical user interface. The mobile client libraries render the mobile-style UI, support the UI logic, provide interfaces for calling the mobile SDK, and support accessing the client/offline database and running the offline mobile application. The graphical user interface allows end users to easily install and run the mobile application from PowerServer.

For information on Appeon Workspace, see Appeon Workspace User Guide.

1.2.2 The PowerServer Mobile n-tier architecture

The mobile application is running on standard n-tier architecture - the client tier (presentation layer), middle tier (business logic layer), and data tier are logically separated.

Figure 1.13: n-tier architecture



Client Tier (Appeon Workspace)

Appeon Workspace consists of a set of mobile client libraries and a graphical user interface that provides an out-of-box running platform for the mobile application.

Appeon Workspace is a free native mobile application which can only be distributed internally or privately. It is also included as an invisible component when you package and compile the PowerServer mobile application as the iOS application archive (IPA) file or Android application package (APK) file.

Middle Tier (Application Server)

The middle tier, which hosts the business logic, is implemented with J2EE or .NET components. These components execute the DataWindows and any Embedded SQL and host the business logic of the mobile application. The middle tier deploys to the Java or .NET compliant PowerServer, leveraging dozens of many-years of investments in DataWindows and other business logic.

Data Tier (Database)

The database stores the raw data for the application and stored procedures, if any. The existing database from the PowerBuilder application can be simply re-used without modification.

1.3 About this tutorial

About this section

This section describes what you will do in this tutorial and how to get set up for it. This tutorial is divided into 3 parts.

1.3.1 Learning to install PowerBuilder and PowerServer Mobile

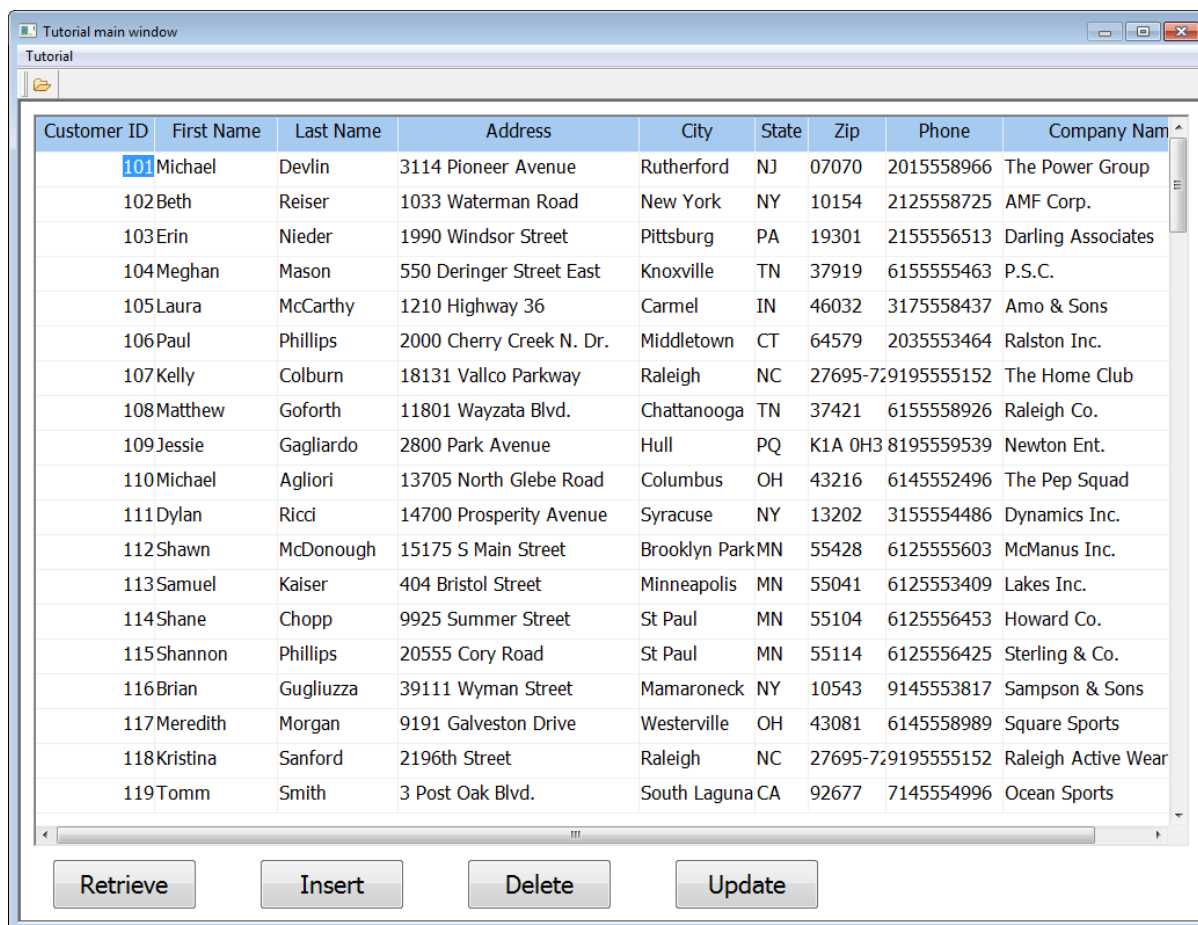
In this first part, you learn to set up the environment and install the required software.

1.3.2 Learning to build a mobile application

This second part is a set of exercises in which you build a mobile Multiple Document Interface (MDI) database application. The application allows you to retrieve customer information from the Appeon Sample database and perform insert, delete, and update functions against the customer data.

The mobile MDI application opens a sheet window that provide access to the Customer tables in the Appeon Sample database.

Figure 1.14: Customer window



1.3.3 Learning to deploy a mobile application

In this third part, you learn to configure and deploy a mobile application using PowerServer Toolkit, and run the deployed application in Apeon Workspace on an Android device.

Figure 1.15: Customer window in Apeon Workspace

iPad 下午6:14 11%

Trial Edition - Tutorial main window

Customer ID	First Name	Last Name	Address	City	State	Zip	Phone	Company Name
101	Michael	Devlin	3114 Pioneer Avenue	Rutherford	NJ	07070	2015558966	The Power Group
102	Beth	Reiser	1033 Waterman Road	New York	NY	10154	2125558725	AMF Corp.
103	Erin	Nieder	1990 Windsor Street	Pittsburg	PA	19301	2155556513	Darling Associates
104	Meghan	Mason	550 Deringer Street East	Knoxville	TN	37919	6155555463	P.S.C.
105	Laura	McCarthy	1210 Highway 36	Carmel	IN	46032	3175558437	Amo & Sons
106	Paul	Phillips	2000 Cherry Creek N. Dr.	Middletown	CT	64579	2035553464	Ralston Inc.
107	Kelly	Colburn	18131 Valco Parkway	Raleigh	NC	27695-7	9195555152	The Home Club
108	Matthew	Goforth	11801 Wayzata Blvd.	Chattanooga	TN	37421	6155558926	Raleigh Co.
109	Jessie	Gagliardo	2800 Park Avenue	Hull	PQ	K1A 0H3	8195559539	Newton Ent.
110	Michael	Agliori	13705 North Glebe Road	Columbus	OH	43216	6145552496	The Pep Squad
111	Dylan	Ricci	14700 Prosperity Avenue	Syracuse	NY	13202	3155554486	Dynamics Inc.
112	Shawn	McDonough	15175 S Main Street	Brooklyn Park	MN	55428	6125555603	McManus Inc.
113	Samuel	Kaiser	404 Bristol Street	Minneapolis	MN	55041	6125553409	Lakes Inc.
114	Shane	Chopp	9925 Summer Street	St Paul	MN	55104	6125556453	Howard Co.
115	Shannon	Phillips	20555 Cory Road	St Paul	MN	55114	6125556425	Sterling & Co.
116	Brian	Gugliuzza	39111 Wyman Street	Mamaroneck	NY	10543	9145553817	Sampson & Sons
117	Meredith	Morgan	9191 Galveston Drive	Westerville	OH	43081	6145558989	Square Sports
118	Kristina	Sanford	2196th Street	Raleigh	NC	27695-7	9195555152	Raleigh Active Wear
119	Tomm	Smith	3 Post Oak Blvd.	South Laguna	CA	92677	7145554996	Ocean Sports
120	Gertrude	Stein	4 Amon Carter Blvd.	Elmsford	NY	10523	9145553476	Carney Co.

Retrieve Insert Delete Update

2 Installing PowerBuilder & PowerServer

2.1 Setting up the environment

2.1.1 Prepare the environment

The simplest scenario will be used in this guide, which requires 1 Windows PC and 1 mobile device.

- **1 Windows PC:** used as the development machine and the server
- **1 Android or iOS device:** in this tutorial, an Android device is used as the mobile client

The **PowerBuilder 2019 R2** version and the 32-bit version of **PowerServer Mobile for .NET** edition will be used to walk you through this guide, so please prepare the environment according to the following requirements.

Software requirements for **Windows PC**:

- Windows 7 or 8.1 (32-bit or 64-bit)
PowerServer Mobile 32-bit version can be installed to the 64-bit OS without any special considerations.
- .NET Framework 4.x
For Windows 7, you will need to first download the **.NET Framework 4.0** setup program from <http://www.microsoft.com/en-us/download/details.aspx?id=17718>.
- IIS
See [Install IIS](#) and [Configure IIS](#).
- PowerBuilder 2019 R2 & PowerServer Mobile 2020 (32-bit)
You can download **PowerBuilder 2019 R2 CloudPro Edition** from the Apeon web site. The universal edition includes the installation of PowerBuilder, PowerServer Mobile, and PowerServer Toolkit.
See [Install PowerBuilder and PowerServer](#) for instructions on installing PowerBuilder, PowerServer Mobile, and PowerServer Toolkit.

Hardware requirements for **Windows PC**:

- Intel processor(s) running at 1.8 GHz or faster
- At least 2 GB RAM (4 GB RAM recommended)
- 4 GB hard drive space (for PowerBuilder installation and PowerServer Mobile installation)

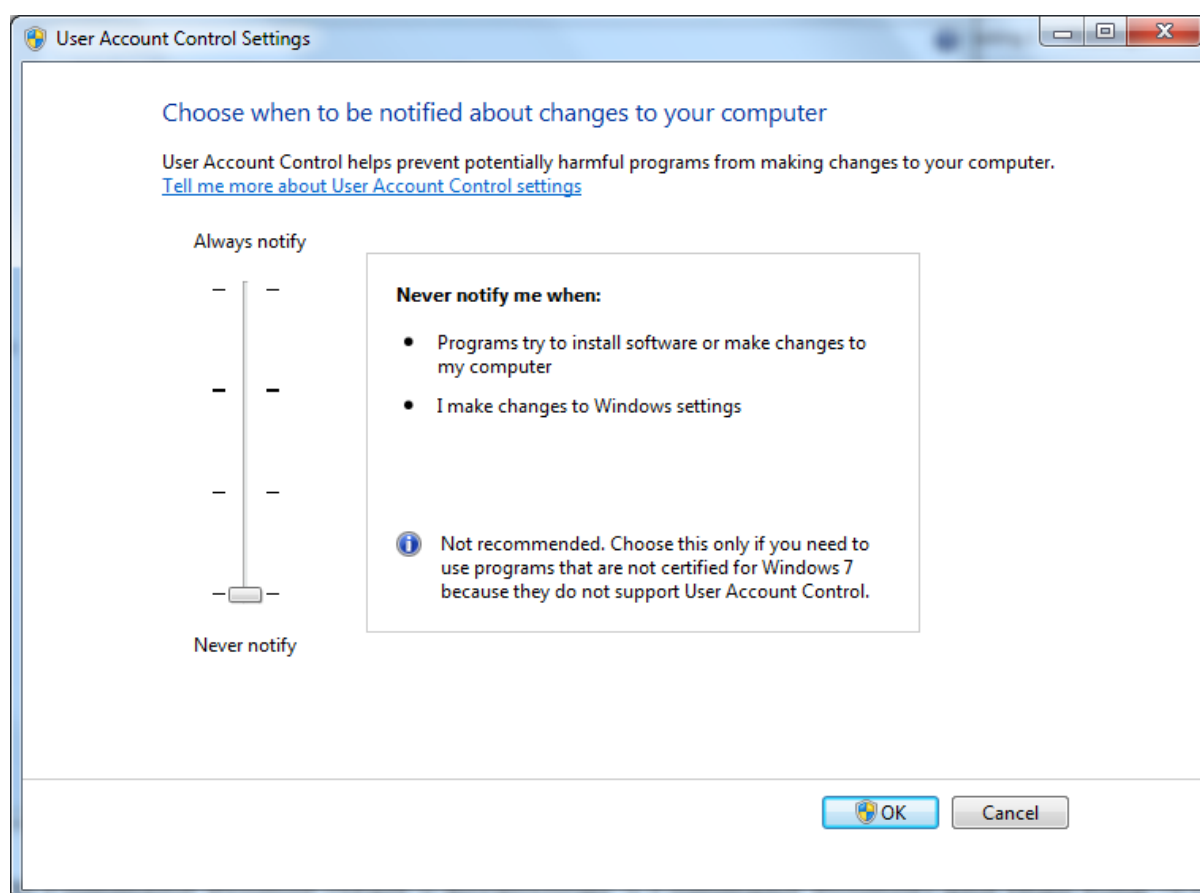
Software requirements for the Android device:

- Google Android 6.x (32-bit), 7.x, 8.x, 9.x, or 10.x
 - Appeon Workspace
- See [Install Appeon Workspace \(on the Android device\)](#)

2.1.2 Disable UAC (User Account Control)

On the Windows PC, go to the **Control Panel > User Accounts** and disable UAC by setting the slider to "**Never notify**" (see screenshot below). After you have disabled UAC you **MUST** restart the computer. If you do not restart the computer the change will **NOT** take effect.

Figure 2.1: Disable UAC



2.1.3 Install IIS

IIS is not installed on Windows 7 or 8.1 by default. You need to manually install it. Before you install IIS make sure you have installed .NET Framework 4.x, for Windows 8.1 please go to the **Turn Windows features on or off** and check the box for **.NET Framework 4.5**; for Windows 7, please download the setup program from <http://www.microsoft.com/en-us/download/details.aspx?id=17718> and install .NET Framework 4.0.

Below are steps for installing IIS.

Step 1: Click **Start** and then click **Control Panel**. In Control Panel, click **Programs** and then click **Turn on or off Windows features**.

Step 2: Select the check box of **Internet Information Services**, then expand the list and select the items under **Web Management Tools**, **Application Development Features** and **Common HTTP Features** according to the figure below. Click **OK** to let Windows finish the install.

Figure 2.2: Select the Web Management Tools

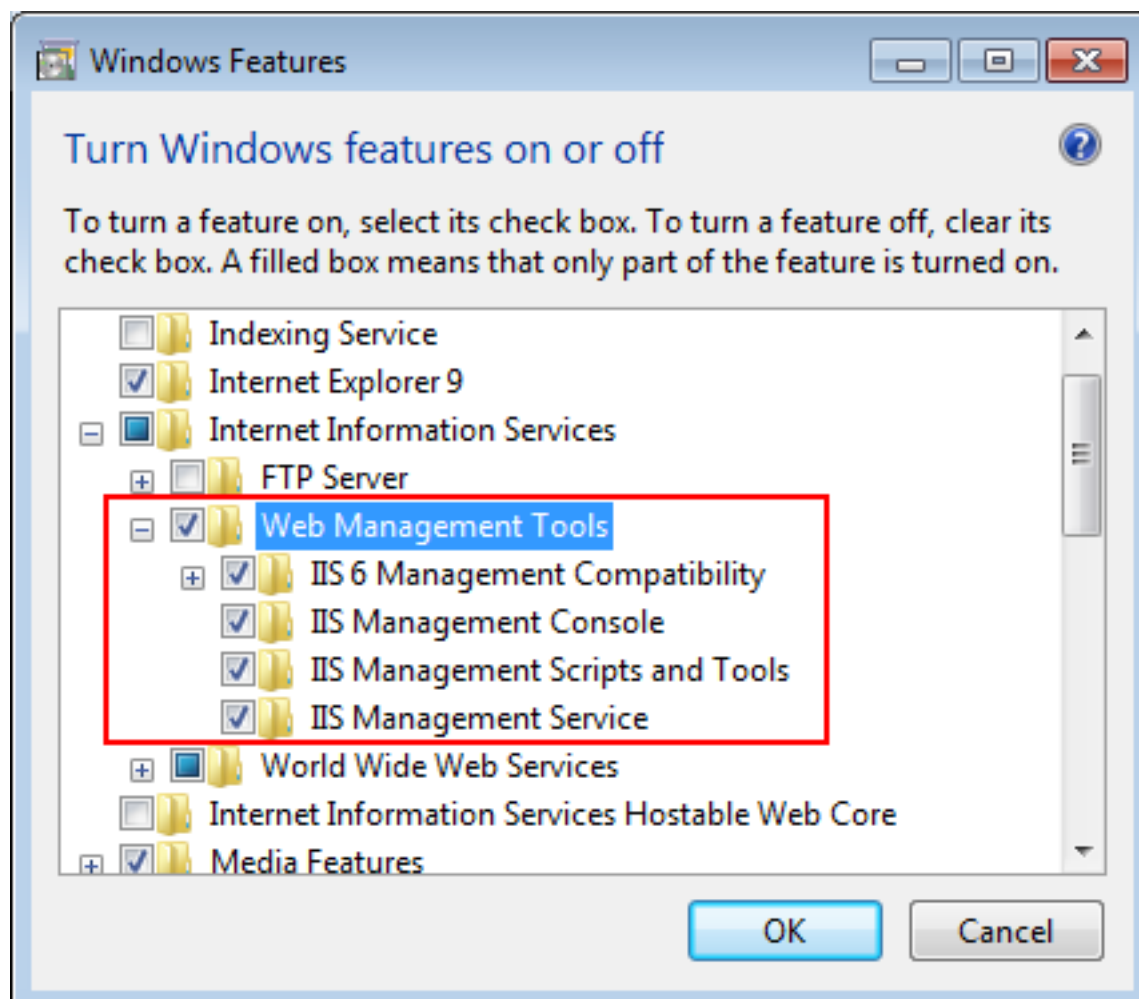


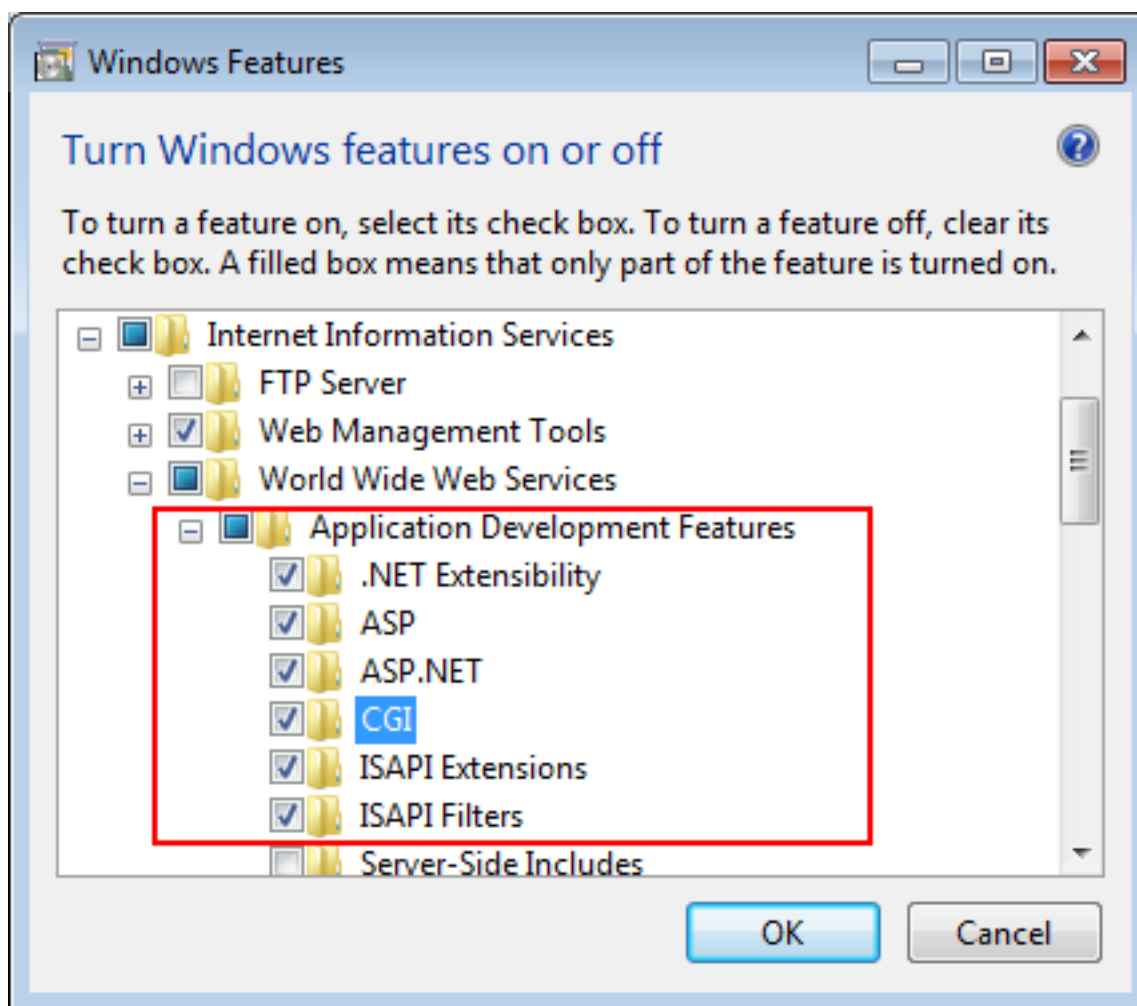
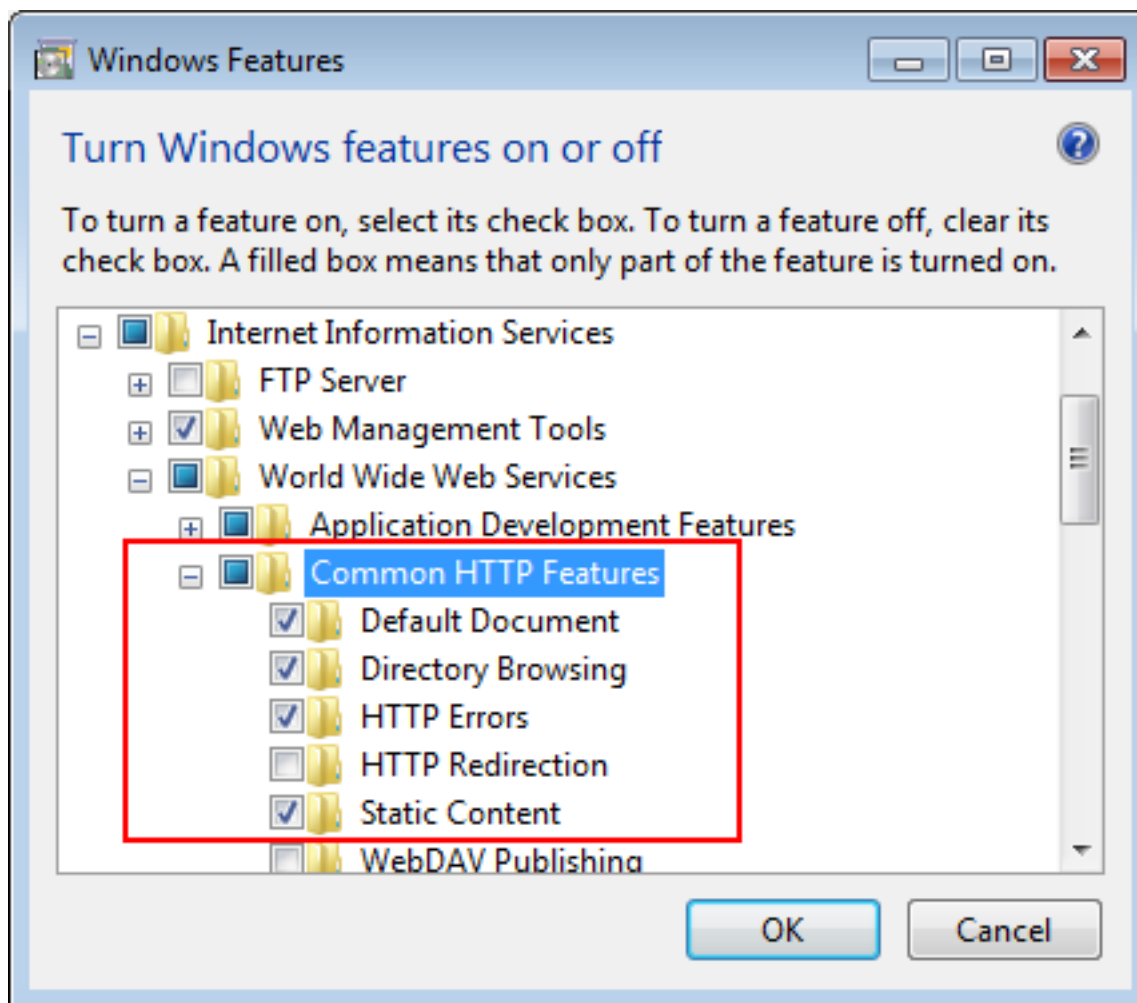
Figure 2.3: Select the Application Development Features

Figure 2.4: Select the Common HTTP Features

Step 3: After IIS is installed, go to **IIS Manager**, right click the **Default Web Site**, select **Binding**, and make sure **IP Address** is indicated with an asterisk "*". If not, please click **Edit** and select **All Unassigned** for the IP address, this will display **IP Address** as an asterisk "*".

Step 4: Run `http://IP_Address:80/` in Internet Explorer. If the IIS welcome screen displays, then IIS is working properly.

TIP: to obtain the IP address of the server, open a command prompt window and then type `ipconfig`<Enter>. Remember this IP address as it is also needed when you configure the mobile app in Apeon Workspace.

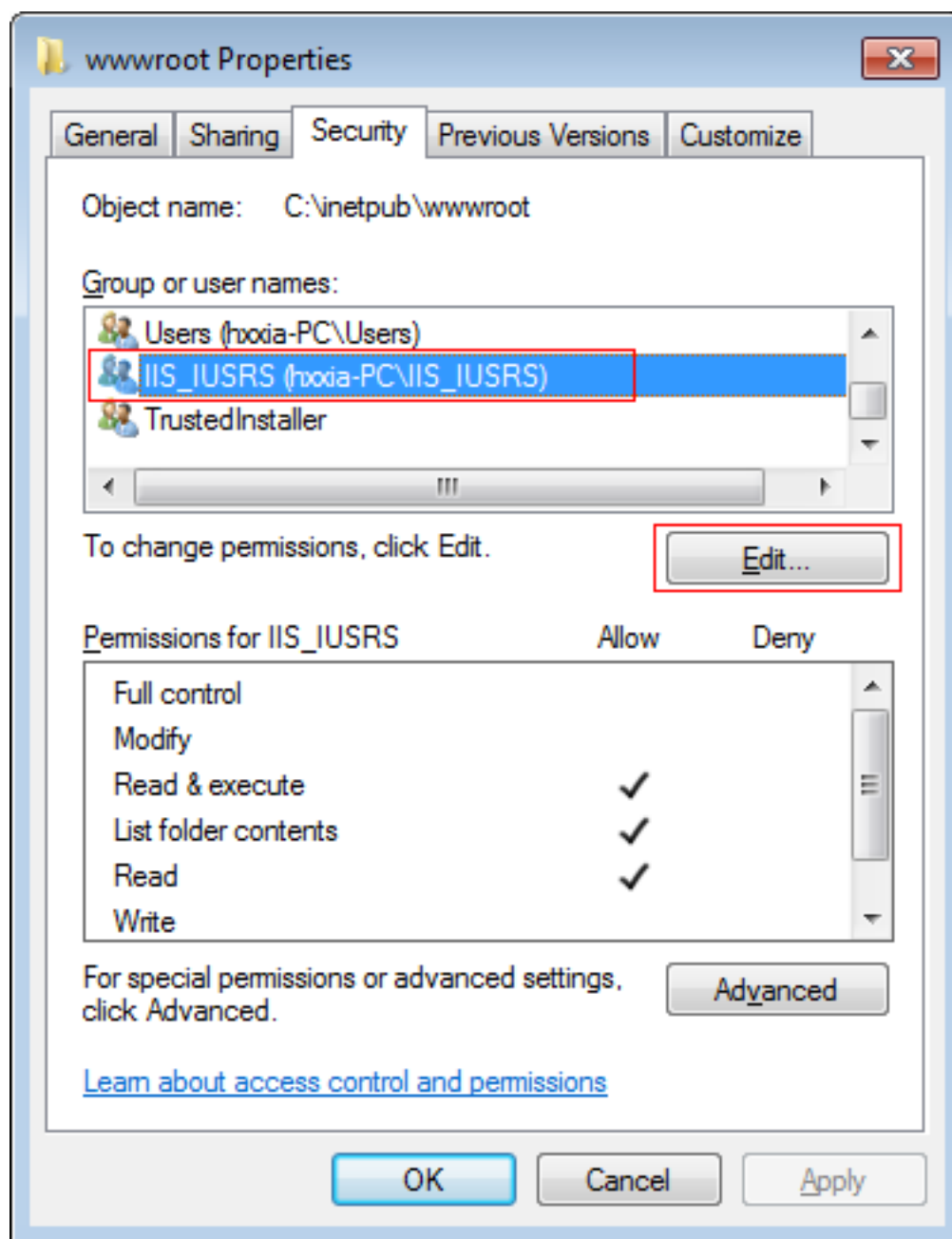
If IIS is not working, please re-install IIS or fix the IIS configuration by following the IIS help.

2.1.4 Configure IIS

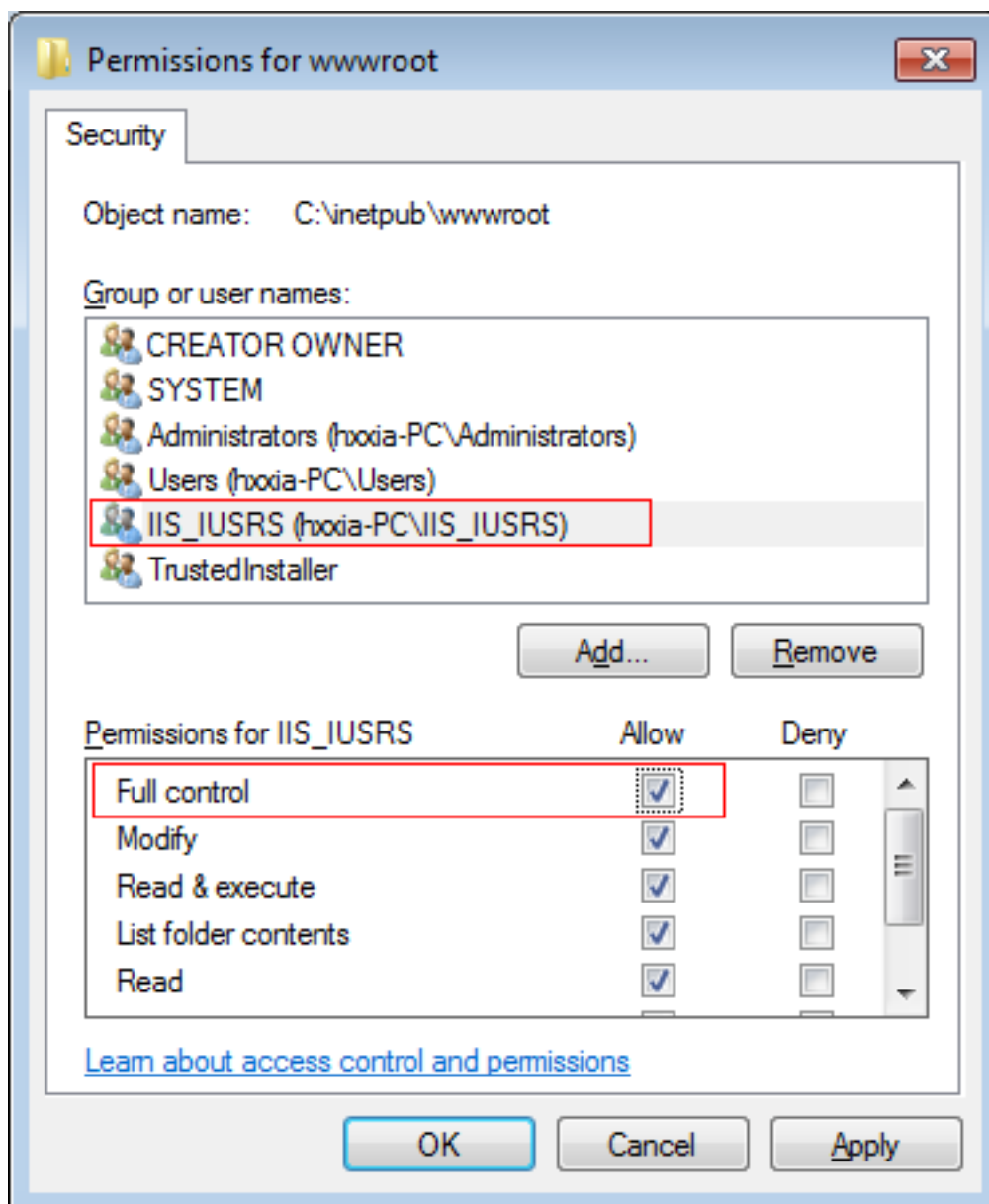
Follow steps below to grant **IIS_IUSRS** account with full controls to the IIS Web root folder:

Step 1: Right click the `C:\inetpub\wwwroot` folder and select **Properties** from the popup menu.

Step 2: On the **Security** tab, select **IIS_IUSRS** in the **Group or user names** list box, and then click the **Edit** button.

Figure 2.5: Select IIS_IUSRS

Step 3: Select **IIS_IUSRS** in the **Group or user names** list box, and then select the **Allow** check box for the **Full control**.

Figure 2.6: Select Full control for IIS_IUSRS

Step 4: Click **OK** to save the settings.

2.2 Installing PowerBuilder and PowerServer

2.2.1 Installing PowerBuilder and PowerServer

After you download **PowerBuilder 2019 R2 CloudPro Edition** from the Apeon web site, extract the installation package and start to install PowerBuilder and PowerServer by the following steps.

Step 1: Start IIS server: open IIS Manager, right click the **top** node (not the website node) in the treeview and select **Start** from the popup menu.

Step 2: Double click **Autorun.exe** in the PowerBuilder CloudPro Edition installation package. The PowerBuilder 2019 R2 Setup program starts.

Step 3: Click **Install** on the start page and then click **Next** on the welcome page.

Step 4: On the License Agreement page, accept the terms in the license agreement page and click **Next**.

Step 5: On the Customer Information page, enter your name and your company's name and click **Next**.

Step 6: On the Choose Destination Location page, click **Next** to use the destination path shown.

Step 7: On the Destination Location for Shared Files page, click **Next** to use the destination path shown.

Step 8: On the Specify the SQL Anywhere Engine page, select the location of a SQL Anywhere engine for running the demo database.

Step 9: On the Select Products page, select **PowerBuilder IDE**, **PowerServer Mobile**, and **PowerServer Toolkit** and click **Next** to continue.

The **PowerBuilder** installation starts first.

1. On the Select Components page, unselect any components you do not want to install, and click **Next**.
2. On the Select Program Folder page, select the program folder to which program icons will be added, or specify a new one; then click **Next**.
3. On the Start Copying Files page, review your settings, then click **Next** to begin installing files for PowerBuilder.

The **PowerServer Mobile** installation starts automatically when the PowerBuilder installation is completed.

1. Click **Next** until you reach the screen for specifying the IIS Web site where PowerServer Mobile will be installed. Make sure **Select an existing Web Site** and **Default Web Site** are selected.
2. Click **Next**. Choose whether to install the **Demo Applications**. If you have SQL Anywhere database server installed, select to install the demo; otherwise, do not install the demo since the demo requires SQL Anywhere database server to be previously installed.
3. If you selected to install the **Demo Applications**, specify the path for the SQL Anywhere database server engine.
4. Click **Next** until Setup begins installing files for PowerServer Mobile.

The **PowerServer Toolkit** installation starts automatically when the PowerServer Mobile installation is completed.

1. Click **Next** until Setup begins installing files for PowerServer Toolkit.

Step 10: Click **Finish** when the PowerServer Mobile and the PowerServer Toolkit installations are completed.

Step 11: After closing all the windows during installation, the **System Reboot Required** screen is displayed. Select the **Yes, I want to restart my computer now** checkbox to reboot the system and then click **Finish**.

2.2.2 Verify the PowerServer installation

On the Windows PC, run `http://IP_Address:80/AEM/`. If AEM is launched successfully, then PowerServer is installed successfully.

2.2.3 Install Appeon Workspace (on the Android device)

Step 1: Make sure your Android device can connect to PowerServer.

Step 2: Enable the **Unknown resources** option (in **Settings > Security**) on the Android device so you can install apps that are not downloaded from Google Play.

Step 3: Visit the Appeon Workspace download center that is posted on PowerServer (`http://server_domain:80/AWS`), and then click the download button.

2.2.4 Configure the network connection

Check and make sure the Windows PC and the Android device connected to the same Wi-Fi router. To verify this, on the Android device, open the Web browser and type `http://server_domain:80/AEM/`. If AEM is launched successfully, it means that the PowerServer is properly installed and that the Android device is able to connect to the Windows PC.

NOTE: Mobile Internet is supported by PowerServer Mobile. However, in order for the Appeon Workspace to connect to your PowerServer you will need an external IP address. You can verify that your external IP address is properly working by typing `http://External_IP_Address:80` into a Web browser of any device connected to the Internet. If you get a page not found error or other HTTP error then your network is not configured properly for external access.

3 Building a Mobile Application

3.1 Starting PowerBuilder

This section provides the information you need to start PowerBuilder and create an application.

In this section you:

- [Create a new workspace](#)
- [Create a target](#)

3.1.1 Create a new workspace

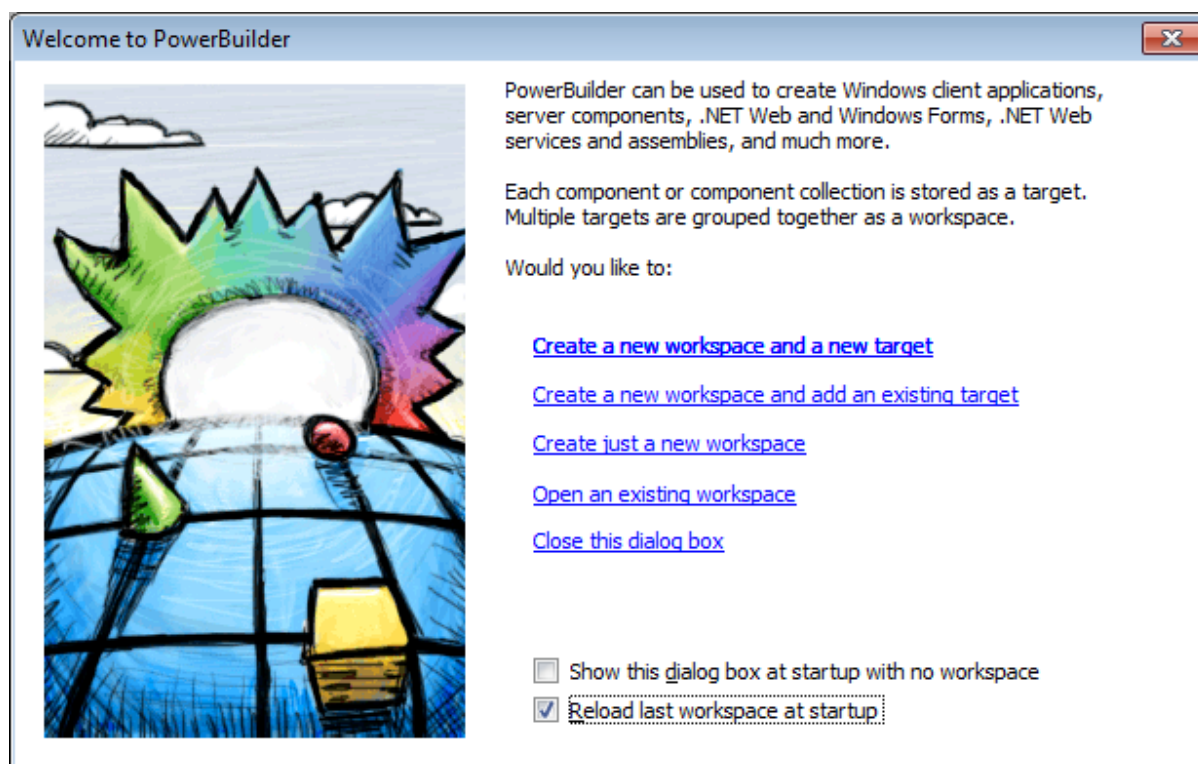
The workspace is where you build, edit, debug, and run PowerBuilder targets. You can build several targets within a single workspace.

Now you start PowerBuilder and create a new workspace.

1. Select **All Programs > Apeon > PowerBuilder 2019 R2 > PowerBuilder 2019 R2** from the Windows **Start** menu.

The **Welcome to PowerBuilder** dialog box is displayed.

Figure 3.1: Welcome page



The **Welcome to PowerBuilder** dialog box allows you create a new workspace and add a new target or an existing target to the workspace.

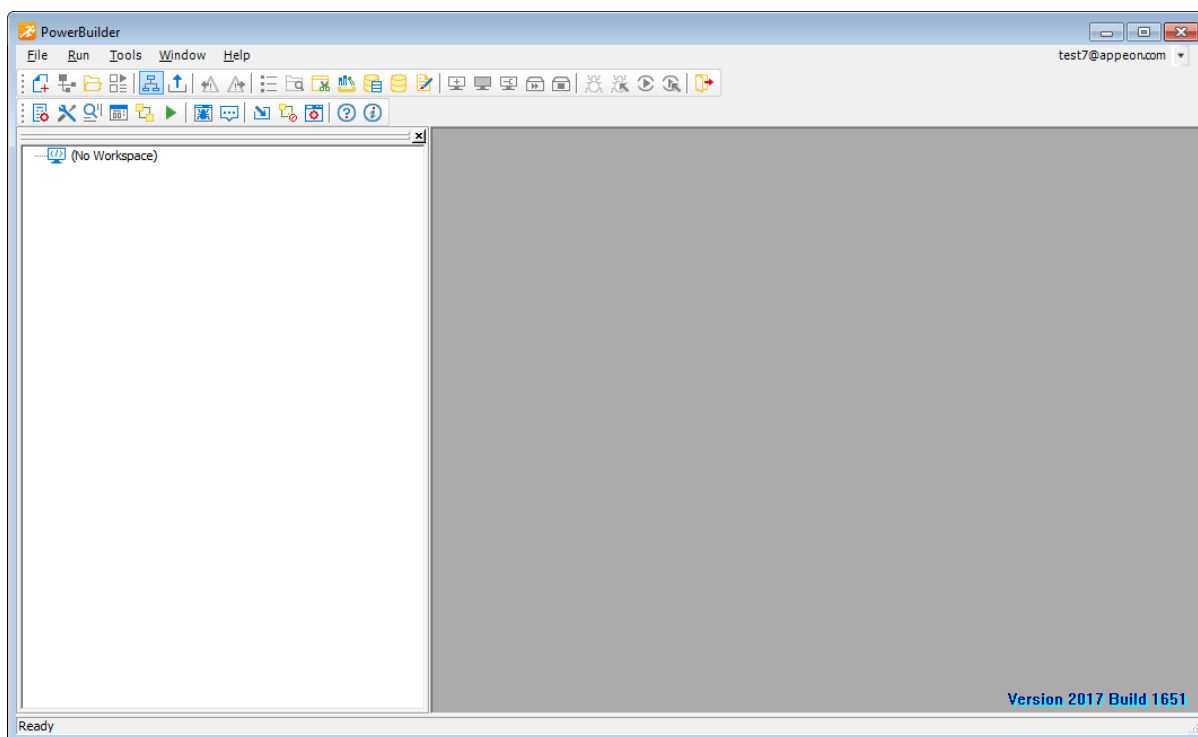
If you do not want PowerBuilder to display the dialog box again

You can leave the **Show this dialog box at startup with no workspace** checkbox unchecked to keep PowerBuilder from displaying the welcome dialog box every time you start PowerBuilder. Select the **Reload last workspace at startup** checkbox to load the most recently used workspace each time you start a PowerBuilder session. Then click **Close this dialog box**.

The PowerBuilder development environment is displayed.

If this is the first time you are opening PowerBuilder on your machine, you see only a top-level entry in the System Tree to indicate that no workspace is currently open. Otherwise, the System Tree might show a workspace with targets and objects in it.

Figure 3.2: PowerBuilder development environment



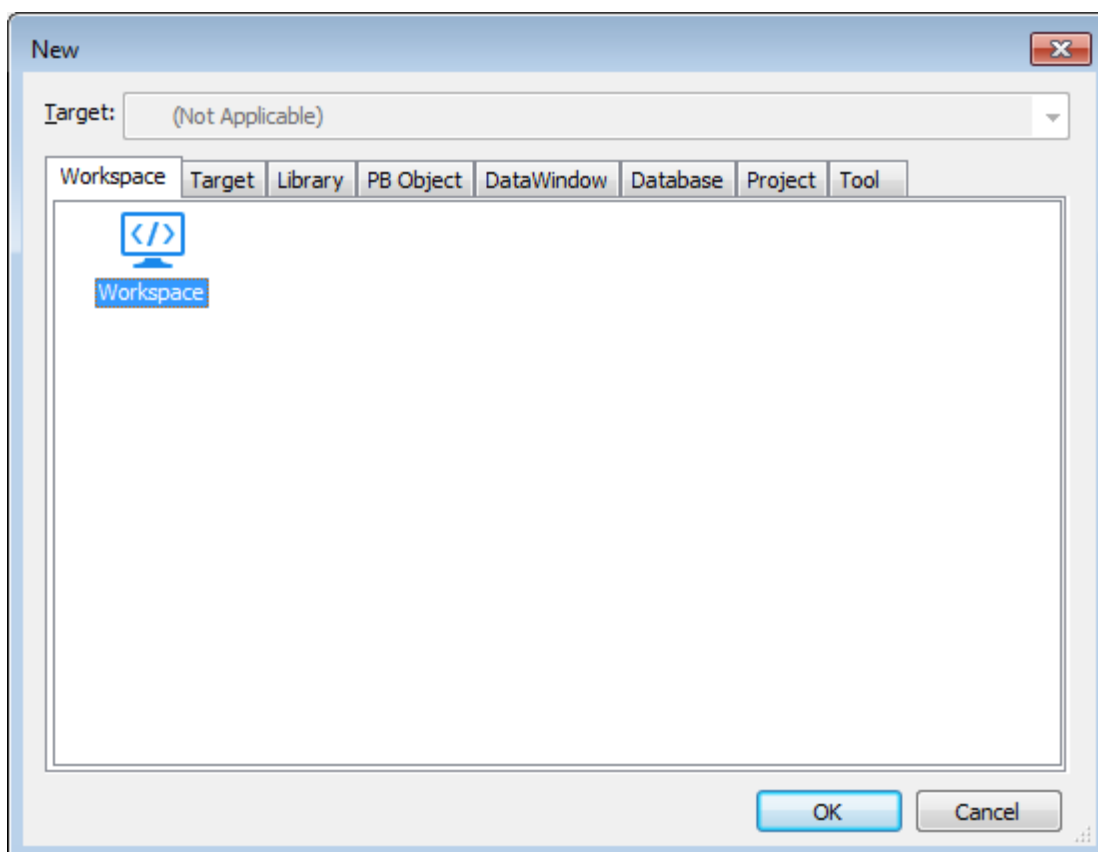
2. Select **New** from the **File** menu

or

Click the **New** button in the PowerBar.

The **Workspace** page of the **New** dialog box is displayed.

PowerBuilder displays the page of the **New** dialog box that was used before the dialog box was last closed. In this exercise, make sure that the **Workspace** page of the **New** dialog box is displayed.

Figure 3.3: Workspace tab on the New dialog box

3. Select **Workspace** from the **Workspace** page of the **New** dialog box. Click **OK**.

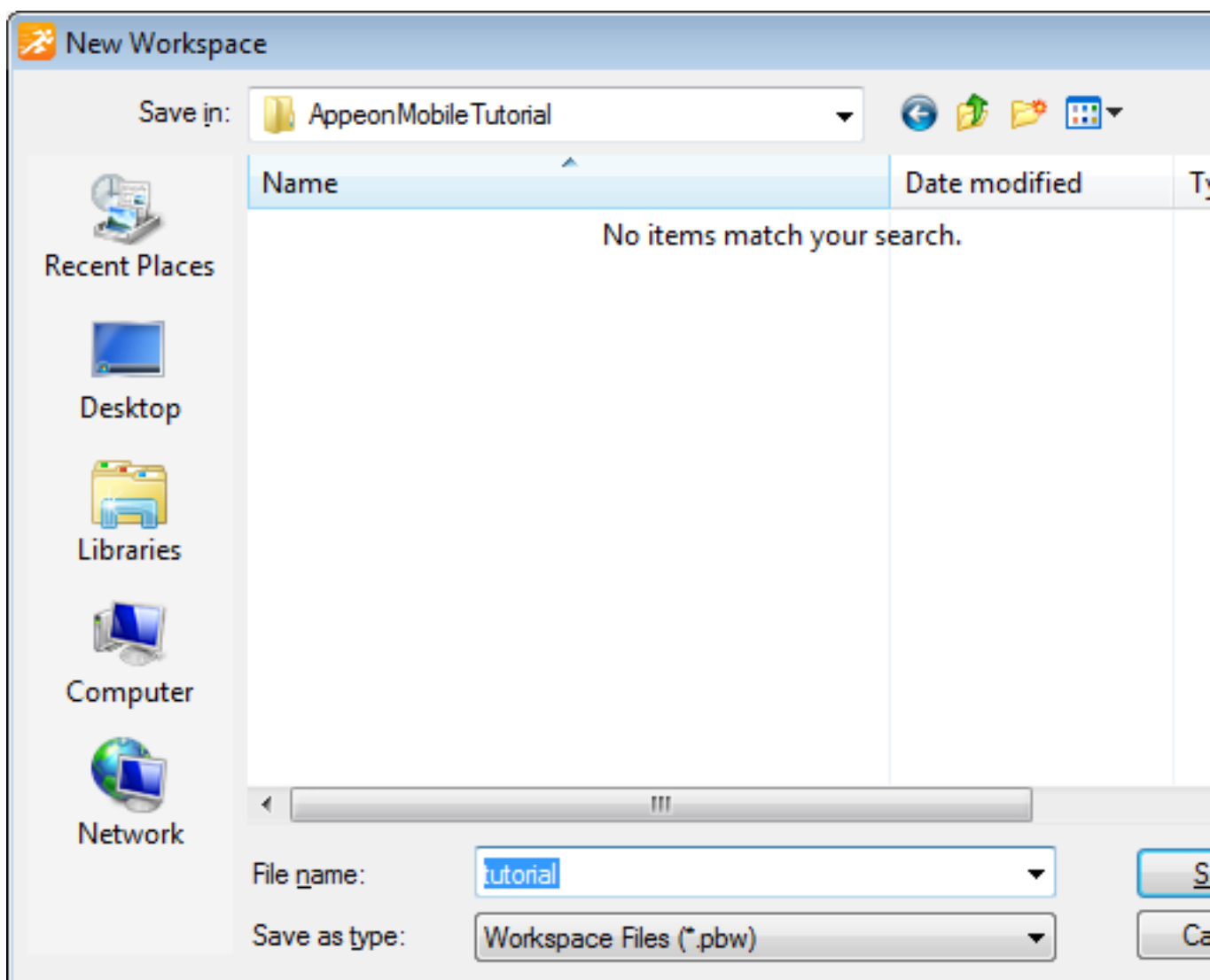
The **New Workspace** dialog box is displayed.

4. Choose the tutorial folder.

If you have created a workspace before, the dialog opens to the location of the most recently-used workspace. For this new workspace, change the location to the path described in the next paragraph.

If this is your first workspace, open the **New Workspace** dialog box to `C:\Users\Public\Documents` on Windows 7 and Windows 8.1. Navigate from this point to `Appoon\Toolkit\appeondemo` and create the **AppoonMobileTutorial** folder.

5. Type *tutorial* in the **File name** text box.

Figure 3.4: Specify the name for the workspace

6. Click **Save**.

The **New Workspace** dialog box closes and the workspace you created appears as the first item in the System Tree.

3.1.2 Create a target

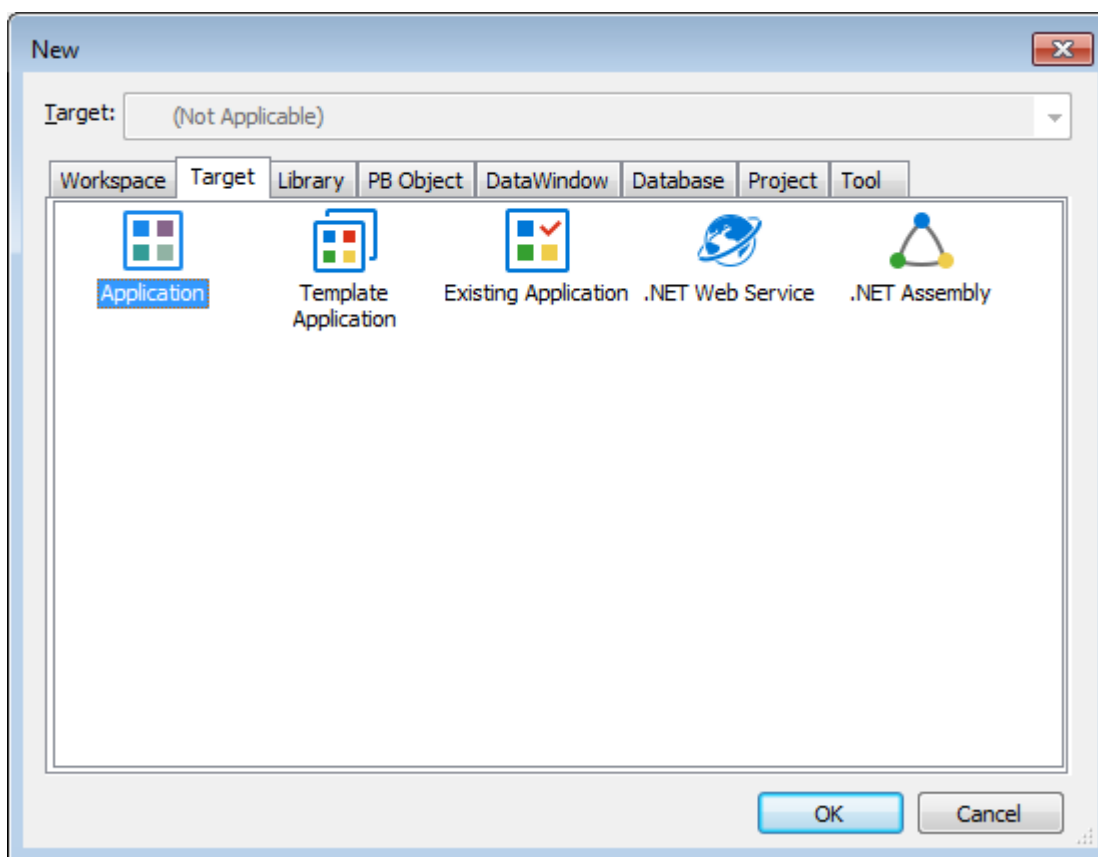
Now you create a new target.

1. Select **New** from the **File** menu and click the **Target** tab

or

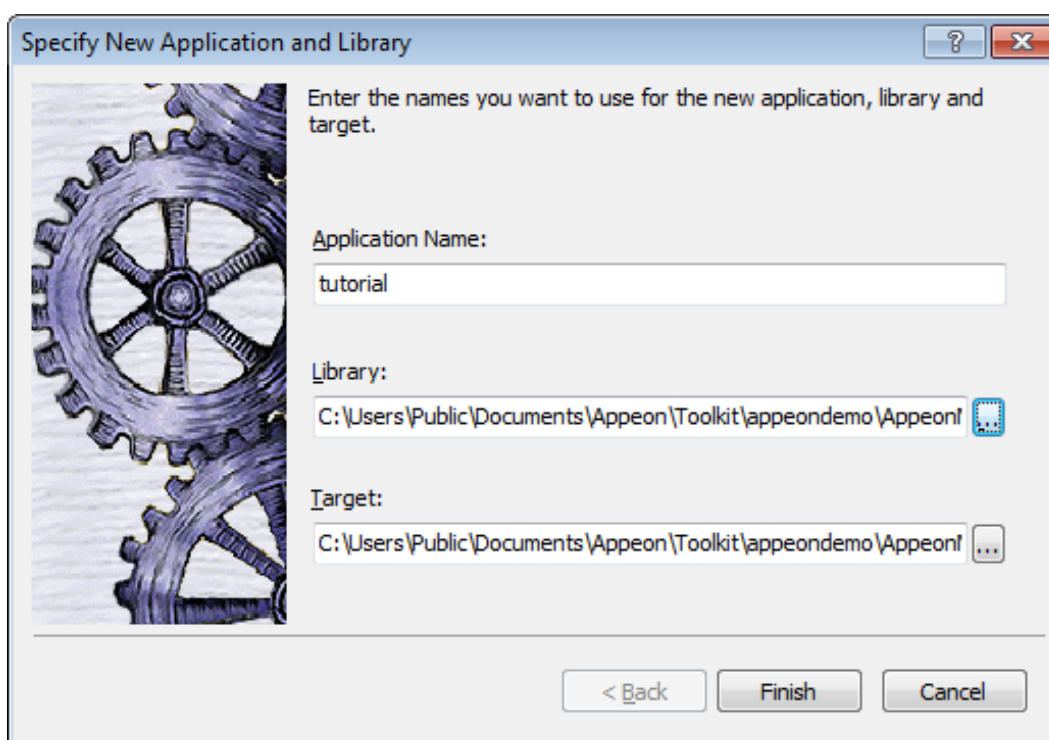
Right-click **tutorial** in the System Tree, select **New** from the pop-up menu, and click the **Target** tab.

The **Target** page of the **New** dialog box is displayed.

Figure 3.5: Create an application

2. Select the **Application** icon and click **OK**.

The **Specify New Application and Library** page is displayed, and you can press **F1** to get Help on most fields.

Figure 3.6: Specify the new application and library

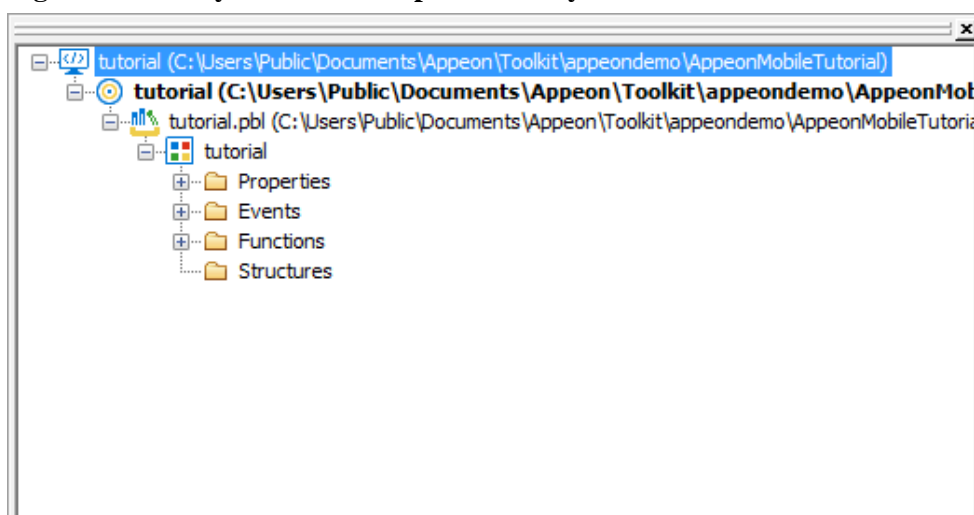
3. Type *tutorial* in the **Application Name** text box.

The file names will be automatically assigned to the library and target that use this application name. It assigns the library a PBL extension and the target a PBT extension.

4. Click **Finish**.

5. You can expand the System Tree to view the Workspace and Target that have been newly created. The System Tree does not display the file extension of the tutorial target, but it does display the directory where the target file is saved.

The tutorial.pbl library is displayed under the tutorial target in the System Tree. It contains the target Application object, which has the same name as the target object but is displayed under the library file.

Figure 3.7: Newly created workspace in the system tree

3.2 Customizing the PowerBuilder Environment

This section provides the information you need in order to become familiar with the PowerBuilder environment and to customize the workspace. This section is optional -- you can skip to the next section if you want to.

In this section you:

- [Manipulate the System Tree window](#)
- [Open an object](#)
- [Manipulate views](#)
- [Set up the toolbars](#)

3.2.1 Manipulate the System Tree window

The Workspace page in the System Tree provides you with an overview of your work. By expanding the workspace and the objects it contains, you can see the content and structure of your target.

You can work directly with all the objects in the workspace. For example, you can edit, run, search, or regenerate a window using its pop-up menu in the System Tree. In this exercise you reposition, close, and open the System Tree. You can reposition the System Tree in relation to the main window using its drag bar. You can also change the way the System Tree, Clip, and Output windows are arranged.

1.



Click the **Output** () button in the PowerBar1 to display the **Output** window.

2. Select **Tools > System Options** from the menu bar. Clear the **Horizontal Dock Windows Dominate** checkbox on the **General** page and click **OK**.

The System Tree and **Clip** windows now occupy the full height of the main window.

3. Click and hold the drag bar at the top of the System Tree. Drag the System Tree to position it above, below, or to the right of the painter workspace.

The painter workspace is the gray (blank) area, initially to the right of the System Tree, where painters are displayed when you open an object.

When you start dragging the **System Tree**, a gray rectangular outline is displayed. It indicates the area that the System Tree would occupy if you released the mouse button.

4. When the gray rectangular outline is positioned where you want the System Tree to be displayed, release the mouse button.

The System Tree is displayed in the new location.

- 5.



Close the System Tree by clicking the **SysTree** () button in the PowerBar1.

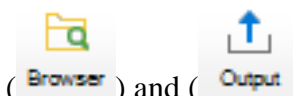
The current workspace remains open, but the System Tree closes. Closing the System Tree leaves more space for the painter workspace views.



6. Reopen the System Tree by clicking the **SysTree** button in the PowerBar1 again.

7. Select **Tools > System Options** from the menu bar. Select the **Horizontal Dock Windows Dominate** checkbox on the **General** page and click **OK**.

You change back to the default selection for this design-time property.

- 8.



Close the **Clip** and **Output** windows by clicking their buttons () and () on the PowerBar1 or by clicking the small *x* in the corner of each window.

9. Right-click **tutorial** workspace and select **Close** from the pop-up menu.

The workspace closes. No workspaces are displayed in the System Tree.

3.2.2 Open an object

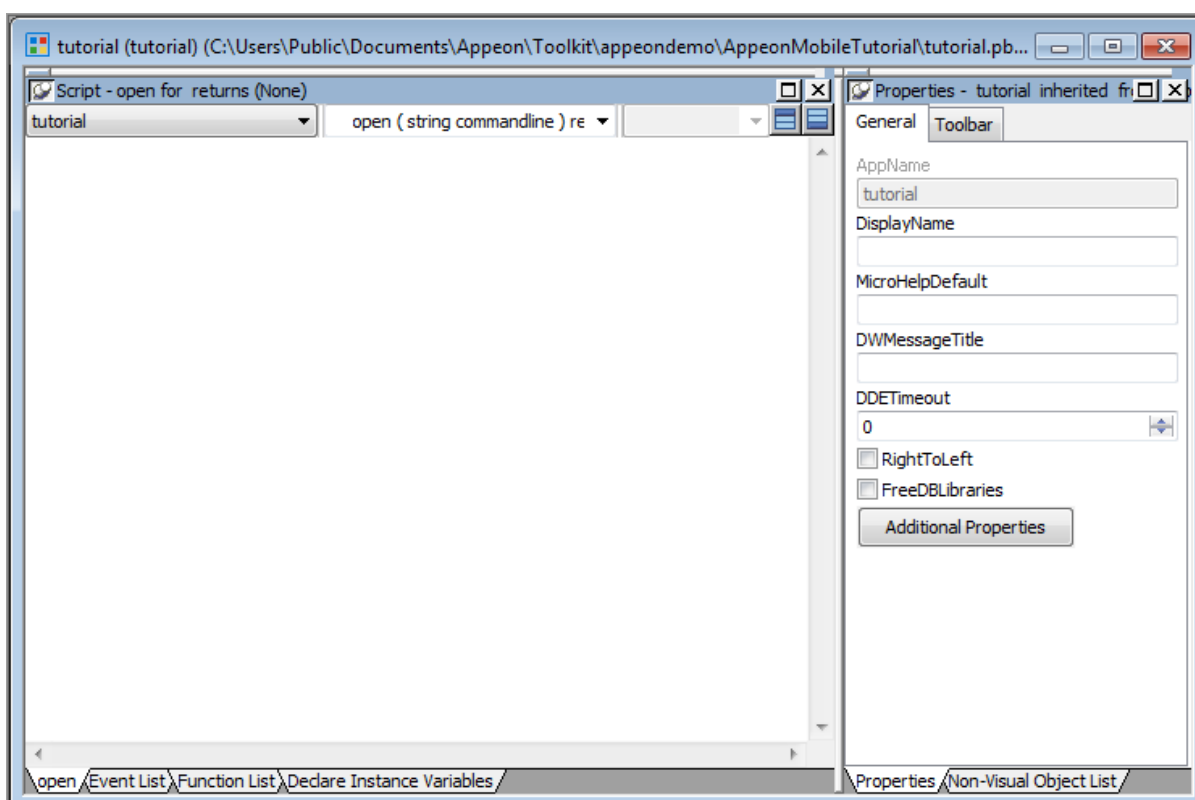
Now you open the object you just created.

1. Select **File > Recent Workspaces** from the menu bar, then **tutorial** workspace from the cascading menu.
2. In the System Tree, expand the **tutorial** workspace, the **tutorial** target, and **tutorial.pbl**.
3. Double-click the **tutorial** Application object

or

Right-click the **tutorial** Application object and select **Edit** from the pop-up menu.

The **Application** painter opens. It displays different views of the **tutorial** Application object. Your view layout scheme may look different. To display the default layout, select **View > Layouts > Default**.

Figure 3.8: Open an object

The default **Application** painter layout displays two stacks of tabbed panes. The left stack contains tabs for a **Script** view (**Open** tab -- it is set to the **Open** event on the **Application** object), an **Event List** view, a **Function List** view, and the **Declare Instance Variables** view. The right stack contains tabs for the **Properties** view and a **Non-Visual Object List** view.

4. Look at the code in the **Open** event in the **Script** view.

The **Application Object Open** event calls a PowerScript function to open the main window in the application.

3.2.3 Manipulate views

Now you learn to control the location and appearance of PowerBuilder painter views. You can add views to a painter workspace by selecting them from the **View** menu in the workspace menu bar.

You can add multiple views of the same type and you can combine views into a stack of panes with selection tabs at the bottom. You can resize a view by grabbing and dragging the separator bars that surround it or that surround neighboring views in the painter workspace.

These exercises demonstrate how you can change the appearance of **Application** painter views, but you can manipulate views in all painters in the same way.

Now you:

- [Add an extra Script view](#)

- [Display view title bars](#)
- [Float and dock views](#)
- [Manipulate tabbed views](#)
- [Save a view layout scheme](#)
- [Reset the default view layout scheme](#)

3.2.3.1 Add an extra Script view

The default Application painter layout actually has two Script views. One of the Script views displays the script for an Application object event, and the other Script view displays the declared variables for the object instance or the entire application. Both of these Script views are in the same stack of tabbed views (panes).

Now you add a third Script view that is not part of a stack of tabbed panes. You can add multiple Script views to your painter layout, but no two Script views can display the same script at the same time.

1. Select **View > Script** from the menu bar.

A new Script view is displayed. It is not attached to a stack of tabbed panes. It lists the Application object in the left drop-down list box. The other two drop-down lists are empty and the right drop-down list is grayed out.

If an existing Script view shows the Open event, the new Script view is empty. Otherwise it displays the Open event.


2. Select the **Close** event from the second drop-down list box.

If another Script view is already open to the Close event, an error message as "Script already opened in another script view" is displayed in the PowerBuilder status bar.

3.2.3.2 Display view title bars

Now you display a view title bar by pinning it to the painter workspace background. If a title bar is unpinned, you see it only when your cursor pauses near the top edge of a view.

1. Move the cursor to the top of the extra Script view you just added.

The view title bar rolls down. It contains a pushpin () button on the left and a **maximize/minimize** button and a close button on the right. The name of the view displays on the left side of the title bar, next to the pushpin button.

2. Click the **pushpin** in the title bar

or

Right-click the view title bar and click **Pinned** from the pop-up menu.

The pushpin button and the Pinned menu item are toggle switches. You can click the pushpin button or the pop-up menu item to pin and unpin the view title bars.

3.2.3.3 Float and dock views

Now you float and dock a view in the painter workspace. Floating a view enables you to move it around outside the painter frame.

1. Right-click the title bar of an unstacked view you want to float

or

Right-click the tab of a view in a stack of tabbed panes.

If the title bar is not pinned, move the cursor over the title bar area and wait until it is displayed before you right-click it.

2. Click **Float** in the pop-up menu.

When a view is floated, the **Float** menu item is not enabled. When a view is docked, the **Dock** menu item is not enabled.

3. Drag the view around the screen.

Notice that the floating property allows you to move the view outside the painter workspace.

4. Right-click the title bar of the floating view. Click **Dock** in the pop-up menu.

The view returns to its original location.

3.2.3.4 Manipulate tabbed views

Now you separate a view from a stack of tabbed panes and place it above the stack. You then return it to the stack and change its position in the stack.

1. Press and hold the **mouse** button on the **Function List** tab. Drag the tab onto the separator bar that separates the two default stacks in the Application painter. Release the **mouse** button.

When you release the **mouse** button, the Function List view is no longer part of a stack. If you drag the tab too far and release it over the right stack with the Properties view and Non-Visual Object List, the Function List becomes part of that stack.

Alternate way to float a view from a stack

If you hold the **Ctrl** or **Shift** key down as you drag a tabbed pane from a stack, the pane becomes a floating view.

2. Press and hold the **mouse** button on the Function List title bar. Drag it over the stack from which you separated it. Release the mouse button when the gray rectangular outline of the Function List view overlaps the stack.

The Function List view returns to its original stack, but it is added as the last pane in the stack.

3. Press and hold the **mouse** button on the Function List tab. Drag it sideways over the other tabs in the same stack. Release the mouse button when the small gray rectangular outline overlaps another tab in the stack of tabbed panes.

The Function List view moves to the position in the stack where you release the **mouse** button.

3.2.3.5 Save a view layout scheme

You can save view layout schemes for a PowerBuilder painter and use them every time you open the painter.

1. Arrange the views in the painter as you like.
2. Select **View > Layouts > Manage** from the menu bar.
3. Click the **New Layout** button in the Layout dialog box.
4. Enter a name for your layout in the text field, click the background of the dialog box, and then click the *x* button in the upper right corner of the dialog box to close it.

Your layout scheme is saved. Now, when you select **View > Layouts**, you see your layout listed on the cascading menu.

Saving the toolbars and System Tree layouts

PowerBuilder saves the customizations you make to the toolbars and System Tree separately from the view layout. It retains those settings and reapplies them to every workspace you access and every view layout you select.

3.2.3.6 Reset the default view layout scheme

Each PowerBuilder painter has a default view layout scheme. You can always reset the layout scheme to this default layout.

1. Select **View > Layouts** from the menu bar.
2. Choose **Default** from the cascading menu.

The default view layout scheme is displayed in the painter workspace.

3.2.4 Set up the toolbars

A painter workspace always includes the PowerBar and other PainterBar toolbars that you can use as you work. The buttons in the toolbars change depending on the type of target or object you are working with. You can also customize the toolbars to include additional functionality.

Now you change the appearance of the toolbars to:

- [Show labels on toolbar buttons](#)
- [Float the toolbars](#)
- [Reposition the toolbars](#)

3.2.4.1 Show labels on toolbar buttons

You can learn a toolbar button's function by placing the cursor over it to view its PowerTip. A PowerTip is pop-up text that indicates a button's function.

You can also display a label on each toolbar button.

1. Move the pointer to any button on the PowerBar, but do not click.



The button's PowerTip is displayed.

2. Select **Tools > Toolbars** from the menu bar.

The Toolbars dialog box is displayed.

3. Select the **Show Text** checkbox, then click the **Close** button.

PowerBuilder displays a label on each of the buttons in the PowerBar and the PainterBars.

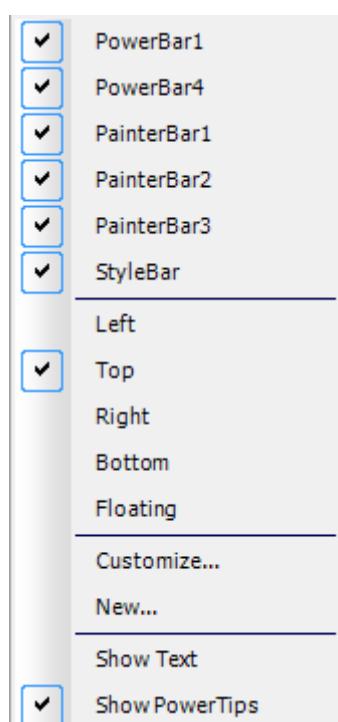
3.2.4.2 Float the toolbars

You can float the toolbars so that you can move them around the painter workspace as you work.

1. Right-click anywhere in the PowerBar.

The pop-up menu for the toolbars displays. From the pop-up menu you can set the toolbars location to the left, top, right, or bottom of the workspace. You can also set it to floating.

Figure 3.9: Toolbar



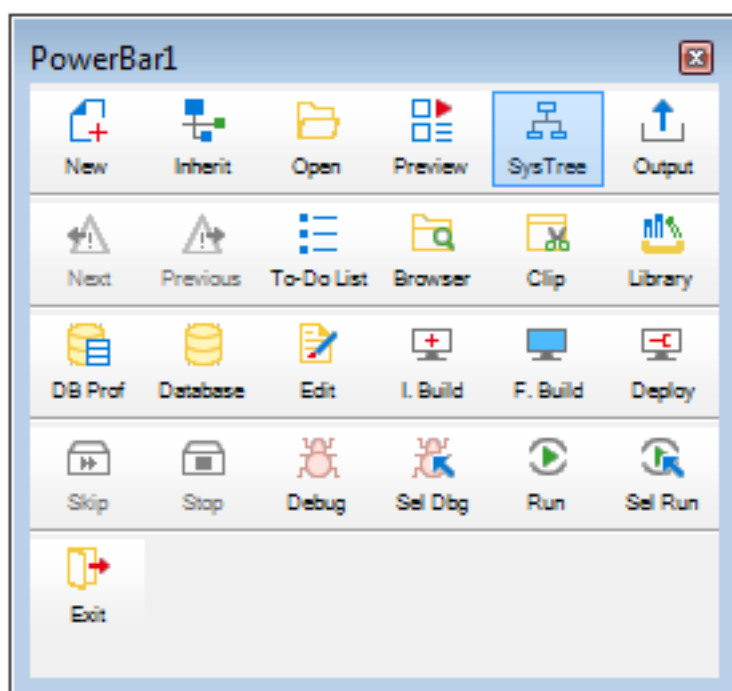
About pop-up menus

Throughout PowerBuilder, pop-up menus provide a fast way to do things. The menu items available in the pop-up depend on the painter you are using and where you are in the workspace when you click the right mouse button.

2. Select **Floating** from the pop-up menu.

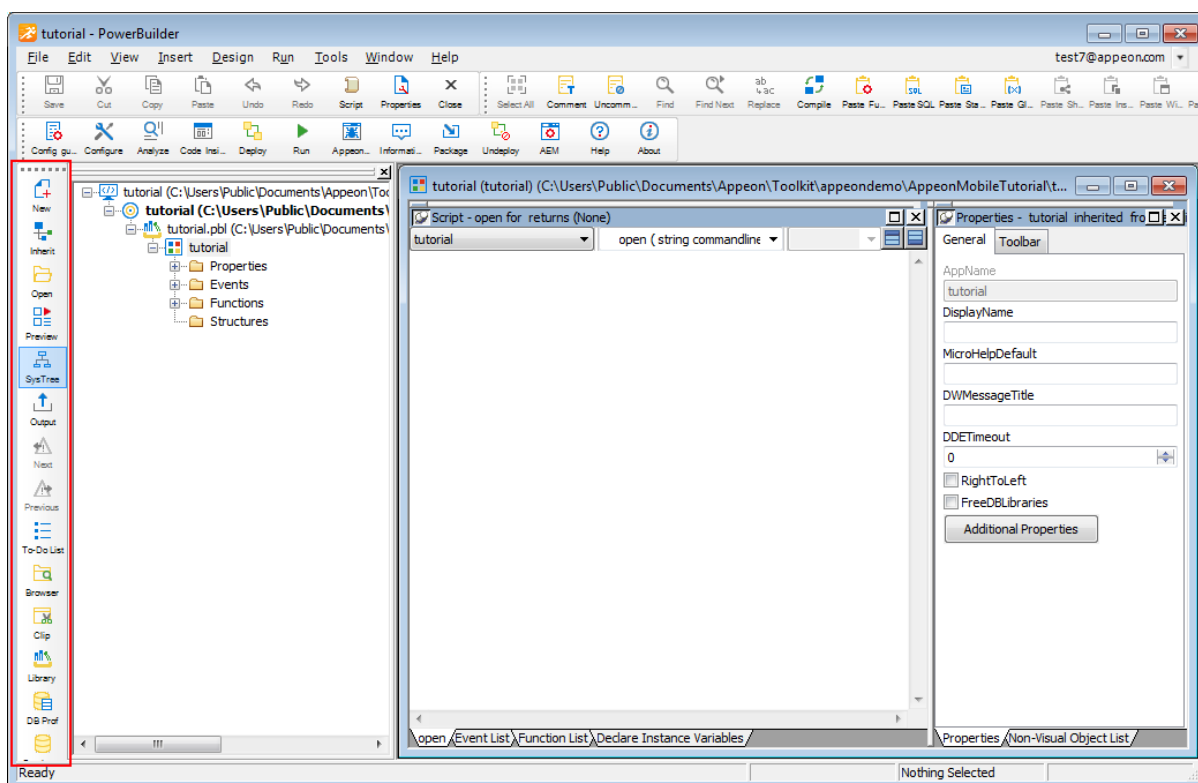
The PowerBar1 changes to a floating toolbar. You can adjust its shape.

Figure 3.10: PowerBar1



3. Move the pointer to an edge or border area in the PowerBar. Press and drag the PowerBar toward the left side of the workspace.

Release the mouse button when the PowerBar becomes a vertical bar.

Figure 3.11: Powerbar becomes a vertical bar

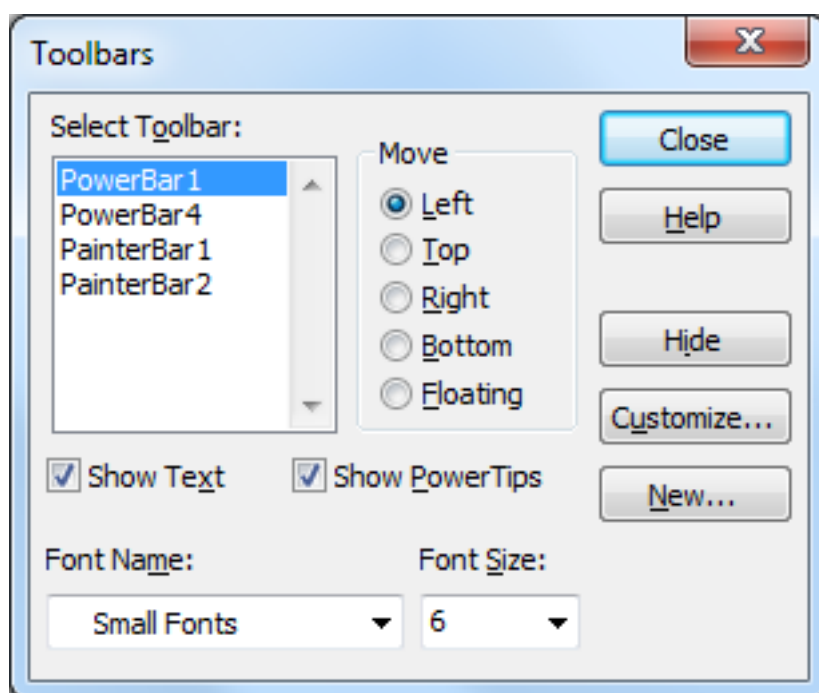
The PowerBar is docked at the left side of the frame.

3.2.4.3 Reposition the toolbars

You can customize the position of the toolbars to suit your work style.

1. Select **Tools > Toolbars** from the menu bar.

The Toolbars dialog box displays. The selected **Move** radio button indicates the position of the currently selected toolbar.

Figure 3.12: Reposition the toolbars

2. Click **Top**.

This repositions the PowerBar at the top of the workspace.

Radio buttons are grayed if a selected toolbar is hidden

If a selected toolbar is hidden (not visible) in the painter, you cannot select where it appears in the workspace. In this case, the radio buttons are grayed and you must first click the **Show** button before you can select a radio button. The **Show** button replaces the **Hide** button when a toolbar is hidden.

3. Click PainterBar1 in the Select **Toolbar** list box and select **Right**.

Click **Close** in the Toolbars dialog box.

4. Right-click PainterBar2 and select **Left** from the pop-up menu.

You have swapped the locations of the two painter bars.

5. Arrange the toolbars to suit your preferences.

You can also drag the toolbars to the top, bottom, left, or right of the painter workspace. When a toolbar is in a fixed location, it has a drag bar at the left or top of its buttons. You can click the drag bar and drag the mouse to move the toolbar around the painter workspace.

PowerBuilder applies toolbar configuration properties to all painters and saves them for the next PowerBuilder session.

6. Close the Application painter.

If you are not continuing immediately with the tutorial

You can close PowerBuilder or the tutorial workspace if you want. In that case, you must open the tutorial workspace before you continue with the next section.

3.3 Connecting to the Database

This section shows you how to make the application connect to the Enterprise Application Sample demonstration database (Apeon Sample) at development time and how to use the Database Profiles dialog box to create the database profile for this database and use the **Database painter** to look at the table definitions.

3.3.1 About database connection

In many organizations, database specialists maintain the database. If this is true in your organization, you might not need to create and maintain tables within the database. However, to take full advantage of PowerBuilder, you should know how to work with databases.

Defining a data source

Using the ODBC administrator or other database connection utilities, you can define a database as a data source for your application. You can access the ODBC Administrator from the DataBase Profiles dialog box. The definitions of ODBC data sources are stored in the *odbc.ini* registry key.

Using database profiles to connect

Once you define a data source, you can create a database profile for it. A database profile is a named set of parameters that specifies a connection to a particular data source or database. Database profiles provide an easy way for you to manage database connections that you use frequently. When you are developing an application, you can change database profiles to connect to a different data source.

When database connections occur

PowerBuilder can establish a connection to the database in either the design-time or runtime environment. PowerBuilder connects to a database when you open certain painters, when you compile or save a PowerBuilder script that contains embedded SQL statements, or when you run a PowerBuilder application that accesses the database.

Note: The database connections in this section is for the PowerBuilder application only. To establish database connection for the mobile application, you will have to create a data source in the PowerServer. Detailed instructions are provided in the PowerServer Configuration Guide for .NET or PowerServer Configuration Guide for J2EE.

To maintain database definitions with PowerBuilder, you do most of your work using the **Database painter**. The Database painter allows you to:

- Create, alter, and drop tables
- Create, alter, and drop primary and foreign keys
- Create and drop indexes

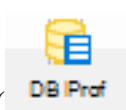
- Define and modify extended attributes for columns
- Drop views


3.3.2 Create the database profile for the Appeon Sample database

After you install PowerServer Mobile, the Appeon Sample database is automatically installed and defined as a data source in the ODBC Administrator. You use the Appeon Sample database in this tutorial.

Appeon Sample is a SQL Anywhere database that is accessed through ODBC. In this section you create the database profile for the Appeon Sample database. PowerBuilder stores database profile parameters in the registry.

1.



Click the **Database Profile** () button in the PowerBar

or

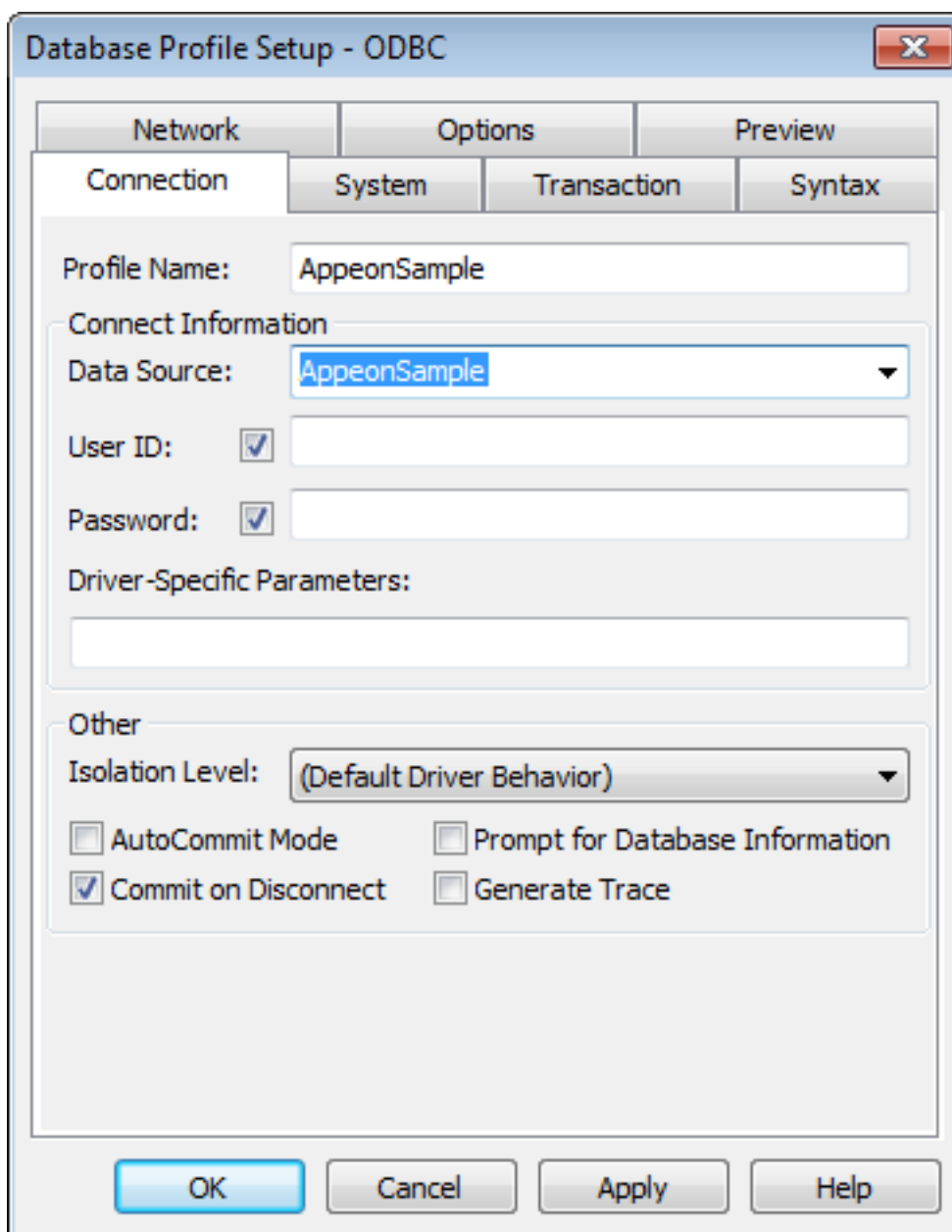
Select **Tools > Database Profile** from the menu bar.

PowerBuilder displays the **Database Profiles** dialog box, which includes a tree view of the installed database interfaces and defined database profiles for each interface. You can click the + signs or double-click the icons next to items in the tree view to expand or contract tree view nodes.

2. Select **ODB ODBC** in the tree view of the **Database Profiles** dialog box and click **New**.

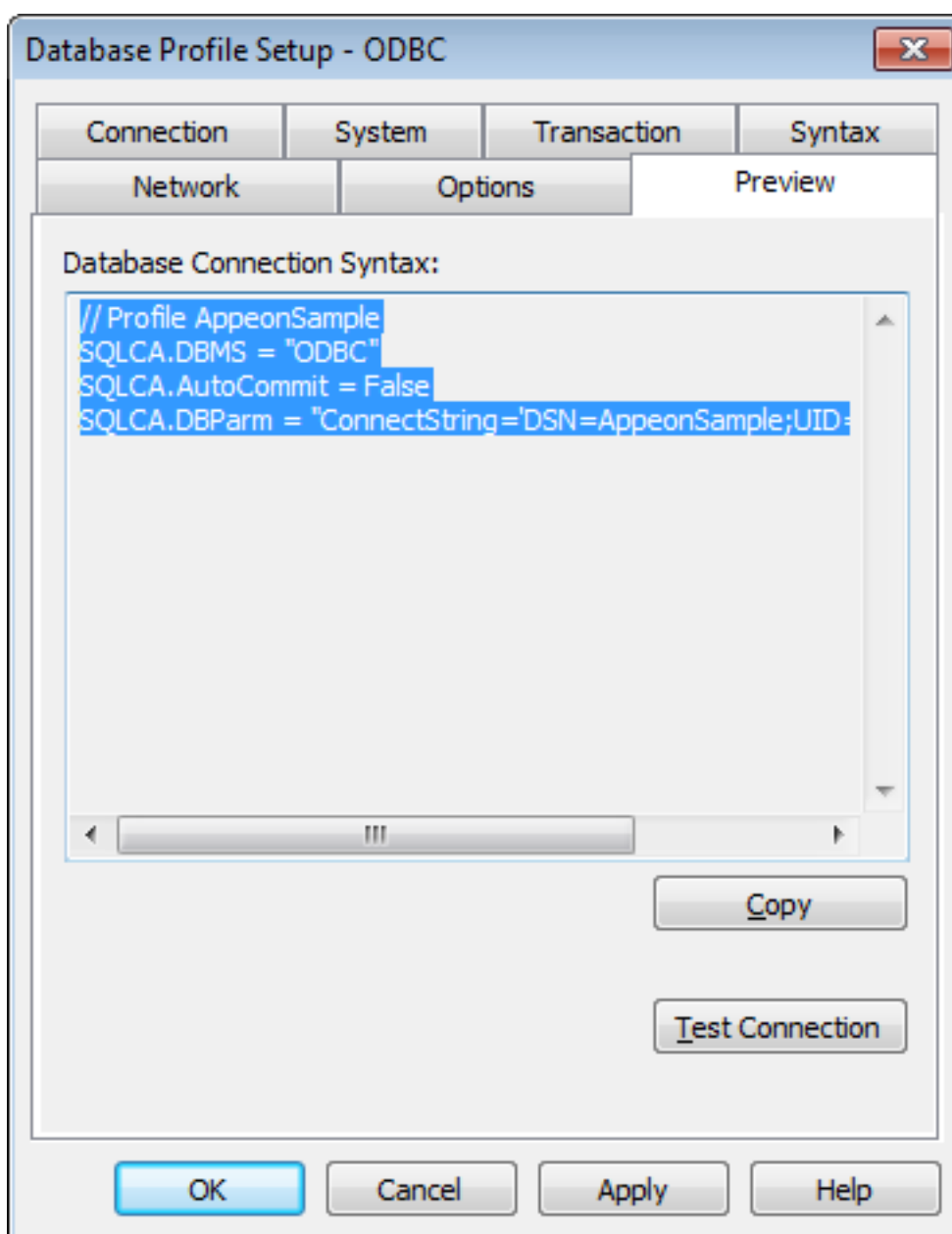
PowerBuilder displays the **Connection** tab of the **Database Profile Setup** dialog box.

3. On the **Connection** tab of the **Database Profile Setup** dialog box, select the *AppeonSample* data source from the **Data Source** drop-down list and type *AppeonSample* in the **Profile Name** text box.

Figure 3.13: Connection tab

4. Select the **Preview** tab.

The PowerScript connection syntax for the profile is shown on the **Preview** tab. If you change the profile connection options, the syntax changes accordingly.

Figure 3.14: Preview tab

5. Click the **Test Connection** button.

A message box tells you that the connection is successful. Click **OK** to close the message box.

If the message box tells you the connection is not successful

Close the message box and verify that the information on the Connection page of the Database Profile Setup dialog box is correct. Then check the configuration of the data source in the ODBC Administrator. You can run the ODBC Administrator by expanding the Utilities folder under the **ODBC** node of the **Database Profile** painter and double-clicking the **ODBC Administrator** item.

6. Click **OK** to close the **Database Profile Setup** dialog box. The AppeonSample database profile is created under the **ODB ODBC** node.
7. Select the AppeonSample database profile and click **Connect**.

What happens when you connect

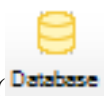
When you connect to a database in the development environment, PowerBuilder writes the connection parameters to the Windows registry. Each time you connect to a different database, PowerBuilder overwrites the existing parameters in the registry with those for the new database connection. When you open a PowerBuilder painter that accesses the database, you automatically connect to the last database used. PowerBuilder determines which database this is by reading the registry.

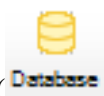
8. Click **Close** to close the **Database Profiles** dialog box.

3.3.3 Look at table definitions in the Appeon Sample database

Now you look at the definitions for the Customer table in the Appeon Sample database. This helps you become familiar with the **Database** painter and the table you will use in the tutorial.

- 1.



Click the **Database** () button in the PowerBar.

PowerBuilder connects to the database and the **Database** painter opens.

The **Database** painter title bar identifies the active database connection.

The **Objects** view of the **Database** painter displays all existing database profiles in a tree view under the **Installed Database Interfaces** heading. The AppeonSample database is visible under the **ODB ODBC** node in the tree view.

If the Objects view is not open

The Objects view is part of the default view layout scheme. To reset to this scheme, select **View > Layouts > Default**. You can also open an Objects view by selecting **View > Objects** from the menu bar.

2. Expand the AppeonSample database node in the **Objects** view.

Notice the folders under the AppeonSample database node.

Figure 3.15: Objects view

3. Expand the **Tables** folder.

You see the list of tables in the database.

Table names might have a prefix

The table names in the Select Tables dialog box might have a prefix such as *dba* or *dbo*. This depends on the login ID you are using. You can ignore the prefix.

4. Right-click the customer table and select **Add To Layout** from the pop-up menu

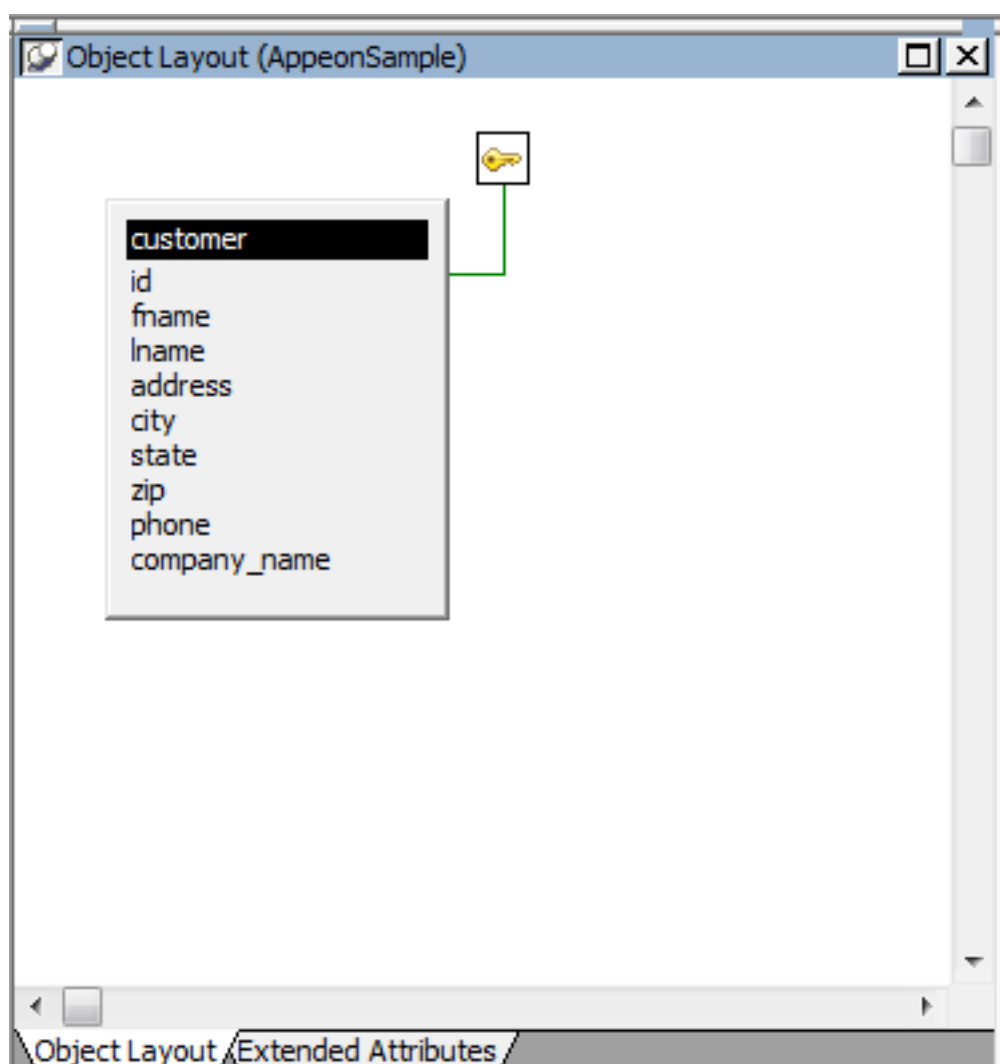
or

Drag the customer table from the **Objects** view to the **Object Layout** view.

Dragging an object from one view to another

When you start dragging an object from the **Objects** view to another view, the pointer changes to a barred circle. If you continue moving the cursor to a view that can accept the object, the barred circle changes back to a pointer with an additional arrow symbol in a small box. When you see this symbol, you can release the object.

The **Object Layout** view shows the table you selected.

Figure 3.16: Object Layout view

5. Right-click the title bar of the customer table in the **Object Layout** view and select **Alter Table** from the pop-up menu

or

Right-click the customer table in the **Objects** tree view and select **Alter Table** from the pop-up menu.

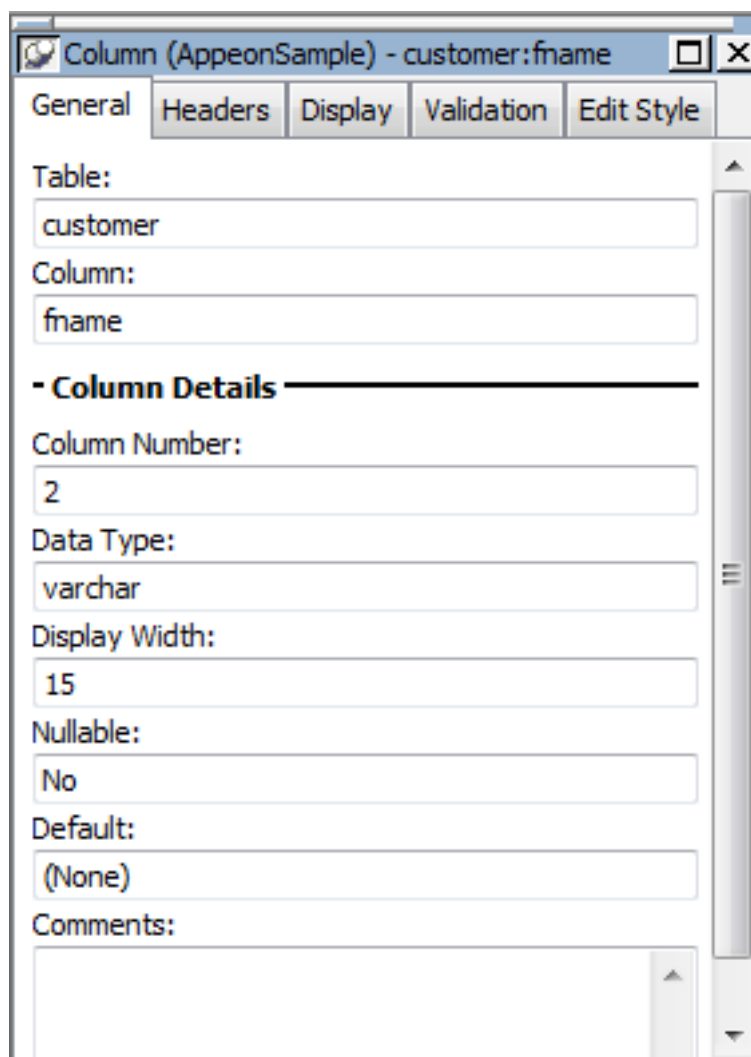
The **Columns** view displays the column definitions for the table.

6. Right-click a column in the customer table in the **Object Layout** view.

Select **Properties** from the pop-up menu. The **Column** view displays the details of the current column.

The title bar and tab headings for the **Column** view change dynamically depending on the current column selection. The title bar gives the database connection, the table name, and the column name.

The **Column** view for a column has five tabs, one for general database properties, one for column header information, and the others for column extended attributes.

Figure 3.17: Column view

About extended attributes

PowerBuilder stores extended attribute information in system tables of the database. Extended attributes include headers and labels for columns, initial values for columns, validation rules, and display formats.

You can define new extended attributes or change the definitions of existing extended attributes from the pop-up menus of items in the Extended Attributes view of the Database painter.

7. Close the **Database** painter.

3.4 Setting Up the Menus

In this section you set up the menus for the application.


Menus are separate objects that you create using the **Menu** painter. After you create a menu, you can attach it to as many windows as you want. You can create menus at any time during the application development process.

3.4.1 Create a new menu

Now you create a new menu that displays whenever the user opens the MDI frame (to be created later).

1.



Click the **New** () button in the **PowerBar**

or

Select **New** from the **File** menu.

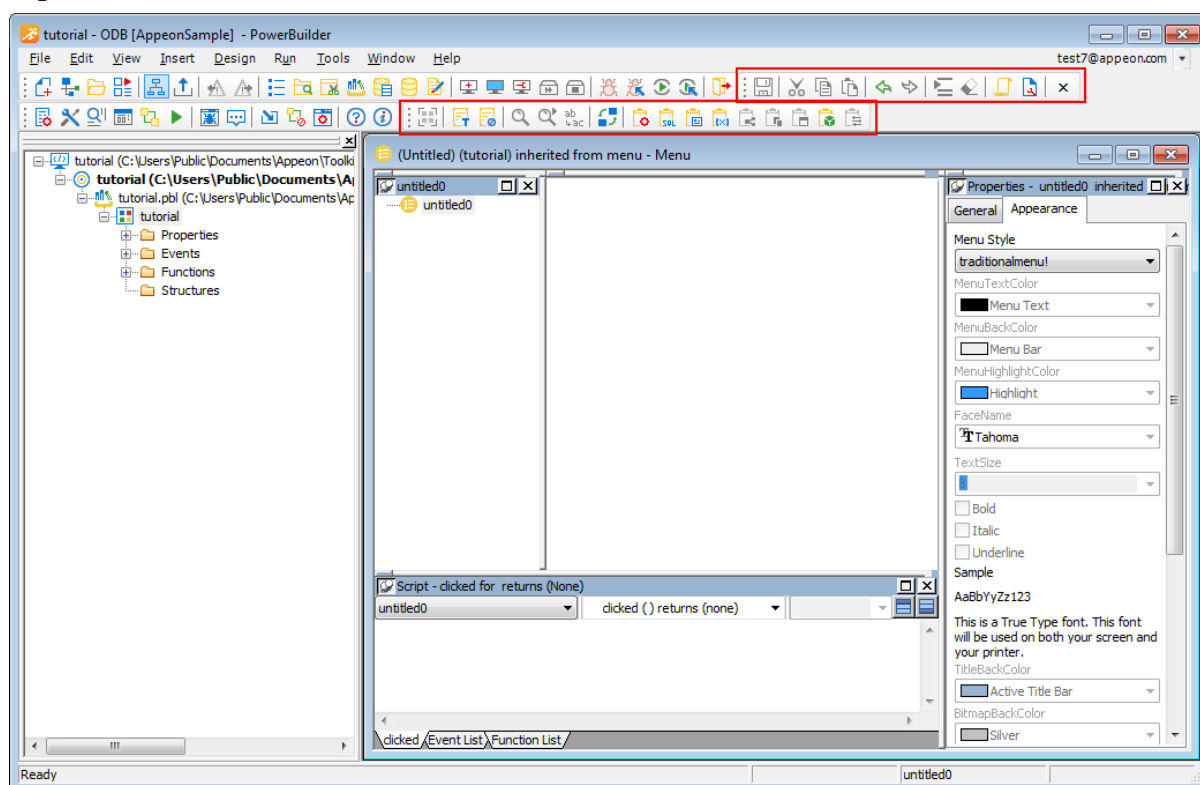
The **New** dialog box appears.

2. Select the **PB Object** tab.

Select the **Menu** icon and click **OK**.

PowerBuilder displays an untitled menu. Notice that you have two new toolbars, the PainterBar1 (with save, cut, copy, paste, close buttons etc.) and PainterBar2 (with buttons used with the Script view).

Figure 3.18: Untitled menu



3.4.2 Add items to the new menu

Next you add items to the **untitled** menu you just created. You use the WYSIWYG and Properties views.

1. Right-click the **untitled** menu in the **WYSIWYG** view.

2. Select **Insert Submenu Item** from the pop-up menu.

The cursor moves into a blank box that appears at the end of the **untitled** menu list.

3. Type *Tutorial* in the text box and press **Enter**.

In the **Properties** view, the menu item name changes to `m_tutorial`. If PowerBuilder displays a message that the default name is incorrect, it suggests an alternative name. If this occurs, click **OK** to accept the suggested name.

4. Right-click the newly created **Tutorial** menu item in the WYSIWYG view.

5. Select **Insert Submenu Item** from the pop-up menu.

The cursor moves into a blank box that appears at the end of the **Tutorial** menu item list.

6. Type *Open* in the text box that appears and press **Enter**.

In the **Properties** view, the menu item name changes to `m_open`.

Alternative method of inserting menu item names

You can type *Open* in the **Text** box in the **Properties** view instead of typing it in the box that appears in the WYSIWYG view. In this case, you do not need to press **Enter** afterwards. To have PowerBuilder automatically reset the menu item name, you will have to clear the **Lock Name** check box if the **Name** box is grayed.

7. Select **File > Save** from the menu bar.

The **Save Menu** dialog box appears.

8. Type `m_main` in the **Menus** box.

9. Click **OK**.

This names the menu. The prefix `m_` is standard for menus.

The name you just assigned to the new menu displays in the title bar of the **Menu** painter workspace and the `m_main` menu appears in the system tree.

3.4.3 Add a new toolbar for the new menu item

Now you add a toolbar button for the **Open** menu item you just defined and then place it in a toolbar. The **Menu** painter should still be open for the `m_main` menu. If it is not, you can double-click the `m_main` menu in the **System Tree** to open it in the **Menu** painter.

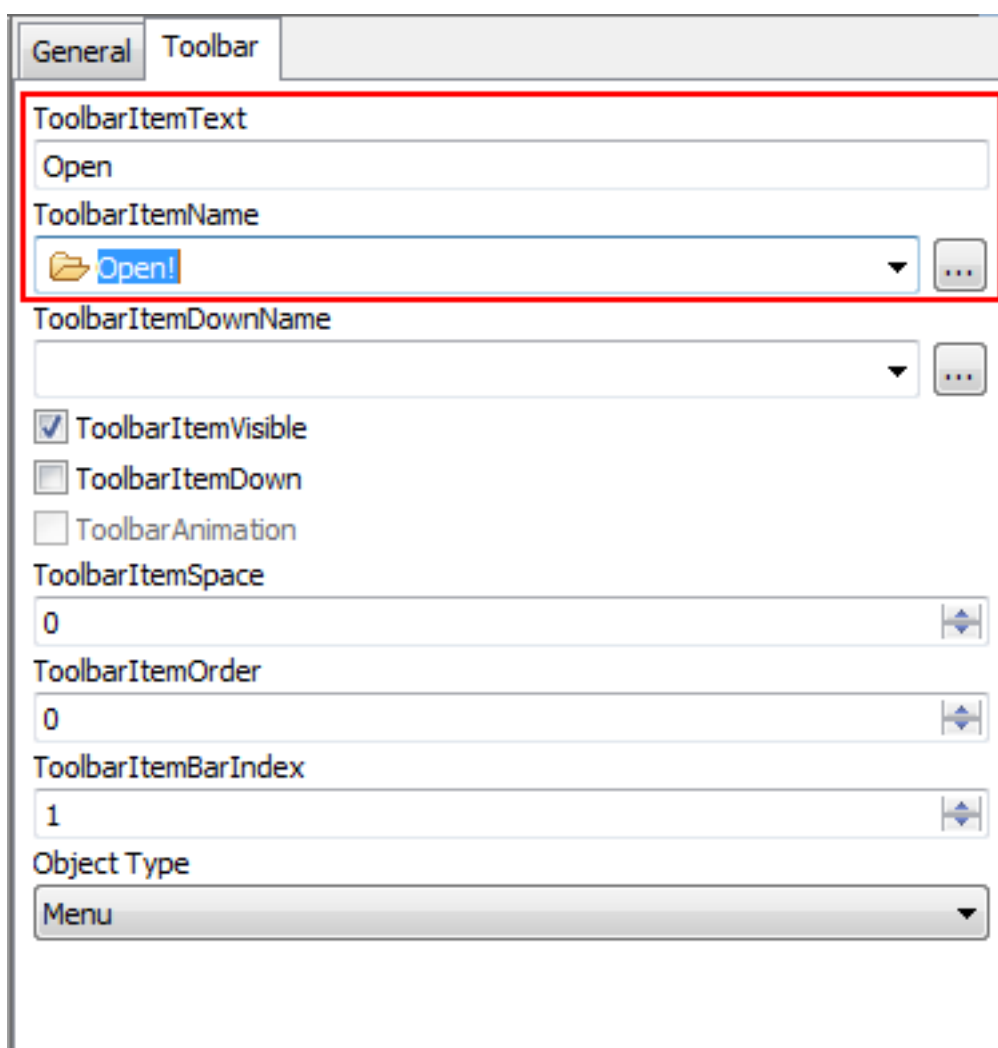
1. Click the **Open** menu item under the **Tutorial** menu in the WYSIWYG view.

2. Click the **Toolbar** tab in the **Properties** view.

Type *Open* in the **ToolbarItemText** box.

Type or select **Open!** in the **ToolbarItemName** drop-down list.

This defines a toolbar button for the **Open** menu item that uses the stock picture called **Open!**.

Figure 3.19: Open toolbar

3. Select **File > Close** from the menu bar.

A message box asks if you want to save your changes.

4. Click **Yes**.

PowerBuilder saves the Menu object and closes the **Menu** painter.

3.5 Building a DataWindow Object

The DataWindow object is one of the most powerful and useful features of PowerBuilder. A DataWindow object can connect to a database, retrieve rows, display rows in various presentation styles, and update the database.

3.5.1 Create and preview a DataWindow object

Now you create a new DataWindow object and display it in the DataWindow painter. Like other painters, the DataWindow painter has an assortment of views that you can open simultaneously.

About the Design view of the DataWindow painter

The Design view in the DataWindow painter is similar to the Layout view in other painters. You can open only one Design view at a time.

The Design view is divided into four areas called bands: header, detail, summary, and footer. You can modify the contents of these bands. For example, you can change their sizes, add objects (controls, text, lines, boxes, or ovals), and change colors and fonts.

In the Preview view of the DataWindow painter, you can see how the object looks in an application at runtime, complete with table data.



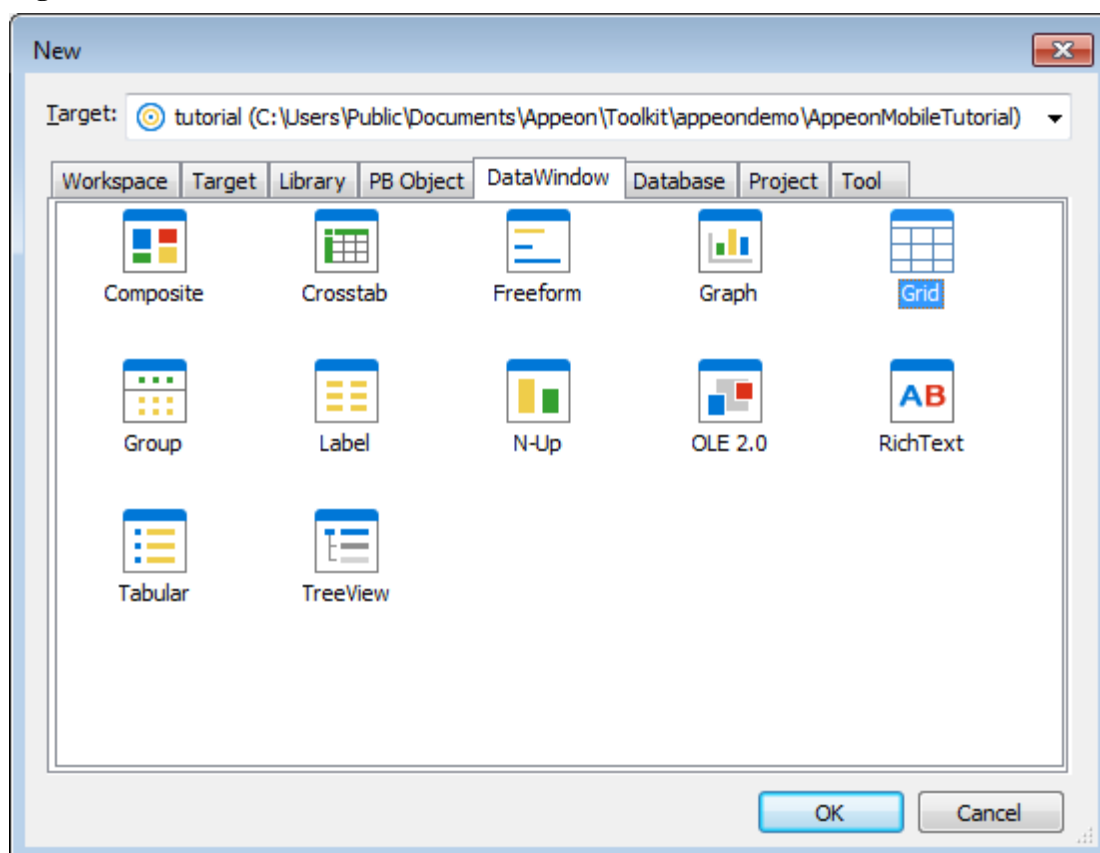
1.  Click the **New** button () in the PowerBar.
or
Select **New** from the **File** menu.
The **New** dialog box appears.
2. Click the **DataWindow** tab.
3. Select **Grid** from the list of presentation styles.

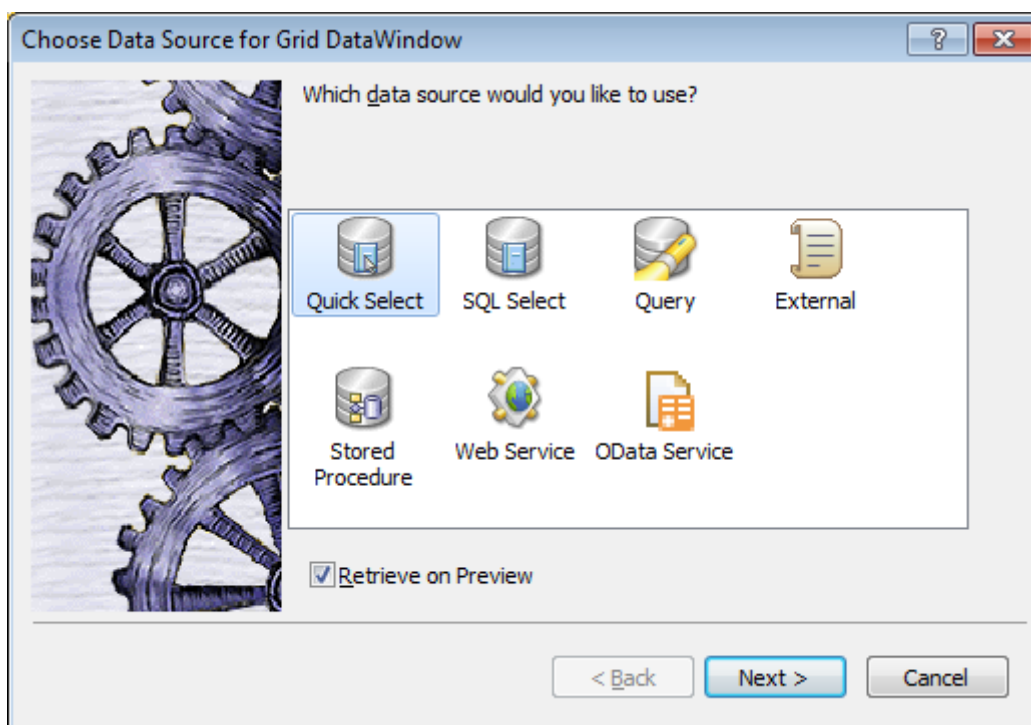
Figure 3.20: Grid in DataWindow tab



4. Click **OK**.

The **Choose Data Source for Grid DataWindow** page of the DataWindow wizard appears.

Figure 3.21: Choose Data Source for Grid DataWindow wizard



5. Select **Quick Select** as the data source and select the **Retrieve On Preview** check box if it is not already selected. Click **Next**.

PowerBuilder connects to the Appeon Sample database, and the **Quick Select** dialog box appears.

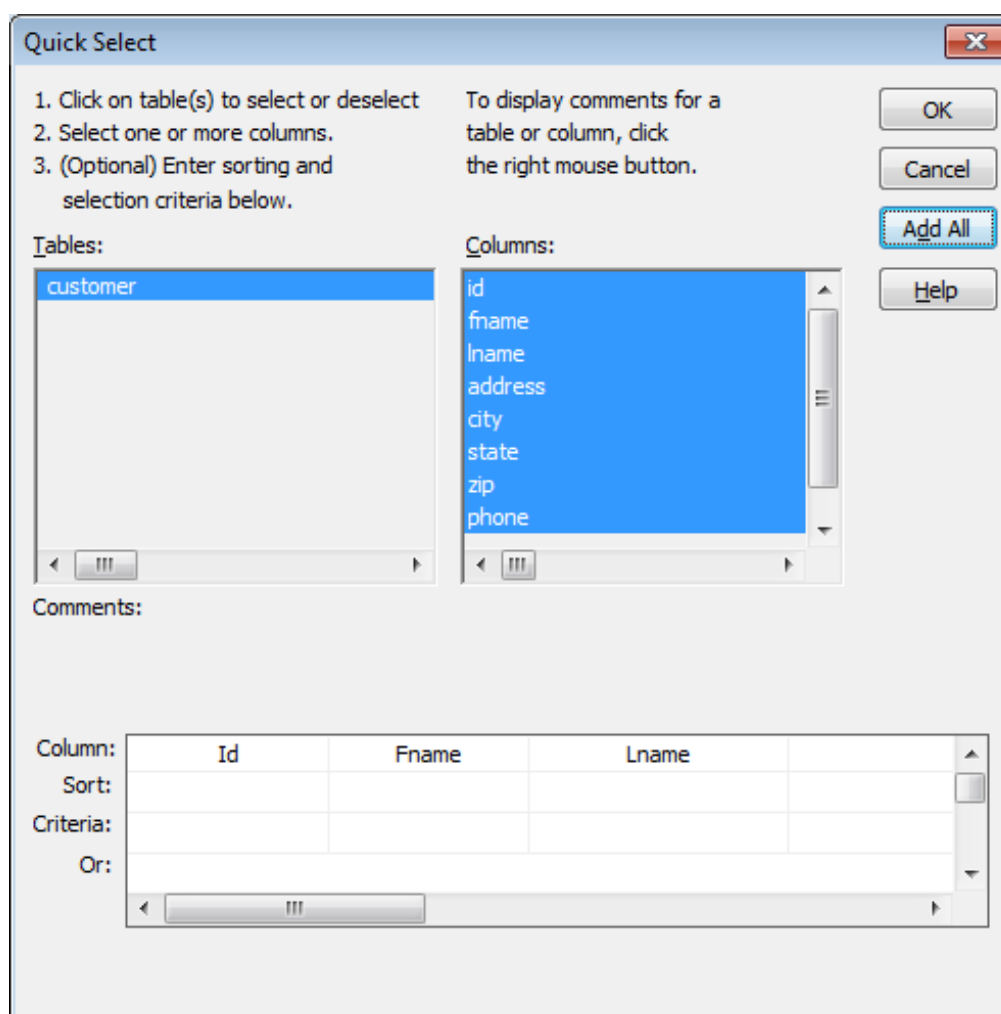
6. Click the **customer** table in the **Tables** list box.

This opens the table and lists its columns. For this DataWindow, you will select all columns.

7. Click the column one by one to select all items in the **Columns** list box

or

Click the **Add All** button to select all items in the **Columns** list box.

Figure 3.22: Quick Select window

PowerBuilder displays the selected columns in a grid at the bottom of the **Quick Select** dialog box.

Selection order determines display order

The order in which you select the columns determines their left-to-right display order in the DataWindow object. If you clicked a column by mistake, you can click it again to clear the selection.

You can use the grid area at the bottom of the dialog box to specify sort criteria (for the SQL ORDER BY clause) and selection criteria (for the SQL WHERE clause). Now you specify sort criteria only. You sort the id column in ascending order.

8. In the grid area of the **Quick Select** dialog box, click in the cell next to **Sort** and below **Id**. A drop-down list box appears.
9. Choose **Ascending** from the drop-down list box.

This specifies that the **Id** column is to be sorted in ascending order.

10. Click **OK**.

The DataWindow wizard asks you to select the colors and borders for the DataWindow object. By default, there are no borders for text or columns.

11. Click **Next**.

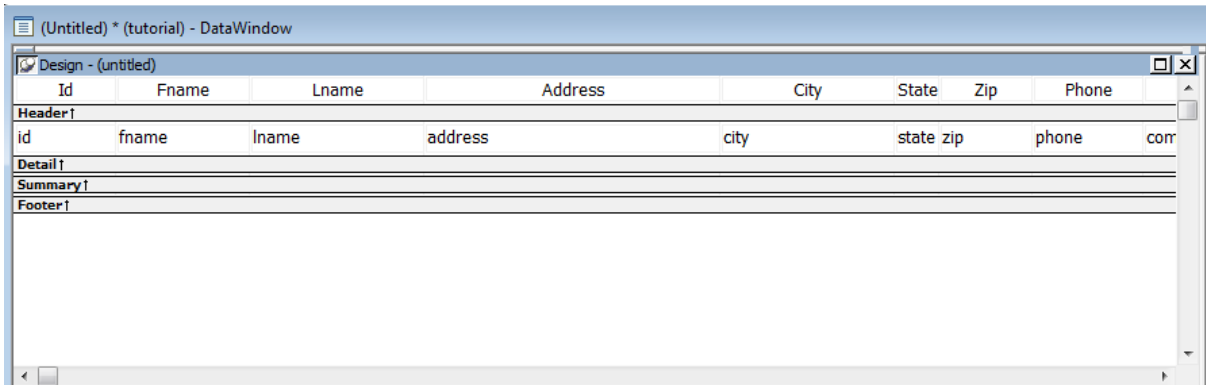
You accept the border and color defaults. The DataWindow wizard summarizes your selections.

12. Click **Finish**.

PowerBuilder creates the new DataWindow object and opens the DataWindow painter. Notice that you have four new toolbars, the StyleBar (with character style and text alignment buttons), PainterBar1 (with save, cut, copy, paste, close buttons etc.), PainterBar2 (with buttons used with the Script view) and PainterBar3 (with color and border buttons, as well as grayed out control alignment buttons).

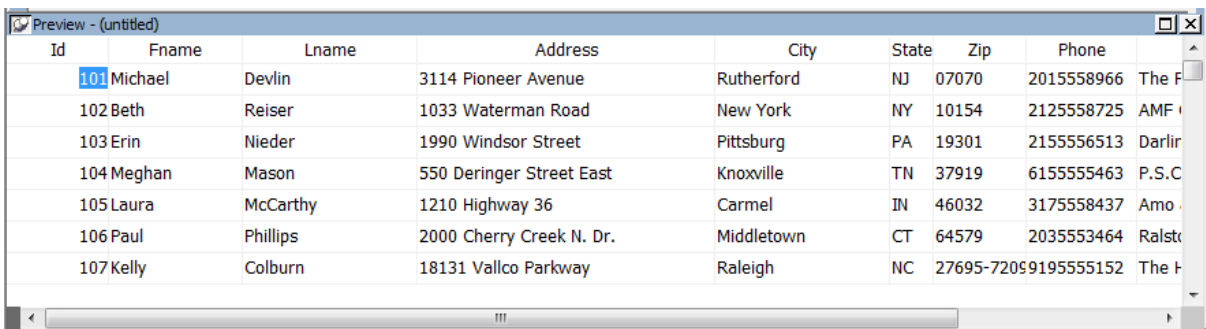
In the **Design** view, PowerBuilder displays a **Header** band with default headings and a **Detail** band with the columns you selected.

Figure 3.23: DataWindow Design view



The **Preview** view displays the DataWindow as it appears during execution. PowerBuilder displays data for all customers. The data is sorted in ascending order by **Id**, just as you specified.

Figure 3.24: DataWindow Preview view



Displaying the Preview view

If the Preview view is not displayed, select **View > Preview** from the menu bar. If **Preview** is grayed, it is already displayed and you cannot select it. You can open only one Preview view at a time.

3.5.2 Save the DataWindow object

Now you name the DataWindow object and save it in the *tutorial.pbl* library.

Saving to another library

You can save objects to different application libraries, but to avoid complications, you save all your new tutorial objects in one library. You can also copy or move objects from one library to another using the Library painter.

Follow the following steps to name the DataWindow object and save it in the *tutorial.pbl* library.

1. Select **File > Save** from the menu bar.

The **Save DataWindow** dialog box appears with the insertion point in the DataWindows box.

2. Make sure *tutorial.pbl* is selected in the **Application Libraries** box.

Type *d_customer* in the DataWindows box.

This names the DataWindow object. The prefix *d_* is standard for DataWindow objects.

3. (Optional) Type the following comments in the **Comments** box:

This DataWindow object retrieves customer names and company associations.

4. Click **OK**.

PowerBuilder saves the DataWindow object and closes the **Save DataWindow** dialog box.

3.5.3 Make cosmetic changes to the DataWindow object

Now you can make cosmetic changes to the DataWindow. You change the font size and height of the columns and column headers and change the text of column headers. You also adjust the column width to make text display completely.

You make these changes in the Design view. You can keep the Preview view open at the same time to see how the changes you make affect the appearance of the DataWindow at runtime.

1. Select **Edit > Select > Select All** from the menu bar

or

Press **Ctrl+A**.

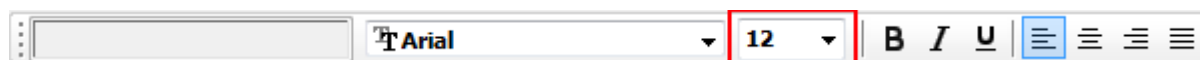
All of the controls in the DataWindow object are selected in the **Design** view.

2. Select the **Font** page in the **Properties** view, and then select or type *12* from/in the **Size** list box.

or

Select or type *12* from/in the StyleBar.

Figure 3.25: Font size property in StyleBar



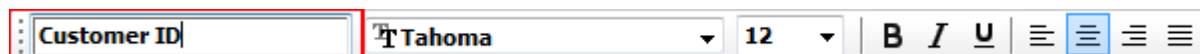
Using the StyleBar

You can also use the StyleBar to change fonts, text alignment etc. If you do not see the StyleBar, select **Tools > Toolbars** from the menu bar, click **StyleBar** in the **Select Toolbar** list box, and then select one of the **Move** positions such as bottom or floating.

This changes the character size to 12 for all texts in the `d_customer` DataWindow object.

3. Click the **Header** band in the **Design** view.
Type *120* in the **Height** text box on the **General** page in the **Properties** view.
Select *Sky* in the **Color** list box on the **Background** page in the **Properties** view.
This changes the height and the background color of the header row.
4. Click the **Detail** band in the **Design** view.
Type *120* in the **Height** text box on the **General** page in the **Properties** view.
This changes the height of the detail row.
5. Select the **Id** header above the **Header** band in the **Design** view.
Type *Customer ID* in the **Text** box on the **General** page in the **Properties** view.
or
Type *Customer ID* in the first box in the StyleBar.

Figure 3.26: Text property in StyleBar



This changes the text of the **Id** header to **Customer ID**.

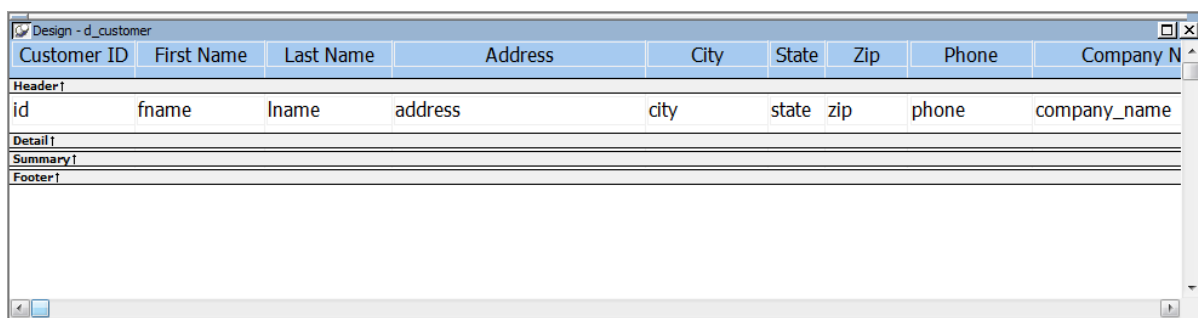
Use the same method to change the text of the **Fname** and **Lname** headers to *First Name* and *Last Name*.

6. Hover over the right of the **Customer ID** header above the **Header** band in the **Design** view with your mouse pointer. When a plus sign with arrows indicating to drag right/left appears, drag the column to right or left to increase the column width, so that the text in the column display completely.

Use the same method to adjust the width for other columns.

When you have finished, the **Design** view should look something like this:

Figure 3.27: Design view



7. Close the DataWindow painter by clicking the close button in PainterBar1. The close button is labeled with an *x*.

Click **Yes** when you are prompted to save your changes.

3.6 Building a Window

Windows are the main interface between users and applications. Windows can display information, request information from a user, and respond to mouse or keyboard actions.

Windows are separate objects that you create using the **Window** painter. In PowerBuilder, you can create windows anytime during the application development process.

In this section you:

- [Build an MDI frame window](#)
- [Build an MDI sheet window](#)
- [Run the application](#)

3.6.1 Build an MDI frame window

Now you create an MDI frame window for the application.

In this exercise you:

- [Create an MDI frame window](#)
- [Attach menu to the frame window](#)
- [Change the size of the frame window](#)
- [Write scripts to open the frame window](#)

3.6.1.1 Create an MDI frame window

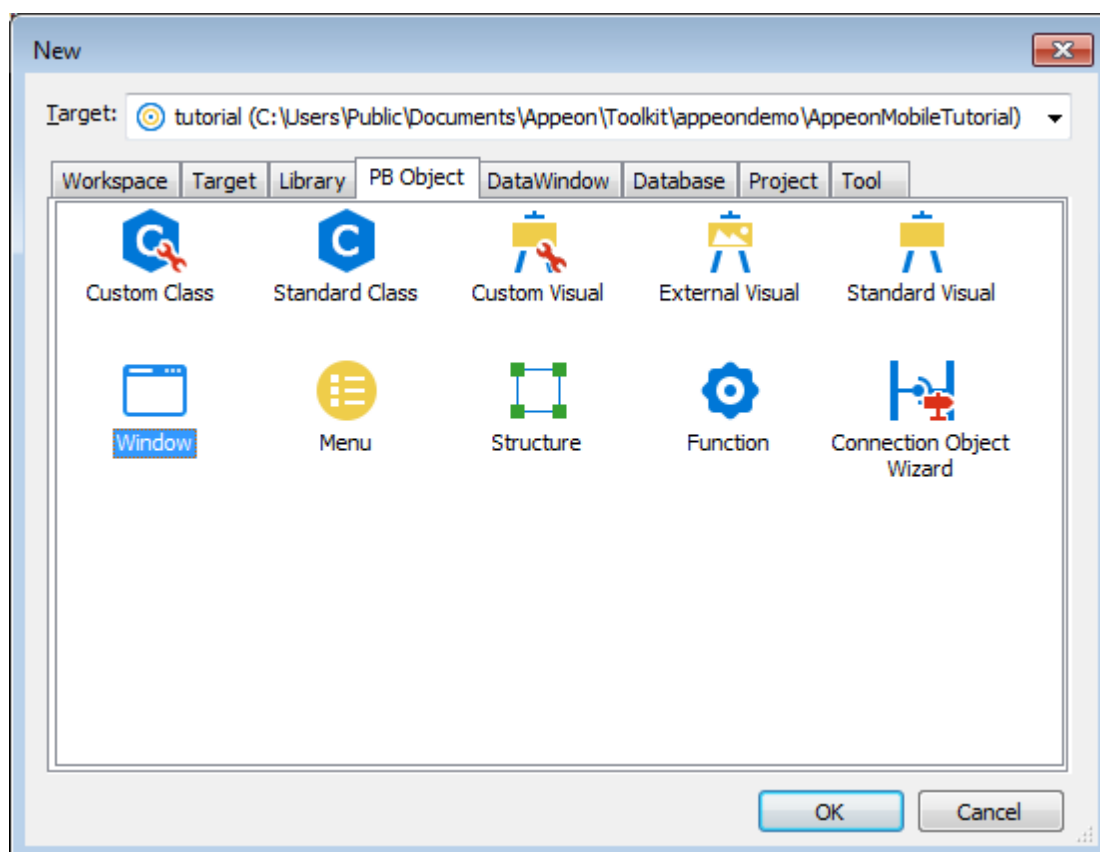
1. Click the **New** button in the PowerBar1.

The **New** dialog box is displayed.

2. Click the **PB Object** tab.

Select the **Window** icon and click **OK**.

Figure 3.28: Create a main window



The **Window** painter opens. Notice that you have four new toolbars, the StyleBar (with character style and text alignment buttons), PainterBar1 (with save, cut, copy, paste, close buttons etc.), PainterBar2 (with buttons used with the Script view) and PainterBar3 (with color and border buttons, as well as grayed out control alignment buttons).

3. Make sure that the **Layout** view and the **Properties** view are displayed in the **Window** painter.

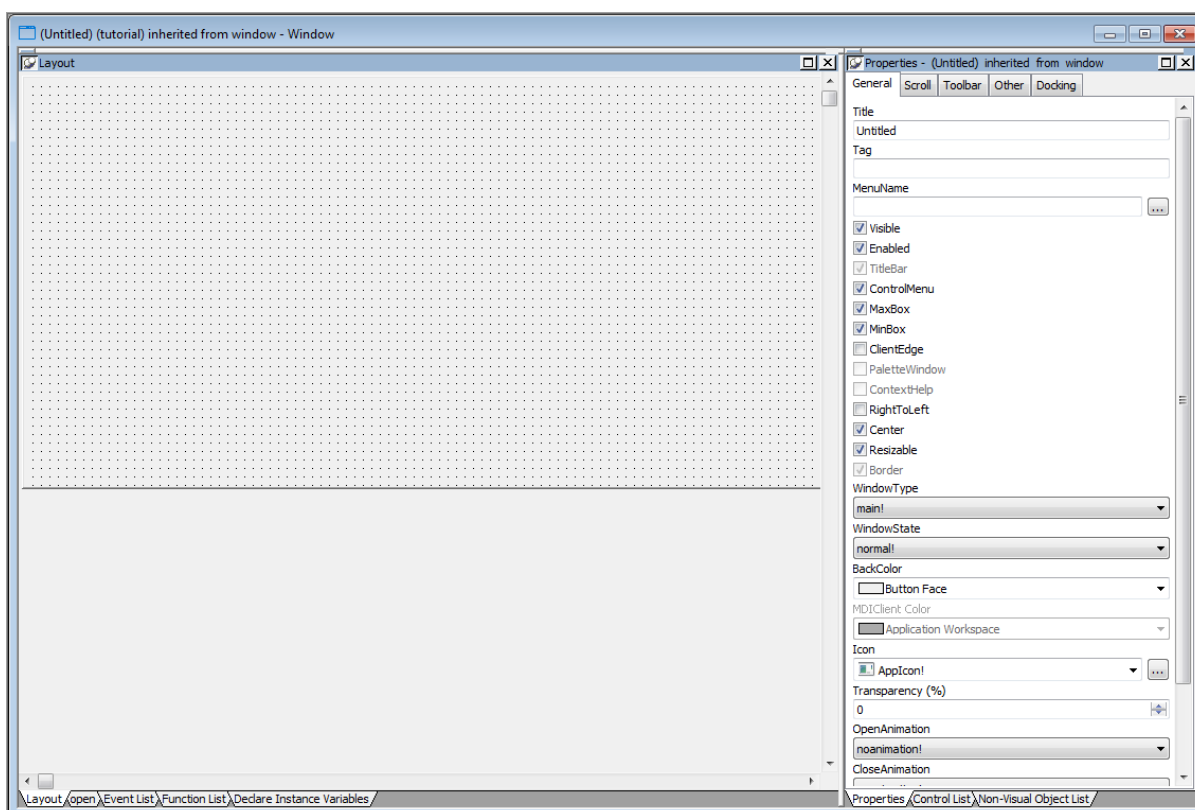
You can display these views by selecting them from the **View** menu. If they are grayed out in the menu, the views are already displayed in the painter.

The default view layout scheme contains both views.

To retrieve the default painter layout

Select **View > Layouts > Default** from the workspace menu bar.

The rectangle in the **Layout** view represents the window you are building.

Figure 3.29: The Layout view and the Properties view

If your window does not have a pegboard look

If the window in your **Layout** view is displayed as a solid color, the **Show Grid** option has been disabled. To enable it, select **Design > Options** from the menu bar. Then select the **Show Grid** checkbox on the **General** page of the **Options** dialog box. Click **Apply**, then **OK** to save the change and close the dialog box.

4. Type **Tutorial main window** in the **Title** text box on the **General** page of the **Properties** view.

Select **mdi!** in the **WindowType** list box.

Select **White** in the **MDIClient Color** list box.

Figure 3.30: MDI frame window properties

The image shows a dialog box for configuring MDI frame window properties. The 'General' tab is active. The 'Title' field contains 'Tutorial main window'. The 'WindowType' dropdown is set to 'mdi!'. The 'MDIClient Color' dropdown is set to 'White'. Other properties include 'Tag', 'MenuName', and various checkboxes for 'Visible', 'Enabled', 'TitleBar', 'ControlMenu', 'MaxBox', 'MinBox', 'ClientEdge', 'PaletteWindow', 'ContextHelp', 'RightToLeft', 'Center', 'Resizable', and 'Border'.

When you set the window type to **mdi!** or **mdihelp!**, you must associate a menu with the frame. Menus usually provide a way to open sheets in the frame.

About MDI

MDI stands for multiple document interface. In an MDI application, the main window for the application is called the MDI frame. Using the MDI frame menu bar, you

can open additional windows known as sheet windows that display inside the frame window.

3.6.1.2 Attach menu to the frame window

Now you attach the `m_main` menu you have created to the MDI frame window.

1. Click the ellipsis button next to the **MenuName** box on the **General** page of the **Properties** view.

The **Select Object** dialog box appears.

2. Select `m_main` in the **Menus** list box and click **OK**.

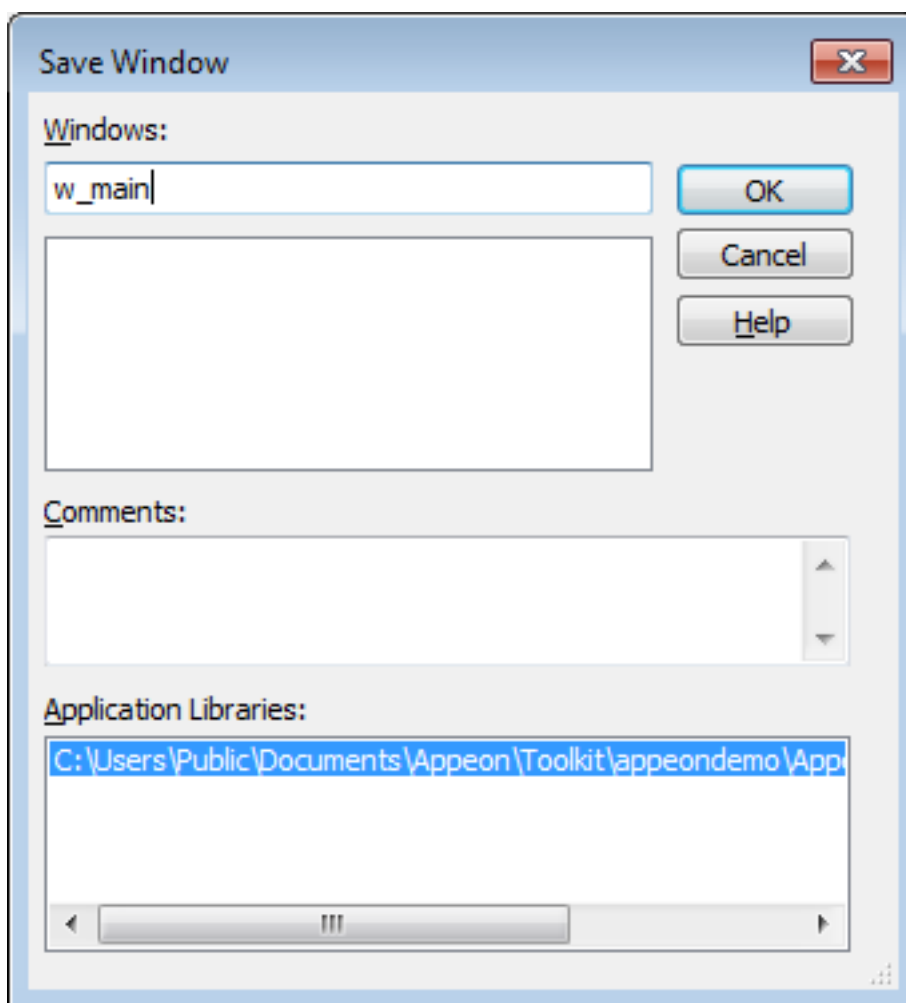
This is the menu you have created. It is now listed as the menu name in the **Properties** view.

3. Select **File > Save** from the menu bar.

The **Save Window** dialog box is displayed. The only library in the **Application Libraries** text box is `tutorial.pbl`, and it is selected.

4. Type `w_main` for the window name.

The prefix `w_` is standard for windows.

Figure 3.31: Save Window

5. Click **OK**.

PowerBuilder saves the new MDI frame window. If you expand **tutorial** workspace, **tutorial**, and **tutorial.pbl** in the system tree, you can see **w_main** listed under it.

3.6.1.3 Change the size of the frame window

Now you change the size of the application's frame window. When you run the application, the frame window is displayed in the size that you specify.

The **Window** painter should still be open for the **w_main** window. If it is not, you can double-click the **w_main** window in the **System Tree** to open it in the **Window** painter.

1. Click the **Other** tab in the **Properties** view.
2. Type **4672** in the **Width** text box and **2950** in the **Height** text box.

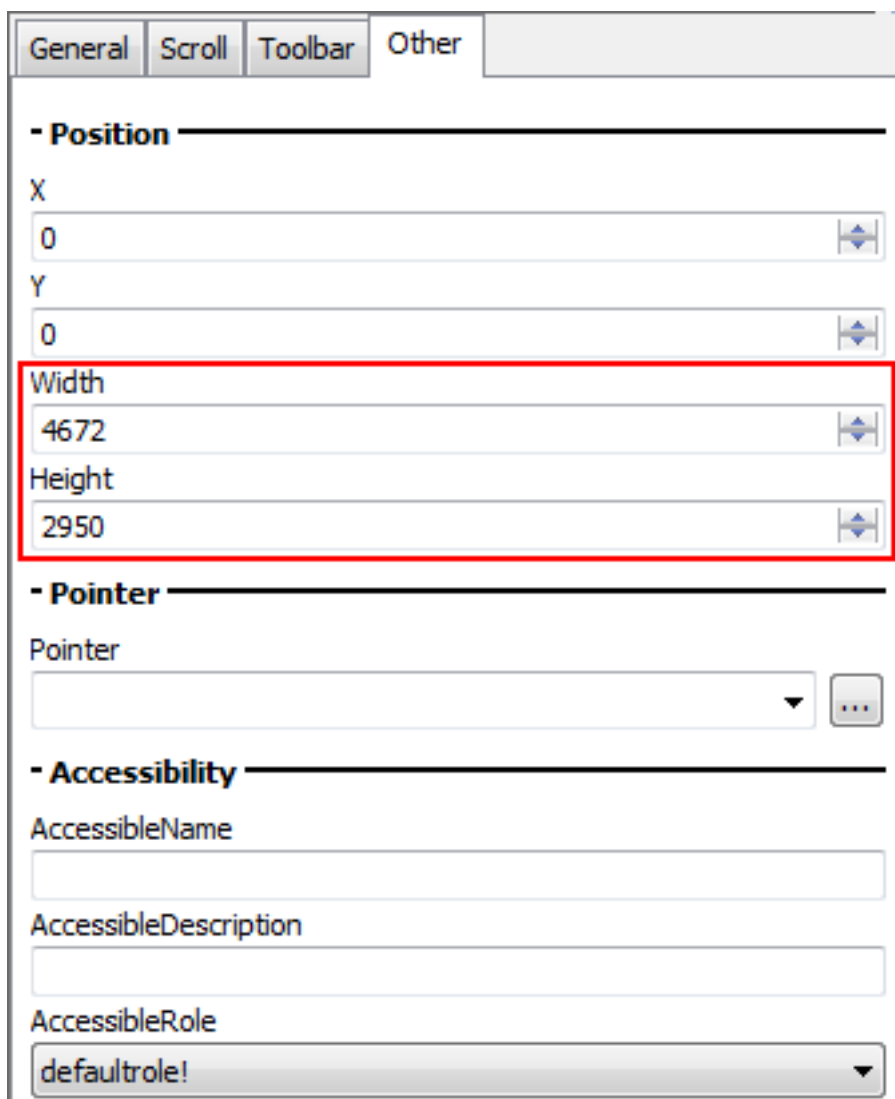
Press the **Tab** key.

The size of the window rectangle in the **Layout** view changes. The values you type are in PowerBuilder Units (PBUs).

Other methods for entering position properties

You can use the spin controls to enter values instead of typing them. Alternatively, you can change the size of the window in the **Layout** view by moving the pointer to the bottom or right edge of the window. When it turns into a double-headed arrow, you can drag the arrow to change the window size.

Figure 3.32: Change the size of w_main



3. Select **File > Close** from the PowerBuilder menu.

Click **Yes** when you are prompted to save your changes.

The **Window** painter closes.

3.6.1.4 Write scripts to open the frame window

Now you add a one-line script to open the w_main window as soon as the application starts executing.

In this exercise you:

- [Modify the frame window Open event](#)
- [Compile the script](#)

3.6.1.4.1 Modify the application Open event

Now you write scripts in the Application object Open event to open the w_main frame window.

1. Expand the **tutorial** workspace, the **tutorial** target, and **tutorial.pbl** in the system tree, double-click the **tutorial** Application object.

or

Right-click the **tutorial** Application object and select **Edit** from the pop-up menu.

The **Application** painter opens and displays the **Open** event in the **Script** view.

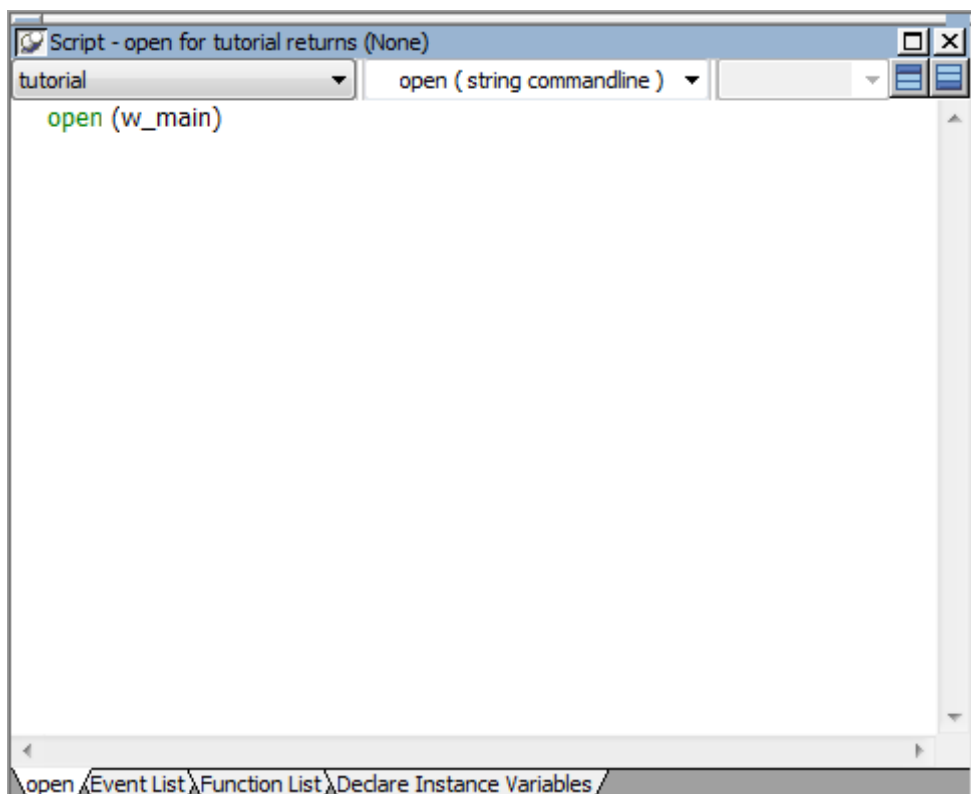
Using the Script view

The Script view has three drop-down list boxes. The first drop-down list box displays the list of available controls for the current object plus two special entries, Functions and Declare. The contents of the second drop-down list box depend on the selection in the first drop-down list box. The third drop-down list box contains all ancestor objects of the current object, if any.

2. Type the following script in the **Script** view:

```
open(w_main)
```

This calls the Open function to display the w_main frame window you created.

Figure 3.33: Scripts in the Application object Open event

3. Select **File** > **Save** from the menu bar.

3.6.1.4.2 Compile the script

Now you compile the script you just typed. In this exercise, you use a pop-up menu item to compile the script. PowerBuilder also compiles a script when you close the **Script** view or when you select a different object, event, or function for display in the **Script** view.

Handling errors in scripts

When there is an error in a script, an error window is displayed at the bottom of the Script view with the line number of the error and the error message.

To find an error

Click on an error message to move the cursor to the line that contains that error. After you correct the error, you can try to compile the script again.

Commenting out errors

PowerBuilder does not save scripts that have errors. If you want to save a script that has errors, select the entire script and click the **Comment** button to comment out the code. You can come back later, uncomment the code, and fix the problem.

-
1. Right-click anywhere in the **Script** view script area. Select **Compile** from the pop-up menu.

The script compiles.

2. Close the **Application** painter by clicking the close button in PainterBar1.

Click **Yes** when you are prompted to save your changes.

3.6.2 Build an MDI sheet window

Now you create a child window and use it as the MDI sheet window for the application.

- [Create an MDI sheet window](#)
- [Change the size of the sheet window](#)
- [Add controls to the sheet window](#)
- [Write scripts to connect with the Appeon Sample database](#)
- [Write menu scripts to open the sheet window](#)

3.6.2.1 Create an MDI sheet window

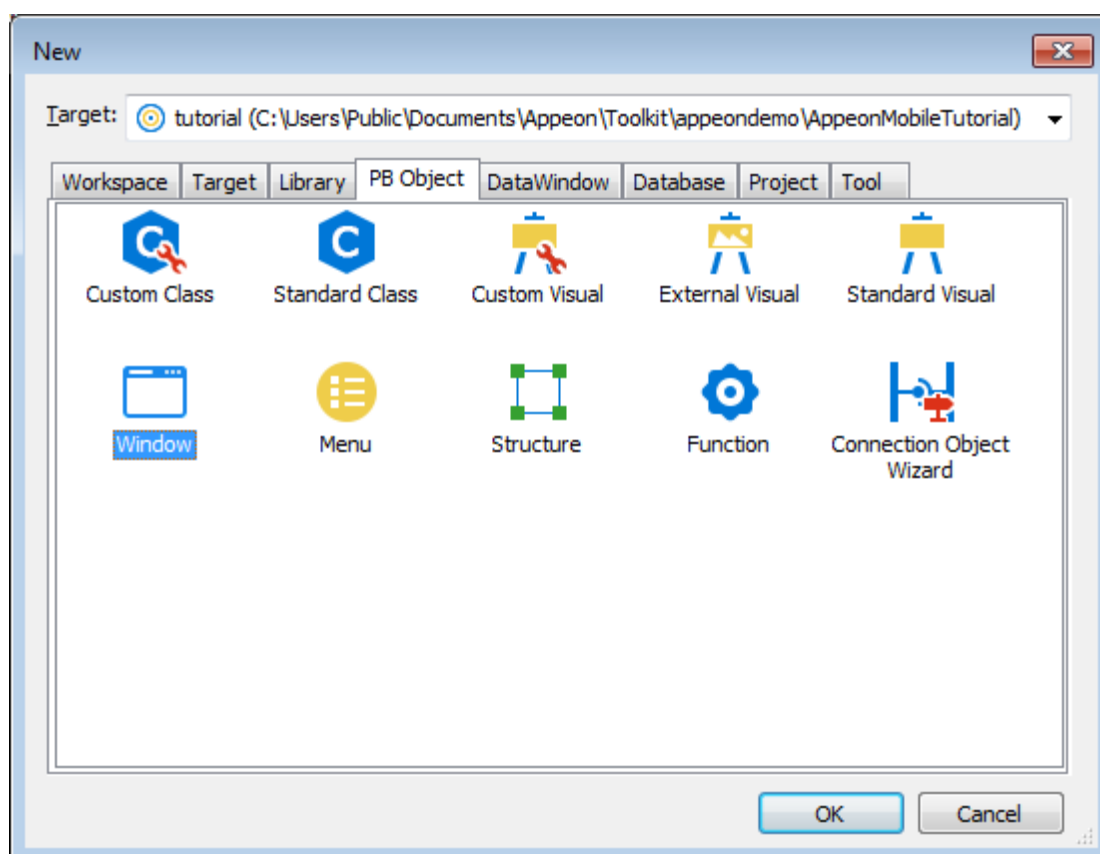
1. Click the **New** button in the PowerBar.

The **New** dialog box is displayed.

2. Click the **PB Object** tab.

Select the **Window** icon and click **OK**.

Figure 3.34: Create a sheet window



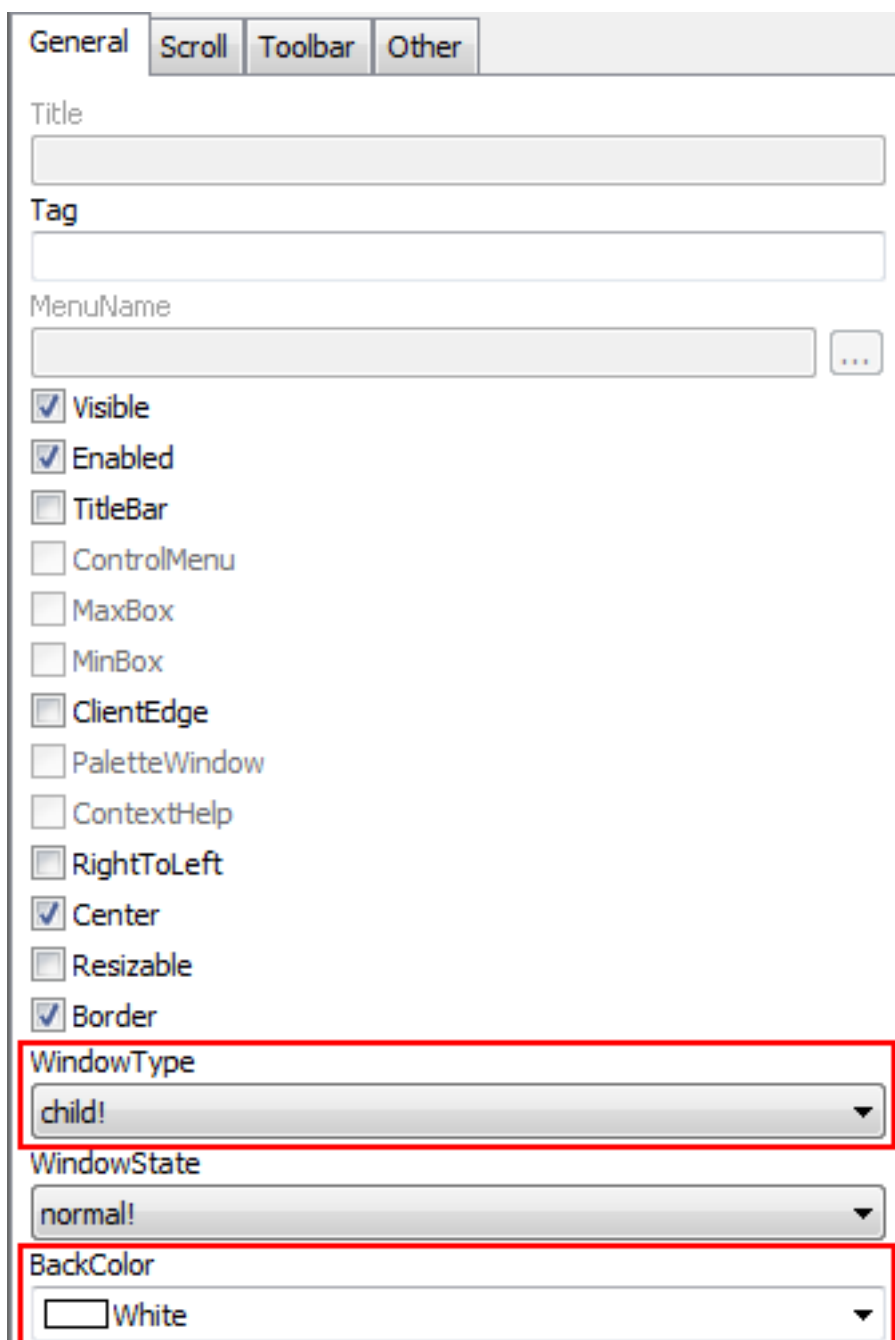
3. Select *child!* in the **WindowType** list box on the **General** page in the **Properties** view.

Child window can only be opened from within another window, such as the MDI window, main window etc. Leave the **TitleBar** check box unchecked so the child window has no title bar.

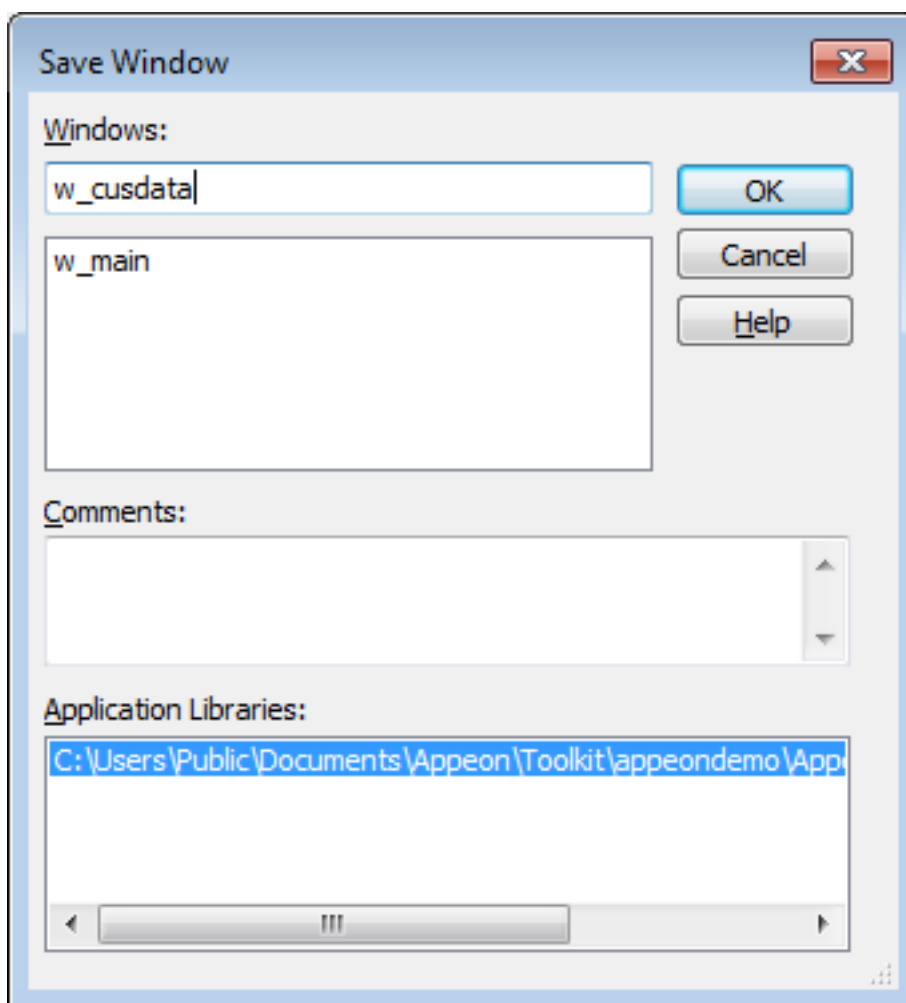
Select **White** in the **BackColor** list box.

The window is displayed with white background in the **Layout** view.

Figure 3.35: Child window properties



4. Select **File > Save** from the menu bar.
5. Type *w_cusdata* for the window name.

Figure 3.36: w_cusdata

6. Click **OK**.

PowerBuilder saves the child window. If you expand the **tutorial** workspace, the **tutorial** target, and **tutorial.pbl** in the system tree, you can see **w_cusdata** listed under it.

3.6.2.2 Change the size of the sheet window

Now you change the size of the **w_cusdata** window. When you run the application, the window is displayed in the size that you specify.

1. Open **w_cusdata** in the **Window** painter if it is not already open.
2. Click the **Other** tab in the **Properties** view.
3. Type **4672** in the **Width** text box and **2836** in the **Height** text box.
The size of the window rectangle in the **Layout** view changes.
4. Select **File > Close** from the PowerBuilder menu.
Click **Yes** when you are prompted to save your changes.
The **Window** painter closes.

When you run the application, the w_cusdata window will be sized as you specified.

3.6.2.3 Add controls to the sheet window






Controls allow users to interact with PowerBuilder objects, such as windows and DataWindows. You set properties of these controls and later add script for control events and functions.

Selecting a control button from the PainterBar

You can add controls from the **Insert** menu or by selecting a control button in PainterBar1. Unless you customize your toolbars, there is only one control button that appears in the PainterBar. When you first open a painter, PainterBar1 includes the CommandButton control button, which has a down arrow to its right. Clicking the down arrow displays a drop-down list of control buttons.

Some of the controls you can select from the drop-down list are described in the table below:

Table 3.1: Controls

Button appears	Control type	Use in tutorial
	CommandButton	Default icon for the control button in PainterBar1. You add command buttons later in this section.
	Picture	To add a picture to the sheet window.
	StaticText	To add text labels to the sheet window.
	SingleLineEdit	To add user entry text boxes to the sheet window.
	DataWindow	To add a DataWindow to the sheet window.

After you select a control, it appears in place of the CommandButton button on PainterBar1.

Now you modify the sheet window you just created by adding controls and changing some of their properties. You:

- [Add a DataWindow control](#)
- [Attach the DataWindow object with the DataWindow control](#)
- [Add CommandButton controls](#)
- [Specify properties of the CommandButton controls](#)
- [Write scripts to retrieve/insert/delete/update data](#)

3.6.2.3.1 Add a DataWindow control

Now you add a DataWindow control to the w_cusdata window.


1. Double-click **w_cusdata** in the System Tree.

The w_cusdata window opens in the **Window** painter.

2. Make sure the **Layout** view is visible in the **Window** painter.

- 3.



Select the **DataWindow** control () from the drop-down list of controls

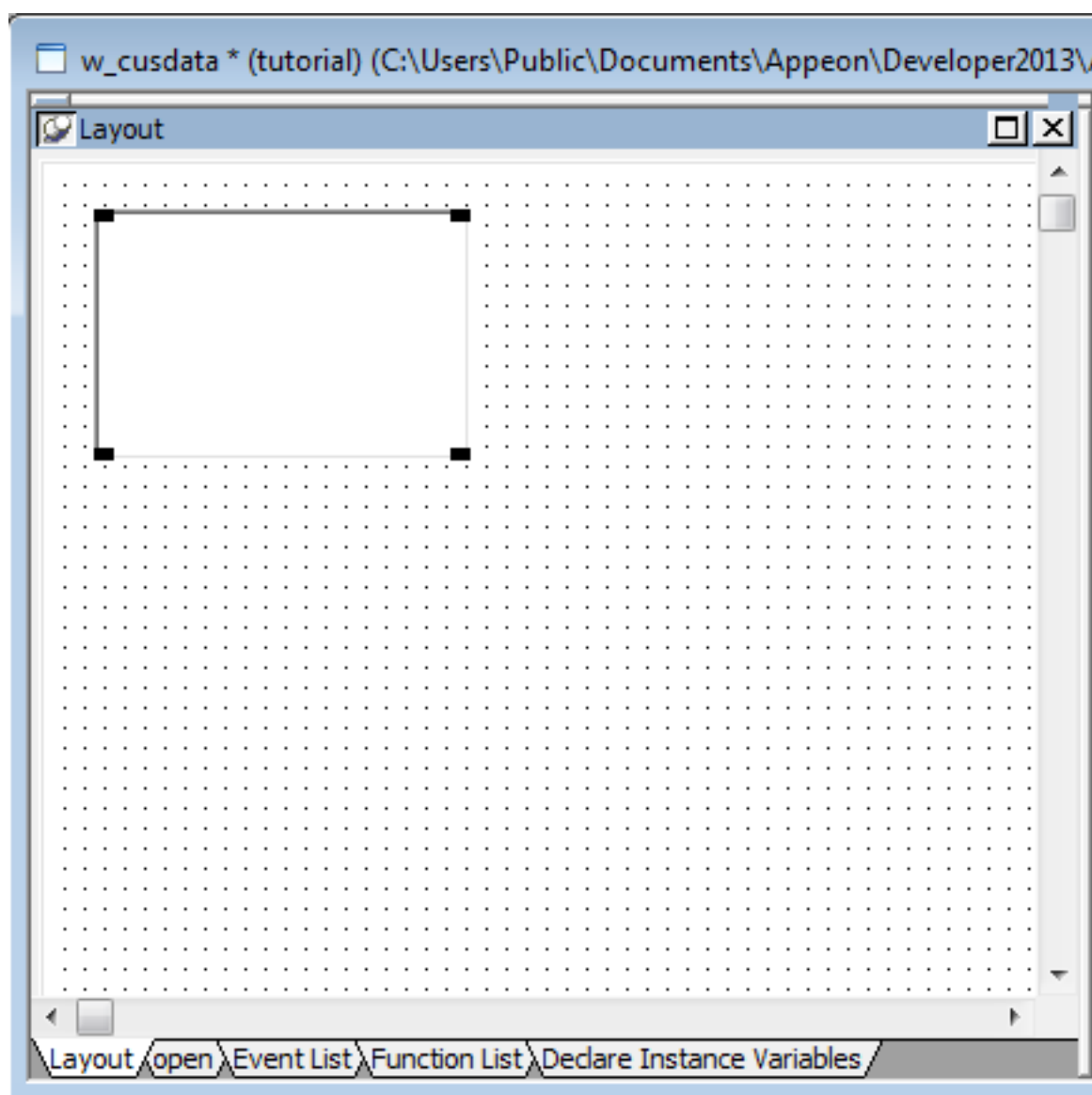
or

Select **Insert > Control > DataWindow** from the menu bar.

Unless you customize your toolbars, there is only one control button that appears in the PainterBar. When you first open a painter, PainterBar1 includes the CommandButton control button, which has a down arrow to its right. Clicking the down arrow displays a drop-down list of control buttons.

4. Click inside the rectangular window in the **Layout** view.

A DataWindow control is displayed where you clicked in the **Layout** view. At the same time you add the control, the **Properties** view switches from displaying the window properties to displaying the control properties.

Figure 3.37: DataWindow control

How to delete controls

If you add a control to the window and later decide you do not want it, select the control and press the **Delete** key. This deletes the control and its scripts.

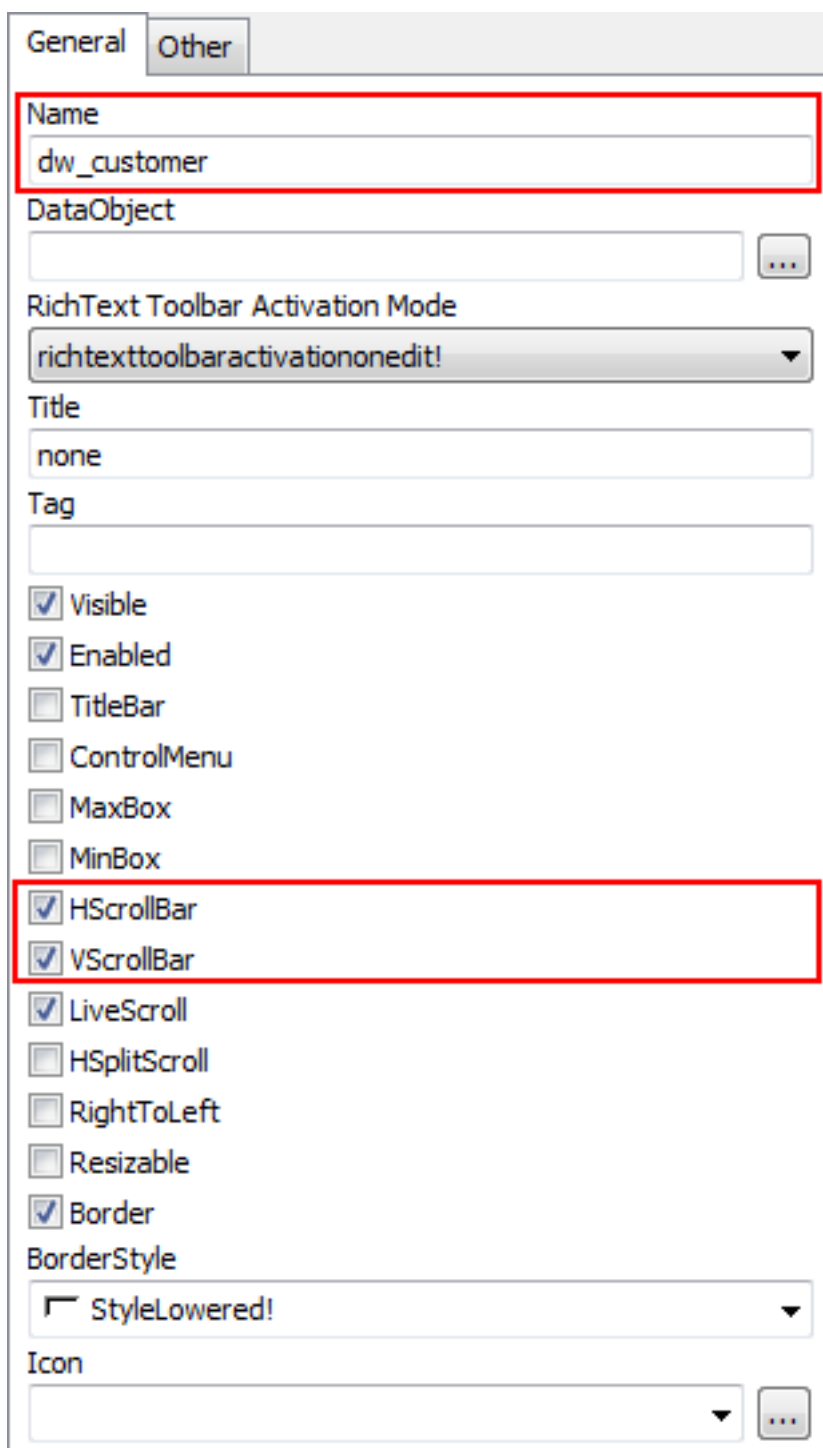
5. Make sure the new control is selected in the **Layout** view.

Small black squares at the corners indicate that the control is selected. The **Properties** view displays the properties of the selected control.

6. Select the text `dw_1` in the **Name** text box on the **General** tab of the **Properties** view.
Type `dw_customer` in the **Name** text box.
Select the **HScrollBar** and **VScrollBar** checkboxes.

PowerBuilder adds a horizontal scroll bar and a vertical scroll bar to the control. It also changes its name to `dw_customer`. The prefix `dw_` is standard for DataWindow controls.

Figure 3.38: DataWindow control properties



7. Click the **Other** tab in the **Properties** view.

Type *50* in the **X** text box, *45* in the **Y** text box, *4560* in the **Width** text box and *2526* in the **Height** text box.

You change the position and size of the DataWindow control. You might want to adjust its size again after you preview the window.

8. Select **File > Save** from the menu bar.

3.6.2.3.2 Attach the DataWindow object with the DataWindow control

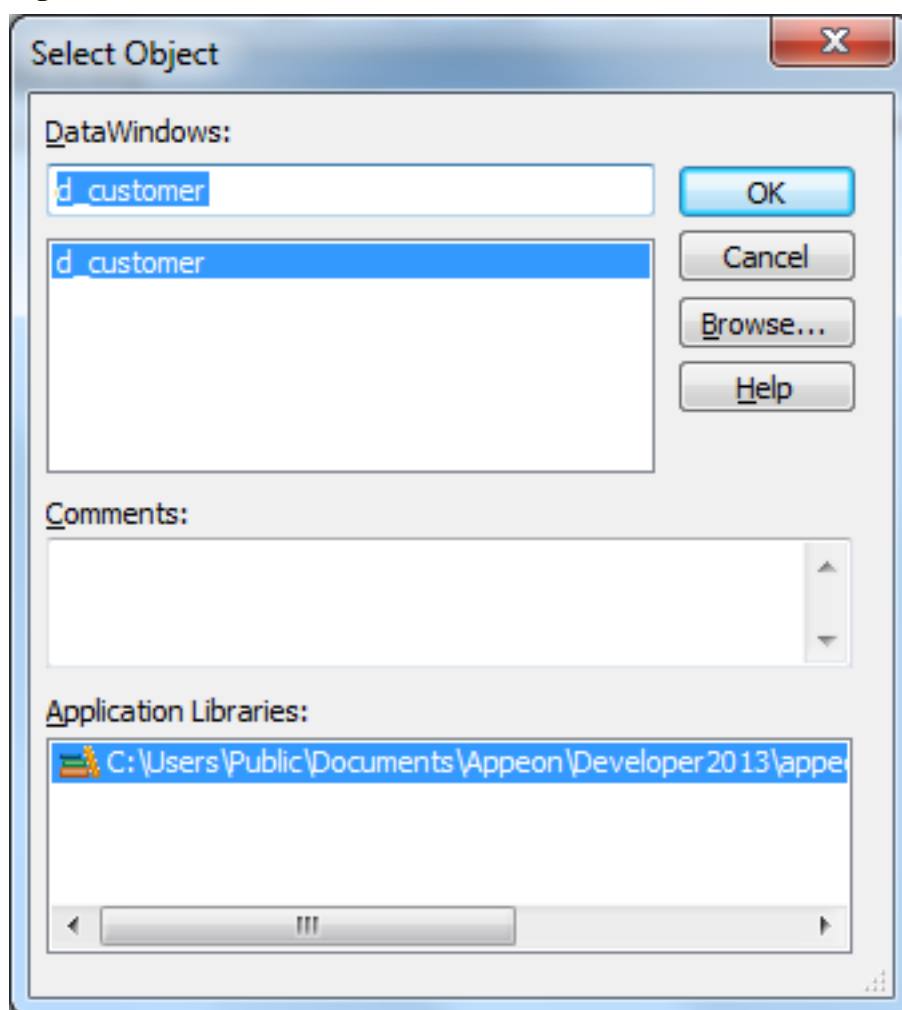
Now you attach the d_customer DataWindow object to the dw_customer DataWindow control.

1. Make sure the dw_customer DataWindow control is selected in the **Layout** view.
2. Click the **General** tab of the **Properties** view.
3. Click the ellipsis button next to the **DataObject** box.

The **Select Object** dialog box appears.

4. Select **d_customer** in the **DataWindows** list box and click **OK**.

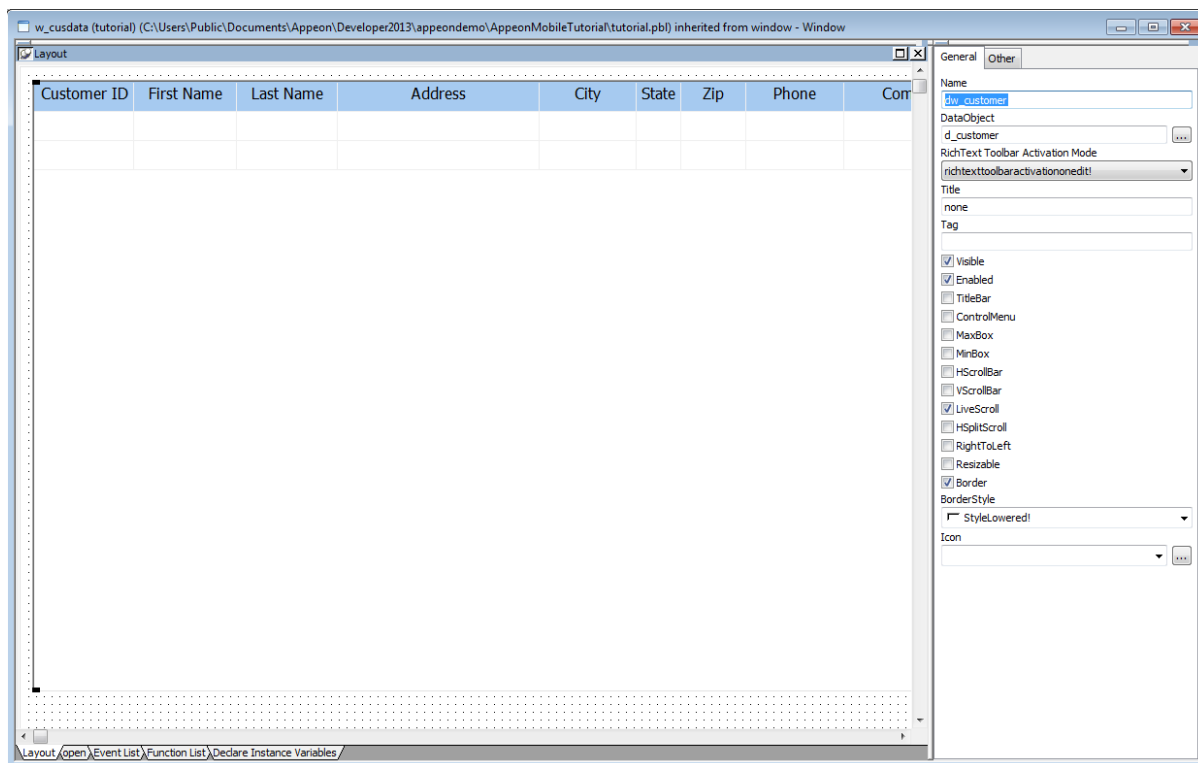
Figure 3.39: d_customer



PowerBuilder associates the d_customer DataWindow object with the dw_customer DataWindow control.

The **Layout** view now shows the `d_customer` DataWindow headings inside the `dw_customer` control, but you do not see any data yet. The DataWindow does not execute its `SELECT` statement until you run the application.

Figure 3.40: `dw_customer` added in `w_cusdata`



Adding DataWindow objects to the window using drag and drop

In this tutorial, you use a standard DataWindow control. Alternatively, you can just simply select the DataWindow object you want from the System Tree and drag it onto the window in the **Layout** view. PowerBuilder creates the DataWindow control for you.


5. Select **File > Save** from the menu bar.

3.6.2.3.3 Add CommandButton controls

Now you add CommandButton controls to the `w_cusdata` window. Later you define scripts that execute when users click these buttons.

- 1.



Select the **CommandButton** control () from the drop-down list of controls

or

Select **Insert > Control > CommandButton** from the menu bar.

2. Click below the DataWindow control at the bottom of the `w_cusdata` window in the **Layout** view.

- A **CommandButton** control is displayed where you clicked.
3. Right-click the **CommandButton** control and select **Duplicate** from the pop-up menu. PowerBuilder creates a duplicate of the selected control to the right. Repeat this step to create another two duplicates.
 4. Adjust the location of these four controls as needed so that there is some space between them.

Selecting an alignment tool from the PainterBar

You can access a drop-down list of alignment tools by clicking the **Align** button on **PainterBar2**.

5. Select **File >Save** from the menu bar.

3.6.2.3.4 Specify properties of the CommandButton controls

Now you define the properties of the **CommandButton** controls.

1. Select the first **CommandButton** control.

The **Properties** view displays the properties of the **CommandButton** control.

2. Type *cb_retrieve* in the **Name** text box on the **General** page of the **Properties** view.

Type *Retrieve* in the **Text** box.

This step changes the default name of the control to something more descriptive and adds a text label (*Retrieve*) to the button.

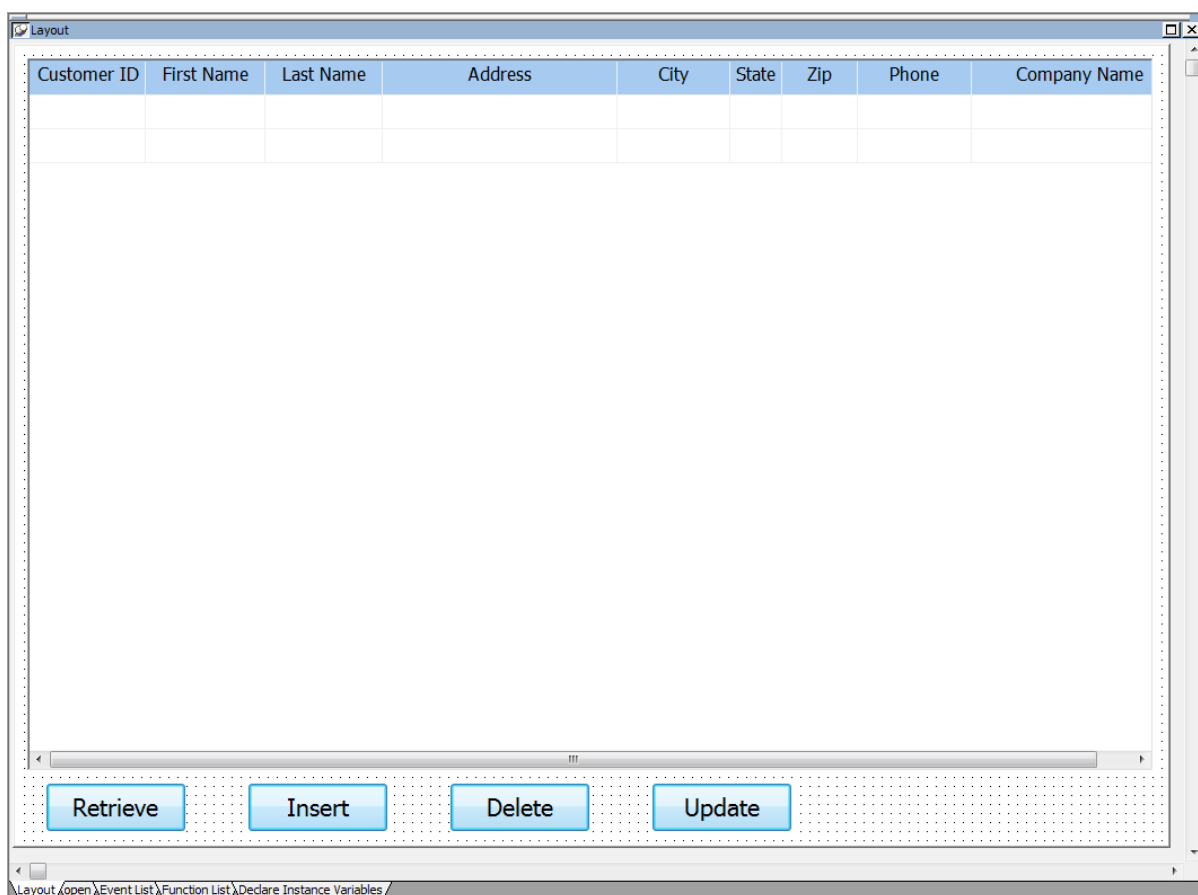
3. Select the second, the third and the fourth **CommandButton** control successively.

Type *cb_insert*, *cb_delete*, and *cb_update* in the **Name** text box, and type *Insert*, *Delete*, and *Update* in the **Text** box respectively.

4. Select these four **CommandButton** controls, and type *16* in the font size box in the **StyleBar**.

5. Select **File >Save** from the menu bar.

The window should now look like this in the **Layout** view:

Figure 3.41: Layout view of window and controls

3.6.2.3.5 Write scripts to retrieve/insert/delete/update data

Now you add scripts to retrieve, insert, delete, and/or update data in the DataWindow.

1. Double-click the **Retrieve** CommandButton in the **Layout** view

or

Select **cb_retrieve** in the first drop-down list box of the **Script** view.

The Clicked event should be the selected event in the second drop-down list box. If it is not, select it. The Clicked event script is empty.

2. Type these lines in the script area for the **Clicked** event.

```
dw_customer.setTransObject(SQLCA)
IF dw_customer.Retrieve () <> -1 THEN
  dw_customer.SetFocus ()
END IF
dw_customer.retrieve()
```

These lines retrieve data for the DataWindow control.

3. Select **File > Save** from the menu bar.

The script should save without error. If you get an error message, make sure you have typed the control and function names correctly.

4. Select **cb_insert** in the first drop-down list box of the **Script** view.

Type the following line in the script area for the **Clicked** event.

```
dw_customer.Reset()
dw_customer.insertRow(0)
dw_customer.SetFocus()
```

These lines insert data into the DataWindow control.

Select **File > Save** from the menu bar.

5. Select **cb_delete** in the first drop-down list box of the **Script** view.

Type the following line in the script area for the **Clicked** event.

```
dw_customer.deleterow( dw_customer.getrow() )
```

This line deletes data in the DataWindow control.

Select **File > Save** from the menu bar.

6. Select **cb_update** in the first drop-down list box of the **Script** view.

Type these lines in the script area for the **Clicked** event.

```
if dw_customer.update() = 1 then
    commit;
    messageBox("Information", "The data is updated successfully!")
else
    rollback;
    messageBox("Error", "Failed to update the data!")
end if
```

These lines update the data to the database.

Select **File > Save** from the menu bar.

3.6.2.4 Write scripts to connect with the Appeon Sample database

Now you write scripts to connect with the Appeon Sample database.

1. Select **w_cusdata** from the first drop-down list box in the **Script** view.

or

Double-click the blank area inside the **w_cusdata** window in the **Layout** view.

The Open event should be the selected event in the second drop-down list box. If it is not, select it. The Open event script is empty.

2. Type these lines in the script area for the **w_cusdata** Open event:

```
SQLCA.DBMS = "ODBC"
SQLCA.AutoCommit = False
SQLCA.DBParm = "ConnectionString='DSN=AppeonSample;UID=;PWD=' "
connect;
```

These lines connect to the Appeon Sample database.

3. Select **File > Save** from the menu bar.

4. Select **File > Close** from the menu bar. The **Window** painter closes.

3.6.2.5 Write menu scripts to open the sheet window

Now you write scripts in the Open menu to open the `w_cusdata` window as the MDI sheet window.

1. Double-click **m_main** in the system tree.

The **Menu** painter displays the menu.

2. Select **m_tutorial.m_open** in the first list box in the **Script** view

or

Double-click the **Open** menu item in the WYSIWYG view.

The full name of the **Open** menu item displays in the first list box of the **Script** view. It includes the `m_tutorial` prefix to indicate that it is in the Tutorial menu.

3. Select **Clicked** in the second drop-down box if it is not already selected.

Type this line for the Clicked event:

```
openSheet(w_cusdata, "w_cusdata", w_main, 1, Layered!)
```

This calls the `OpenSheet` function to display the sheet window you created.

4. Select **File > Save** from the PowerBuilder menu bar.


PowerBuilder compiles and saves the menu scripts.

3.6.3 Run the PowerBuilder application

Now you run the PowerBuilder application to see how it works, and you can test the retrieve, insert, delete, and update capabilities of the `DataWindow`.

- 1.

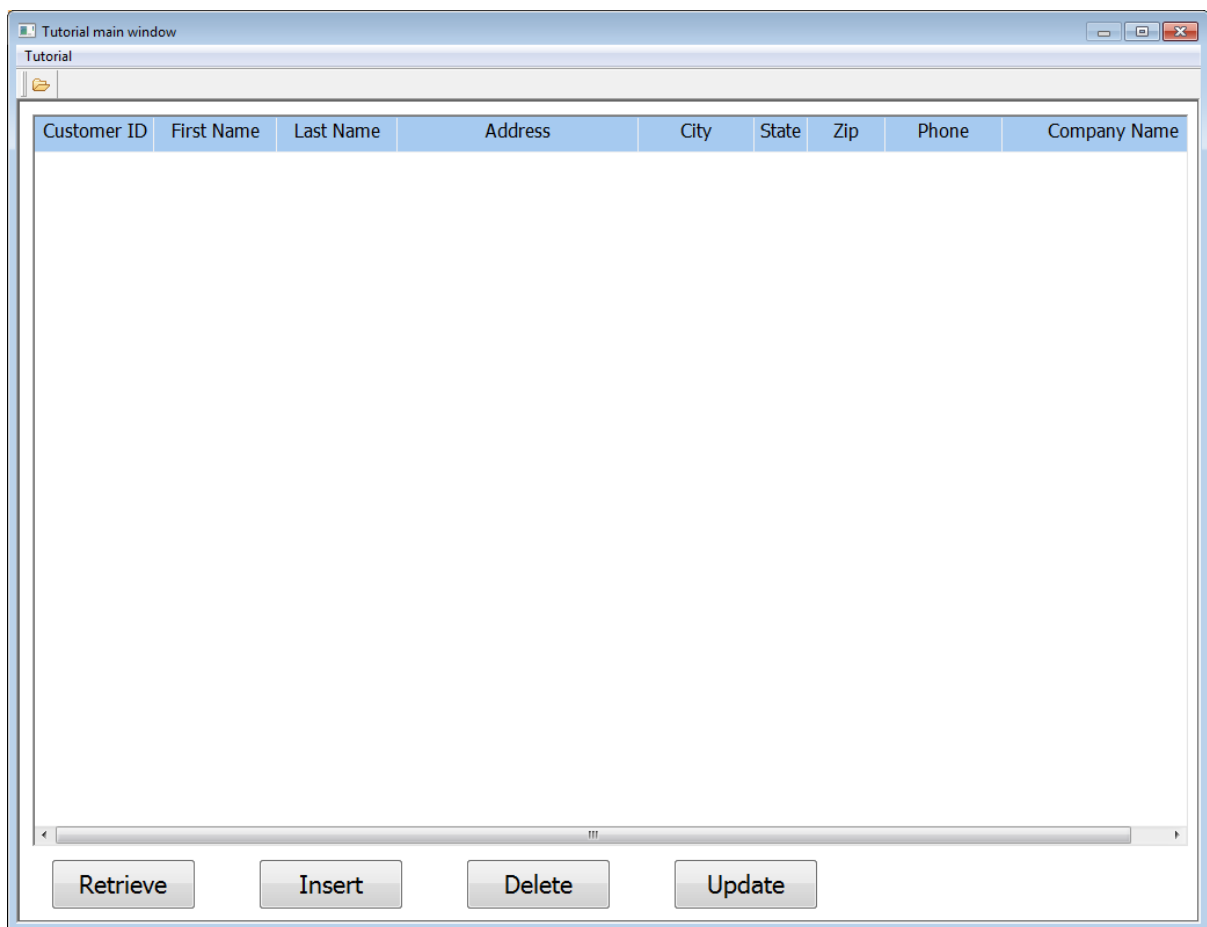


Click the **Run** button () in the **PowerBar1**.

The database connection is established, and the MDI frame for the application appears.

2. Select **Tutorial > Open** from the menu bar.

The `w_cusdata` sheet window appears.

Figure 3.42: w_cusdata window

The DataWindow control shows all of the columns retrieved from the Customer table. And you are ready to retrieve, insert, delete, and/or update data for the DataWindow control.

3. Click the **Retrieve** button.

This retrieves data from the Customer table.

Figure 3.43: Data retrieved

Customer ID	First Name	Last Name	Address	City	State	Zip	Phone	Company Name
101	Michael	Devlin	3114 Pioneer Avenue	Rutherford	NJ	07070	2015558966	The Power Group
102	Beth	Reiser	1033 Waterman Road	New York	NY	10154	2125558725	AMF Corp.
103	Erin	Nieder	1990 Windsor Street	Pittsburg	PA	19301	2155556513	Darling Associates
104	Meghan	Mason	550 Deringer Street East	Knoxville	TN	37919	6155555463	P.S.C.
105	Laura	McCarthy	1210 Highway 36	Carmel	IN	46032	3175558437	Amo & Sons
106	Paul	Phillips	2000 Cherry Creek N. Dr.	Middletown	CT	64579	2035553464	Ralston Inc.
107	Kelly	Colburn	18131 Vallco Parkway	Raleigh	NC	27695-7291	9195555152	The Home Club
108	Matthew	Goforth	11801 Wayzata Blvd.	Chattanooga	TN	37421	6155558926	Raleigh Co.
109	Jessie	Gagliardo	2800 Park Avenue	Hull	PQ	K1A 0H3	8195559539	Newton Ent.
110	Michael	Agliori	13705 North Glebe Road	Columbus	OH	43216	6145552496	The Pep Squad
111	Dylan	Ricci	14700 Prosperity Avenue	Syracuse	NY	13202	3155554486	Dynamics Inc.
112	Shawn	McDonough	15175 S Main Street	Brooklyn Park	MN	55428	6125555603	McManus Inc.
113	Samuel	Kaiser	404 Bristol Street	Minneapolis	MN	55041	6125553409	Lakes Inc.
114	Shane	Chopp	9925 Summer Street	St Paul	MN	55104	6125556453	Howard Co.
115	Shannon	Phillips	20555 Cory Road	St Paul	MN	55114	6125556425	Sterling & Co.
116	Brian	Gugliuzza	39111 Wyman Street	Mamaroneck	NY	10543	9145553817	Sampson & Sons
117	Meredith	Morgan	9191 Galveston Drive	Westerville	OH	43081	6145558989	Square Sports
118	Kristina	Sanford	2196th Street	Raleigh	NC	27695-7291	9195555152	Raleigh Active Wear
119	Tomm	Smith	3 Post Oak Blvd.	South Laguna	CA	92677	7145554996	Ocean Sports

4. Click the **Insert** button.

This clears (resets) the DataWindow, allowing you to add information for a new row that you will insert into the data source. The cursor is in the Customer ID box in the DataWindow control.

5. Add a new customer row by entering information in the boxes in the DataWindow.

Typing information for a new customer

The Customer ID number must be unique. To avoid duplicate numbers, use a four-digit number for your new database entry, or scroll down the list of three-digit customer numbers in the DataWindow and select an ID number that does not appear in the list.

Enter values for the remaining fields.

The phone number and zip code use edit masks to display the information you type. You must enter numbers only for these data fields. To specify the state in which the customer resides, you must click the arrow next to the state column and select an entry from the drop-down list box.

6. Click the **Delete** button.

The customer is deleted from the DataWindow immediately but is not deleted from the database unless you select the Update option. In this particular situation, the Update operation may fail, because rows in other tables in the database may refer to the row that you are trying to delete.

You should be able to delete any row that you have added to the database.

7. Click the **Update** button.

This sends the new customer data to the database and displays a confirmation message, as coded in the script for the ue_update event.

8. Click **OK** in the message box.
9. Click the **Close** button on the top right corner of the window to exit the application.

The application terminates and you return to the **Menu** painter.


10. Close the **Menu** painter.

4 Deploying the Mobile Application

This tutorial will walk through steps for deploying the PowerBuilder application to be a mobile application.

4.1 Configuring and deploying the application in PowerServer Toolkit

This part describes how to configure and deploy the mobile application using the PowerServer Toolkit.

1. On the PowerServer Toolkit, click the **Config Wizard** ( icon).
2. In the welcome window, click **Next**.
3. Do the following in the **Configure basic settings** window.
 - a. Type *tutorial* as the application profile name in the **Application Profile Name** text box.

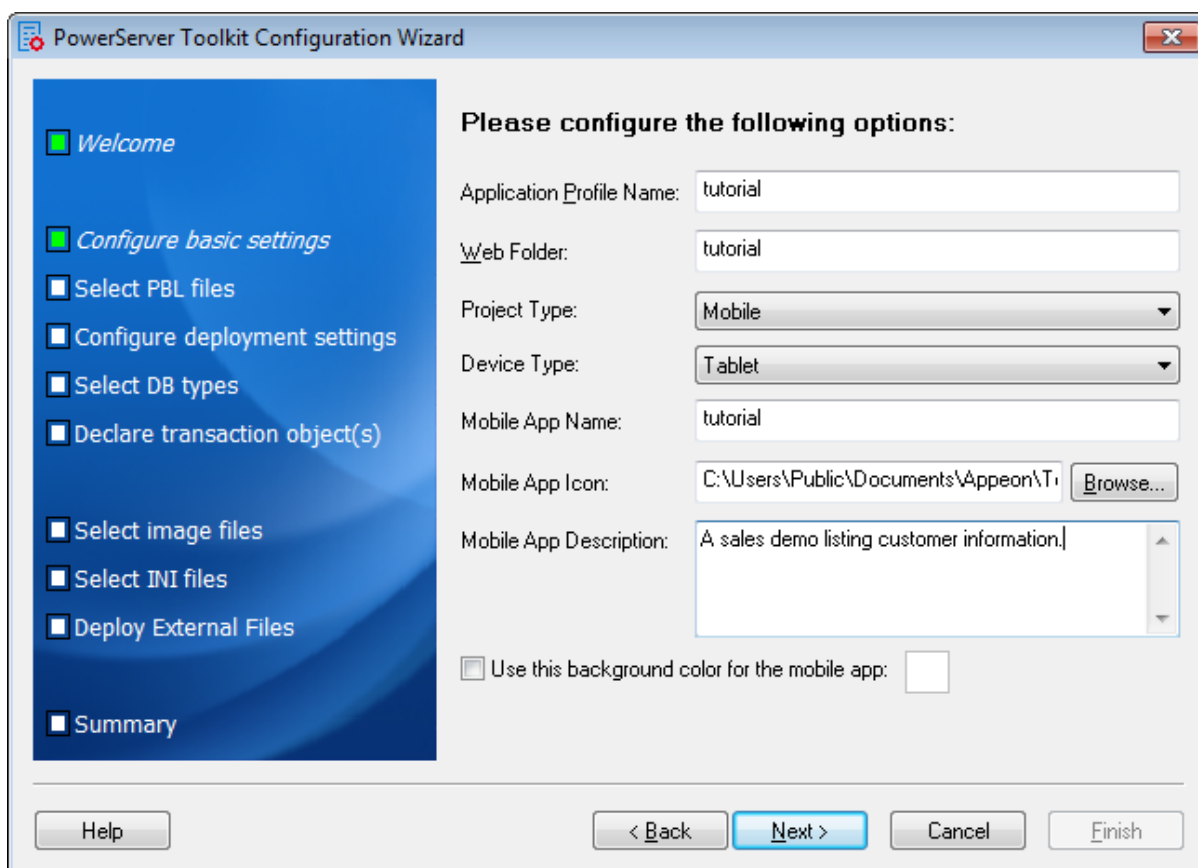
The same text will be displayed in the **Web Folder** text box and the **Mobile App Name** text box. You can modify these fields to use different text.
 - b. Select **Mobile** from the **Project Type** dropdown list box.
 - c. Select **Tablet** from the **Device Type** dropdown list box.

For applications that are designed for tablets, select **Tablet**; for applications that are designed for smartphones, select **Smartphone**; for applications that are designed for both devices, select **Both**.
 - d. Click **Browse** next to **Mobile App Icon** and select an image file.

The image will be displayed as the application icon in the Apeon Workspace. If you leave this field blank, PowerServer Mobile will display the default icon for the app.

The image file is recommended to be 172 x 172 pixels (or above) at PNG or JPG format. PowerServer Mobile will automatically adjust the image to fit properly.
 - e. Type the following lines as the brief description in the **Mobile App Description** text box.

A sales demo listing customer information.
 - f. Click **Next**.

Figure 4.1: PowerServer Toolkit Configuration Wizard

4. Do the following in the **Select PBL files** window:

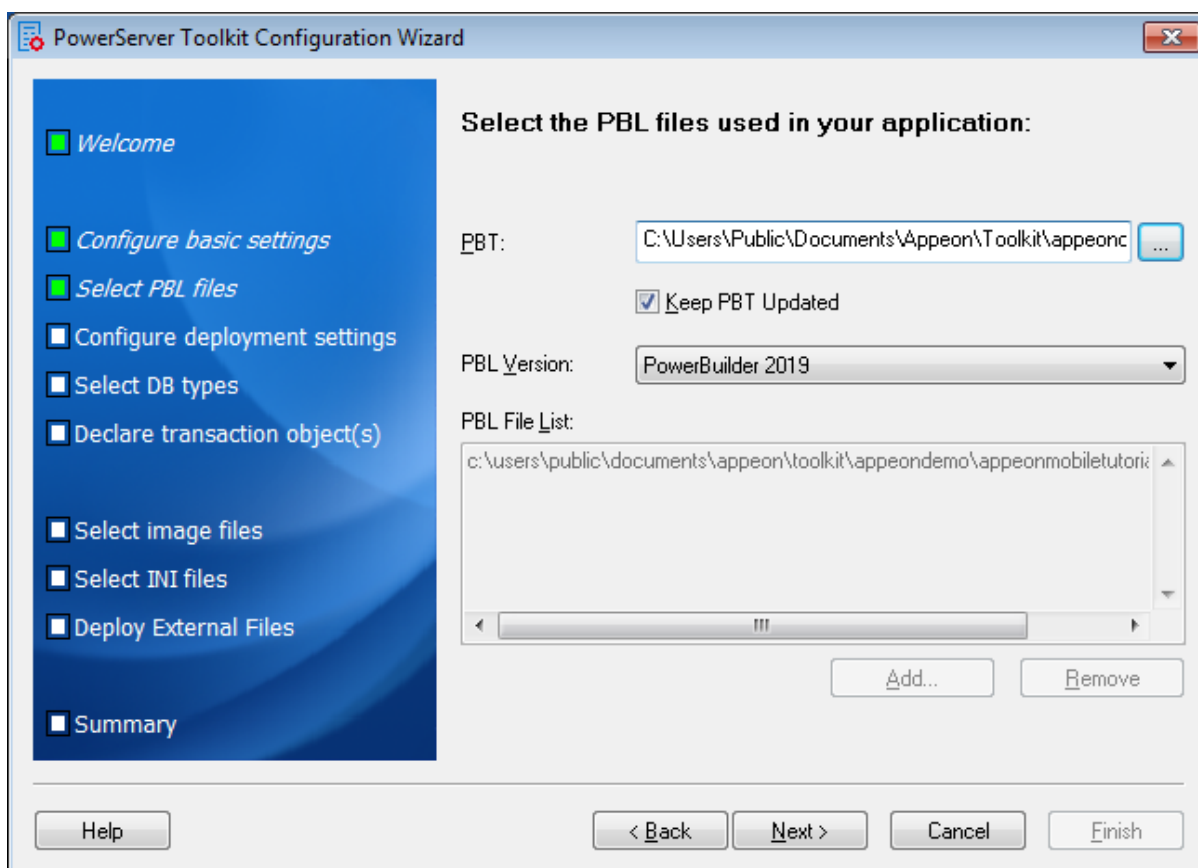
- a. Add the **tutorial.pbt** by clicking the ellipsis button next to the **PBT** dropdown list box (and all related PBLs will be automatically added for you, if multiple PBLs are used)

or

Click **Add**, navigate to *C:\Users\Public\Documents\Appeon\Toolkit\appeondemo\AppeonMobileTutorial*, (make sure the **PBL Files (*.pbl)** type is selected in the **Files of type** dropdown list box) select **tutorial.pbl**, and click **Open** to add the *tutorial.pbl*.

If multiple PBLs are used in your application, you can select multiple PBLs once by using the **Ctrl+click** technique, and click **Open** to add multiple PBLs.

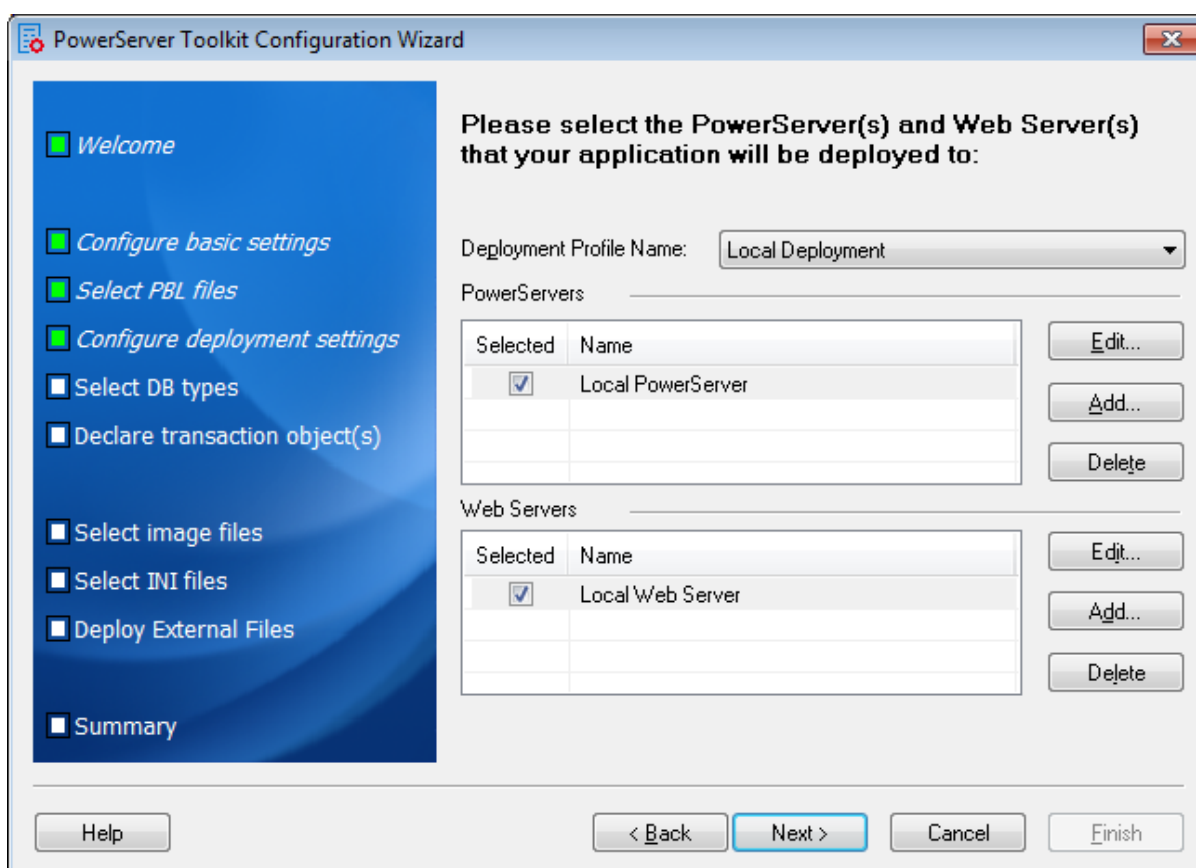
- b. Select **PowerBuilder 2019 R2** from the **PBL Version** dropdown list box.
- c. Click **Next**.

Figure 4.2: Add the PBT in ADT Wizard

5. Keep the default settings and click **Next**.

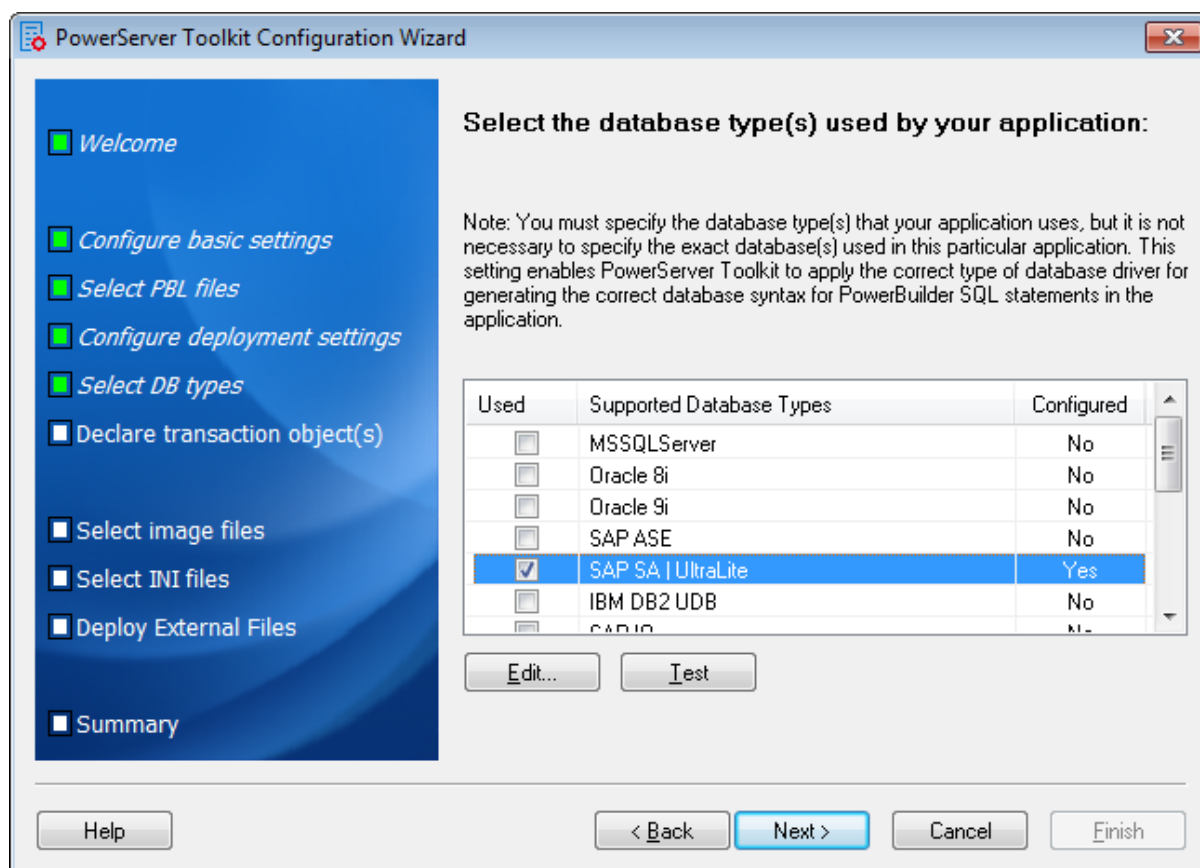
Since you have installed PowerServer Toolkit and PowerServer on the same machine, you can directly use the **Local PowerServer** profile, the **Local Web Server** profile, and the **Local Deployment** profile, all of which are configured automatically by the product setup program.

When you click **Next** the config wizard will automatically test the connection to the server. You will not be able to proceed if the connection test failed.

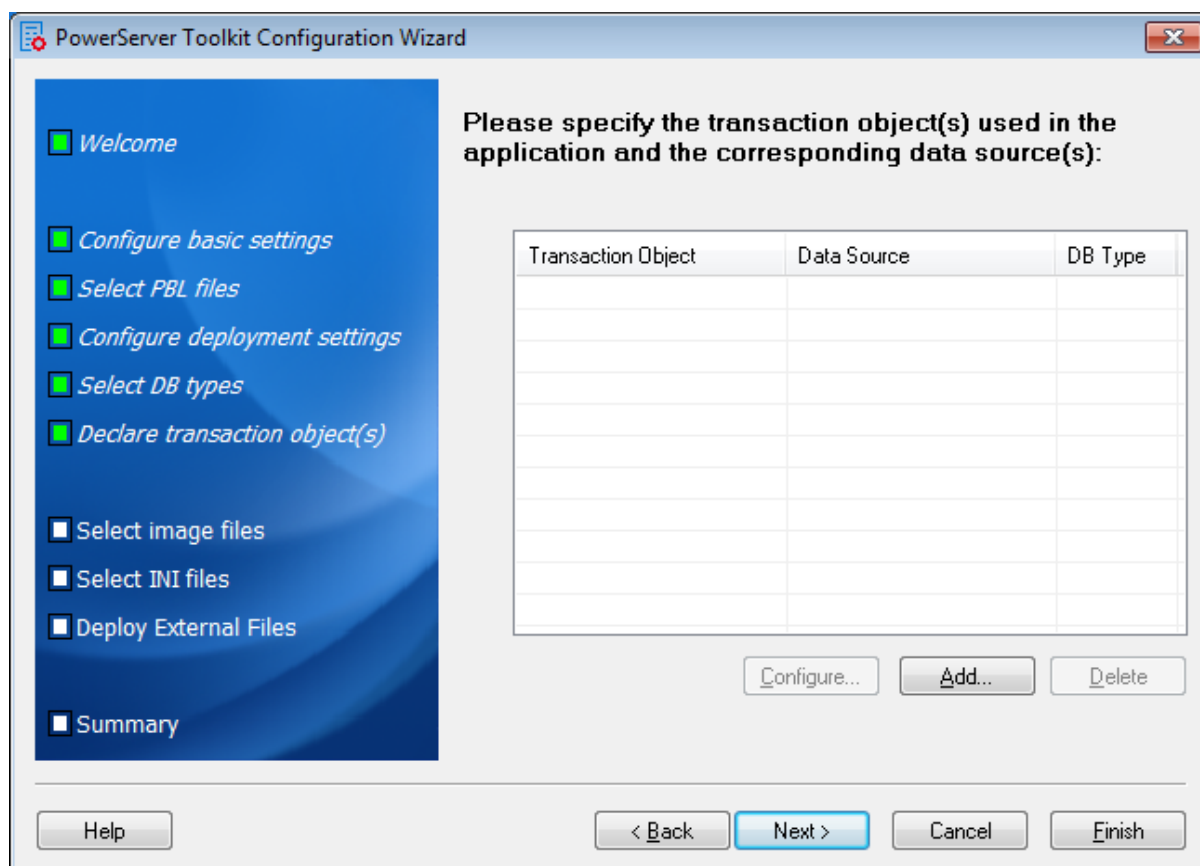
Figure 4.3: Select the PowerServer(s) and Web Server(s)

6. Select and highlight the **SAP SA | UltraLite** database type by checking the checkbox under the **Used** column, and click **Next**.

The SAP SA | UltraLite database is automatically configured (indicated with **Yes** under the **Configured** column) by Apeon PowerServer during installation, so you can use it directly.

Figure 4.4: Select the database type(s)

7. Click **Add** to specify the transaction object(s) used in the **tutorial** application.

Figure 4.5: Specify the transaction object

In the **Add Transaction Object** window, specify the following settings:

- Use the default **sqlca** as the transaction object name in the **Transaction Object** text box.
- Select **SAP SA | UltraLite** from the **Database Type** list box.
- Select **Local PowerServer** from the **PowerServer** dropdown list box.
- Select the **appeonsample** data source from the table.

The data source should connect to the same database that the PowerBuilder application connects to.

The Data Source text box will be automatically filled with *appeonsample*.

- Click **OK**.

Figure 4.6: Add transaction object

Add Transaction Object

Transaction Object: sqlca

Database Type: SAP SA | UltraLite

Data Source: appeonsample

If your PowerServer is the .NET edition, you can also select, edit, add, or delete the data source in the below Data Source group box.

Data Source

PowerServer: Local PowerServer

Selected	Name	DB Host
<input checked="" type="radio"/>	appeonsample	AppeonSampleForServer
<input type="radio"/>	appeonsample2	AppeonSample2ForServer
<input type="radio"/>	sa_db	AppeonSample
<input type="radio"/>	northwind	192.0.0.205

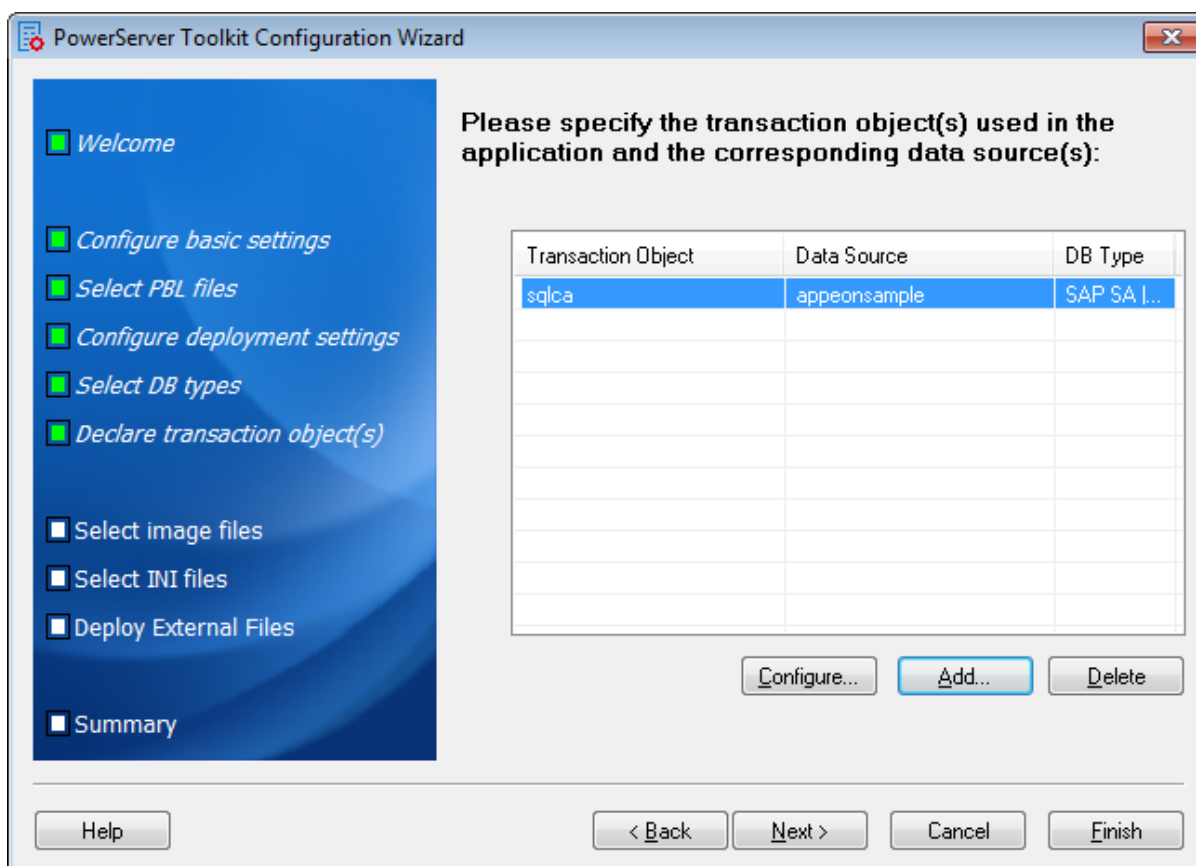
Edit... Add... Delete

OK Cancel

Tip: If you have more than one transaction object you can add additional transaction objects by repeating the above steps.

The newly added transaction object is now listed.


Click **Next**.

Figure 4.7: The added transaction object is listed

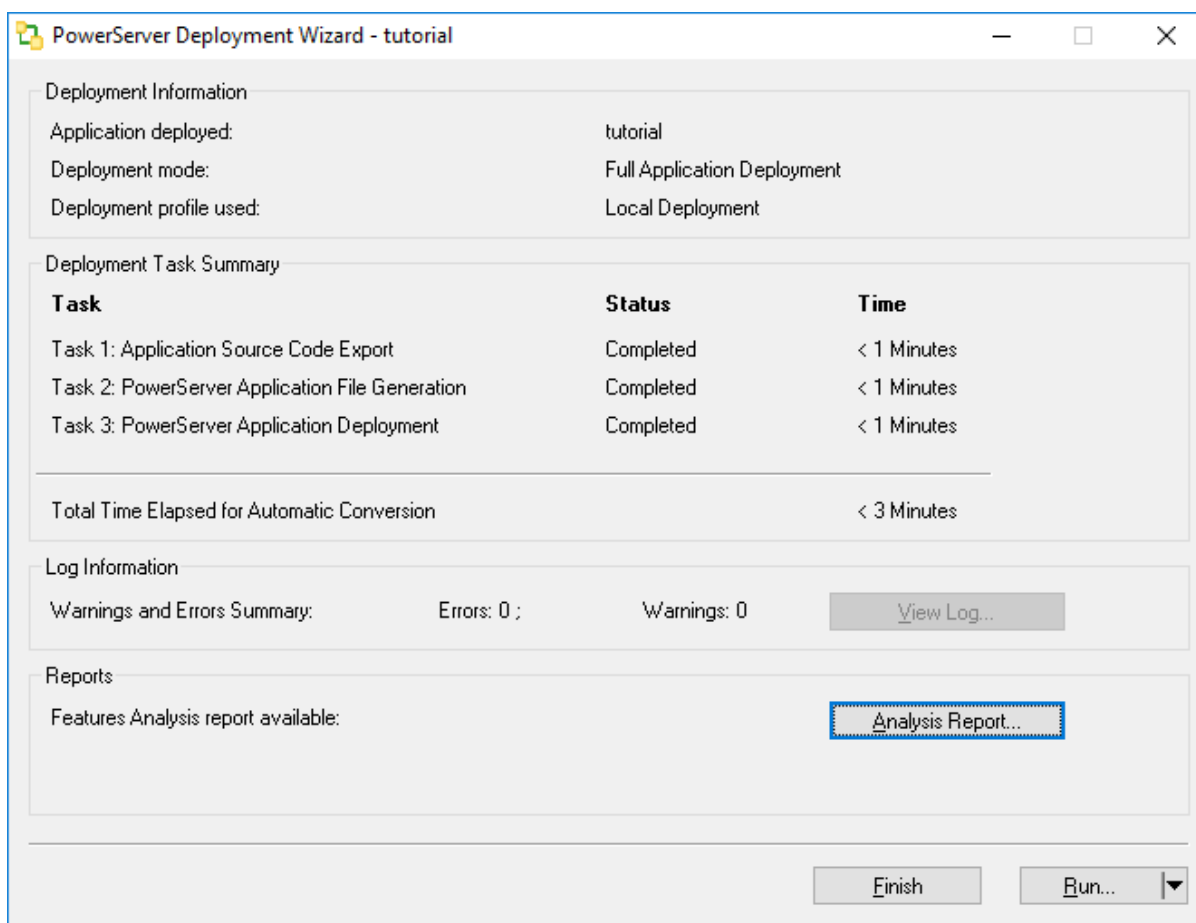
- Keep the default settings and click **Next** several times until you reach the **Summary** screen, keep the **Deploy the application now** option as selected and click **Finish**.

Once you click **Finish**, the **PowerServer Application Deployment Wizard** automatically starts deploying the application.

Manually deploy an application

To manually start **PowerServer Application Deployment Wizard**, you can click the **Deploy** icon () on the PowerServer Toolkit.

- Click **Finish** when the deployment process is complete.

Figure 4.8: PowerServer Application Deployment Wizard tutorial

4.2 Running the mobile application in Apeon Workspace

Now that you have successfully deployed the application, you are ready to run the application on a supported mobile device with Apeon Workspace installed.

1. Install Apeon Workspace (on an Android device for example).

Step 1: Make sure your Android device can connect to PowerServer.

Step 2: Enable the **Unknown resources** option (in **Settings > Security**) on the Android device so you can install apps that are not downloaded from Google Play.


Step 3: Visit the Apeon Workspace download center that is posted on PowerServer (http://server_domain:80/AWS), and then click the download button.

2. Configure the network connection of the mobile device.


Make sure that the Windows PC and the mobile device are connected to the same Wi-Fi router.

3. Tap the **ApeonMobile** icon on your mobile device to launch Apeon Workspace.



4. Tap the **New** icon (⊕) to the left of the title bar.

5. In the **App URL** text box, enter the application URL for the **tutorial** application in this format: *http://server_domain/appname*. For example, if your IIS domain is *www.abc.com* and you specified *tutorial* in the PowerServer Toolkit configuration as the Web folder name then the URL would be *http://www.abc.com/tutorial/*.
6. Tap the **Test Connection** button to test the server connections. If successful please proceed to Step 7, otherwise please enter the correct URL.
7. Tap the **Back** icon () on the title bar to save the information and return to the main screen of the Apeon Workspace.

Once you return to the main screen of the Apeon Workspace, the downloading and installation process of the **tutorial** application occurs automatically.

8. After the installation process has completed, tap the **tutorial** application icon on the home screen to run it.
9. In the **tutorial** application window, click the menu icon () , and then select **Tutorial > Open**

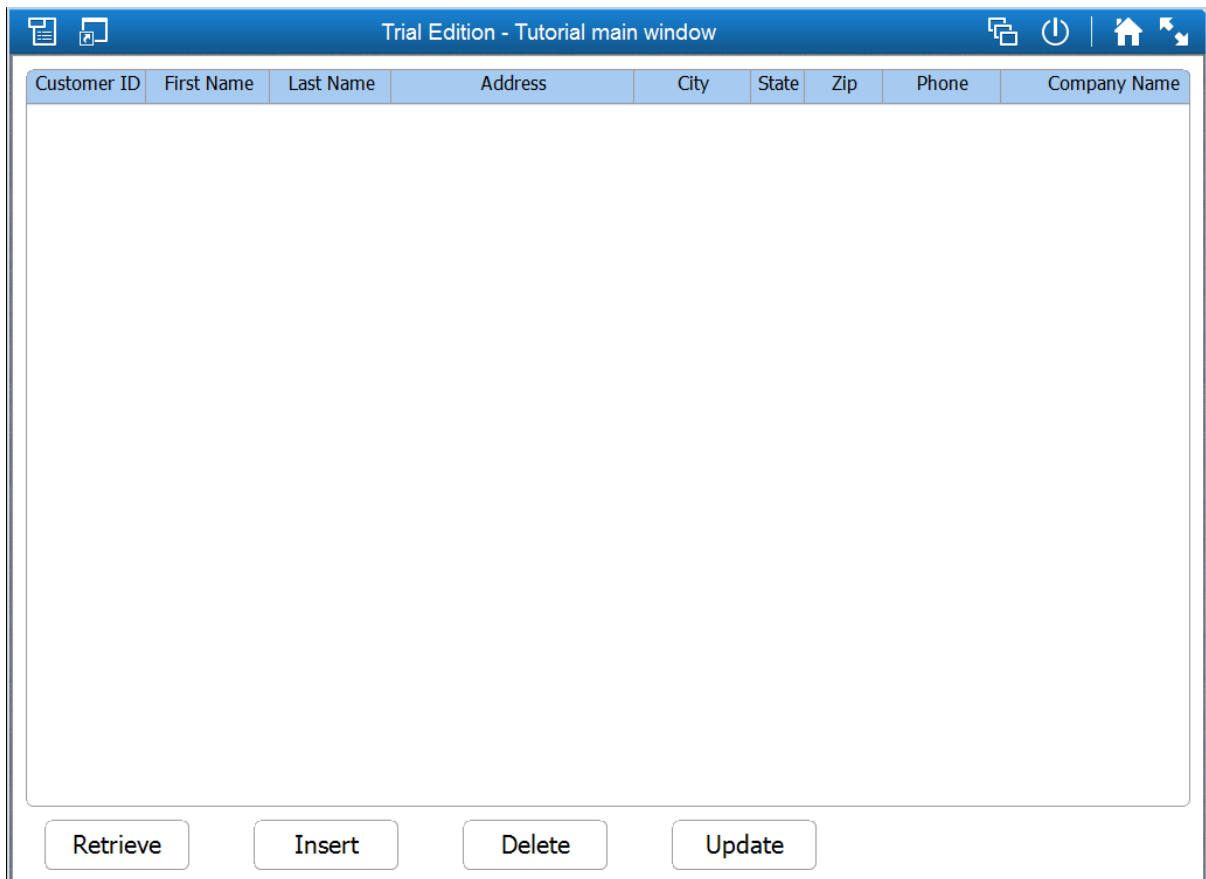
or

Click the toolbar icon () and then click the **Open** icon () .

This opens w_cusdata sheet window.

Mobile-style title bar, menu, & toolbar

The layout of title bar, menu and toolbar are automatically adjusted by PowerServer Mobile, to make more room for the window and controls.

Figure 4.9: Application run in Apeon Workspace

10. Click the **Retrieve** button.

This retrieves data from the database.

Figure 4.10: Data retrieved


Customer ID	First Name	Last Name	Address	City	State	Zip	Phone	Company Name
101	Michael	Devlin	3114 Pioneer Avenue	Rutherford	NJ	07070	2015558966	The Power Group
102	Beth	Reiser	1033 Waterman Road	New York	NY	10154	2125558725	AMF Corp.
103	Erin	Nieder	1990 Windsor Street	Pittsburg	PA	19301	2155556513	Darling Associates
104	Meghan	Mason	550 Deringer Street East	Knoxville	TN	37919	6155555463	P.S.C.
105	Laura	McCarthy	1210 Highway 36	Carmel	IN	46032	3175558437	Amo & Sons
106	Paul	Phillips	2000 Cherry Creek N. Dr.	Middletown	CT	64579	2035553464	Ralston Inc.
107	Kelly	Colburn	18131 Vallco Parkway	Raleigh	NC	27695-7	9195555152	The Home Club
108	Matthew	Goforth	11801 Wayzata Blvd.	Chattanooga	TN	37421	6155558926	Raleigh Co.
109	Jessie	Gagliardo	2800 Park Avenue	Hull	PQ	K1A 0H3	8195559539	Newton Ent.
110	Michael	Agliori	13705 North Glebe Road	Columbus	OH	43216	6145552496	The Pep Squad
111	Dylan	Ricci	14700 Prosperity Avenue	Syracuse	NY	13202	3155554486	Dynamics Inc.
112	Shawn	McDonough	15175 S Main Street	Brooklyn Park	MN	55428	6125555603	McManus Inc.
113	Samuel	Kaiser	404 Bristol Street	Minneapolis	MN	55041	6125553409	Lakes Inc.
114	Shane	Chopp	9925 Summer Street	St Paul	MN	55104	6125556453	Howard Co.
115	Shannon	Phillips	20555 Cory Road	St Paul	MN	55114	6125556425	Sterling & Co.
116	Brian	Gugliuzza	39111 Wyman Street	Mamaroneck	NY	10543	9145553817	Sampson & Sons
117	Meredith	Morgan	9191 Galveston Drive	Westerville	OH	43081	6145558989	Square Sports
118	Kristina	Sanford	2196th Street	Raleigh	NC	27695-7	9195555152	Raleigh Active Wear
119	Tomm	Smith	3 Post Oak Blvd.	South Laguna	CA	92677	7145554996	Ocean Sports
120	Gertrude	Stein	4 Amon Carter Blvd.	Elmsford	NY	10523	9145553476	Carney Co.

Retrieve Insert Delete Update

You can click the **Insert**, **Delete**, and/or **Update** buttons to insert, delete, or update the data.

Mobile-style scroll bar

The horizontal scroll bar and vertical scroll bar for the DataWindow control is automatically replaced with the mobile-style scroll bar, instead of the Windows-style scrollbar. The mobile-style scroll bar appears when you scroll in the DataWindow, and disappears when you stop scrolling.

11. Click the close app icon () on the right end of the title bar. Click **Yes** when prompted to close the application.

Index

A

Add a DataWindow control, [76](#)
Add a new toolbar for the new menu item, [55](#)
Add an extra Script view, [38](#)
Add CommandButton controls, [80](#)
add controls to the sheet window, [75](#)
Add items to the new menu and save the menu, [54](#)
attach menu to the frame window, [67](#)
Attach the DataWindow object with the DataWindow control, [79](#)

B

build a DataWindow object, [56](#)
build an MDI frame window, [63](#)
build an MDI sheet window, [72](#)

C

change the size of the frame window, [68](#)
change the size of the sheet window, [74](#)
Compile the script, [71](#)
configure & deploy application, [88](#)
Create a new menu, [54](#)
Create a new workspace, [29](#)
Create a target, [32](#)
create an MDI frame window, [63](#)
create an MDI sheet window, [72](#)
Create and preview a new DataWindow object, [56](#)
Create the database profile for the Appeon Sample database, [46](#)

D

deploy the mobile application, [88](#)
Display view title bars, [38](#)

F

Float and dock views, [39](#)
Float the toolbars, [41](#)

L

Look at table definitions in the Appeon Sample database, [49](#)
Look at the Appeon Sample database, [45](#)

M

Make cosmetic changes to the DataWindow object, [61](#)

Manipulate tabbed views, [39](#)
Manipulate the System Tree window, [35](#)
Manipulate views, [37](#)
Modify the application Open event, [70](#)

O

Open an object, [36](#)

R

Reposition the toolbars, [43](#)
Reset the default view layout scheme, [40](#)
run the application in Appeon Workspace, [96](#)
run the PowerBuilder application, [84](#)

S

Save a view layout scheme, [40](#)
Save the DataWindow object, [61](#)
Set up the toolbars, [40](#)
setting up the menus, [53](#)
Show labels on toolbar buttons, [41](#)
Specify properties of the CommandButton controls, [81](#)
Starting PowerBuilder, [29](#)

W

write menu scripts to open the sheet window, [84](#)
write scripts to connect with the Appeon Sample database, [83](#)
write scripts to open the frame window, [69](#)
Write scripts to retrieve/insert/delete/update data, [82](#)