

PowerServer Toolkit User Guide

PowerServer® 2017

FOR WINDOWS & UNIX & LINUX

DOCUMENT ID: ADC20231-01-0700-01

LAST REVISED: October 27, 2017

Copyright © 2000-2017 by Appeon Limited. 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 Limited.

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

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

Contents

1	About This Book	1
1.1	Audience	1
1.2	How to use this book	1
1.3	Related documents	2
1.4	If you need help	3
2	Introduction	4
2.1	PowerServer Toolkit in PowerBuilder	4
2.2	Web and/or mobile application development with PowerServer Toolkit	6
3	Before You Begin	7
3.1	Installing Appeon PowerServer	7
3.2	Configuring application database connectivity	7
3.3	Starting PowerServer	7
4	Configuring PowerServer Toolkit	9
4.1	Using PowerServer Toolkit Configuration Wizard	9
4.1.1	Configuring basic settings	10
4.1.2	Selecting PBL file(s)	12
4.1.3	Configuring deployment settings	13
4.1.4	Selecting DB Type(s)	14
4.1.5	Declaring transaction object(s)	14
4.1.6	Selecting image files	17
4.1.7	Selecting INI files	17
4.1.8	Selecting External files	18
4.1.9	Summary	19
4.2	Using Configure Tool	20
4.2.1	Managing application profiles	20
4.2.1.1	Application Profiles tab page	21
4.2.1.2	Application profile settings	22
4.2.2	Managing database type profiles	48
4.2.3	Managing server profiles	51
4.2.3.1	Server Profiles tab page	51
4.2.3.2	PowerServer profile settings	52
4.2.3.3	Web Server profile settings	55
4.2.4	Managing deployment profiles	60
4.2.4.1	Deployment Profiles tab page	61
4.2.4.2	Deployment profile settings	61
4.2.5	Managing data source profiles	63
5	Using UFA Tool	68
5.1	Analyzing an application	69
5.1.1	Tasks required before you perform feature analysis	69
5.1.2	Accessing the UFA tool	70
5.1.3	Performing feature analysis	71
5.1.4	Undetected Unsupported Features	73
5.2	Working with UFA Report	75
5.2.1	Modifying unsupported features	76
5.2.2	Manipulating the UFA Report	78
5.2.2.1	Opening or saving a UFA Report	78

5.2.2.2	Selecting report view mode	79
5.2.2.3	Searching for UFA Report items	79
5.2.2.4	Filtering UFA Report items	79
5.2.2.5	Specifying report display level	82
5.2.2.6	Defining the priority settings of unsupported features	83
5.2.2.7	Customizing the general settings of the UFA Report	84
6	Deploying PowerBuilder Applications	85
6.1	Deployment performance	85
6.1.1	Speed of Deployment Process	85
6.1.2	Deployment duration for full deployments	86
6.1.3	Deployment duration for incremental deployments	86
6.2	Deployment process	87
6.2.1	Preparing the PowerBuilder application	87
6.2.2	Specifying the deployment settings	87
6.2.2.1	Selecting the deployment mode	89
6.2.3	Deploying the PowerBuilder application	90
7	Debugging Appeon Web Applications	94
7.1	Important Requirements	94
7.2	Introduction to the debugging procedure	95
7.3	Starting Appeon Debugger	95
7.3.1	Views in Appeon Debugger	97
7.4	Setting breakpoints	98
7.4.1	Code lines that can be set as breakpoints	98
7.4.2	Methods for setting breakpoints	99
7.5	Running the application in debug mode	100
7.6	Examining an application at a breakpoint	100
7.6.1	Special variable and expression handlings	100
7.6.2	Adding variables or expressions to Watch view	101
7.6.3	Changing the value of a variable or expression	102
7.6.4	Evaluating an expression	103
7.6.5	Examining context in Call Stack view	103
7.6.6	Stepping through the application	103
7.7	Fixing the code/stopping the debug procedure	103
8	Running Appeon Applications	104
8.1	Requirements for running Web app	104
8.1.1	Windows account privileges	104
8.1.2	Internet Explorer settings	104
8.1.3	Language setting requirements	110
8.1.4	Disabling anti pop-up software	111
8.2	URLs of Appeon applications	112
8.3	Running Appeon applications	113
8.3.1	Launching applications from the <i>Run</i> button	113
8.3.1.1	Configuring the project loader	115
8.3.2	Installing Appeon Workspace and mobile apps on mobile device	116
8.3.3	Installing IWA apps	117
8.3.4	Selecting a run mode for the Web app	118
8.4	Appeon DataWindow menu	119

8.4.1	Enabling Apeon DataWindow menu	120
8.4.2	Using Apeon DataWindow Menu	120
8.4.2.1	Find	120
8.4.2.2	Sort and filter	121
8.4.2.3	DataWindow printing	124
8.4.2.4	SaveAs	125
8.4.2.5	Additional Enhanced Features	126
9	Using Information Manager	129
9.1	Viewing the reports and logs	129
10	Packaging Applications	132
10.1	Packaging a server deployment project	132
10.1.1	What can be packaged?	132
10.1.2	Packaging instructions	133
10.1.3	Modifying the deploy-config file	139
10.1.4	Installing and uninstalling an application to the server	140
10.1.4.1	Points to note before installation	140
10.1.4.2	Installing an application	141
10.1.4.3	Uninstalling an application	149
10.2	Preparing for the mobile package	155
10.2.1	Preparations for the iOS package	155
10.2.2	Preparations for the Android package	156
10.2.2.1	(Required) Obtain a private key	156
10.2.2.2	(Optional) Obtain a Google Maps API key	157
10.2.2.3	(Optional) Prepare the icons/images for your application	161
10.2.3	Preparations for the Apeon Workspace package	162
10.2.3.1	(Optional) Prepare your own banner	162
10.2.3.2	(Optional) Prepare the UI language package	163
10.2.3.3	(Optional) Customize the UI text	163
10.3	Packaging a stand-alone mobile project	165
10.3.1	Points to check before packaging	165
10.3.2	What will be packaged?	166
10.3.3	Packaging instructions	166
10.4	Customizing and packaging Apeon Workspace	182
11	Undeploying Apeon Applications	199
11.1	Undeploying instructions	199
11.1.1	Undeploying with the Deployment Profile mode	200
11.1.2	Undeploying with the PowerServer mode	203
12	Developing with Code Insight	207
12.1	Activating Code Insight	207
12.1.1	Configuring Code Insight	208
12.1.2	Enabling Code Insight	210
12.2	Coding with Code Insight	211
13	Launching AEM	213
13.1	Requirements	213
13.2	Launching AEM	213
14	Apeon PowerServer Help	215
15	Technical Support	216

Index	218
-------------	-----

1 About This Book

1.1 Audience

This book is written for PowerBuilder developers who want to use PowerServer Toolkit to complete the various Web or mobile deployment tasks, including analyzing, configuring, deploying, debugging, running, and packaging the application.

1.2 How to use this book

There are eighteen chapters in this book.

Chapter 1: About This Book

A general description of this book.

Chapter 2: Introduction

An overview of PowerServer Toolkit.

Chapter 3: Before you begin

Some important instructions that you **MUST** follow before using PowerServer Toolkit.

Chapter 4: Configuring PowerServer Toolkit

Instructions for configuring PowerServer Toolkit.

Chapter 5: Using UFA Tool

Instructions for analyzing the unsupported features in a PowerBuilder application.

Chapter 6: Deploying PowerBuilder Applications

Instructions for deploying PowerBuilder applications to the Web or the Mobile.

Chapter 7: Debugging Appeon Web Applications

Instructions for using Appeon Debugger to debug an Appeon Web application.

Chapter 8: Running Appeon Applications

Instructions for running Appeon applications after deployment.

Chapter 9: Using Information Manager

Instructions for viewing reports and log files generated in the Appeon migration process.

Chapter 10: Packaging Applications

Instructions for using Appeon application package tool to generate a deployment package, a native mobile project, or customization and package for Appeon Workspace.

Chapter 11: Undeploying Appeon Applications

Instructions on how to undeploy Appeon applications.

Chapter 12: Developing with Code Insight

Instructions on how to use the Appeon Code Insight feature.

Chapter 13: Launching AEM

Instructions for launching AEM from PowerServer Toolkit.

Chapter 14: Appeon PowerServer Help

Instructions for launching the Appeon HTML help file.

Chapter 15: Extended Toolkit

Instructions for using the extended tools provided by Appeon: the DLL/OCX Files Package tool, the Appeon Silent Installer, and the PowerServer Toolkit Register tool.

Chapter 16: Technical Support

Important information for technical support

1.3 Related documents

Appeon provides the following user documents to assist you in understanding Appeon PowerServer and its capabilities:

- **Introduction to Appeon:**
Gives general introduction to Appeon PowerServer and its editions.
- **Getting Started (for PowerServer Mobile):**
Guides you through installing PowerBuilder and Appeon PowerServer, and developing and deploying a mobile application.
- **New Features Guide:**
Introduces new features and changes in Appeon PowerServer.
- **PowerServer Mobile Tutorials:**
Gives instructions on deploying, running, and debugging the mobile application, distributing native mobile apps, and configuring the PowerServer cluster.
- **PowerServer Mobile (Offline) Tutorials:**
Gives instructions on setting up the PowerServer Mobile (Offline) environment, and configuring, deploying, running, updating, and debugging the offline application.
- **Appeon Installation Guide:**
Provides instructions on how to install Appeon PowerServer successfully.
- **Mobile UI Design & Development Guide:**
Introduces general guidelines on designing and developing the mobile app and UI.
- **Migration Guidelines for PowerServer Web:**
A process-oriented guide that illustrates the complete diagram of the Appeon Web migration procedure and various topics related to steps in the procedure, and includes a tutorial that walks you through the entire process of deploying a small PowerBuilder application to the Web.
- **Supported PB Features:**

Provides a detailed list of supported PowerBuilder features which can be converted to the Web/Mobile with Appeon as well as which features are unsupported.

- **Workarounds & API Guide:**

Provides resolutions for unsupported features and various APIs to facilitate you to implement the features (including Web and mobile) that are not easy or impossible to implement in the PowerBuilder IDE.

- **Appeon Workspace User Guide:**

Gives a general introduction on Appeon Workspace and provides detailed instructions on how to use it.

- **PowerServer Configuration Guide:**

Provides instructions on how to configure PowerServer Monitor, establish connections between PowerServer and database servers, and configure AEM for maintaining PowerServer and the deployed applications.

- **Web Server Configuration Guide:**

Describes configuration instructions for different types of Web servers to work with the PowerServer.

- **Troubleshooting Guide:**

Provides information on troubleshooting issues; covering topics, such as product installation, application deployment, AEM, and Appeon application runtime issues.

- **Appeon Performance Tuning Guide:**

Provides instructions on how to modify a PowerBuilder application to achieve better performance from its corresponding Web/mobile application.

- **Testing Appeon Web Applications with UFT:**

Provides instructions on how to test Appeon Web applications with QTP.

1.4 If you need help

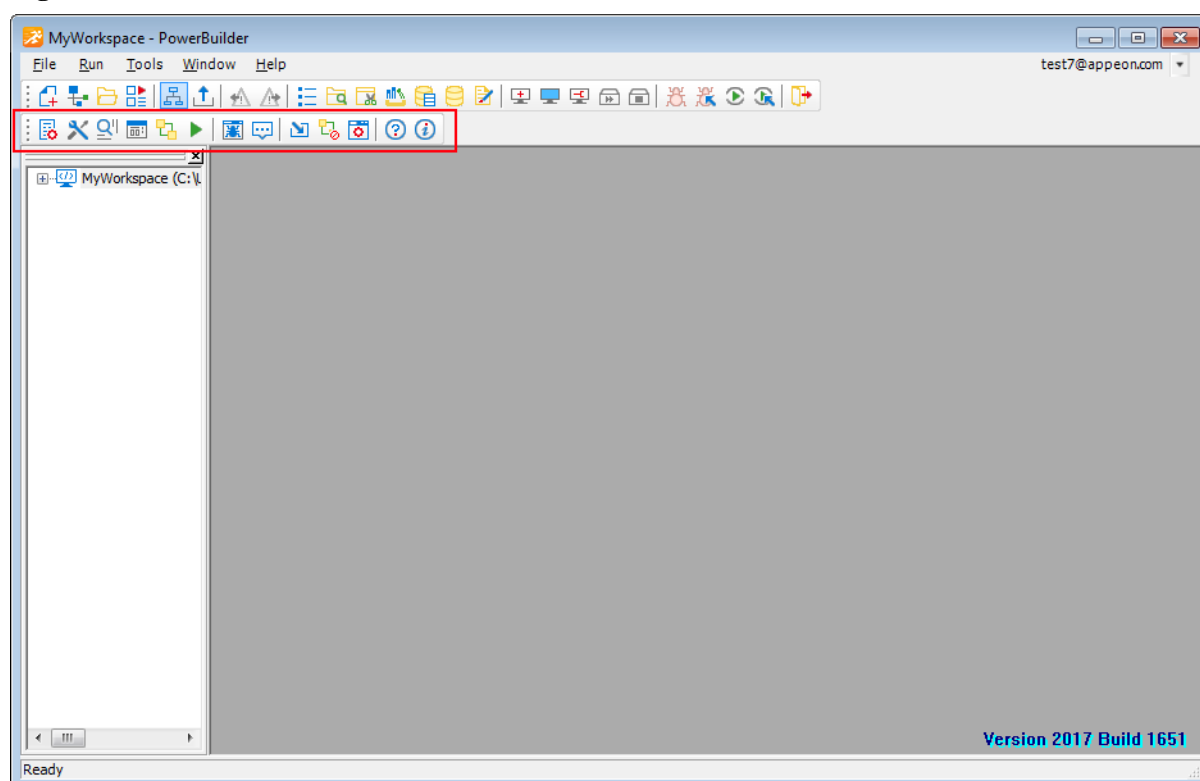
If you have any questions about this product or need assistance during the installation process, access the Technical Support Web site at <http://www.appeon.com/support>.

2 Introduction

PowerServer Toolkit, a component of Appeon PowerServer, extends the capabilities of PowerBuilder, allowing a new or an existing PowerBuilder application to be converted into a *bona fide* Web application or a native mobile application using only PowerBuilder skills.

PowerServer Toolkit provides a set of tools that enable the entire PowerBuilder-to-Web or PowerBuilder-to-Mobile process to take place within the PowerBuilder IDE. These tools are accessed via a toolbar in the PowerBuilder IDE, which automatically loads each time PowerBuilder is opened.

Figure 2.1: PowerServer Toolkit embedded in the PowerBuilder IDE



PowerServer Toolkit converts a PowerBuilder application by parsing the PBLs (source code) of PowerBuilder applications and generating a set of corresponding HTML, JavaScript, and XML files. When deployed by PowerServer, these generated files form an exact Web-based or mobile-OS-based replica of the source PowerBuilder application. Any user can open a standard Windows Web browser and access the Web-based version of the original PowerBuilder application over the Internet, an intranet, or an extranet; or access the native mobile application deployed by Appeon using a mobile device, such as an iPhone, an iPad, Android-powered tablets and smartphones, etc.

2.1 PowerServer Toolkit in PowerBuilder

Installing PowerServer Toolkit

Starting from version 2017, PowerServer Toolkit is only available in the PowerServer (PB Edition) which is installed along with the PowerBuilder Universal Edition only. PowerServer Toolkit 2017 can deploy apps to PowerServer Web/Mobile 2017 or

2016, while PowerServer Web/Mobile 2017 can only host apps deployed from PowerServer Toolkit 2017 (not 2016 or earlier toolkit).

The PowerServer Toolkit conforms to the US Government Section 508 Accessibility Guidelines.

The following figure represents the user interface of the PowerServer Toolkit.

Figure 2.2: PowerServer Toolkit



The PowerServer Toolkit has 13 buttons. The following table describes the functionality of each button.

Table 2.1: PowerServer Toolkit functions

Button	Name	Function
	Config Wizard	Provides a wizard for quick configuration of PowerServer Toolkit for Web or mobile conversion.
	Configure	Sets configurations of PowerServer Toolkit for Web or mobile conversion.
	Analyze	Analyzes application source code for unsupported features before deployment. This analysis can cover the whole application, at object level or within an inheritance hierarchy.
	Code Insight	Develops PowerBuilder applications that are free of Appeon unsupported features.
	Deploy	Starts the complete process of deploying a PowerBuilder application to the Web or Mobile.
	Run	Runs the deployed Web application (in Web browsers) or the deployed mobile application (in Appeon Workspace).
	Appeon Debugger	Debugs Appeon applications.
	Information	Manages logs and reports.
	Package	Packages Appeon application files for installation.
	Undeploy	Undeploys Web or mobile applications from Web Server(s) and PowerServer(s).
	AEM	Launches AEM.
	Help	Searches, browses, prints, copies, and pastes useful information from the Appeon user guide
	Get Support	Displays the product version and the support information. The Check Update button in it allows you to check the Appeon updates.

Apart from Code Insight and Information, only one PowerServer Toolkit function can be launched at any given time. This prevents a PowerServer Toolkit process from interfering with another.

2.2 Web and/or mobile application development with PowerServer Toolkit

For either developing a new Web or native mobile application, or migrating an existing PowerBuilder application onto the Web or to the mobile, you need to perform three key tasks in PowerServer Toolkit: Analyze, Modify and Deploy Automatically.

If you use Appeon PowerServer for developing a new Web or native mobile application, the first step is to write a new PowerBuilder application (refer to Chapter 2, *Web RAD with PowerBuilder and Appeon in Migration Guidelines for PowerServer Web*). It is recommended that the new PowerBuilder application conforms to Appeon coding styles, as laid out in Supported PB Features for PowerServer Mobile, or in Supported PB Features for PowerServer Web then you will expend less effort in the first two tasks: Analyze and Modify.

- Task 1: Analyze

You need to analyze the application PBLs for unsupported PowerBuilder objects and features, using the PowerBuilder IDE extended with PowerServer Toolkit. General rules of thumb regarding how the application should be structured are outlined in Chapter 2, *Basic Requirements and Recommendations in Supported PB Features for PowerServer Mobile* or in Supported PB Features for PowerServer Web. The application must meet these requirements before the analysis can begin.

The first step results in an analysis report highlighting unsupported PowerBuilder objects and code within the application.

- Task 2: Modify

Next, work around or remove the unsupported PowerBuilder objects and code that prevent the Web or mobile application from running, using standard PowerBuilder programming. Supported PB Features for PowerServer Mobile, or Supported PB Features for PowerServer Web and *Appeon Code Examples* (an Appeon demo PowerBuilder application) will guide you through this process by providing information regarding supported PowerBuilder features as well as example code for implementing these features.

Then, perform a **full build** of the application in PowerBuilder, ensure that there are no bugs in the PowerBuilder code and that the application functions correctly.

The second task results in a PowerBuilder application that is ready for automatic conversion to the Web or to the Mobile.

- Task 3: Deploy automatically

At the push of a button, you can now automatically generate a precise replica of the PowerBuilder application, that deploys to the n-Tier architecture and can be accessed by Web browsers (Appeon Web applications) or in Appeon Workspace on various mobile devices (Appeon mobile applications).

Then, automatically deploy the generated Web or mobile application files to PowerServer. The third task results in a *bona fide* Web application or a native mobile application with the look and feel of the source PowerBuilder application.

For detailed information about the Web conversion, please refer to the Migration Guidelines for PowerServer Web.

3 Before You Begin

3.1 Installing Appeon PowerServer

Follow Installation Guide for .NET carefully; make sure that both PowerServer Toolkit and PowerServer have been installed.

Verify that PowerServer Toolkit has been installed to the developer PC.

Verify that PowerServer has been installed to the application server, such as, SAP NetWeaver Application Server, Oracle WebLogic, IBM WebSphere, TmaxSoft JEUS, JBoss, or Microsoft .NET Framework\IIS. Once PowerServer has been installed, the machine hosting PowerServer is then referred to as PowerServer.

If using a separate Web server (e.g. Apache) instead of the application server built-in Web server, verify that the PowerServer Web Component has been installed to the Web Server.

Verify that all system requirements have been met for the developer PC, Web server, PowerServer, and the database server. Refer to Chapter 3, *Installation Requirements in Installation Guide for .NET* for details.

3.2 Configuring application database connectivity

Perform the following database configurations for all prospective PowerBuilder applications that will be deployed with PowerServer Toolkit:

1. Set up the database used by the PowerBuilder application to interface with PowerServer and the JDBC driver. Refer to the documentation from appropriate database vendors for instructions.
2. Create the data source in PowerServer. Appeon provides systematic instructions for certified database systems. Refer to Section 4.4, “Setting up PowerServer data sources” in *PowerServer Configuration Guide for .NET* or in PowerServer Configuration Guide for J2EE for more information.
3. Map the transaction object in the target PowerBuilder application to the newly created data source for the prospective Web or mobile application. You can either dynamically set up the mapping via PowerScript following the instructions in Section 4.5.1, “Dynamic transaction object to data source mapping” in *PowerServer Configuration Guide for .NET* or in PowerServer Configuration Guide for J2EE or establish the mapping statically in PowerServer Toolkit or AEM.

3.3 Starting PowerServer

PowerServer Toolkit interacts with PowerServer and Web server during Web or mobile deployment. Web or mobile applications are deployed to one or more PowerServer and one or more Web servers. You should verify that the PowerServer and Web server are running before deployment. If you use the application server built-in Web server as the Web server, you only need to start the application server/PowerServer. For example, if you use the same .NET IIS server as the PowerServer and the Web server, you only need to start the .NET IIS/PowerServer. Otherwise, start both the Web server and PowerServer. Refer to

the documents provided by the Web server vendor for how to start the Web server, and refer to the following instructions to start the PowerServer.

Table 3.1: How to start PowerServer

PowerServer	Operating System	Detailed Instructions
.NET	Windows	Select Start > All Programs > Appeon PowerServer 2017 > PowerServer for .NET > IIS Manager
WebLogic	Windows	Select Windows Start > All Programs > Appeon PowerServer 2017 > PowerServer for WebLogic > Instances > InstanceName > Start WebLogic
	UNIX/ Linux	Run the appeonservice.sh file in the \$appeon/bin/ folder, for example: \$ \$BEA_HOME/user_projects/domains/mydomain/appeon/bin/appeonservice.sh
WebSphere	Windows	Select Windows Start > All Programs > Appeon PowerServer 2017 > PowerServer for WebSphere > Start WebSphere
	UNIX/ Linux	Run the appeonservice.sh file in the \$appeon/bin/ folder, for example: \$ \$WebSphere/AppServer/appeon/bin/appeonservice.sh
JBoss	Windows	Select Start > All Programs > Appeon PowerServer 2017 > PowerServer for JBoss > Instances > InstanceName > Start JBoss
	UNIX/ Linux	Run the appeonservice.sh file in the \$appeon/bin/ folder, for example: \$ \$wildfly-10.0.0.Final/appeon/bin/appeonservice.sh or \$ \$jboss-eap-6.4/appeon/bin/appeonservice.sh
JEUS	Windows	Select Start > All Programs > Appeon PowerServer 2017 > PowerServer for JEUS > Start JEUS
	UNIX/ Linux	Run the appeonservice.sh file in the \$appeon/bin/ folder, for example: \$ \$tmaxsoft/JEUS6.0/appeon/bin/appeonservice.sh
EAServer	Windows	Select Windows Start > All Programs > Appeon PowerServer 2017 > PowerServer for EAServer > Start EAServer
	UNIX/ Linux	Run the appeonservicestart.sh file in the \$appeon/bin/ folder, for example: \$ \$JAGUAR/appeon/bin/appeonservicestart.sh You can run EAServer/PowerServer in different modes, debug and normal, using different Java runtime versions and different Java VMs. For details on how to specify the mode options in the above syntax, refer to the <i>EAServer System Administration Guide</i> .

4 Configuring PowerServer Toolkit



The settings configured in PowerServer Toolkit are critical; they are used throughout the entire PowerBuilder-to-Web or PowerBuilder-to-Mobile process. PowerServer Toolkit settings determine which PowerBuilder application will be converted to the Web or to the Mobile, and the manner in which it will be deployed.

Before you attempt to use any other functionality on the toolkit, complete the following tasks:

1. Set up an application profile for each of the PowerBuilder applications intended for conversion. Each application profile tells PowerServer Toolkit important information about the application, such as which PBLs compose the PowerBuilder application, the database type, etc.


During the application profile setup, set up a profile for the database type used by the application. This enables PowerServer Toolkit to generate the correct database syntax.

2. Set up at least one PowerServer profile and one Web Server profile. This enables PowerServer Toolkit to utilize the PowerServer and Web Server for deployment.
3. Set up at least one deployment profile, which links at least one PowerServer and one Web Server together. This tells PowerServer Toolkit where to deploy the Web or the mobile application.

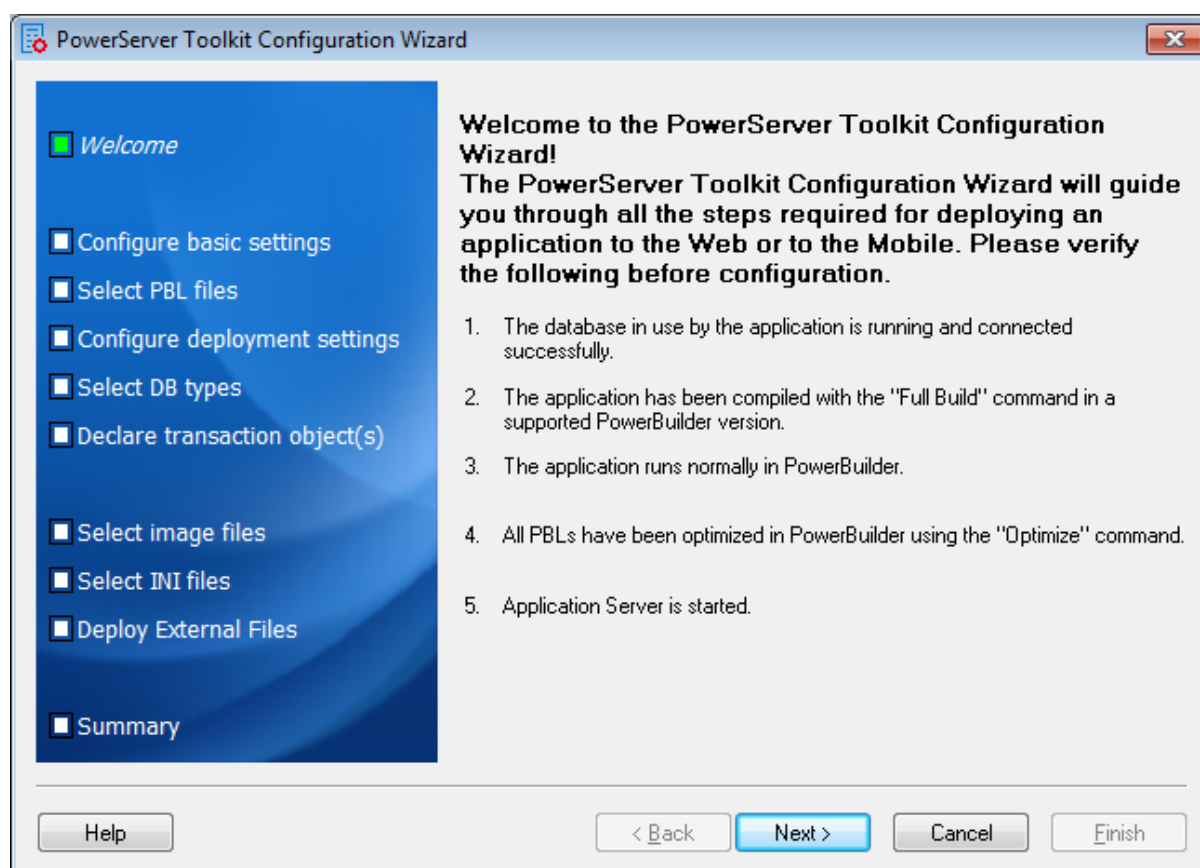
Apeon provides a Config Wizard () to quickly configure an application. With your specifications, the wizard creates the application profile, database type profile, PowerServer profile, Web server profile, data source profile, and transaction object mapping. After creation, all these profiles can be modified in the Configure tool (). Besides modifying the above profile settings, the Configure tool provides you with advanced settings, such as performance settings, parsing options, Web Services profiles, runtime settings, etc.

4.1 Using PowerServer Toolkit Configuration Wizard

The settings in PowerServer Toolkit Configuration Wizard are the same as those in the Configure tool, therefore, only basic descriptions will be provided in this section. For detailed descriptions, refer to [Section 4.2, “Using Configure Tool”](#).

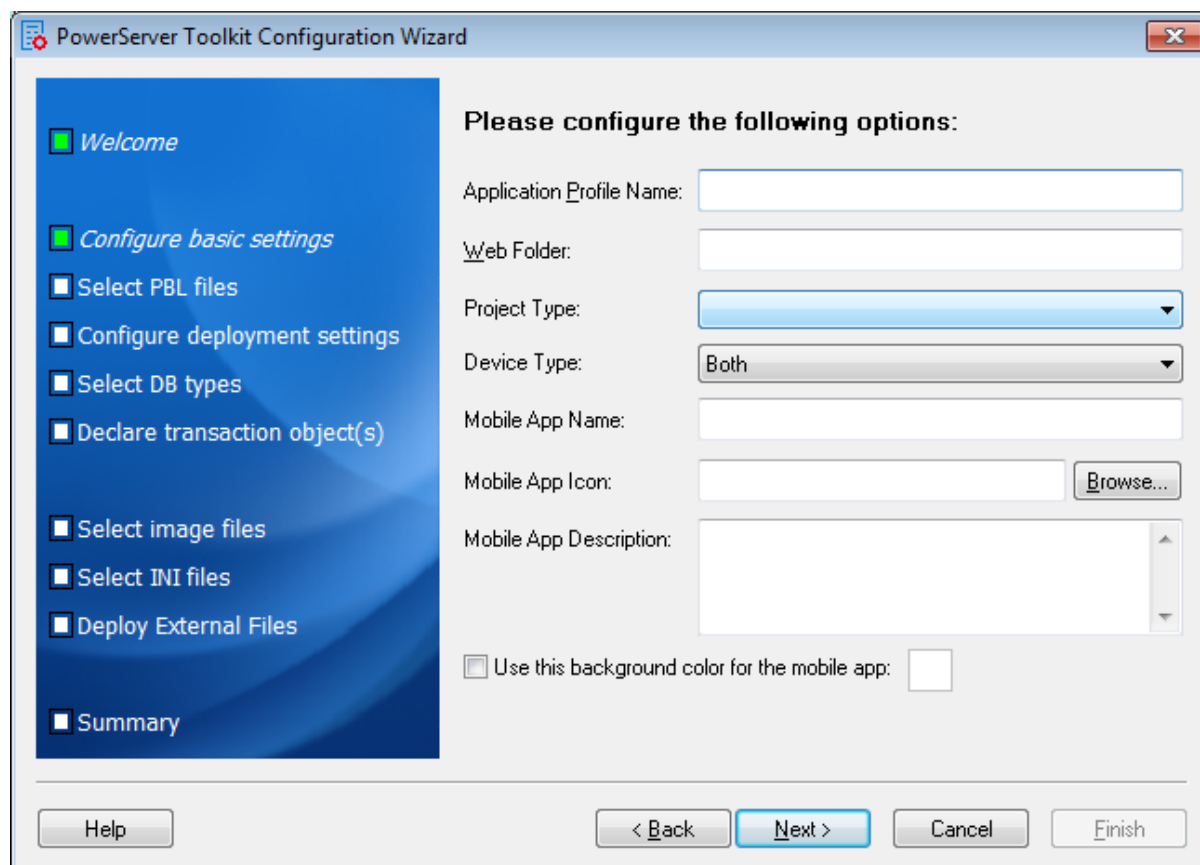
Click the **Config Wizard** button () on the PowerServer Toolkit to launch the PowerServer Toolkit Configuration Wizard.

Carefully read the notes and requirements on the Welcome page of the PowerServer Toolkit Configuration wizard. Click **Next** to proceed.

Figure 4.1: Welcome page

4.1.1 Configuring basic settings

The basic settings refer to the settings of an application that are essential for deployment, including application profile name, web folder, project type, device type, mobile app name, mobile app icon, and mobile app description. For more information about them, refer to [Section 4.2.1.2.1, "Basic Settings"](#) in [Section 4.2, "Using Configure Tool"](#).

Figure 4.2: Configure basic settings of an application

Step 1: Specify the application profile name in the **Application Profile Name** text box.

The application profile name is used to identify the PowerBuilder application during the entire PowerBuilder-to-Web or PowerBuilder-to-Mobile process.

Step 2: Specify the application URL in the **Web Folder** text box.

The specified text is also used as the name of the folder created under the Web root of the Web server for storing the files when the application is deployed.

Step 3: Select a project type from the **Project Type** dropdown list box.

Select "Mobile" if you intend to deploy your PowerBuilder application to mobile devices; select "Web" if you intend to deploy your PowerBuilder application to the Web; select "Both" if you intend to deploy your PowerBuilder application to both the Web and Mobile.

Step 4: (For mobile apps) Select a device type from the **Device Type** dropdown list box.

Tablet, Smartphone, and Both are listed for choices. 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".

Step 5: (For mobile apps) Enter an app name in the **Mobile App Name** text box.

The mobile app name specified will be displayed in the Appoon Workspace home screen after the application is installed to Appoon Workspace.

Step 6: (For mobile apps) Specify an icon for your mobile applications by clicking the **Browse...** button.

The icon specified will be displayed on the Apeon Workspace home screen after your application is installed to Apeon Workspace.

Step 7: (For mobile apps) Enter a brief description in the **Mobile App Description** text box.

The description will be displayed on the home screen of Apeon Workspace as the application's description after deployment.

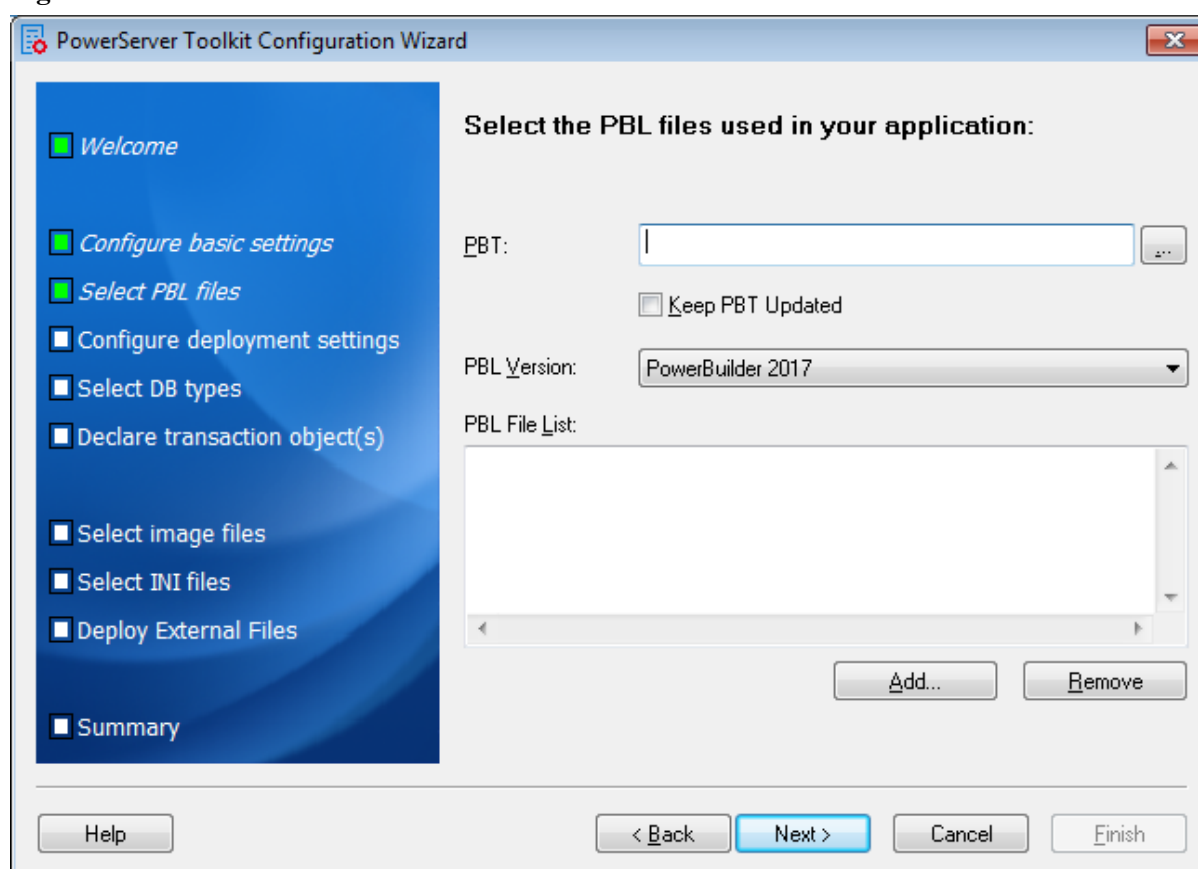
Step 8: (For mobile apps) Specify the background color of the mobile application. If this option is not set, the default image will be used as the background for the mobile application.

Step 9: Click **Next**.

4.1.2 Selecting PBL file(s)

Specify the version and the location of the PowerBuilder application source code.

Figure 4.3: Select PBLs



You can add application PBLs using one of the following methods:

- Method 1: Select the PBT file to automatically add the PBL files. Details are as follows:
Click the browse (...) button to select the PBT file.

When a PBT file is selected, all PBL files contained in the selected PBT are added to the PBL File List.

This is the recommended method to add PBLs, as it reduces the likelihood of forgetting to add a required PBL.

- Method 2: You can also add PBL files without specifying the PBT file first.

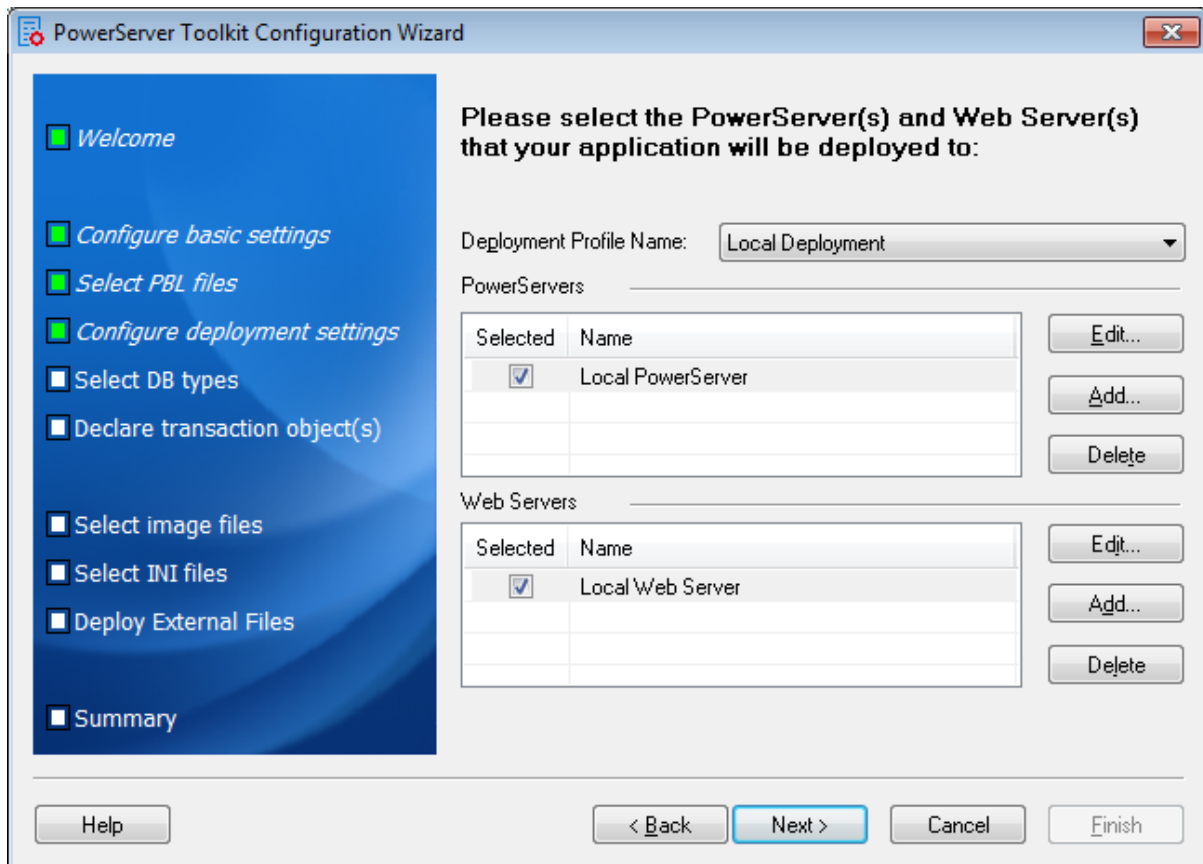
Click **Add...** to add one or multiple PBL files. Refer to [Section 4.2.1.2.1, “Basic Settings”](#) for detailed instructions.

Note: Keep the **Keep PBT Updated** checkbox checked if you want Apeon to automatically update the PBLs every time you deploy an application. This ensures that any newly added PBLs will be deployed and yet avoid missing PBLs after deployment.

4.1.3 Configuring deployment settings

The deployment settings associate the PowerServer(s) and Web server(s) as a group used for the application deployment.

Figure 4.4: Select the PowerServer and Web server



Step 1: Select an existing deployment profile.

To create or modify a deployment profile, you must use the Configure tool. For detailed instructions, refer to [Section 4.2.4.2, “Deployment profile settings”](#).

Step 2: Select the check boxes to include the PowerServer(s) and the Web server(s) in the deployment profile.

All the PowerServer profiles and Web server profiles you have created are listed. More than one PowerServer and one Web server can be selected.

If the required PowerServer profiles or Web server profiles are not available, click the **Add** button to create them. Refer to [Section 4.2.3.2, “PowerServer profile settings”](#) for detailed instructions on creating a PowerServer profile and refer to [Section 4.2.3.3, “Web Server profile settings”](#) for detailed instructions on creating a Web server profile.

Step 3: Click **Next** to proceed.

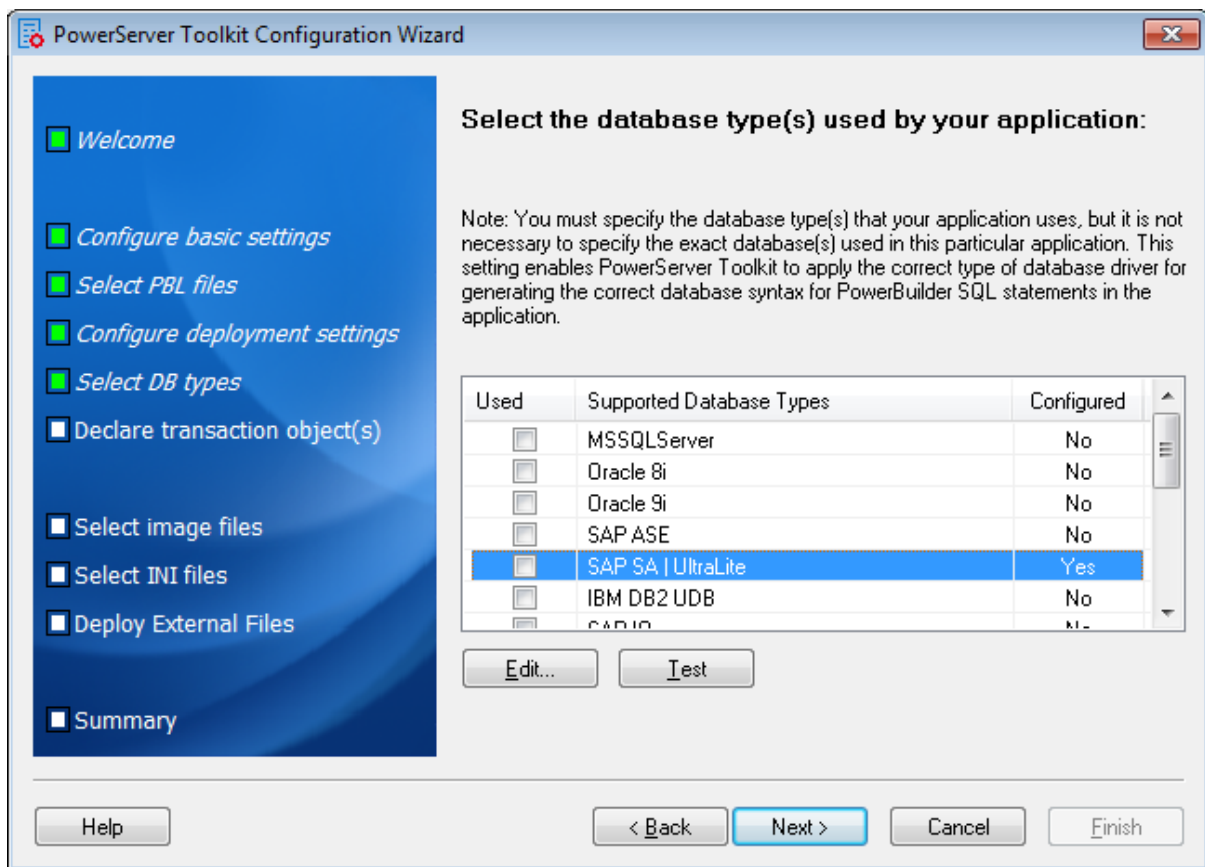
When the **Next** button is clicked, the wizard will test the connection to the PowerServer and the Web server. If the connection is successful, the deployment profile is saved with the associated PowerServer(s) and Web server(s).

4.1.4 Selecting DB Type(s)

The database type is required during the application deployment if the application connects to one or more databases.

Select the database type(s) used by the application by selecting the check boxes from the **Used** column of the database type.

Figure 4.5: Database types



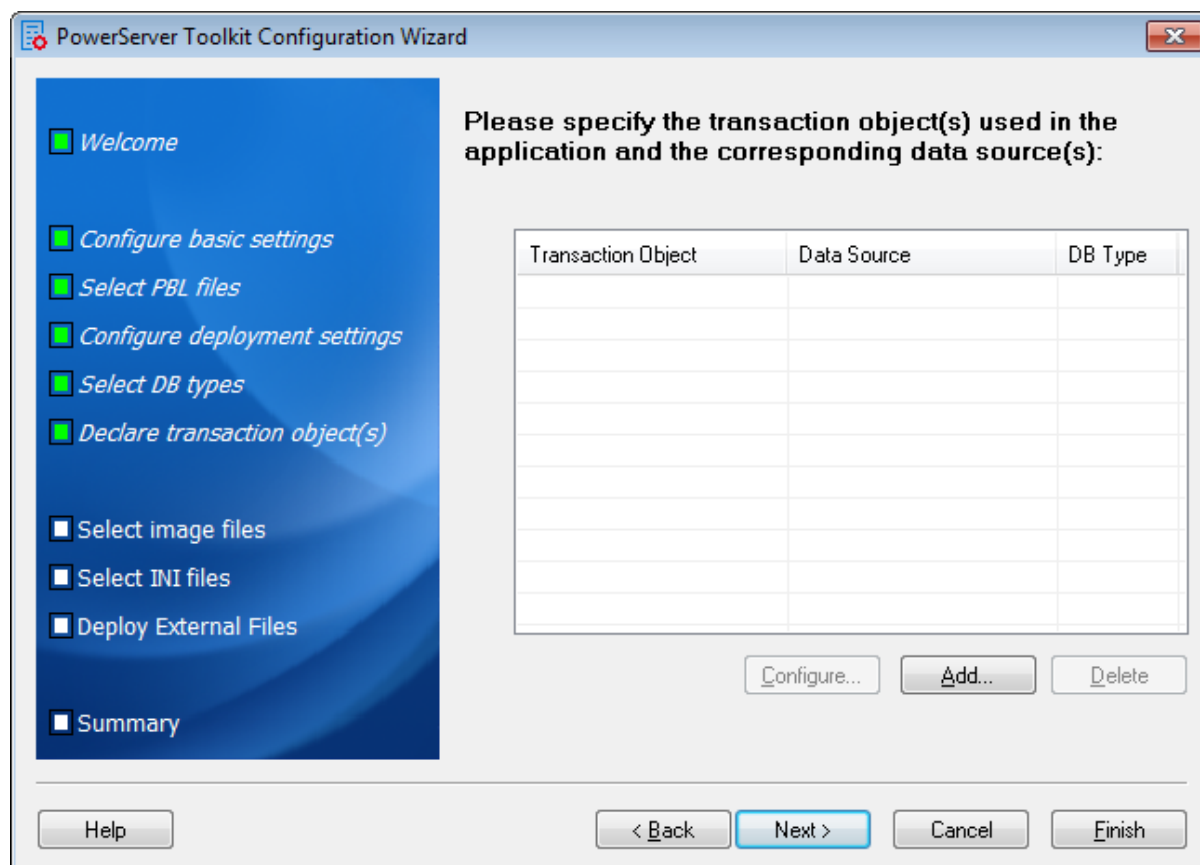
If the database type is not configured (The **Configured** column is indicated with "No"), you should select it and click the **Edit** button to create a profile for it. For detailed instructions on creating a database type profile, refer to [Section 4.2.2, "Managing database type profiles"](#).

4.1.5 Declaring transaction object(s)

When the application is deployed to the Web or to the Mobile, PowerServer handles the connection with the server consolidated database (not the local offline database) using data sources rather than transaction objects defined in the PowerBuilder application. You must associate the transaction objects used by the PowerBuilder application with proper data sources created in PowerServer.

You can create, modify or delete the mapping between transaction objects and data sources either in PowerServer Toolkit or AEM. If you create, modify or delete the mapping in PowerServer Toolkit, make sure you deploy the application to synchronize the change to AEM; or you can directly create, modify or delete the mapping in AEM without needing to re-deploy the application.

Figure 4.6: Specify transaction objects



Click **Add** to create the mapping of the transaction object and the data source of an application.

Figure 4.7: Add a transaction object

Add Transaction Object

Transaction Object:

Database Type:

Data Source:

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:

Selected	Name	DB Host

Step 1: Input the transaction object name used by the application to the **Transaction Object** text box.

Step 2: Select the database type from the **Database Type** dropdown list box.

Step 3: Specify the data source name in the **Data Source** text box. The data source should connect to the same database that the transaction object connects to.

Make sure that the data source exists in all PowerServer selected for this application deployment. If the data source does not exist, you can create it by following instructions in [Section 4.2.5, “Managing data source profiles”](#). After you create a data source, you can select the **Selected** column to associate the data source with the transaction object.

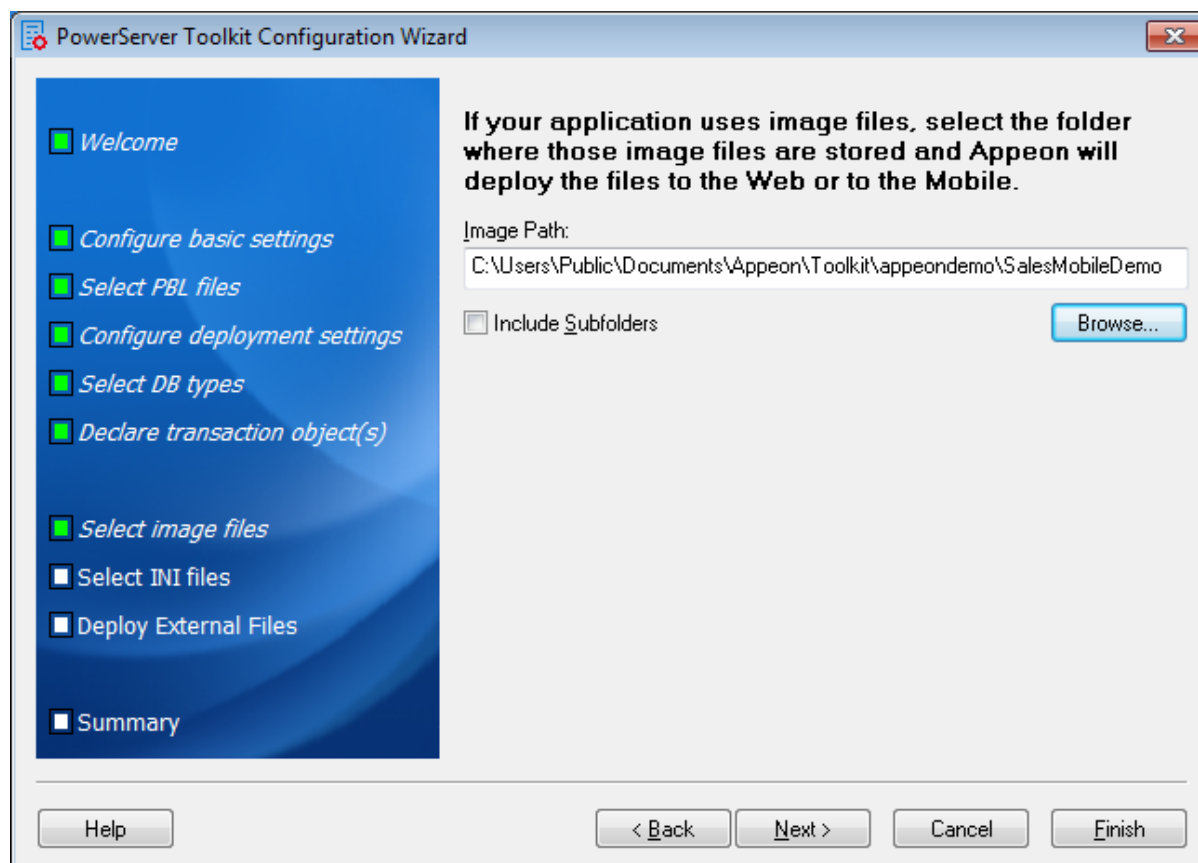
Note: The data source tool will be available if PowerServer is installed to Microsoft .NET Framework\IIS, and will not be available if PowerServer is installed to WebLogic, WebSphere, JBoss, NetWeaver Application Server, or JEUS. You should go to the corresponding application server administration console to configure the data source for WebLogic, WebSphere, JBoss, NetWeaver Application Server, or JEUS. For detailed instructions, refer to Chapter 4, *Database Connection Setup* in *PowerServer Configuration Guide for J2EE*.

4.1.6 Selecting image files

Click **Browse** to specify which folder contains the image files that will be used in the Web or mobile application. Select **Include Subfolders** to deploy the sub-folders under the specified directory.

For detailed information about the image files, refer to [Deploy Images](#).

Figure 4.8: Select image files

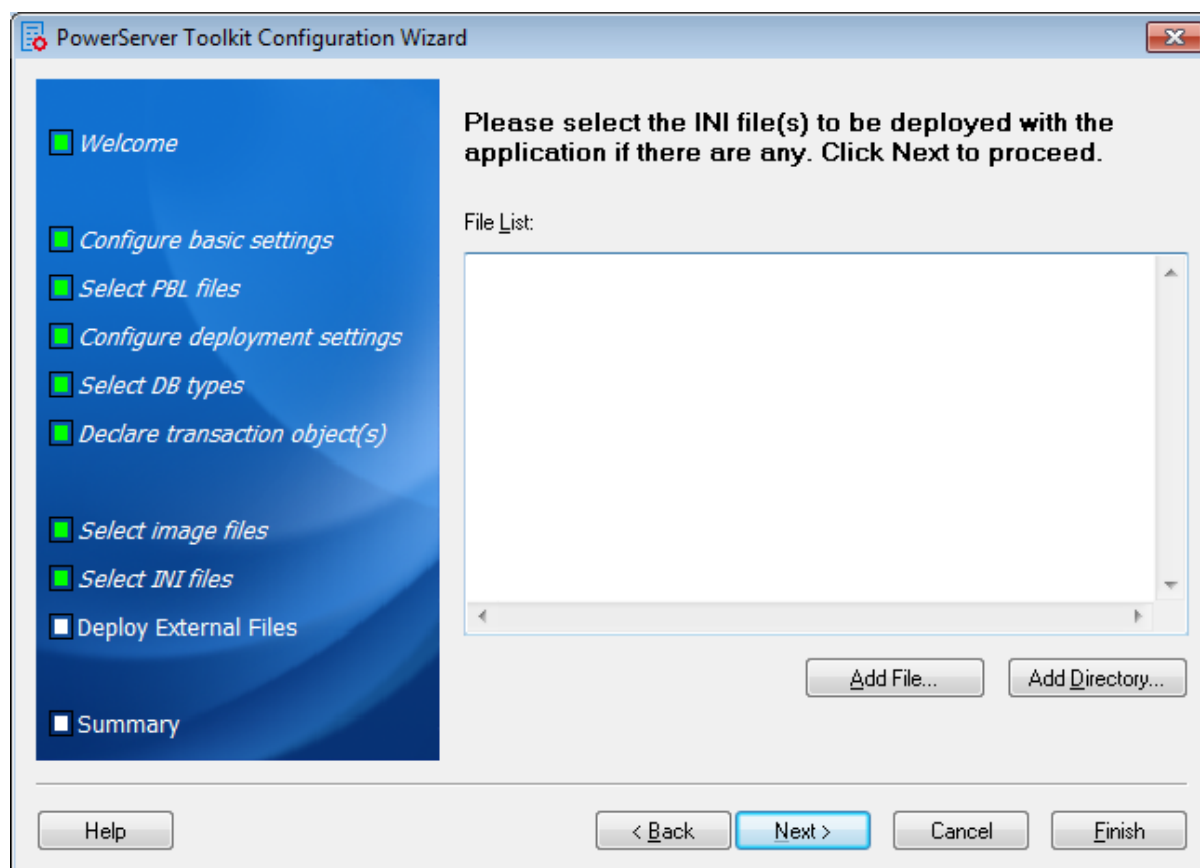


4.1.7 Selecting INI files

If a PowerBuilder application uses an initialization (INI) file which contains user preferences, specify the INI file so that Appeon can deploy the file for the application.

Click **Add File...** to add one or multiple INI files or click **Add Directory** to add all INI files in the selected directory.

For detailed information about the INI files, refer to [Deploy INI Files](#).

Figure 4.9: Select INI files

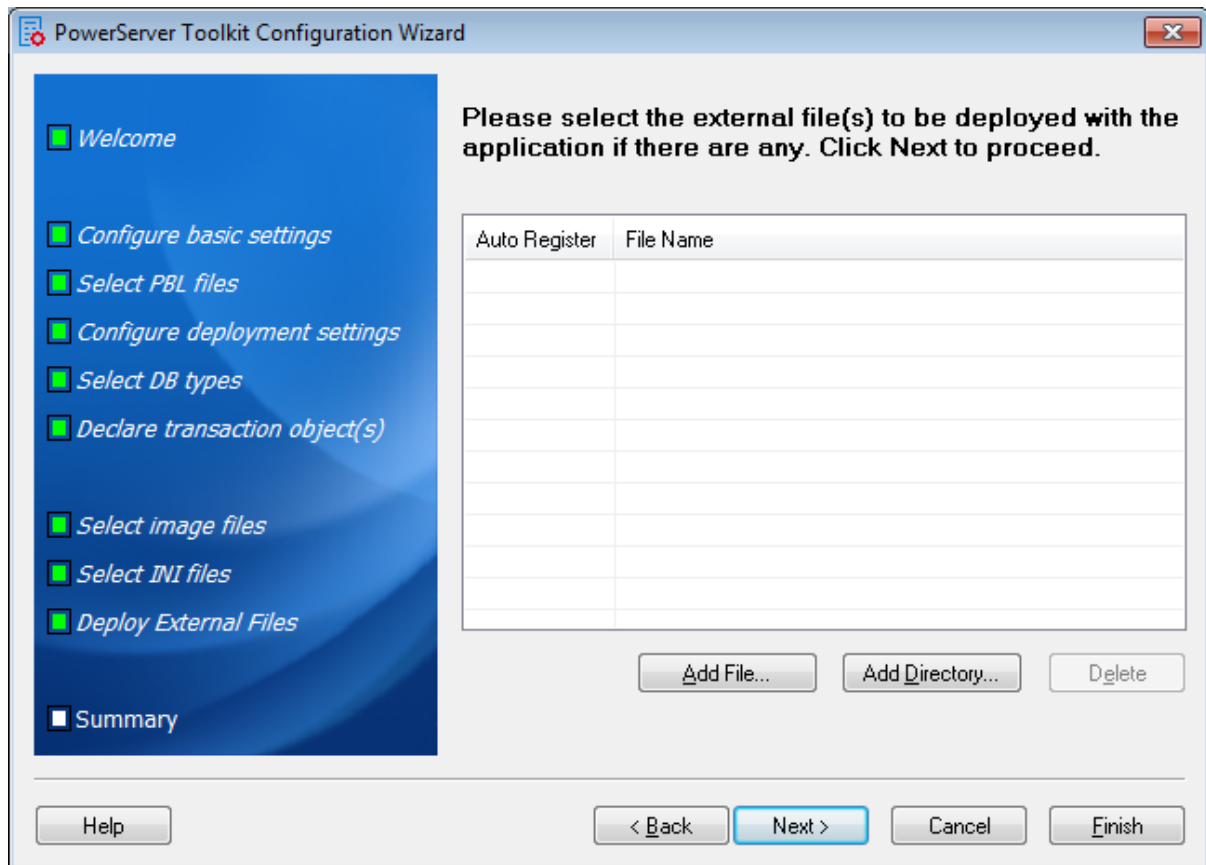
4.1.8 Selecting External files

If your application calls any custom user external files, such as DLL/OCX files, text files, etc., you can specify them here and deploy them to the Web or the Mobile.

Click **Add File** or **Add Directory** to add external files to the table. Then select the **Auto Register** column if the file needs to be automatically registered after it is downloaded to the client. If you choose not to automatically register a file, you can register it manually after it is downloaded.

For more information about the external files, refer to [Deploy External Files](#).

Figure 4.10: Select External files

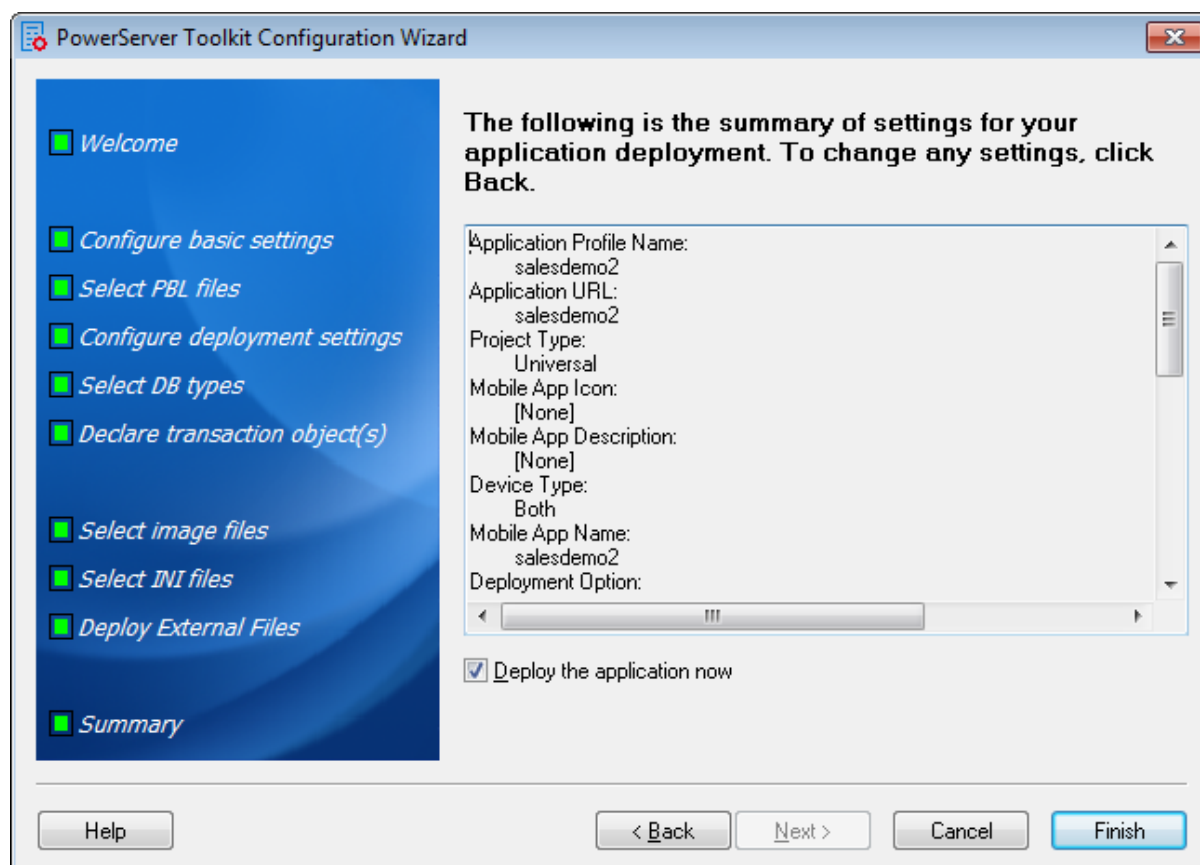


4.1.9 Summary

Review the settings. Click **Finish** to complete the configuration.

If the **Deploy the application now** option is selected, the Apeon Deployment Wizard will be launched for you to deploy the application. For detailed information, refer to [Chapter 6, Deploying PowerBuilder Applications](#).

After the configuration is complete, the application will be added to the Application Profiles tab of the Configure tool where you can manage or change any settings of the application. For detailed instructions, refer to [Section 4.2, “Using Configure Tool”](#).

Figure 4.11: Summary page

4.2 Using Configure Tool

The Configure tool allows you to modify the settings of existing application profiles, database type profiles, PowerServer profiles, Web server profiles, deployment profiles and data source profiles, and specify the advanced settings for an application profile, such as the performance settings, parsing options, Web service profiles, and runtime settings.

4.2.1 Managing application profiles

The Configure tool provides an Application Profiles tab for you to modify the settings specified in the PowerServer Toolkit Configuration Wizard, and use the following advanced functions:

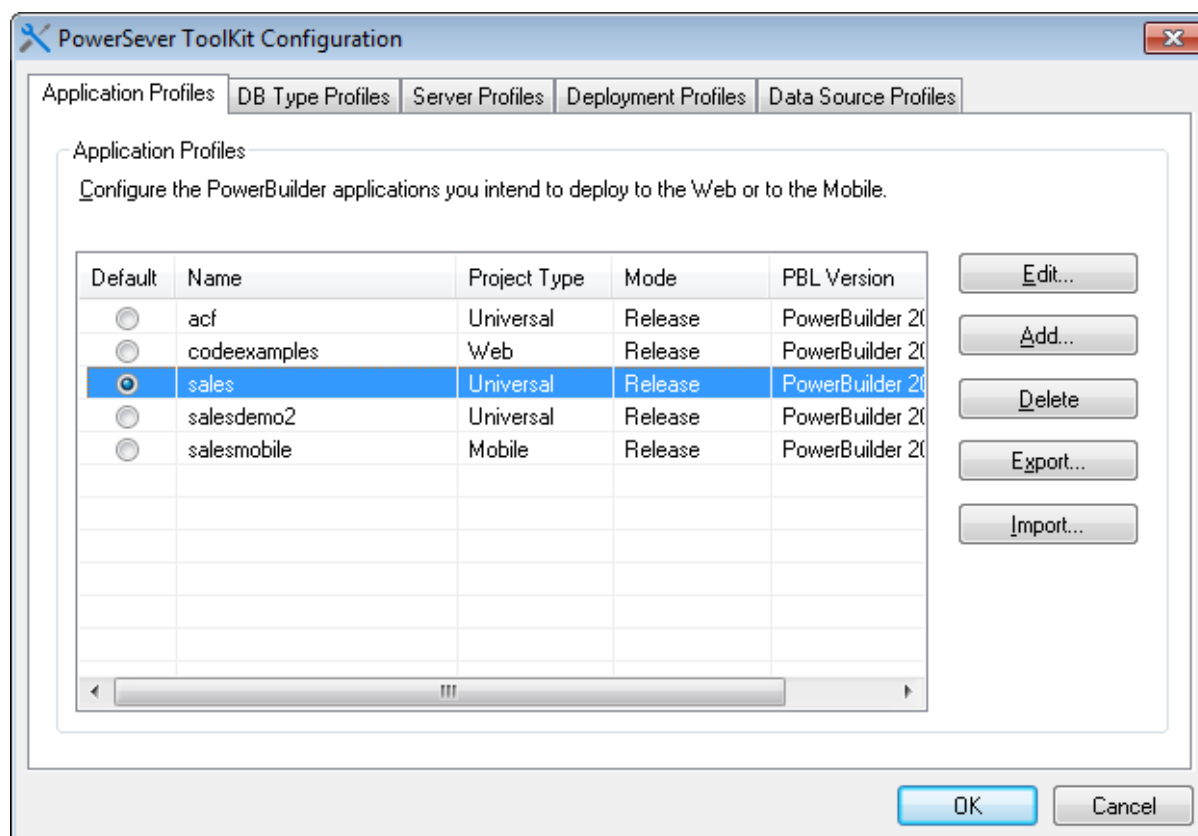
- Specify the default application profile
- Select the file generation mode for an application
- Delete an application profile
- Boost the application runtime performance
- Specify parsing options
- Specify runtime settings

- Configure Web Service profiles

4.2.1.1 Application Profiles tab page

When you click the **Configure** button (✂) on the PowerServer Toolkit, the PowerServer Toolkit Configuration window appears with the Application Profiles tab selected by default, as shown in the following figure.

Figure 4.12: PowerServer Toolkit Configuration Window



4.2.1.1.1 Specifying the default application profile

To specify which application profile will be used as the default application profile, select the **Default** radio button.

The PowerBuilder application defined in the default application profile is selected for unsupported features analysis, PowerBuilder-to-Web or PowerBuilder-to-mobile conversion, application packaging, and application undeployment.

4.2.1.1.2 Deleting an application profile

Click the **Delete** button to delete a selected application profile.

Note: You cannot delete the default application profile.

Deleting the application profile will automatically delete the temporary fold for the application profile on the PowerServer Toolkit machine. The folder has the same name as the application profile and is created in the \Project folder under the root directory where PowerServer Toolkit is installed (e.g. C:\Program Files\Appeon\PowerServer\Toolkit\).

4.2.1.1.3 Exporting/Importing an application profile

The Export and Import buttons are mainly to back up and restore the configuration of application profile(s), for instance, if you want to transfer the configuration of application profile(s) to another PowerServer Toolkit.

- Click the **Export** button to export the selected application profile(s). In the pop-up dialog box, type a file name, then the selected application profile(s) will be saved into the file as XML file format.

To select multiple application profiles, you can use Shift+click or Ctrl+click, or drag the mouse pointer to create a selection.

- Click the **Import** button to import application profile(s) from the XML file which is generated by using the Export button.

If the application profile being imported has the same name as an existing application profile, you will be asked to overwrite or ignore the existing application profile.

4.2.1.2 Application profile settings

The Application Profiles Configuration window provides eight tab pages for you to edit the required application information:

Table 4.1: Application Profiles settings

Tab	Settings	To make changes effective, you must...
Basic Settings	The application profile name, project type, application URL, PBL version and PBL location of an application. See Section 4.2.1.2.1, “Basic Settings” .	Perform a full or incremental deployment on the application.
DB Settings	The database types, transaction objects and the corresponding data sources. See Section 4.2.1.2.2, “DB Settings” .	Perform a full or incremental deployment on the application.
Additional Files	INI files, .NET/COM components, image files and External files. See Section 4.2.1.2.3, “Additional Files” .	Perform a full or incremental deployment on the application.
Misc Settings	Command line arguments, application language, runtime performance, parsing option, and log-writing mode. See Section 4.2.1.2.4, “Misc Settings” .	Perform a full deployment on the application.
Web Service Profiles	The WSDL file, service name and port for the Web service(s). See Section 4.2.1.2.5, “Web Service Profiles” .	Perform a full or incremental deployment on the application.
Offline Settings	The offline capacity, start mode, and the local database files. See Section 4.2.1.2.6, “Offline Settings” .	Perform a full or incremental deployment on the application.
Runtime Settings	Company name and application name displayed on the Web application downloading page, and mobile application name, icon,	Perform a full or incremental deployment on the application.

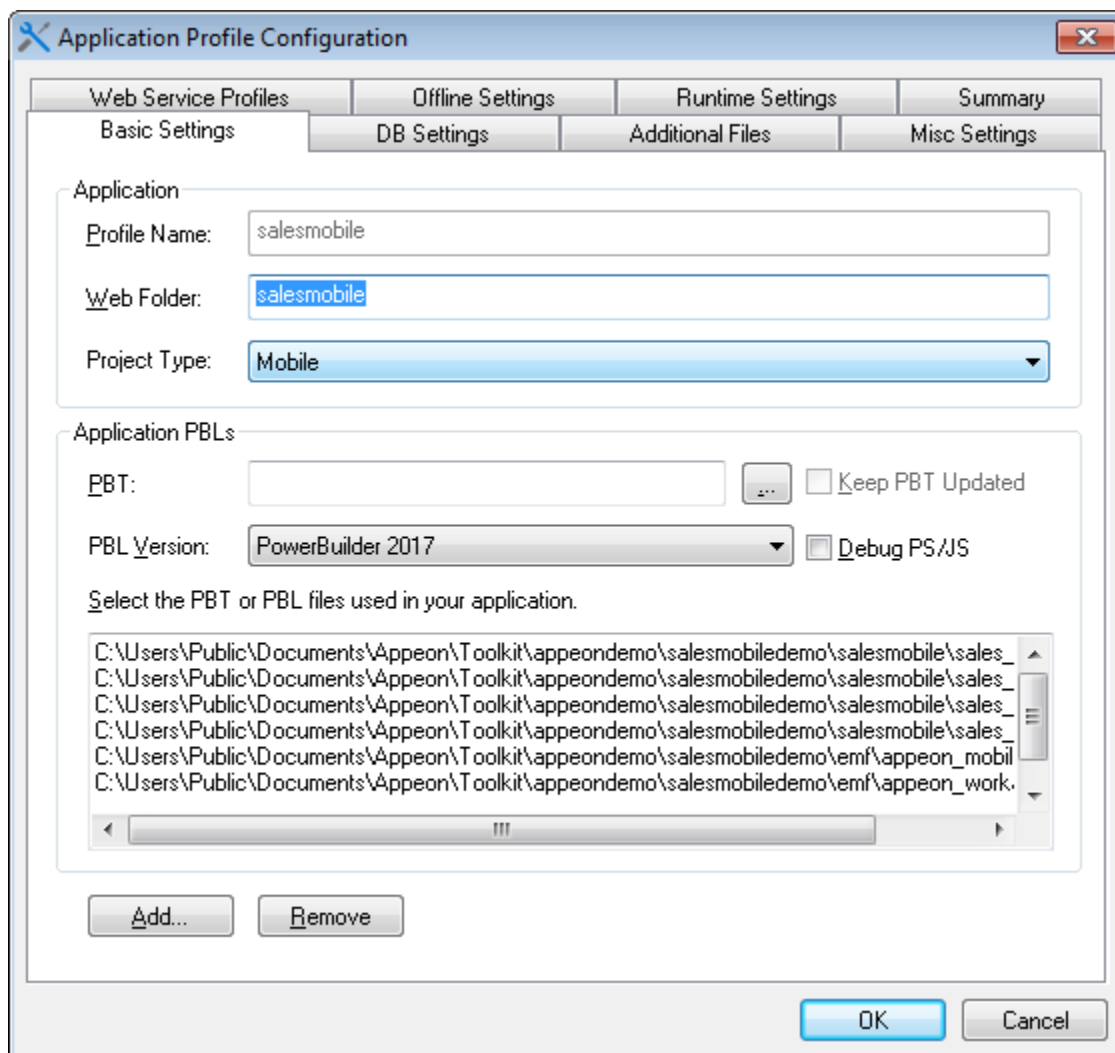
Tab	Settings	To make changes effective, you must...
	and description displayed in the Appeon Workspace and the target mobile device the application apply for. See Section 4.2.1.2.7, “Runtime Settings” .	
Summary	A summary view of the settings. See the Section 4.2.1.2.8, “Summary” section for more details.	Perform a full deployment on the application.

4.2.1.2.1 Basic Settings

After you click **Add** or **Edit** on the **Application Profiles** tab, the **Application Profile Configuration** window is displayed with the **Basic Settings** tab selected by default, as shown in the following figure.

The **Basic Settings** tab allows you to modify the web folder, project type, PBT, PBL version, PBL location, and file generation mode.

Figure 4.13: Basic Settings



Profile Name -- The application profile name is used to identify the PowerBuilder application during the entire PowerBuilder-to-Web or PowerBuilder-to-mobile process. It is also used in the appeondb database in PowerServer to identify the DataWindow syntax, profile, and registry information for an Appeon application.

The Profile Name cannot contain double-byte characters (such as Chinese, Korean, or Japanese characters) or special characters (such as, \, /, :, *, ?, ", <, >, or |).

Web Folder -- The specified text is used as the name of the folder created under the Web root (or document root) of the Web server for storing the Web or mobile application files when the application is deployed, therefore, the URL for accessing the deployed application will be in this format: *http://web_server:port/web_folder/*. The Web Folder can contain a combination of letters, underscores ("_"), and numbers.

Avoid setting Web Folder as existing folder names in the Web server Web root (or document root), because: (1) If a Web or mobile application uses an existing folder under the Web root as the Web Folder, a large number of Web or mobile files belonging to the application will be deployed to that folder, making it difficult to use the folder for its original purpose. (2) If a Web or mobile application uses a folder that is reserved for the Web server or PowerServer, some important files belonging to the Web server or PowerServer may be replaced or even removed when the application is undeployed. The following are folder names reserved for the Web server or PowerServer that should not be used as Web Folders: *apeon*, *classes*, *docs*, *images*, *ir*, *wst*, and *WEB-INF*.

The Web Folder for an application profile can be changed, and the Web or mobile files stored at the Web server can have different versions for the client to access, but Appeon only remembers the Web Folder in the last deployment, and performs the application packaging and application undeployment based on the Web or mobile files generated in the corresponding folder.

Project Type -- The Project Type determines whether your PowerBuilder application will be configured for the Web migration process, or the mobile migration process, or both.

- **Web** -- Deploys a Web app only. The Web project type supports the most PB features including Windows PC technologies, such as DLLs/OCXs.

Note: This type is not supported by the PowerServer Mobile (PB Edition).

- **Mobile** -- Deploys a native Mobile app only. The Mobile project type supports mobile-specific functionalities instead of Windows PC technologies.
- **Both** -- Deploys both a native Mobile app and a Web app. The Both project type supports the same features as the Mobile project type. If your app uses Windows PC technologies, such as DLLs/OCXs, you cannot use the Both project type.

Note: This type is not supported by the PowerServer Mobile (PB Edition).

Application PBLs -- Select the PBL version and add application PBL files using any of the following methods:

- Click the browse ... button and choose to add an application Target file (*.pbt).

By adding PBLs using an application Target, all PBLs in the Target are automatically added to the application PBLs list. This is the recommended way to add PBLs, as it reduces the likelihood of forgetting to add a required PBL.

- Click the **Add** button and add PBL files without using an application Target.
You can add one or multiple PBL files at the same time. To select multiple files, hold down the Ctrl key and choose which files to add.
- Copy and paste (Ctrl+C and Ctrl+V) the path list of PBLs into the Application PBLs field.

Keep PBT Updated -- Keep this checkbox checked if you want Appeon to automatically update the PBLs every time you deploy an application. This ensures that any newly added PBLs will be deployed and so avoids missing PBLs after deployment.

Debug PS/JS -- When you are debugging or tuning the application, you can check this checkbox, so that unencrypted JavaScript files will be generated, and the PowerBuilder source code will be provided as comments in the JavaScript files for easy reference. When you are releasing the application to the production use (after the debug and fine-tune), you are recommended to **not** check this checkbox, so that encrypted JavaScript files will be generated under the **Release** mode to prevent the source code being viewed. When you change the mode (between **Debug PS/JS** and **Release**), you must perform a full deployment to make the new setting effective.

4.2.1.2.2 DB Settings

The DB Settings tab allows you to specify the database types and the transaction objects that the application uses, as shown in the following figure. When configuring the database types, it is not necessary to specify the actual databases that the application uses, it can be any database of the same type. However, when specifying the transaction objects, you must specify the actual transaction objects that the application uses.

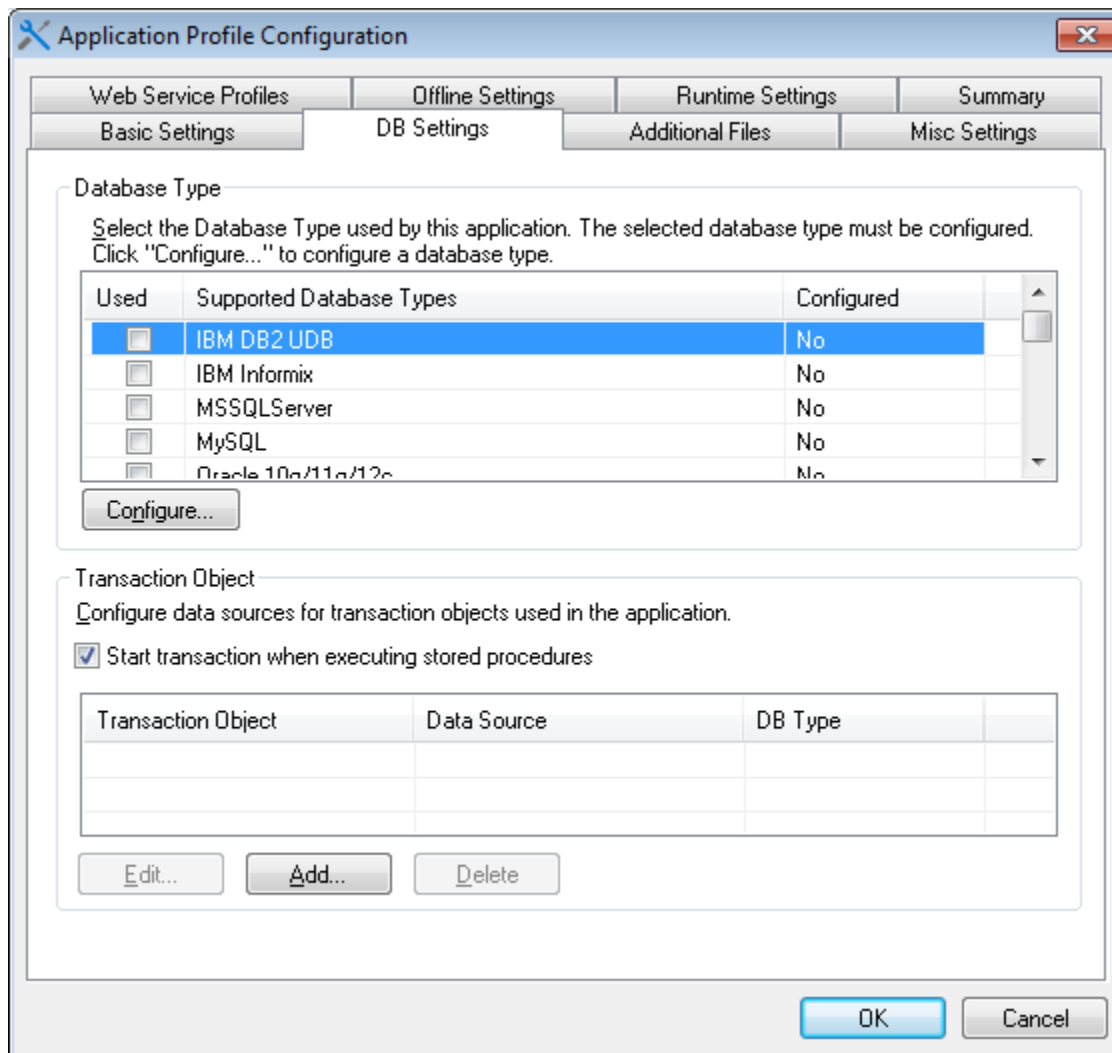
What the DB Type is used for

PowerBuilder SQL statements must be converted into the correct database syntax before they can be used to retrieve data from the database that the Web or mobile application connects to. The DB Type settings enable PowerServer Toolkit to apply the correct database driver type for generating correct database syntax for PowerBuilder SQL statements.

Set the database types for the application

Select the database type(s) used by the application, as shown in the following figure.

Figure 4.14: Database types



The following table describes each element on the DB Type tab.

Table 4.2: Database type settings

Column	Description
Used	You can click the check boxes to select the database types used by the application. More than one database type can be selected, but only the configured database types are selectable.
Supported Database Types	This column lists all the database types that are supported by Appeon.
Configured	This column is identical to the Configured column in Section 4.2.2, “Managing database type profiles” . It indicates whether a profile has been configured for the database type. If this column of the desired database type is "No", you must click the Configure button to configure a profile for it before you can use it. Configuring a database type profile in this tab is the same as configuring it in the DB Type Profiles tab. Once a database type profile is configured

Column	Description
	successfully, the Configured column in both this tab and the DB Type Profiles tab changes from "No" to "Yes".

Specify the transaction object and the data source

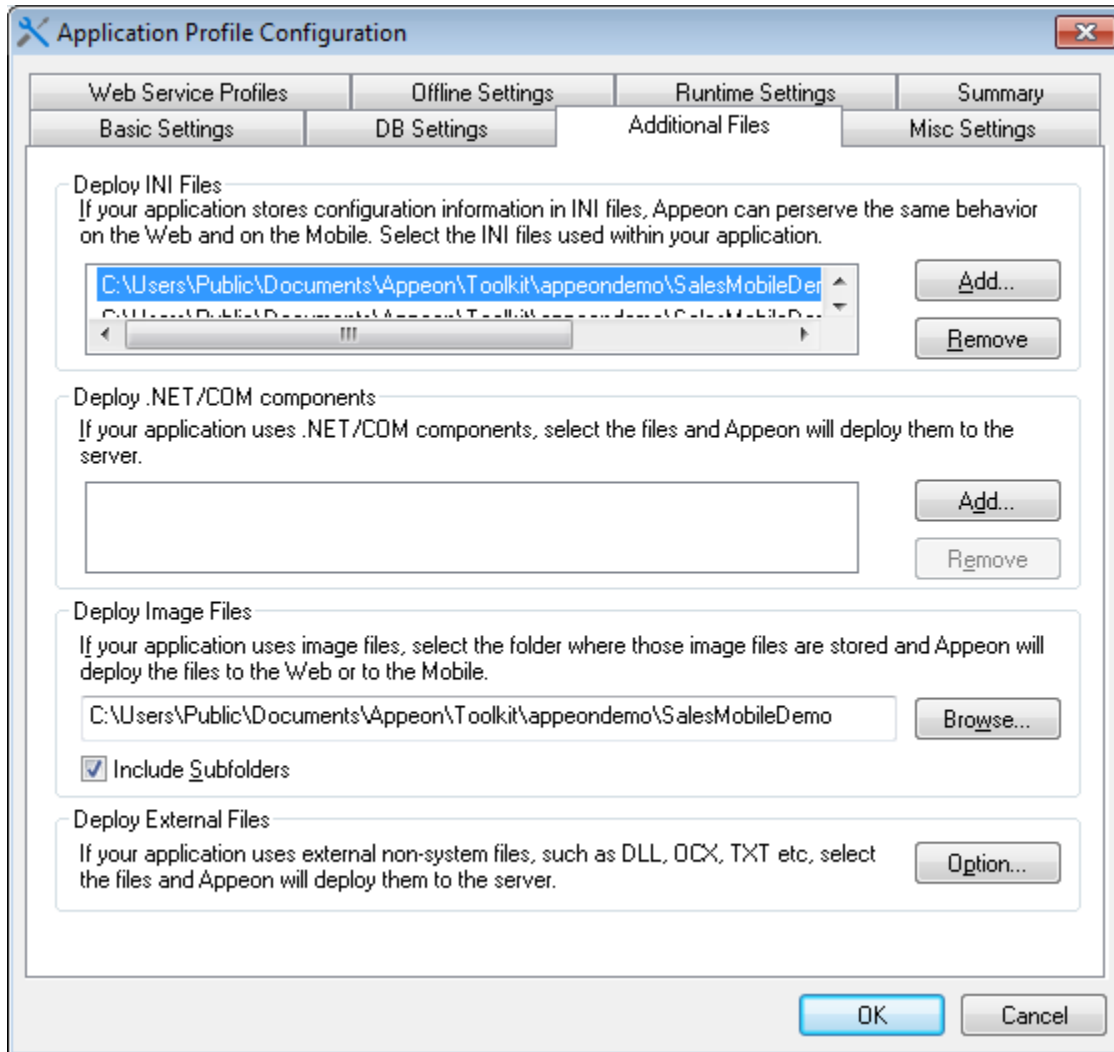
Click **Add** or **Edit** to configure a transaction object and specify the corresponding data source. Refer to [Section 4.1.5, “Declaring transaction object\(s\)”](#) for detailed description.

Start transaction when executing stored procedures

It is strongly recommended that you keep this option enabled. The only circumstance you may consider disabling it is when there is commit statement executed inside a stored procedure (SP) and there is no need to commit in the PB code; the reason is that by default PowerServer will start a transaction once an SP is executed, and the user will be able to see details of this transaction in the Active Transactions tab in AEM; when commit is executed inside the SP, the transaction will end and disappear from the database end, but as there is no commit in the PB code, this transaction will still appear as an active transaction in AEM, which will mislead users to think this is an uncommitted transaction. To avoid this misunderstanding, you can disable this option, so PowerServer will not start a transaction when an SP is executed.

4.2.1.2.3 Additional Files

"Additional files" refer to the files that are outside the application source code, but are necessary for running the PowerBuilder application, including INI files, .NET/COM components, image files, and external files, such as DLL/OCX files, text files, etc.

Figure 4.15: Additional Files tab

Deploy INI Files

If a PowerBuilder application uses an initialization (INI) file which contains user preferences, specify the INI file in this tab so that Appeon can deploy the file for the application.

Click **Add** to add the INI file(s) into the Deploy INI Files list box.

How INI files are supported in Appeon

During deployment, the specified INI file(s) are converted to XML file(s) and deployed to PowerServer. The XML file(s) act as mock INI file(s). When a Client runs the deployed application for the first time, a copy of the XML file is created in PowerServer and keeps the Client's profile information.

For a Client to use the correct profile information in its subsequent visits to the application, the Client browser must be cookie-enabled. Otherwise, each time the Client visits the application, a new copy of the XML file is created in PowerServer, and the new copy only contains the information of the original INI file.

You can delete the XML files that are unused for a period using AEM. For more information on how to maintain XML files in PowerServer, refer to Section 5.3.3.3.1, “Auto cleanup”

in *PowerServer Configuration Guide for .NET* or in *PowerServer Configuration Guide for J2EE*.

Deploy .NET/COM components

If a PowerBuilder application uses .NET/COM components which contains executable files (.exe), and COM/COM+ components, specify the .NET/COM components file in this tab so that Appeon can deploy the file for the application.

Click **Add** to add the .NET/COM components file(s) into the Deploy .NET/COM components list box.

Deploy Images

Click **Browse** to specify which folder contains the image files that will be used in the Web or mobile application. If you do not specify the path of the image files, the path to the application PBL files will be displayed as the default path of the image files.

The image files can be in any format supported by PowerBuilder, apart from:

- unsupported run-length encoded (RLE), and
- unsupported Windows metafile (WMF)

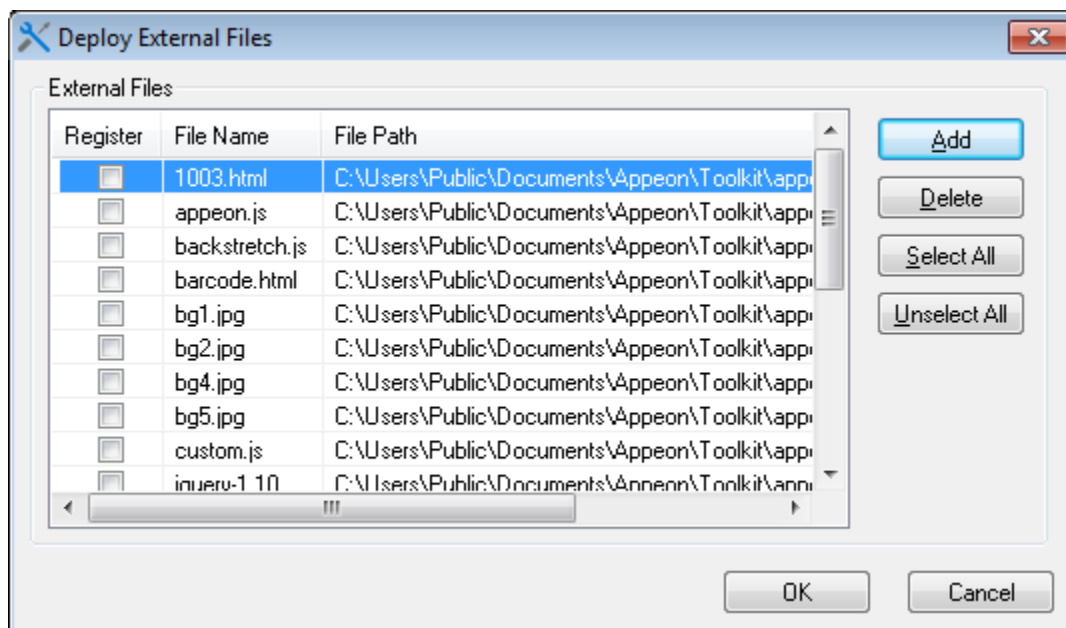
All the image files in the specified folder will be deployed to the *%Web Folder%/images* folder in the Web Server. *%Web Folder%* stands for the folder storing the Web application files or the mobile application files. The image files may fail to be displayed on the Web or in Appeon Workspace, or on mobile devices if the file names are stored in the database or dynamically generated. To resolve this, you can copy the image files directly to the *%Web Folder%/images* folder, but make sure that the file names are all lowercase; otherwise, the Web or mobile application may fail to load them.

Deploy External Files

If your application calls any custom user DLL/OCX files (for Web applications only), which are also called external non-system DLL/OCX files in this tab; or if your application calls any other external files, such as text files, PDF files, image files etc., you can specify them here and deploy them to the Web server. Deploying DLL/OCX files is used as an example in this guide.

Click **Option...** in the **Deploy External Files** group box to open the Deploy External Files dialog box (as shown in the following figure) and select the required external files to be deployed.

Figure 4.16: External Files Automatically Deploy and Download



The following table introduces the elements in the **Deploy External Files** dialog box.

Table 4.3: Deploy External Files dialog box

Column	Description
Register	Allows you to select whether the file will be automatically registered after it is downloaded to the Client at runtime. Click Select All to select all files and Unselect All to de-select all of them. If you choose not to automatically register a file, you can register it manually after it is downloaded.
File Name	Lists the added files. Click the header of this column to display the files in alphabetical order. You can add any file of any type.
File Path	Displays the file location.

The following table describes how to add and remove files in the **Deploy External Files** dialog box.

Table 4.4: Deploy External Files dialog box

To Do This	Do This
To add files	Click the Add button. A standard File Selection dialog box is displayed. And then select the files from your local machine. The files can be stored in any location.
To remove files	Select a file or multiple files using Ctrl or Shift keys, and click the Delete button to remove the selected files from the list.
To save the settings	Click OK .

To Do This	Do This
	Appeon will copy the files to a temporary folder and then deploy them to the Web Server during deployment.

How the deployed DLL/OCX files work on Web

To successfully call the deployed DLL/OCX files in the Web application, you need to make sure that the DLL/OCX files are successfully downloaded and installed to the Client during the initial run of the deployed Web application.

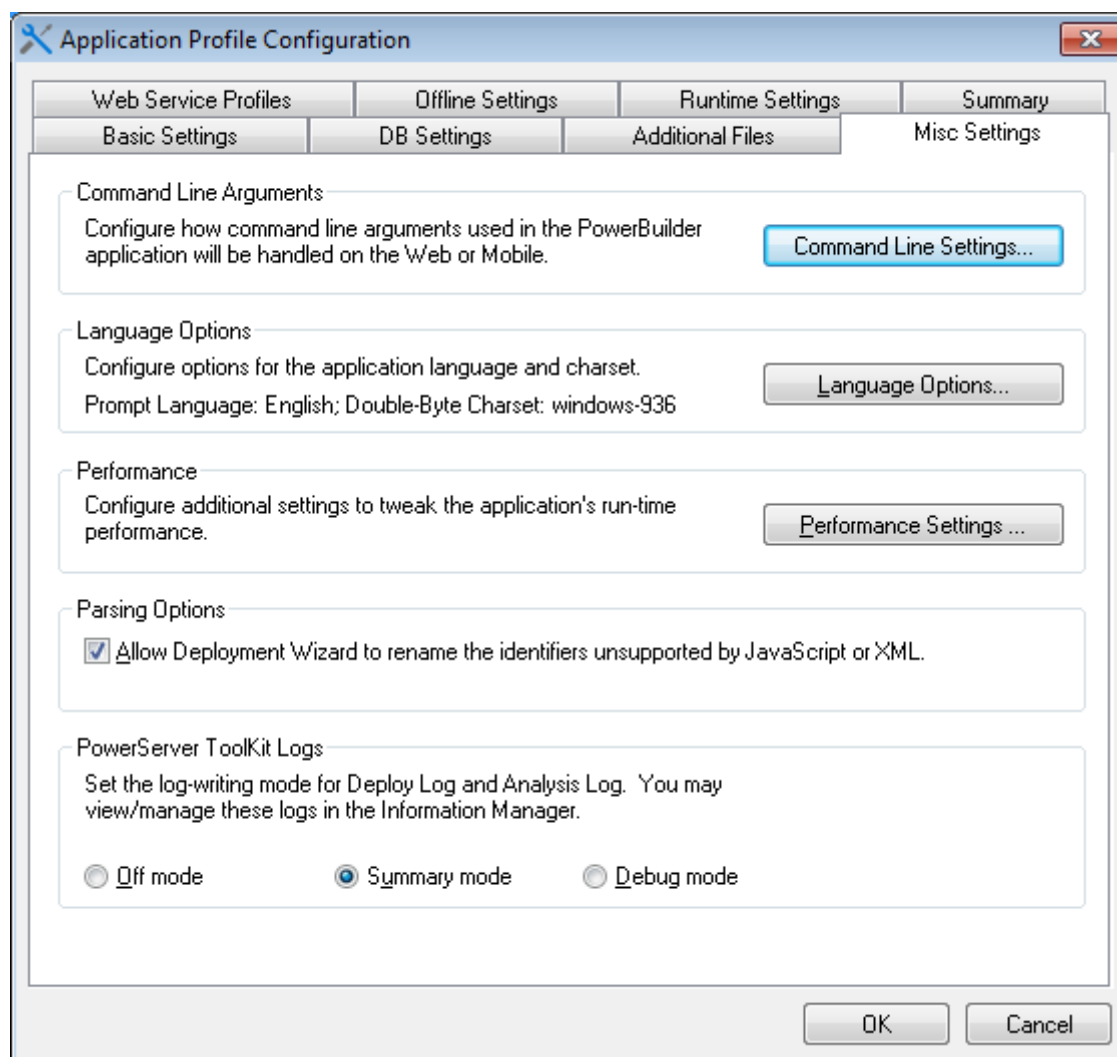
AEM provides options for manually or automatically downloading and installing the DLL/OCX files, and for specifying the installation location (by default the files are saved to the "plugin" folder under the application directory on the client machine). For detailed instructions, refer to Section 5.4.4.3, "DLL/OCX Files" in *PowerServer Configuration Guide for .NET* or in *PowerServer Configuration Guide for J2EE*.

Where the external file will be saved after downloaded to the client

The external file will be automatically downloaded and saved to the client when the deployed application is run for the first time. The default location is the "plugin" folder for each application. For more details, refer to Section 7.1, "What files will go to the plugin folder and how to access them" in *Workarounds & API Guide*.

4.2.1.2.4 Misc Settings

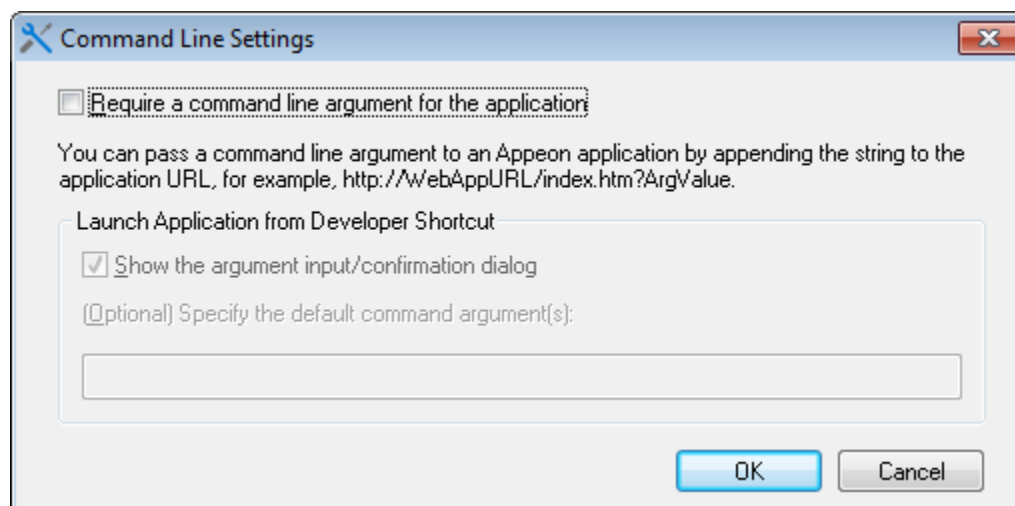
The Misc Settings tab page enables you to modify settings for command line arguments, application languages, double-byte charset, and log-writing modes, and allows you to configure the runtime performance and parsing options.

Figure 4.17: Misc settings

Command Line Arguments

If an application needs to use command line arguments, enable the command line argument option, as shown in the following figure.

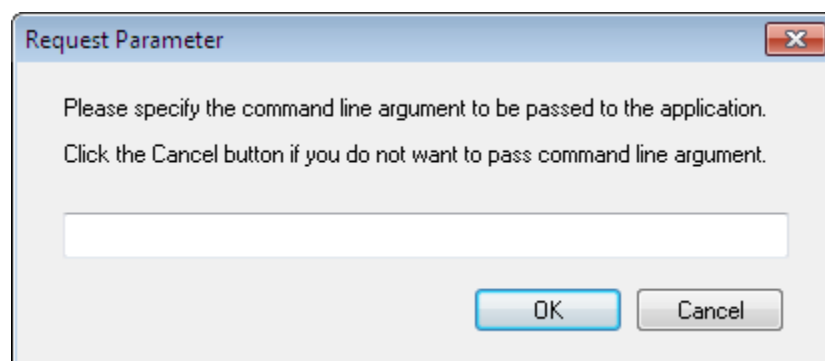
Click **Command Line Settings** on the Misc Settings tab. The Command Line Settings dialog box is displayed, as shown in the following figure.

Figure 4.18: Command Line Settings dialog box

Require a command line argument for the application option: Enables the command line argument option.

With this check box selected, the following two options will be enabled. Note that they are effective only if you run the application from the PowerServer Toolkit.

- **Show argument input/confirmation dialog** option: A dialog box is displayed after you launch an application from the PowerServer Toolkit but before the application starts. This dialog box enables you to specify the arguments for the application.
- **Specify the default command arguments** field: The specified default arguments are directly attached to the application URL when the application is run from the PowerServer Toolkit.
- If both options are specified: A dialog box pops up with the default arguments filled in, as shown in the following figure. You can either change the default arguments or leave them alone.

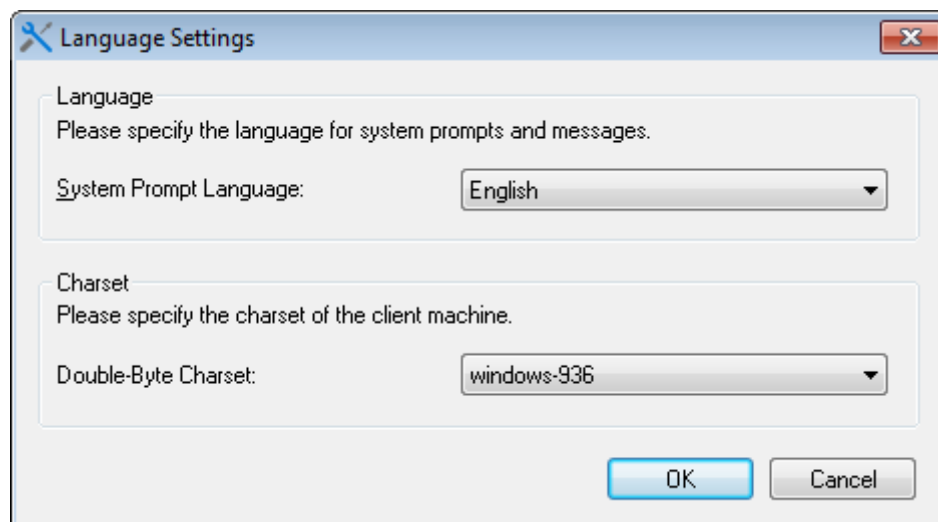
Figure 4.19: Command line argument dialog box

To run an application with arguments from Clients, type the application URL with the arguments. The format of an application URL with arguments is: `http://AppURL/index.htm?ArgValue`. *ArgValue* indicates the arguments that will be passed to the application. The arguments must be separated by commas if there are more than one argument.

Application Language and Double-Byte Charset

Step 1: Click **Language Options** on the Misc Settings tab. The **Language Settings** dialog box is displayed, as shown in the following figure.

Figure 4.20: Application language settings



- **System Prompt Language:** Select a language from the **System Prompt Language** dropdown list box (Only English available when you are using an English version of Apeon PowerServer).

System prompts are error messages, informational messages, warnings, notices, and prompts that are displayed when you run Apeon applications.

- **Client Charset:** **Skip this if your application is a Web application or if your mobile application does not use any string function that ends with "A"**, otherwise, select a charset the same as that of your PowerBuilder development environment for the mobile client.

If your application uses string functions that end with "A" (e.g. FillA, LeftA, LenA, MidA, PosA, ReplaceA, and RightA) to process the double-byte character, these functions may return different values on iOS and Windows, because iOS system uses UTF-8 encoding while the Windows system uses UTF16 encoding. Due to this difference, in order for these functions to return the same values on iOS and Windows, you need to specify the charset here the same as the charset of your PowerBuilder development environment.

And make sure the mobile client supports the specified charset, otherwise, these functions could not return the correct values or may even cause the application to crash. For the sake of safety, we suggest you not use string functions that end with "A" (e.g. FillA, LeftA, LenA, MidA, PosA, ReplaceA, and RightA) to process the double-byte character, use the functions without "A" instead (e.g. Fill, Left, Len, Mid, Pos, Replace, and Right), so as to avoid any potential issue that could be caused by specifying an improper charset or the mobile client not supporting the specified charset.

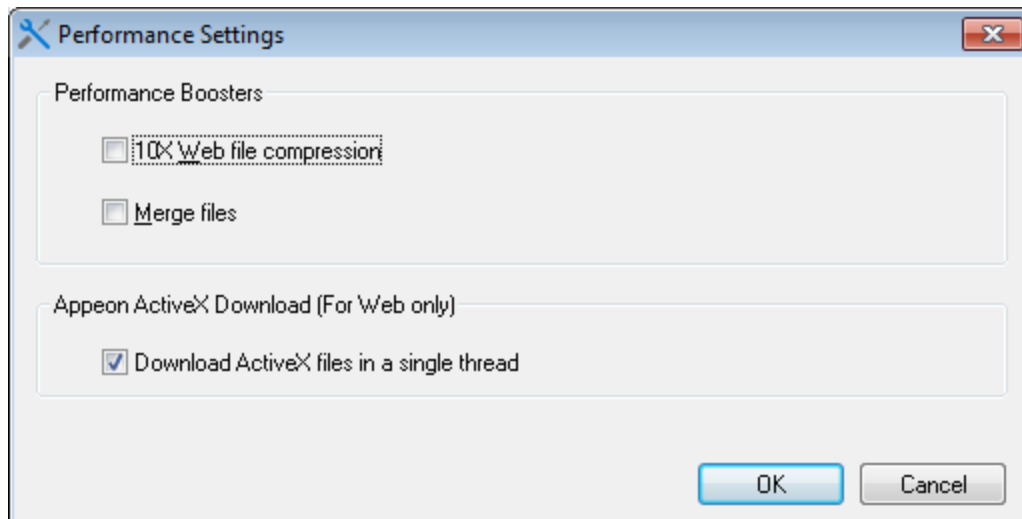
Step 2: Click **OK** to save the settings.

Performance Settings

Appoon recommends that you take advantage of the performance settings to boost the application's runtime performance. To achieve the best performance, perform the following steps:

Step 1: Click the **Performance Settings** on the Misc Settings tab. The Performance Settings dialog box is displayed, as shown in the following figure.

Figure 4.21: Performance settings



Step 2: Select the performance features based on your needs.

The following table describes how to use the performance features.

Table 4.5: Performance features

Setting	Description	When To Use
10X Web File Compression	Compresses files when they are transferred over the network.	Always
Merge files	Merges the small files during the application deployment. The small files will be downloaded to the client in one file package at one call, instead of being downloaded one by one at separate calls.	In the production stage.
Download ActiveX files in a single thread (for Web applications only)	Downloads the two ActiveX files in the same thread at runtime. Using the same thread to download the two ActiveX files can speed up the download under particular network conditions. For more information about the Appoon ActiveX file, refer to Section 5.1.2, “Installing Appoon Xcelerator plug-in” in <i>Supported PB Features for PowerServer Web</i> .	Always

Parsing options

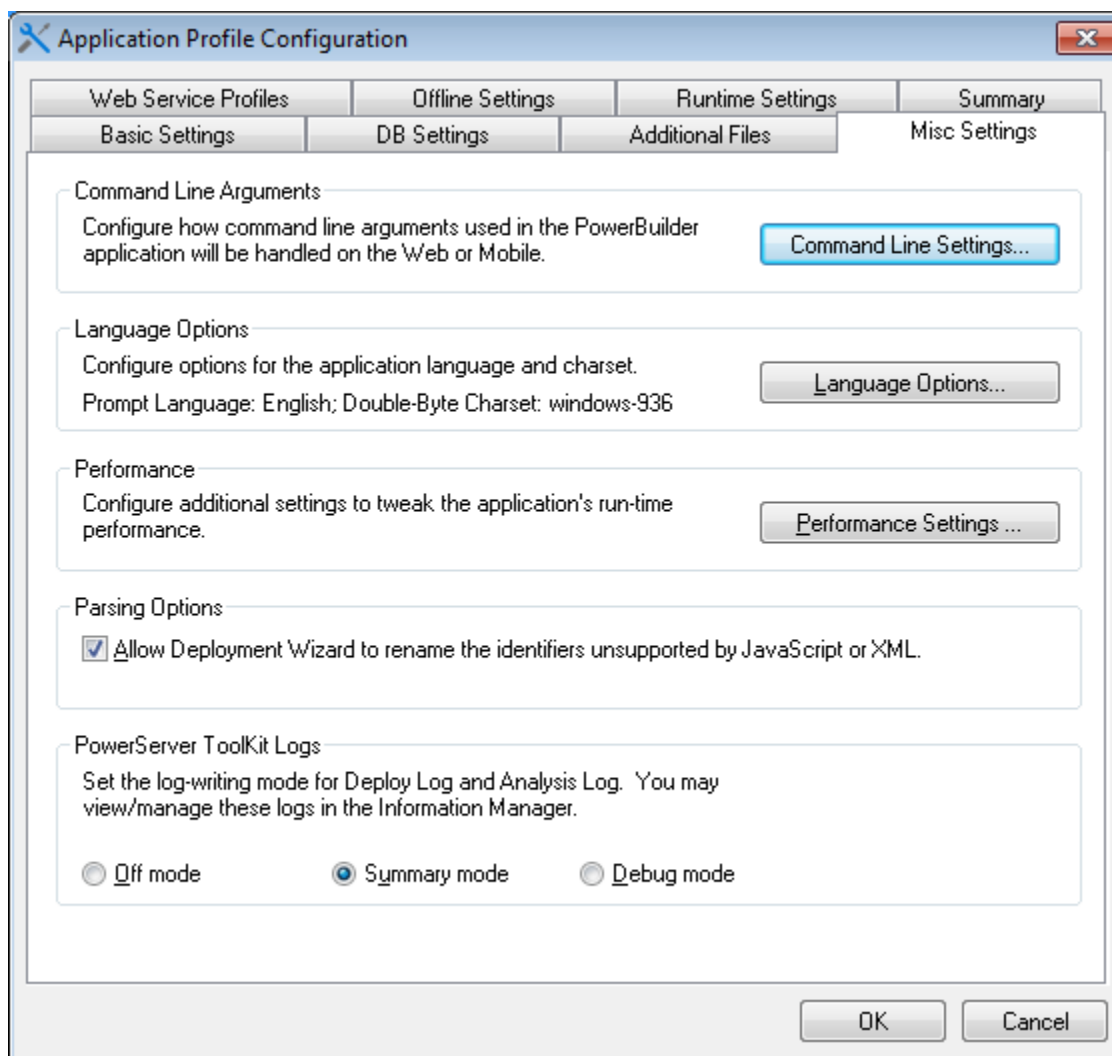
Unsupported identifiers refer to identifiers that are supported in PowerBuilder but unsupported in Appoon, due to the different naming convention between PowerScript

and JavaScript/XML. For detailed explanation of such unsupported identifiers, refer to Section 5.1.2.2, “Identifiers” in *Supported PB Features for PowerServer Mobile* or in *Supported PB Features for PowerServer Web*.

With this automatic-renaming option enabled, when the Apeon Deployment Wizard deploys an application, it automatically detects unsupported identifiers in the application and renames these identifiers to supported ones in the files it generates for the application, while keeping the application source code unchanged. After the deployment, you can get the list of all the identifiers that are renamed in the deployment log in [Chapter 9, Using Information Manager](#).

The automatic-renaming feature will not work if the option is disabled. In this case, you must rename the unsupported identifiers by yourself, following the guidance of the UFA report and Section 5.1.2.2, “Identifiers” in *Supported PB Features for PowerServer Mobile* or in *Supported PB Features for PowerServer Web*.

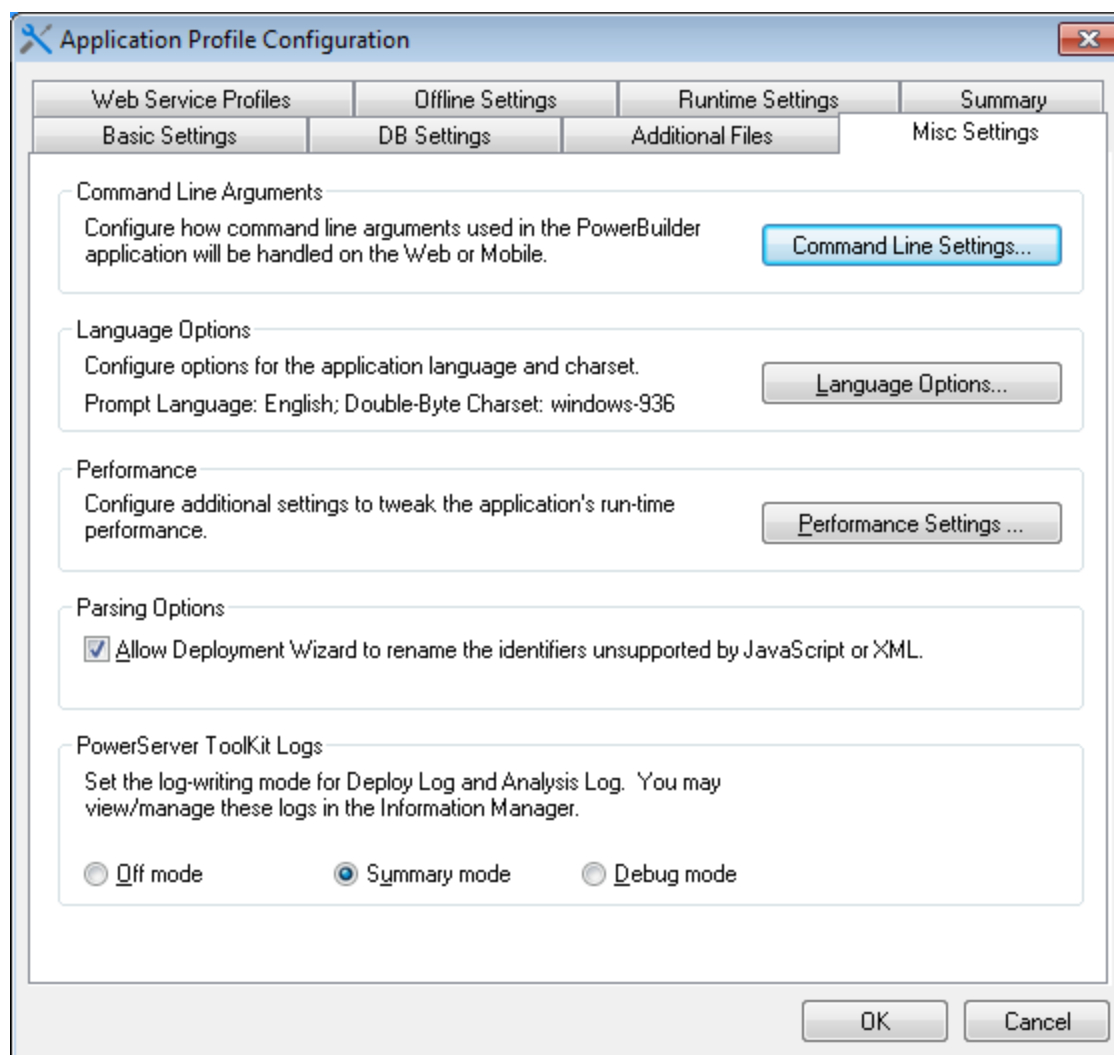
Figure 4.22: Parsing options



PowerServer Toolkit log-writing mode

The log-writing mode determines the content of the log files generated by PowerServer Toolkit, including the deployment logs and feature analysis logs. Select the desired mode from the PowerServer Toolkit Logs box, as shown in the following figure.

Figure 4.23: Log-writing mode



The logging options enable you to select the level of information contained in the logs according to your needs:

Table 4.6: Log writing options

Mode	Description
Off mode	Generates no log files. This mode offers the fastest performance since nothing is written to a log file.
Summary mode (Default)	Generates log files with basic execution information. This is useful for tracking errors that have occurred, but inadequate for detailed troubleshooting. Use this mode once the application is stable.
Debug mode	Generates log files with detailed execution information for troubleshooting obscure and esoteric issues. This is useful for technical support, but performance speed will slow down when using this mode.

You can view or delete these log files. For detailed instructions, refer to [Chapter 9, Using Information Manager](#).

4.2.1.2.5 Web Service Profiles

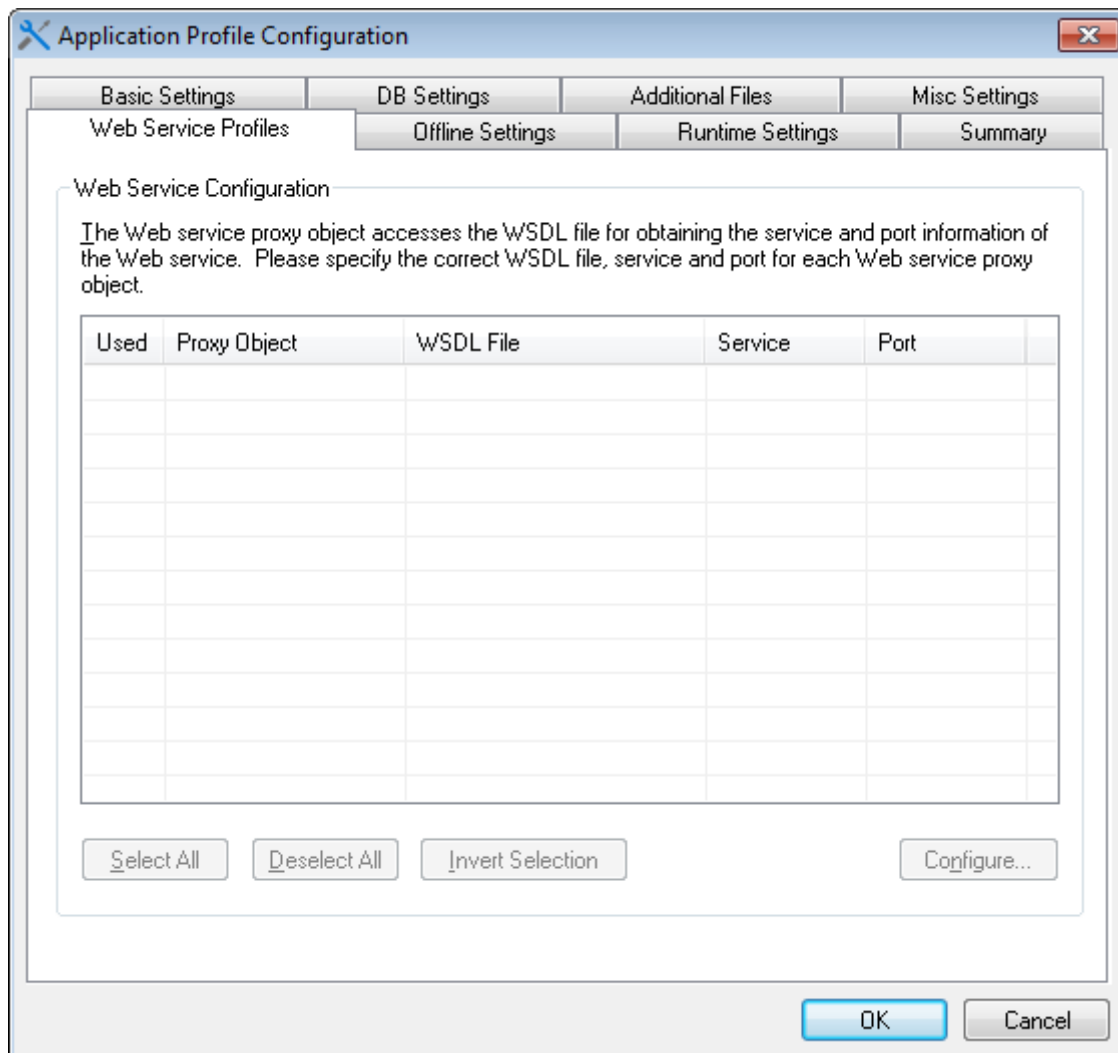
A Web service profile contains the settings for the Web service(s) to be invoked in the Apeon deployed application, including the WSDL file, service name and port to be used by the Web service.

After you specify the PBL files of the application in the Basic Settings tab, the Web service proxy objects, WSDL files, service, and port, which are used in the application, are automatically added to the Web Service Profiles tab, as shown in the following figure.

A valid WSDL file must meet the following requirements:

- The WSDL file must be a .wsdl or .asmx file.
- The WSDL file must reside on the local computer. Apeon cannot access the WSDL file in a remote machine.
- WSDL files used in the same application cannot have the same name, even if they reside in different locations.

Figure 4.24: Web Server Profiles tab

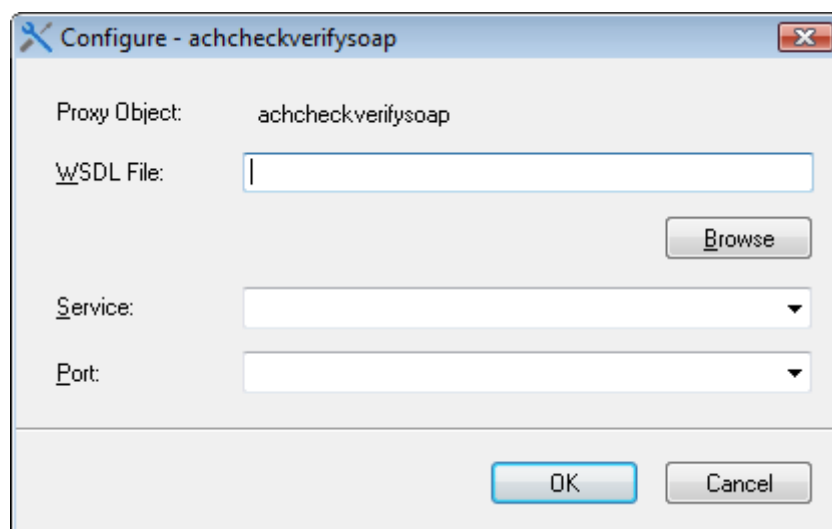


Choose which proxy object(s) will be invoked in the deployed application by selecting the Used column of the object. Click **Select All** to select all objects; click **Deselect All** to deselect all objects; or click **Invert Selection** to select all objects which were not selected and deselect the previously selected objects.

If the WSDL file, service and port are not automatically specified for a proxy object, follow the steps below to select them:

Step 1: Select the proxy object and click **Configure**. The Configure dialog appears, as shown in the following figure.

Figure 4.25: Configure dialog box



Step 2: Click **Browse** to select a WSDL file on the local computer, and then select the service and port.

A WSDL file may contain multiple services and each service may contain multiple ports.

Step 3: Click **OK** to save the configuration and return to the Web Service Profiles tab.

4.2.1.2.6 Offline Settings

This section is for mobile applications with offline capabilities and users that have purchased the **Enterprise** edition with offline support.

If you are using the **PowerServer (PB Edition)**, you will **not** be able to use this offline feature. Be sure to de-select (not check) the **Support Offline** box on the **Offline Settings** tab, otherwise you will not be able to deploy the application to the **PowerServer (PB Edition)**.

Introduction

The offline feature enables developers to

- design an application that can read and write data from the local databases (SQLite or UltraLite), and synchronize the local database with the back-end database server.

You will need to first use the DBParm properties (see transaction object in *Supported PB Features for PowerServer Mobile*) and then configure and deploy the database file to the

local device through PowerServer Toolkit, so the application can access the local database. For detailed instructions, refer to PowerServer Mobile Offline Tutorials.

To synchronize the UltraLite local database with the database server, you could take advantage of the PB synchronization object **MLSync** and the SQL Anywhere synchronization server **MobiLink**. To synchronize the SQLite local database with the database server, you would need to develop your own synchronization solution, because SQLite cannot work with MobiLink or any other equivalent product or solution.

- deploy the application to start at the offline mode

If the application is deployed to start at the offline mode, the app can start without connecting to the PowerServer except for the first time when it needs to download the configuration information for local use.

Local database requirements

The local database must be either UltraLite or SQLite. When using the local database, be aware of the following requirements or limitations:

- Required database version.

PowerServer Mobile 2017 only supports UltraLite 17, and the corresponding MobiLink server must be at the same version. If they are not at the required versions, upgrade them by following the steps in the SQL Anywhere 17 help.

The SQLite database should be at version 3.x or later.

- Using SQL Anywhere for development.

When developing the PowerBuilder application, the SQL Anywhere database should be used rather than the UltraLite database, because there are many issues using UltraLite with PowerBuilder according to our test results. It is recommended that after finishing the PowerBuilder application development, export the table schema and data from the SQL Anywhere database to the UltraLite database (by using the tool provided by Mobilink), and then use the UltraLite database in the offline application.

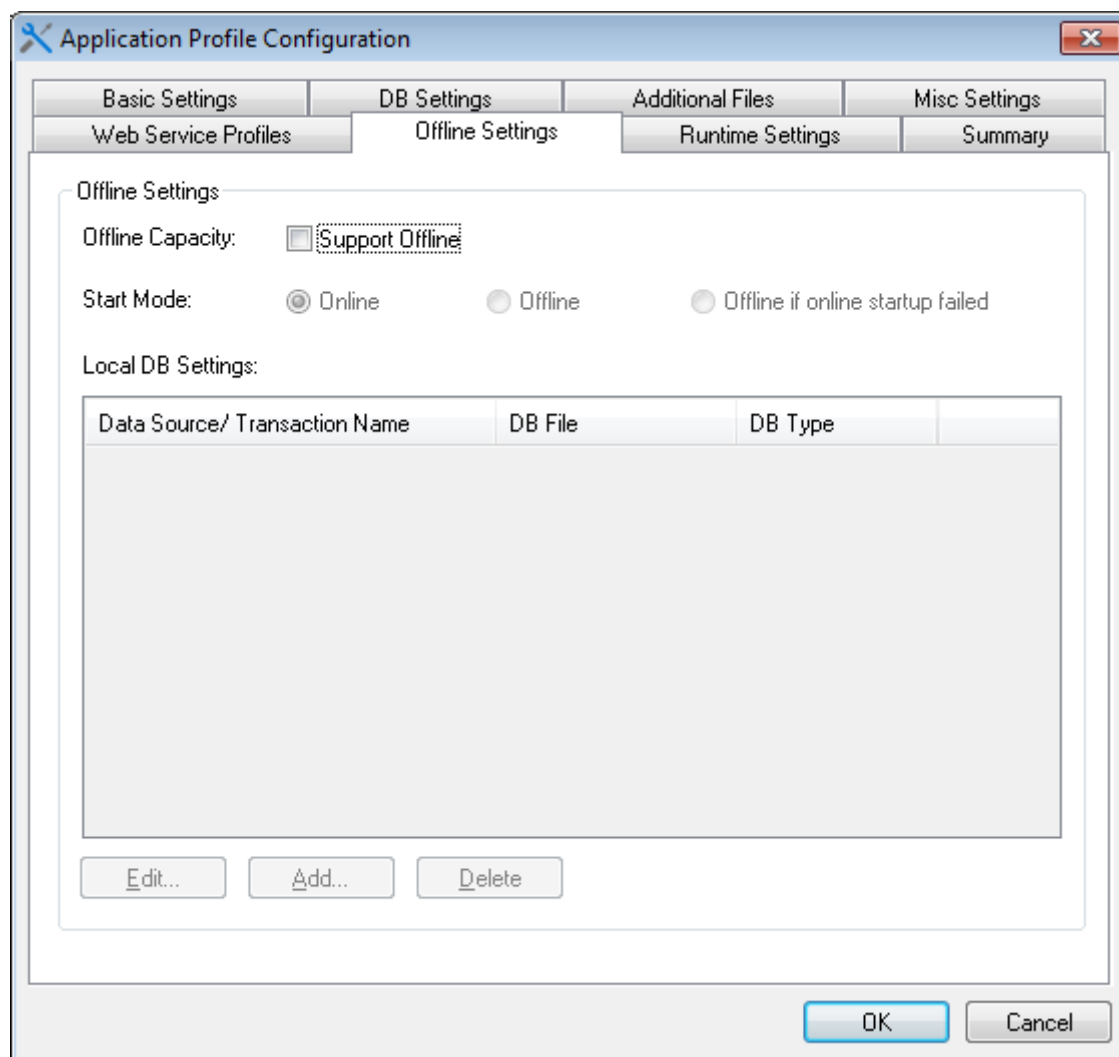
- Creating synchronization scripts for SQLite.

Unlike UltraLite which can synchronize data through the MobiLink server, SQLite cannot work with MobiLink servers, or any other synchronization servers similar to MobiLink. Therefore, you would need to develop your own synchronization scripts if synchronizing the SQLite local database with the database server.

Specify the offline settings for the application

The **Offline Settings** tab allows you to configure the offline capacity, the start mode, and the local DB settings for the mobile applications. These configurations and configured files will be deployed to the PowerServer along with the application.

Figure 4.26: Offline settings



Offline Capacity: Check the **Support Offline** checkbox if your application needs to enable support for the offline features.

If it is not checked, the application works as a standard Apeon mobile application that always connects to the PowerServer.

Start Mode: Allows you to set whether an application to be launched online or offline. The online mode requires network connections from the mobile device to the PowerServer, while the offline mode does not.

The following table describes the three start modes in details.

Table 4.7: Start mode

Start mode	Descriptions
Online	<p>Network connection from the device to the PowerServer is needed when the application starts and runs.</p> <p>If the application is set to start online, then it starts and runs the same way as the standard Apeon applications (except that the application can access the</p>

Start mode	Descriptions
	local database), which means, it must connect to the PowerServer for loading the application file, getting updates, managing user sessions, etc.
Offline	<p>The application starts without connecting with the PowerServer, except for the first time.</p> <p>For applications that starts offline, the application must connect to the PowerServer to download the configuration information only for the first time, after that, the application can run locally without needing any network connection.</p> <p>If your application needs to access the database, then you would need to enable the local database accessibility using DBParm of transaction object in <i>Supported PB Features for PowerServer Mobile</i> and configure the database file deployment in the Local DB Settings table below, in order for the database file to be downloaded and accessed on the local device.</p> <p>Unlike the other Appeon applications, the offline application will not be automatically updated, because it runs locally without connecting with the PowerServer; therefore, you will need to write code in the application to call the APIs to get updates from the PowerServer. Refer to the section called “of_checkupdate” in <i>Workarounds & API Guide</i> and the section called “of_applyupdate” in <i>Workarounds & API Guide</i>.</p>
Offline if online startup failed	The application will first try to start online (in an attempt to connect with the PowerServer), if the online startup fails (due to the bad network connection, wrong URL, etc.), then it starts offline.

If the application has already run as an offline application and then is re-deployed as an online application, it will need to get updates (the new start mode) from PowerServer first and then restart, then it will run as an online application. Which means, you will need to write code in the app to call APIs to get updates first, and then the end user will need to quit and run the application again as the application cannot automatically restart.

If the application has already run as an online application and then is re-deployed as an offline application, it will need to restart and then it will run as an offline application. Which means, the end user will need to quit and run the application again as the application cannot automatically restart.

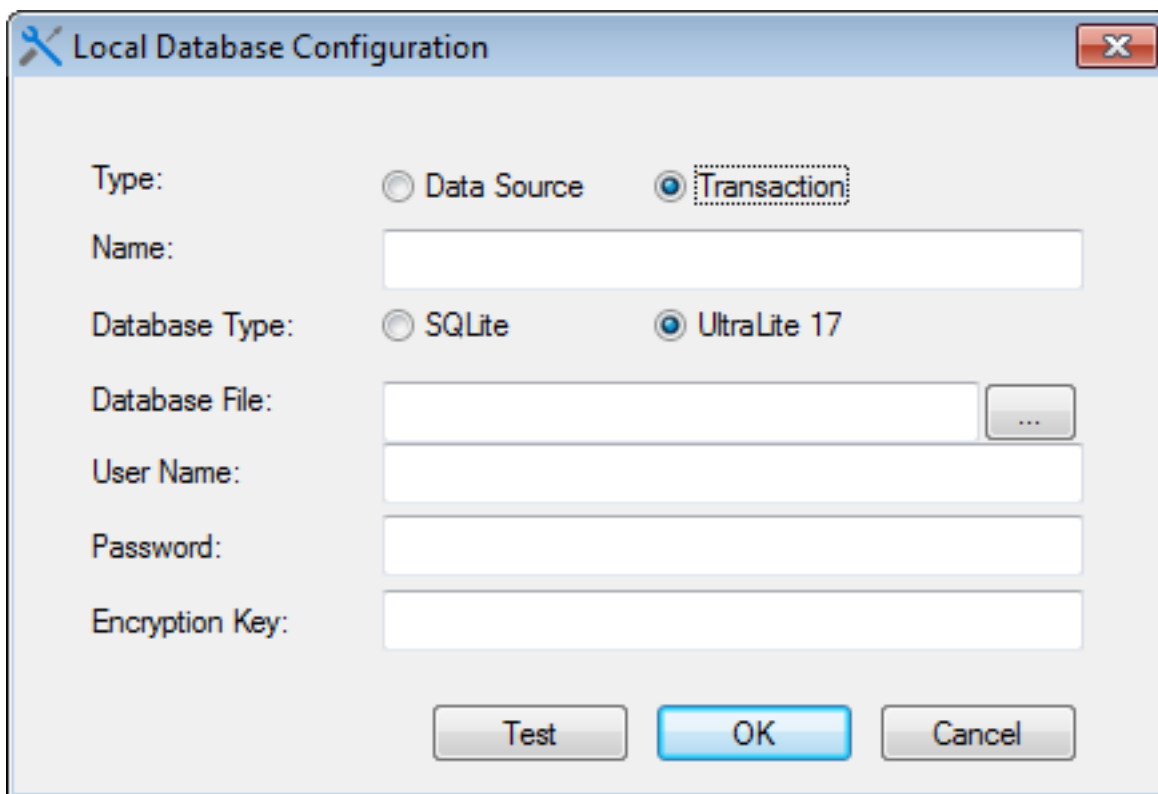
Local DB Settings: Specify the data source/transaction name and the database file that will be deployed to the PowerServer, and later downloaded to the mobile device and accessed by the application at runtime.

You can specify these items by adding a new local database, editing, or deleting an existing one.

- To add a new local database, click **Add**, and the **Local Database Configuration** window appears.

In the **Local Database Configuration** window, specify the items (according to the following table), click **Test** to test the connections, and then click **OK** to save the settings.

Figure 4.27: Add a local database



The following table describes the items in the **Local Database Configuration** window.

Table 4.8: Local database configuration

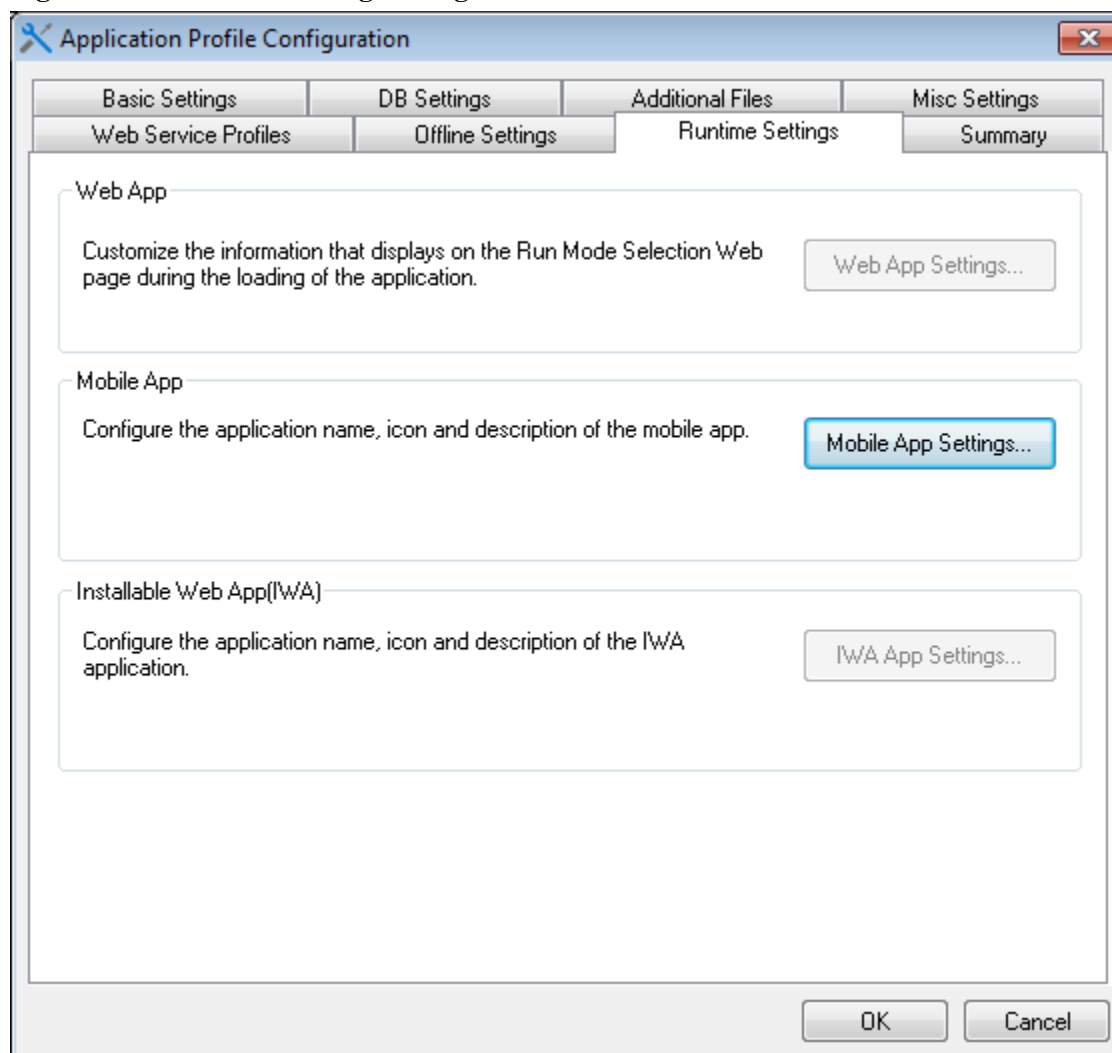
Items	Descriptions
Type	<p>Select Data Source or Transaction.</p> <p>To use the transaction object to connect with the local database for the application, select Transaction; to use the CacheName parameter of DBParm in the transaction object to connect with the database, select Data Source. By using the CacheName DBParm, you can conveniently switch database connections in the same transaction object. For more about the preferable scenarios using the PB transaction object or using the CacheName DBParm, refer to Section 4.1, “Connect with the local database” in <i>PowerServer Mobile Offline Tutorials</i>.</p> <p>For more about the syntax of CacheName DBParm, see Section 4.3.1.21, “Transaction” in <i>Supported PB Features for PowerServer Mobile</i>.</p>
Name	<p>Enter a data source or transaction name.</p> <p>If the Type is Data Source, the Name must be the same value of the CacheName DBParm; if the Type is Transaction, the Name must be the exact name of the transaction object in the application source code.</p>
Database Type	<p>Select SQLite or UltraLite 17.</p>

Items	Descriptions
Database File	<p>Specify the database file.</p> <p>The database file must exist and be valid, and the name cannot be "appeondb.db". The database file will be deployed to PowerServer when the application is deployed, and will be downloaded to the local device when the application is downloaded for the first time.</p> <p>The database file on the local device will not be updated unless you write code in the application to call of <code>_applydbupdate</code> in <i>Workarounds & API Guide</i> to get the latest database file from the PowerServer.</p>
User name	<p>Enter the user name for the database.</p> <p>This field is only required for UltraLite.</p>
Password	<p>Enter the password for the database.</p> <p>This field is only required for UltraLite.</p>
Encryption Key	<p>Enter the encryption key for the database.</p> <p>This field only supports UltraLite. You should input the same encryption key which is specified when creating the UltraLite database in the Create Database Wizard in Sybase Central, so that the Appeon mobile application can read/write the encrypted database.</p> <p>For more about encrypting the UltraLite database in Sybase Central, see Section 4.4, "See also: Encrypt UltraLite database" in <i>PowerServer Mobile Offline Tutorials</i>.</p>

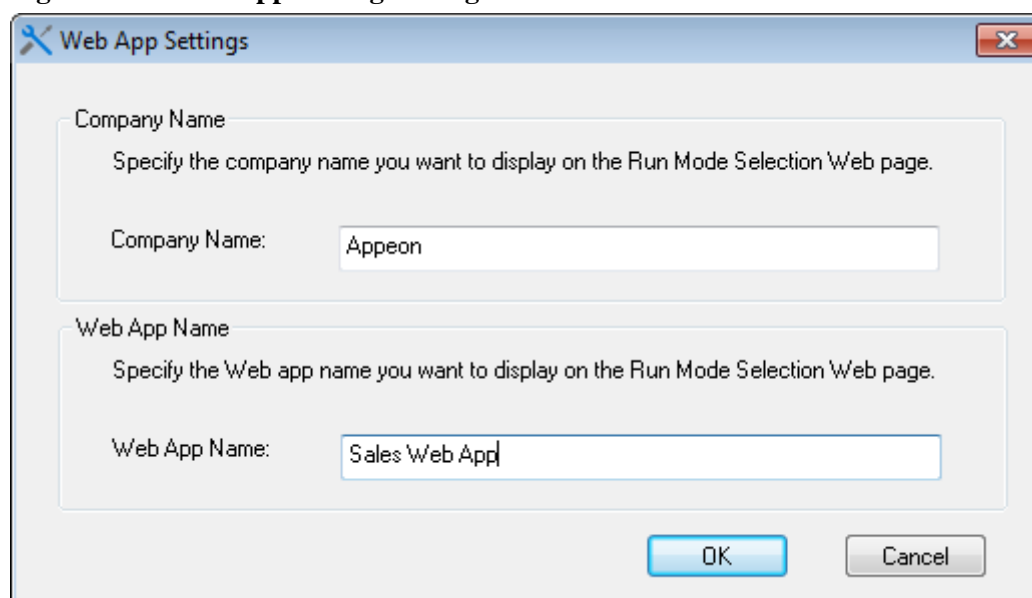
- To edit a local database, select the local database in the **Local DB Settings** group box, and then click **Edit**. In the **Local Database Configuration** window that appears, make the changes according to your needs, and then click **OK**.
- To delete a local database, select the local database in the **Local DB Settings** group box, and then click **Delete**. Click **Yes** to confirm the deletion in the popup message box.

4.2.1.2.7 Runtime Settings

The Runtime Settings tab specifies the runtime settings for the Web application, mobile application, and IWA application.

Figure 4.28: Runtime Settings dialog box

For the Web application, you can set the company name and the application name that will be displayed on the run mode selection Web page when the application starts.

Figure 4.29: Web App Settings dialog box

For the mobile application, you can set:

1. the mobile device type (tablet or smartphone) on which the mobile application will run.
2. the mobile application name, icon, and app description that will be displayed in Apeon Workspace.

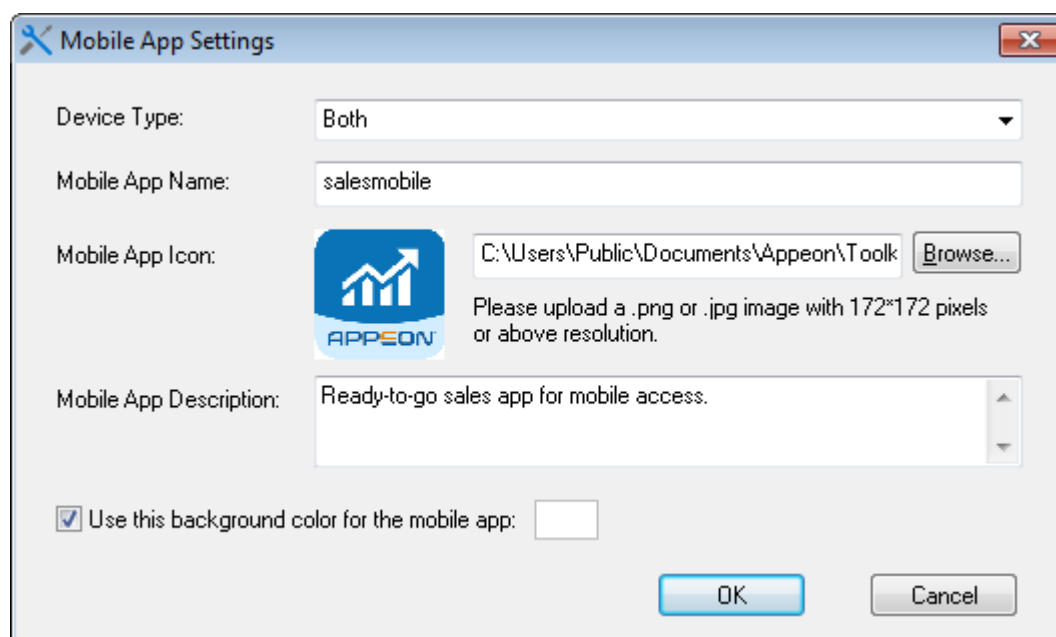
For optimal display quality, we recommend using icons with 86 x 86 pixels for low-resolution screens (such as iPad 2), and 172 x 172 pixels for high-resolution screens (such as iPad 4). But generally you can use icons with any size, because icons smaller than the above recommended size will be automatically stretched to the proper size for the corresponding devices; and icons larger than the above recommended size will be automatically shrunk to the proper size for the corresponding devices. Therefore, for convenience, you can use icons with 172 x 172 pixels or above for all devices.

The recommended formats are PNG and JPG. For other supported formats, refer to Section 2.10, “Image & Icon” in *Mobile UI Design & Development Guide*.

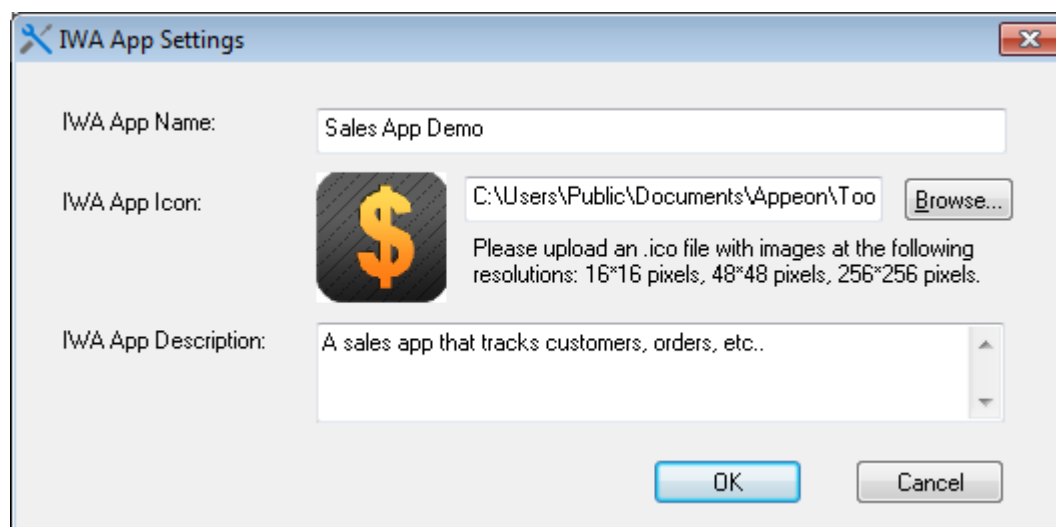
If you leave the **Mobile App Icon** field empty, the default icon will be displayed in Apeon Workspace.

3. the background color of the mobile application. If this option is not set, the default image will be used as the background for the mobile application.

Figure 4.30: Mobile App Settings dialog box



For the IWA app, you can set the application name, icon, and app description. The app name and the icon will be used in the desktop shortcut for the IWA app. If you leave the **IWA App Name** field empty, the **Web Folder** value in the **Basic Settings** tab will be used as the default app name. If you leave the **IWA App Icon** field empty, the default icon will be used.

Figure 4.31: IWA App Settings dialog box

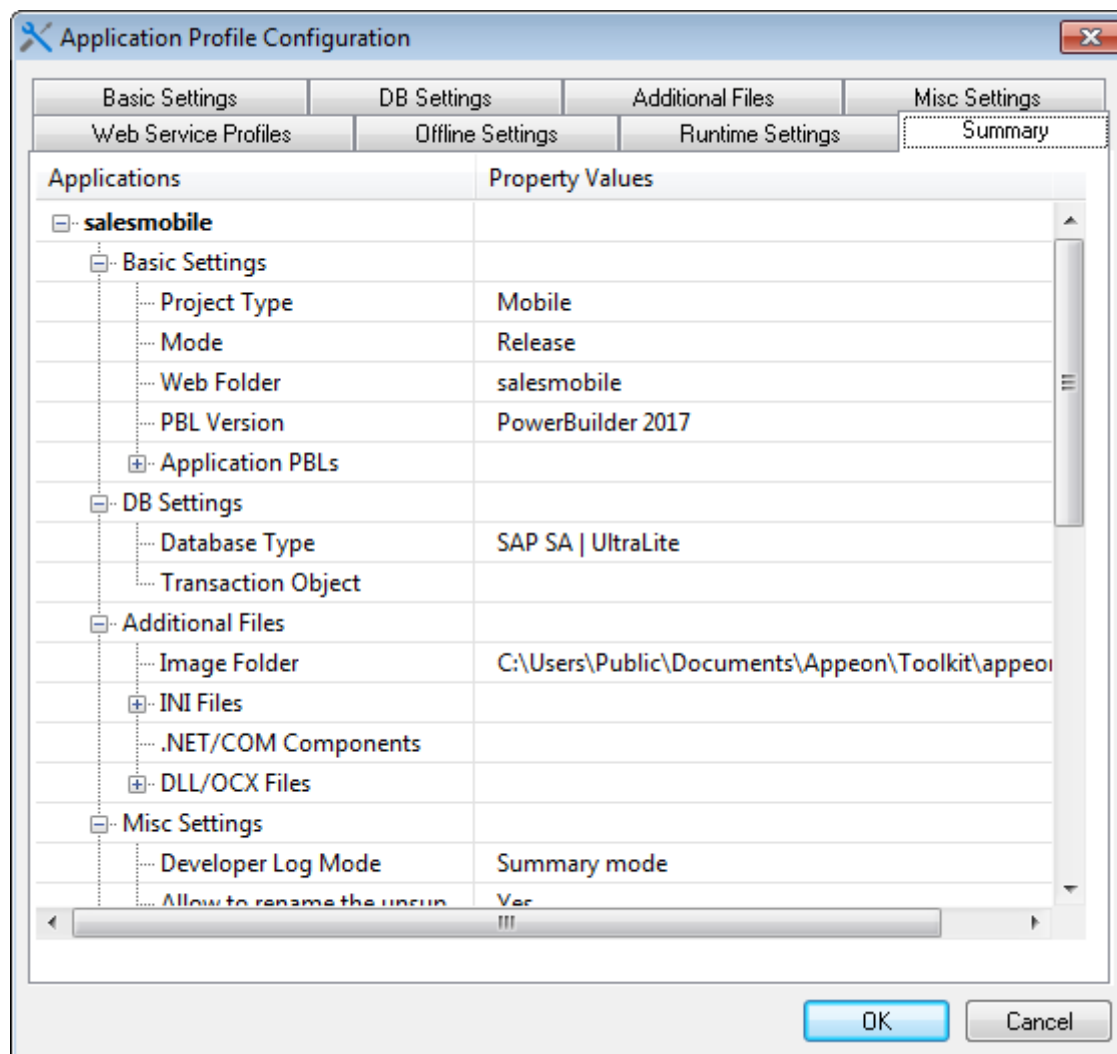
4.2.1.2.8 Summary

The Summary tab offers a summary view of all configurations of an application profile. It lists the configurations in a treeview structure, and allows you to review and modify their values directly.

Use the following two methods to input or modify the configuration (property values):

- For properties that you can input or select values by using a text field, radio buttons, or dropdown list, such as Mode, Web Folder, PBL version, etc., you can directly double click their Property Values column and then input, modify or select values.
- For properties that you can add/delete multiple items by using a dialog box, such as application PBLs, INI files, etc., you can right click the property name in the Application column and then select **Add** or **Delete All** from the popup menu.

The configuration changes you make in this Summary tab page will be immediately synchronized to the other tab pages, but only after you click the **OK** button, the changes will be saved.

Figure 4.32: Summary tab

4.2.2 Managing database type profiles

Database type profiles configuration is the easiest way to manage the database types that Apeon Deployment Wizard supports for application deployments. You can configure database type profiles in the **DB Type Profiles** tab, and apply a database type profile for an application deployment by selecting the profile in the PowerServer Toolkit Configuration Wizard or in the [DB Settings](#) tab of the application profile settings.

Database type profiles in PowerServer Toolkit are different from database profiles in PowerBuilder:

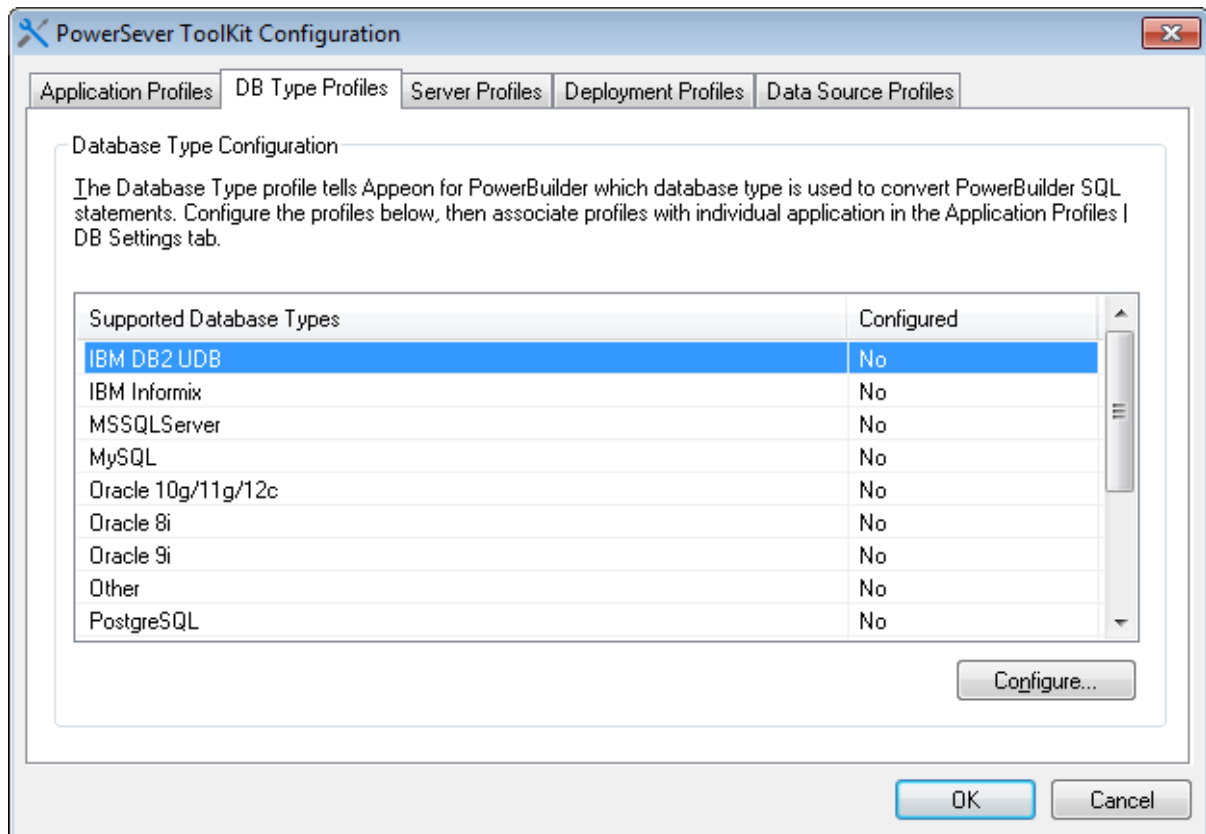
- You only need to configure one database type profile for a given database type. For example, if application A connects to SAP SQL Anywhere database d1 and application B connects to SAP SQL Anywhere database d2, you can use the database type profile *SAP SA | UltraLite* for both applications, although the connection information in the profile *SAP SA | UltraLite* sets up connection to SAP SQL Anywhere database d1.
- Database type profiles are not used for setting up connections with application databases. Instead, they are mainly used by Apeon Deployment Wizard to call the relevant database driver for converting PowerBuilder SQL statements.

To modify a database type profile, take the following steps:

Step 1: Click the **Configure** button on the PowerServer Toolkit, and select the **DB Type Profiles** tab on the Application Profile Configuration page as shown in the following figure.

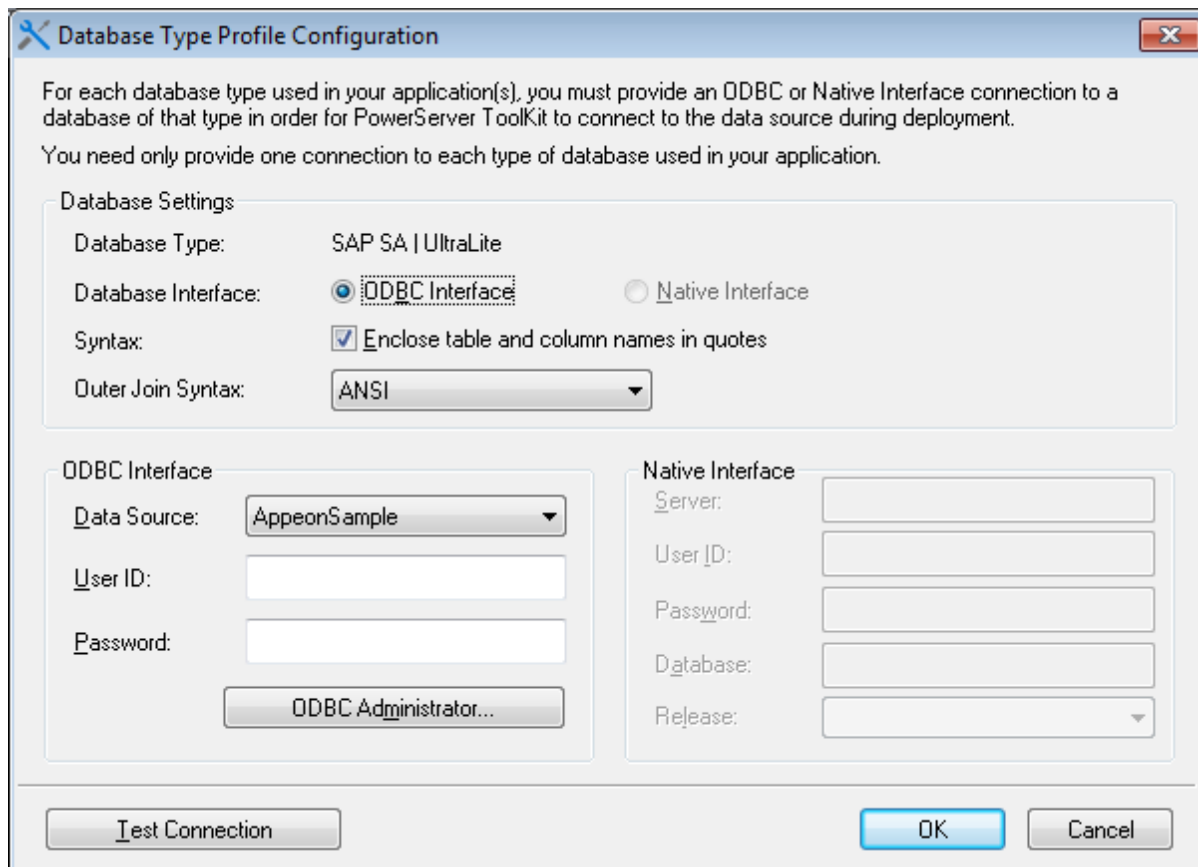
The Database Type column lists all the database types that are supported by Apeon; and the Configured column indicates whether a database type profile has been configured.

Figure 4.33: DB Type Profiles



Step 2: Select a database type profile (for example, SAP SA | UltraLite), and click the **Configure** button to configure the database type profile, as shown in the following figure.

Figure 4.34: Database Type configuration



Step 3: Specify the information needed for setting up the database type profile.

Outer Join Syntax: Specify how Appeon format the SQL syntax for outer joins, you can select corresponding format from the drop-down list in the Database Settings frame. Regardless of the settings in PowerBuilder, it is recommended to use ANSI as the outer join syntax format in Appeon.

ODBC Interface: Select this option if you use an ODBC interface. Specify the ODBC data source, User ID, and Password.

Table 4.9: ODBC interface settings

Setting	Instructions
Data Source	All the data sources configured in Windows ODBC Administrator are listed. Select a proper data source of the configured database type from the dropdown list. It is not necessary to specify the actual databases that the application uses. Click ODBC Administrator to configure an ODBC data source if it is not available.
User ID	Type in a recognized (set) user name to login to the database. If no user name has been set, leave this field blank.
Password	Type in the password used to log in to the database. If no password has been set, leave this field blank.

Native Interface: Select this option if you use a native database interface. Specify the server name/IP, User ID, Password, and the database name.

Note: Not all of the listed database types are supported by Apeon for the native database interface. If SAP IQ, SAP SA | UltraLite or IBM DB2 is your database type, the Native Interface option is disabled.

Table 4.10: Native database interface settings

Setting	Instructions
Server	Enter the machine name or IP address of the server where the target database resides.
User ID	Type in a recognized (set) user name to login to the database. If no user name has been set, leave this field blank.
Password	Type in the password used to login to the database. If no password has been set, leave this field blank.
Database	Specify the name of the database that will be used by the application.
Release	(ASE only) Specify the version number of ASE database.

Step 4: Specify the "**Enclose table and column names in double quotes**" option. It determines whether PowerServer Toolkit encloses the names of tables, columns, indexes, and constraints in double quotes when it generates the database syntax during deployment.

Notes:

- 1) When this option is selected, verify that the `quoted_identifier` setting is set to **ON** in the database server.
- 2) Clear the check box of this option when the updated table name of DataWindow contains the owner name.

Step 5: Click **Test Connection** to test the connection to the database.

4.2.3 Managing server profiles

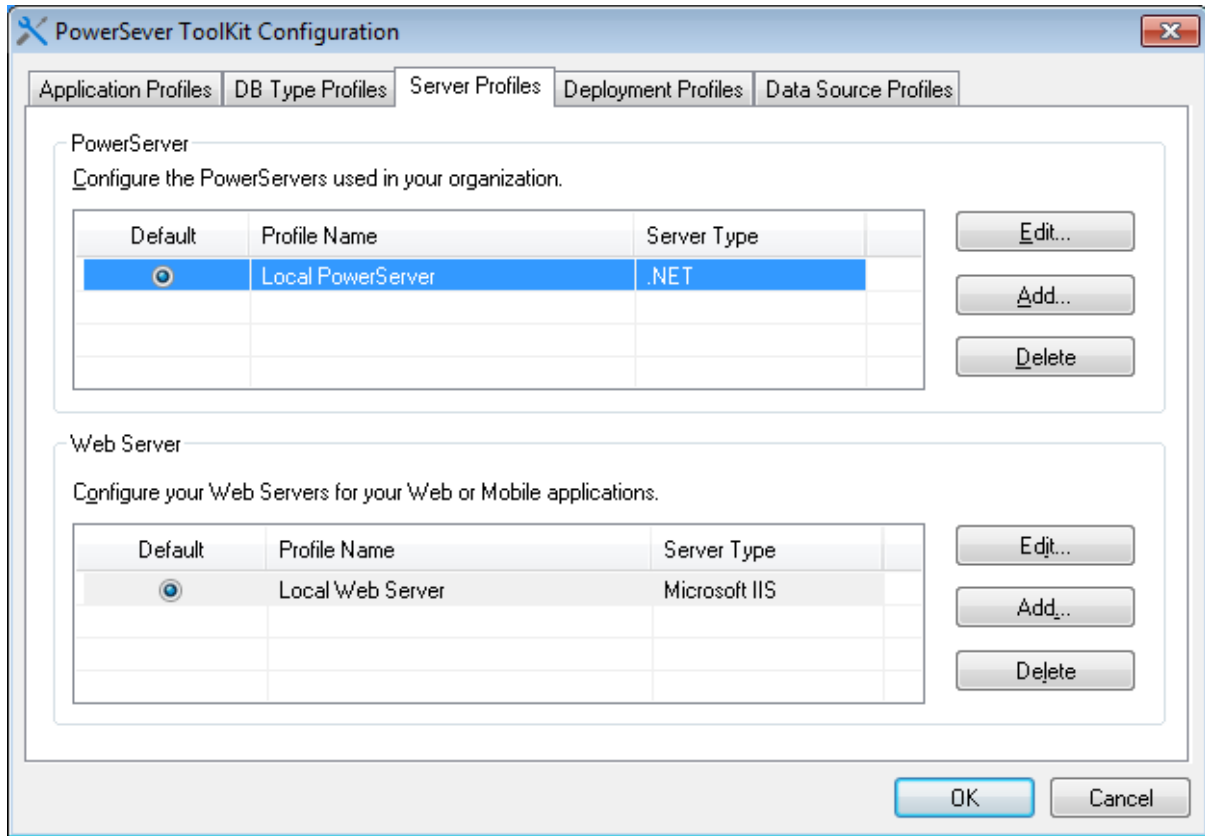
A server profile is a set of parameters for PowerServer Toolkit to connect to, and deploy applications to, a particular PowerServer or Web server. Creating and using server profiles is the easiest way to manage your application deployments because you can:

- Select one or more PowerServer profiles and Web Server profiles to set up a deployment profile.
- Easily add, edit, and remove server profiles.
- Test connections in server profiles. If the connection is successful, it means that the application can be deployed to the server configured in the server profile.
- Manage which applications are accessible from the PowerServer Toolkit shortcut **Run** button. The shortcut lists the applications that are deployed to the default server profiles.

4.2.3.1 Server Profiles tab page

There are two main parts on the **Server Profiles** tab page, as shown in the following figure. The upper part is for management of PowerServer profiles, while the lower part is for management of Web server profiles.

Figure 4.35: Server Profiles



4.2.3.1.1 Default server profiles

You can select a PowerServer profile or a Web Server profile as the DEFAULT server profile on the Server Profiles tab page. PowerServer Toolkit's **Run** shortcut reads the default server profiles and lists the applications that are deployed to default server profiles, enabling you to quickly access them. Applications that are deployed to non-default server profiles are not available in the shortcut. For details about the shortcut, refer to [Chapter 8, Running Apeon Applications](#).

4.2.3.1.2 Edit, add or delete server profiles

Two sets of **Edit**, **Add**, and **Delete** buttons are available on the Server Profiles tab page for you to modify, create, or remove PowerServer or Web Server profiles. [Section 4.2.3.2, “PowerServer profile settings”](#) and [Section 4.2.3.3, “Web Server profile settings”](#) provide instructions on how to configure the properties for a PowerServer or Web Server profile when you modify or create a new server profile.

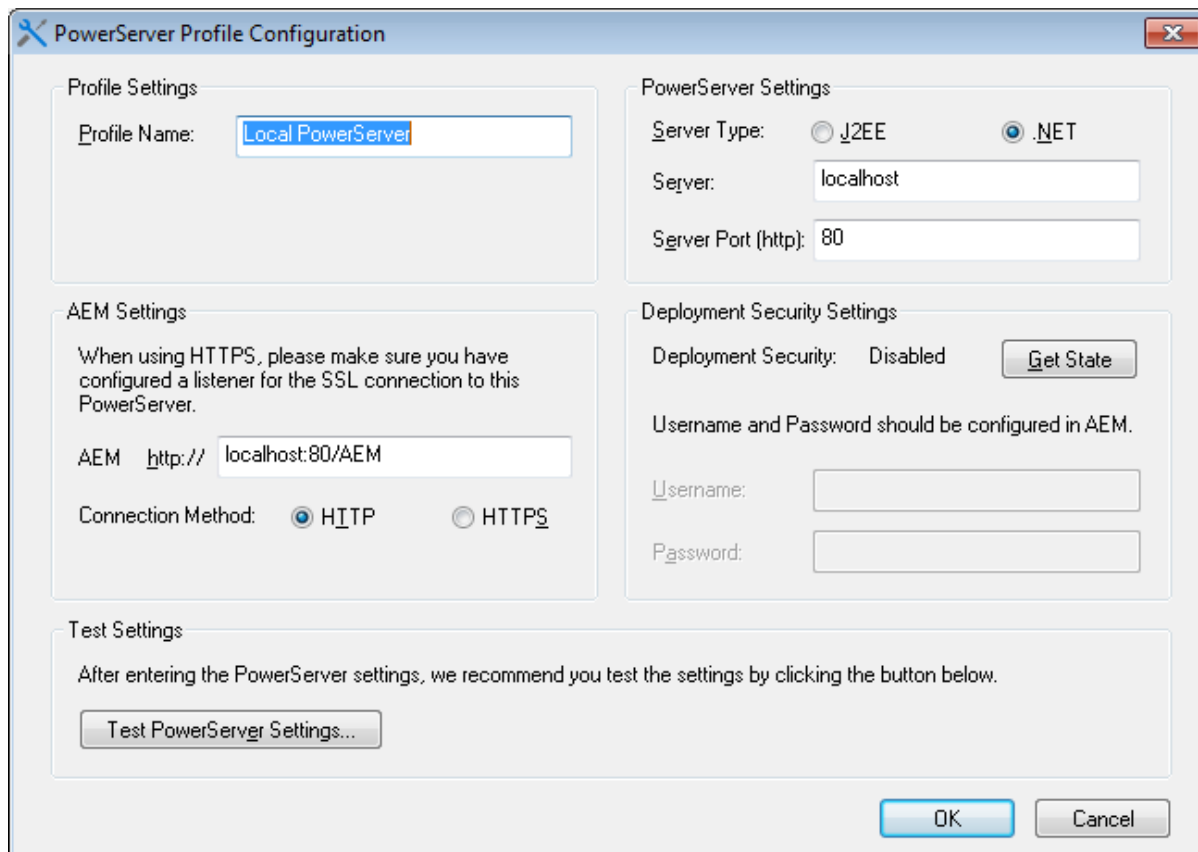
4.2.3.2 PowerServer profile settings

Before you edit or add a PowerServer profile, make sure that the PowerServer specified in the profile is running, and that the PowerServer Toolkit computer can successfully connect to the PowerServer computer.

If you want to deploy the application to a PowerServer cluster, you must create profiles for each participating PowerServer and deploy the application to all PowerServer in the cluster.

The following figure shows the PowerServer Profile Configuration window that displays when you click the **Edit** or **Add** button in the PowerServer group box of the **Server Profiles** tab page.

Figure 4.36: PowerServer Profile



The following table lists detailed instructions on how to specify the properties for a PowerServer profile.

Table 4.11: PowerServer profile properties

	Property	Instructions
Profile Settings	Profile Name	Assign a name to the PowerServer profile. You should use names that are easy to remember and identify (for example "PowerServer for Test" or "Remote PowerServer").
PowerServer Settings	Server Type	Select J2EE if PowerServer is installed to JBoss, JEUS, WebLogic, NetWeaver, WebSphere, or EAServer; select .NET if PowerServer is installed to Microsoft .NET Framework.
	Server	Enter the IP address or the machine name of the PowerServer.
	Server Port (http)	Enter the HTTP port number used by PowerServer. Note: Do not enter an HTTPS port here, PowerServer Toolkit must use an HTTP port to connect with

	Property	Instructions
		PowerServer, though the Apeon deployed application can be connected with an HTTPS port.
AEM Settings	AEM URL	The URL for AEM will be automatically generated after you specify the Server and Server Port. The URL will be in the following format: http://server:port/AEM, (for example, http://localhost:8080/AEM). Note: Do not use a "localhost" listener in a production environment.
	Connection method	Select HTTP or HTTPS to connect to AEM. If the Web server is configured as an SSL Web Server, check the HTTPS (secure) option. Otherwise, check the HTTP (insecure) option.
Deployment Security Settings	Deployment Security	The Get State button can get the state of the Deployment Security settings in AEM. If Deployment Security is set to ON, the state will be "Enabled" and you must specify the deployment user name and password. By default, the Deployment Security in AEM is OFF, so the state is "Disabled".
	Username	Enter the username used to deploy the application. The username and password for the deployment security feature are configured in AEM. The username can be left blank if the deployment security feature is turned off in AEM.
	Password	Enter the password used to deploy the application. The username and password for the deployment security feature are configured in AEM. The password can be left blank if the deployment security feature is turned off in AEM.

After profile configuration, perform the following steps to make sure the PowerServer profile can be successfully used for application deployments:

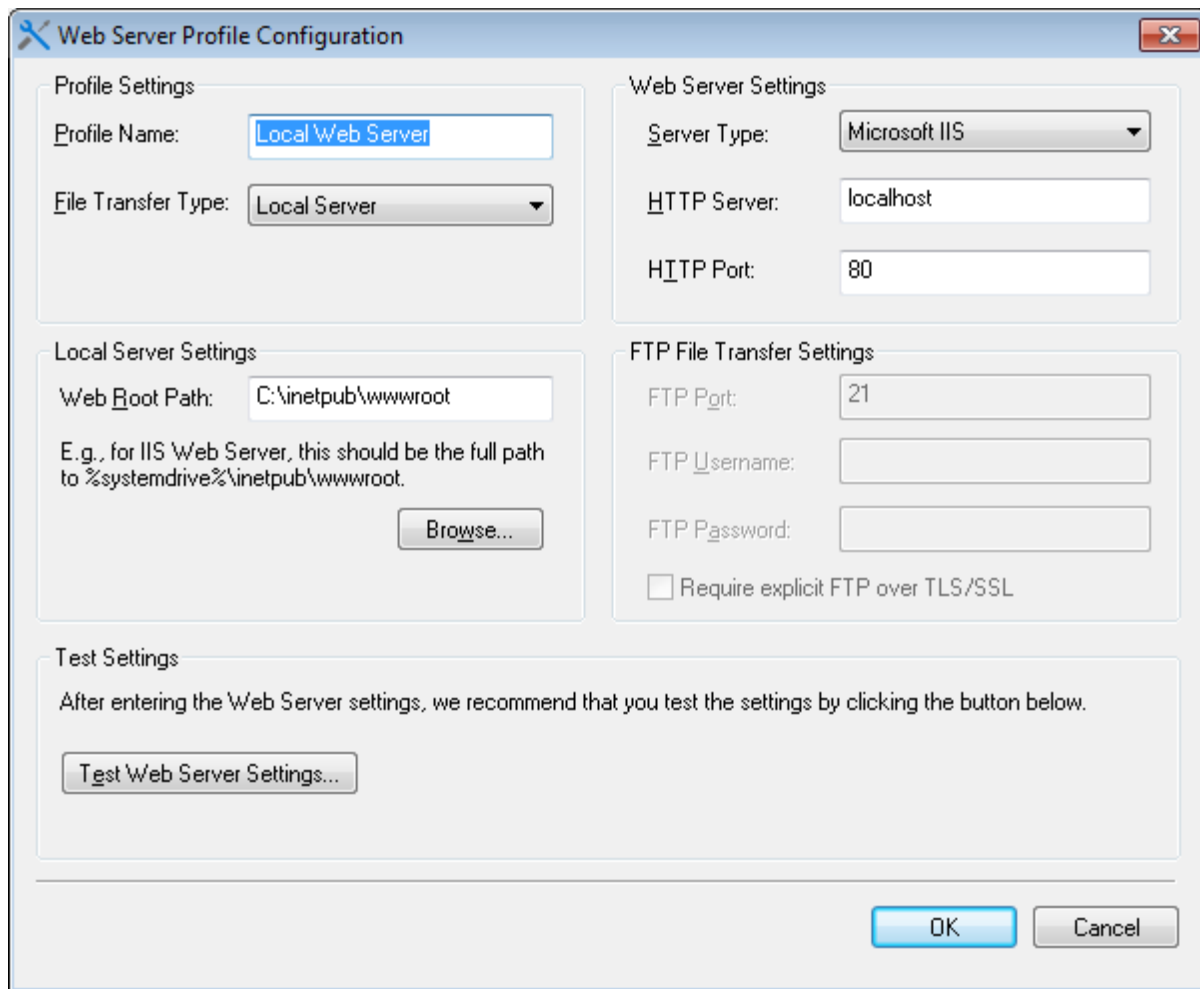
- Check whether the deployment security settings are configured correctly. The username and password in the deployment security settings must be the same as those configured in AEM. Make sure you get the correct deployment username and password from the AEM administrator.
- Test PowerServer settings by clicking the **Test PowerServer Settings** button. Do NOT proceed to the next step until the testing succeeds.

4.2.3.3 Web Server profile settings

Before you edit or add a Web Server profile, make sure the Web Server specified in the profile is running and that the PowerServer Toolkit computer can successfully connect to the Web Server computer.

The following figure shows the Web Server Profile Configuration window that displays when you click the **Edit** or **Add** button in the **Web Server** group box of the **Server Profiles** tab page.

Figure 4.37: Web Server profile configuration window



The following table lists detailed instructions for how to specify the properties for a Web Server profile.

Table 4.12: Instructions for creating a Web Server profile

	Property	Instructions
Profile Settings	Profile Name	Assign a name to the Web Server profile. You should use names that are easy to remember and identify such as "Web Server for Test" or "Production Web Server".
	File Transfer Type	Select Local Server if the Web Server is on the local machine.

	Property	Instructions
		Select Remote Server if the Web Server is on a machine different from the PowerServer Toolkit machine.
Web Server Settings	Server Type	Select the Web server type. Appeon supports the following Web Server types: Apache, IIS, JBoss, WebLogic, WebSphere, NetWeaver, JEUS, and EAServer.
	HTTP Server	Enter the IP address or the machine name of Web Server.
	HTTP Port	Enter the Web Server port number. Note: Do not enter an HTTPS port here, even though the Web server is configured as an SSL Web server, because PowerServer Toolkit must use an HTTP port to connect with Web server. However, you can access the deployed application with the HTTPS port.
Local Server Settings	Web Root Path	Enter the home directory of the Web server where the application WAR file will be deployed to. For example: For Apache, the home directory is <apache root>\htdocs For IIS, the home directory is <systemdrive>:\inetpub\wwwroot; For WildFly and JBoss EAP, the home directory is <jboss root>\standalone\deployments; For WebLogic 8, the home directory is <systemdrive>:\bea\user_projects\domains\appeon_domain\applications; For WebLogic 9/10, the home directory is <systemdrive>:\bea\user_projects\domains\appeon_domain\autodeploy; For WebSphere, the home directory is <systemdrive>:\IBM\WebSphere6\AppServer\installableApps; For JEUS, the home directory is 1. <jeus root>\webhome\app_home\autodeploy (for automatic deployment); 2. <jeus root>\webhome\app_home (for manual deployment). For NetWeaver Application Servers, specify any path and then deploy the WAR file in the NetWeaver's administrative console. For EAServer, the home directory is <systemdrive>:\Program Files\Sybase\EAServer6\html;

	Property	Instructions
		<p>Note: For WebLogic (started in production mode) and WebSphere, you will need to manually deploy the WAR file in the server administrative control. This is required by the WebLogic and WebSphere, not by Apeon. It is similar to deploying the PowerServer EAR package; you can follow the instructions in the section called "Deploying apeonserver.ear package" in the Installation Guide.</p>
FTP File Transfer Settings	FTP Port	<p>Enter the FTP server port number. The typical FTP port is port 21.</p> <p>If you use a remote Web Server, PowerServer Toolkit will upload the files to Web Server using FTP protocol. Note that PowerServer Toolkit works as a passive FTP client when uploading files. So you need to make sure that you are using passive FTP and the FTP data port and command port are configured properly in the firewall.</p>
	FTP Username	<p>Enter the username for FTP login.</p> <p>If the FTP server offers anonymous access then the username should be <i>anonymous</i>.</p>
	FTP Password	<p>Enter the password for FTP login.</p> <p>If no password is set for the FTP server or the FTP Username is anonymous, leave this field blank.</p>
	Require explicit FTP over TLS/SSL	<p>Select this checkbox if you want PowerServer Toolkit to upload the files via FTPS protocol.</p> <p>Make sure the FTP server is configured to "Require SSL Connections" (not "Allow SSL Connections") for the control channel and the data channel. Also, a server SSL certificate should be specified, although PowerServer Toolkit will not verify whether the certificate is signed by a trusted certificate authority, or verify the certificate's name against the host.</p> <p>The supported TLS/SSL protocols include TLSv1.2, TLSv1.1, TLSv1.0, SSLv3, and SSLv2.</p> <p>And the FTP server should not be configured to require the client-side SSL certificate, otherwise PowerServer Toolkit will fail to connect with the server, because PowerServer Toolkit does not support validating the client-side SSL certificate.</p>

After the profile configuration, perform the following steps to make sure the Web Server profile can be successfully used for application deployments:

- Test Web Server settings by clicking the **Test Web Server Settings** button. Do NOT proceed to the next step until the testing succeeds.

If the Web Server is a remote server, refer to [Section 4.2.3.3.1, “Two requirements for FTP settings”](#) to make sure the configuration for the Web Server profile is successful.

If the Web Server is an SSL Web server, refer to [Section 4.2.3.3.2, “If the Web Server is an SSL Web Server”](#) to make sure the configuration for the Web Server profile is successful.

4.2.3.3.1 Two requirements for FTP settings

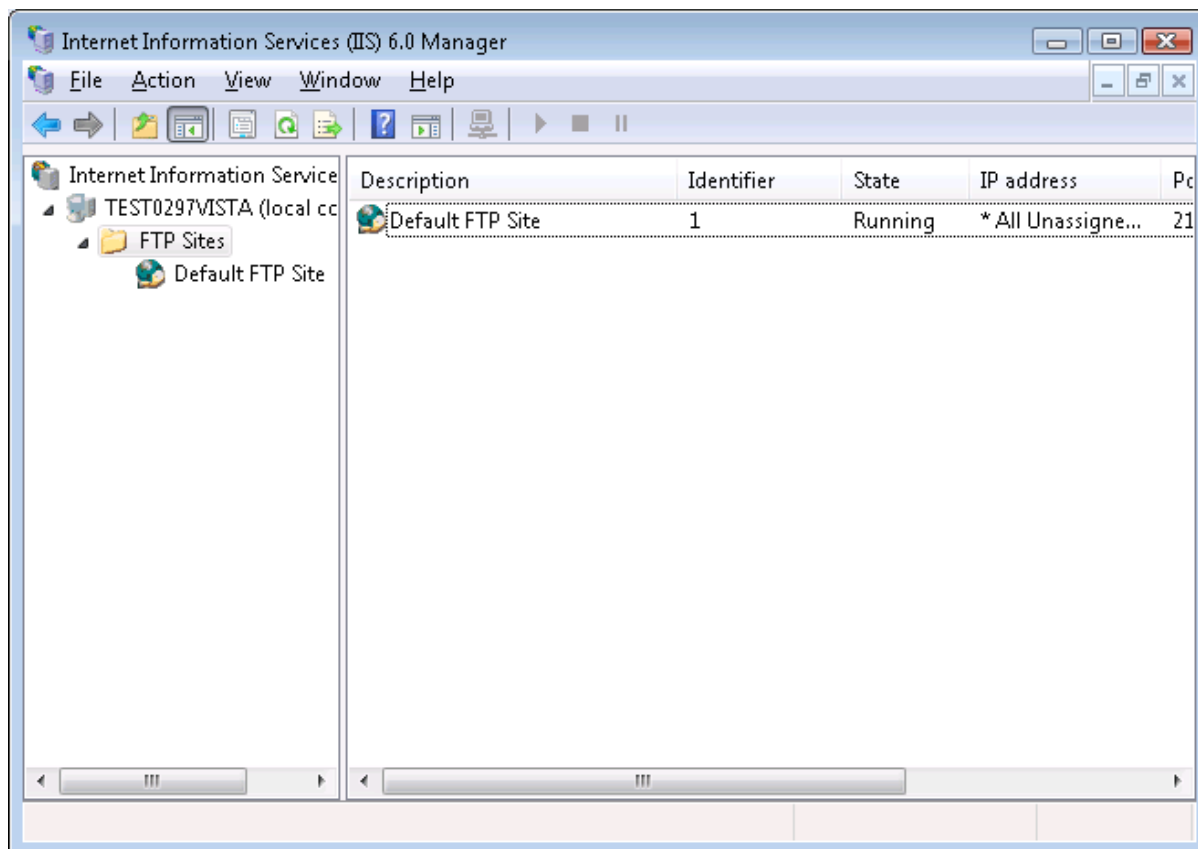
If you configure a Web Server profile for a remote Web Server, make sure the FTP settings in the Web Server profile meet the following two requirements:

- The user name and password for accessing the FTP server should have permission to read and write files to the FTP server of the Web Server.
- The FTP home directory should be mapped to the Web root of the Web Server.

The following steps use the Microsoft IIS FTP service as an example to show you how to fulfill the requirements. You should find that the settings for other FTP types are similar to the settings for Microsoft IIS FTP.

Step 1: On the FTP server (Web Server), open the Internet Services Manager in Administrative Tools, as shown in the following figure.

Figure 4.38: FTP configuration

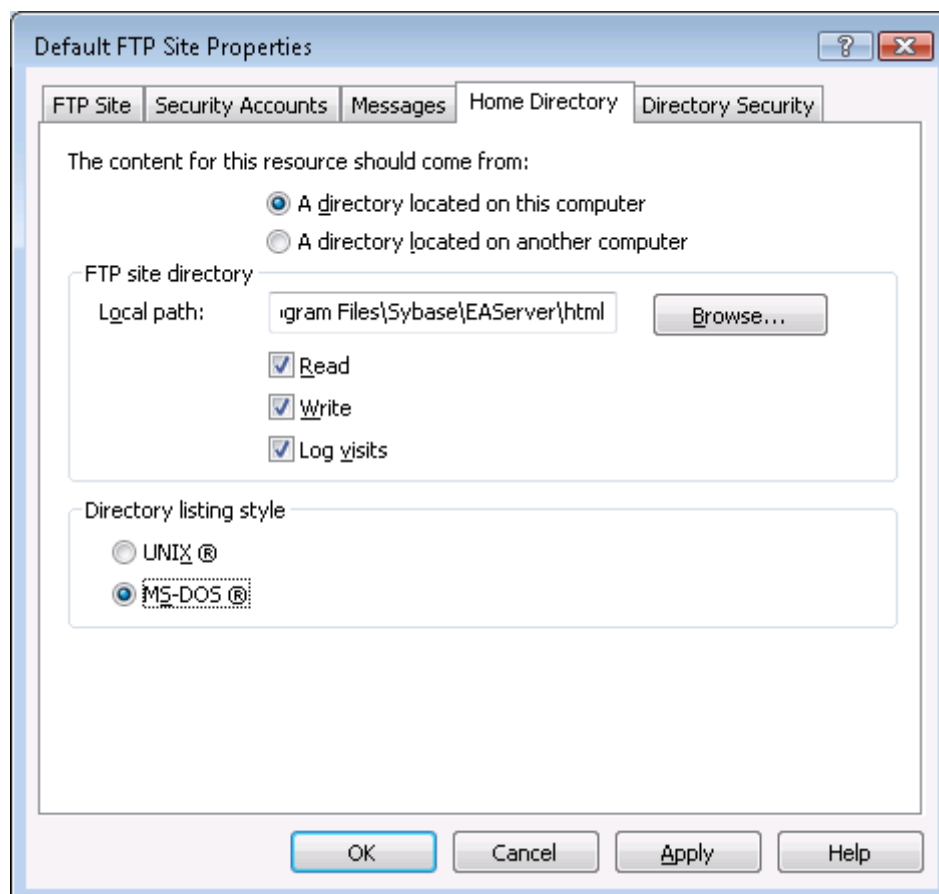


Step 2: Right click on **Default FTP Site** and select **Properties** in the popup menu. The Default FTP Site Properties window is displayed.

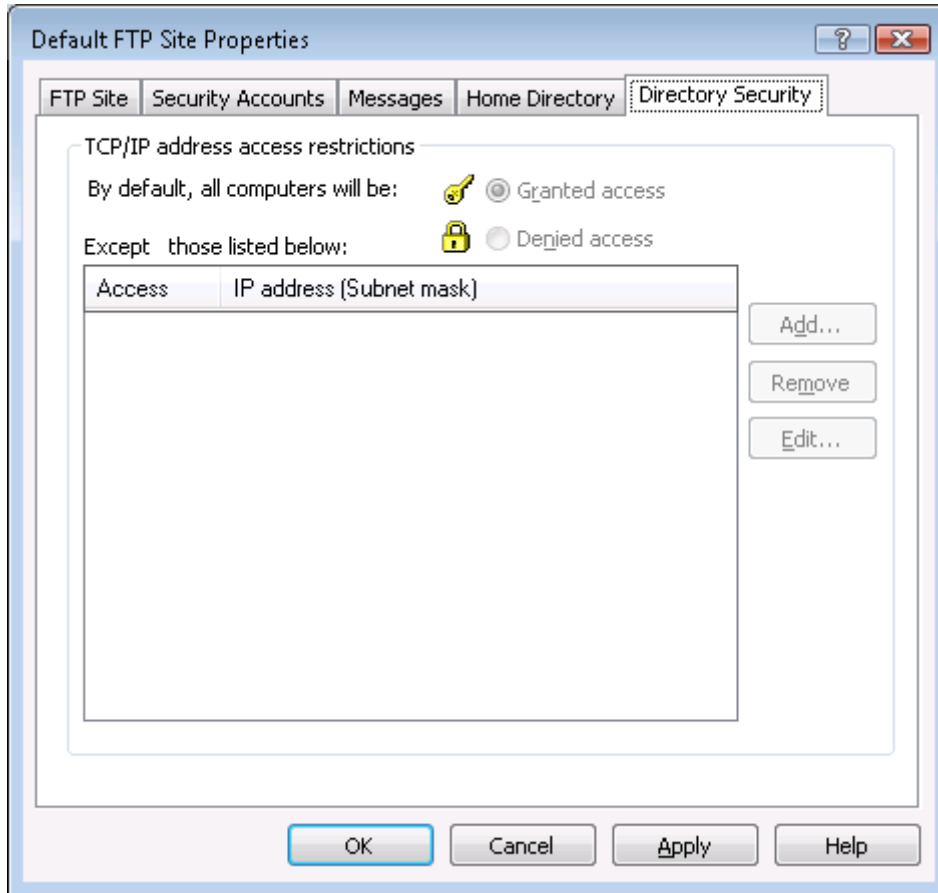
Step 3: Go to the Home Directory tab and verify that:

- The Local Path is the full path to the Web Server document root.
- The Write property of the FTP Site Directory is enabled.

Figure 4.39: FTP site properties



Step 4: Go to the Directory Security tab and verify that the Granted Access option is checked, as shown in the following figure.

Figure 4.40: Directory security

4.2.3.3.2 If the Web Server is an SSL Web Server

If the Web Server is an SSL or secure Web Server, you can configure the Web Server profile following the configuration instructions for non-SSL Web Servers, except for the following two points:

1. You need to configure an HTTPS listener and port number for the Web Server.
2. You must specify an HTTP (not HTTPS) listener and its port number in the HTTP server settings of the Web Server profile, because PowerServer Toolkit must use the HTTP protocol when deploying the Apppeon application files to the server. After the application is deployed to the Web server, you can input "https" instead of "http" and the HTTPS port in the URL to access the application, (for example, <https://192.0.0.80:8181/appendemo/>).

4.2.4 Managing deployment profiles

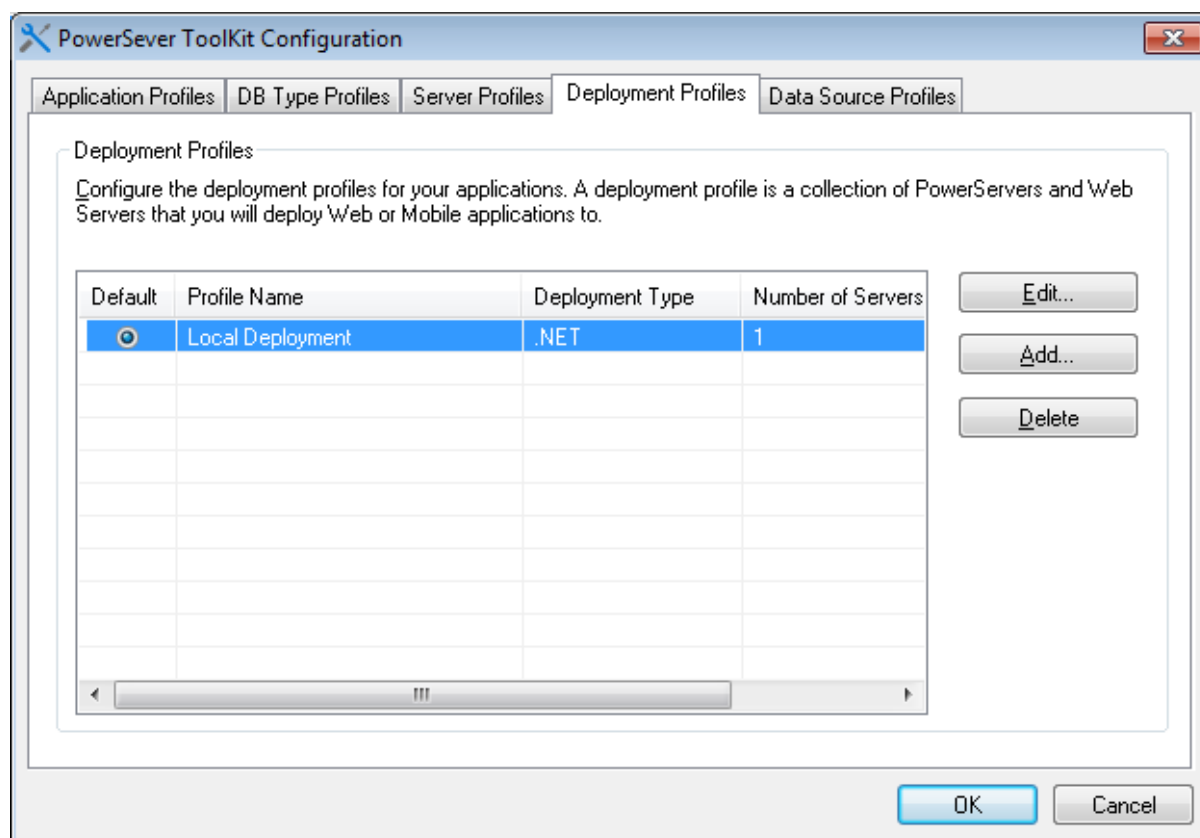
A deployment profile associates specified Web Server(s) and PowerServer(s) as a group used for Web or mobile deployment. You may create several deployment profiles (e.g. local deployment, test deployment, production deployment, and so on), and set the most commonly used profile as the default profile.

A deployment profile is based on server profiles. Before configuring the deployment profiles, make sure you have set up one server profile for each of the PowerServer and Web Servers to be used for the Web or mobile conversion. You can add as many deployment profiles as you need, but there can be only one default deployment profile.

4.2.4.1 Deployment Profiles tab page

On the Deployment Profile tab, you can edit, add, or delete deployment profiles and specify a default deployment profile.

Figure 4.41: Deployment Profile tab page



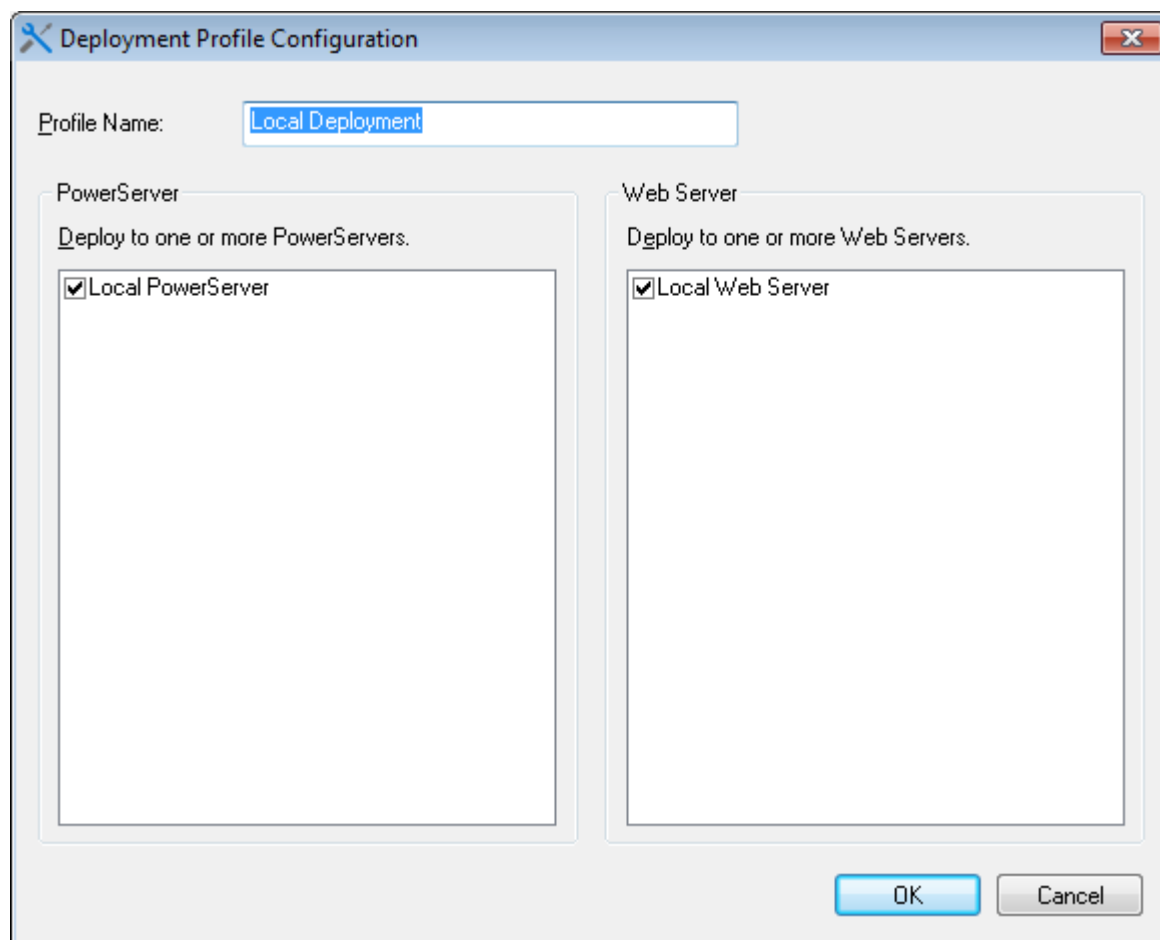
The following table describes the tasks you can perform on the Deployment Profile tab.

Table 4.13: Instructions to use the Deployment Profile tab

Use this button	To
Default radio button	Specify the default deployment profile used for Web or mobile deployment.
Edit button	Modify an existing deployment profile. This will open the Deployment Profile Configuration dialog box.
Add button	Create a new deployment profile. This will open the Deployment Profile Configuration dialog box.
Delete button	Remove a deployment profile.

4.2.4.2 Deployment profile settings

In the Deployment Profile Configuration dialog box, set up the necessary configuration, as shown in the following figure.

Figure 4.42: Deployment Profile

The following table describes the settings on the Deployment Profile Configuration dialog box.

Table 4.14: Deployment Profile settings

In this field	You can
Profile Name	Type the deployment profile name. You can use names that are easy to remember and identify such as Test Deployment , Remote Deployment .
PowerServer	Select a check box to include the PowerServer in the deployment profile. All the PowerServer profiles you have created are listed. More than one PowerServer can be selected.
Web Server	Select a check box to include the Web Server in the deployment profile. All the Web Server profiles are listed. More than one Web Server can be selected. Note: If the server type of PowerServer profile is .NET, then the server type of Web Server profile must be Microsoft IIS.

Click **OK** to save the new settings and return to the Deployment Profiles tab.

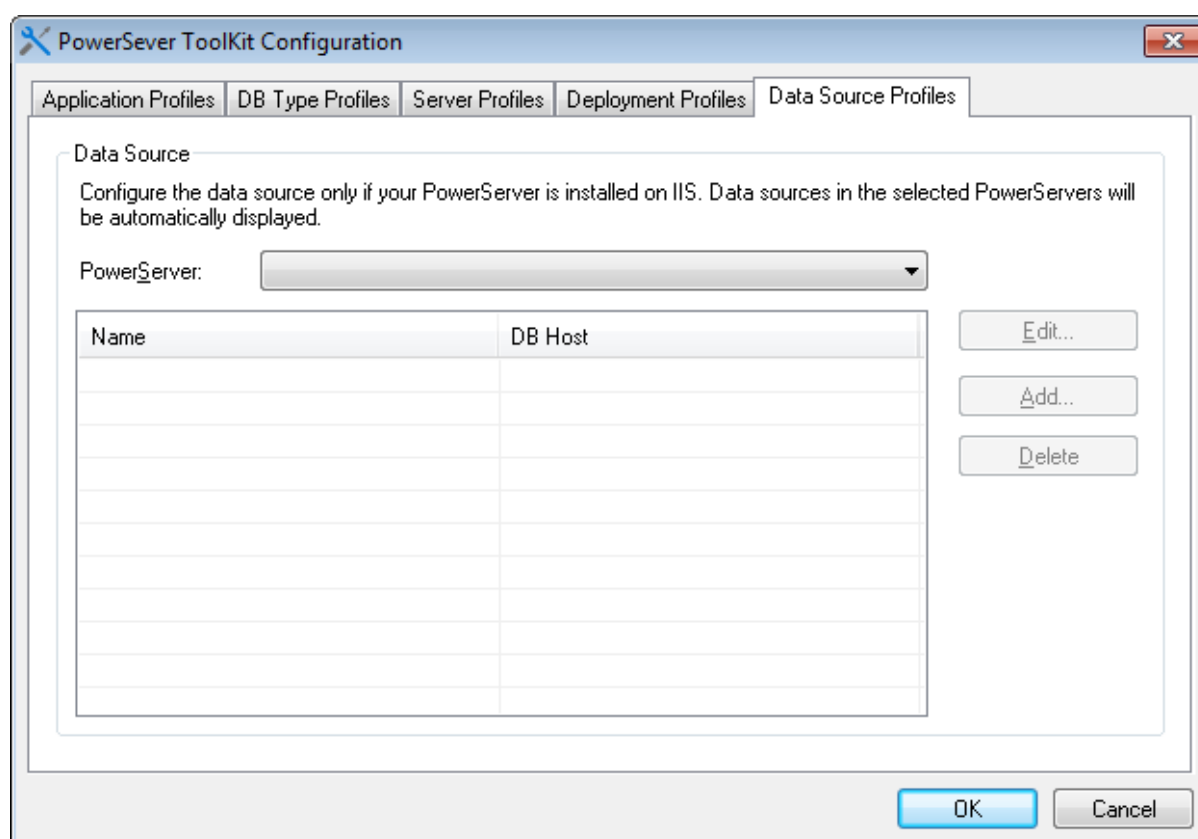
4.2.5 Managing data source profiles

Note: This Data Source Profiles tool works for Microsoft .NET Framework\IIS, but does not work for JBoss, JEUS, WebLogic, NetWeaver, WebSphere, and EA Server 6.x.

To create data source for JBoss, JEUS, WebLogic, NetWeaver, WebSphere, and EA Server 6.x, refer to Chapter 4, *Database Connection Setup in PowerServer Configuration Guide for J2EE*.

Appeon applications use JDBC data sources created in PowerServer to connect to the database. The Data Source Profile tab allows you to maintain a storage of data sources which you can specify to use for Appeon applications. Its function is the same as the one provided in AEM, and changes made on the tab or AEM are automatically synchronized.

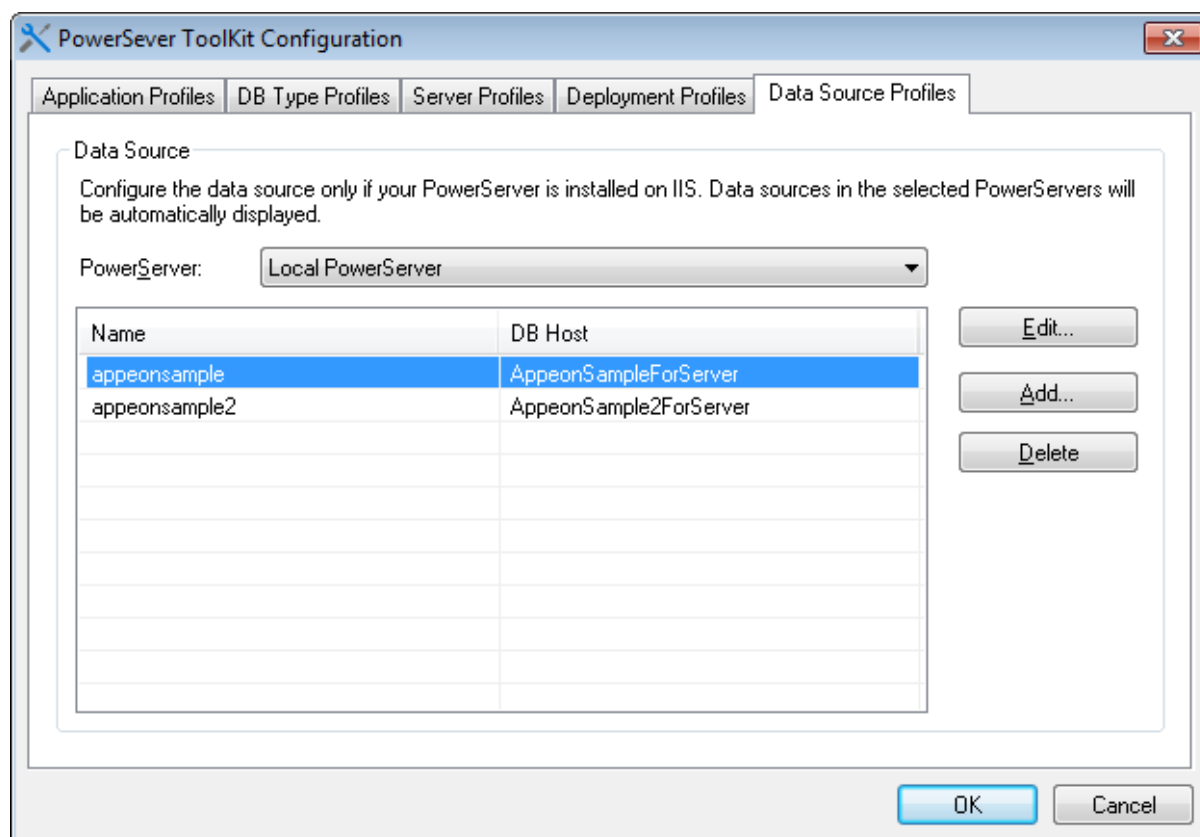
Figure 4.43: Data source profile



Follow the instructions below to create a data source on the Data Source Profile tab:

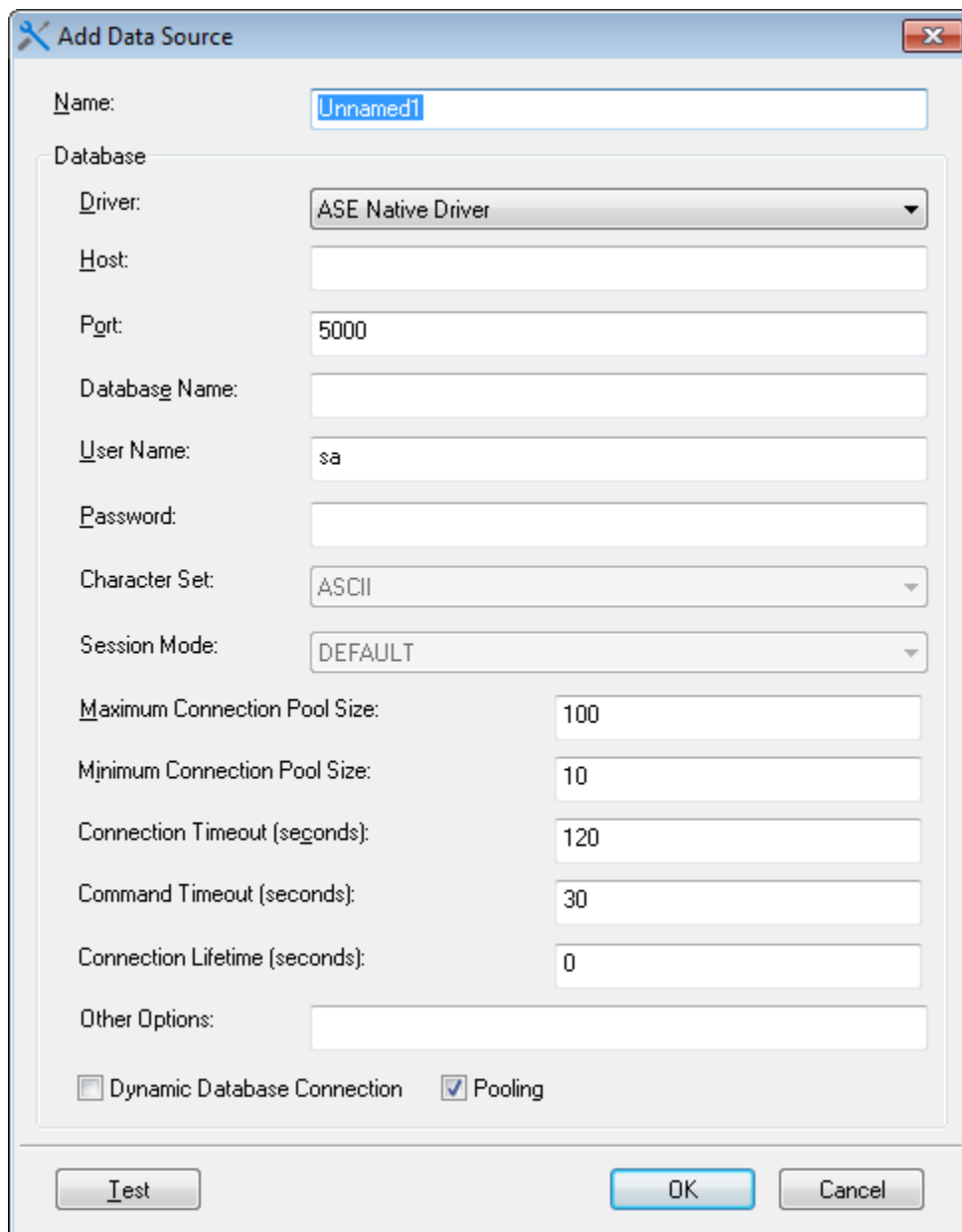
Step 1: Select a PowerServer from the **PowerServer** list box. Only the configured PowerServer profiles are listed.

The data sources existing on the application server will be displayed.

Figure 4.44: Data sources existing on the application server

Step 2: Click the **Add** button to create a data source in the application server. The Add dialog box is displayed.

Figure 4.45: Add a data source



The following table provides instructions for how to specify the data source settings.

Table 4.15: Instructions to specify data source settings

Settings	Instructions
Name	Type the name of the data source. The data source name can contain a combination of letters, underscores ("_"), and numbers. Do not use double-byte characters (such as Chinese, Korean, Japanese characters, etc.) or spaces.
Driver	Select the driver type for the data source.
Host	Type the database host.
Port	Type the database port.
Database Name	Type the database name.

Settings	Instructions
ODBC Data Source	Type the data source name only when the driver type is ODBC-JDBC Bridge.
User Name	Type the database login username. The username is set on the database server.
Password	Type the database login password. The password is set on the database server.
Maximum Connection Pool Size	Specify the maximum number of connections PowerServer opens and pools on startup.
Minimum Connection Pool Size	Specify the minimum number of connections PowerServer opens and pools on startup.
Connection Timeout	Specify the timeout period for the connection.
Command Timeout	Specify the timeout period for the commands.
Connection Lifetime	<p>Specify the lifetime period for the pooled connection.</p> <p>To be specific, you can specify the timeout period from the time a connection is created to the time when the connection returns to the pool. If the actual lifetime of a connection exceeds that specified in Connection Lifetime, the connection will be terminated. It is recommended to be used in cluster configuration when forcing the load balancing between a running server and a server just brought online.</p> <p>If the value is set to 0, the pooled connection allows the maximum connection timeout.</p>
Other Options	<p>Specify the connection_authentication property for SAP SQL Anywhere database.</p> <p>The value should be "authentic_code=####". "authentic_code" is the key word, text after "=" should be the authorization code.</p> <p>PowerServer will automatically set the connection_authentication property when connecting with the SAP SQL Anywhere database.</p> <p>This setting is effective to SAP SQL Anywhere database only.</p>
Dynamic Database Connection	<p>Select whether to enable dynamic database connection.</p> <p>When it is on, the LogID and LogPass of the Transaction object will be used to connect to the database; when it is off, the user name and password specified in the data source will be used to connect to the database.</p>
Pooling	Select whether to use the connection pool.

Refer to Section 4.4, “Setting up PowerServer data sources” in *PowerServer Configuration Guide for .NET* or in *PowerServer Configuration Guide for J2EE* for how to specify the settings for different JDBC drivers and database types.

5 Using UFA Tool

Unsupported Features Analysis (UFA) tool helps you analyze an application for unsupported features.

The following table gives brief descriptions of the unsupported features report.

Table 5.1: UFA Report

	UFA Report
Purpose of the report	Enables you to work around or remove unsupported features in order to make the PowerBuilder application suitable for Apeon Web or mobile migration.
What is reported	<p>Most of the unsupported PowerBuilder coding features including keyword, event, data type, system function, control (itself and its property, function, event), object (itself and its property, function, event).</p> <p>IMPORTANT: There are still some unsupported features that cannot be reported. These undetected features, though only a small number, may make up the largest proportion of your workload depending on how they are used, so we strongly recommend you carefully check the list in Section 5.1.4, “Undetected Unsupported Features”.</p>
How to generate the report	Use the Unsupported Features Analysis tool, or perform the first two tasks in Apeon Deployment Wizard.
Features	<p>These features apply for the report:</p> <p><u>Automatic incremental analysis.</u> After the first analysis, the subsequent analysis is automatically incremental to save time, but the report generated still contains all the information as if it were a full analysis.</p> <p><u>Analysis at application, PBL or object level.</u> You can choose to generate UFA Report for an application, PBL(s) in the application, or object(s) in the application.</p>

This chapter describes:

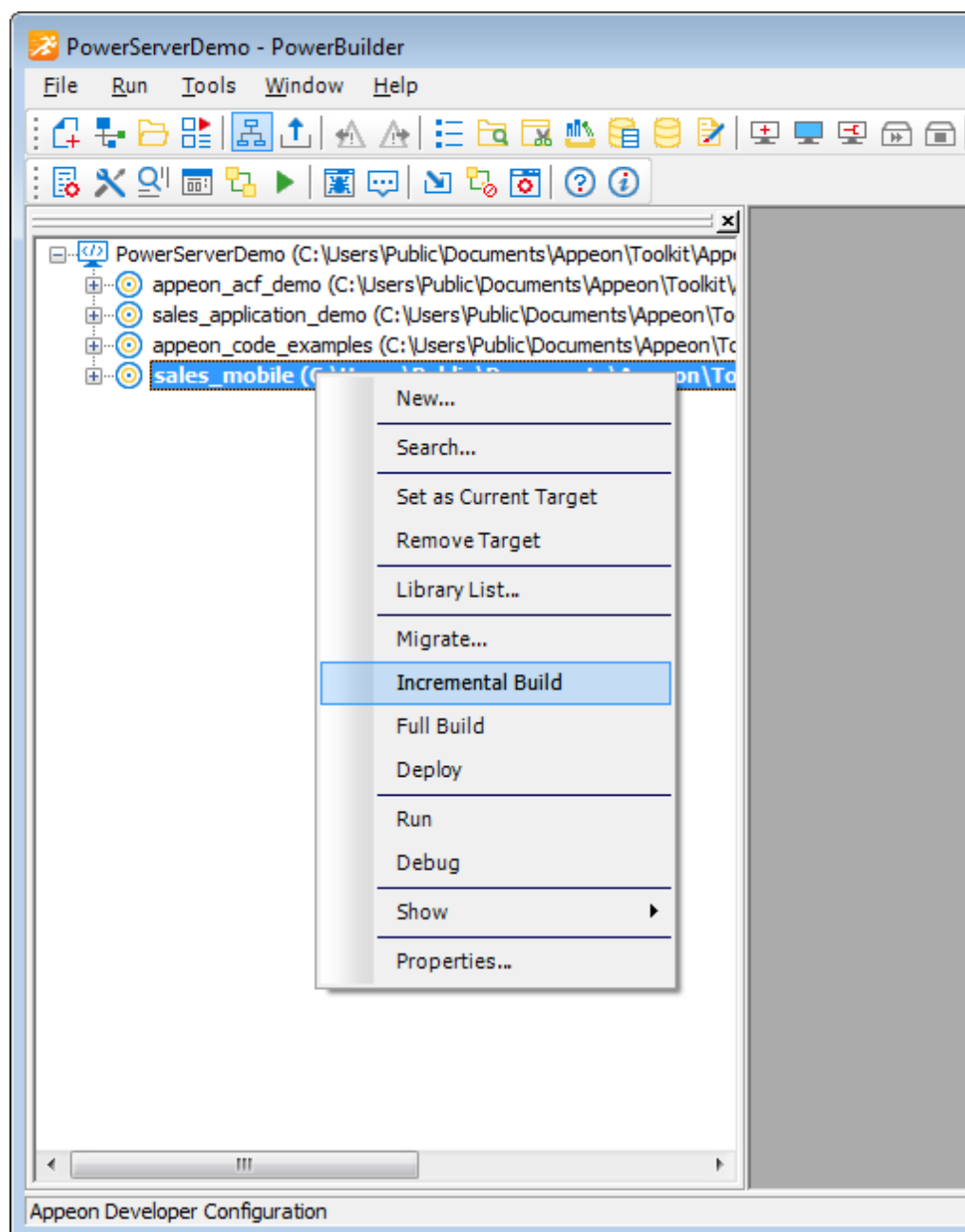
- How to use the UFA tool to analyze an application and generate the UFA Report.
- Undetected unsupported features. There are a small number of unsupported features that the UFA Report cannot detect.
- How to use the UFA tool to work around unsupported features effectively.
- How to use the UFA tool to manipulate the UFA Report. For example, you can adjust the view of the UFA Report so it suits your preferences.

5.1 Analyzing an application

5.1.1 Tasks required before you perform feature analysis

Step 1: Perform an incremental build or object regeneration for the PowerBuilder application. Right-click on the application target and select a build option (for example, **Incremental Build**) as shown in the following figure.

Figure 5.1: Incremental Build



The following table shows you which type of build options you should select for the application.

Table 5.2: Recommended build options

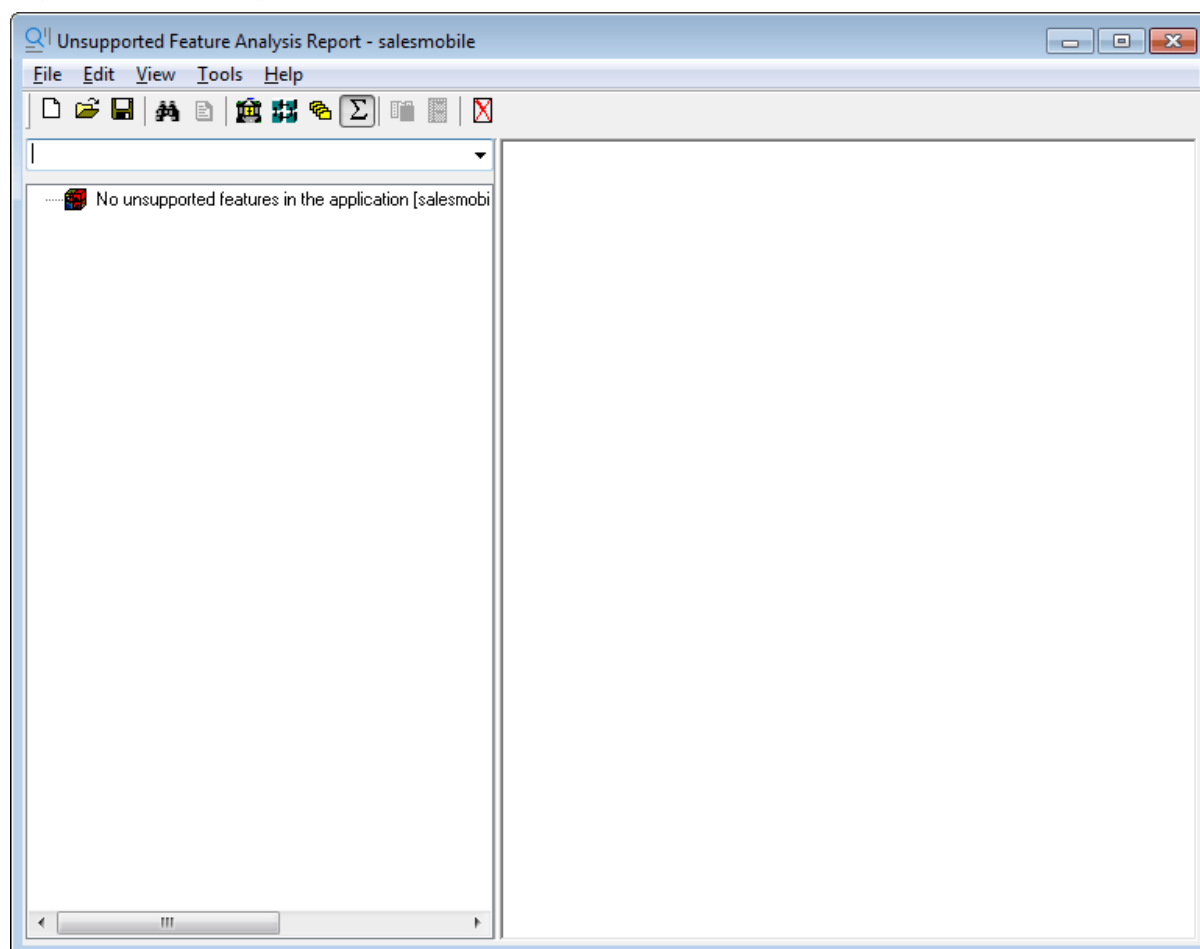
Build Option	What It Does	Recommended For
Regenerate	Refreshes the timestamp for a PowerBuilder object.	Object-level Features Analysis. If the object is regenerated, the object will be analyzed even if no change has made to it.
Incremental	Refreshes the objects that have been changed.	Application-level Features Analysis.
Full	Refreshes all objects.	Not recommended.

Step 2: Perform PBL optimization for the PowerBuilder application.

The purpose of PBL optimization is to remove gaps from libraries and defragment the storage of objects to get rid of potential problems in PBL files.

5.1.2 Accessing the UFA tool

Click the **Analyze** button (🔍) in the PowerServer Toolkit to launch the UFA tool. The Unsupported Feature Analysis Report window (UFA Report window) will be displayed.

Figure 5.2: UFA Report Window

UFA Report window provides a variety of menus to help you manage unsupported features. The following table gives a brief description of these menus. Some of them are also listed as shortcut menus on the toolbar.

Table 5.3: UFA Report Window

Menu	Description
File	Generates a UFA report. See Section 5.1.3, “Performing feature analysis” . Opens or saves a UFA report. See Section 5.2.2.1, “Opening or saving a UFA Report” .
Edit	Opens and modifies the unsupported source code in the PowerBuilder painter. See Section 5.2.1, “Modifying unsupported features” . Searches for the unsupported features. See Section 5.2.2.3, “Searching for UFA Report items” . Filters the unsupported features according to objects, unsupported feature types or priorities. See Section 5.2.2.4, “Filtering UFA Report items” . Expands or collapses treeview. Displays the report in the specified level. See Section 5.2.2.5, “Specifying report display level” .
View	Selects a display mode of the objects in the Unsupported Feature List treeview. See Section 5.2.2.2, “Selecting report view mode” .
Tools	Accesses the workaround for the selected unsupported feature. Defines the priority settings of UFA Report items. See Section 5.2.2.6, “Defining the priority settings of unsupported features” . Customizes the general settings of the UFA Report. See Section 5.2.2.7, “Customizing the general settings of the UFA Report” .
Help	Provides the version number of the UFA tool.

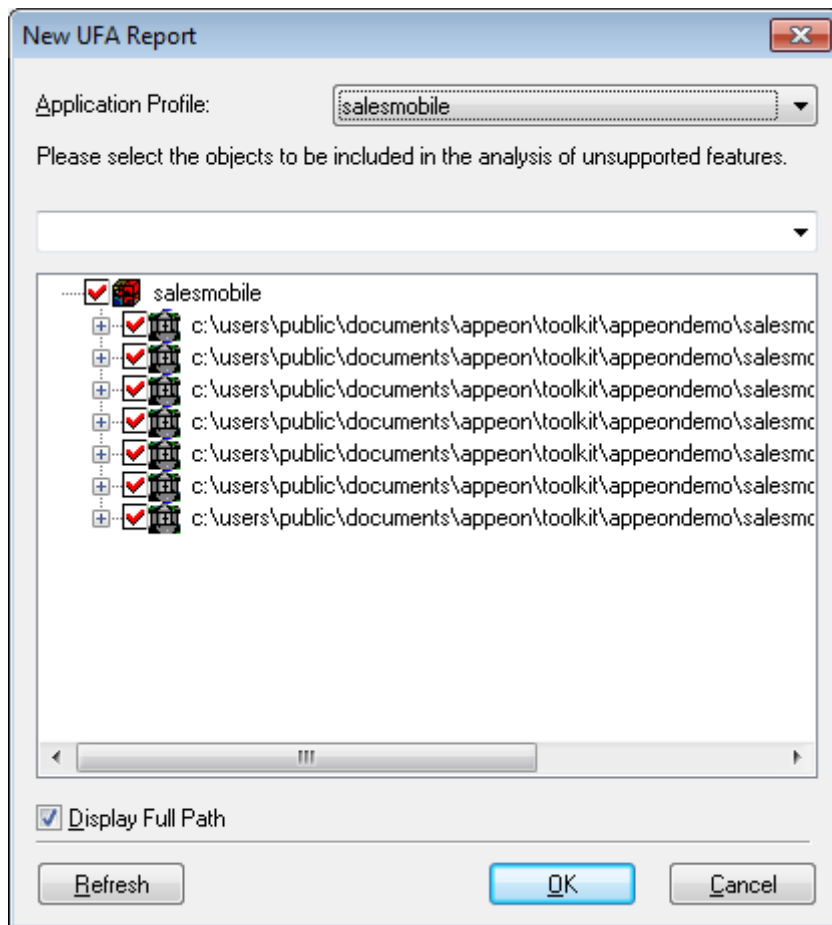
5.1.3 Performing feature analysis

There are two ways to perform a feature analysis of an application:

- Use the Unsupported Features Analysis tool. This option enables you to do a feature analysis of the whole application, or PBL(s) in the application, or objects in the application.
- Use the first two tasks in Appeon Deployment Wizard. When you use Appeon Deployment Wizard to deploy an application, the first task the wizard performs is a feature analysis of the whole application.

This section mainly gives instructions on the first option (using the Unsupported Features Analysis tool). For more information about the second option, refer to [Section 6.2, “Deployment process”](#).

Step 1: Select the **File > New Report** menu in the UFA Report window. The New UFA Report dialog box appears, as shown in the following figure.

Figure 5.3: New UFA Report dialog box

The following table gives a brief description of the elements in the New UFA Report dialog box.

Table 5.4: New UFA Report window

Element	Description
Application profile dropdown list	Provides a list of application profiles for you to select and analyze. Only application profiles configured in the PowerServer Toolkit Configuration window will be listed.
Search field	Searches for PBLs or objects that contains the text you enter.
Treeview of objects	Gives the treeviews of PBLs and objects for the application selected in the application dropdown list and allows you to select the whole application or only some objects to analyze.
Display Full Path option	Gives you options to display or hide the full path of the PBLs and objects in the treeview.
Refresh button	Refreshes the PBL and object list in the treeview.

Step 2: Select the application that you want to analyze from the **Application profile** dropdown list.

The default target to analyze is the default application profile in the **PowerServer Toolkit Configuration** window.

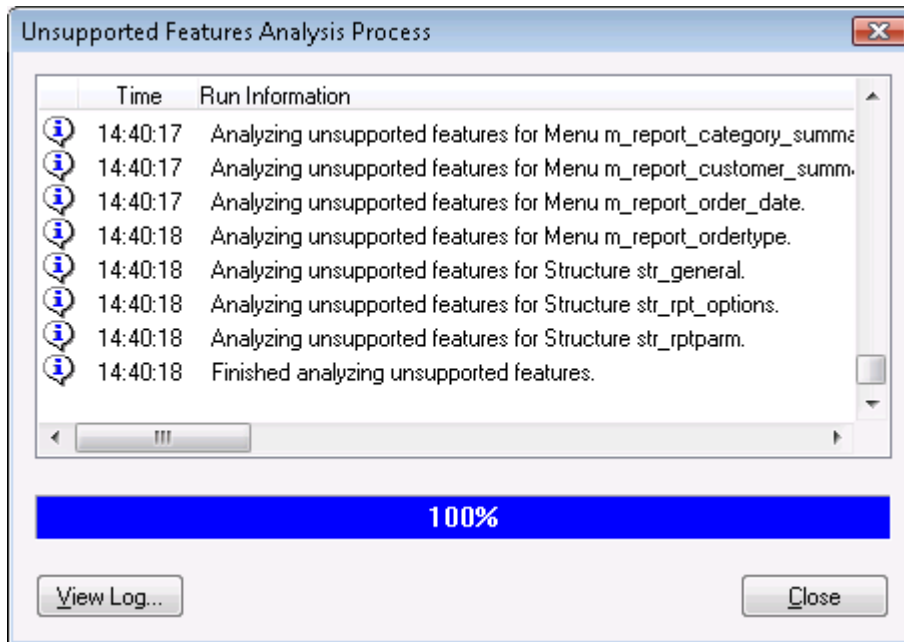
Step 3: Check the PBLs and/or objects in the treeview for which you want to generate the UFA Report.

The treeview lists all the PBLs and objects contained in the selected target. By default, all objects are selected.

Step 4: Click the **OK** button to start the feature analysis.

The feature analysis of the selected objects begins, as shown in the following figure.

Figure 5.4: Unsupported Features Analysis Process dialog



Step 5: Click **Close** when the analysis is completed. The UFA Report is loaded into the UFA Report Window.

5.1.4 Undetected Unsupported Features

Appeon Unsupported Features Analysis is capable of highlighting the majority of unsupported features contained in the PowerBuilder application. There are still a small number of features that the Unsupported Features Analysis will not detect, so they will not be listed in the UFA Report.

An application containing unsupported features can still be converted to the Web or to the Mobile and may work correctly, depending on the type and number of unsupported features. If the deployed application does not work correctly and the UFA Report does not indicate any unsupported features that are likely to cause such problems, the problems may be caused by unsupported features that have not been detected. In this case, it is strongly recommended that you carefully read the undetected features listed below and examine your application to determine if it contains any of these features.

Table 5.5: Undetected features

Naming conventions	<ul style="list-style-type: none"> • Duplicate object names. No two objects should have the same name in an application, whether they are of the same type or not. The Unsupported Features Analysis cannot detect whether object names are duplicated in an application.
---------------------------	--

	<ul style="list-style-type: none"> • More than one application object. Having more than one application object in an application is unsupported. The Unsupported Features Analysis cannot detect whether more than one application object is present. • Applications named as objects or controls. Application names that have the same name as PowerBuilder control/object types are not supported. The Unsupported Features Analysis cannot detect whether this unsupported issue is present in an application.
Null values	<ul style="list-style-type: none"> • Unsupported operation • Expressions with Null values
ASCII characters	<p>Unsupported special ASCII characters:</p> <ul style="list-style-type: none"> • Vertical tab (~v)
Non-visual UserObject	A NonVisualObject object assigned to an autoinstantiated NVO or an autoinstantiated NVO assigned to a NonVisualObject object.
Variables and constants	<ul style="list-style-type: none"> • Instance variables have identical names as global variables. • longlong variable • PUBLIC, PROTECTED, PRIVATE qualifier in the variable declaration
Forced conversion	Forced conversion between types
Window	Multiple MDI windows in an application.
Overloading, overriding functions	<ul style="list-style-type: none"> • Dynamic calling for overloaded functions • Dynamic calling for overriding functions
Using the return value of some supported functions	The return value of the Open or OpenSheet functions
Operators	The operator '^' with embedded SQL statements.
Stored procedure	<ul style="list-style-type: none"> • Stored procedures declared in the conditional statement • DB2 stored procedures • Stored procedures placed inside an Oracle package • Oracle stored procedures with Appeon unsupported features
Cursor declare requirement	<ul style="list-style-type: none"> • Cursor declared in the conditional statement; • If a cursor is declared for retrieving rows from table X, the table X (insert, delete, update) is modified during the cursor declare-close period.

Cursor statements	<ul style="list-style-type: none"> The following two syntax: <ol style="list-style-type: none"> UPDATE TableName SetStatement WHERE CURRENT OF CursorName; DELETE FROM TableName WHERE CURRENT OF CursorName. 			
DataWindow expression function	lastpos	lastposw	mode	pageAcross
	pageCountAcross	profileint	profilestring	stdev
	var	varp		
Partially supported features	Some partially supported features, for example, reading Object.DataWindow.CrossTab.Rows is supported by Appeon, but writing is not, therefore, using DataWindow Modify function or equivalent to write this property cannot be detected by UFA.			
User interface interactions	Unsupported features in the UI such as shortcut key.			
Enumerated data type	Default values of enumerate type variable			
Data source	The data sources of dynamically created DataWindows are stored procedures with input parameters.			
Dynamic calls	<ul style="list-style-type: none"> Dynamically call the method of a menu object. Dynamically call the method that contains reference arguments. 			
Others	<ul style="list-style-type: none"> PBX PSR, for example: dw.dataobject='*.psr' Unsupported DBParm parameters Encoding parameter of the Blob functions filename & importtype arguments of ImportFile, ImportString, & ImportClipboard (filename can only be a text file (TXT)) Property defined in the string variable Structure member has comment property 			
System events	For a complete list of undetected system events, refer to Chapter 13, <i>Undetected Unsupported Features in Supported PB Features for PowerServer Mobile</i> or in <i>Supported PB Features for PowerServer Web</i> .			

5.2 Working with UFA Report

This section describes:

- How to modify unsupported features. You can use the UFA Report to effectively remove unsupported features in the application.
- How to manipulate the UFA Report in the UFA Report Window. You can adjust the view of the UFA Report so it suits your preferences.

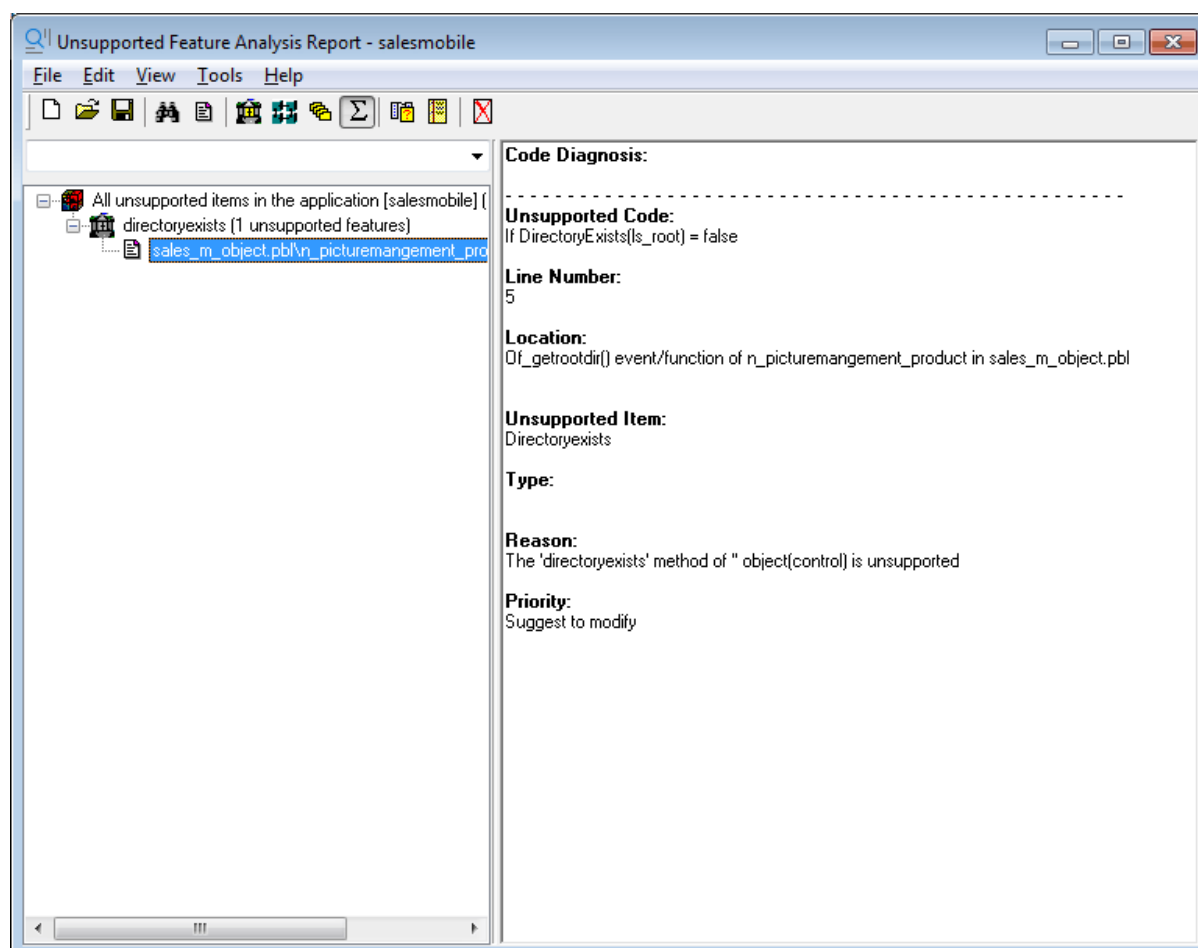
5.2.1 Modifying unsupported features

When the UFA Report is loaded, you can view all the unsupported features in the left treeview and modify them one by one.

Step 1: Expand the unsupported feature list treeview and select the unsupported item.

The detailed analysis of the selected method will be displayed on the right of the window, as shown in the following figure.

Figure 5.5: Unsupported feature details

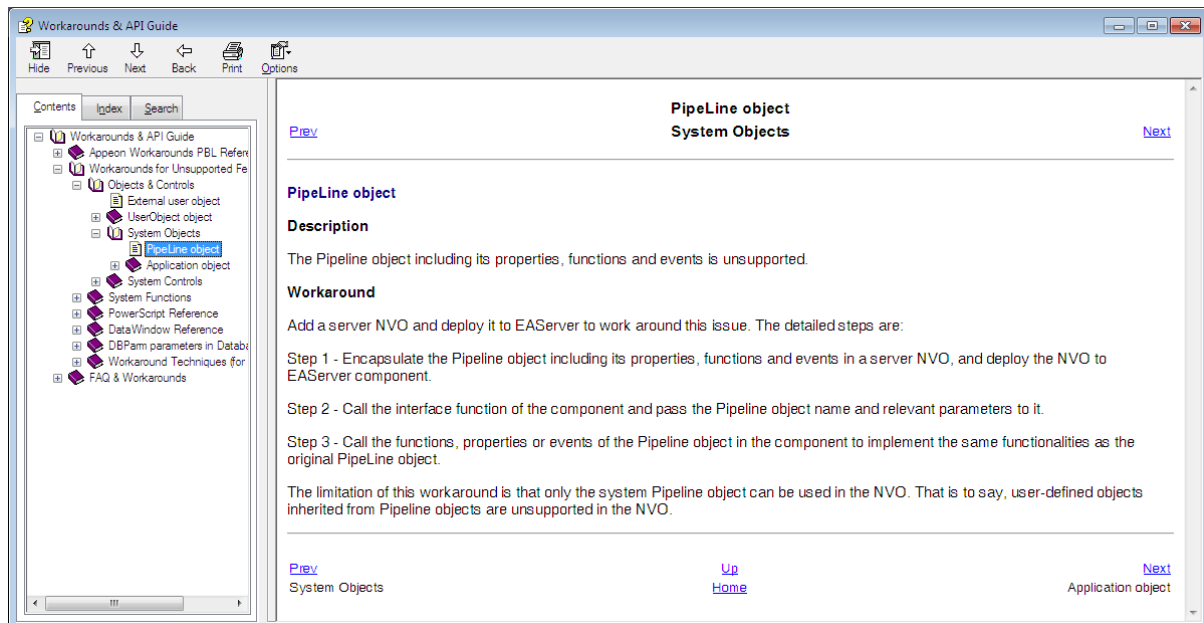


The right text box shows a detailed code diagnosis on the unsupported code in seven aspects: the Unsupported Code, Line Number, Location, Unsupported Item, Type, Reason, and Priority.

Step 2: View the workaround for the unsupported item by right-clicking the unsupported item in the unsupported feature list treeview and selecting **Workarounds solutions**.

- **Go To Workarounds Online:** Opens the online *Workarounds & API Guide* at URL: https://www.appeon.com/support/documents/appeon_online_help/workarounds_and_api_guide/.
- **Go To Local Workarounds:** Opens the local *Workarounds & API Guide* installed with PowerServer Help, as shown in the following figure.

Figure 5.6: Local Workarounds Guide

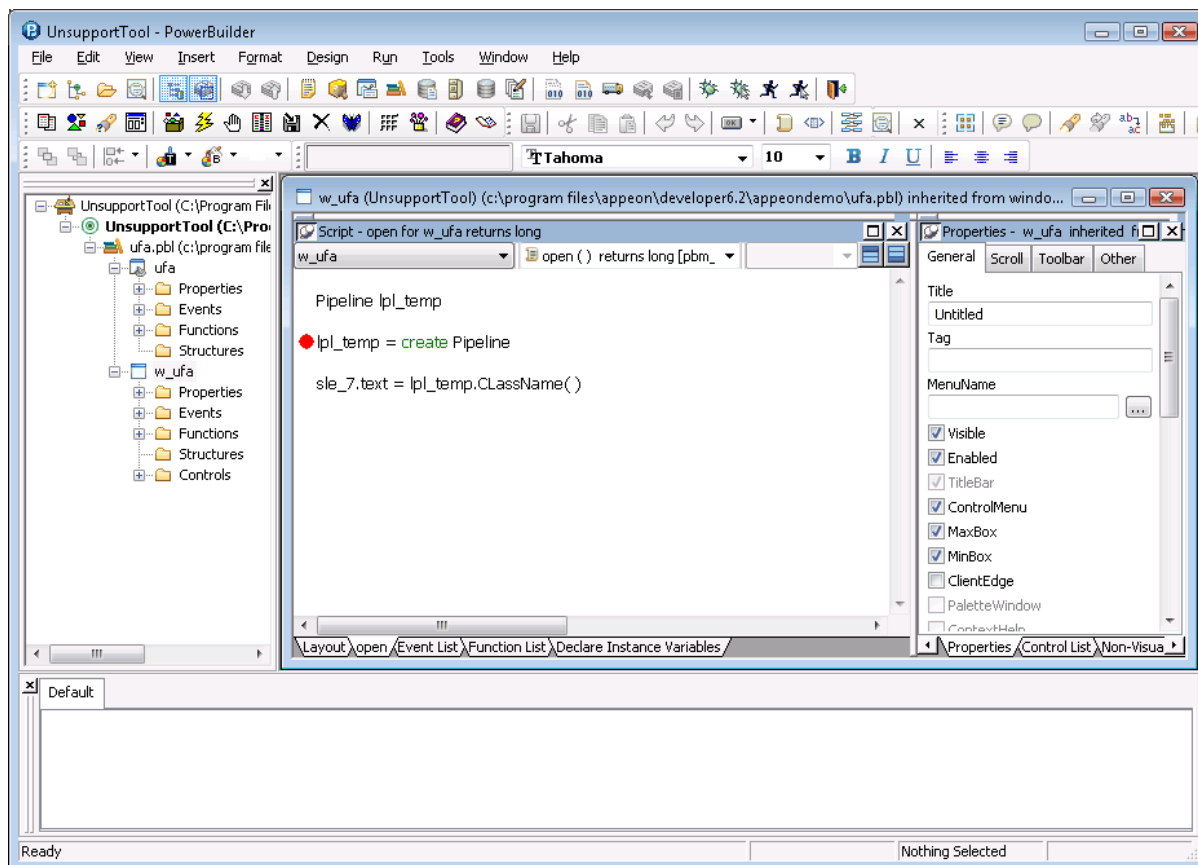


Step 3: Modify the unsupported source code in the PowerBuilder painter according to the Workarounds & API Guide.

Right click the unsupported item in the unsupported feature list treeview, and select **Edit** or **Edit Source** to open the source code in the PowerBuilder Script view.

- **Edit:** Displays the source code of the event or function, which contains the unsupported feature. This button is available only if the unsupported code resides in an event or function and it is not DataWindow-related.
- **Edit Source:** Displays all source code of the object, which contains the unsupported feature. This button is always available.

The PowerBuilder Script view automatically opens, as shown in the following figure.

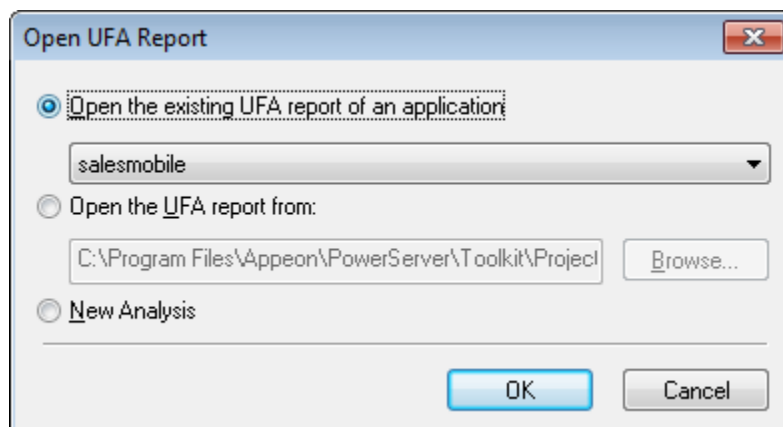
Figure 5.7: Unsupported code in the PowerBuilder painter

5.2.2 Manipulating the UFA Report

5.2.2.1 Opening or saving a UFA Report

You can open or save a UFA Report using the **File** menu in the **UFA Report** window. To save a UFA report as an Excel workbook (.xlsx), it is recommended to use Microsoft Excel 2007 or above.

- To open an existing UFA Report, select the **File > Open Report** menu. The Open UFA Report dialog box is displayed. If there is no UFA report available, select **New Analysis** and click **OK** to analyze an application.

Figure 5.8: Open UFA Report

- To save a UFA Report, select the **File > Save** menu.
- To save a UFA Report with a new name, select the **File > Save As** menu.

5.2.2.2 Selecting report view mode

You can view the UFA Report in four different modes. Go to the **View** menu in the **UFA Report** window and select the desired mode:

- **Category:** Enables you to view the unsupported features in the following categories: Unsupported Objects, Unsupported PowerScript, Unsupported Embedded SQL and Others.
- **Hierarchy:** Enables you to view the unsupported features in hierarchical order: PBL, Object, Control, Method (Event/Function).
- **Priority:** Enables you to view the unsupported features in priorities (you can go to the **Tools > Define Priority** menu to define the priorities of the PowerBuilder features): Have to modify, Suggest to modify, Can be ignored.
- **Summary:** Enables you to view summary information of each unsupported feature.

You can set the default view mode in the **Tools > Options** menu. For detailed descriptions, refer to [Section 5.2.2.7, “Customizing the general settings of the UFA Report”](#).

5.2.2.3 Searching for UFA Report items

To do a quick search, you can directly enter the text that you want to search in the search field of the UFA Report window. Or follow steps below to use the standard Find Text dialog box:

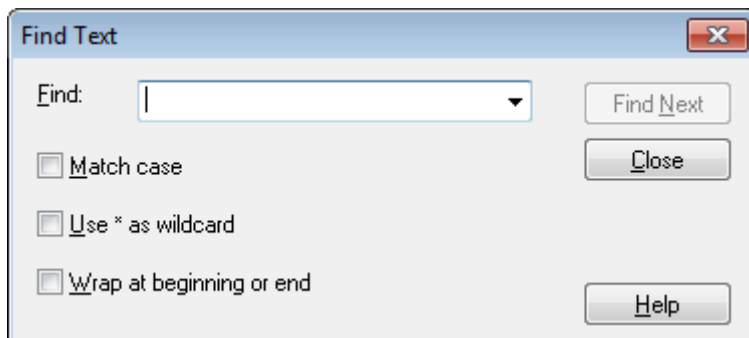
Step 1: Select the **Edit > Find** menu. Or right click in the unsupported feature list treeview and select **Find**.

Step 2: In the **Find Text** box, enter the text that you want to search.

Step 3: Select any other options that you want.

Step 4: Click **Find Next**.

Figure 5.9: Find Text



5.2.2.4 Filtering UFA Report items

Filtering helps you restrict the number of the unsupported features displayed in the UFA Report and quickly find the target unsupported features, especially when there are a large number of unsupported features in the application.

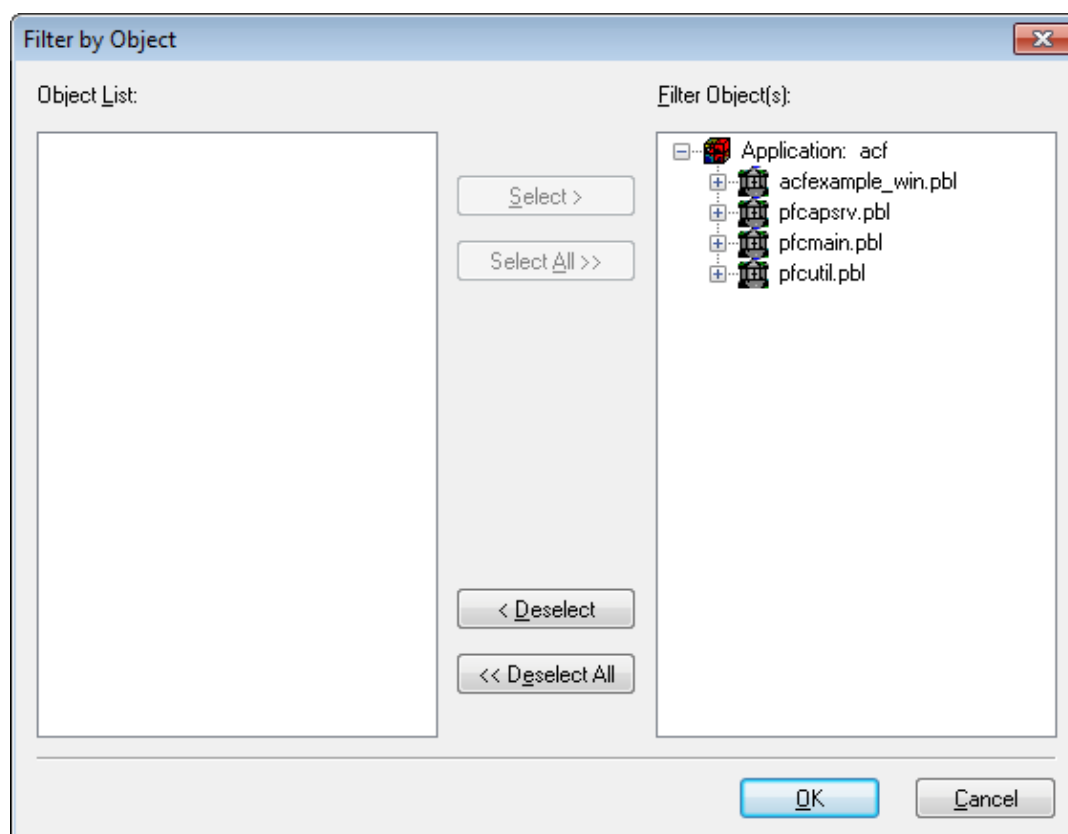
You can filter the unsupported features by the objects they reside in, or by the feature types, or by the priorities.

5.2.2.4.1 Filtering by objects

Step 1: Select the **Edit > Filter by Object** menu. Or right click in the unsupported feature list treeview and select **Filter by Object**.

The **Filter by Object** dialog box is displayed, as shown in the following figure.

Figure 5.10: Filter by object



Step 2: Make sure that the unsupported objects you want to display in the report are on the right list box, whereas the unsupported objects you want to hide are on the left listbox. Click **OK**.

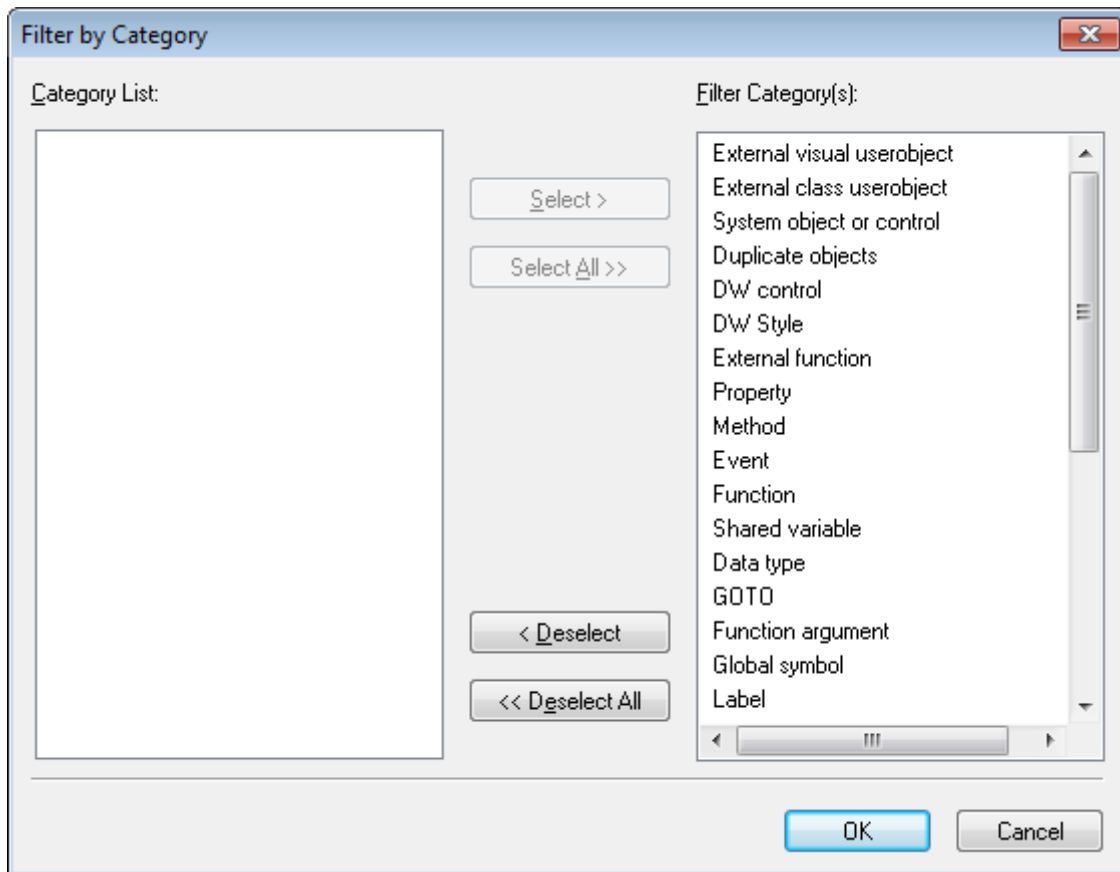
To quickly select all objects of the same type, right click the list and select an object type.

The UFA Report hides the unsupported objects that are on the left listbox in the **Filter by Object** window.

5.2.2.4.2 Filtering by categories

Step 1: Select the **Edit > Filter by Category** menu. Or right click in the unsupported feature list treeview and select **Filter by Category**.

The **Filter by Category** dialog box is displayed, as shown in the following figure.

Figure 5.11: Filter by Category

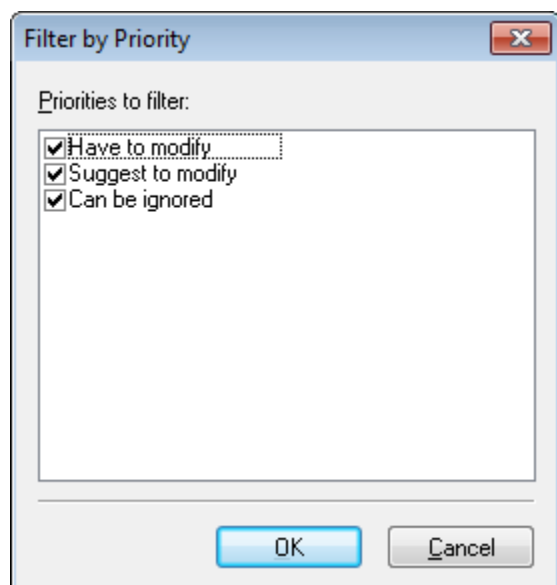
Step 2: Make sure the categories of the unsupported features you want to display in the report are on the right listbox, whereas the categories of the unsupported features you want to hide are on the left listbox. Click **OK**.

The UFA Report hides unsupported features that belong to the categories that are on the left in the **Filter by Category** window.

5.2.2.4.3 Filtering by priorities

Step 1: Select the **Edit > Filter by Priority** menu. Or right click in the unsupported feature list treeview and select **Filter by Priority**.

The **Filter by Priorities** dialog box is displayed, as shown in the following figure.

Figure 5.12: Filter by Priority

Step 2: Uncheck the priorities of the unsupported features which you want to hide. For example, if you want to hide the features that can be ignored, uncheck the **Can be ignored** option.

Step 3: Click **OK**.

The UFA Report only displays unsupported features of the specified priorities.

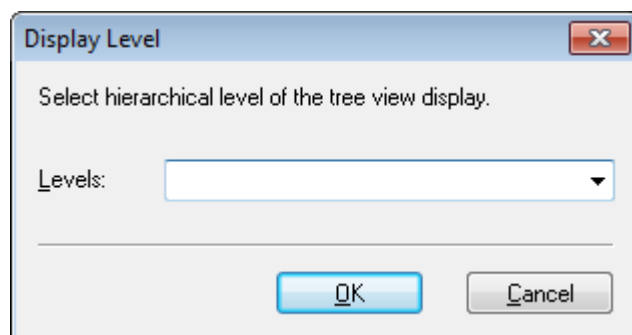
5.2.2.5 Specifying report display level

You can quickly expand or collapse a node of the unsupported feature list treeview by selecting **Expand Sub-node** or **Collapse Sub-node** from the **Edit** menu or from the popup menu on any node.

You can also specify the display level using the following steps:

Step 1: Select the **Edit > Display Level** menu.

Step 2: In the **Display Level** box, enter the number of the level at which the unsupported feature list treeview will display.

Figure 5.13: Display Level

Step 3: Click **OK**.

The UFA Report only displays unsupported features of the specified level.

5.2.2.6 Defining the priority settings of unsupported features

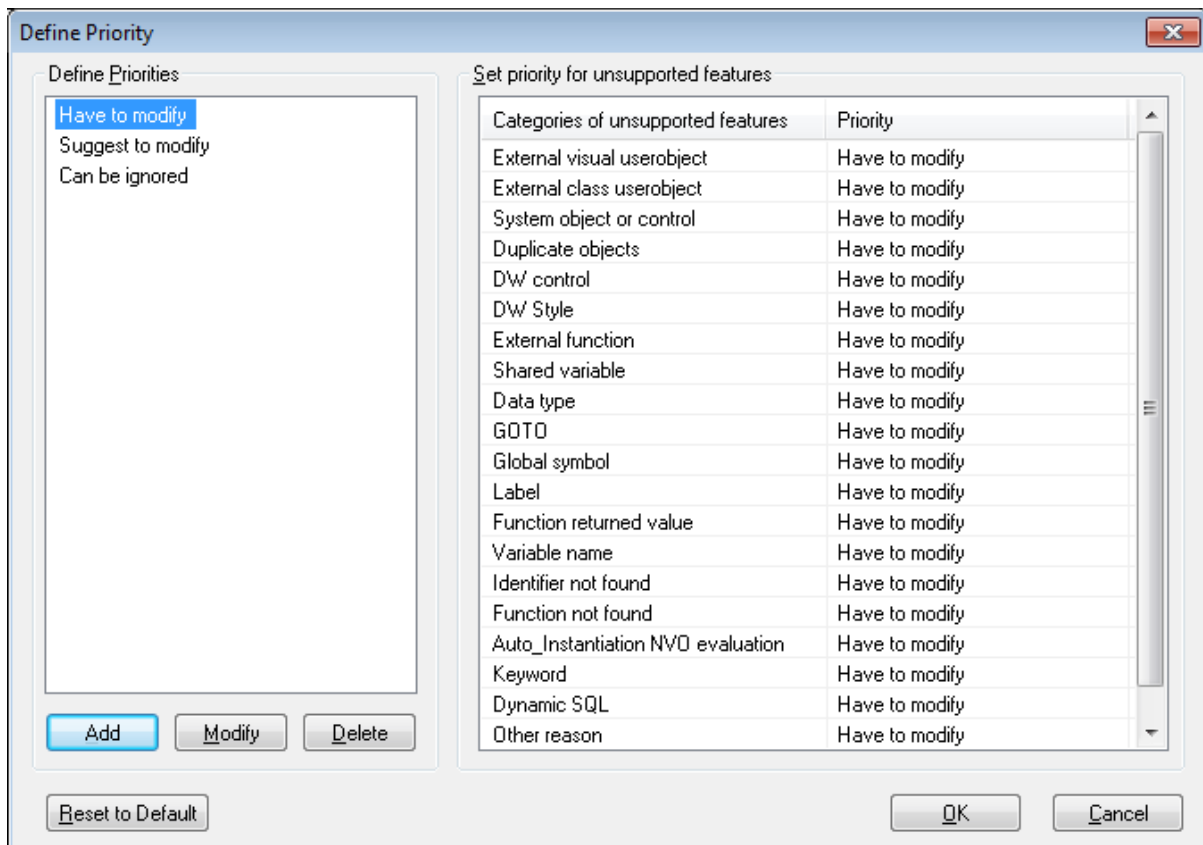
By default, Apeon divides all unsupported features into three priority levels:

- **Have to modify:** The features flagged with **Have to modify** are important in the application and will cause significant functionality loss if they are not modified or worked around.
- **Suggest to modify:** The features flagged with **Suggest to modify** will not necessarily cause functionality loss if they are not modified or worked around. You can decide whether to modify them according to their functionalities in the application.
- **Can be ignored:** The features flagged with **Can be ignored** are small or trivial features which will not compromise application functionality even if they are not modified or worked around.

To customize or change priority levels for unsupported features, take the following steps:

Step 1: Select the **Tools > Define Priority** menu. The **Define Priority** window is displayed, as shown in the following figure.

Figure 5.14: Define Priority dialog box



Step 2: Add, modify or delete the system default priorities in the left **Priority Level** box.

Step 3: Change priority levels of unsupported feature types by selecting a priority level in the **Priority Name** column.

Step 4: Click **Save** to save the priority settings.

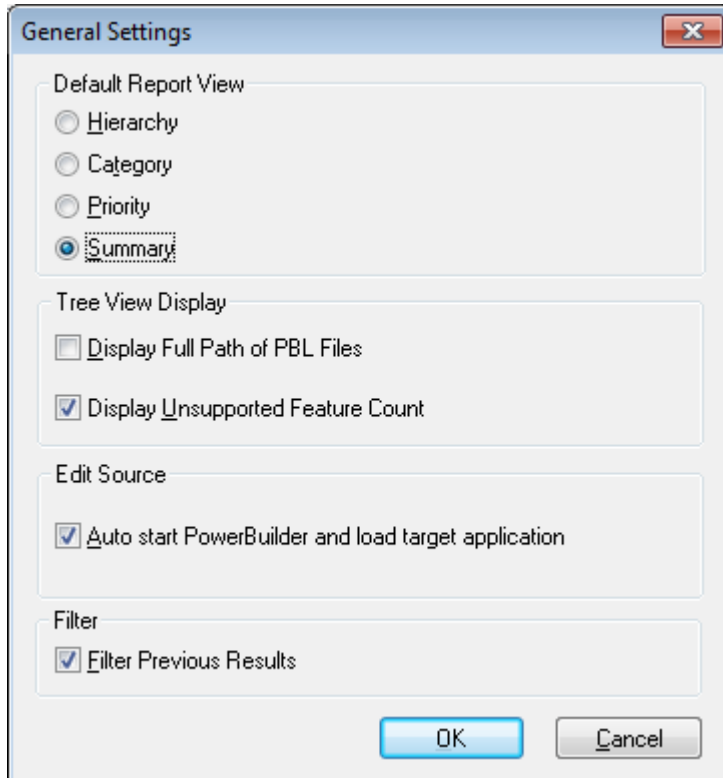
Click **Reset to Default** to restore the system default priority settings.

5.2.2.7 Customizing the general settings of the UFA Report

You can define your preference settings of the **UFA Report** window. These settings include default report view mode, display formats of PBLs, feature count, unsupported source code edition, and filtering. The new settings will be saved as the default settings.

Select the **Tools > Options** menu in the **UFA Report** window. The **General Settings** window is displayed, as shown in the following figure.

Figure 5.15: General Settings dialog box



- *Default Report View group box*: Select the default view mode of the UFA Report. For detailed description of each view mode, refer to [Section 5.2.2.2, “Selecting report view mode”](#).
- *Display Full Path of PBL Files*: Displays the full path and the file name of the PBLs in the UFA Report.
- *Display Unsupported Feature Count*: Displays the total number of the unsupported features in the UFA Report.
- *Auto start PowerBuilder and load target application*: Enables PowerBuilder to automatically start and load the target application when you click the **Edit** or **Edit Source** button in the **UFA Report** window.
- *Filter Previous Results*: Allows you to filter only the results of a previous filter. It narrows a filter that resulted in too many data the first time. If you want to search all data of the report, leave this option as unselected.

6 Deploying PowerBuilder Applications

Appeon Deployment Wizard deploys PowerBuilder applications to be Web or mobile applications. The deployment should take place only after the application has undergone careful analysis and is in compliance with the Basic and Architectural Requirements defined in Chapter 2, *Basic Requirements and Recommendations in Supported PB Features for PowerServer Mobile* or in Supported PB Features for PowerServer Web.

The Appeon Deployment Wizard handles the deployment in five steps:

1. Exports the PowerBuilder application source code.
2. Analyzes the PowerBuilder application to verify that it does not contain unsupported features.
3. Parses the PowerBuilder source code (PBL files) into Web or mobile application files that are stored on the local machine.
4. Uploads the DataWindow syntax to PowerServer.
5. Transfers local application files to the Web Server by either file copy or FTP.

Notes:

- 1) Deploying to an SSL Web Server is no different than deploying to a standard Web Server. You only need to make sure that the SSL Web Server is set up correctly and is fully functional. For detailed instructions on configuring a Web Server as an SSL Web Server, refer to the corresponding documents provided by the Web server vendor.
- 2) If you want to re-deploy an application which is now running online:
 - You can incrementally deploy the application while keeping the application running online, if non-database related source code and image files are changed.
 - You must shut down the running deployed application and then full or incrementally deploy the application if the INI files or database-related source code are changed. Otherwise, the application may have runtime problems such as database connection failure or missing tables or fields.

6.1 Deployment performance

The deployment process can be very CPU and memory intensive, so we recommend that you use a developer computer that conforms to the recommended specifications outlined in Chapter 3, *Installation Requirements in Installation Guide for .NET* for a developer PC to run PowerServer Toolkit.

Be sure to close any unnecessary programs and Windows services in order to have an optimal amount of memory available for the deployment process.

6.1.1 Speed of Deployment Process

The table below benchmarks the time taken or speed to compile various applications in PowerBuilder and deploy to the Web or mobile with Appeon. In general, the results indicate

that the speed of the Appeon deployment process is similar to the speed of the PowerBuilder compilation/build process.

Table 6.1: Appeon Application Deployment Time Benchmarks

	PB EXE w/ PBDs	PB EXE w/ DLLs	Appeon
Sales Application Demo (5.55M)	20 seconds	50 seconds	22 seconds
Appeon Code Example Demo (20.5M)	1 minutes 3 seconds	12 minutes 37 seconds	1 minutes 12 seconds
ACF Demo (21.5M)	2 minutes 5 seconds	24 minutes 46 seconds	2 minutes 12 seconds

As a general rule of thumb, each additional megabyte of PBLs will add approximately 8 seconds to the full deployment process. For example, it would take 4 minutes to deploy a 30MB PFC application. This would only apply for the initial or full deployment. All subsequent deployments should take no more than few minutes or even possibly as little as a few seconds.

This rule of thumb applies to hardware configurations that meet the minimum developer PC requirements stated in Chapter 3, *Installation Requirements in Installation Guide for .NET*. Machines with a slower clock-speed will take longer to perform the deployment.

6.1.2 Deployment duration for full deployments

Remember the following main points regarding the deployment time:

- The process is very CPU and memory intensive.
- The size and complexity of the application affect the deployment time.

The following is an example of how long it takes to deploy an application with 3.69 MB in PBLs and images using full deployment:

Table 6.2: Deployment duration

Deployment Task	Elapsed Time
Task 1: Application Source Code Export	8 seconds
Task 2: Application File Generation	27 seconds
Task 3: Web or Mobile Deployment	15 seconds (for a local deployment)
Test environment: a single CPU Intel P4 1.8 GHz with 256 MB RAM and 60 GB IDE hard drive	

All of the generated application files deployed to the server are a combined 1.25 MB in size.

6.1.3 Deployment duration for incremental deployments

Since the initial full deployment can be time-consuming, Appeon provides an Incremental Deployment for maintaining or upgrading an already-deployed application. This feature

only re-deploys the incremental changes that have been made in the application. Appeon can detect whether an object has been modified and re-deploy only the changed objects.

6.2 Deployment process

6.2.1 Preparing the PowerBuilder application

Appeon recommends that you perform a build and PBL optimization on the PowerBuilder application before performing an Appeon Application Deployment. The following table shows you which type of build options you should select for the application.

Table 6.3: Build options

Build Option	What It Does	When To Perform
Incremental build	Refreshes the objects that have been changed.	Before a full or incremental deployment.
Full build	Refreshes all objects.	Not recommended.
Optimize PBLs	Removes unused space in the PBLs as stored on disk, and reduces object fragmentation.	After building the application but prior to deployment.

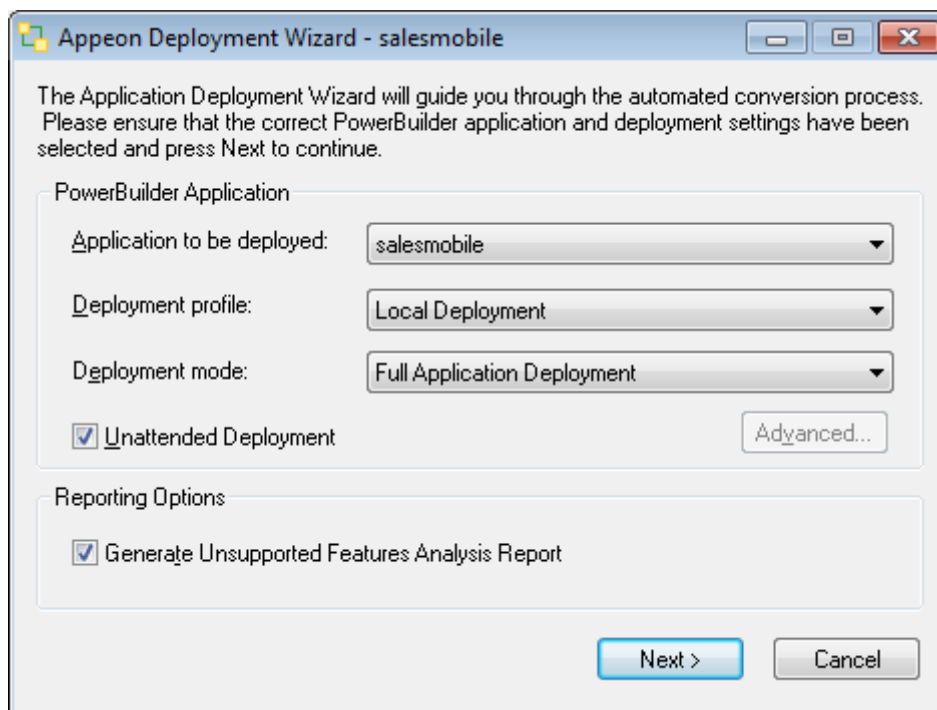
Performing an incremental build in PowerBuilder is necessary because when modifications are made to a parent class, the child class does not reflect the changes unless one of three things occurs:

1. The child class is opened and saved.
2. An incremental build is performed.
3. A full build is performed.

6.2.2 Specifying the deployment settings

Click the **Deploy** button () in the PowerServer Toolkit. The Appeon Deployment Wizard is displayed.

Figure 6.1: Apeon Deployment Wizard



The following table describes how to specify the deployment settings for an application:

Table 6.4: Apeon Deployment Wizard settings

Setting	Description
Application to be deployed list box	Select an application to be deployed. The default application is listed by default. The application chosen for deployment will become the default application under Application Profiles in PowerServer Toolkit Configuration.
Deployment Profile list box	Select a deployment profile to be used in the Web or mobile deployment. The selected deployment profile determines which PowerServer(s) and Web Server(s) the application will be deployed to. You must make sure the PowerServer(s) and Web Server(s) are started before you perform the deployment.
Deployment Mode list box	Select one of the three available deployment modes: <ul style="list-style-type: none"> • Full Application Deployment • Incremental Application Deployment • Deploy Already Generated Application
Unattended Deployment check box	Specify whether the whole deployment process will automatically proceed without displaying the Next button or waiting for interaction. If you want to view the messages or reports generated at each step, you can set the unattended deployment to false; the wizard will require you to click the Next button to start the next task until the whole conversion is complete.

Setting	Description
Advanced button	<p>Select the types of objects or files that will be uploaded to servers: DataWindow objects, other objects, INI files, external DLL/OCX files, or image files. By default, all objects and files will be uploaded.</p> <p>This button is effective only for the Deploy Already Generated Application option.</p> <p>When use this function, verify that the target servers are the same ones used in the application deployments.</p>
Generate Unsupported Features Analysis Report check box	<p>Specify whether to generate an unsupported feature analysis report during deployment.</p> <p>This report lists all unsupported PowerBuilder coding features in the application. You can use this report to remove or work around the unsupported features. For detailed instructions, refer to Section 5.2, “Working with UFA Report”.</p>

6.2.2.1 Selecting the deployment mode

The following table describes the three different deployment modes and guides you in choosing the proper deployment mode.

Table 6.5: Deployment mode

Deployment Mode	What It Does	Elapsed Time	When To Use It
Full deployment	Exports all PowerBuilder application objects and code, generates the corresponding files for the Web or mobile application, and deploys all files to the server.	Depends on the size and complexity of the PowerBuilder application.	<ol style="list-style-type: none"> 1. The first time an application is deployed to the Web or to mobile. 2. After making changes to any ESQL.
Incremental deployment	Exports objects, re-generates and re-deploys files for the changed objects only. The Deployment Wizard continues even if it finds unsupported features during this process.	Up to 80% less time than a full deployment.	After making changes to an already fully-deployed application's source code or features in the application profile.
Deploy Already Generated Application	Bypasses the object exporting and file generation tasks and only re-deploys the Web or mobile files for the changed objects or for the entire application.	Much less time than a Full or Incremental Deployment.	When you intend to: <ol style="list-style-type: none"> 1) deploy the files for the changed objects to the server after an Incremental Deployment has been cancelled during the Web or mobile deployment (Task 3).

Deployment Mode	What It Does	Elapsed Time	When To Use It
			2) refresh the set of files on the server after a Full Deployment has been cancelled during the Web or mobile deployment (Task 3). 3) deploy the application to a different server.

PowerServer Toolkit will automatically set the default deployment mode by detecting whether a folder named after the application exists in the %Apeon%\PowerServer\Toolkit\Projects\ directory on the developer machine. If the folder is detected, the application will be regarded as an already deployed application and incremental deployment mode is selected; otherwise, full deployment mode is selected. This detection can mislead developers into selecting incremental deployment mode, because the folder will exist in the following situations:

Table 6.6: Situations when the folder exists

The folder will exist if you have ...	In this case, choose ...
Performed an unsupported feature analysis on the application before it has undergone a full deployment. The folder is created during analysis.	Incremental deployment mode.
Undeployed an Apeon deployed application from servers using the Application Undeployment Wizard. The folder is preserved.	Full deployment mode.
Deployed the application using a previous version of PowerServer Toolkit. The folder is preserved unless you delete it manually. This legacy folder can be detected by the new PowerServer Toolkit.	Full deployment mode.

6.2.3 Deploying the PowerBuilder application

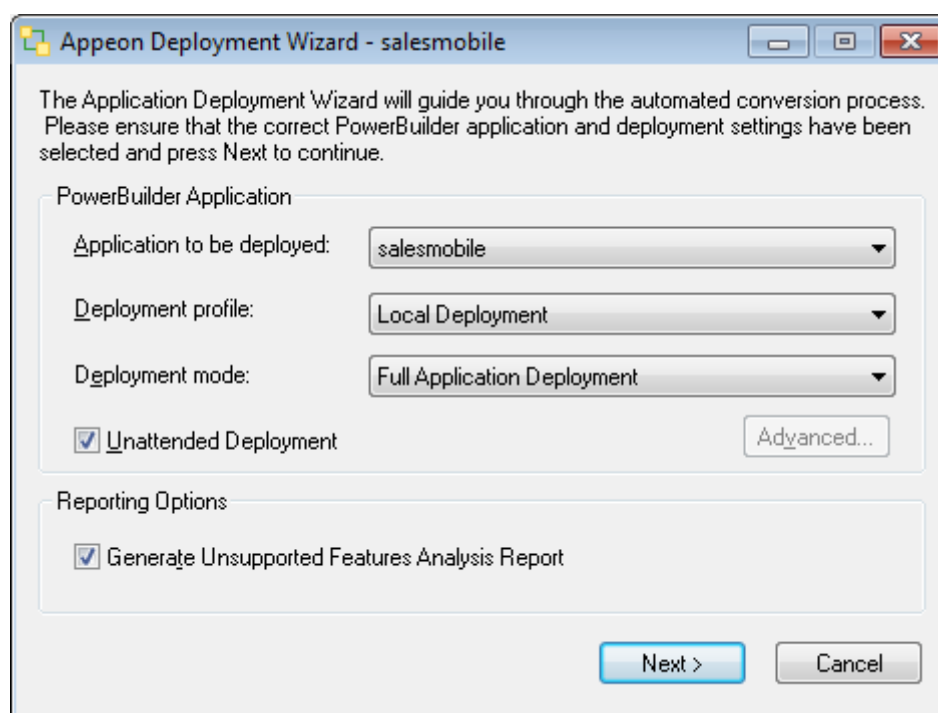
The deployment process of an application contains three major deployment tasks. Different deployment modes perform different tasks. The following table describes the tasks performed during each deployment mode.

Table 6.7: Different tasks performed during each deployment mode

Deployment Task	Full Deployment	Incremental Deployment	Deploy Already Generated Application
Task 1: Application Source Code Export Exports the source code of the original PowerBuilder application.	Y	Y	N
Task 2: Application File Generation	Y	Y (Partial)	N

Deployment Task	Full Deployment	Incremental Deployment	Deploy Already Generated Application
Analyzes unsupported features and generates Web or mobile application files.			
Task 3: Web or Mobile Deployment Deploys the generated Web or mobile application files to PowerServer.	Y	Y (Partial)	Y

Figure 6.2: Full application deployment



As shown in the above figure, the application to be deployed is the **sales_application_demo**, and the deployment mode is **Full Application Deployment**.

The following are the complete steps in the deployment process:

Step 1: Click **Next**. The Apeon Deployment Wizard begins Task 1, Application Source Code Export.

After Task 1 is completed, Task 2, Application File Generation, will automatically start if the **Unattended Deployment** option is selected, otherwise, you must click **Next** at the bottom of the Apeon Deployment Wizard to begin Task 2.

After Task 2 is completed, Task 3, Web or Mobile Deployment, will automatically start, or click **Next** at the bottom of the Apeon Deployment Wizard to begin Task 3 if the **Unattended Deployment** option is not selected.

Confirm that the Web Server and PowerServer have been started and correctly configured in PowerServer Toolkit Configuration; otherwise, deployment will fail. If deployment fails, a **Retry** button appears so you can retry Task 3 without running through a full deployment again.

Click the **Cancel** button or press the **Esc** key to terminate Task 3. If you cancel Task 3 during a Full Deployment, only some of the application files will have been uploaded to the server and the application will not run correctly. To correct this, select *Incremental Deployment* or *Deploy Already Generated Application* as the deployment mode the next time you attempt to deploy the application.

Step 2: After Task 3 has been completed, click the **Next** button at the bottom of the Apeon Deployment Wizard to display the deployment report page, as shown in the following figure.

Figure 6.3: Apeon Deployment Wizard

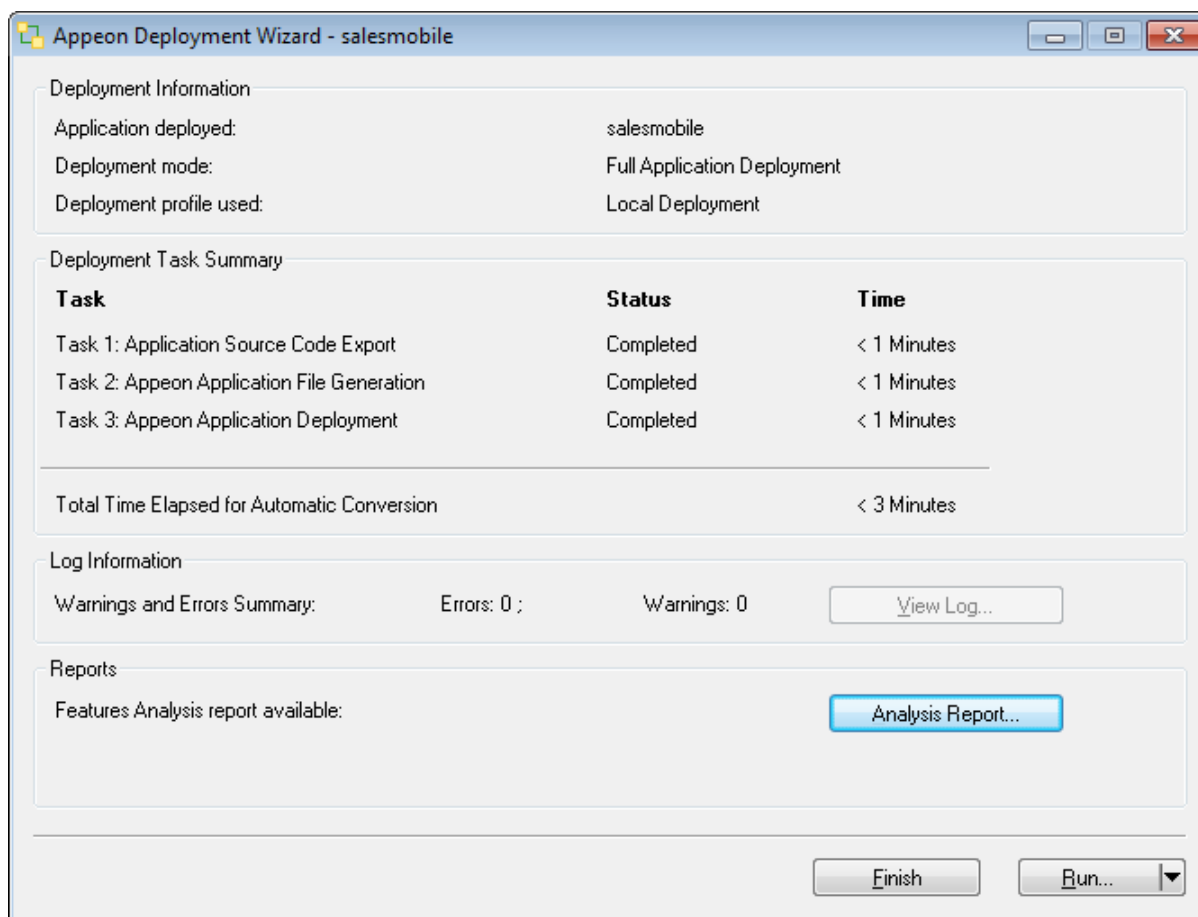


Table 6.8: Deployment report page

Element	Description
Deployment Information	Displays the name of the application deployed, the deployment mode used, and the deployment profile used.
Deployment Task Summary	Displays the status and time of each of the three deployment tasks, as well as the total time for the entire PowerBuilder-to-Web or PowerBuilder-to-mobile conversion process.
Log Information	Displays the number of errors and warnings that occurred during the deployment process. Click the View Log button to view the errors and warnings generated during the deployment process. The log file can be managed in the Information Manager. For detailed instructions, refer to Chapter 9, Using Information Manager .

Element	Description
Reports	Displays the reports (Analysis Report) generated during the deployment process (mainly Task 2). Analysis Report button: This report is available if the Generate Unsupported Feature Analysis Report option on the Appeon Deployment Wizard is selected before deployment. After you click this button, the UFA Report Window will be displayed. For detailed instructions, refer to Section 5.2, “Working with UFA Report” .

Step 3: Click the **Finish** button to close the Appeon Deployment Wizard, or click the **Run** dropdown list and then select a project loader to run the newly deployed application. If you click the **Run** button directly, the last used project loader will be launched to run the application.

For more information about the project loader in the **Run** dropdown list, see [Launching applications from the Run button](#).

Note:

- Before you select a project loader from the **Run** dropdown list to run the application, please make sure the database connection from PowerServer to the database server has been properly configured. For detailed instructions, refer to the Chapter 4, *Database Connection Setup* in *PowerServer Configuration Guide for .NET* or in *PowerServer Configuration Guide for J2EE*.
- If the WebLogic (production mode) or WebSphere is used as the Web server, you will need to go to the Web server and manually deploy the application. This is required by the WebLogic (production mode) and WebSphere, not by Appeon. It is similar to deploying the PowerServer EAR package; you can follow the instructions in the section called "Deploying appeonserver.ear package" in the Installation Guide for Appeon PowerServer.

7 Debugging Appeon Web Applications

Sometimes a deployed application may not behave well although the original application does. In these situations, you can examine the execution results of the JavaScript code in the deployed application using Appeon Debugger.

The usage of Appeon Debugger is similar to that of PowerBuilder Debugger. You set breakpoints or watch variables in the PowerBuilder source code, same as in PowerBuilder Debugger. The main difference is that Appeon Debugger runs the generated JavaScript code but not the PowerBuilder source code, and shows the execution results of the JavaScript code.

This chapter assumes that you are familiar with the functions of PowerBuilder Debugger; therefore it does not provide descriptions of common debugging features, but focuses on introducing the specialties of Appeon Debugger. For common debugging features, refer to the *Debugging an application* section in the *PowerBuilder User's Guide*.

7.1 Important Requirements

To start and use Appeon Debugger successfully, be aware of the following important prerequisites and note:

1. Appeon Debugger can work on both 32-bit and 64-bit machines. **But for 64-bit machines, Appeon Debugger can only work with 64-bit Internet Explorer.**
2. Appeon Debugger can work with Internet Explorer 9.0 or later (both 32-bit and 64-bit), and does not work with other Web browser such as IE 7/8, Edge, Chrome, Firefox, & Opera. However, you can run (not debug) the Appeon Web application in the other Web browser.
3. PowerServer Mobile provides no debug tool, but you can still take advantage of Appeon Debugger to debug the business logic of the mobile application in Internet Explorer.
4. Microsoft Script Debugger must have been correctly installed and registered on the same machine and script debugging is enabled in Internet Explorer. Otherwise, you may encounter problems in starting and using Appeon Debugger.
 - **Modify the Windows registration table:** In order to make sure that the Appeon Debugger work correctly, you need to modify the Windows registration table after the installation of the Microsoft script debugger.

For 64-bit machines, go to HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Internet Explorer\Main\ in the Windows registry, add the following

- one **DWORD** value: *TabProcGrowth* = 0
- two **String** values: *Isolation* = PMIL, *IsolationImmersive* = PMIL

Go to HKEY_CURRENT_USER\SOFTWARE\Microsoft\Internet Explorer\Main\, and if the same items exist there, make sure they have the same value as shown above.

For 32-bit machines, go to HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Internet Explorer\Main\ in the Windows registry, add one **DWORD** value:

TabProcGrowth = 0. Also go to HKEY_CURRENT_USER\SOFTWARE\Microsoft\Internet Explorer\Main, and if TabProcGrowth exists there, make sure it is set to 0.

- **Enable the script debugging option:** Simply uncheck the **Disable script Debugging (Internet Explorer)** checkbox under the **Internet Options > Advanced** tab in Internet Explorer.


If you encounter any error when starting the Appeon Debugger, refer to the solutions in Section 2.4, “Appeon Debugger” in *PowerServer Troubleshooting Guide*.

5. A script in the deployed application must meet all of the following requirements to be debugged; otherwise Appeon Debugger will simply ignore it:

- The script is deployed in the **Debug PS/JS** mode using the local PowerServer Toolkit.

If you have changed the application PBLs, or the local application files and the remote application files are not the same, you must re-deploy the application. Or the newly-added code will not be selectable in Appeon Debugger.

- The original PowerScript code of the script is supported by Appeon.

If the code is unsupported, it is marked with a  icon in the Appeon Debugger **Source** view, and cannot be set as breakpoint, nor be located with the **Run To Cursor** or **Set Next Statement** commands in the debugger.

7.2 Introduction to the debugging procedure

Although Appeon Debugger is used for debugging the JavaScript in the deployed application, breakpoints are not set in the JavaScript, but directly in the PowerBuilder code loaded in the Appeon Debugger Source view. The generated JavaScript is invisible throughout the entire process, however the results shown in the Watch view are the execution results of the JavaScript.

The debugger procedure with Appeon Debugger is the typical 5-step procedure as with PowerBuilder Debugger:

Step 1: Start the debugger.

Step 2: Set breakpoints.

Step 3: Run the application in debug mode.

Step 4: Examine the application at the breakpoints by watching variables and expressions, and monitoring the call stack. You may also step through the application after a breakpoint to closely examine the effects of each code.

Step 5: Fix the code.

The actions performed in each step are not very different from those required in PowerBuilder Debugger. However, there are some special requirements related with the actions, and the action results are from the running of JavaScript. Continue reading the following sections for detailed descriptions of these debugging steps.

7.3 Starting Appeon Debugger

To start Appeon Debugger:

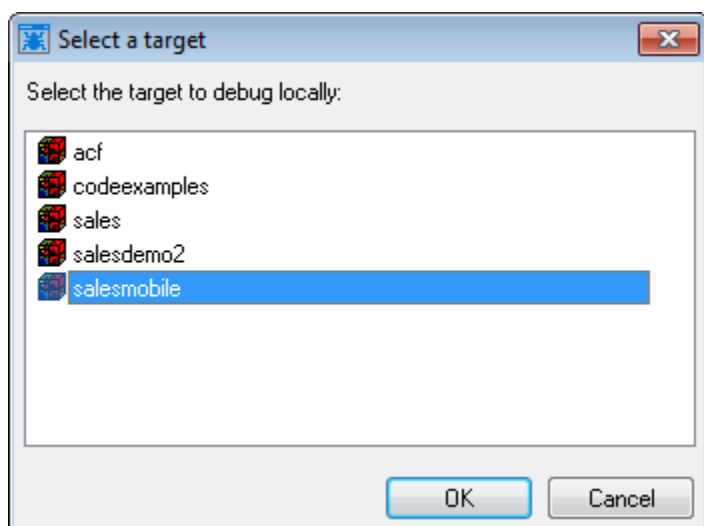
Step 1: Close the other PowerServer Toolkit tools.

Step 2: Start PowerServer and the Web server that host the Web application.

Step 3: Click **Appeon Debugger** (🔧) on the PowerServer Toolkit.

The "Select a target" window is displayed.

Figure 7.1: Select an application

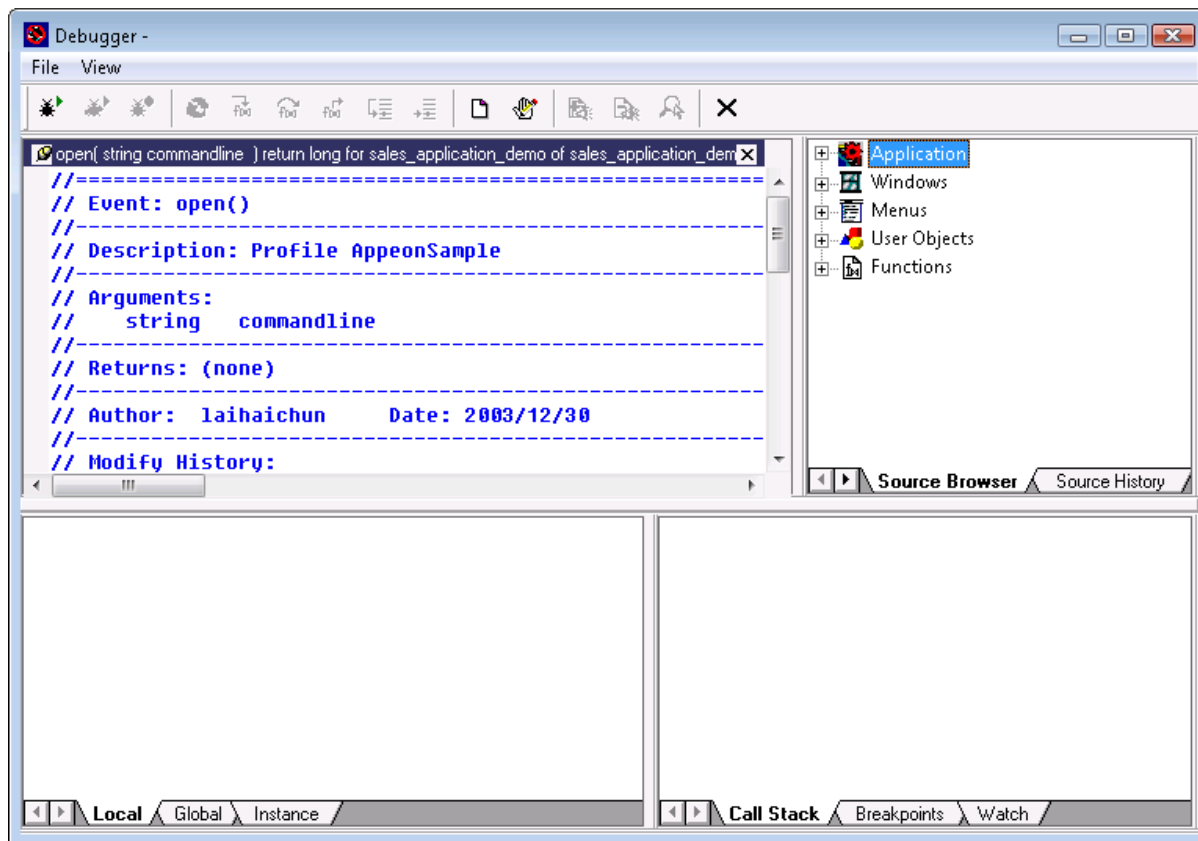


Step 4: Select an application from the list and click **OK**.

Appeon Debugger starts and the selected application is loaded in the Appeon Debugger.

The menu bar of the debugger provides three simple menus: File, View and Help. The toolbar is almost the same as that in PowerBuilder Debugger, except that the "Start Remote" button is currently unavailable.

Figure 7.2: Appeon Debugger



7.3.1 Views in Appeon Debugger

Appeon Debugger contains all the views provided by PowerBuilder Debugger except for Instances View and Objects in Memory view. Each Appeon Debugger view shows almost the same information as its counterpart in PowerBuilder Debugger. The following table summarizes the general differences between the two.

Table 7.1: Comparing the views in Appeon Debugger and PowerBuilder Debugger

Views	Unique to Appeon Debugger (Unavailable in PowerBuilder Debugger)	Unique to PowerBuilder Debugger (Unavailable in Appeon Debugger)
Source	Identifies Appeon unsupported features with the icon (🔴)	<ul style="list-style-type: none"> • Ancestor Script popup menu item • Descendant Script popup menu item
Source Browser	None	List of system events, or the events and functions defined in parent classes, or some special events (such as Menu OnCreate event)
Source History	None	None
Variables	The values of variables unsupported by Appeon are marked as "unsupported".	<ul style="list-style-type: none"> • Parent and Shared views • Parent popup menu item

Views	Unique to Appeon Debugger (Unavailable in PowerBuilder Debugger)	Unique to PowerBuilder Debugger (Unavailable in Appeon Debugger)
	The Global view displays all global variables used in the application, while PowerBuilder Debugger only displays instantiated global variables.	<ul style="list-style-type: none"> • Shared popup menu item • Break on changes popup menu item
Call Stack	None	None
Breakpoints	None	<ul style="list-style-type: none"> • Setting special breakpoints • Setting occasional or conditional breakpoints • Setting a breakpoint when a variable changes
Watch	Appeon Debugger handles some expressions in a unique way.	Break on changes popup menu item

Although the views of Appeon Debugger and PowerBuilder Debugger look similar, they cannot be used together. For example, the breakpoints set in Appeon Debugger cannot be recognized by PowerBuilder Debugger, and vice versa.

7.4 Setting breakpoints

7.4.1 Code lines that can be set as breakpoints

Pay attention to the following **do's** and **don'ts** before setting breakpoints:

1. **Do** set breakpoints to the code lines supported by Appeon.
2. **Do** set simple breakpoints to script, **don't** set special breakpoints, such as breakpoints triggered when a statement has been executed a specified number of times (an occasional breakpoint), when a specified expression is true (a conditional breakpoint), or when the value of a variable changes.
3. To set a SQL statement that spans multiple lines as a breakpoint, **do** set the breakpoint at the first line, **don't** set breakpoints at the other lines. If you try to set a breakpoint at the other lines, the breakpoint will be set to the line that follows the SQL statement.
4. To set a PowerScript non-SQL statement, which spans multiple lines with continuation characters (&) as a breakpoint, **do** set the breakpoint at the last line of the statement, and if it is set at the other lines, the breakpoint icon is still added to the last line.

Note: you can set a breakpoint to the FOR statement, but the application will be suspended for only once when this statement is executed.

7.4.2 Methods for setting breakpoints

You can set breakpoints in the Source, Breakpoints, Variables, or Watch view, before or after the application starts to run in debug mode.

Method #1: Setting a breakpoint in the Source view

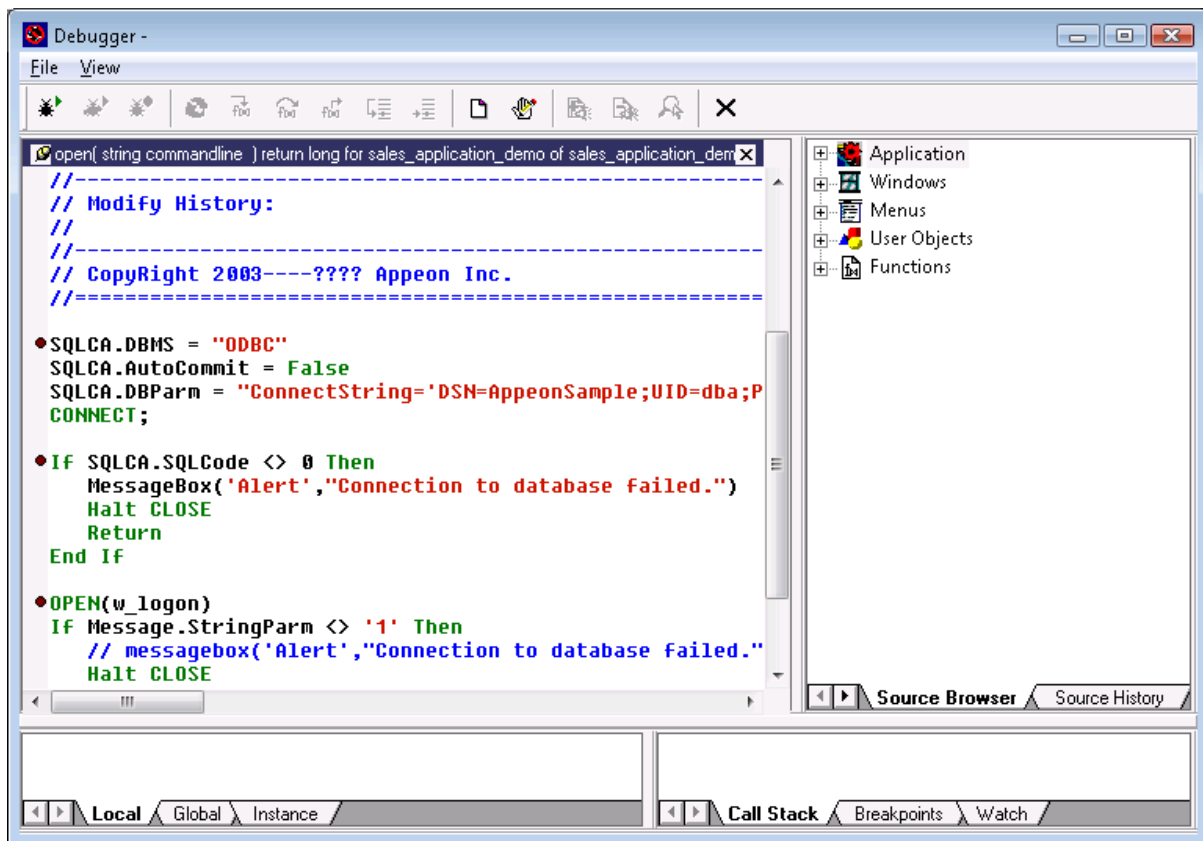
Step 1: Select the script to display in the Source View.

Select a function or event in the Source Browser view to display its script in the Source View, or right click on the Source view and select **Select Script** from the popup menu to select the script to be displayed.

Step 2: Right click a line in the Source view and select **Insert Breakpoints** from the popup menu.

Once a breakpoint is set, a red circle displays at the beginning of the line.

Figure 7.3: Breakpoints are set



Method #2: Setting a breakpoint in the Breakpoints, Variables, or Watch view

Step 1: Right click the Breakpoints, Variables, or Watch view;

Step 2: Select **Breakpoints** from the popup menu to add a breakpoint using the Edit Breakpoints dialog.

Method #3: Removing or adding a breakpoint at runtime

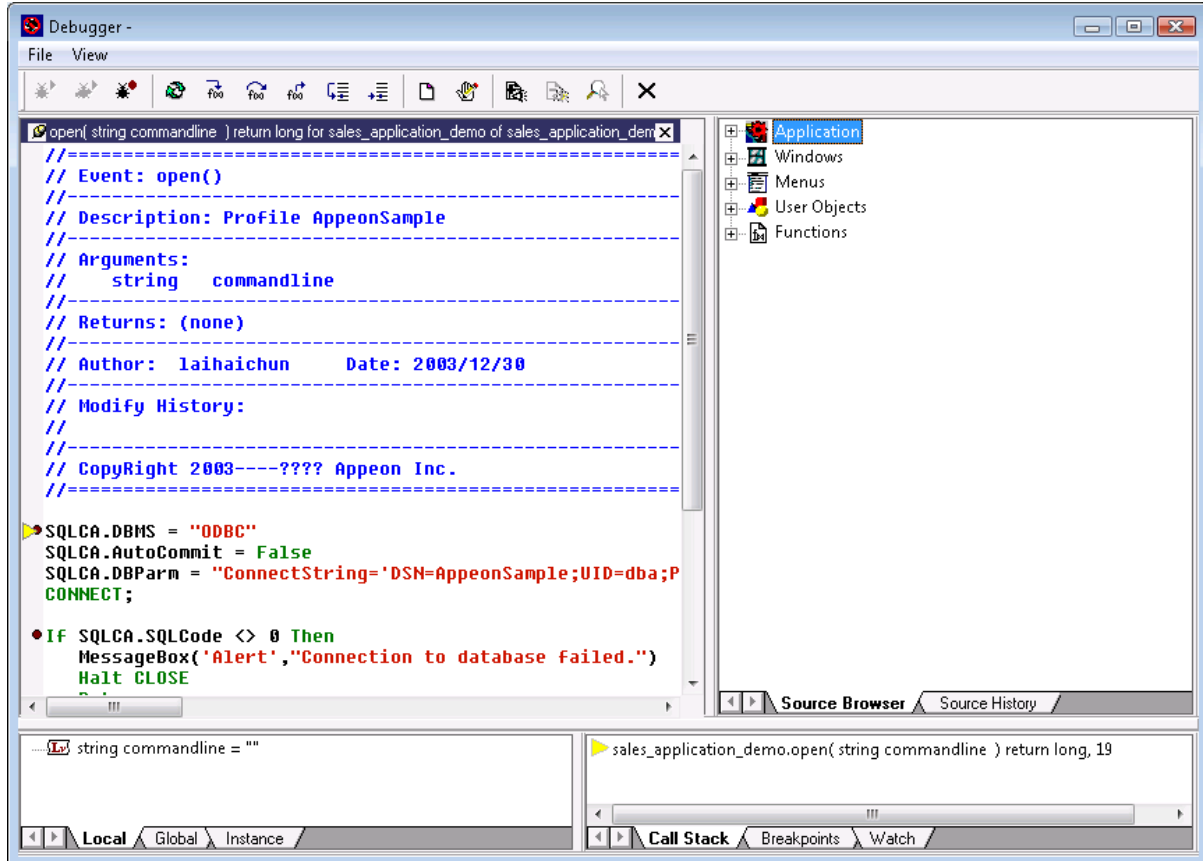
When the application is already running in debug mode, you can still change the way the debug works by removing or adding breakpoints at runtime:

Double-click a line in the Source view, or select **Insert Breakpoint** from the popup menu in the Source view.

7.5 Running the application in debug mode

Click **Start** (🐞) on the Appeon Debugger toolbar. The application starts to run in the Internet Explorer. It works normally until it reaches a statement containing a breakpoint. At this point it stops so that you can examine the application in the Appeon Debugger.

Figure 7.4: Application running in debugger mode



7.6 Examining an application at a breakpoint

Appeon Debugger provides a Quick Watch function, and the Variables, Watch and Call Stack views that work the same as PowerBuilder Debugger for examining the state of the application. It does not provide Break on Changes and TipWatch functions, nor the Object in Memory view.

7.6.1 Special variable and expression handlings

Appeon Debugger watches the variable and expression results out of JavaScript code but not PowerBuilder code. This is the main difference with PowerBuilder Debugger. Despite the difference, the results shall be mostly the same as in PowerBuilder Debugger. If the results are different, there are two possible reasons:

1. The deployment of the application contains an error. In this case, determine the cause of the error and correct the problem.

2. Apeon Debugger handles some variables and expressions differently from PowerBuilder Debugger. Such differences are introduced below. Be aware of them, so not be false alarmed and conduct unnecessary complex examination work.

Apeon Debugger does NOT support the following when examining variables:

- Expanding the values of ClassDefinition, PowerObject, and Any variables;
- Expanding the runtime values of System variables. Instead, it just expands the inherent properties of such variables.

For example, with a Window type variable w_1, PowerBuilder Debugger shows all the values of all the controls and variables contained in w_1, while Apeon Debugger only shows the values of the Window properties of for w_1.

Apeon Debugger gives different results when evaluating the following expressions:

- It reports errors for expressions that contain unsupported code;
- If the same expression is added multiple times -- in different context -- into the Watch, Apeon Debugger regards it as one expression, and evaluates its value according to the context of the breakpoint;
- Apeon Debugger does not validate expressions. Therefore, if an expression does not pass validation in PowerBuilder, PowerBuilder Debugger does not give evaluation results, however Apeon Debugger may give a result, sometimes even an incomprehensible result, such as "[object]+3" for the expression "this + 3".
- If two expressions are interconnected, because the evaluation results of the expressions are dependent on the orders they are executed, and the ordering may follow different rules in PowerBuilder Debugger and Apeon Debugger, Apeon Debugger may produce different results from PowerBuilder Debugger.

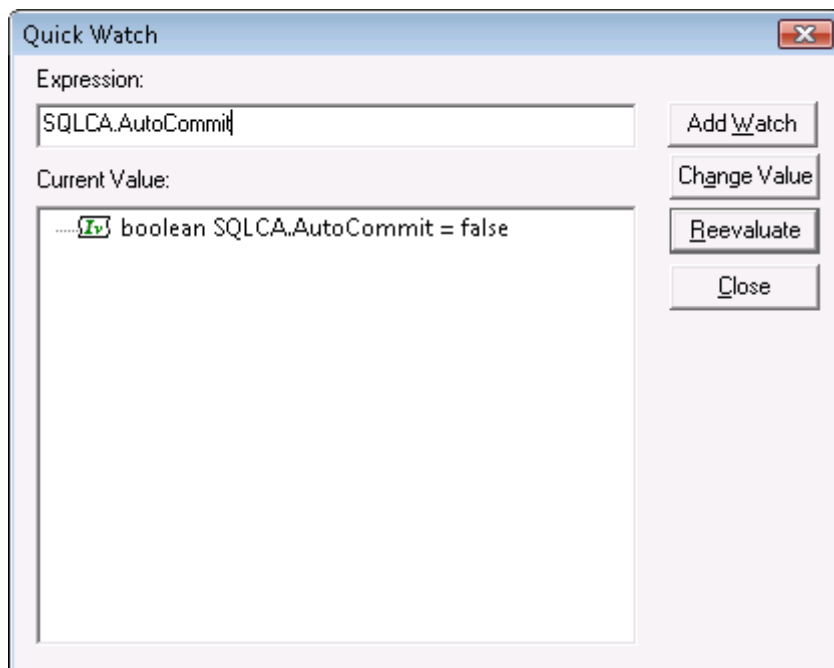
7.6.2 Adding variables or expressions to Watch view

Method #1: Adding variables or expressions to Watch with the Quick Watch

Step 1: Select a variable or expression or place the cursor to a variable or expression in the Source view, right click and select **Quick Watch** from the popup menu or press Shift+F9.

The variable or the expression will be added to the Expression box on the Quick Watch dialog.

Step 2: Click the **Add Watch** button to add the variable or expression to the Watch view.

Figure 7.5: Quick Watch**Method #2: Adding variables or expressions directly in the Watch**

Step 1: Right click the Watch view and select **Insert** from the popup menu.

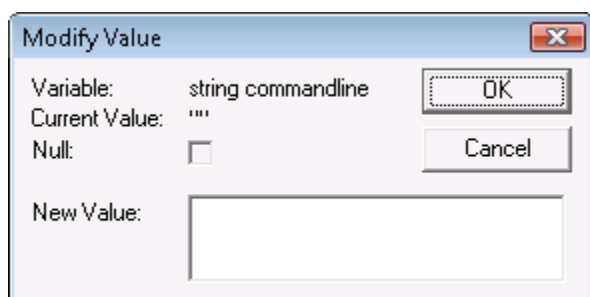
Step 2: Type valid PowerScript expression in the New Expression dialog box, and click **OK** to add it to the Watch view.

7.6.3 Changing the value of a variable or expression

To change the value of a variable in the Watch view:

Step 1: Double click a variable or right click a variable and select **Edit Variable**. The Modify Value dialog box will be displayed.

Step 2: In the Modify Value dialog box, type a new value for the variable in the New Value box, click **OK** to close the Modify Value dialog box and continue debugging the application with the variable set to the new value.

Figure 7.6: Modify value

To modify an expression in the Watch view:

Step 1: Double-click an expression or right click an expression and select **Edit Expression**. The Modify Expression dialog box will be displayed.

Step 2: In the Modify Expression dialog box, type the new expression in the New Value box and click **OK**.

Notice that it is unsupported to select the Null check box in the Modify Value or Modify Expression dialog to set the value of the variable or expression to null.

7.6.4 Evaluating an expression

To evaluate an expression in the Quick Watch dialog box and add it to the Watch view:

Step 1: Change the variable or expression in the Expression box.

Step 2: Click **Reevaluate** to display the new value in the tree view.

Step 3: (Optional) Click **Add Watch** to add the expression to the Watch view.

7.6.5 Examining context in Call Stack view

You can examine the context of the application by clicking the code lines in the call stack.

To show a different context from the **Call Stack** view: double-click a line or right-click a line and then select **Set Context**.

Once a function is selected in the Call Stack view, the Variable views, the Source view, and the Watch view will all be updated.

7.6.6 Stepping through the application

You can use the Step In, Step Over, Step Out, Run To Cursor, and Set Next Statement commands to step through an application. Using these commands in Apeon Debugger is no different from using them in PowerBuilder Debugger.

7.7 Fixing the code/stopping the debug procedure

Once you locate the cause of the erroneous execution results, you can stop the debugging procedure, return to the PowerBuilder application to fix the code, and re-deploy the application to verify that the error has been corrected.

To stop the debugging process: close Internet Explorer where the application is running, or select **Stop** on the toolbar. The breakpoints you set to the application will be saved when Apeon Debugger is closed, and will be available when Apeon Debugger is re-opened.

When fixing the code, you must return to the PowerBuilder IDE and make code changes in the original PowerBuilder application. Redeploy the application and run the application in Internet Explorer to verify if the problem has been corrected. If the problem still exists and you need to return to Apeon Debugger to redeploy the application. Otherwise, the latest code changes will not be reflected in Apeon Debugger.

8 Running Apeon Applications

Apeon Web applications are accessed via Web browsers and Apeon mobile applications are accessed via Apeon Workspace.

Confirm that the Web Server(s) and PowerServer(s) hosting the Web or mobile applications are running, and verify that the Web browser meets the necessary requirements outlined in Chapter 3, *Installation Requirements in Installation Guide for .NET*.

8.1 Requirements for running Web app

8.1.1 Windows account privileges

Note: This is required for the Web application only.

When you run an Apeon Web application for the first time, the Apeon Xcelerator plug-in (for improving the runtime performance of Web applications) must be downloaded to the client.

- If using the Edge, Firefox, Chrome, or Opera Web browser, you can log into the machine as a normal user, because you need no special privileges to download and install the Apeon Xcelerator plug-in.
- If using Internet Explorer, then different Internet Explorer versions require different privileges for installing the Apeon Xcelerator plug-in.

If using IE 8.0 or 9.0 on Windows 7 or later, you can log into the machine as a normal user, because you need no special privileges to download and register the Apeon Xcelerator plug-in. However, if using IE 7.0 or earlier, you must log in as a member of the Administrator group so you have the permissions to successfully download and register the Apeon Xcelerator plug-in.

8.1.2 Internet Explorer settings

Note: This is required only if you run the Web application in Internet Explorer. If you run the Web application in other Web browsers such as Edge, Firefox, Chrome, & Opera, you can skip this section.

Settings in Internet Explorer can affect the deployed Web application. Sometimes outdated files that are cached in the browser can interfere with how a Web application functions and can cause errors when they are re-deployed many times. Internet Explorer's settings may also block proxy connections or ActiveX control downloads when the Web application is opened. Therefore, you need to make sure you configure Internet Explorer's settings accordingly. Note that you may need to have the administrator rights to modify certain settings.

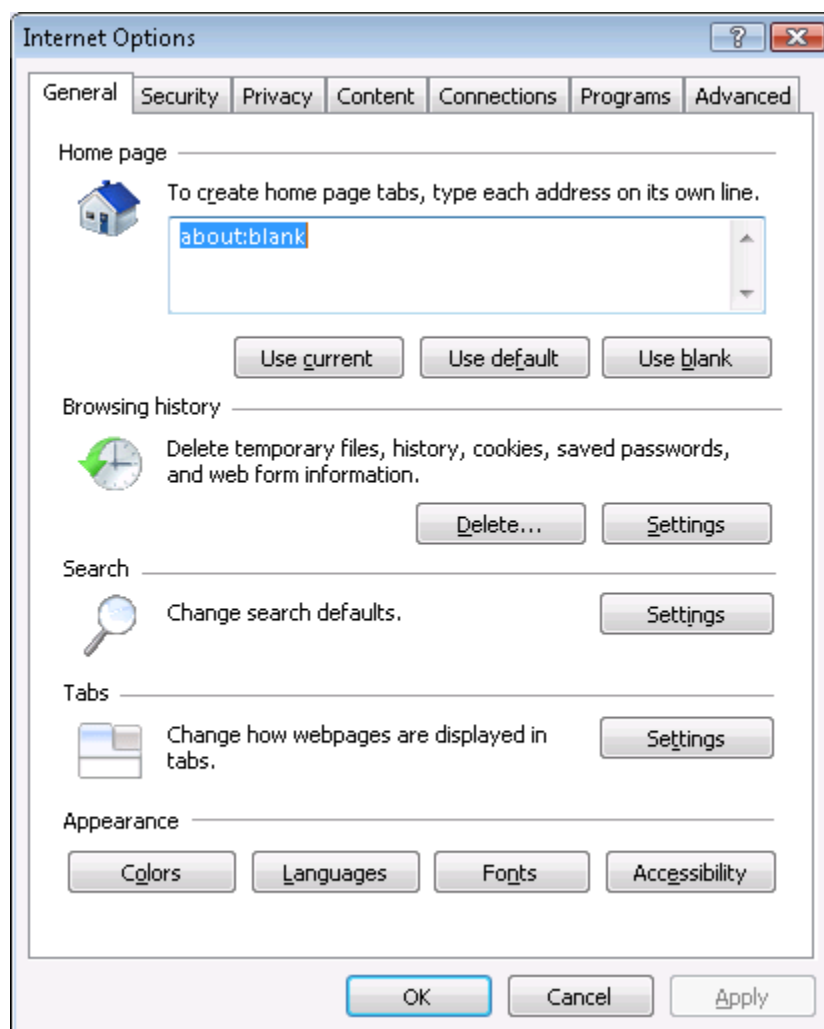
8.1.2.1 Settings for temporary Internet files

Sometimes cached files in Internet Explorer can cause problems when accessing the converted Web application after deployment. In order to avoid the risk of providing *stale* data, it is imperative that Internet Explorer has the cache options set to **Automatically** and that there is sufficient disk space on the Client to enable the Web files to be cached.

To set the cache options:

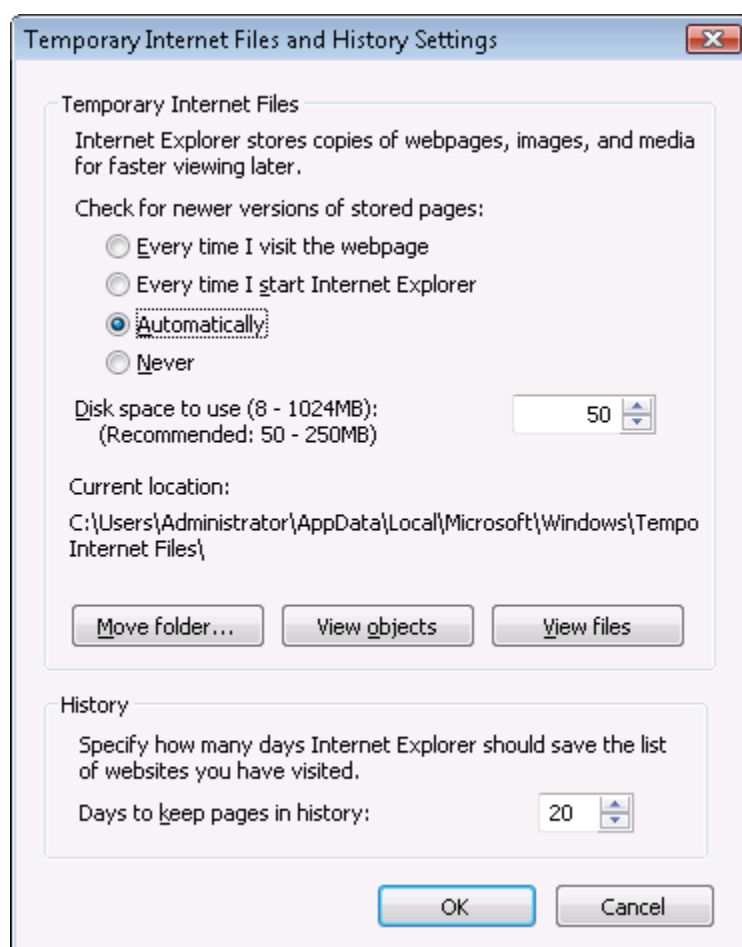
Step 1: Open Internet Explorer and select **Tools > Internet Options**, as shown in the following figure.

Figure 8.1: Internet Options in Internet Explorer



Step 2: Click the **Settings** button in the **Temporary Internet Files** group box.

Step 3: Select the **Automatically** radio button to check for newer versions of stored pages, as shown in the figure below.

Figure 8.2: Temporary Internet File Settings

Step 4: Verify that the **Disk space to use** scroll box is set to at least 200 MB. Click **OK** in the **Settings** window.

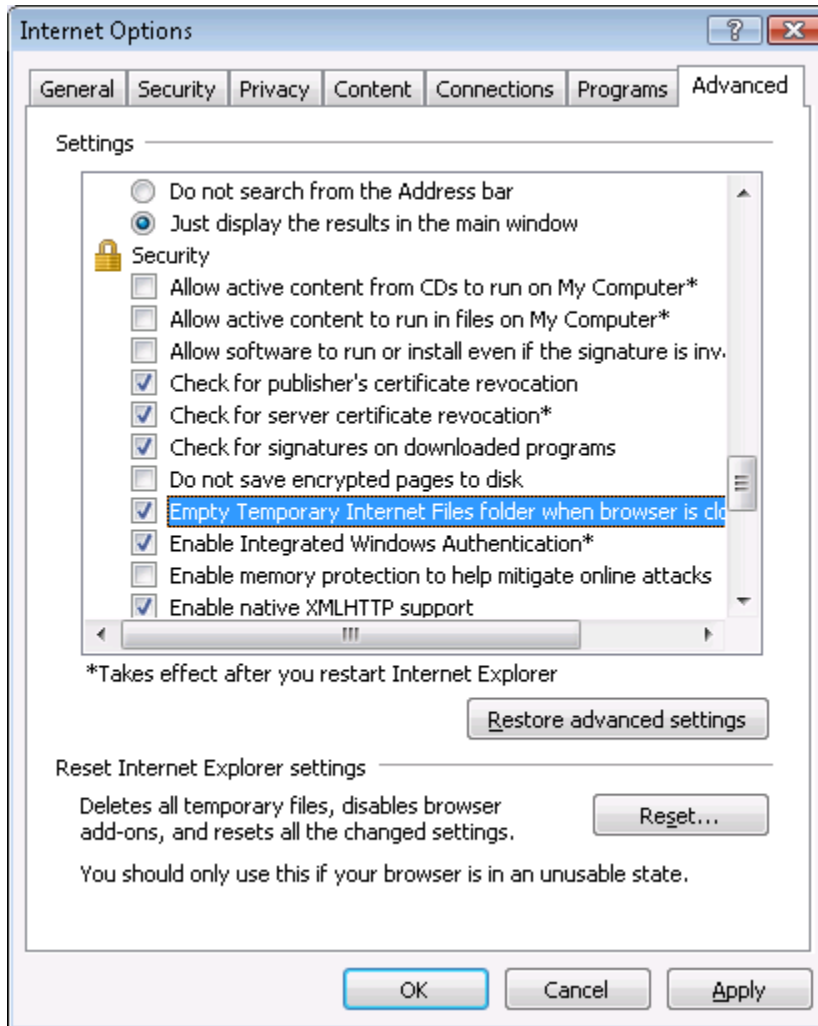
8.1.2.2 Advanced settings for temporary Internet Explorer files folder

During development, the Temporary Internet Files folder should be set to automatically empty each time Internet Explorer is closed. This will force Internet Explorer to re-download the entire application every time it is opened, to ensure the latest version of the application is present, and to prevent interference from outdated cached files.

Here are the necessary steps required to configure Internet Explorer to flush its temporary files each time the browser is closed:

Step 1: Click the **Advanced** tab in the **Internet Options** window, as shown in the following figure.

Step 2: Check the **Empty Temporary Internet files folder when browser is closed** check box under the Security section, as shown in the following figure.

Figure 8.3: Empty Temporary Internet Files folder when browser is closed

Step 3: Click **Apply** in the **Advanced** tab and then **OK** in the **Internet Options** window to allow the settings to take effect.

Allowing the temporary Internet files folder to be emptied each time the browser is closed, is recommended only during the development stage. Once your Web application is ready for production deployment, and frequent changes are not being made to the application, this setting can be disabled once again (unchecked). This setting should be reset once development is complete so the Web application can be cached in each Client PC for better Client-side performance. Using this set of guidelines, caching files should be enabled when they are needed for performance improvement, and disabled during the development and debugging stages.

8.1.2.3 Advanced settings for proxy server

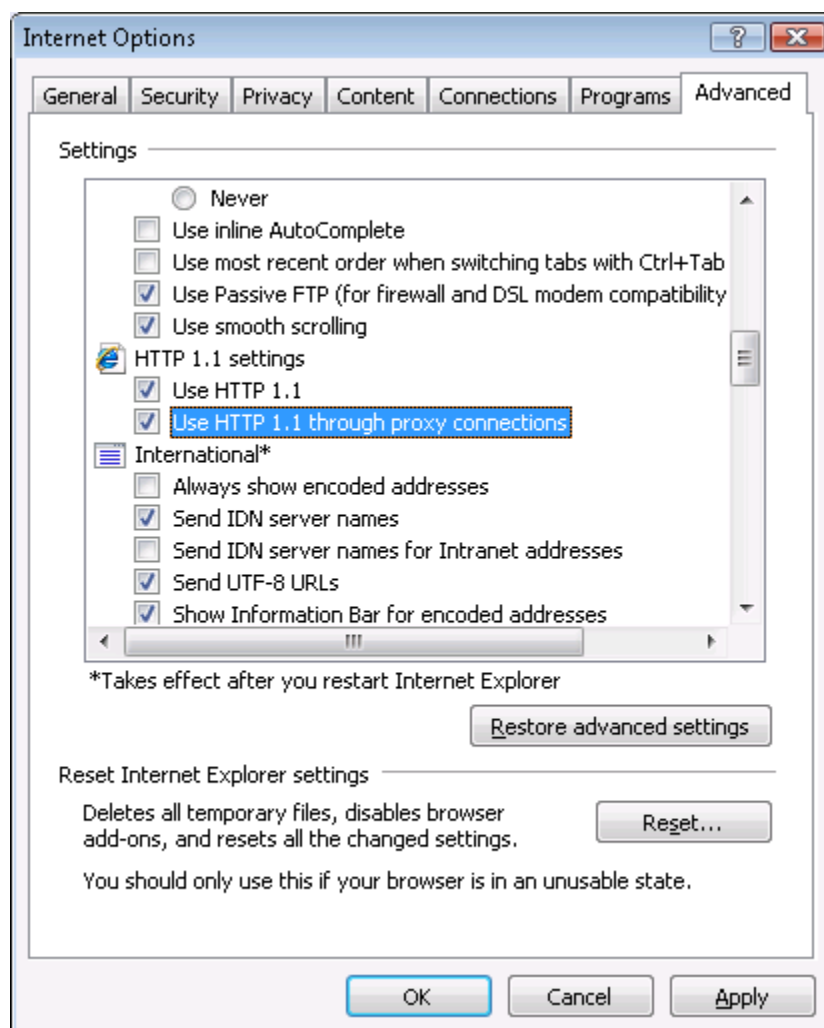
If the Client visits the Web application through a proxy server, you need to select the *Use HTTP 1.1 through proxy connections* option under the HTTP 1.1 settings section, as shown in the following figure.

The following steps detail the configuration needed in Internet Explorer:

Step 1: Click the **Advanced** tab in the **Internet Options** window, as shown in the following figure.

Step 2: Check the **Use HTTP 1.1 through proxy connections** check box under the HTTP 1.1 settings section, as shown in the following figure.

Figure 8.4: Proxy connection settings



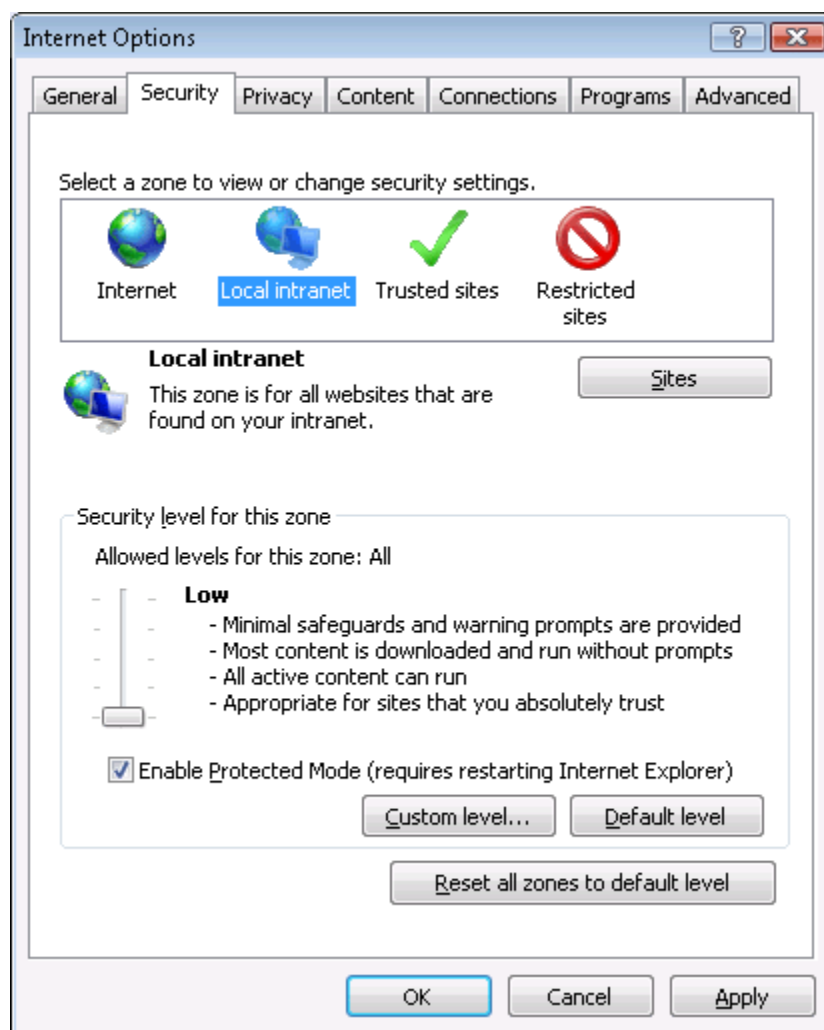
Step 3: Click **Apply** in the **Advanced** tab and then **OK** in the **Internet Options** window to allow the settings to take effect.

8.1.2.4 Security settings for Apeon Xcelerator plug-in download

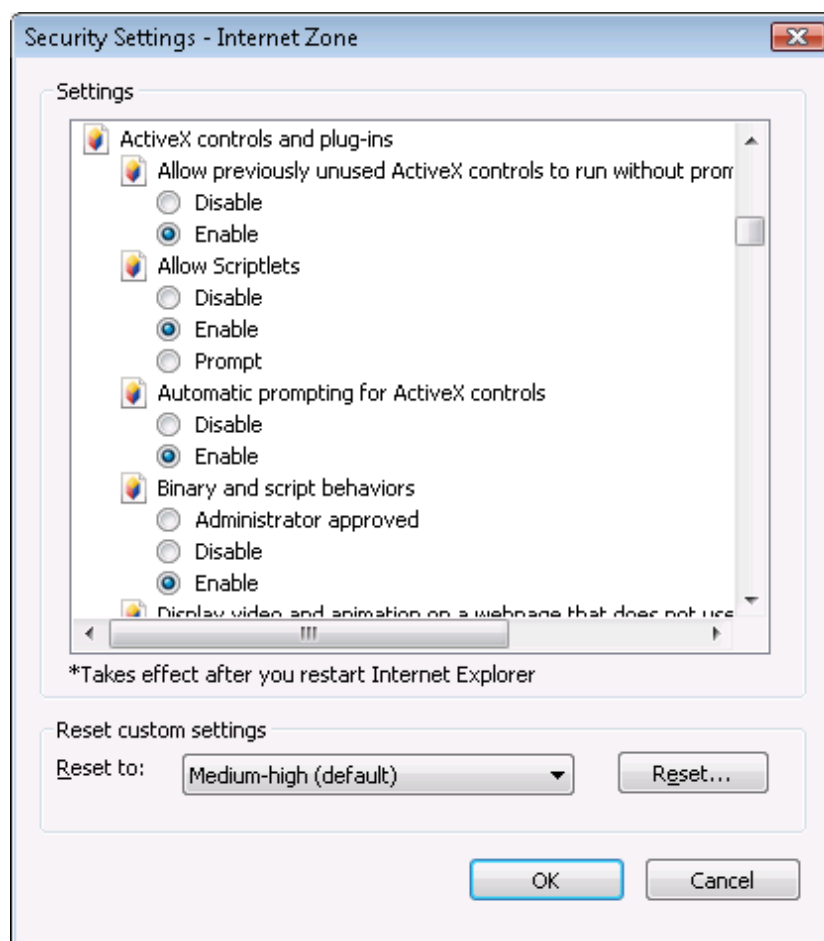
If the Web application calls OLE objects, DLL files, or runs executable programs on the Client, then when the Web application is run for the first time on the Client the user will need to download an ActiveX control (for executing functions that involve client-side integration) and the Apeon Xcelerator plug-in (for improving the runtime performance of Web applications) from the Web Server. To ensure that Internet Explorer will not block the download, configure Internet Explorer's settings, using the following steps:

Step 1: Open Internet Explorer and navigate to **Tools > Internet Options > Security**.

Step 2: Verify that you selected an appropriate Web content zone. Apeon recommends you select the **Trusted sites** zone and add the Web application URL to the **Trusted sites**.

Figure 8.5: Select a Web content zone

Step 3: Click the **Custom Level** button in the **Security** tab. The Security Settings page is displayed, as shown in the following figure.

Figure 8.6: Security Settings page

Step 4: Check the **Enable** or the **Prompt** (recommended) radio button for the following options:

- Download signed ActiveX controls and plug-ins
- Initialize and script ActiveX controls not marked as safe
- Run ActiveX controls and plug-ins
- Script ActiveX controls marked safe for scripting
- File download
- Active scripting

If **Enable** is selected, the ActiveX control will download automatically without prompting. If **Prompt** is selected, your approval will be required before downloading takes place.

8.1.3 Language setting requirements

Note: This is required for the Web application only.

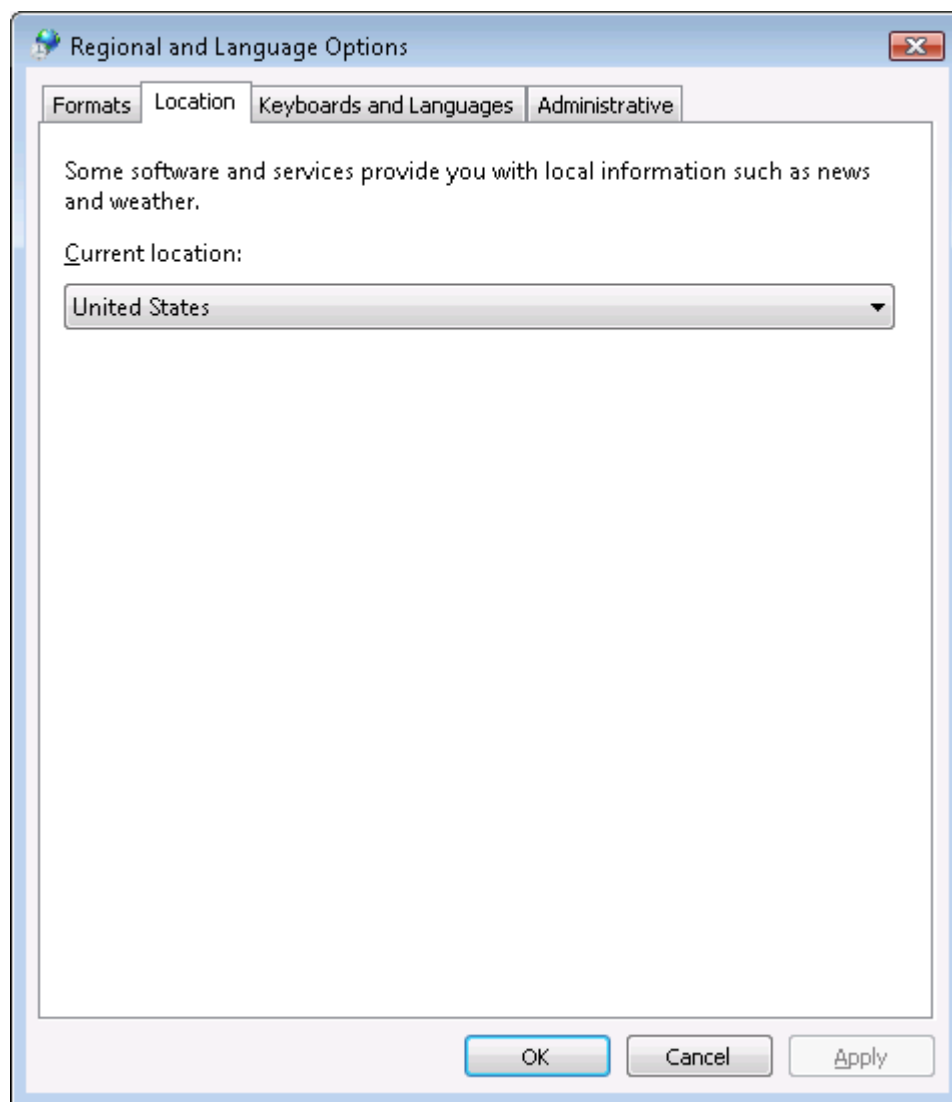
The language of the data in the database used by the application should match the Windows operating system language settings on the machine where PowerServer is installed. If the language settings conflict, problems will occur when the Web application is run.

Verify that the language of the data in the database used by the Apeon-deployed Web application matches the language settings of the client operating system:

Step 1: Select Windows **Start > Settings > Control Panel**.

Step 2: Click **Regional Options**, as shown in the following figure.

Figure 8.7: Regional Options



Step 3: Verify that the language selected in the **Your locale (location)** list box and the language checked as default in the **Language settings for the system** settings are the same. The language can be from different regions as long as it is the same language.

This configuration is not required for English language applications.

8.1.4 Disabling anti pop-up software

Note: This is required for the Web application only.

If anti-popup software (example: Popup Stopper by Panicware) is installed and is active on the Client machine, it will not allow any browser pop-ups to occur. This can cause certain operations to become disabled in the converted Web applications such as the enhanced Web features detailed below, or other operations that automatically load a browser window or popup. To prevent any problems, it is recommended that users disable any anti-popup software while using the converted Web applications.

8.2 URLs of Apeon applications

A typical URL to an Apeon application (both Web and mobile) that is deployed to the PowerServer is similar to this: `http://192.0.0.80:80/sales/`, which normally consists of four or five parts.

It is important to know the application URL, if you or the end user want to run the Web application in the Web browser, or add the mobile application in the Apeon Workspace. If you are not sure of the application URL, you can run the application via the project loader provided by Apeon where the application URL will be automatically input; and remember to remove any index file from the end of the URL to make sure the URL is generic for different environment (see below table for detailed explanation).

Table 8.1: Apeon Application URL

Application URL	Description
Protocol	Uses "http://" typically. If your Web server is configured as an SSL Web server, use "https://".
IP address or domain name	Uses the IP address or domain name of the Web server. Using <i>localhost</i> listeners in a production environment is not recommended.
HTTP port (Optional)	Specifies the port number that your Web server accepts for HTTP connections. If several Web servers are running on the same machine, make sure that the port specified is a port of the PowerServer Web server.
Application name	Specifies the application name. It is the name that you input in the Web Folder field in the application profile configuration window (For more, see Basic Settings).
Index file (Optional)	For mobile apps, make sure to specify no index file at all. For Web apps, make sure to specify no index file, as different index file will be loaded automatically according to the Web browser type and the environment (32-bit or 64-bit). When you send the URL to end users, make sure you remove any index file from the end of the URL, so that the URL is generic for different browsers and environment, for example, <code>http://192.0.0.80:80/sales/</code> is generic, while <code>http://192.0.0.80:80/sales/x32_application.htm</code> is not (it is accessible in 32-bit environment but not in 64-bit environment) and also <code>http://192.0.0.80:80/sales/multi_browser_index.htm</code> is not (it is accessible in FireFox, Chrome and Opera but not in Internet Explorer).

8.3 Running Apeon applications

PowerBuilder application can be deployed as Web application and/or mobile application. Web application can be run in the Web browser (so called browser-based app), or run as a desktop application without needing a browser (so called installable Web app or IWA); and mobile application can be run in the Apeon Workspace on the real mobile device. To help you developers conveniently run the Web application and mobile application in all different ways, Apeon provides a **Run** button in the PowerServer Toolkit; and from this **Run** button, you can easily use different project loaders to run the app.

When you decide to send the application URL to end users, make sure to read the instructions in [URLs of Apeon applications](#) to double check the URL is correct and generic.

8.3.1 Launching applications from the *Run* button

You can launch the application by clicking the **Run** button in the PowerServer Toolkit. PowerServer Toolkit automatically remembers the URLs of the applications that are deployed to the default PowerServer and Web Server.

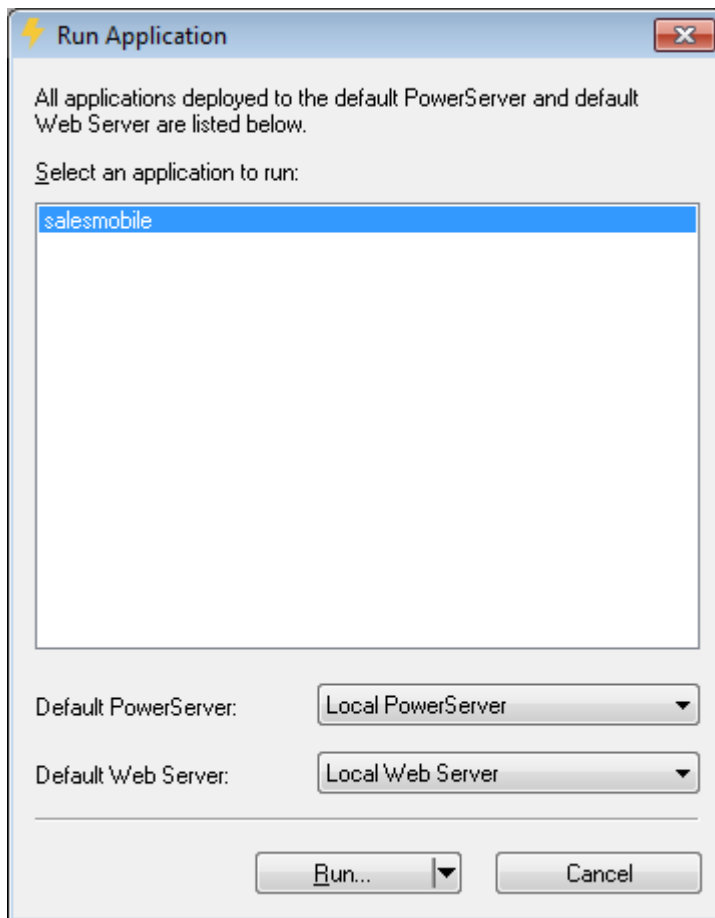
To run an application from the PowerServer Toolkit:

Step 1: Verify that the default PowerServer and Web Server in the Server Profiles settings have been started.

Step 2: Click the **Run** button () on the PowerServer Toolkit. The **Run Application** window is displayed as shown in the following figure.

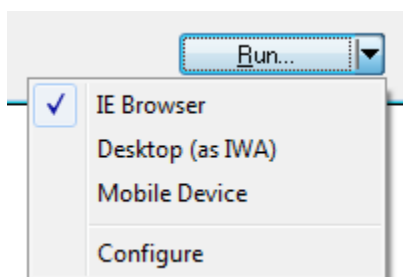
Only applications deployed to the Default PowerServer and Default Web Server are listed in the **Run Application** window.

If you have deployed an application to several different Web folders on the same Web Server, PowerServer Toolkit will only remember the URL used in the last deployment, although the URLs to access the other folders used in the previous deployments will also be valid.

Figure 8.8: Run Application window

Step 3: Select an application from the application list.

Step 4: From the **Run** dropdown list, select a project loader for the application to run. If you directly click the **Run** button, the last used project loader will be launched to run the application.

Figure 8.9: Run dropdown list

Whether the application loaders are selectable depends on the project type the app is deployed as:

- For applications that are deployed as **Both** project type, all including **IE Browser**, **Desktop (as IWA)**, & **Mobile Device** are the selectable project loaders.
 - **IE Browser**: runs the Web application in the Internet Explorer Web browser.

- **Desktop (as IWA):** installs the Web application as a desktop app on the current machine and then you can run it directly without needing any browser.

For more about how to install an IWA (Installable Web App), see [Installing IWA apps](#).

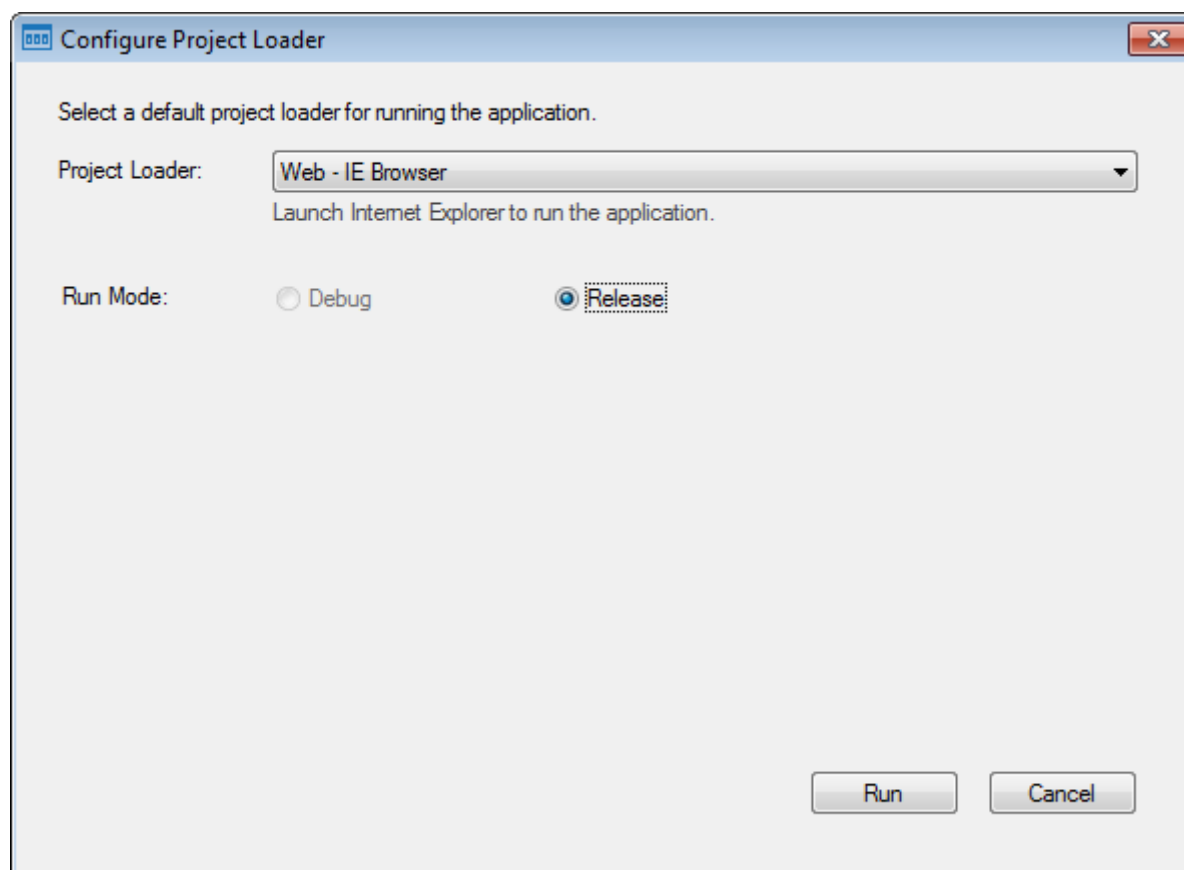
- **Mobile Device:** runs the mobile application in the Apeon Workspace on the real mobile device. When you select this project loader, you will be given guidelines for how to install the Apeon Workspace and the mobile app on the mobile device. For step-by-step instructions, see [Installing Apeon Workspace and mobile apps on mobile device](#).
- For applications that are deployed as **Web** project type, only **IE Browser** and **Desktop (as IWA)** are the selectable project loaders.
- For applications that are deployed as **Mobile** project type, **Mobile Device** is the only selectable project loader.

8.3.1.1 Configuring the project loader

You can select the **Configure** option under the **Run** dropdown list to configure the project loader for the selected application.

For the Web application that you want to run in the IE browser, if you have deployed the application in both the **Release** and **Debug** mode, you will be given a choice to select at which mode the application will run.

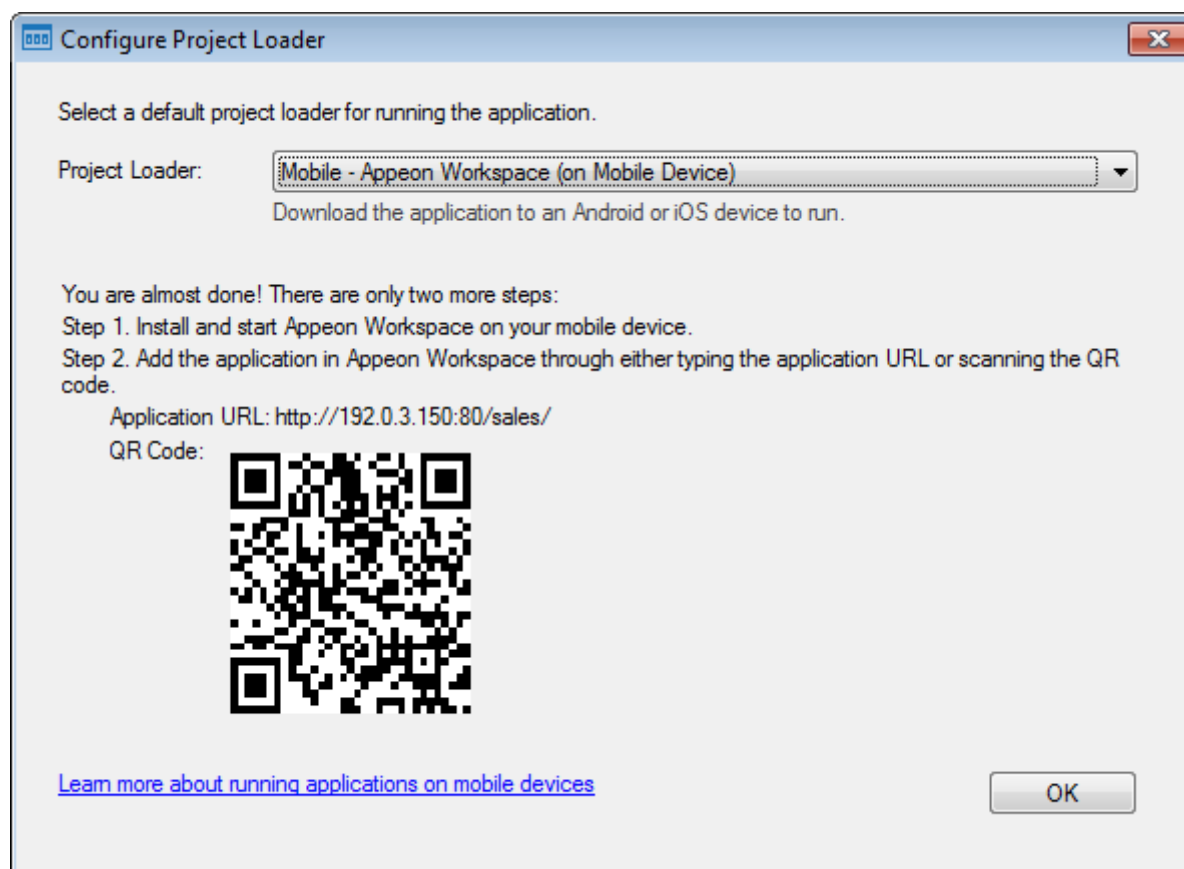
Figure 8.10: Configure for project loader - IE Browser



For the mobile application that you want to run in the Appeon Workspace on the real mobile device, you can follow the instructions on this screen to install Appeon Workspace on the mobile device and then install the mobile app in the Appeon Workspace.

The URL and QR code for the selected application is automatically generated. You can directly use these information to add the application in Appeon Workspace.

Figure 8.11: Instructions for running the app in Appeon Workspace



8.3.2 Installing Appeon Workspace and mobile apps on mobile device

Install Appeon Workspace on iOS devices:

Step 1: Make sure your iOS device can connect to Internet.

Step 2: On your iOS device, open **App Store**, search for "Appeon Workspace", and then tap **Install**.

Alternatively, you can download Appeon Workspace from the Apple App Store to your Windows PC via IE browser, and then synchronize it to your iOS device via iTunes.

Note that the Appeon Workspace installed from App Store only works with the latest PowerServer Mobile on the market. If you have installed an earlier version of PowerServer Mobile, you will need to create and install a customized version of Appeon Workspace; and if you intend to distribute your own Appeon Workspace on Apple App Store, then refer to [Tutorial: Package & Distribute Appeon Workspace to App Store](#); if you want to host Appeon Workspace on your own Web site, then refer to Section 5.2, "Package & Distribute iOS Apps" in *PowerServer Mobile Tutorials*.

For more detailed information about installing Appeon Workspace, refer to Section 3.2, “Installing and upgrading Appeon Workspace” in *Appeon Workspace User Guide*.

Install Appeon Workspace on Android devices:

Step 1: On the Android device, enable the **Unknown resources** option (in **Settings** > **Security**), so you can install apps that are not downloaded from Google Play.

Step 2: Make sure your Android device can connect with PowerServer.

Step 3: On your Android device, access the Appeon Workspace download center that is posted in PowerServer (http://server_ip:port/aws) and then click the download icon to download and install Appeon Workspace.

Install apps in Appeon Workspace:

Launch the Appeon Workspace and add the application URL using one of these methods:

- Manually input the application URL: http://server_ip:port/app_name, or
- Scan the QR code of the application URL.

Double check that your mobile device can connect with PowerServer, so you can download and install the apps successfully.

For more detailed information about installing apps in Appeon Workspace, refer to Section 4.1, “Adding applications” in *Appeon Workspace User Guide*.

8.3.3 Installing IWA apps

IWA is short for installable Web apps. As its name suggests, it is also a Web application, and it is configured and deployed in the exact same way as the other Appeon Web application that runs in the Web browser (browser-based apps). The only difference is IWA app is installed as a desktop app on the client, and it can be launched directly from the desktop.

To run an IWA app on the client machine, you will need to install it first. You will need to append **"iwarunner.html"** at the end of the application URL in a Web browser, for example, <http://192.0.3.150:80/sales/iwarunner.html>. The Web browser will download the IWA Runner setup wizard (`appeoniwarunner.exe`) if IWA Runner is not installed before; after download is complete, you should run the setup wizard to install the IWA Runner; and after that the Web app will be automatically installed as an IWA app on the desktop.

To run an IWA app with arguments, append the argument to the end of the URL: <http://192.0.3.150:80/sales/iwarunner.html?parm=test>. The app shortcut will be created with this URL: <http://192.0.3.150:80/sales/?parm=test> (be sure to keep the "/" right after the app name in the URL if you want to manually change this URL).

Compared to the browser-based app, IWA app may have a slightly different look and feel for some UI controls, because IWA apps and browser-based apps use different frameworks to implement UI controls.

About IWA Runner

IWA Runner can be considered as a lightweight Web browser that supports running the Appeon IWA apps only. It must be installed on the client, for only one time. User will be prompted to upgrade IWA Runner if it is updated.

IWA Runner can support IWA apps of different versions (such as Version 2017, Version 2017) and it is backwards compatible after upgraded.

IWA Runner can only be run by one Windows user at a time, which means, multiple Windows users on the same client machine cannot run the IWA app at the same time.

IWA Runner is a 32-bit program that can run on both 32-bit and 64-bit OS, however, it cannot call 64-bit DLLs/OCXs on 64-bit OS.

If any issue installing or running the IWA app, refer to Section 4.5, “IWA issues” in *PowerServer Troubleshooting Guide*.

8.3.3.1 Configuring IWA Runner

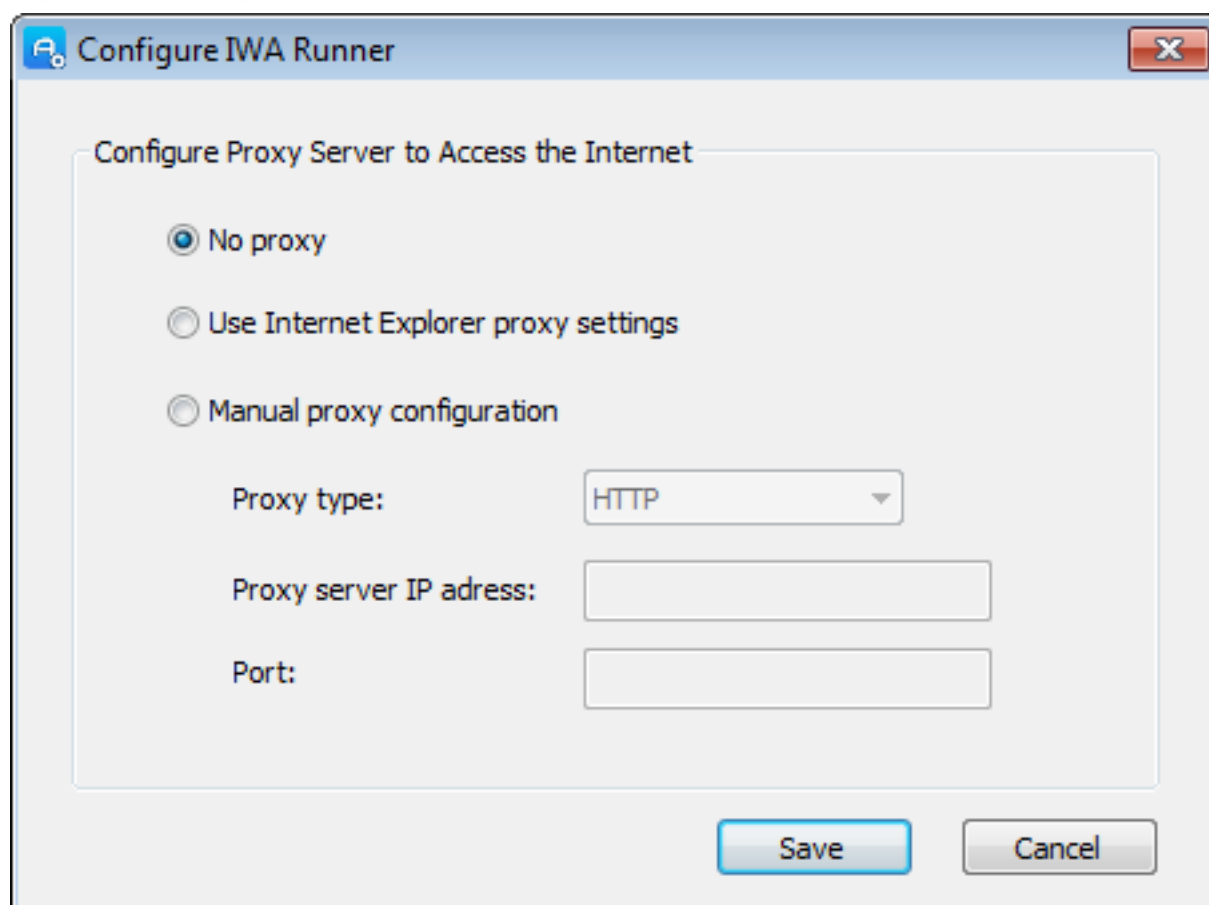
The end user can configure IWA Runner to connect with the PowerServer through a proxy server by the following steps:

Step 1: Select Windows **Start > All Programs > Appeon IWA > Configure IWA Runner**.

Step 2: Configure to use or not to use a proxy server.

To configure to use a proxy server, you can choose to directly use the Internet Explorer proxy settings, or configure the proxy settings manually.

Figure 8.12: Configure IWA Runner



8.3.4 Selecting a run mode for the Web app

Note: This is for the Web application only.

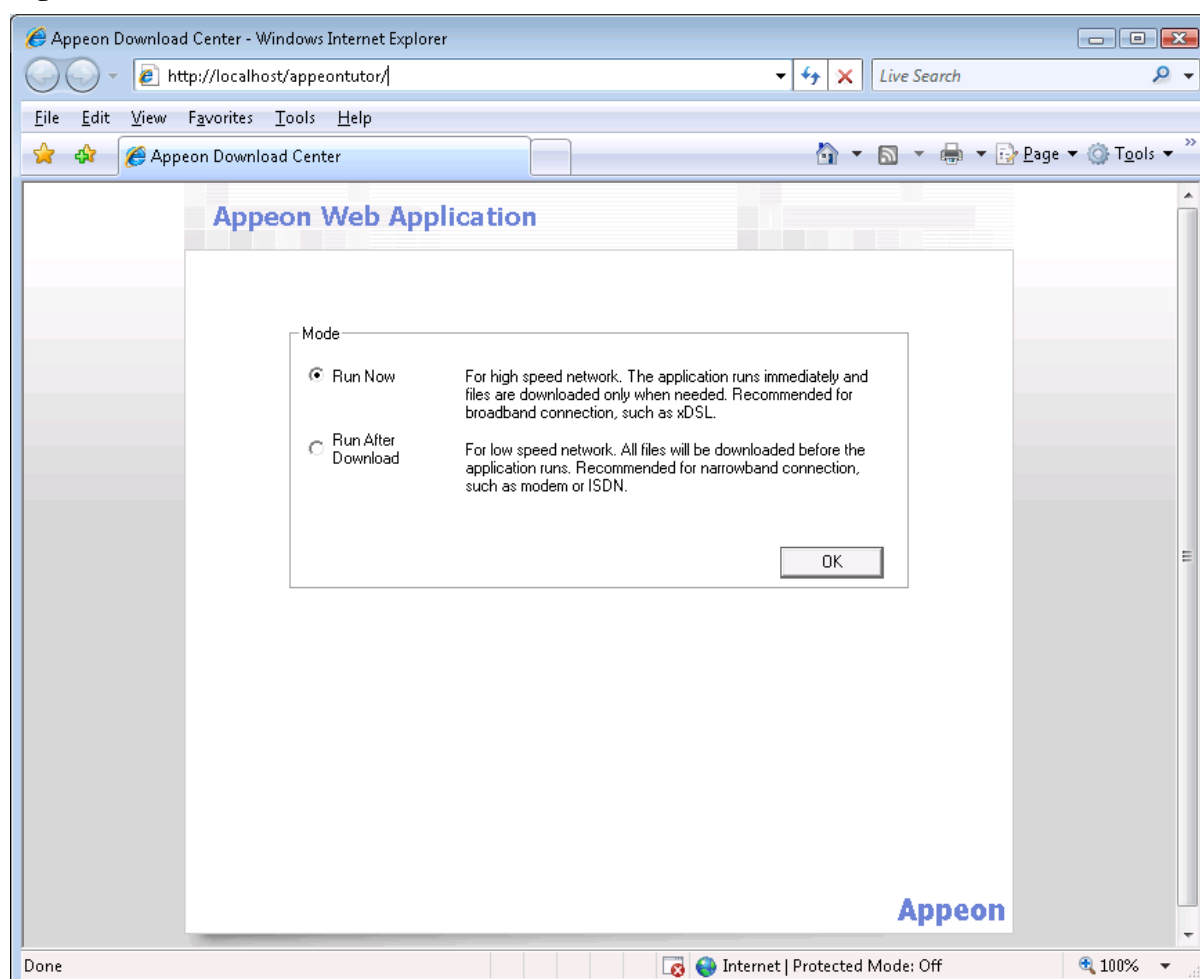
When you run a Web application in the Web browser for the first time, a run mode selection page will be displayed, if you have enabled the **Allow user to select run mode** option in AEM (see PowerServer Configuration Guide for .NET or PowerServer Configuration Guide for J2EE for details).

This page allows you to select from the following run modes (as shown in the following figure):

- **Run Now:** The application runs immediately and files will be downloaded only when used. This mode is recommended for high speed network.
- **Run After Download:** Files will be downloaded before the application runs. This mode is recommended for low speed network.

You can also set a run mode to be the default mode in AEM (see PowerServer Configuration Guide for .NET or PowerServer Configuration Guide for J2EE for details).

Figure 8.13: Run Mode window



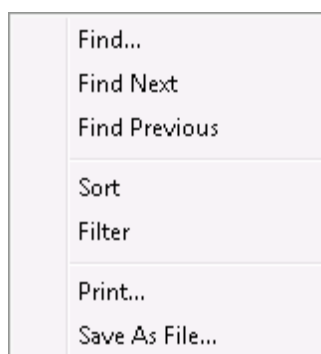
8.4 Appeon DataWindow menu

Note: This section is for Web applications only.

Appeon provides a Web enhancement, the Appeon DataWindow menu, for all DataWindows deployed to the Web. When right-clicking on a deployed DataWindow, you can access the

Appeon DataWindow popup menus: Find, Find Next, Find Previous, Sort, Filter, Print, and Save As File (Obsolete), as shown in the following figure.

Figure 8.14: Appeon DataWindow popup menu



8.4.1 Enabling Appeon DataWindow menu

To use the Appeon DataWindow menu, you must enable it by using the following two functions: *appeonextfuncs.of_popmenu* and *appeonextfuncs.of_popmenuon* that are defined in the Appeon Workarounds PBL. For detailed instructions on how to use these functions in a DataWindow, refer to Section 2.3.1, “AppeonExtFuncs Object” in *Workarounds & API Guide*.

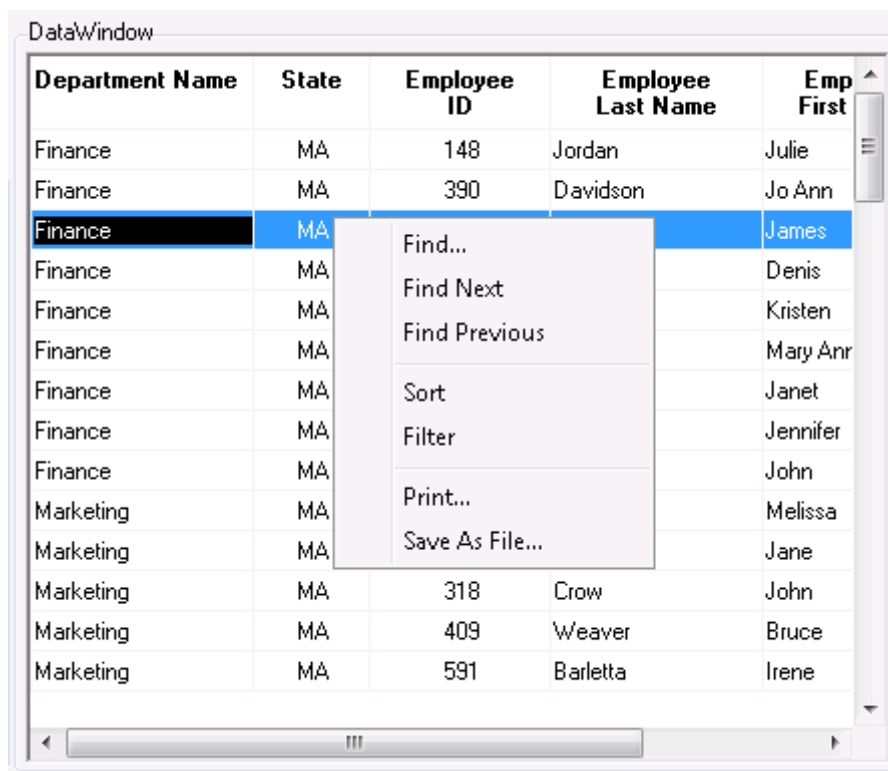
8.4.2 Using Appeon DataWindow Menu

Appeon DataWindows menu have the following features: Find, Find Next, Find Previous, Sort, Filter, Print, and Save As File (Obsolete). Perform the following steps to use the menu in the Web DataWindows.

8.4.2.1 Find

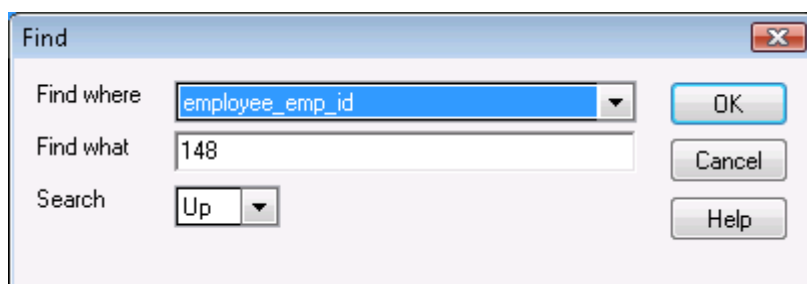
The user can search for data using the Find, Find Next and Find Previous functions.

Step 1: Right-click the Web DataWindow and select **Find** from the popup menu, as shown in the following figure.

Figure 8.15: Find

Step 2: Select the column and search scope and input criteria for the search.

Use the Find function to find all rows where the orders count column is equal to 2, as shown in the following figure.

Figure 8.16: Find

Step 3: The first row that matches the search criteria will be highlighted.

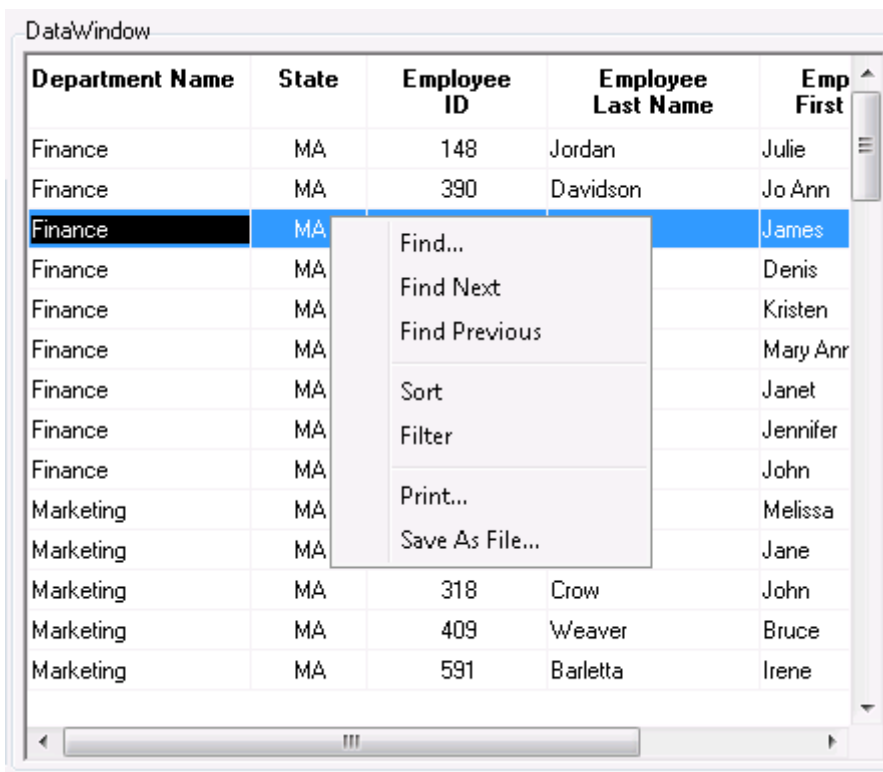
Step 4: Right-click and select **Find Previous** or **Find Next** to highlight the rows (one at a time) that match the criteria.

8.4.2.2 Sort and filter

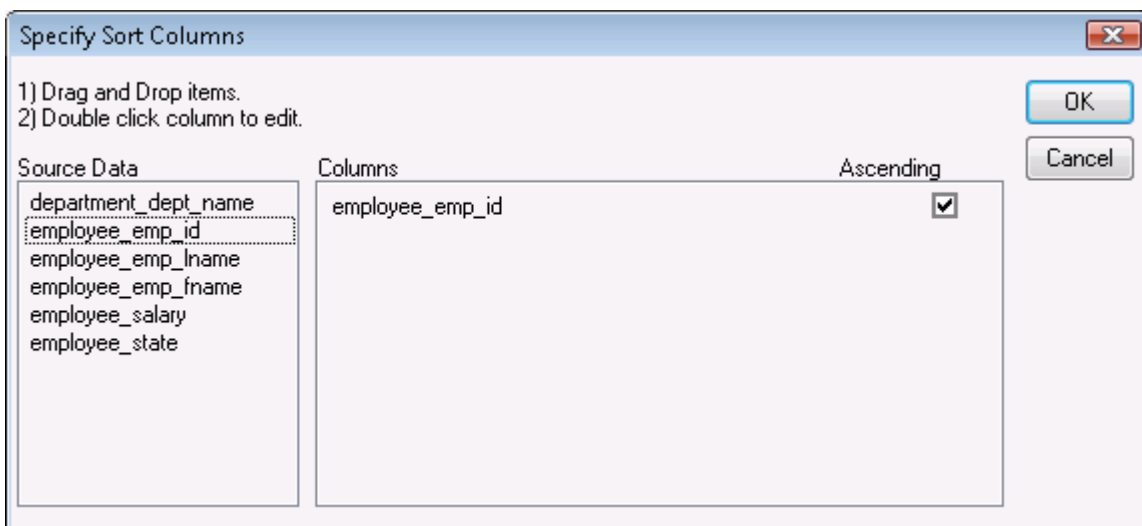
The user can sort data by the column in ascending or descending order, or filter data using a number of functions.

To sort data:

Step 1: Right-click the Web DataWindow and select **Sort** from the popup menu, as shown in the following figure.

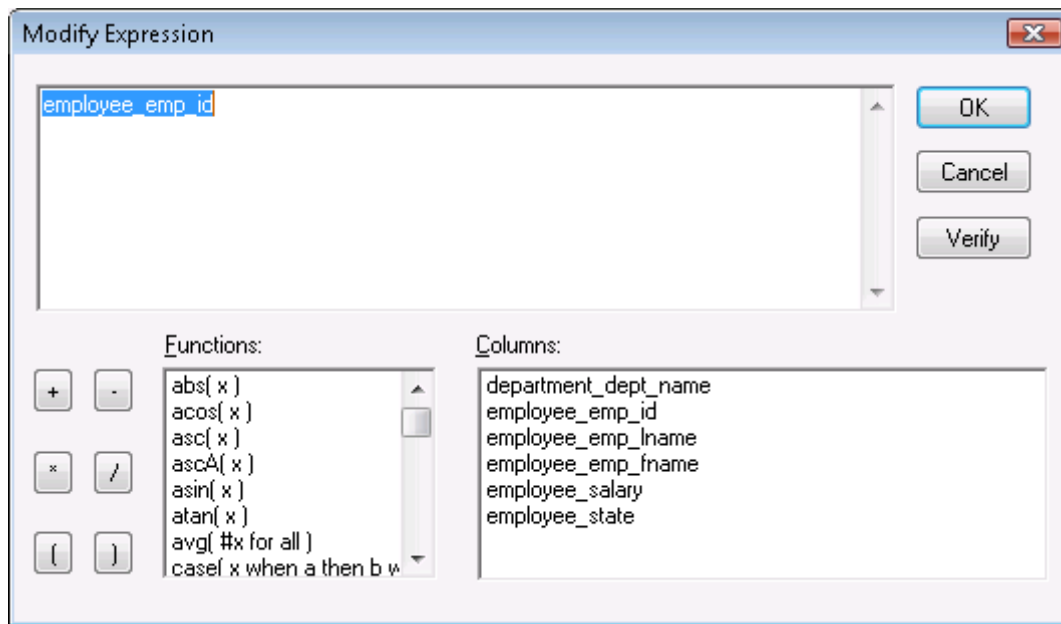
Figure 8.17: Sort

Step 2: Specify the column used to sort the data in the DataWindow by clicking on the column name in the left list box. The column name appears on the right, as shown in the following figure.

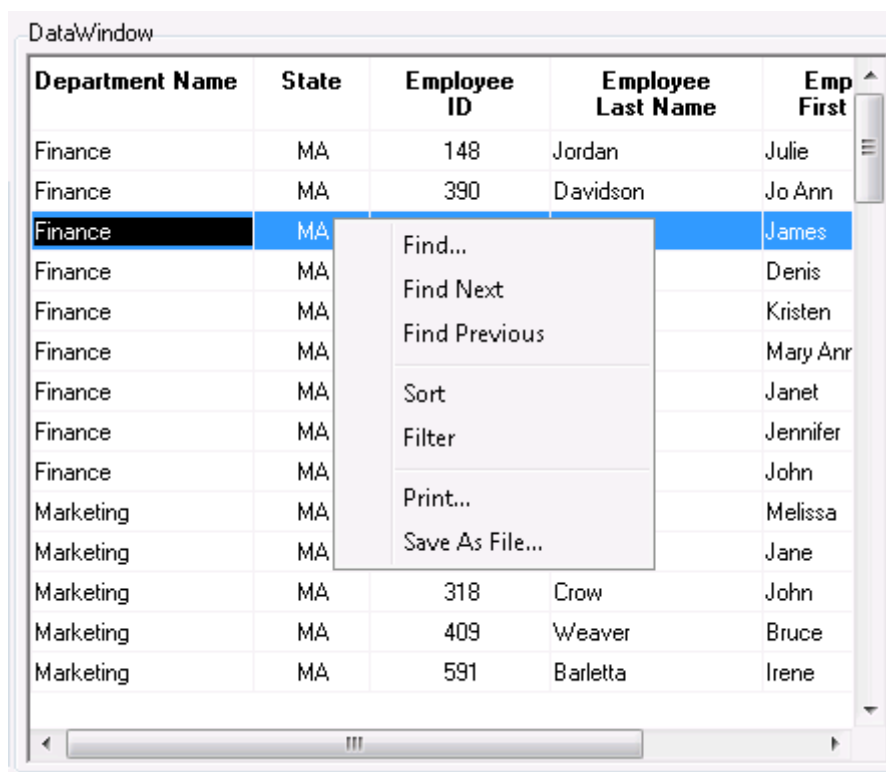
Figure 8.18: Specify sort column

Step 3: Check the **Ascending** check box to sort the data in ascending order or uncheck the **Ascending** check box to sort the data in descending order. Click **OK** to accept the changes. The DataWindow automatically refreshes and the data is sorted.

If you want to edit the selected column, double click it to open the "Modify Expression" dialog, as shown in the following figure.

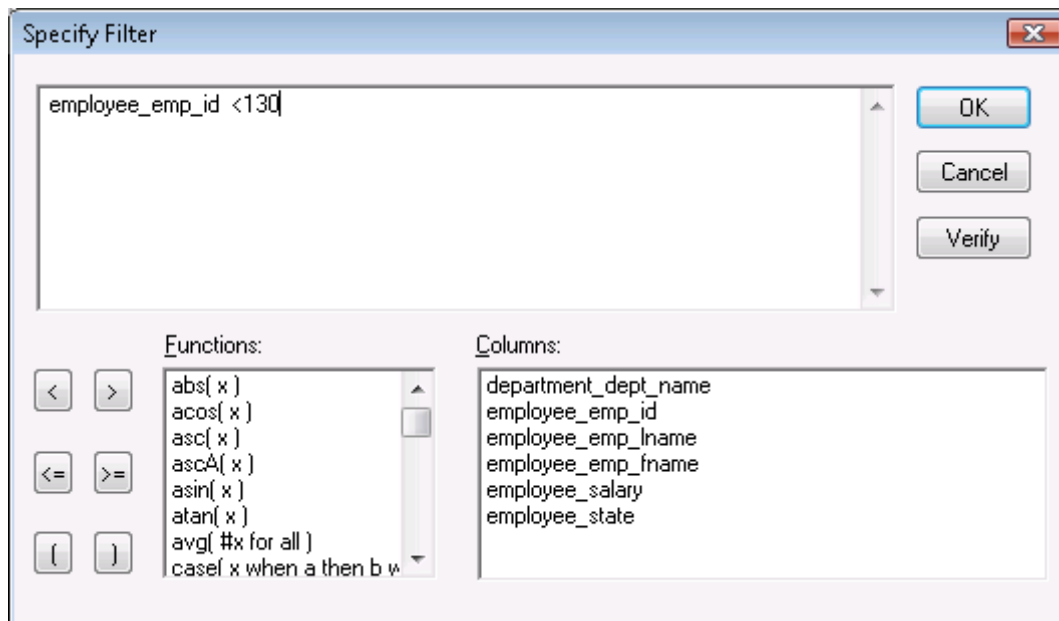
Figure 8.19: Modify Expression dialog**To filter data:**

Step 1: Right-click the Web DataWindow and select *Filter* from the popup menu, as shown in the following figure.

Figure 8.20: Filter

Step 2: Specify the filter by selecting the functions and columns. Click **Verify** to test whether the filter expression is valid. Click **OK** to accept the filter, as shown in the following figure. The DataWindow automatically refreshes and the data is filtered.

Figure 8.21: Specify Filter

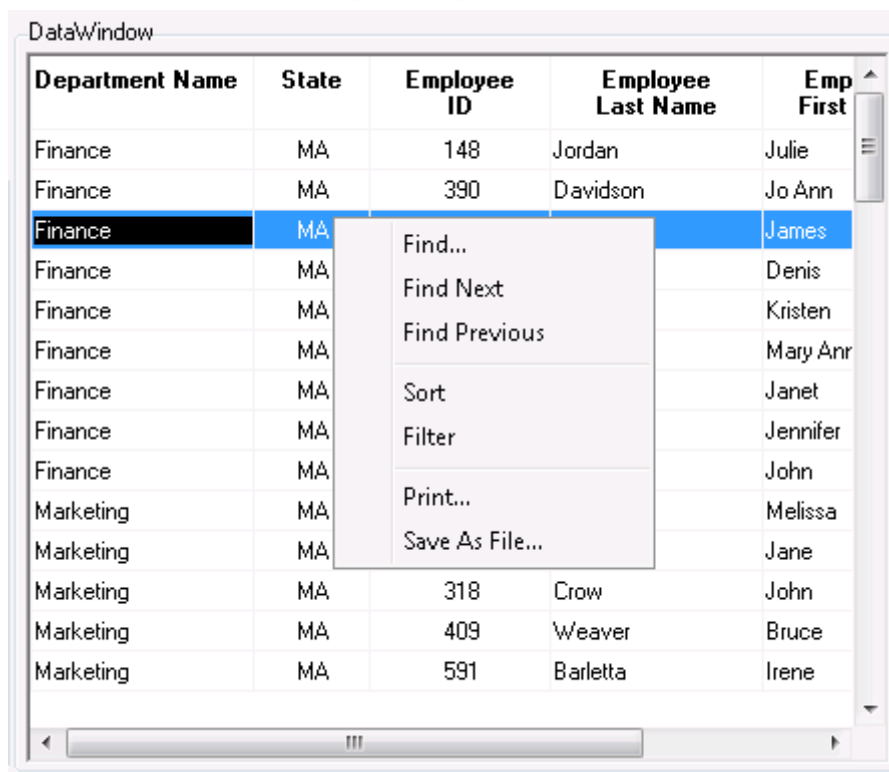


8.4.2.3 DataWindow printing

The Print menu provides user ability to print DataWindows directly on printers connected to the Client.

Step 1: Right-click the DataWindow and select **Print** from the popup menu, as shown in the following figure.

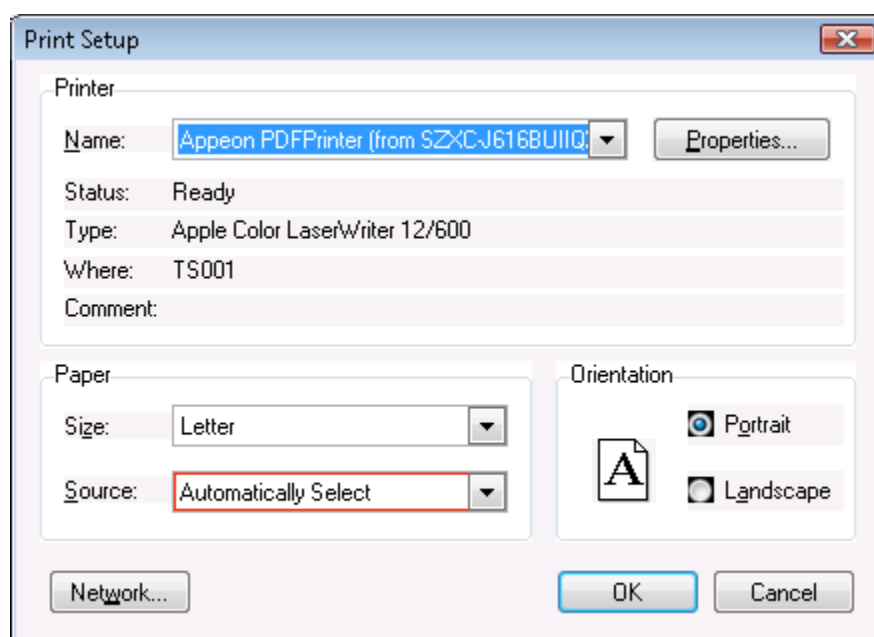
Figure 8.22: Print file with physical printers



Step 2: Specify the printer settings and paper settings in the **Print Setup** window, as shown in the following figure. Click **OK**.

The DataWindow will be printed directly to the local printer.

Figure 8.23: Print setup window



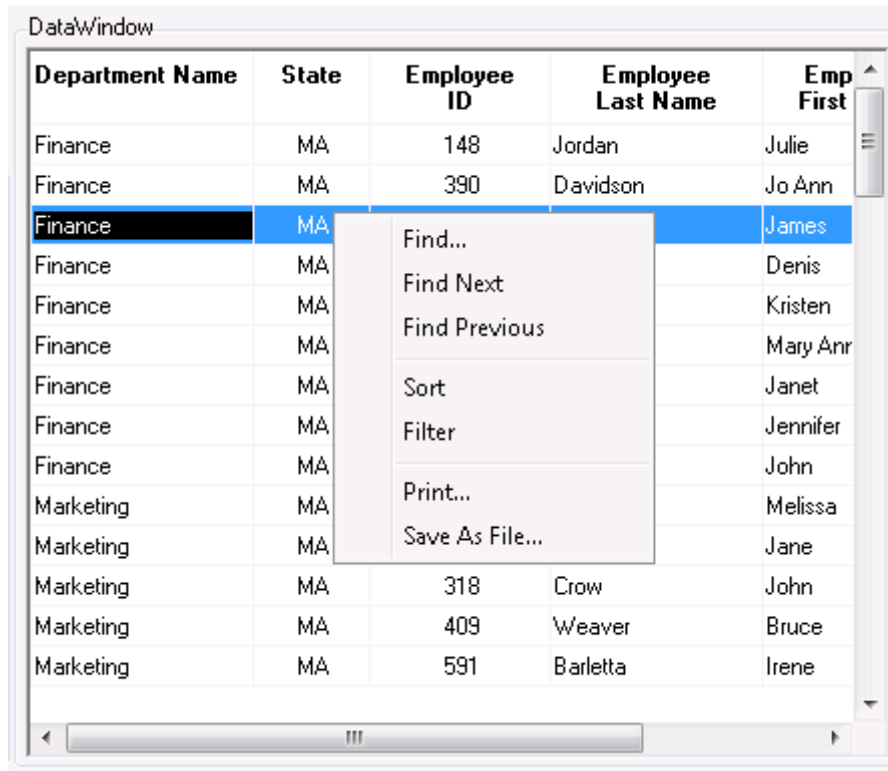
8.4.2.4 SaveAs

This menu is only available in PowerServer Web 6.5.1 or prior. In PowerServer Web 6.6 or later, you can directly call the DataWindow SaveAs function.

The user can save the contents of a DataWindow in different file formats.

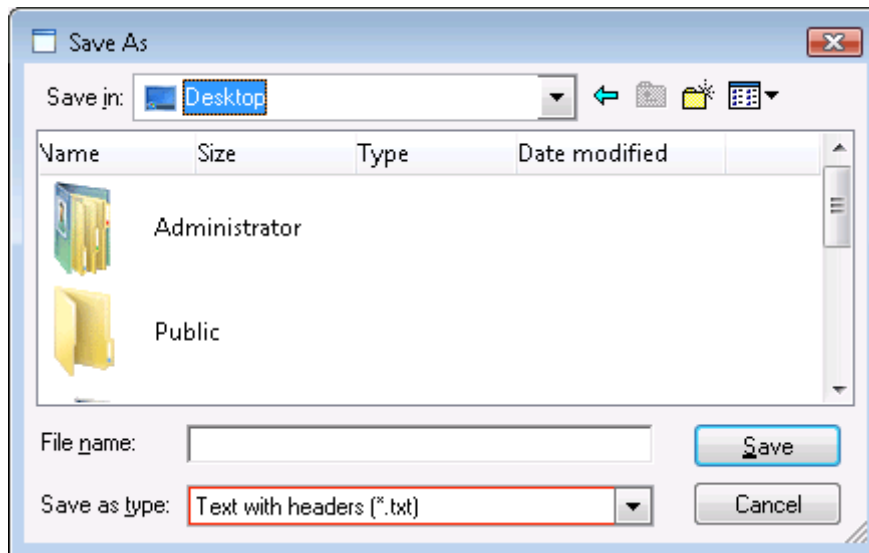
Step 1: Right-click the DataWindow and select **Save As File** from the popup menu, as shown in the following figure.

Figure 8.24: Save As File



Step 2: Specify the file location, file name and file type in the Save As window, as shown in the following figure.

Figure 8.25: Save As window



Note:

If the DataWindow without any data is saved as the HTML file or the XML file, on the Web the DataWindow header will not be saved, whereas it will be saved on PB.

8.4.2.5 Additional Enhanced Features

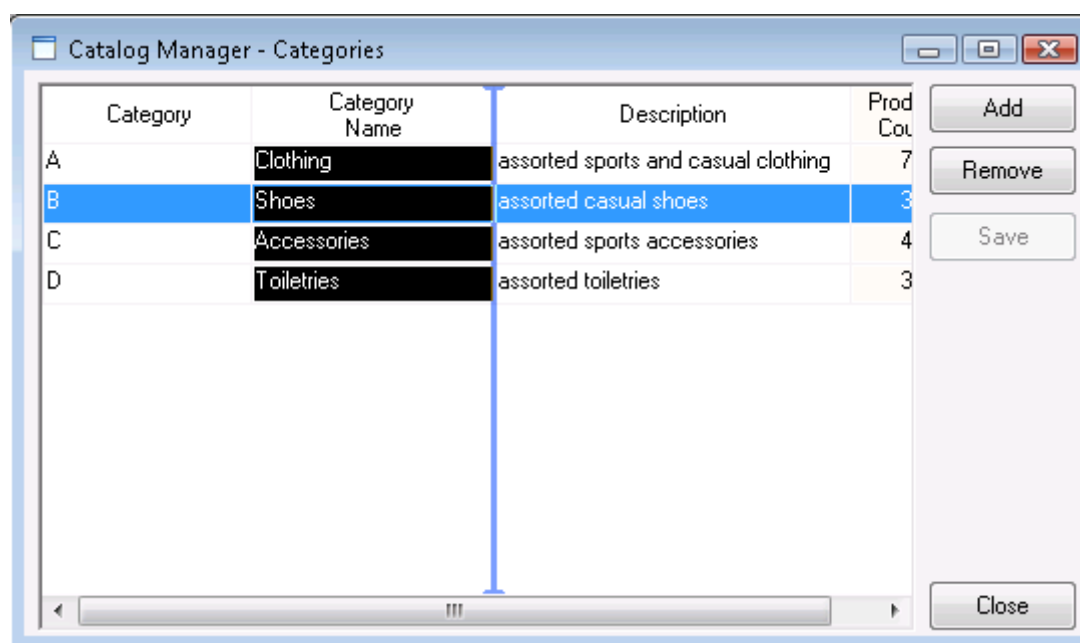
In addition to the popup menu (Find, Find Next, Find Previous, Sort, Filter, Print, Save As File), Apeon offers more features for users to manage DataWindows on the Web. Users

can place DataWindow columns in a different order by dragging and dropping them to new positions. When navigating an editable DataWindow, users can use the popup menu to quickly edit the contents. These features are always available and require no extra coding to be enabled. The following steps show you how to access them:

Step 1: Select a column header in the Web DataWindow and the selected location is highlighted, as shown in the following figure.

Step 2: Drag and drop it to the desired location, and the location to be dropped is highlighted as well.

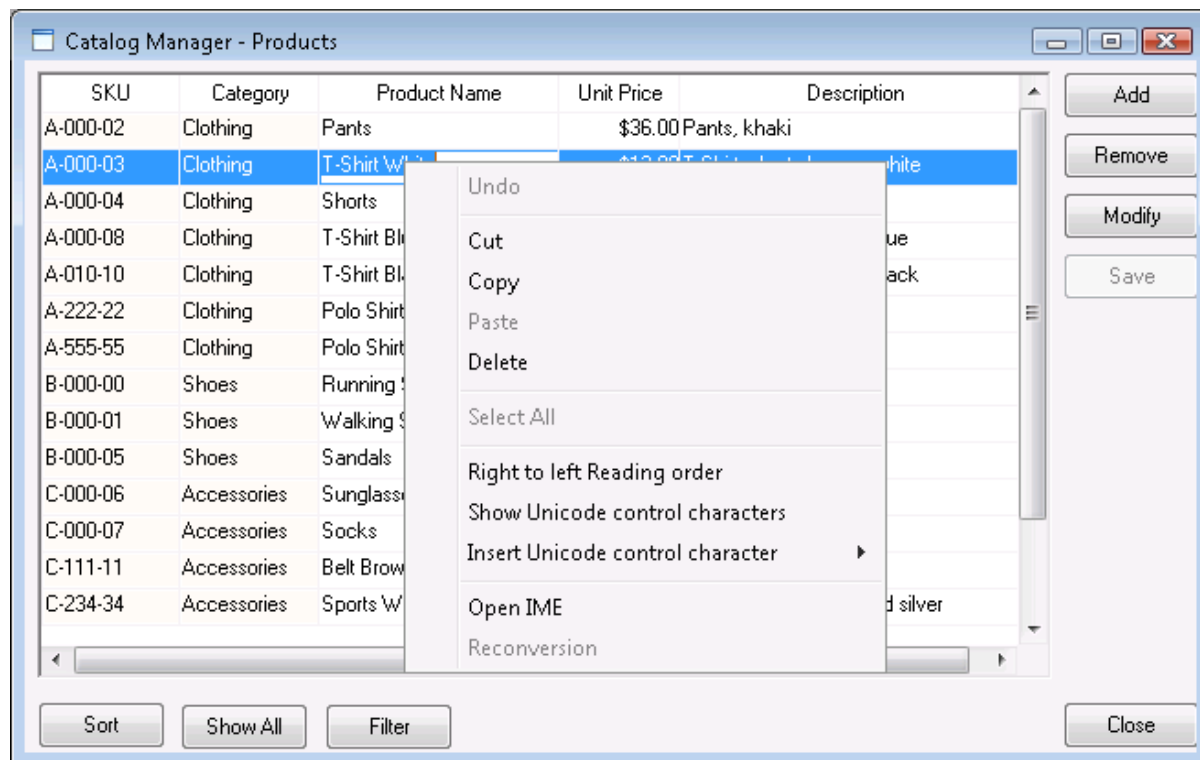
Figure 8.26: Select a column header



Step 3: Right click on an item in an editable HTML DataWindow and a menu pops up, as shown in the following figure.

The user can select the following menu items from the popup menu to perform quick editing: Undo, Cut, Copy, Paste, Delete, Select All.

Figure 8.27: Popup menu for quick edition



9 Using Information Manager

Information Manager provides easy access to all the available reports or log files generated during the application migration process, including debug, development, and deployment. The reports are categorized as follows: Analysis. The log files are categorized as follows: Deploy Log and Analysis Log.

The following table provides a brief description of all the reports generated during the Web or mobile migration process and the report types they are associated with.

Table 9.1: Report Description

Report Type	Description
Analysis	<p>Analysis Report is generated during a feature analysis or application deployment, and records the unsupported features in the application.</p> <p>After you click this button, the UFA Report Window will be displayed. For detailed instructions, refer to Section 5.2, “Working with UFA Report”.</p>

The following table provides a brief description of all the log files generated during the Web or mobile migration process, and the log types they are associated with.

Table 9.2: Log Type Description

Log Type	Description
Deploy Log	<p>Deploy Log is generated during application deployment, and records the deployment process status.</p> <p>All log files are named according to the following format:</p> <p>Log Type (i.e. <i>DeployWizardLog</i>, or <i>FeatureAnalysisLog</i>) + time when the log file is generated (yyyy/mm/dd/hh/mm/ss) + xml.</p> <p>For example, <i>DeployWizardLog_20050220101918.xml</i>.</p>
Analysis Log	<p>Analysis Log is generated during the feature analysis process, and records the analyzing status.</p>

9.1 Viewing the reports and logs


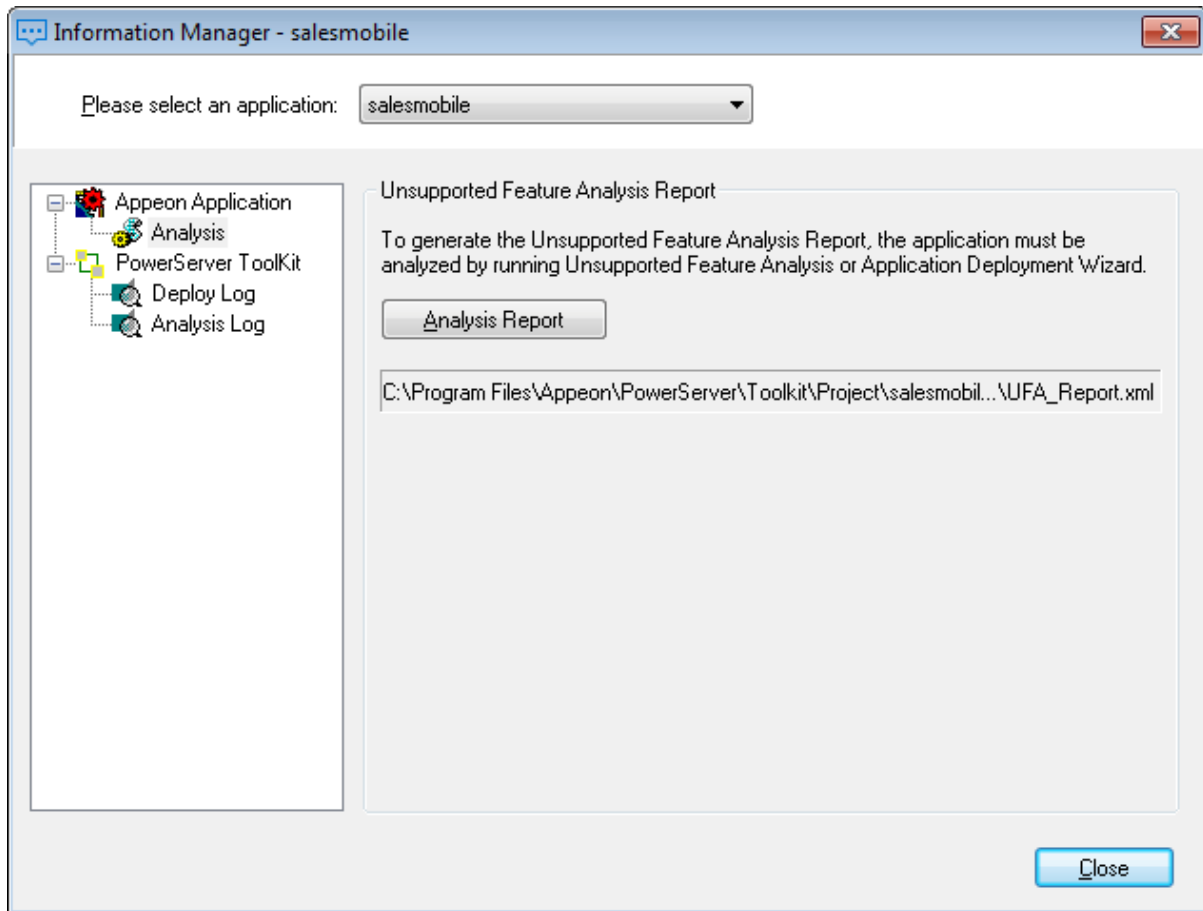
Step 1: Click the **Information** button () on the PowerServer Toolkit and the **Information Manager** is displayed, as shown in the following figure.

Figure 9.1: Information Manager

Step 2: Select the application in the top dropdown listbox.

Step 3: Select a report type or log type in the left box and the corresponding report button or log files are displayed in the right box.

If there is more than one log file generated, the log files are listed in order according to the date and time of generation; the latest log will be listed at the top.

You can open one log at a time, or open several logs at one time: select a continuous range of log files by holding down Shift and click log files, or select a disconnected range of log files by holding down Ctrl and clicking log files. Then click **View** to open the selected log files.

The **View** and **Delete** buttons will be disabled if no log files were generated.

Note: For Deploy Log, you can click **View All** to view all information such as errors, warnings, and process description, or click **View Warning** to view warnings only, or click **View Errors** to view errors only.

Step 4: Click the report button or log file to view the report or log.

You can continue with other operations while the Information Manager opens. For example, you can perform feature analysis, deploy the application, or run the Web or mobile application. If any new reports or logs are generated when the Information Manager is still open, the new reports or logs are not reflected in the Information Manager immediately. You need to refresh Information Manager in either of the following ways:

- Close the Information Manager and open it again, or

- Select a different application and then select the original application again in the Information Manager.

10 Packaging Applications

The Appeon Application Package Wizard helps PowerBuilder developers to

- Generate portable server deployment packages for Web and/or mobile applications, so the user can directly deploy the Web and/or mobile application to servers without installing PowerServer Toolkit and using it to do Web or mobile deployments. The PowerBuilder developers do not need to provide the source PowerBuilder application to the user; this protects the author's intellectual property rights. Refer to [Packaging a server deployment project](#).
- Generate the standalone native application package for iOS and Android, so the user can publish the application in the online app stores or distribute it over-the-air. For iOS, the Package Wizard generates an Xcode project which can be compiled to an iOS application archive (IPA) file by using Xcode later. For Android, the Package Wizard generates an Android application package (APK) file which can be published or distributed to the end users right away. Refer to [Packaging a stand-alone mobile project](#).
- Customize Appeon Workspace and package Appeon Workspace to a standalone application. Refer to [Customizing and packaging Appeon Workspace](#).

Before using the Appeon Application Package Wizard, verify that

1. The target PowerBuilder application has been deployed as required:

You must have performed a **full deployment** on the target PowerBuilder application on the **same developer machine** where you will use the packaging wizard to pack this application later, this will generate a complete set of files on the local developer machine where the package wizard will be able to obtain and pack files.

2. The deployed application can run correctly.

10.1 Packaging a server deployment project

10.1.1 What can be packaged?

The package wizard allows you to pack the following files and settings into a server deployment project:

- Mandatory Web or mobile application files
Include all application files, DataWindow syntax, INI files, image files, DLL/OCX files etc.
- Mandatory application profile, PowerServer profile, and Web server profile settings
Include the profile name, Application URL, PowerBuilder version, file encryption, performance settings, server type, PowerServer version, PowerServer and Web server connection settings, AEM, data sources, etc. All these settings will be saved to the configuration files in the INI folder of the generated package. Of these configuration files, you can modify the config.xml file using the config.exe tool. For detailed information, refer to [Section 10.1.3, “Modifying the deploy-config file”](#).

- Optional EAServer components

When PowerServer is installed to EAServer, you can package the local or remote EAServer components used by the application. Note that EAServer components work with EAServer application server only.

10.1.2 Packaging instructions

To package a server deployment project and the related settings, take the following steps:

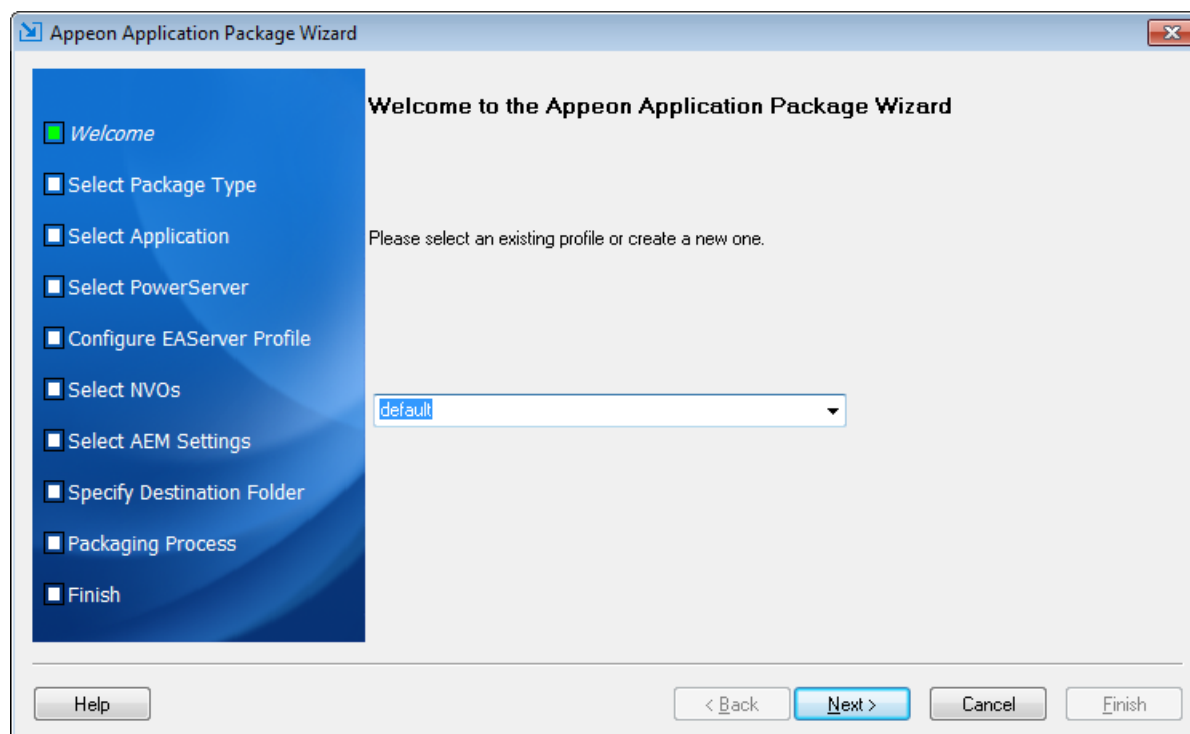
Step 1: Start the servers, as the package wizard will pack settings of PowerServer, AEM, and NVOs (if any).

Step 2: Click the **Package** (📦) button on the PowerServer Toolkit to open the Appeon Application Package Wizard.

Step 3: Select or create a profile from the dropdown list box and click **Next** to proceed.

A profile is a configuration file containing configurations that you specify when packaging the application. You can select an existing profile or create one by entering a name in the text field. The profile will be automatically saved and listed for selection next time when you launch the Appeon Application Package Wizard again.

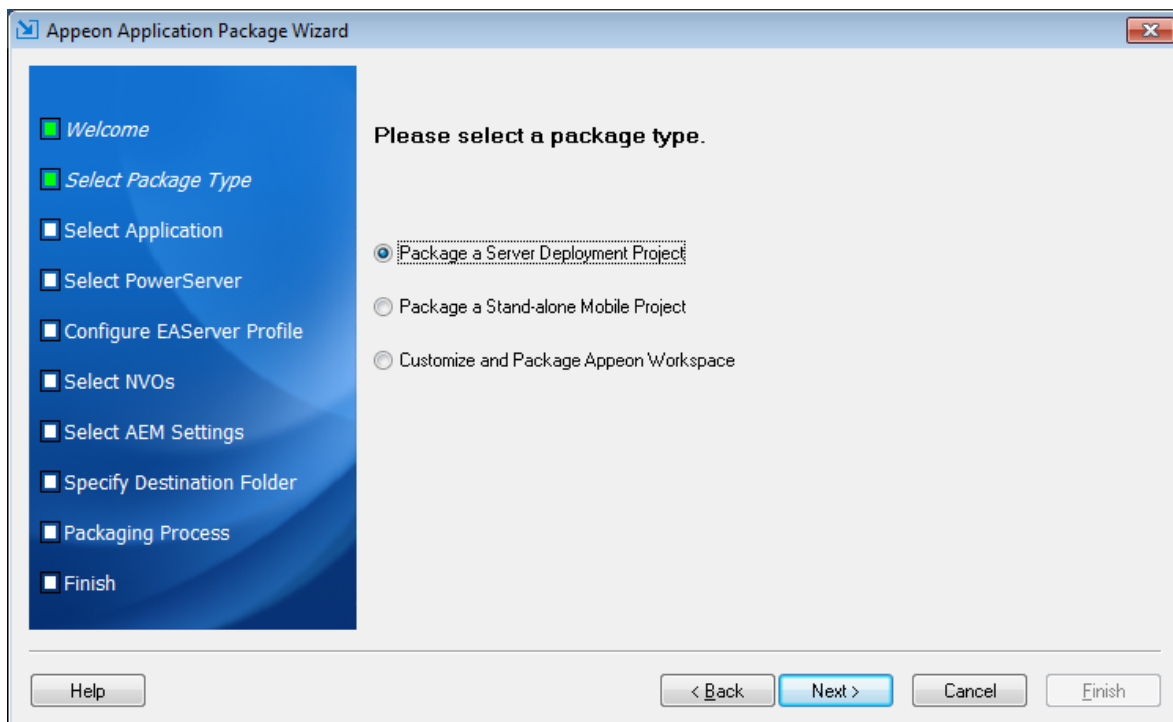
Figure 10.1: Welcome page



Step 4: Select the **Package a Server Deployment Project** radio button.

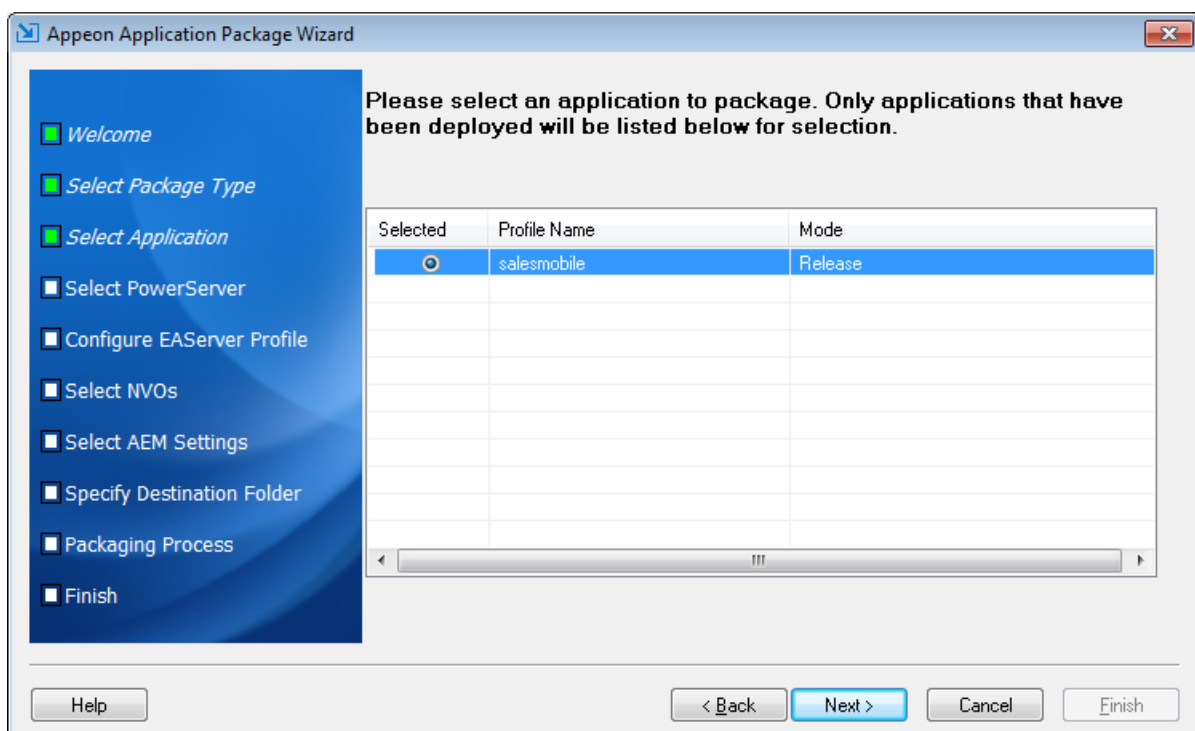
To package the mobile application to a stand-alone mobile app, select **Package a Stand-alone Mobile Project** and follow detailed instructions in [Packaging a stand-alone mobile project](#).

To customize Appeon Workspace and then package Appeon Workspace to a standalone application, select **Customize and Package Appeon Workspace** and follow detailed instructions in [Customizing and packaging Appeon Workspace](#).

Figure 10.2: Select package type

Step 5: Select the profile of the application that you want to package and click **Next**.

Only applications that have been deployed will be listed here for selection. If the application you intend to package is not listed here, you would need to deploy the target application using the Appeon Deployment Wizard first (See [Chapter 6, Deploying PowerBuilder Applications](#)).

Figure 10.3: Select an application

The following table gives a brief introduction to the columns:

Table 10.1: Application Package Wizard

Column	Description
Selected	Identical to the default application profile selected in the PowerServer Toolkit Configuration window.
Profile Name	Identical to the application profiles configured in the PowerServer Toolkit Configuration window.
Mode	Identical to the current mode the application profile is set to in the PowerServer Toolkit Configuration window. It is recommended that the packaged applications should be available in Release mode, as this mode prevents most forms of reverse engineering, and like application packaging, it protects the author's intellectual property.

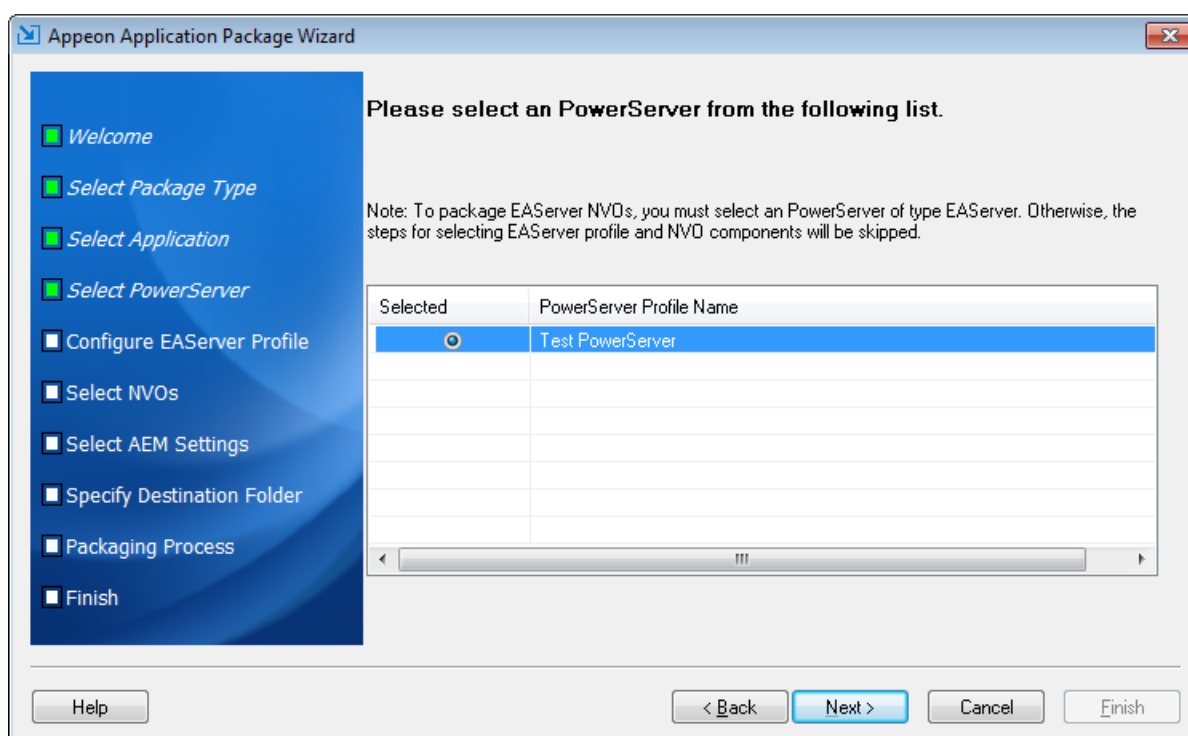
Step 6: Select the PowerServer whose settings will be packaged and click **Next**.

The PowerServer settings, such as the server type and the PowerServer version, will be saved to the config.xml file in the INI folder of the generated package. If necessary, modify this file using the config.exe tool after the packaging is complete (refer to [Section 10.1.3, “Modifying the deploy-config file”](#)).

All the configured PowerServer selected into the default deployment profile in the PowerServer Toolkit Configuration window are listed, as shown in the following figure.

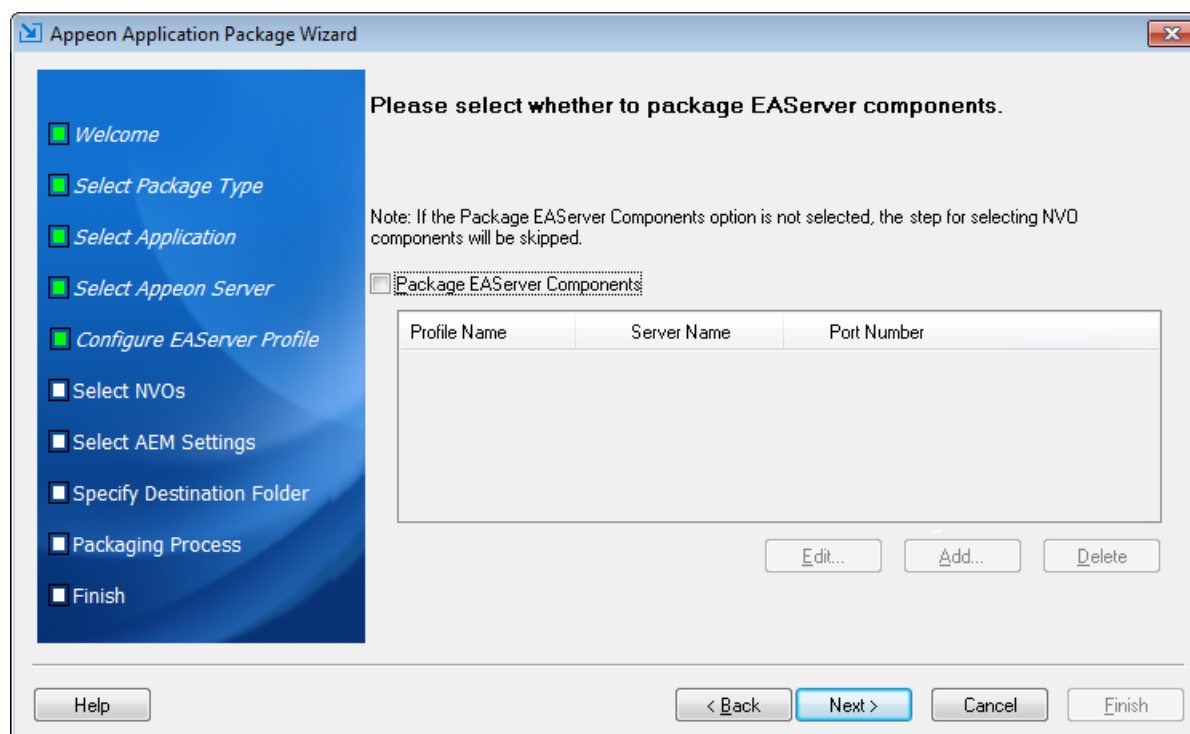
The selected PowerServer determines that its AEM settings will be packaged and that the packaged application can only be installed to PowerServer of the same version, such as 2017, but not necessarily the same type.

Figure 10.4: Select a PowerServer



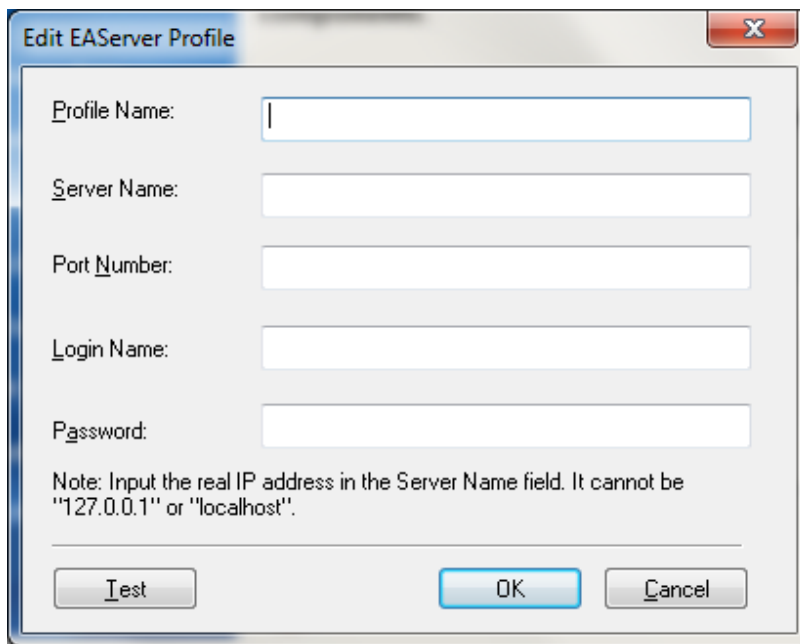
Step 7: If the selected PowerServer is installed to EAServer, select whether to package EAServer components.

Figure 10.5: Configure EAServer profiles



If EAServer components are used in the application, select the **Package EAServer Components** option and then create connection profiles for EAServers which host the components. Follow instructions below to create EAServer profiles:

- Click **Add** to create an EAServer profile.
- In the **Edit EAServer Profile** dialog box, input the following connection parameters: profile name, host name or IP address, port number, user name, and password.
- Click **Test** to ensure that the connection is successful.
- Click **OK** to save the new EAServer profile.

Figure 10.6: Edit EAServer profile

Edit EAServer Profile

Profile Name:

Server Name:

Port Number:

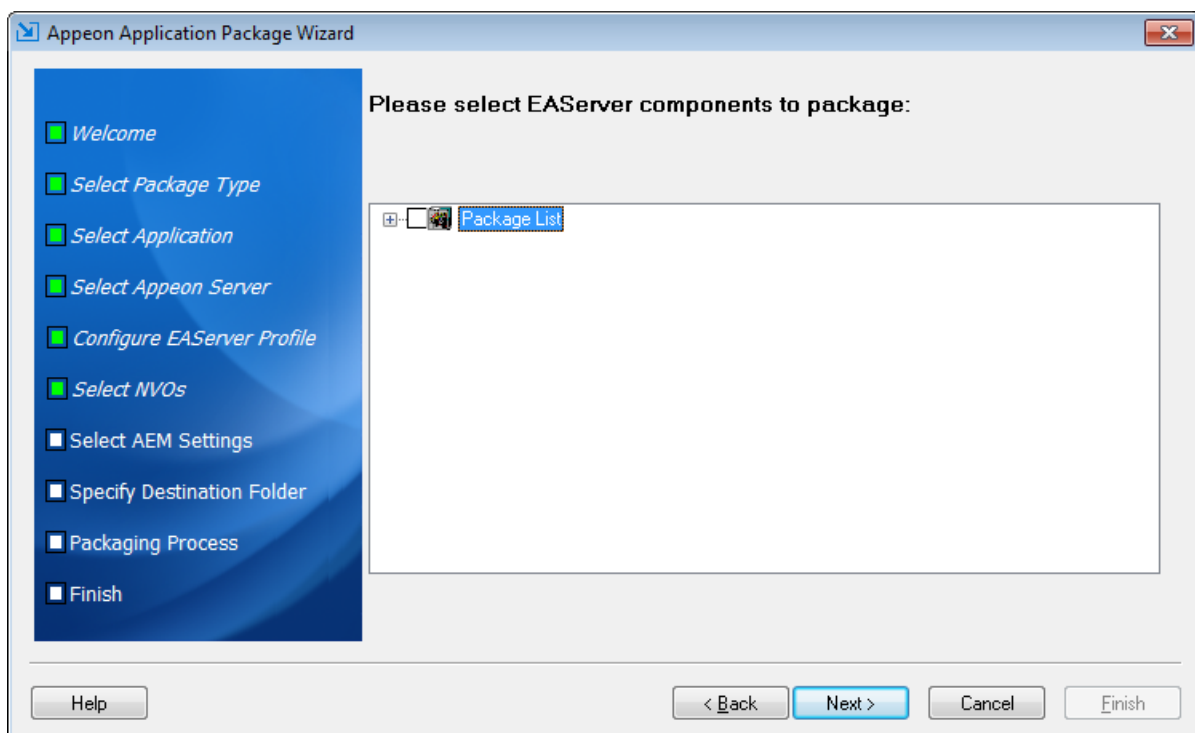
Login Name:

Password:

Note: Input the real IP address in the Server Name field. It cannot be "127.0.0.1" or "localhost".

Step 8: If the **Package EAServer Components** option is selected in the previous step, select which components to package and click **Next**.

The selected NVOs will be packaged as *ComponentName.jar* and saved under the /NVO/ server profile name/package folder.

Figure 10.7: Select NVOs

Appeon Application Package Wizard

Please select EAServer components to package:

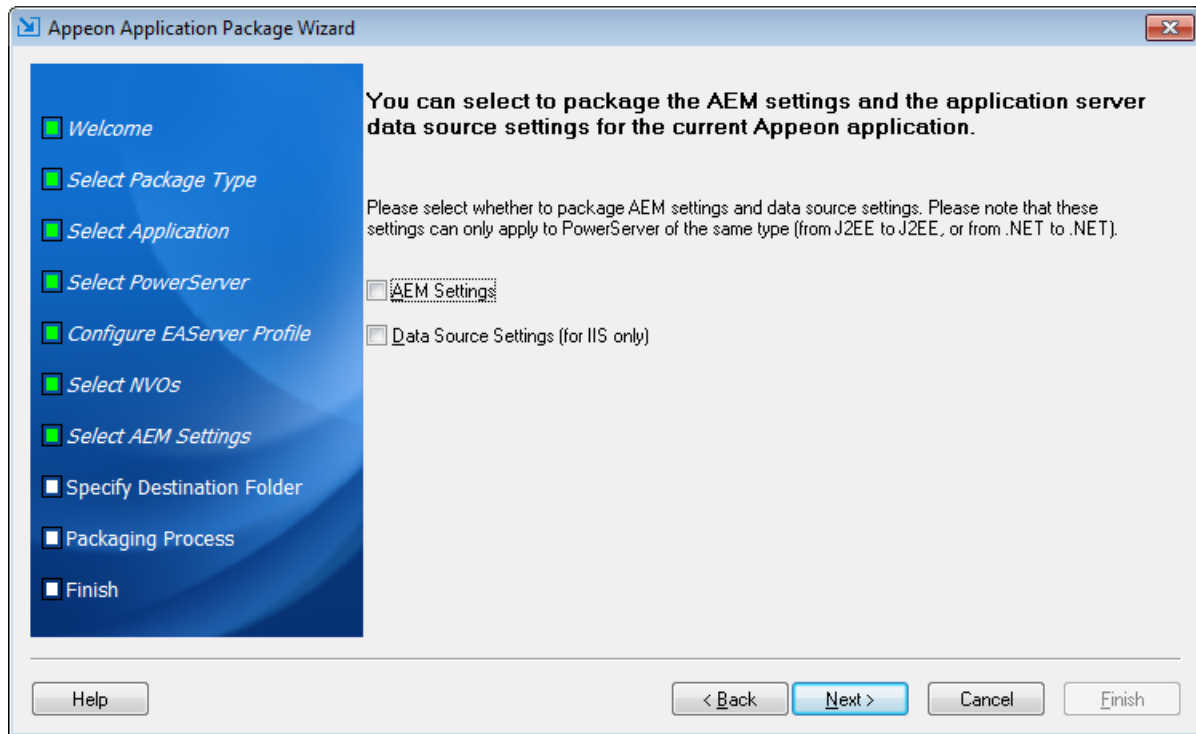
Package List

Help

Step 9: Select whether AEM settings and data sources for the current application will be packaged and click **Next**.

The AEM settings will be saved in the AEM.xml file and data sources will be saved in the config.xml file.

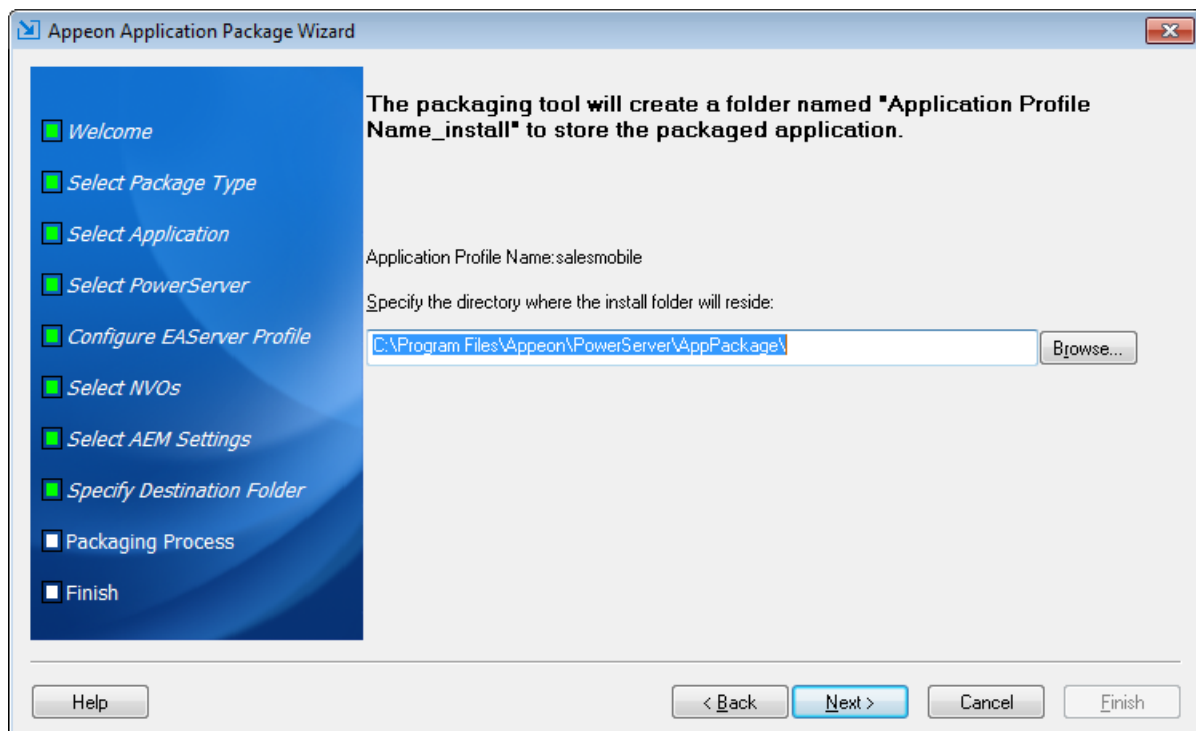
Figure 10.8: Select AEM settings and data source



Step 10: Specify the storage location for the generated package and click **Next**.

The generated package will be stored under a folder named "*Application Profile Name_install*" under the specified location.

Figure 10.9: Specify destination location for the package

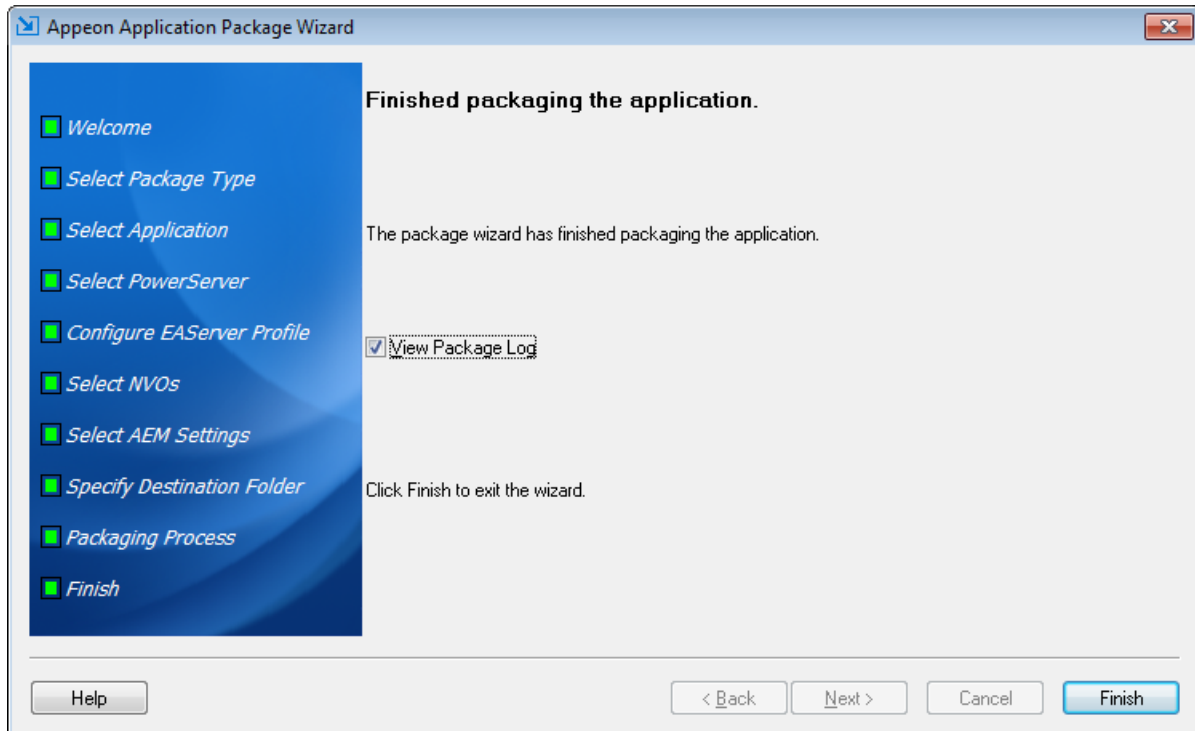


Step 11: Wait while the wizard is packing files and settings.

Step 12: Once the operation is complete, click **Finish** to exit the package wizard.

To view the log information generated during the packaging process, select the **View Package Log** box and then click **Finish**. The log file will be displayed.

Figure 10.10: Application packaging finished



After the application is successfully packaged, all the necessary application files will be packaged into an executable deployment project under the "*Application Profile Name_install*" folder under the specified location. You can copy the deployment project to other computers and deploy the application to any PowerServer and Web Servers by following instructions in [Section 10.1.4, "Installing and uninstalling an application to the server"](#).

10.1.3 Modifying the deploy-config file

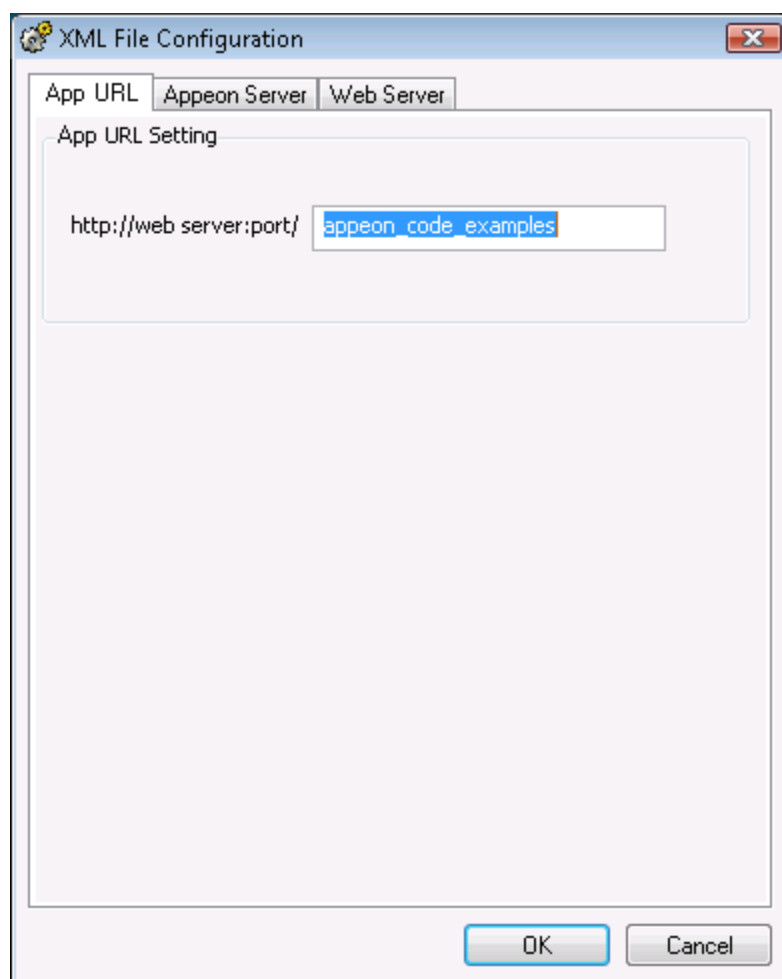
The deploy-config file (config.xml) is created during the packaging process and includes the default settings of the application: application profile, PowerServer profile, Web server profile, EAServer profile, EAServer components, and data sources. You can modify the config.xml file using the config.exe tool to have identical settings of the target environment, so that the user can run the installation very conveniently without needing to specify them.

When you run the config.exe file in the generated package, an XML File Configuration dialog is displayed and loads the Application URL setting, PowerServer settings, and Web server settings from the config.xml file. You can modify these settings on the corresponding tab pages and then click **OK** to save them to the config.xml file. The instructions to modify these settings are the same as to modify in the PowerServer Toolkit Configuration window (see [Section 4.2, "Using Configure Tool"](#) for detailed instructions).

Important Note: If you are installing a Web application to a JEUS Web server, or installing a mobile application, be sure to use the default Application Profile Name and Application

URL. Changing the default Application Profile Name and Application URL will cause the application files inaccessible.

Figure 10.11: Configuring the XML file



10.1.4 Installing and uninstalling an application to the server

10.1.4.1 Points to note before installation

Be aware of the following points before you run the deployment project to install an application to the server:

- The Setup program of the generated deployment project runs on the Windows platform only. To install applications to servers running on Unix/Linux, you must run the Setup program on the Windows platform and then install the application to remote servers running on Unix/Linux.
- The target server must have PowerServer correctly installed. If the Web server is separated from the PowerServer, then the Web server must have the PowerServer Web Component installed.
- The packaged EAServer components, if any, can only be installed to EAServer application server.

10.1.4.2 Installing an application

The generated deployment project runs on the Windows platform only. Therefore, you will need to copy the deployment project to a computer running Windows and then install the application to any number of PowerServer and Web servers located in the same network area. Installing an application using the deployment project is the same as deploying an application using the Appeon Deployment Wizard.

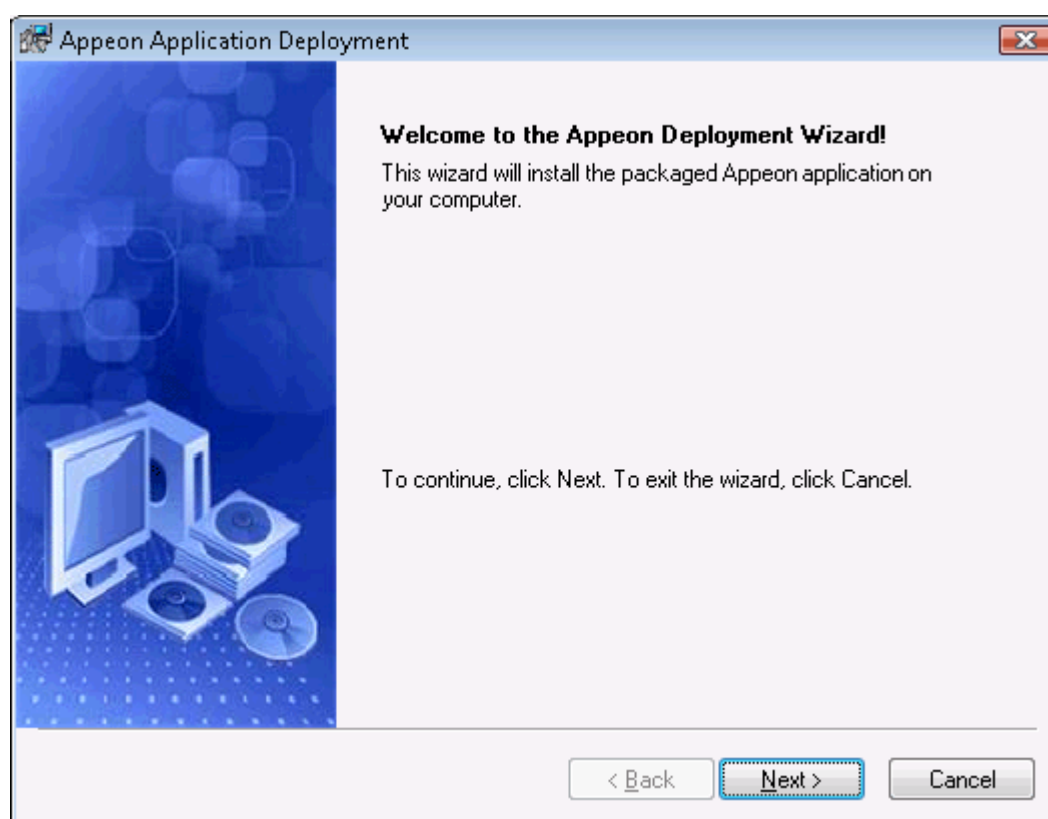
Step 1: Start the target PowerServer(s) and Web server(s) where you want to install the application.

Verify that the target servers already have PowerServer installed.

Step 2: Run the **Setup.exe** file in the generated deployment project.

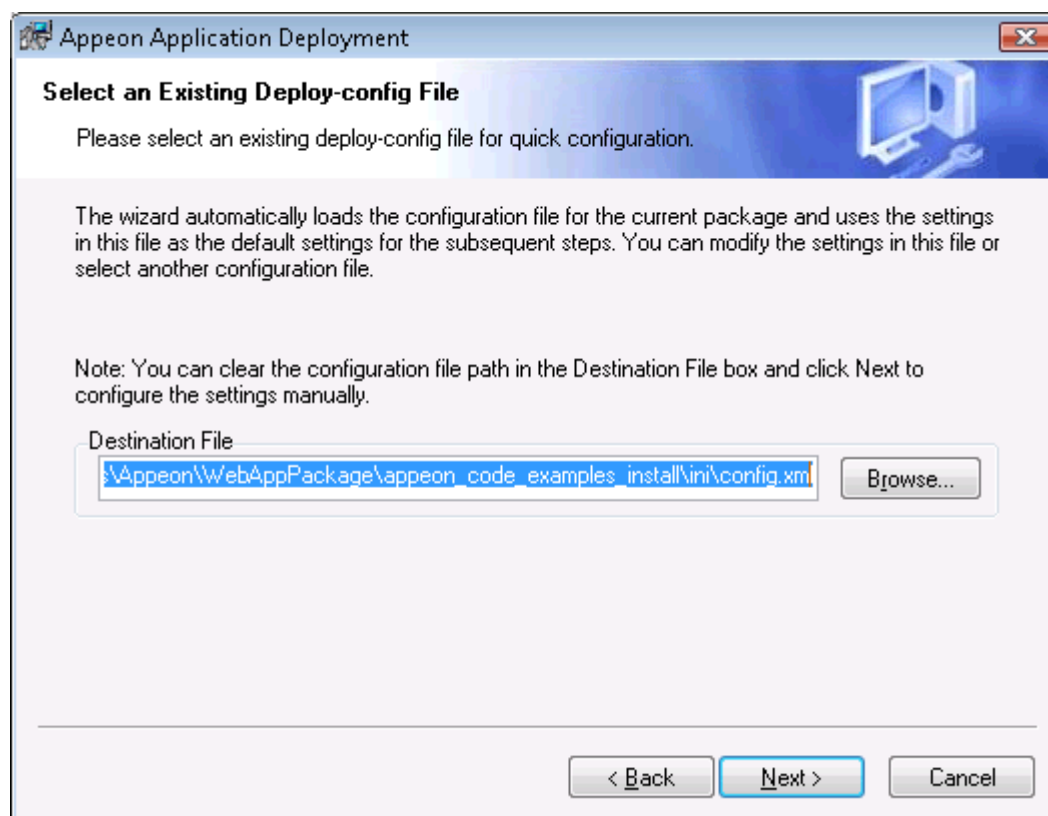
The Appeon Application Deployment wizard is displayed. Click **Next** to proceed.

Figure 10.12: Welcome page



Step 3: Select the deployment configuration file (config.xml) to ease your configuration of the installation wizard. The installation wizard will use the settings in the config.xml file.

If you do not want to use the deployment configuration file, simply skip this option and click **Next** to configure the settings step by step.

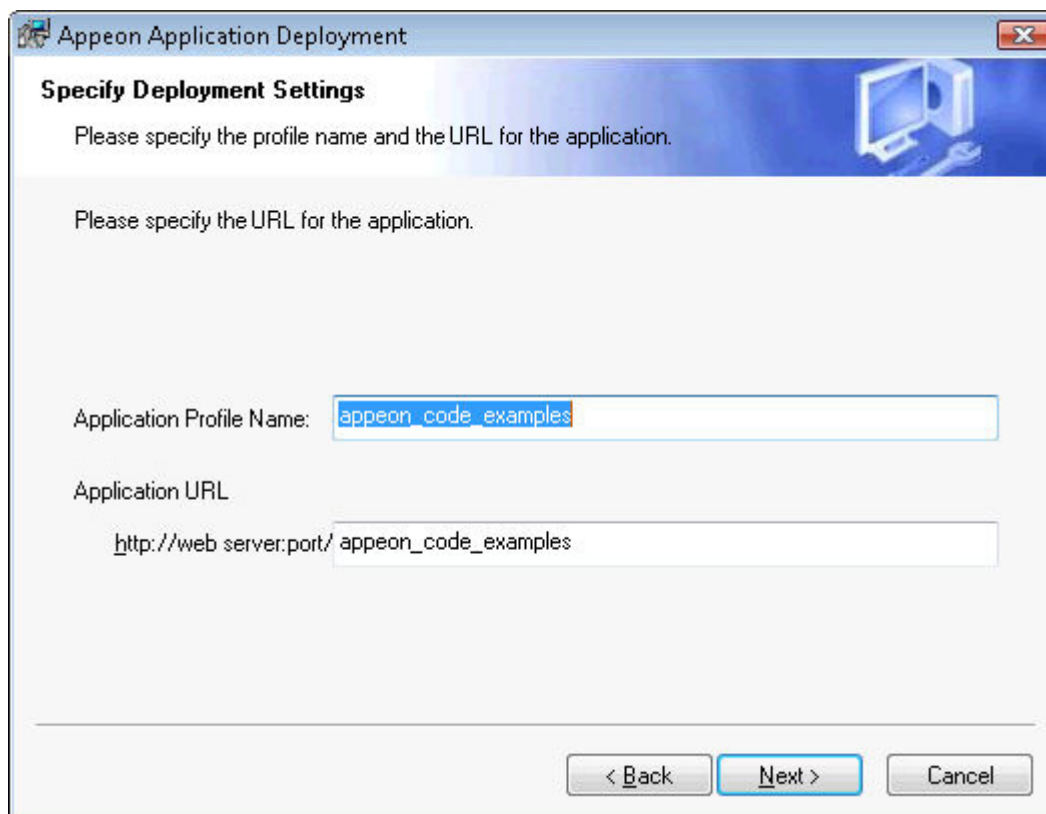
Figure 10.13: Select the configuration file

Step 4: Input the **Application Profile Name** and the **Application URL** for the application and click **Next**.

The **Application Profile Name** is used as the application name to identify an application. If you want to install multiple instances of the same application on the same server, you can run the setup package repeatedly and specify different Application Profile Name here. For example, input `appeon_code_examples_test` as the Application Profile Name in the first installation, and input `appeon_code_examples_production` as the Application Profile Name in the second installation. The Application Profile Name will be used as the application name to distinguish the multiple application instances on the same server, so they can be run independently from each other.

However, there are the following circumstances that you should **not** use different Application Profile Name and Application URL from those specified when creating the package, otherwise the application would not run:

- when you are installing a mobile application
- when you are installing a Web application to a JEUS Web server

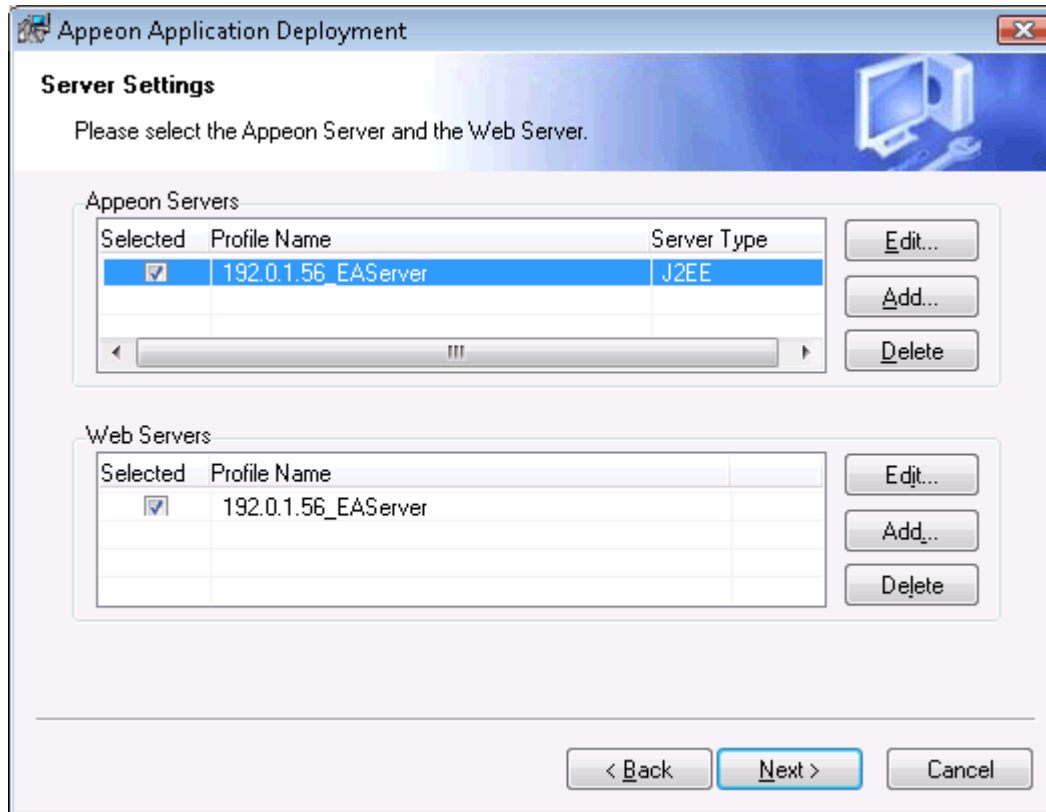
Figure 10.14: Specify Application Profile Name and Application URL

The screenshot shows a dialog box titled "Apeon Application Deployment" with a close button in the top right corner. The main heading is "Specify Deployment Settings". Below the heading, there are two lines of instructional text: "Please specify the profile name and the URL for the application." and "Please specify the URL for the application." The dialog contains two input fields. The first is labeled "Application Profile Name:" and contains the text "apeon_code_examples". The second is labeled "Application URL" and contains the text "http://web server:port/ apeon_code_examples". At the bottom of the dialog, there are three buttons: "< Back", "Next >", and "Cancel".

Step 5: Configure and select profiles for the PowerServer(s) and the Web server(s) where the application will be installed.

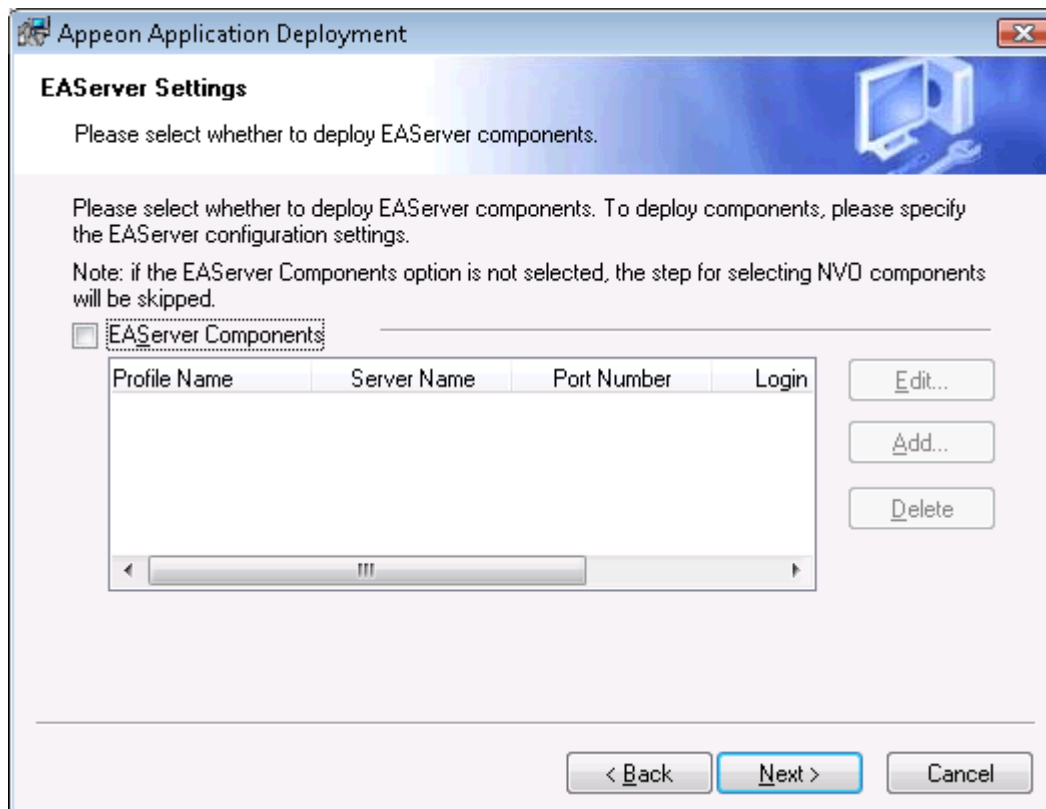
To install the application to more than one PowerServer, please make sure the PowerServer are of the same version (such as 2017) as the PowerServer selected for packaging.

To install the application to a PowerServer cluster, please create a PowerServer profile for each PowerServer in the cluster. For detailed instructions on how to create the PowerServer profile and Web server profile, refer to [Section 4.2.3, "Managing server profiles"](#).

Figure 10.15: Specify server settings

Step 6: Select whether to deploy EAServer components.

After NVO components are deployed, you must generate stub/skeleton in EAServer.

Figure 10.16: Specify EAServer settings

If EAServer components are used in the application, select the **EAServer Components** option and then create connection profiles for EAServers where you want to deploy the components. Follow instructions below to create EAServer profiles:

- Click **Add** to open the **Edit EAServer Profile** dialog box.
- Input the connection parameters: profile name, host name or IP address, port number, user name and password.
- Click **Test** to verify the connection and the click **OK** to finish the creation.

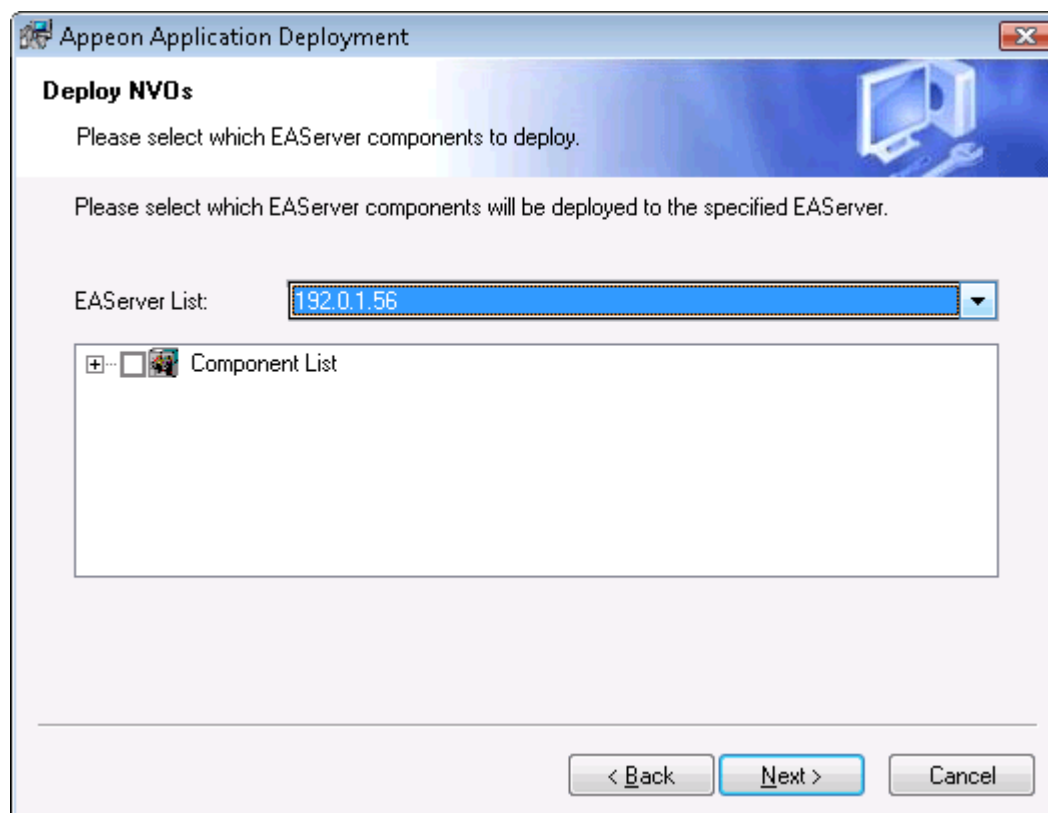
Figure 10.17: Edit EAServer Profile



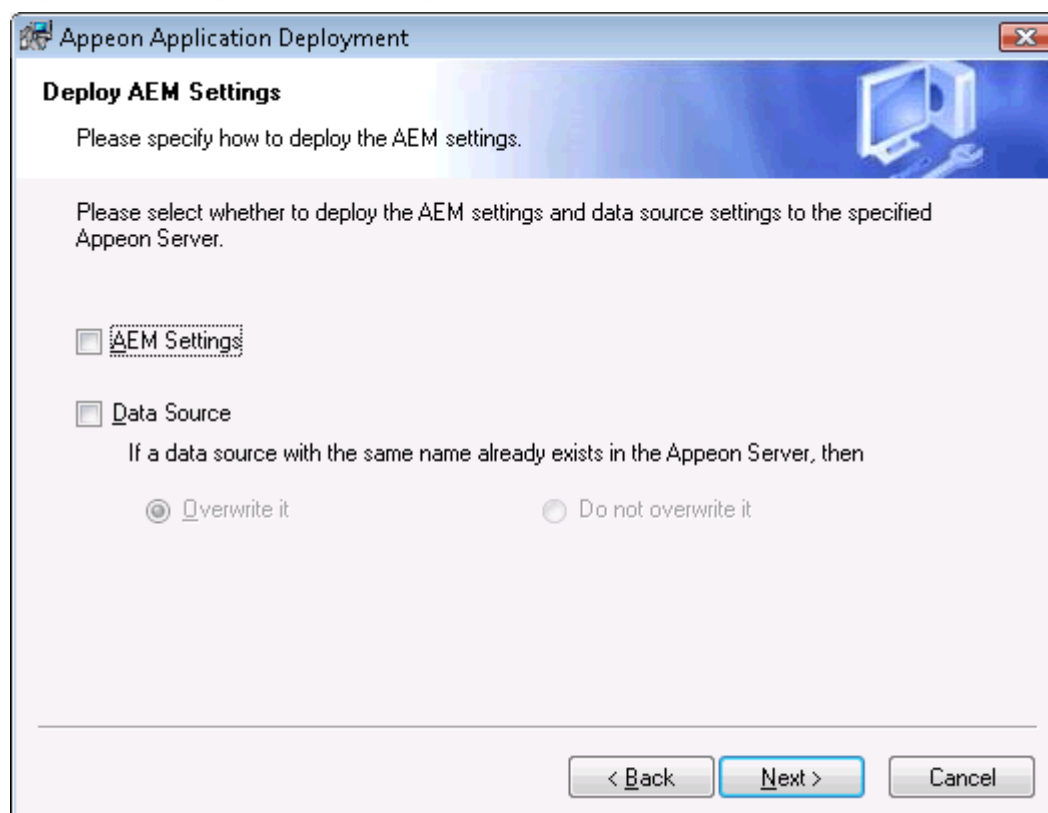
The screenshot shows a dialog box titled "Edit EAServer Profile". It contains the following fields and controls:

- Profile Name:** A text input field.
- Server Name:** A text input field.
- Port Number:** A text input field.
- Login Name:** A text input field.
- Password:** A text input field.
- Note:** "Input the real IP address in the Server Name field. For example, 192.0.0.1."
- Buttons:** "Test", "OK", and "Cancel".

Step 7: If the **EAServer Components** option is selected in the previous step, select the EAServer profile from the list and the components to deploy. Click **Next** to proceed.

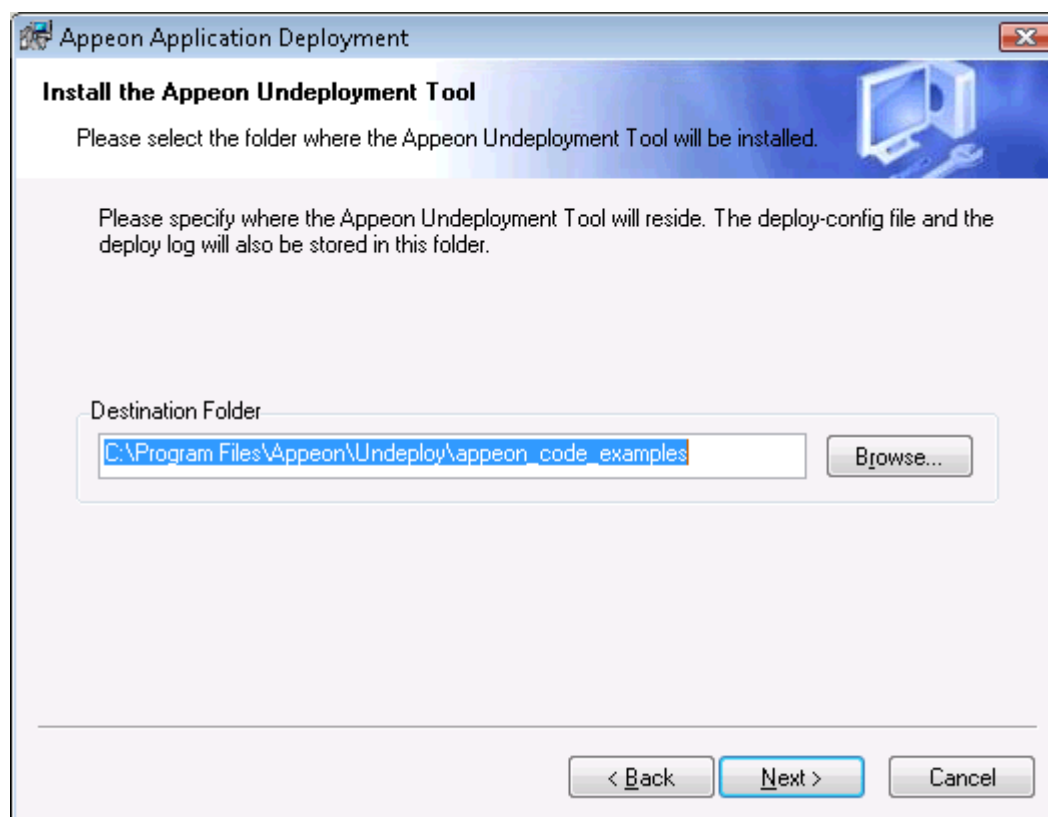
Figure 10.18: Deploy NVOs

Step 8: Select whether to install AEM settings and data sources for the current application and click **Next**.

Figure 10.19: Deploy AEM settings

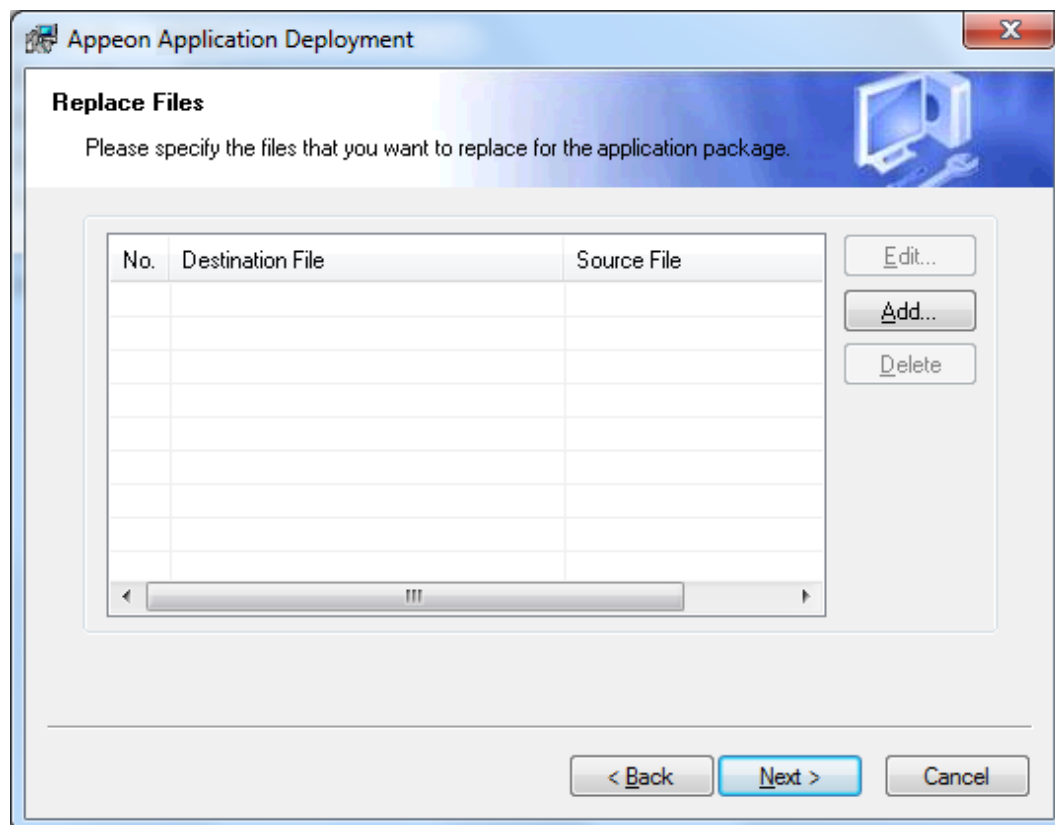
Step 9: Specify the destination folder for the undeployment tool and the log file and click **Next**.

Figure 10.20: Specify location for the Appeon Undeployment Tool

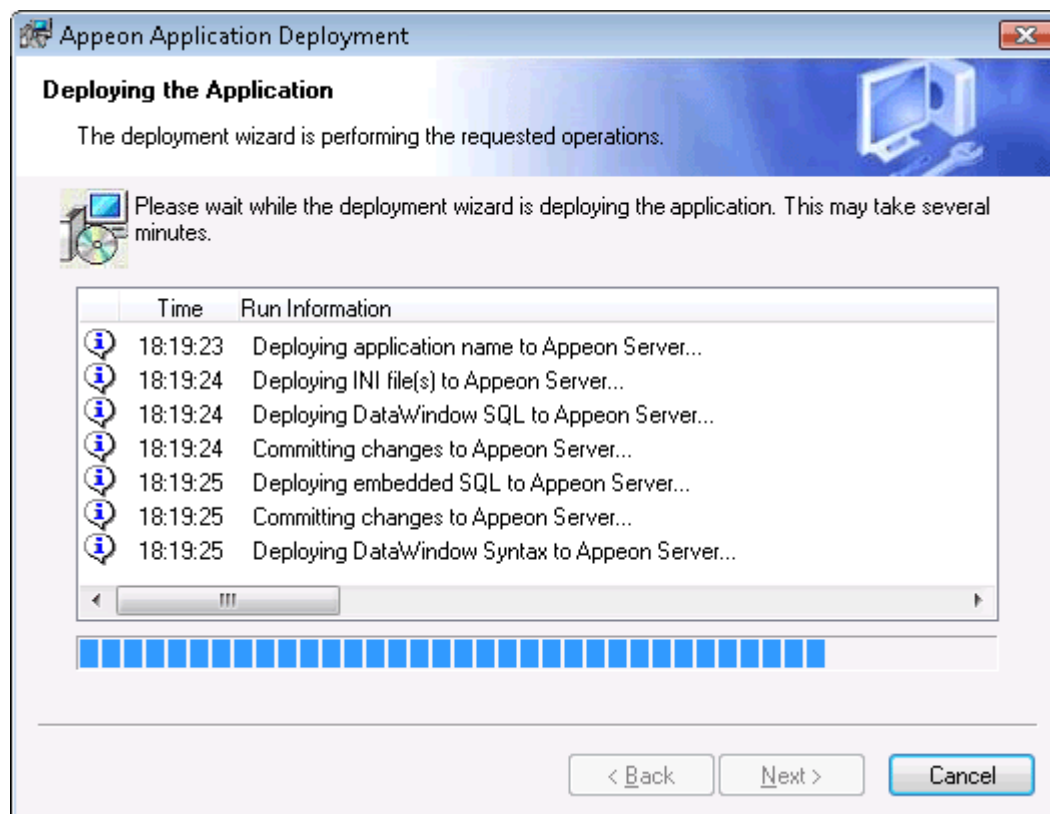


Step 10: Specify the destination file and source file that you want to replace for the Appeon application package, and click **Next**.

Files (also called "destination file") under the "wcode" folder of Appeon application package can be replaced by files of the same type (also called "source file"), so that the source file will be installed instead of the destination file. When you click the **Add** button, the **File Replacement** dialog box is opened. To select the destination file, click **Browse** to display the Open file dialog which will automatically open the "wcode" folder of Appeon application package, so you can conveniently select a file from the "wcode" folder or from its subfolder. To select the source file, click **Browse** to navigate to the folder where the source file is stored.

Figure 10.21: Specify files to replace for the application package

Step 11: Wait while the wizard is installing files and settings.

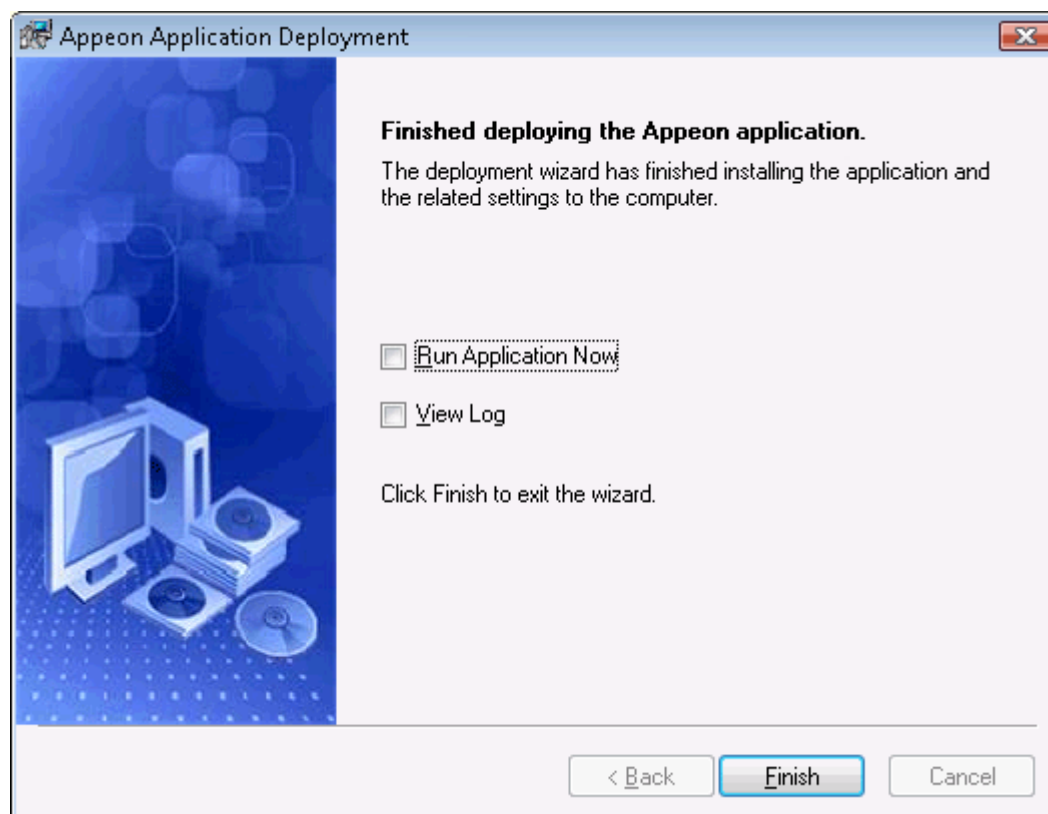
Figure 10.22: Deploying the application

Step 12: Once the operation is complete, click **Finish** to exit the installation wizard.

To run the application immediately, select the **Run Application Now** option and click **Finish**. Or you can run the application later from the Windows **Start > Programs > Appeon Web Application > ApplicationName**.

To view the log information generated during the installation process, select the **View Log** box and then click **Finish**. The log file will be displayed.

Figure 10.23: Deployment complete

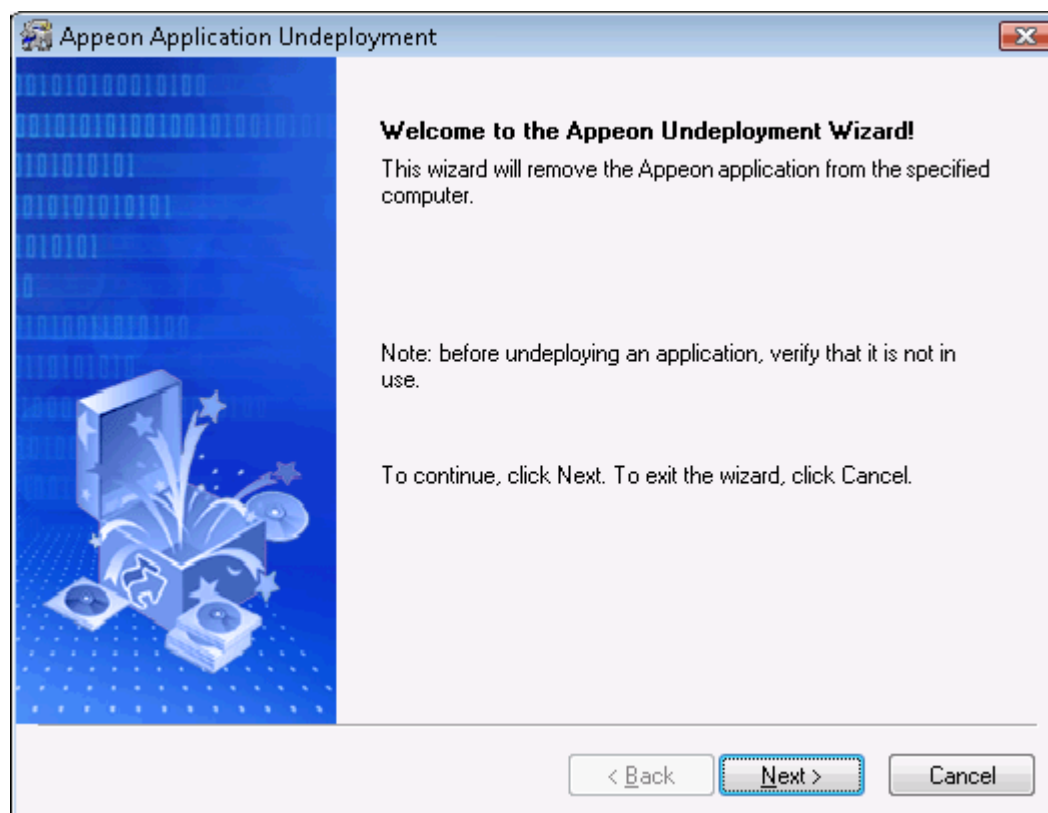


10.1.4.3 Uninstalling an application

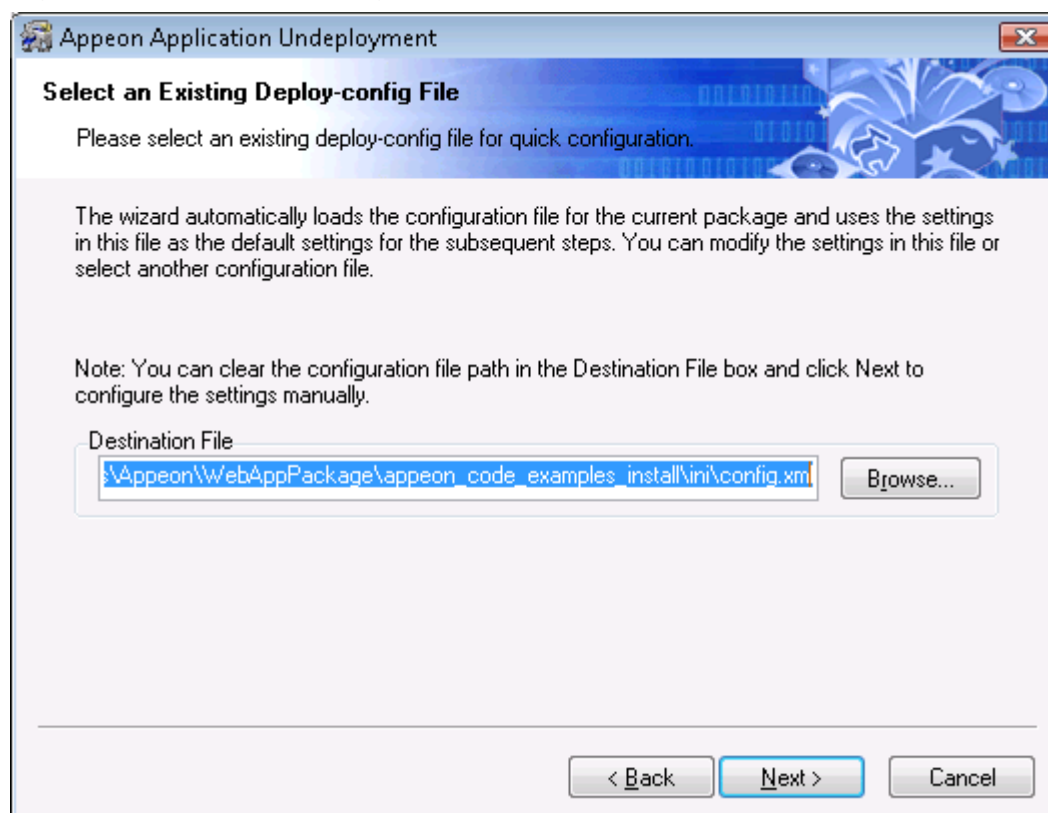
Step 1: Start the PowerServer(s) and/or Web server(s) where you intend to uninstall the application.

Step 2: Select Windows **Start > All Programs > Appeon Application > Undeploy ApplicationName**.

The Appeon Application Undeployment Wizard is displayed. Click **Next** to proceed.

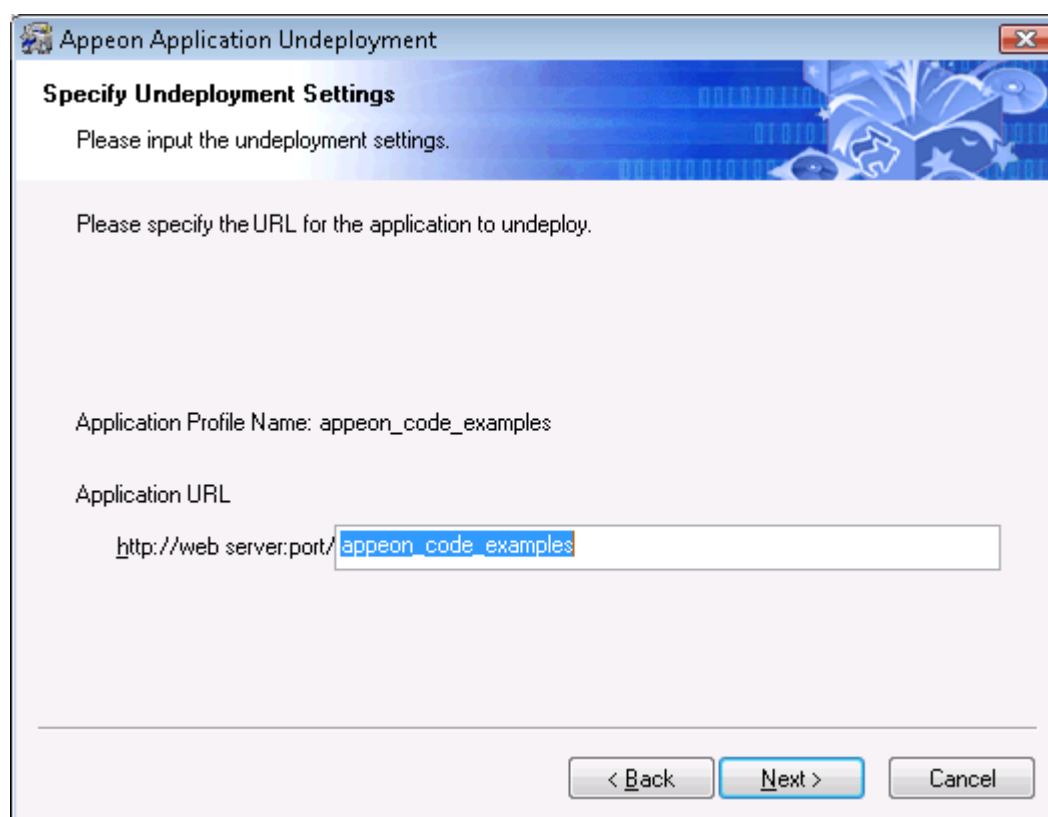
Figure 10.24: Appeon Application Undeployment wizard

Step 3: Select an existing deploy-config file to automatically use the configuration previously used for the deployment wizard.

Figure 10.25: Select the configuration file

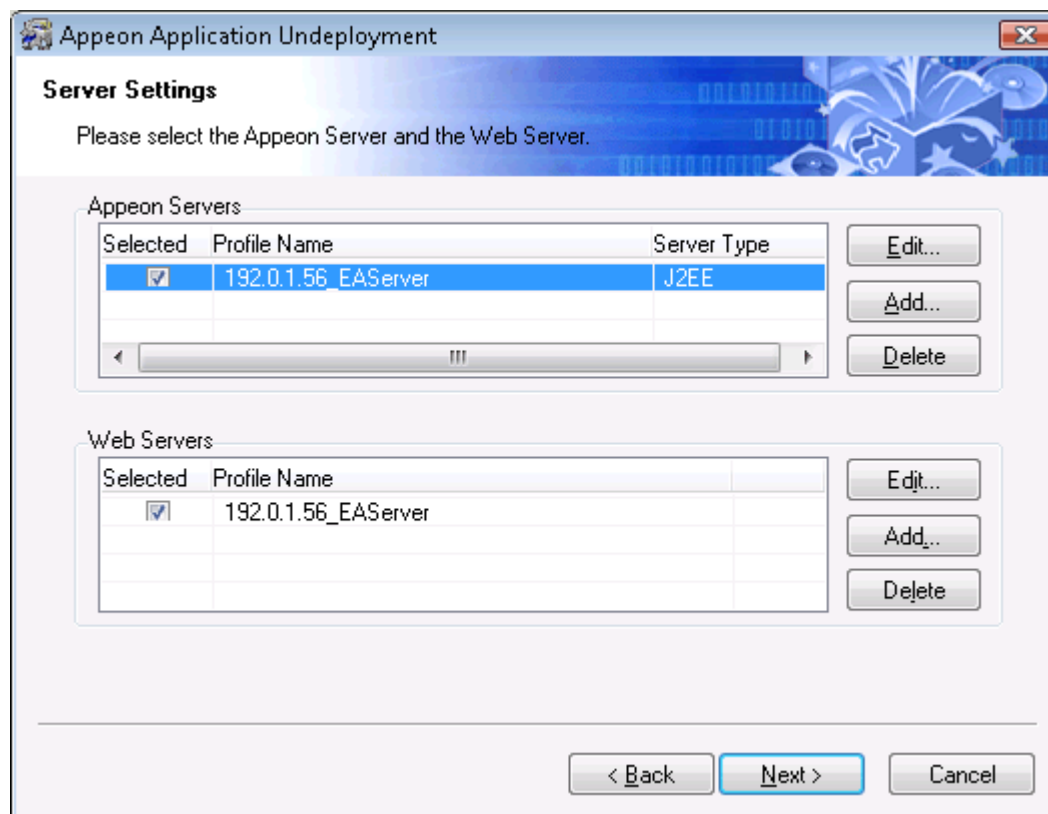
Step 4: Specify the Application URL.

Figure 10.26: Specify Application URL

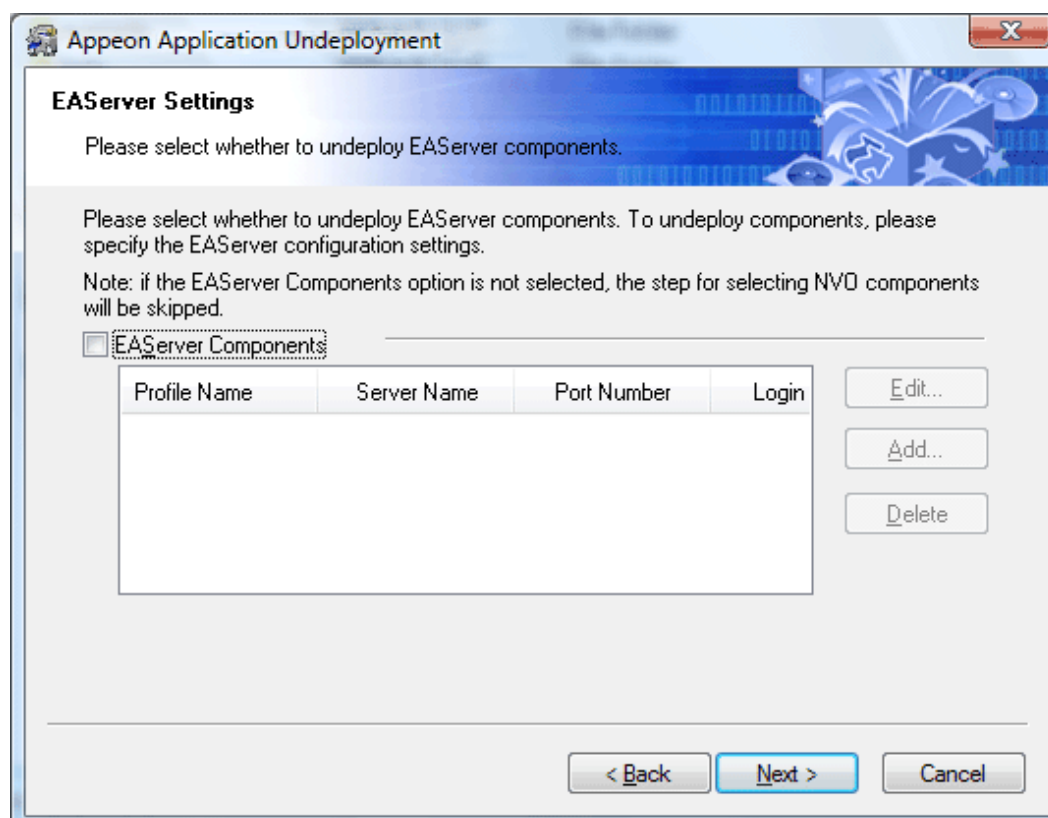


Step 5: Select the PowerServer(s) and Web server(s) where application will be undeployed. Click **Next** to proceed.

To create the PowerServer profile and Web server profile, refer to [Section 4.2.3, "Managing server profiles"](#).

Figure 10.27: Specify server settings

Step 6: Select whether to undeploy EAServer components.

Figure 10.28: Undeploy EAServer components

Select the **EAServer Components** option and then create connection profiles for EAServers where you want to undeploy the components. Follow instructions below to create EAServer profiles:

- Click **Add** to open the EAServer Connection dialog box.
- Input the connection parameters: profile name, host name or IP address, port number, user name and password.
- Click **Test** to verify the connection and the click **OK** to finish the creation.

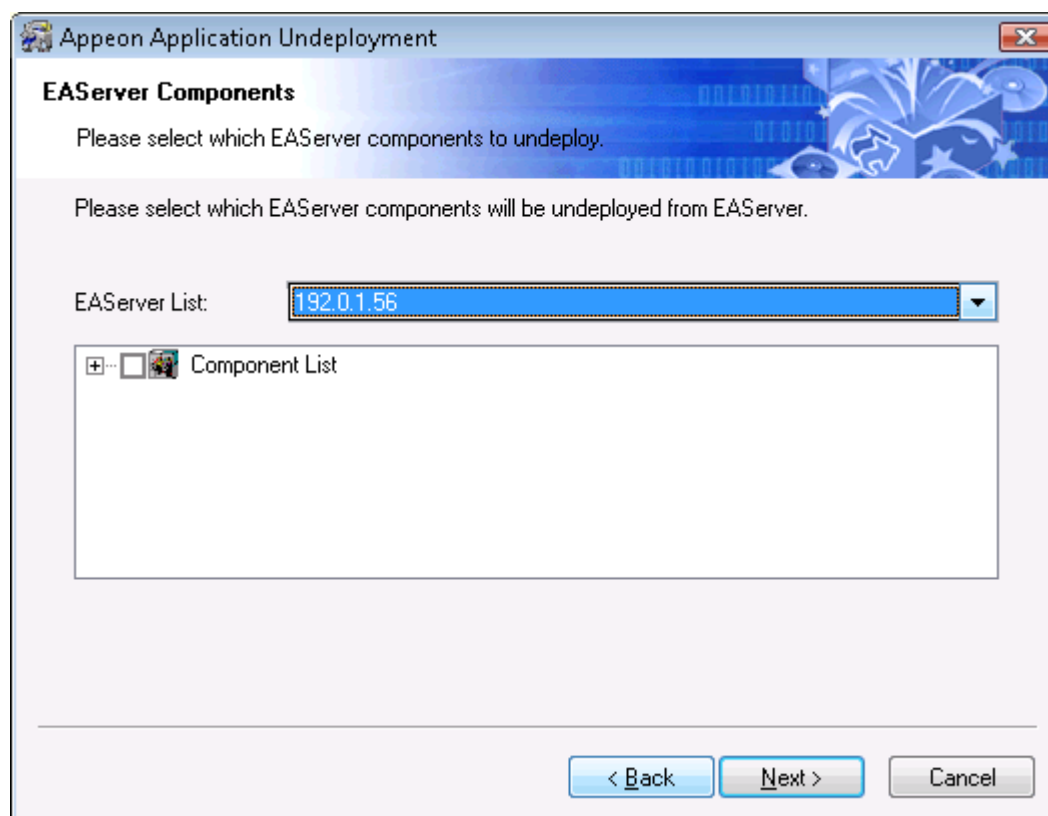
Figure 10.29: Edit EAServer profile



The screenshot shows a dialog box titled "Edit EAServer Profile". It contains the following fields and controls:

- Profile Name:
- Server Name:
- Port Number:
- Login Name:
- Password:
- Note: Input the real IP address in the Server Name field. For example, 192.0.0.1.
- Buttons: Test, OK, Cancel

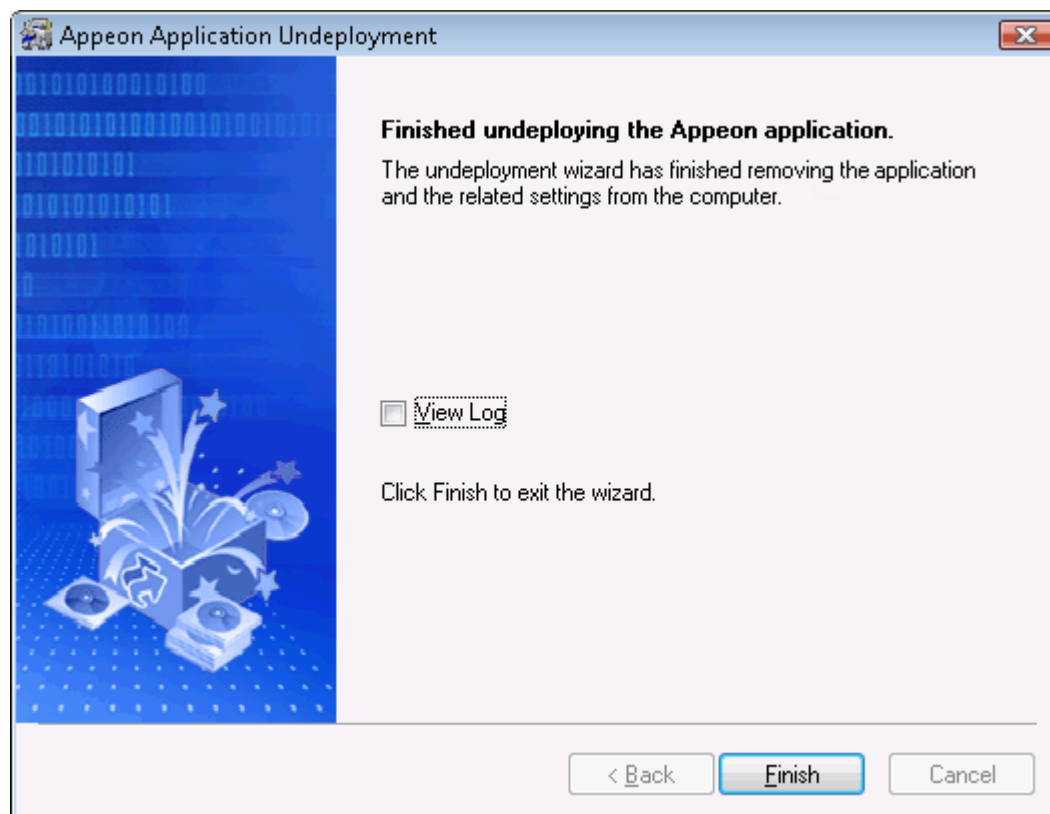
Step 7: If the **EAServer Components** option is selected in the previous step, select the EAServer profile from the list and the components to undeploy. Click **Next** to proceed.

Figure 10.30: Select EAServer components

Step 8: Wait while the wizard is removing files and settings.

Step 9: Once the operation is complete, click **Finish** to exit the undeployment wizard.

To view the log information generated during the uninstall process, select the **View Log** box and then click **Finish**. The log file will be displayed.

Figure 10.31: Undeployment complete

10.2 Preparing for the mobile package

These preparations are for packaging the stand-alone mobile apps (for iOS and Android) and customizing and packaging the Apeon Workspace.

10.2.1 Preparations for the iOS package

Before you packaging the iOS application, make the following preparations accordingly:

1. **(Required)** Register for an Apple ID.

You can follow the onscreen instructions on [My Apple ID](#) to create an Apple ID if you do not have one.

2. **(Required)** Prepare a Mac machine with the latest Xcode installed.

The **Package Wizard** will generate an Xcode project which you will need to compile into the IPA file using Xcode. More instructions are provided in Section 5.2.1.2, “Task 1.2: Prepare the Mac machine” in *PowerServer Mobile Tutorials*.

Note that if you have used the Payment APIs in the application, make sure you compile the application using Xcode 5.1 or above, otherwise Payment APIs will not work.

3. **(Required)** Enroll in an iOS Developer program.

The iOS Developer program determines how you can distribute the app. More instructions are provided in Section 5.2.1.3, “Task 1.3: Enroll in an iOS Developer Program” in *PowerServer Mobile Tutorials*.

4. **(Required)** Create an App ID.

It is strongly recommended that you consistently use the same App ID in every place that requires you to input an App ID. This is mainly for the following two reasons: 1) Using the same App ID when packaging the iOS applications can help Appeon determine whether the apps are running on the same device, so Appeon can correctly manage the device sessions in the license file. 2) Using the same App ID when working with Xcode can eliminate the likelihood of conflicts between the certificate and the provisioning profile.

Instructions for creating an App ID are provided in Section 5.2.1.4, “Task 1.4: Create an App ID” in *PowerServer Mobile Tutorials*.

5. **(Required)** Create & install a distribution certificate & a distribution provisioning profile.

More instructions are provided in Section 5.2.1.5, “Task 1.5: Create & install a distribution certificate” in *PowerServer Mobile Tutorials* and Section 5.2.1.6, “Task 1.6: Create & install a distribution provisioning profile” in *PowerServer Mobile Tutorials*.

6. (Optional) Prepare the icons/images for your application.

You can provide your own app icons, settings icons, spotlight icons, and launch images for your iOS application. If you do not provide the icons and images, the **Package Wizard** will use the default ones provided by Appeon.

- a. App icons, spotlight icons, and settings icons: Refer to [App Icon \[174\]](#) for the size requirement for different iOS devices.
- b. Launch images: Refer to [Launch Image \[174\]](#) for the size requirement for different iOS devices.

It is optional to reduce the iOS status bar (20 pixels for standard display and 40 pixels for retina display) for the launch image, for example, 1004 x 768 pixels as the launch image for standard display in landscape on iPad.

10.2.2 Preparations for the Android package

Before you packaging the Android application, make the following preparations accordingly:

10.2.2.1 **(Required)** Obtain a private key

The Android operating system requires that all installed applications be digitally signed with a certificate whose private key is held by the application's developer and uses the certificate as a means of identifying the author of an application and establishing trust relationships between applications. You can either use the default keystore file generated by Appeon or use the Keytool utility included in the Java Development Kits (JDK) to generate your private key. For details, refer to <http://developer.android.com/tools/publishing/app-signing.html>.

If you choose to use the keystore file provided by Appeon, you can bypass this section and go ahead to create the Android package, the Appeon keystore file will be automatically used by default.

Here is an example of a Keytool command that generates a private key (you can execute the command in any folder if the JAVA_HOME environment variable correctly points to the

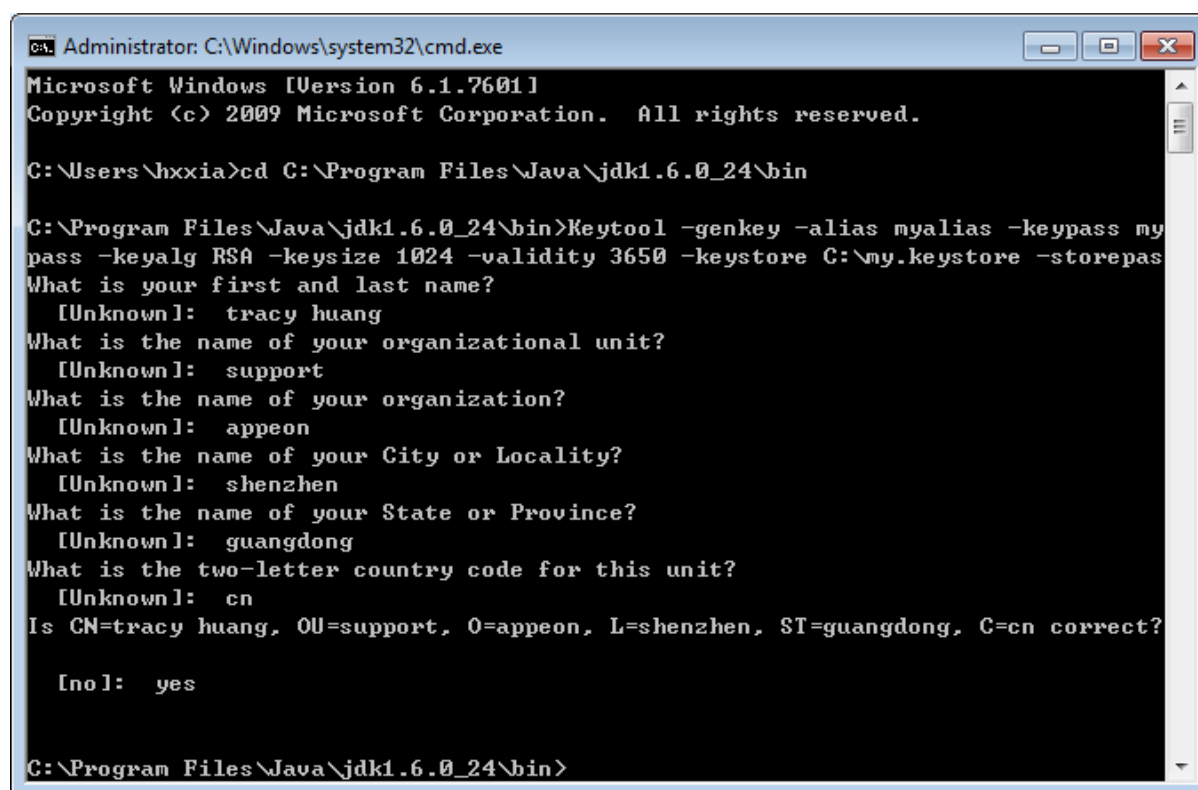
JDK directory, otherwise, you should first change to the JDK bin directory (using the "cd" command as shown below), and then execute the Keytool command there):

```
cd C:\Program Files\Java\jdk1.6.0_24\bin
```

```
Keytool -genkey -alias myalias -keypass mypass -keyalg RSA -keysize 1024 -validity  
3650 -keystore C:\my.keystore -storepass mypass
```

The Keytool command prompts you to provide the owner information.

Figure 10.32: Generate keystore



```
Administrator: C:\Windows\system32\cmd.exe
Microsoft Windows [Version 6.1.7601]
Copyright (c) 2009 Microsoft Corporation. All rights reserved.

C:\Users\hxxia>cd C:\Program Files\Java\jdk1.6.0_24\bin

C:\Program Files\Java\jdk1.6.0_24\bin>Keytool -genkey -alias myalias -keypass my  
pass -keyalg RSA -keysize 1024 -validity 3650 -keystore C:\my.keystore -storepas
What is your first and last name?
 [Unknown]: tracy huang
What is the name of your organizational unit?
 [Unknown]: support
What is the name of your organization?
 [Unknown]: apeon
What is the name of your City or Locality?
 [Unknown]: shenzhen
What is the name of your State or Province?
 [Unknown]: guangdong
What is the two-letter country code for this unit?
 [Unknown]: cn
Is CN=tracy huang, OU=support, O=apeon, L=shenzhen, ST=guangdong, C=cn correct?

 [no]: yes

C:\Program Files\Java\jdk1.6.0_24\bin>
```

After the keystore file is generated, remember the file name and location (C:\my.keystore in the above example) and the password, as you will need to include the keystore file when packaging the application later. To include the file, specify the relevant information in the [App Signing \[178\]](#) section in the **Apppeon Application Package Wizard**.

10.2.2.2 (Optional) Obtain a Google Maps API key

If you plan to provide the Google Maps functionality in the application, you would need to register for a Google Maps API key first and then include the key when packaging the application in the **Apppeon Application Package Wizard**. The key is free and you can use it with any of your applications that call the Map APIs (for how to call Map APIs, see Section 2.4.9, "Map" in *Workarounds & API Guide*), and it supports an unlimited number of users.

Below are instructions for generating the Google Map API key in the Google Developers Console.

1. Obtain the SHA1 fingerprint of the keystore file that will be used to sign the application.

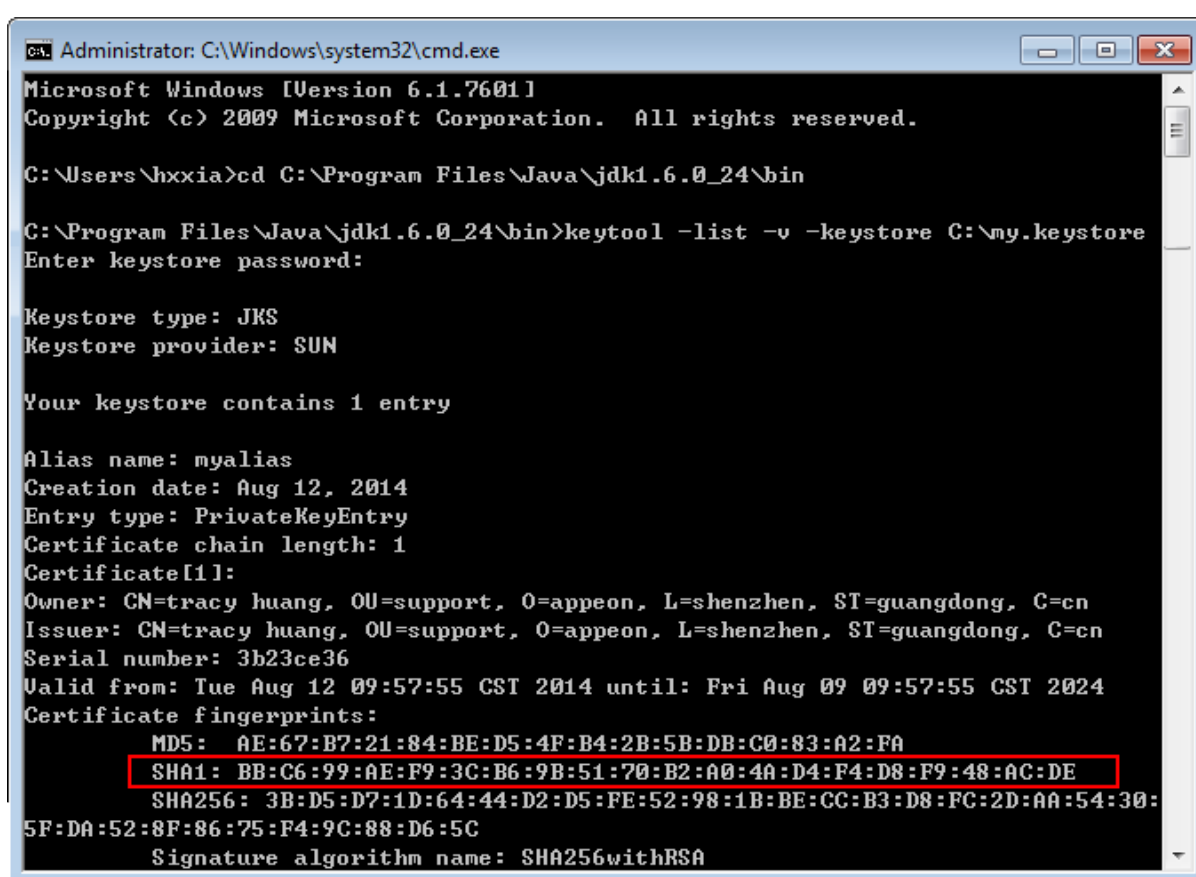
The keystore file should have already been generated, if not, follow instructions in the previous section [Obtain a private key](#) to generate a keystore file. Make sure you have installed JDK so the Keytool utility included in it is available to use.

Here is an example of a Keytool command that retrieves the SHA1 fingerprint of the keystore file (you can execute the command in any folder if the JAVA_HOME environment variable correctly points to the JDK directory, otherwise, you should first change to the JDK bin directory (using the "cd" command as shown below), and then execute the Keytool command there):

```
cd C:\Program Files\Java\jdk1.6.0_24\bin
```

```
keytool -list -v -keystore C:\my.keystore
```

Figure 10.33: SHA1 fingerprint



```
Administrator: C:\Windows\system32\cmd.exe
Microsoft Windows [Version 6.1.7601]
Copyright (c) 2009 Microsoft Corporation. All rights reserved.

C:\Users\hxxia>cd C:\Program Files\Java\jdk1.6.0_24\bin

C:\Program Files\Java\jdk1.6.0_24\bin>keytool -list -v -keystore C:\my.keystore
Enter keystore password:

Keystore type: JKS
Keystore provider: SUN

Your keystore contains 1 entry

Alias name: myalias
Creation date: Aug 12, 2014
Entry type: PrivateKeyEntry
Certificate chain length: 1
Certificate[1]:
Owner: CN=tracy huang, OU=support, O=appeon, L=shenzhen, ST=guangdong, C=cn
Issuer: CN=tracy huang, OU=support, O=appeon, L=shenzhen, ST=guangdong, C=cn
Serial number: 3b23ce36
Valid from: Tue Aug 12 09:57:55 CST 2014 until: Fri Aug 09 09:57:55 CST 2024
Certificate fingerprints:
    MD5: AE:67:B7:21:84:BE:D5:4F:B4:2B:5B:DB:C0:83:A2:FA
    SHA1: BB:C6:99:AE:F9:3C:B6:9B:51:70:B2:A0:4A:D4:F4:D8:F9:48:AC:DE
    SHA256: 3B:D5:D7:1D:64:44:D2:D5:FE:52:98:1B:BE:CC:B3:D8:FC:2D:AA:54:30:
5F:DA:52:8F:86:75:F4:9C:88:D6:5C
    Signature algorithm name: SHA256withRSA
```

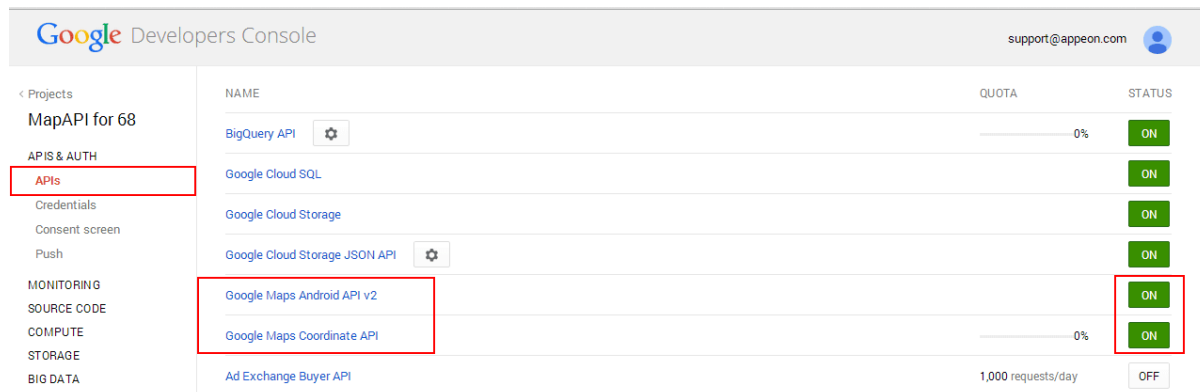
2. Turn on the Google Maps APIs.

You should register for a Google developer account first if you do not have one, as you will need to log into the Google Developers Console (<https://code.google.com/apis/console>) to turn on the Google Maps API and create the key.

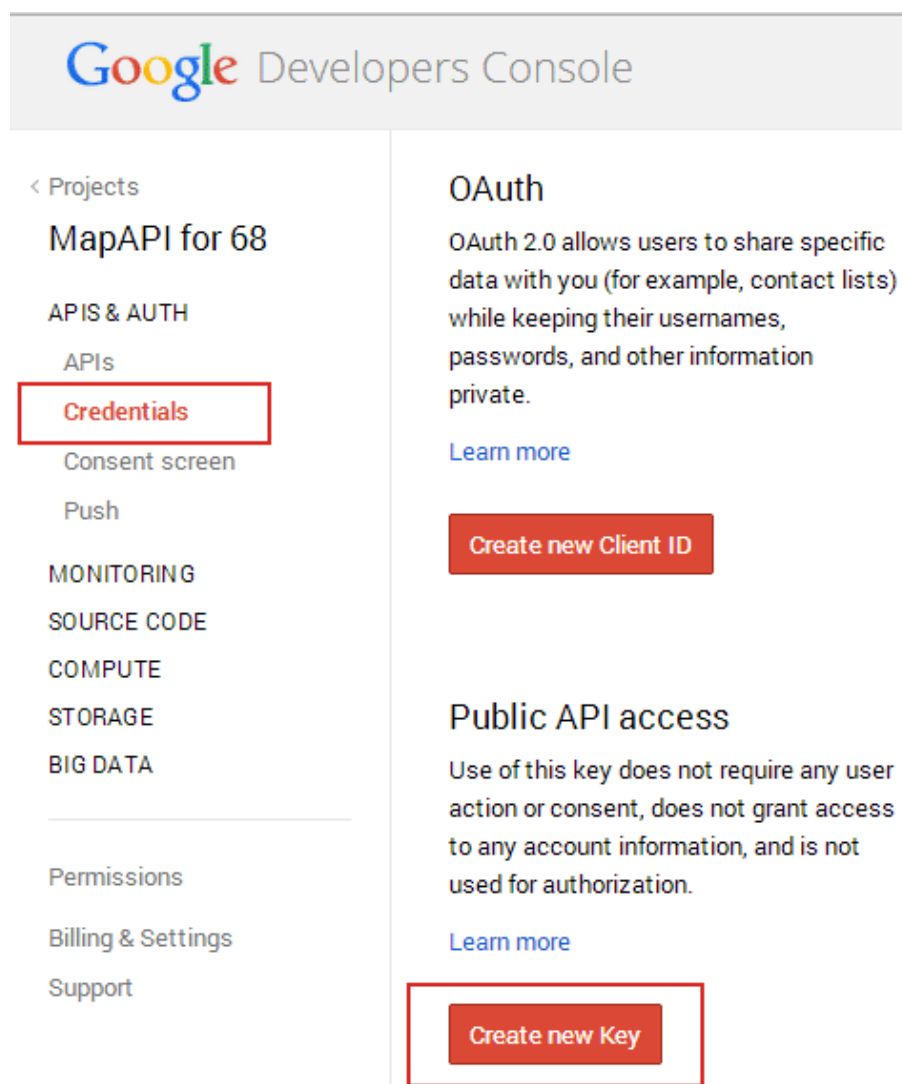
- a. Log into the Google Developers Console.
- b. Click **Create Project** to create a new project for Google Maps API, or use an existing project.

- c. Click **APIs** under **APIs & AUTH** on the left navigation pane, and then click the switch indicator on the right to turn on **Google Maps Android API v2** and **Google Maps Coordinate API** on the right side.

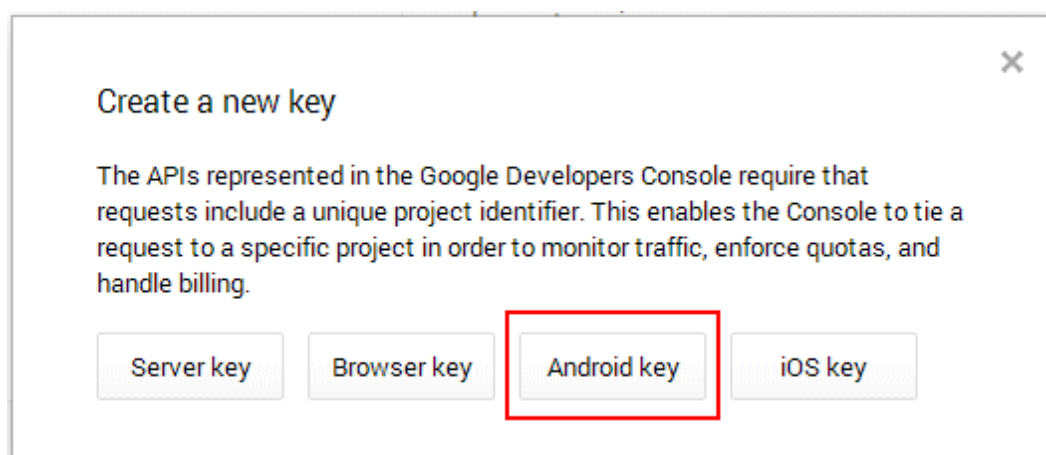
Figure 10.34: Create the key



3. Create the API key that binds with the specified application(s).
 - a. In the Google Developers Console, click **Credentials** under **APIs & AUTH** on the left navigation pane, and then click **Create new key** under **Public API access** on the right.

Figure 10.35: SHA1 fingerprint

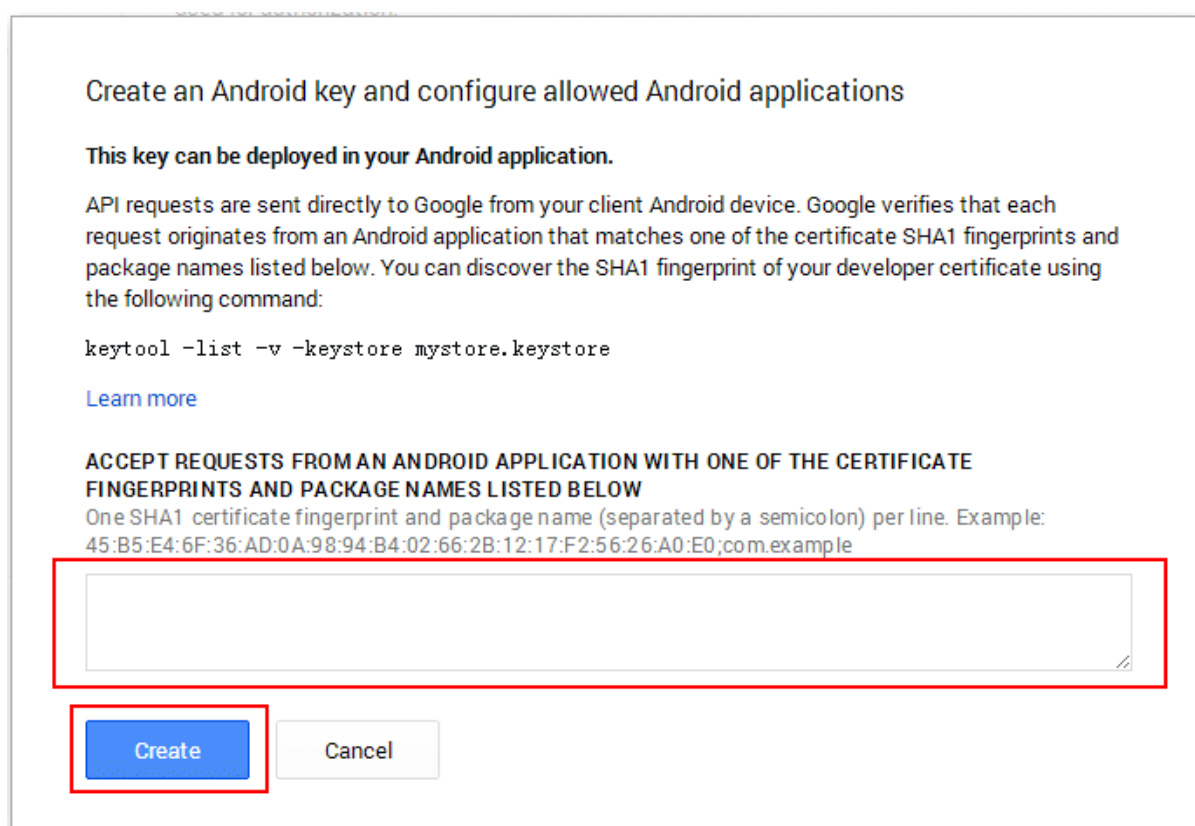
- b. In the popup **Create a new key** dialog, click **Android key**.

Figure 10.36: Create Android key

- c. Input the SHA1 fingerprint and the application name (separated by a semicolon) per line in the text box, and then click **Create**.

Notes:

- The same application name must be specified as the [App Identifier \[176\]](#) in the **Appeon Application Package Wizard** later.
- The generated API key should be included in the application. To do this, specify the key in the [Map API Key \[179\]](#) field in **Appeon Application Package Wizard** later.

Figure 10.37: Create the key**10.2.2.3 (Optional) Prepare the icons/images for your application**

You can provide your own app icons and splash/launch images for your Android application. If you do not provide the icons and images, the **Package Wizard** will use the default ones provided by Appeon.

Android powers a variety of devices with different screen sizes and densities, in order to get a proper UI look on those different screens, you may need to provide different app icons and splash/launch images for your app.

- a. App Icons: Refer to [App Icon \[179\]](#) for the size requirement for different Android screens.

- b. Splash/Launch Images: Refer to [Splash Image \[180\]](#) for the size requirement for different Android screens.

10.2.3 Preparations for the Appeon Workspace package

Except for the above two preparations, you may also need to do the following two preparations if you want to customize and package the Appeon Workspace, otherwise, the **Package Wizard** will use the default settings.

10.2.3.1 (Optional) Prepare your own banner

The banner in the Appeon Workspace is an HTML Web app. You can design your own banner and replace the Appeon Workspace Banner.

1. Banner size:

- a. For **iOS** devices: The height is 248 pixels for iPad (including iPad 2, 3, 4, mini and Air), and 124 pixels for iPhone (including iPhone 4, 4S, 5, 5C, 5S, 6, 6 Plus); the width is subject to the device (usually the same width as the device screen). Below is the width for different screens:

For iPad 2, 3, 4, and mini, the width in Portrait view is 768 pixels, and the width in Landscape view is 1024 pixels.

For iPad mini with retina display and iPad Air, the width in Portrait view is 1536 pixels, and the width in Landscape view is 2048 pixels.

For iPhone 4, 4S, 5, 5C, and 5S, the width in Portrait view is 640 pixels, and the width in Landscape view is 960 pixels for iPhone 4/4S, and 1136 pixels for iPhone 5/5C/5S.

For iPhone 6, the width in Portrait view is 750 pixels, and the width in Landscape view is 1334 pixels.

For iPhone 6 Plus, the width in Portrait view is 1080 pixels, and the width in Landscape view is 1920 pixels.

- b. For **Android-powered** devices: You need to design your own height (or you can use the recommended 248 pixels for tablets, and 124 pixels for smartphones) and the width is subject to the devices. You may need to write scripts to retrieve the size of the device screen first and then design your own HTML Web app as the Workspace Banner accordingly.

2. HTML file name:

Make sure that the HTML Web app contains an **index.html** file under the root directory of the app. The index.html file is the default file that the Workspace Banner loads.

3. Important Notes:

- a. Keep the banner height as 248 pixels for iPad, and 124 pixels for iPhone; and the width the same as the device screen.
- b. There are different widths for the portrait view and for the landscape view on a device if your workspace supports both orientations.

- c. Pay attention to the compatibilities of the related JavaScript.

10.2.3.2 (Optional) Prepare the UI language package

The workspace UI can be displayed in English, French, Simplified Chinese, Traditional Chinese, Japanese, Korean, Italian, and Spanish.

To make Appeon Workspace display in other language as well, you will need to add the other language packages before you package Appeon Workspace:

1. Navigate to one of the following folders under the PowerServer Toolkit installation directory according to your package type.
 - a. For **iOS** packages: ...\\Appeon\\PowerServer\\Toolkit\\AppTemplate\\iOS\\Xcode8\\AppeonWS.bundle\\config\\Aws\\;
 - b. For **Android** packages: ...\\Appeon\\PowerServer\\Toolkit\\AppTemplate\\Android\\AndroidNativeApp\\assets\\config\\Aws\\.

Open the **languages.xml** file in a text editor and uncomment the line of script that corresponds to the language you want to add. It is recommended that you back up this file before modifying it.

For example, to add the support for German, find the corresponding line of script (or add it if it is not pre-defined), and then remove the comment as shown below:

```
<lang code="de" name="Deutsch" c="##" />
```

The value of **code** must be used as the file name for the language in the next step.

The value of **name** will be used as the display name for the language in the **Language** tool of Appeon Workspace later.

2. Make a copy of an existing language file under one of the following folders (according to your package type) under the PowerServer Toolkit installation directory.
 - a. For **iOS** packages: ...\\Appeon\\PowerServer\\Toolkit\\AppTemplate\\iOS\\Xcode8\\AppeonWS.bundle\\config\\Aws\\lang\\;
 - b. For **Android** packages: ...\\Appeon\\PowerServer\\Toolkit\\AppTemplate\\Android\\AndroidNativeApp\\assets\\config\\Aws\\lang\\.
- Put the copy under the same directory, and name the copy after the value of **code**. Take German for example, the language file must be named as *de.xml*.
- Then open the file, and translate the content into the language you want to add.
3. In the **Package Wizard**, when you click the ellipse button in the **Default Language** field, the language you added will be displayed for selection.

10.2.3.3 (Optional) Customize the UI text

The workspace UI text can be customized according to your needs. You will need to make the change before you package Appeon Workspace.

To customize the UI text, you edit the language file under one of the following folders (according to your package type) under the PowerServer Toolkit installation directory.

- For **iOS** packages: ...\\Appeon\\PowerServer\\Toolkit\\AppTemplate\\iOS\\Xcode8\\AppeonWS.bundle\\config\\Aws\\lang\\;
- For **Android** packages: ...\\Appeon\\PowerServer\\Toolkit\\AppTemplate\\Android\\AndroidNativeApp\\assets\\config\\Aws\\lang\\.

For example, you can open en.xml and change the text (id="1") from "Appeon Workspace" to "ABC Workspace", so "ABC Workspace" will display in the center of workspace titlebar instead of "Appeon Workspace".

The following figures show most of the customizable texts and the matching IDs in the language file.

Figure 10.38: Customizable UI Text

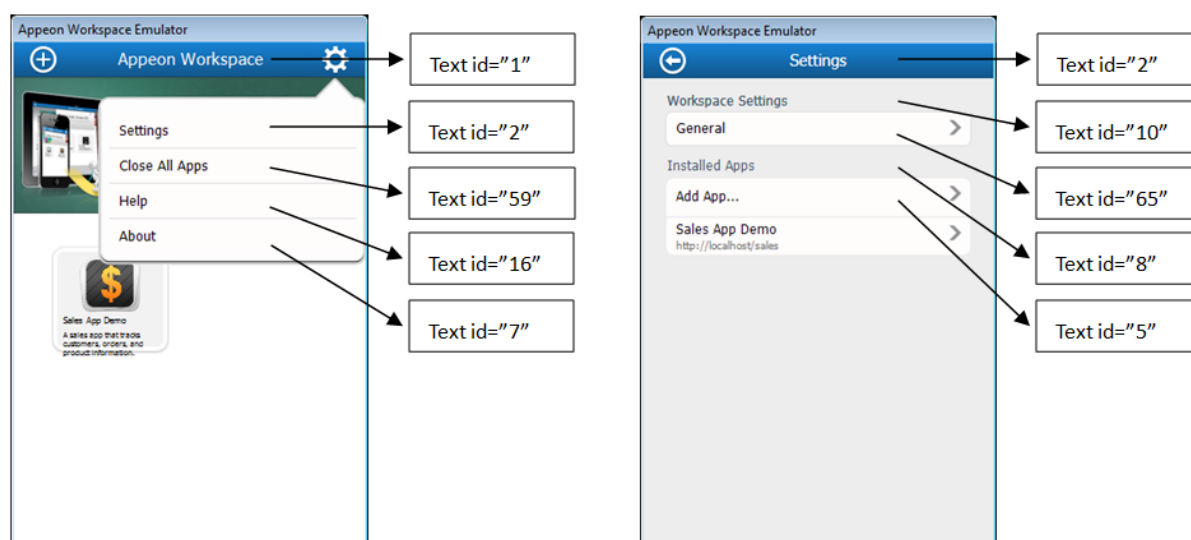


Figure 10.39: Customizable UI Text

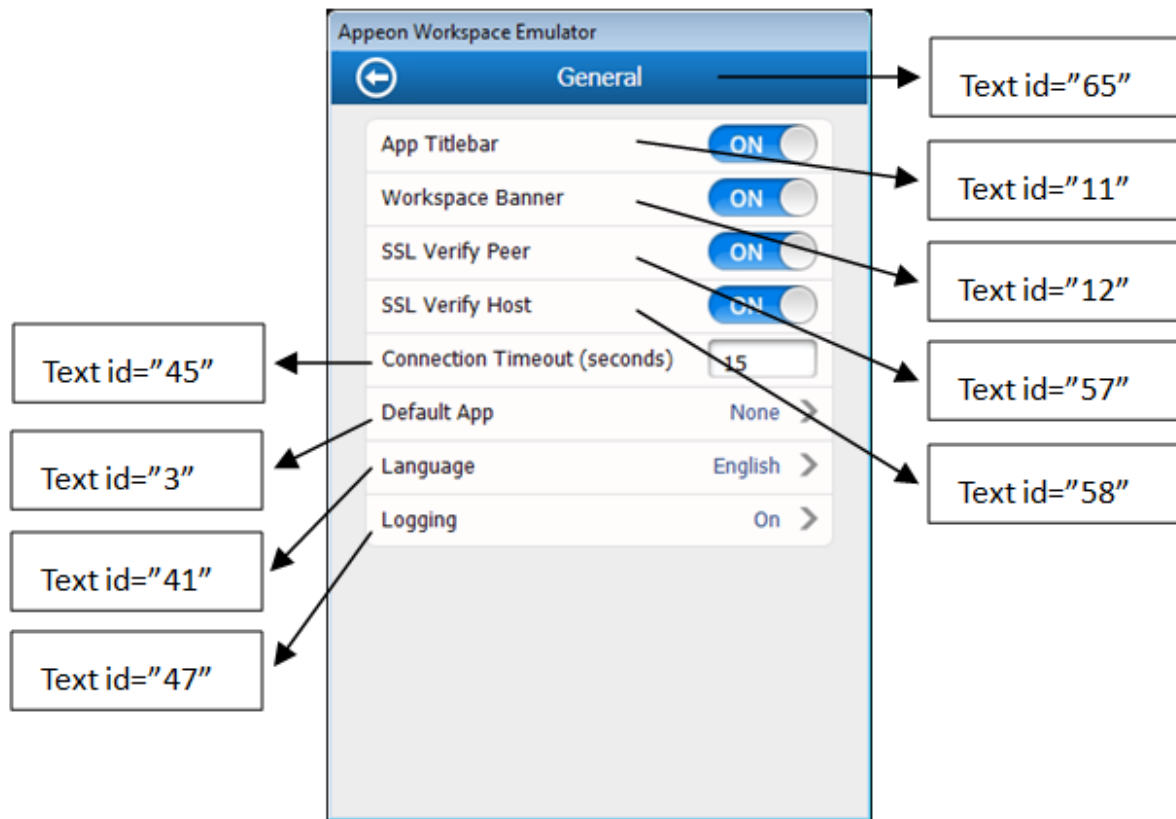
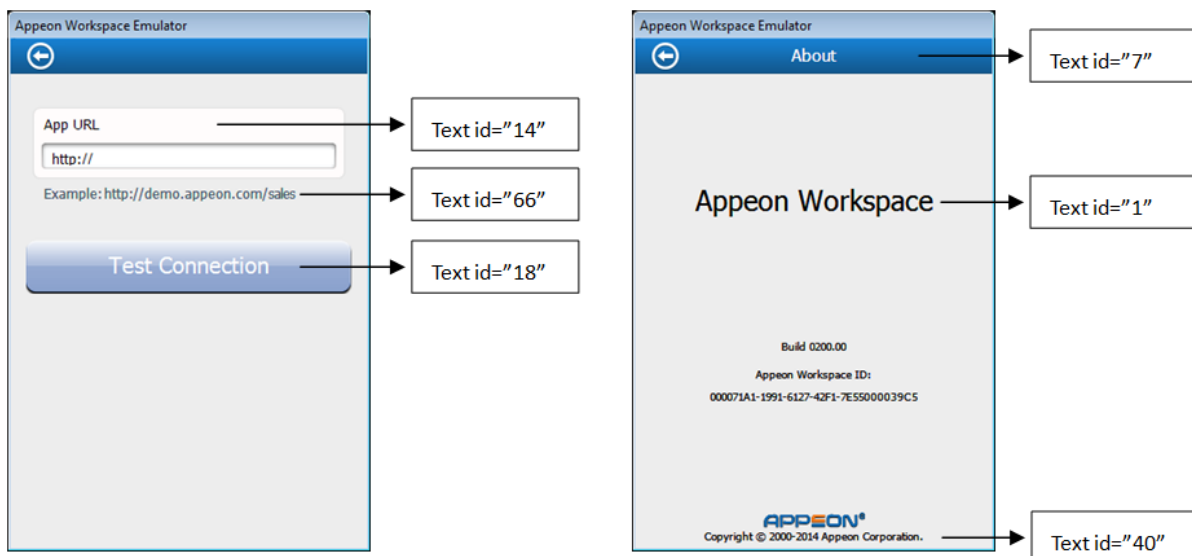


Figure 10.40: Customizable UI Text



10.3 Packaging a stand-alone mobile project

10.3.1 Points to check before packaging

After packaging the mobile application into an iOS IPA file or Android APK file, you will need to upload the IPA or APK file to the online application stores to get the app approved.

For the iOS IPA file, Apple follows strict guidelines to review the app before it approves and posts the app to the Apple App Store. To get the app approved, you may need to carefully read the [App Store Review Guidelines](#) (you will need to log in first to view the page) and verify the app is in compliance with these guidelines.

The following lists the most common areas and reasons we have found so far that will cause the app to be rejected by Apple.

1. Check that if your app includes UI controls for quitting the app. For example, a button labeling "Quit", "Exit", etc. You'd better remove any mechanism for quitting the app, as this is not in compliance with the [iOS Human Interface Guidelines](#), as required by the [App Store Review Guidelines](#) (you will need to log in first to view the page).
2. Check that if your app appears to be for demonstration or trial purposes only. For example, "demo", "trial", "test" or "beta" is displayed on the app UI, app description or release notes. You'd better remove any these references and complete or fully configure any partially implemented features.

10.3.2 What will be packaged?

The package wizard will generate different packages for iOS and Android.

- For iOS, the package wizard will generate an executable Xcode project which consists of the following packages:
 - i. *App Name.zip*

Including all of the application files, DataWindow syntax, image files, etc. for the iOS application, and also including an Xcode project file and the Xcode project settings which will be run later to compile the IPA file.
 - ii. *AppeonMobile.framework.zip*

Including the mobile client libraries of the Appeon Workspace. The client libraries support the running of the stand-alone mobile app, such as rendering the mobile-style UI, providing interfaces for calling the mobile SDK, supporting the offline capabilities, etc.

Copy the above packages to a Mac machine with Xcode installed, and then compile them into an IPA file.
- For Android, the package wizard will directly generate an APK file which can be installed on the Android device right away:
 - i. *App Name_install_Android.apk*

Including all of the application files, DataWindow syntax, image files, etc. for the mobile application.

10.3.3 Packaging instructions

Before you start, make sure you have completed the preparation tasks in [Preparing for the mobile package](#).

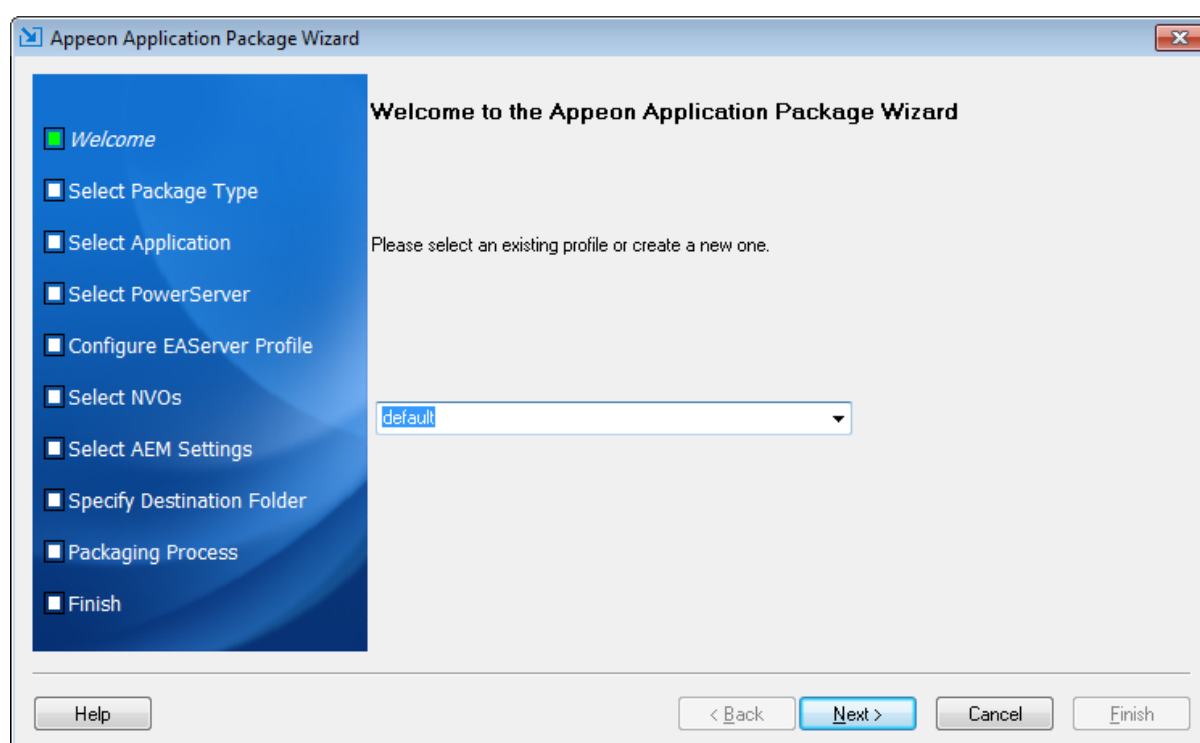
To package an Xcode project or an Android application package (APK) file, follow the steps below:

Step 1: Click the **Package** (📦) button on the PowerServer Toolkit to open the **Appeon Application Package Wizard**.

Step 2: Select or create a profile from the dropdown list and click **Next** to proceed.

A profile is a configuration file containing the settings that you specify when packaging the application. You can select an existing profile or create one by entering a name in the text field. The profile will be automatically saved and listed for selection next time when you launch the **Appeon Application Package Wizard** again.

Figure 10.41: Welcome page

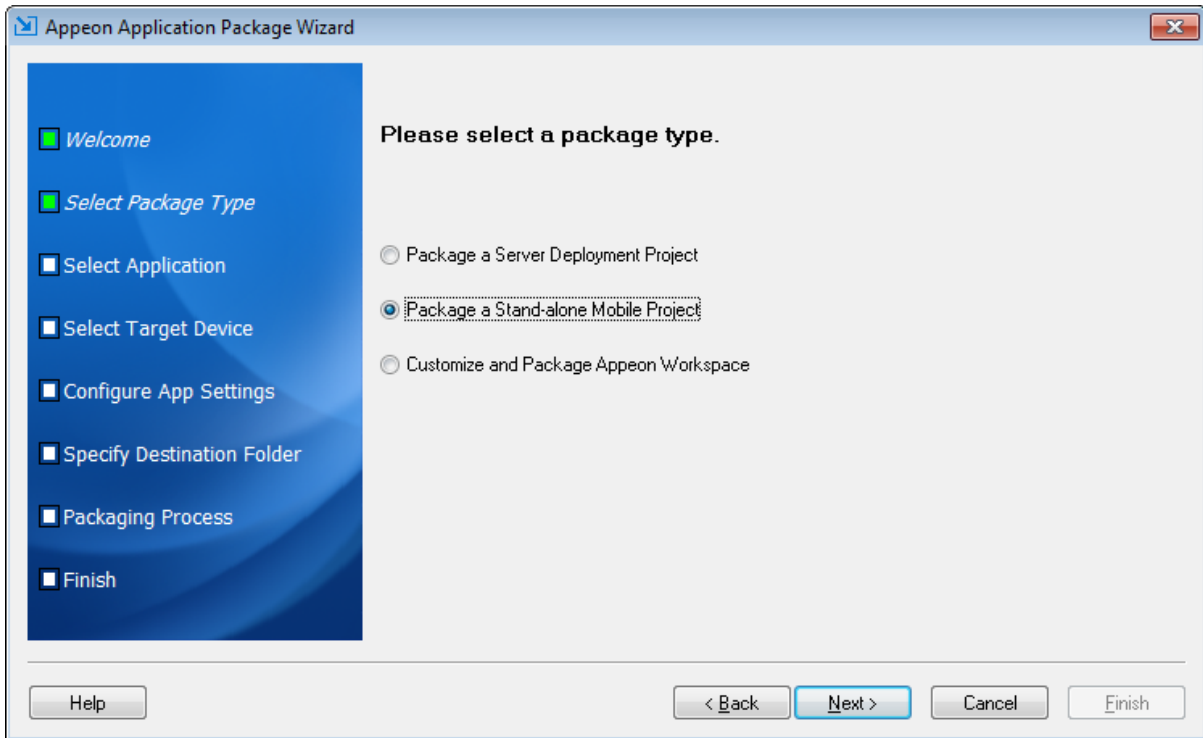


Step 3: Select the **Package a Stand-alone Mobile Project** radio button.

To package a deployment project that can deploy the application to servers, select **Package a Server Deployment Project** and follow detailed instructions in [Packaging a server deployment project](#).

To customize Appeon Workspace and then package Appeon Workspace to a standalone application, select **Customize and Package Appeon Workspace** and follow detailed instructions in [Customizing and packaging Appeon Workspace](#).

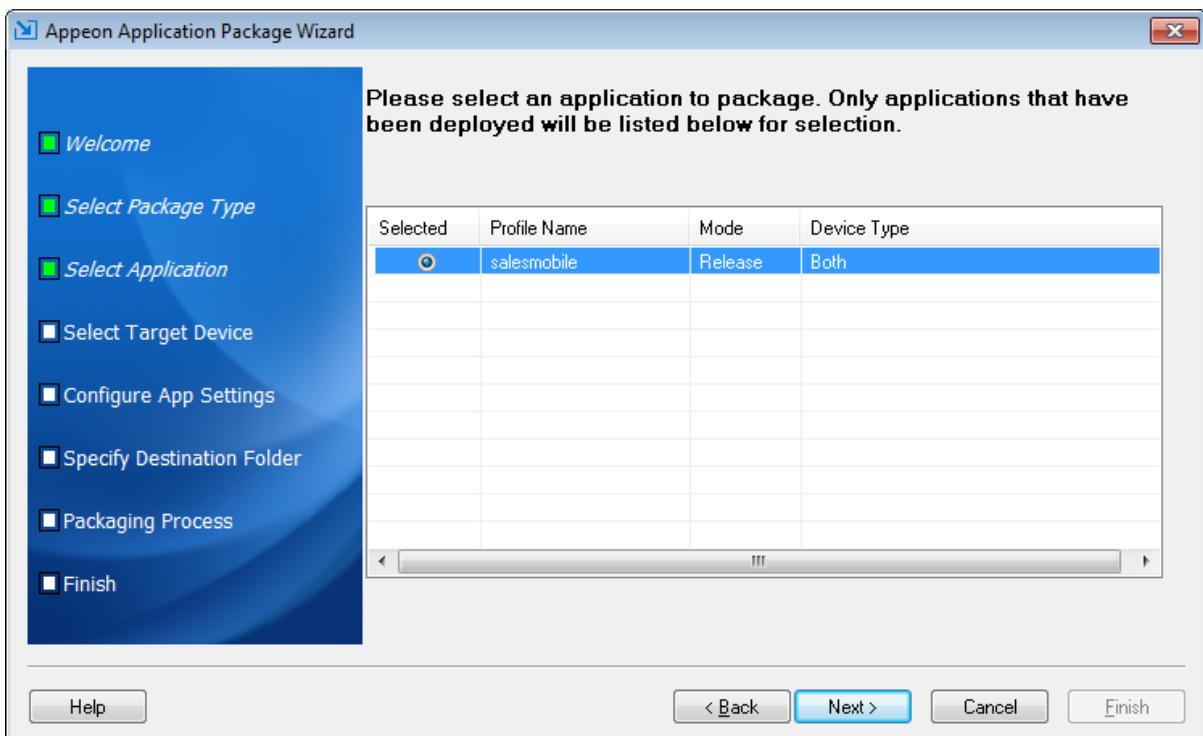
Figure 10.42: Select package type



Step 4: Select the profile of the application that you want to package and click **Next**.

Only applications that have been deployed will be listed here for selection. If the application you intend to package is not listed here, you would need to deploy the target application using the **Appeon Deployment Wizard** first (See [Chapter 6, Deploying PowerBuilder Applications](#)).

Figure 10.43: Select apps to be packaged



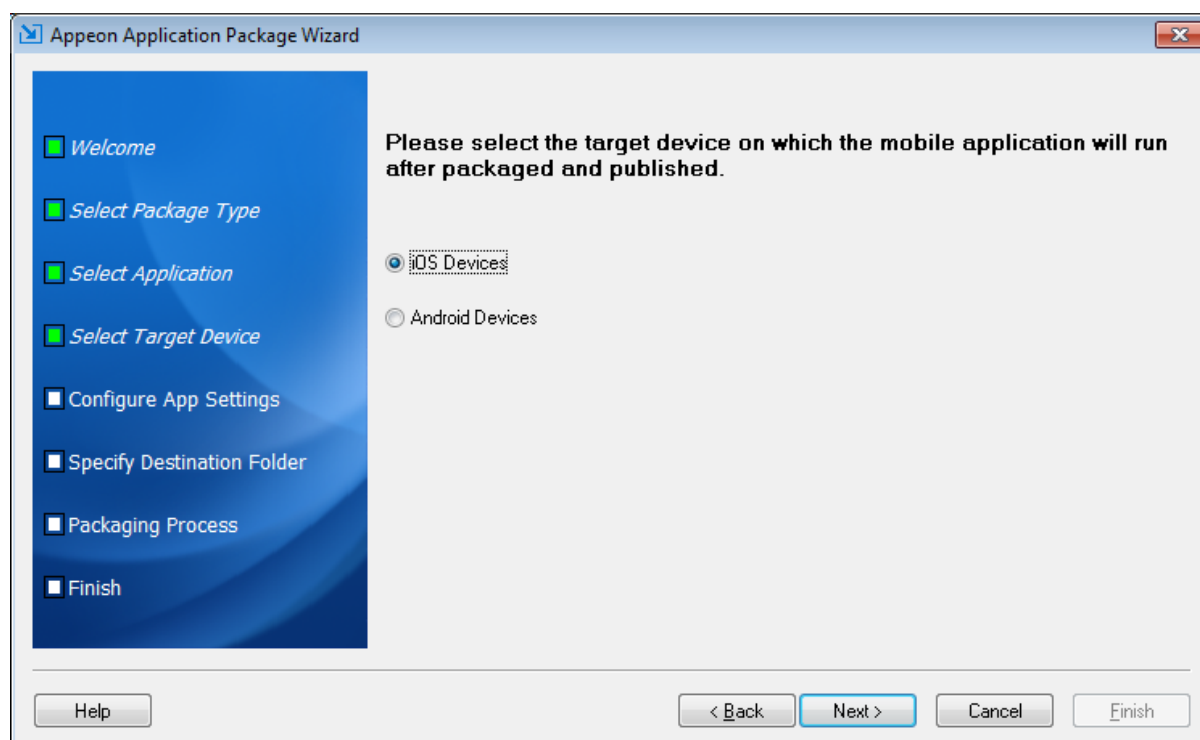
The following table gives a brief introduction to the columns:

Table 10.2: Application Package Wizard

Column	Description
Selected	Identical to the default application profile selected in the PowerServer Toolkit Configuration window.
Profile Name	Identical to the application profiles configured in the PowerServer Toolkit Configuration window.
Mode	Identical to the current mode the application profile is set to in the PowerServer Toolkit Configuration window. It is recommended that the packaged applications should be available in Release mode, as this mode prevents most forms of reverse engineering, and like application packaging, it protects the author's intellectual property.
Device Type	Identical to the current device type settings in the PowerServer Toolkit Configuration window.

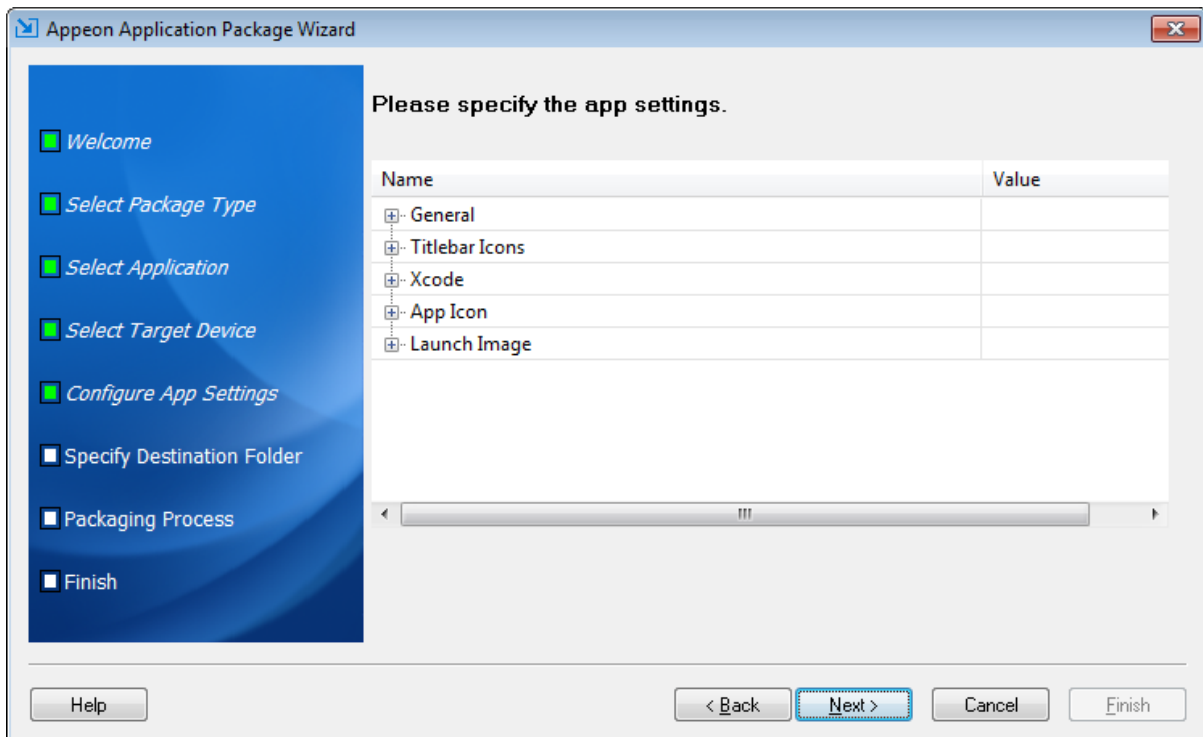
Step 5: Select the device type (iOS or Android) on which the application will run after packaged and then click **Next**.

Figure 10.44: Select a target device type



Step 6: Specify the parameters according to the device type and then click **Next**.

- For apps running on iOS, specify the parameter according to the following table.

Figure 10.45: Configure parameters for the Xcode project**Table 10.3: Parameters for the Xcode project**

Group	Value	Description
General	App Name (Required)	<p>Due to a limitation of the package tool, the app name cannot contain double-byte characters (such as Chinese, Korean, or Japanese characters).</p> <p>The app name will be used in two areas:</p> <ol style="list-style-type: none"> part of the file name of the generated package. <ul style="list-style-type: none"> For the app to have the auto-upgrade feature, you should always use the same app name every time when creating the package for the same application. the name of your app that will appear in the Apple App Store. <ul style="list-style-type: none"> For the name to be displayed completely, it should be about 12 letters but it depends on the width of each individual letter, for example, w takes more room than i.
	App Description	Provide the description of your app that will appear in the screen under iOS Settings > <i>App Name</i> after the app is installed on the iOS-based device.

Group	Value	Description
	App URL (Required)	<p>Specify the URL to the mobile application that is deployed to the PowerServer, for example, http://192.0.0.201/sales. For more information, see URLs of Appeon applications.</p> <p>The URL supports both HTTP and HTTPS formats, for example, https://192.0.0.201/sales.</p> <p>The URL can be changed in the screen under iOS Settings > App Name after the app is installed on the iOS-based device.</p> <p>Make sure the URL is valid, as the mobile application will connect with the PowerServer via this URL at runtime.</p>
	App ID Prefix (Required)	<p>Specify the application's App ID Prefix, and it must be the same App ID Prefix used in the provisioning profile for building the Xcode project later.</p> <p>The App ID Prefix is a 10-character hexadecimal string automatically generated when you generate the App ID in the Apple Developer Member Center. It is unique to you and your developer account.</p> <p>Keep using the same App ID Prefix for different apps, so that these different apps running on the same device can be recognized as running on one device, not on multiple devices by the Appeon license file. For details on the Device control type in the Appeon license file, refer to Section 5.3.4, "Product Activation" in <i>PowerServer Configuration Guide for .NET</i> or in <i>PowerServer Configuration Guide for J2EE</i>.</p>
	App ID Suffix (Bundle ID) (Required)	<p>Specify the application's App ID Suffix (also called Bundle ID).</p> <p>The App ID Suffix is a name you enter called the Bundle Identifier when you generate the App ID in the Apple Developer Member Center. The Bundle Identifier can be explicit or a wildcard. If you have specified a wildcard bundle identifier when creating the App ID, for example, com.abcxample.*, you will need to replace "*" with an explicit string, for example, com.abcxample.app or com.abcxample.1 etc.</p> <p>The App ID Suffix is also used as part of the file name of the generated package. To help your</p>

Group	Value	Description
		app's auto-upgrade feature to work, you should keep this App ID Suffix unchanged every time when creating the package for the same app.
	Startup Orientation	Set the initial interface orientation when the app starts.
	SSL Verify Peer	Enable or disable the SSL peer verification.
	SSL Verify Host	Enable or disable the SSL host name verification.
	Connection Timeout (seconds)	Specify the timeout seconds for your app connecting to the server.
	Record Logs	Set whether to record app logs.
	URL Scheme Name	<p>A URL scheme is a specially formatted URL that allows users to open your app from other Appeon mobile apps or third-party apps. A typical URL scheme looks like this: <i>schemename://?url=http://ipaddress/appname[&parm1=aaa&parm2=bbb...&parmN=zzz]</i></p> <p><i>schemename</i> is what you specify in the URL Scheme Name field as the unique scheme identifier for the application to be packaged. The scheme name is highly recommended to be unique; if you use the same scheme name as another app, iOS or Android will not know which app to launch. Scheme name is case sensitive; it can only contain lower-case letters and numbers, and can only start with lower-case letters.</p> <p><i>[&parm1=aaa&parm2=bbb...&parmN=zzz]</i> is the parameter name and value pair(s) to be passed to the destination app. It is optional. When it is passed to the Appeon mobile app, it can be obtained by the app using APIs. For more about the APIs, refer to the section called “of_geturlschemeparm” in <i>Workarounds & API Guide</i> and the section called “oe_urlschemesucceed” in <i>Workarounds & API Guide</i>.</p> <p>At the time of writing this note, URL scheme is only supported in Safari and Opera Mini. If you want to use the other browser to open an app via URL scheme, please double check that the browser is supported.</p> <p>Examples</p> <p>The Appeon Workspace published in the online app store (e.g. Apple App Store, Google Play)</p>

Group	Value	Description
		<p>by Appeon has <i>appeonaws</i> as its scheme name, therefore the URL scheme to open this Appeon Workspace is: <code>appeonaws://</code></p> <p>Suppose you specify <i>ABCaws</i> as the scheme name for your workspace, then the URL scheme to open your workspace will be: <code>ABCaws://</code></p> <p>And the URL scheme to open the app XXX (suppose its app URL is <code>http://demo.appeon.com/xxx</code>) installed in your workspace will be: <code>ABCaws://?url=http://demo.appeon.com/xxx</code></p> <p>Suppose you specify <i>ABCyyy</i> as the scheme name for the standalone mobile app YYY, then the URL scheme to open YYY will be: <code>ABCyyy://?url=http://192.0.3.111/yyy&name=jacky age=27</code></p>
	Support Workspace Auto-Upgrade	<p>If the workspace will be uploaded to the online app store such as Apple App Store, Google Play etc., this option should be disabled, because the workspace's auto-upgrade will be controlled by the online app store. If the workspace will be provided on your own Web site, we recommend you enable this option as well as the "Check for Workspace Updates" option, so that the workspace can be automatically upgraded.</p> <p>There are other factors affecting the auto-upgrade feature of workspace, refer to Section 3.2.1, "Enabling auto-upgrade of Appeon Workspace" in <i>Appeon Workspace User Guide</i>.</p>
	Check for Workspace Updates	<p>Set whether to automatically check with the server if updates of workspace are available. If updates are detected, the end user will be prompted whether to install the update.</p>
Titlebar Icons	Full Screen by Default	<p>Whether to display the app in full screen view (with the titlebar hidden) by default when it is opened. In the full scree view, the normal view icon will be available on the top right corner of the window, and when it is tapped, the application will return to the normal view (with the titlebar visible).</p>
Xcode	App Major Version	Specify the version number for your app.
	App Build No.	Specify the build number for your app.

Group	Value	Description
	Xcode Version	Select the Xcode version which will be used to compile the application. For now, only Xcode 8 is selectable.
App Icon	iPhone Settings (2x)	Specify the settings icon for the retina display on iPhone on iOS 9/10 (58 x 58 pixels).
	iPhone Settings (3x)	Specify the settings icon for the retina HD display on iPhone on iOS 9/10 (87 x 87 pixels).
	iPhone Spotlight (2x)	Specify the spotlight icon for the retina display on iPhone on iOS 9/10 (80 x 80 pixels).
	iPhone Spotlight (3x)	Specify the spotlight icon for the retina HD display on iPhone on iOS 9/10 (120 x 120 pixels).
	iPhone App (2x)	Specify the app icon for the retina display on iPhone on iOS 9/10 (120 x 120 pixels).
	iPhone App (3x)	Specify the app icon for the retina HD display on iPhone on iOS 9/10 (180 x 180 pixels).
	iPad Settings (1x)	Specify the settings icon for the standard display on iPad on iOS 9/10 (29 x 29 pixels).
	iPad Settings (2x)	Specify the settings icon for the retina display on iPad on iOS 9/10 (58 x 58 pixels).
	iPad Spotlight (1x)	Specify the spotlight icon for the standard display on iPad on iOS 9/10 (40 x 40 pixels).
	iPad Spotlight (2x)	Specify the spotlight icon for the retina display on iPad on iOS 9/10 (80 x 80 pixels).
	iPad App (1x)	Specify the app icon for the standard display on iPad on iOS 9/10 (76 x 76 pixels).
	iPad App (2x)	Specify the app icon for the retina display on iPad on iOS 9/10 (152 x 152 pixels).
Launch Image	iPhone Portrait (Retina HD 5.5)	Specify the launch image for the retina HD display 5.5-inch in portrait view on iPhone on iOS 9/10 (1242 x 2208 pixels).
	iPhone Portrait (Retina HD 4.7)	Specify the launch image for the retina HD display 4.7-inch in portrait view on iPhone on iOS 9/10 (750 x 1334 pixels).
	iPhone Landscape (Retina HD 5.5)	Specify the launch image for the retina HD display 5.5-inch in landscape view on iPhone on iOS 9/10 (2208 x 1242 pixels).
	iPhone Portrait (2x)	Specify the launch image for the retina display in portrait view on iPhone on iOS 9/10 (640 x 960 pixels).

Group	Value	Description
	iPhone Portrait (Retina 4)	Specify the launch image for the retina display 4-inch in portrait view on iPhone on iOS 9/10 (640 x 1136 pixels).
	iPad Portrait (1x)	Specify the launch image for the standard display in portrait view on iPad on iOS 9/10 (768 x 1024 pixels).
	iPad Portrait (2x)	Specify the launch image for the retina display in portrait view on iPad on iOS 9/10 (1536 x 2048 pixels).
	iPad Landscape (1x)	Specify the launch image for the standard display in landscape view on iPad on iOS 9/10 (1024 x 768 pixels).
	iPad Landscape (2x)	Specify the launch image for the retina display in landscape view on iPad on iOS 9/10 (2048 x 1536 pixels).

- For apps running on Android, specify the parameter according to the following table.

Figure 10.46: Configure parameters for the Android APK file

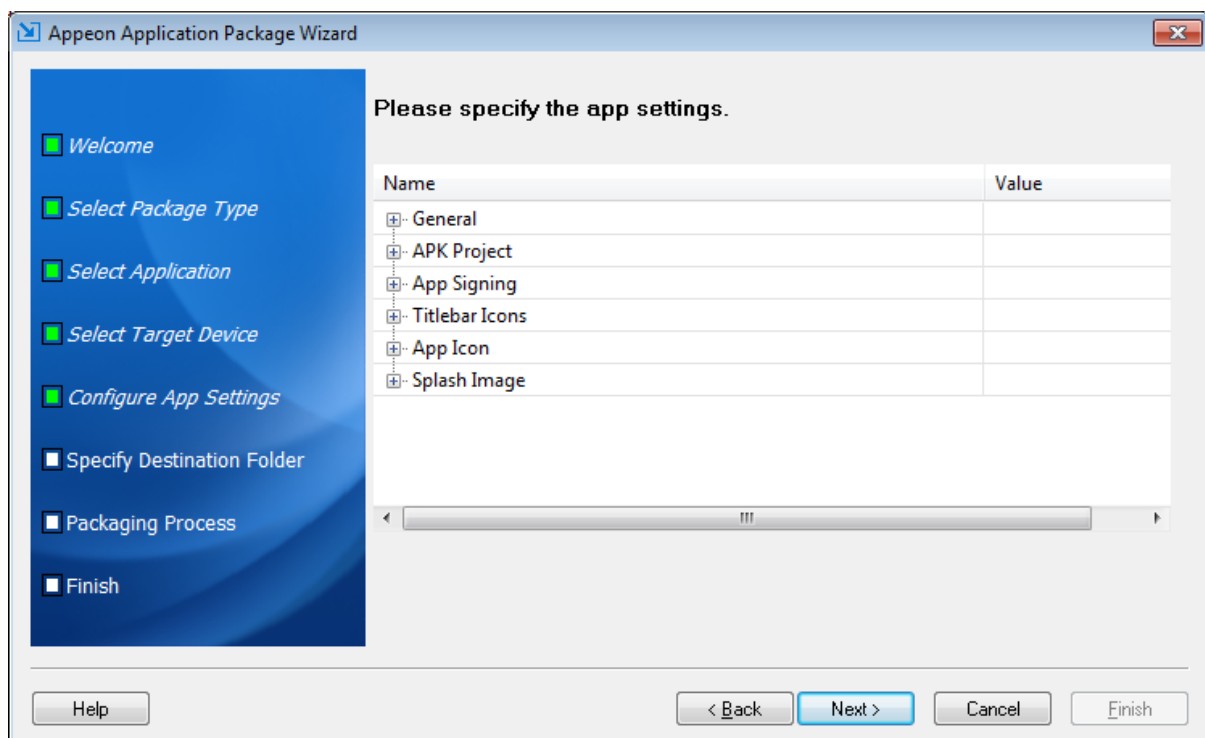


Table 10.4: Parameters for the Android APK file

Group	Value	Description
General	App Name (Required)	Due to a limitation of the package tool, the app name cannot contain double-byte characters (such as Chinese, Korean, or Japanese characters).

Group	Value	Description
		<p>The app name will be used in two areas:</p> <ol style="list-style-type: none"> 1. the name of the generated package. <p>For the app to have the auto-upgrade feature, you should always use the same app name every time when creating the package for the same application.</p> <ol style="list-style-type: none"> 2. the name of your app that will appear in the Android marketplace.
	App URL (Required)	<p>Specify the URL to the mobile application that is deployed to the PowerServer, for example, http://192.0.0.201/sales. For more information, see URLs of Appeon applications.</p> <p>The URL supports both HTTP and HTTPS formats, for example, https://192.0.0.201/sales.</p>
	App Identifier (Required)	<p>Specify the identifier for the app(s).</p> <p>This field is used to identify the app, and when the app is to be installed on a device where another app with the same identifier has already been installed, the installation will fail.</p> <p>The app identifier is also used as part of the file name of the generated package. To help your app's auto-upgrade feature to work, you should keep this app identifier unchanged every time when creating the package for the same app.</p> <p>This app identifier must follow the same rules as those for naming packages in the Java programming language. It can only contain lower-case letters, dots, and/or numbers without spaces, and cannot start or end with dots or contain only numbers between dots. It is normally a combination of reversed domain name and the app name (com.companyname.appname), such as, com.appeon.sales, net.abc.helloworld, org.xyz.myfirstapp etc.</p>
	App Version Code	<p>Specify an integer value that uniquely identifies the APK file of your app to be uploaded to Google Play. It is used by Google Play for internal purpose and it is invisible to the end user. You can set the value to any integer you want, but each time when you are going to</p>

Group	Value	Description
		upload an updated APK file for your app, make sure you increase it to a greater integer value.
	App Version Name	Specify a string value that represents the release version of your app. It will be displayed to the end user.
	Startup Orientation	Set the initial interface orientation when the app starts.
	SSL Verify Peer	Enable or disable the SSL peer verification.
	SSL Verify Host	Enable or disable the SSL host name verification.
	Connection Timeout (seconds)	Specify the timeout seconds for your app connecting to the server.
	Record Logs	Set whether to record app logs.
	URL Scheme Name	<p>A URL scheme is a specially formatted URL that allows users to open your app from other Appeon mobile apps or third-party apps. A typical URL scheme looks like this: <i>schemename://?url=http://ipaddress/appname[&parm1=aaa&parm2=bbb...&parmN=zzz]</i></p> <p><i>schemename</i> is what you specify in the URL Scheme Name field as the unique scheme identifier for the application to be packaged. The scheme name is highly recommended to be unique; if you use the same scheme name as another app, iOS or Android will not know which app to launch. Scheme name is case sensitive; it can only contain lower-case letters and numbers, and can only start with lower-case letters.</p> <p><i>[&parm1=aaa&parm2=bbb...&parmN=zzz]</i> is the parameter name and value pair(s) to be passed to the destination app. It is optional. When it is passed to the Appeon mobile app, it can be obtained by the app using APIs. For more about the APIs, refer to the section called “of_geturlschemeparm” in <i>Workarounds & API Guide</i> and the section called “oe_urlschemesucceed” in <i>Workarounds & API Guide</i>.</p> <p>At the time of writing this note, URL scheme is only supported in Safari and Opera Mini. If you want to use the other browser to open an app via URL scheme, please double check that the browser is supported.</p> <p>Examples</p>

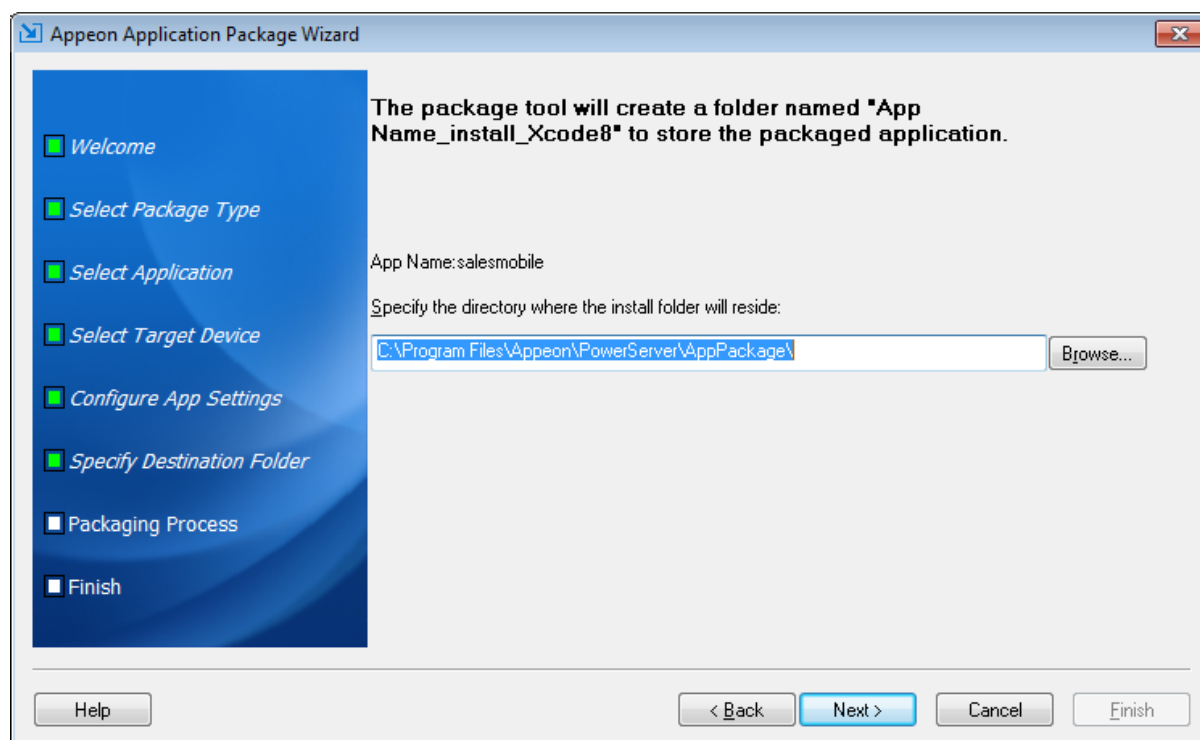
Group	Value	Description
		<p>The Appeon Workspace published in the online app store (e.g. Apple App Store, Google Play) by Appeon has <i>appeonaws</i> as its scheme name, therefore the URL scheme to open this Appeon Workspace is: <code>appeonaws://</code></p> <p>Suppose you specify <i>ABCaws</i> as the scheme name for your workspace, then the URL scheme to open your workspace will be: <code>ABCaws://</code></p> <p>And the URL scheme to open the app XXX (suppose its app URL is <code>http://demo.appeon.com/xxx</code>) installed in your workspace will be: <code>ABCaws://?url=http://demo.appeon.com/xxx</code></p> <p>Suppose you specify <i>ABCyyy</i> as the scheme name for the standalone mobile app YYY, then the URL scheme to open YYY will be: <code>ABCyyy://?url=http://192.0.3.111/yyy&name=jacky age=27</code></p>
	Support Workspace Auto-Upgrade	<p>If the workspace will be uploaded to the online app store such as Apple App Store, Google Play etc., this option should be disabled, because the workspace's auto-upgrade will be controlled by the online app store. If the workspace will be provided on your own Web site, we recommend you enable this option as well as the "Check for Workspace Updates" option, so that the workspace can be automatically upgraded.</p> <p>There are other factors affecting the auto-upgrade feature of workspace, refer to Section 3.2.1, "Enabling auto-upgrade of Appeon Workspace" in <i>Appeon Workspace User Guide</i>.</p>
	Check for Workspace Updates	<p>Set whether to automatically check with the server if updates of workspace are available. If updates are detected, the end user will be prompted whether to install the update.</p>
APK Project	Delete project after packaged	<p>Whether to delete the APK project after the app is packaged.</p> <p>If not, there will be a folder named after the App Name under <code>...\PowerServerToolkit\AppTemplate\Android\NativeConfig\</code>.</p>
App Signing	Alias (Required)	<p>Enter the alias name for key.</p> <p>Only the first 8 characters of the alias name are used.</p>

Group	Value	Description
		You can use the default alias name generated by Appeon (which is <i>appeon</i>) or use your own name. If you use the default keystore file in the Directory field, you need to keep this field as default.
	Alias Password	Enter the key password. You can use the default alias password generated by Appeon (which is <i>appeon</i>) or use your own password. If you use the default keystore file in the Directory field, you need to keep this field as default.
	Directory (Required)	Click the browse button (...) to select the keystore file. You can use the default keystore file or click the browse button to select your own one. For how to create the keystore file, refer to Obtain a private key .
	Keystore Password	Enter the keystore password. You can use the default keystore password generated by Appeon (which is <i>appeon</i>) or use your own password. If you use the default keystore file in the Directory field, you need to keep this field as default.
	Map API Key	Specify the Google Map API key. The Map APIs (see Section 2.4.9, “Map” in <i>Workarounds & API Guide</i>) read the key value and pass it to the Google Maps server, which then confirm that the app has access to Google Maps data. For how to create the Map API key, refer to Obtain a Google Maps API key .
Titlebar Icons	Full Screen by Default	Whether to display the app in full screen view (with the titlebar hidden) by default when it is opened. In the full screen view, the normal view icon will be available on the top right corner of the window, and when it is tapped, the application will return to the normal view (with the titlebar visible).
App Icon (in PNG format)	App Icon in Google Play	Specify the app icon displayed in Google Play (512 x 512 pixels).
	App Icon for MDPI Screen	Specify the app icon for MDPI (~160 DPI) device screens (48 x 48 pixels).

Group	Value	Description
	App Icon for HDPI Screen	Specify the app icon for HDPI (~240 DPI) device screens (72 x 72 pixels).
	App Icon for XHDPI Screen	Specify the app icon for XHDPI (~480 DPI) device screens (96 x 96 pixels).
	App Icon for XXHDPI Screen	Specify the app icon for XXHDPI (~640 DPI) device screens (144 x 144 pixels).
Splash Image	Splash Image in Landscape	Specify the splash image in landscape for the app. The recommended size is 1024 x 768 pixels, and the tool will stretch or compress the images to accommodate various heights and widths.
	Splash Image in Portrait	Specify the splash image in portrait for the app. The recommended size is 768 x 1024 pixels, and the tool will stretch or compress the images to accommodate various heights and widths.

Step 7: Specify the storage location for the package and click **Next**.

- For iOS, the generated package will be stored under a folder named "*App Name_install_Xcode8*" (for example, Sales Demo_install_Xcode8) under the specified location.
- For Android, the generated package will be stored under a folder named "*App Name_install_Android*" (for example, Sales Demo_install_Android) under the specified location.

Figure 10.47: Specify directory

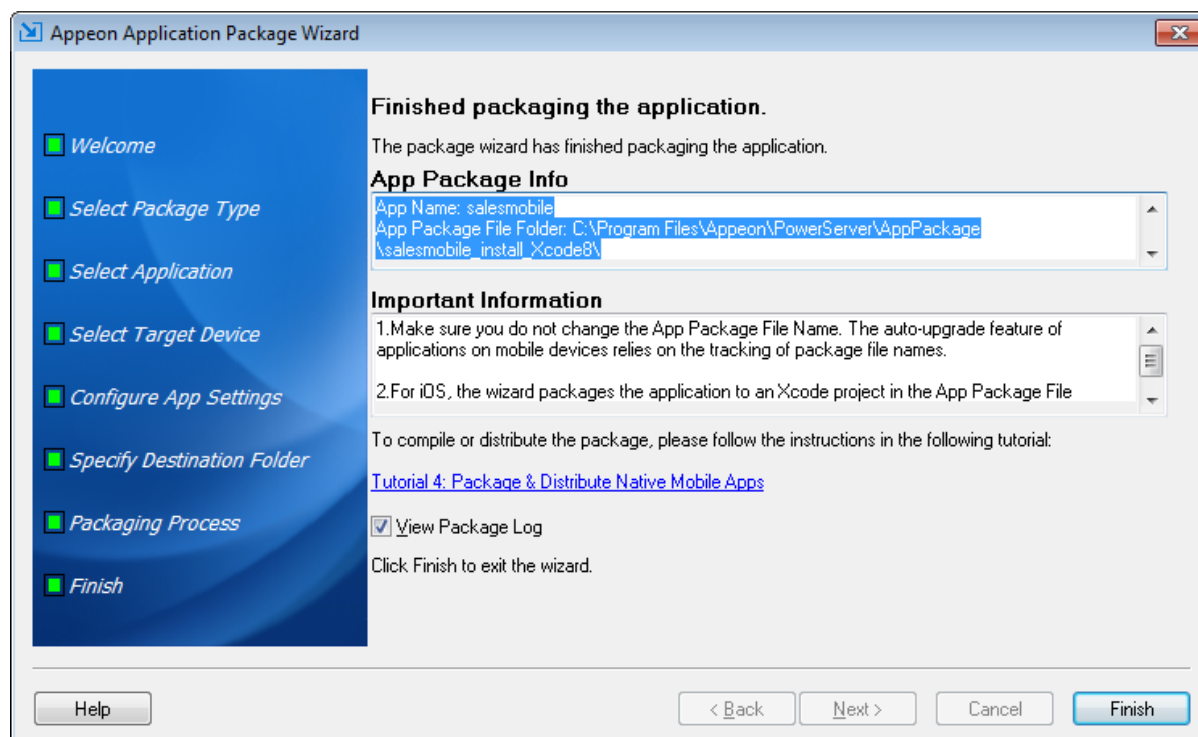
Step 8: Wait until the package process is complete.

Step 9: Click **Finish** when the package process is complete.

If the package log reports the error "Failed to build the native mobile app" when creating the Android APK file, you may try the solution in Section 2.5.1, "Failed to build the native mobile app" error when creating the Android APK package" in *PowerServer Troubleshooting Guide*.

App Package File Name

The **App Package File Name** is automatically generated with a combination of the **App Name**, **App ID Suffix (Bundle ID)** (for iOS) or **App Identifier** (for Android), and the PowerServer Toolkit version and build number, and you should keep this file name unchanged and also when you upgrade the Appeon product and want to create an updated package of the mobile app, be sure to input the same **App Name** and **App ID Suffix (Bundle ID)** (for iOS) or **App Identifier** (for Android) as before so that the mobile app installed on the device can correctly identify the updated package and automatically get upgraded.

Figure 10.48: Package complete

Step 10: Go to the folder "*App Name_install_Xcode8*" for iOS or "*App Name_install_Android*" for Android under the specified location.

- For iOS, you will find the following two zip files:
 - i. *App Name.zip*: the application file package
 - ii. *ApeonMobile.framework.zip*: the mobile client libraries

Then follow the instructions in Section 5.2, “Package & Distribute iOS Apps” in *PowerServer Mobile Tutorials* to compile them into an IPA file and then distribute the IPA file.
- For Android, you will find the following APK file:
 - i. *App Name_install_Android.apk*

Then follow the instructions in Section 5.3, “Package & Distribute Android Apps” in *PowerServer Mobile Tutorials* to distribute the APK file.

10.4 Customizing and packaging Apeon Workspace

Before you start, make sure you have completed the preparation tasks in [Preparing for the mobile package](#).

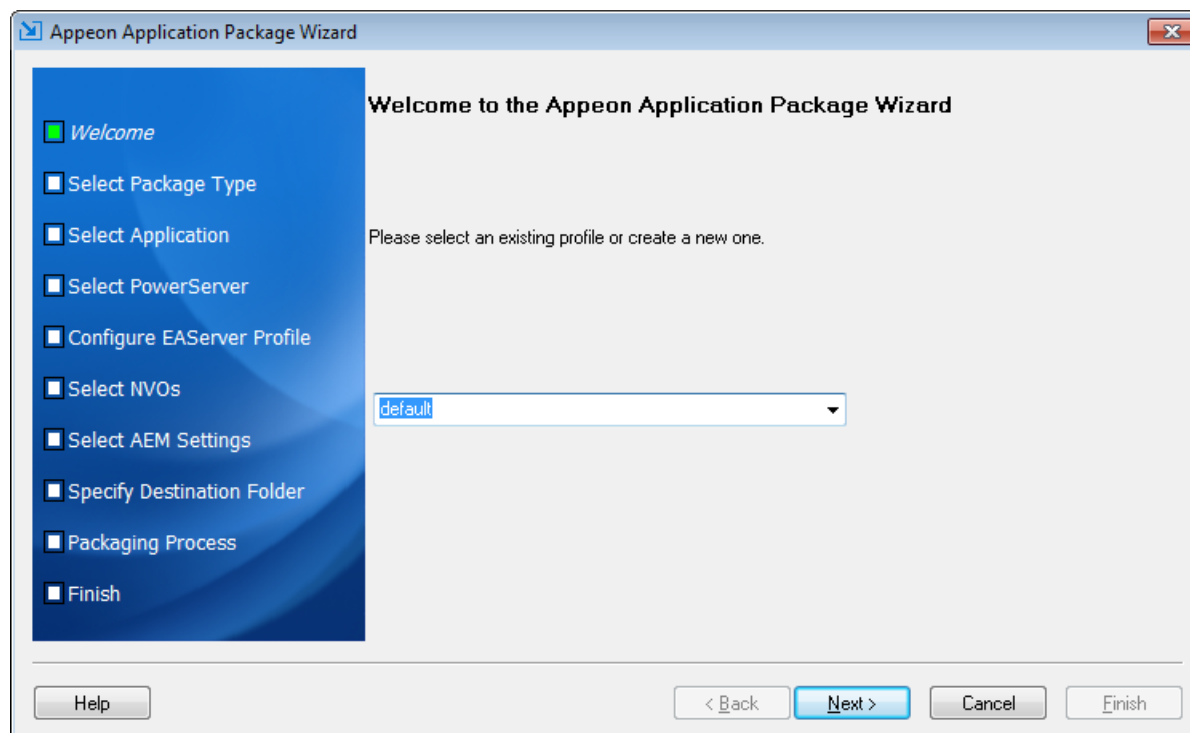
To customize Apeon Workspace and package it as a standalone application, follow the steps below:

Step 1: Click the **Package** (📦) button on the PowerServer Toolkit to open the **Apeon Application Package Wizard**.

Step 2: Select or create a profile from the dropdown list box and click **Next** to proceed.

A profile is a configuration file containing the settings that you specify when packaging the application. You can select an existing profile or create one by entering a name in the text field. The profile will be automatically saved and listed for selection when you launch the Appeon Application Package Wizard again.

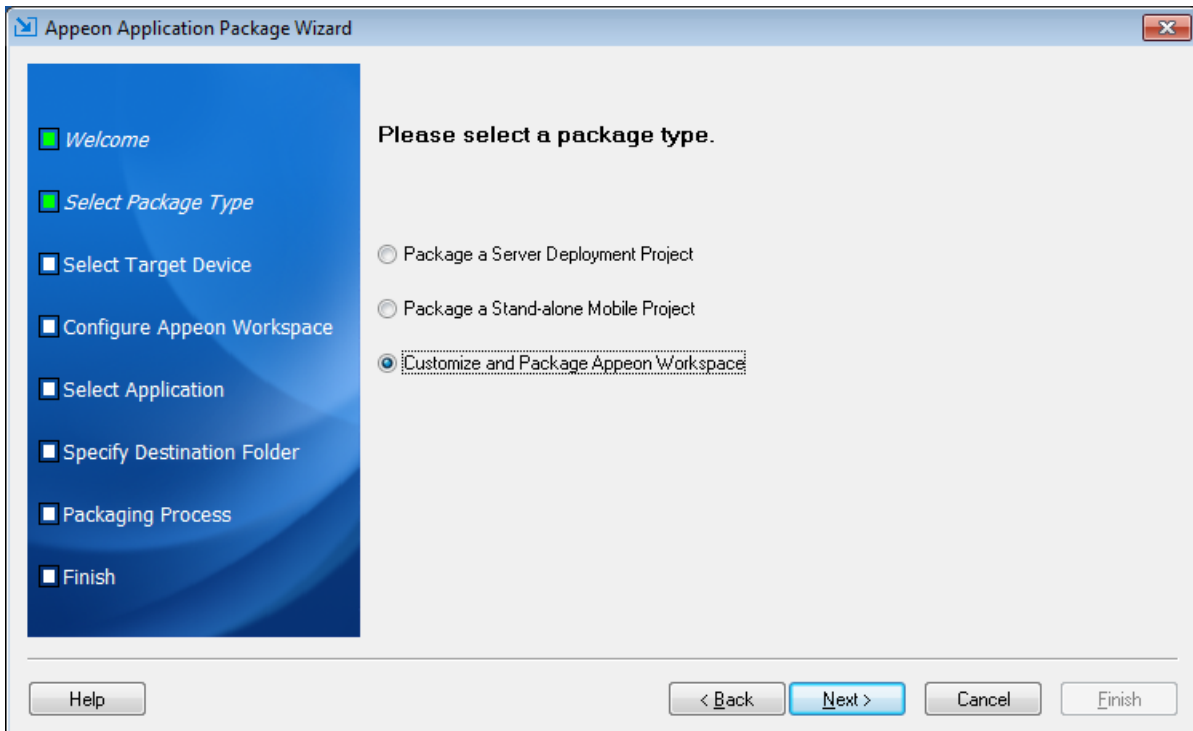
Figure 10.49: Welcome page



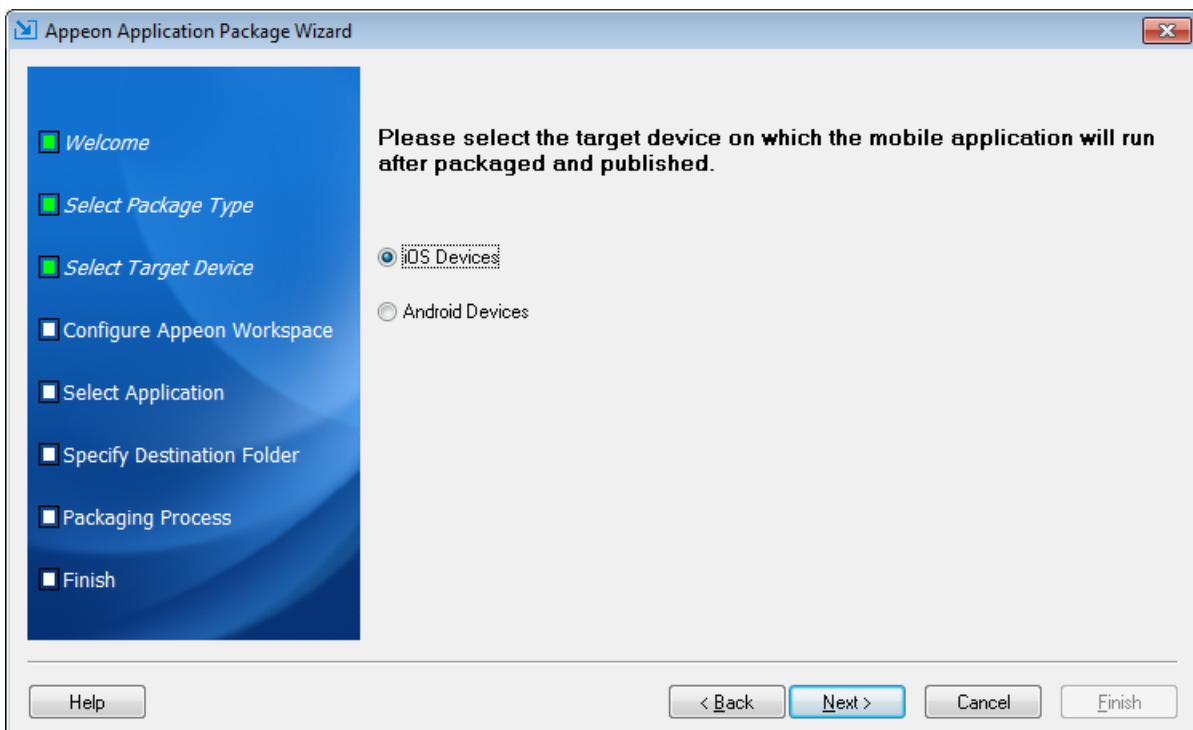
Step 3: Select the **Customize and Package Appeon Workspace** radio button and click **Next**.

To package a deployment project that can deploy the application to servers, select **Package a Server Deployment Project** and follow detailed instructions in [Packaging a server deployment project](#).

To package the mobile application to a stand-alone mobile app, select **Package a Stand-alone Mobile Project** and follow detailed instructions in [Packaging a stand-alone mobile project](#).

Figure 10.50: Select package type

Step 4: Select the device type (iOS or Android) on which the Apeon Workspace application will run after packaged and then click **Next**.

Figure 10.51: Select the target device

Step 5: Specify and customize the parameters according to the device type and then click **Next**.

- For apps running on iOS, follow the instructions in the following table.

Figure 10.52: Workspace parameters for iOS

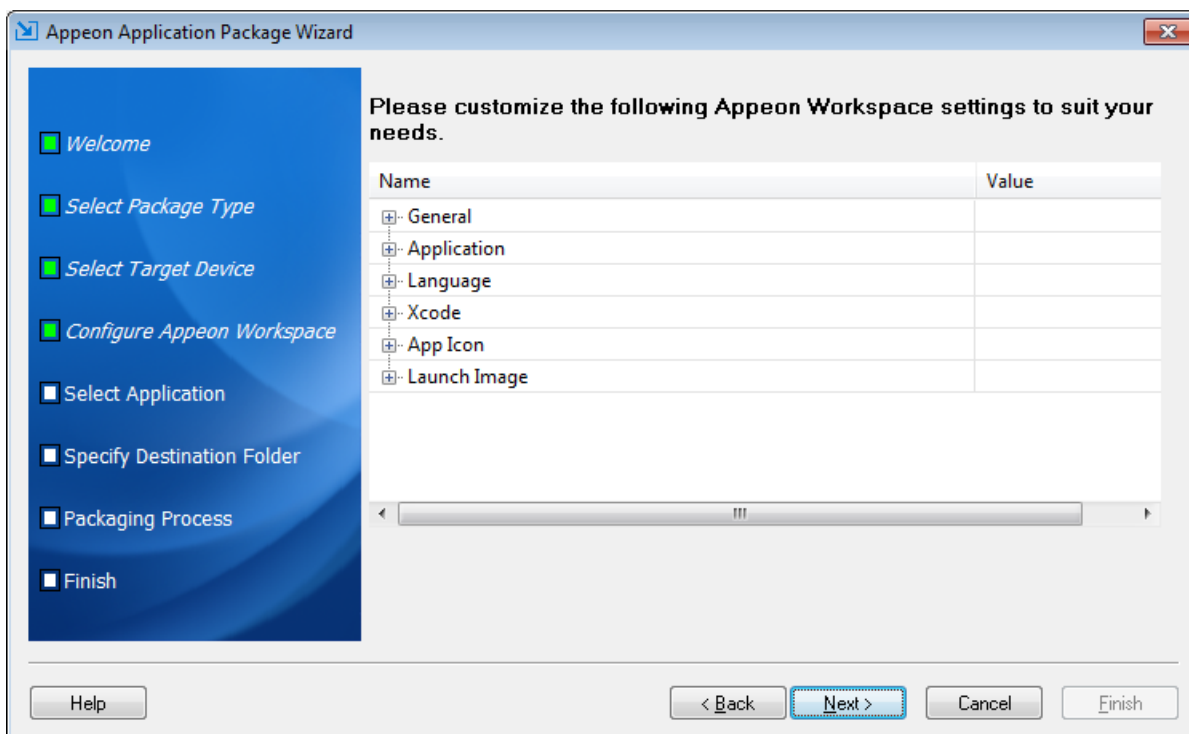


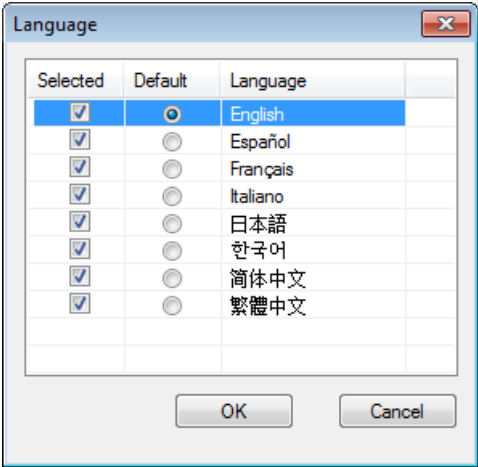
Table 10.5: Workspace parameters for iOS

Group	Parameter	Description
General	App ID Prefix (Required)	<p>Specify the App ID Prefix, and it must be the same App ID Prefix used in the provisioning profile for building the Xcode project later.</p> <p>The App ID Prefix is a 10-character hexadecimal string automatically generated when you generate the App ID in the Apple Developer Member Center. It is unique to you and your developer account.</p> <p>Keep using the same App ID Prefix for different apps, so that these different apps running on the same device can be recognized as running on one device, not on multiple devices by the Apeon license file. For details on the Device control type in the Apeon license file, refer to Section 5.3.4, “Product Activation” in <i>PowerServer Configuration Guide for .NET</i> or in <i>PowerServer Configuration Guide for J2EE</i>.</p>
	App ID Suffix (Bundle ID) (Required)	<p>Specify the App ID Suffix (also called Bundle ID).</p> <p>The App ID Suffix is a name you enter called the Bundle Identifier when you generate the App ID in the Apple Developer Member Center. The Bundle Identifier can be explicit or a wildcard. If you have specified a wildcard bundle identifier when creating the App ID, for example, com.abcxample.*, you will need to replace "*" with</p>

Group	Parameter	Description
		<p>an explicit string, for example, com.abcxample.aws or com.abcxample.1 etc.</p> <p>The App ID Suffix is also used as part of the file name of the generated package. To help your workspace's auto-upgrade feature to work, you should keep this App ID Suffix unchanged every time when creating the package for the same workspace.</p>
	Display Name (Required)	<p>Due to a limitation of the package tool, the display name cannot contain double-byte characters (such as Chinese, Korean, or Japanese characters).</p> <p>The display name will be used in two areas:</p> <ol style="list-style-type: none"> part of the file name of the generated package. <p>For your workspace to have the auto-upgrade feature, you should always use the same display name every time when creating the package for the same workspace.</p> the name of your workspace that will appear in the Apple App Store. <p>For the name to be displayed completely, it should be about 12 letters but it depends on the width of each individual letter, for example, w takes more room than i.</p>
	Workspace Banner	<p>Whether to display the Workspace Banner in your workspace.</p>
	Workspace Banner Package	<p>If you have created your own workspace banner, you can specify to use your own banner here. For how to create your own workspace banner, refer to Prepare your own workspace banner for details.</p> <p>Click the Workspace Banner Package button to select the folder containing your own HTML Web app. Make sure the default page of your own banner is "index.html". The Package tool will check whether the specified folder contains the index.html file.</p>
	SSL Verify Peer	<p>Enable or disable the SSL peer verification.</p>
	SSL Verify Host	<p>Enable or disable the SSL host name Verification.</p>
	Connection Timeout (Seconds)	<p>Specify the timeout seconds for your app connecting to the server.</p>
	Record Logs	<p>Set whether to record app logs.</p>
	Help Path	<p>If you have created your own help file for the workspace, specify the folder that contains your help file. The startup</p>

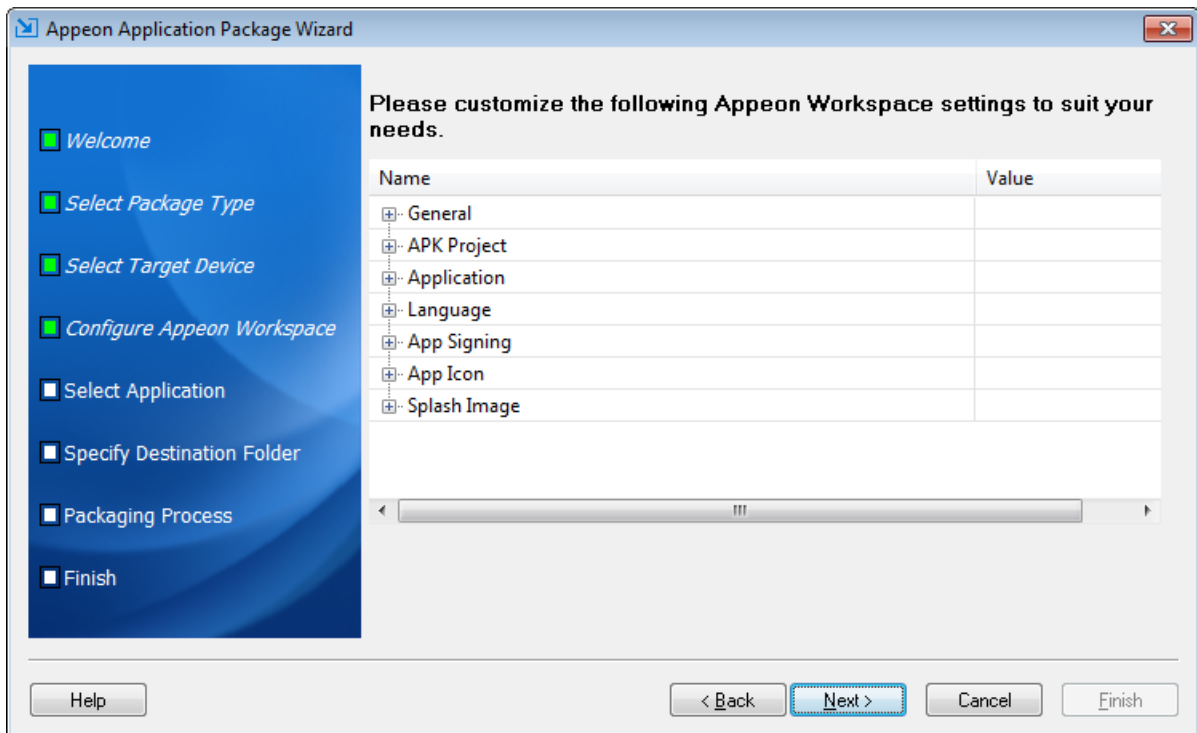
Group	Parameter	Description
		<p>page of your help must be index.html. The Package tool will check whether the specified folder contains the index.html file.</p>
	URL Scheme Name	<p>A URL scheme is a specially formatted URL that allows users to open your app from other Appeon mobile apps or third-party apps. A typical URL scheme looks like this: <i>schemename://?url=http://ipaddress/appname[&parm1=aaa&parm2=bbb...&parmN=zzz]</i></p> <p><i>schemename</i> is what you specify in the URL Scheme Name field as the unique scheme identifier for the application to be packaged. The scheme name is highly recommended to be unique; if you use the same scheme name as another app, iOS or Android will not know which app to launch. Scheme name is case sensitive; it can only contain lower-case letters and numbers, and can only start with lower-case letters.</p> <p>[&parm1=aaa&parm2=bbb...&parmN=zzz] is the parameter name and value pair(s) to be passed to the destination app. It is optional. When it is passed to the Appeon mobile app, it can be obtained by the app using APIs. For more about the APIs, refer to the section called “of_geturlschemeparm” in <i>Workarounds & API Guide</i> and the section called “oe_urlschemesucceed” in <i>Workarounds & API Guide</i>.</p> <p>At the time of writing this note, URL scheme is only supported in Safari and Opera Mini. If you want to use the other browser to open an app via URL scheme, please double check that the browser is supported.</p> <p>Examples</p> <p>The Appeon Workspace published in the online app store (e.g. Apple App Store, Google Play) by Appeon has <i>appeonaws</i> as its scheme name, therefore the URL scheme to open this Appeon Workspace is: <i>appeonaws://</i></p> <p>Suppose you specify <i>ABCaws</i> as the scheme name for your workspace, then the URL scheme to open your workspace will be: <i>ABCaws://</i></p> <p>And the URL scheme to open the app XXX (suppose its app URL is <i>http://demo.appeon.com/xxx</i>) installed in your workspace will be: <i>ABCaws://?url=http://demo.appeon.com/xxx</i></p> <p>Suppose you specify <i>ABCyyy</i> as the scheme name for the standalone mobile app YYY, then the URL scheme to</p>

Group	Parameter	Description
		open YYY will be: ABCyyy://?url=http://192.0.3.111/yyy&name=jacky age=27
	Support Workspace Auto-Upgrade	<p>If the workspace will be uploaded to the online app store such as Apple App Store, Google Play etc., this option should be disabled, because the workspace's auto-upgrade will be controlled by the online app store. If the workspace will be provided on a Web site, we recommend you enable this option as well as the "Check for Workspace Updates" option, so that the workspace can be automatically upgraded.</p> <p>There are other factors affecting the auto-upgrade feature of workspace, refer to Section 3.2.1, "Enabling auto-upgrade of Appeon Workspace" in <i>Appeon Workspace User Guide</i>.</p>
	Check for Workspace Updates	Set whether to automatically check with the server if updates of workspace are available. If updates are detected, the end user will be prompted whether to install the update.
Application	Enable Edit App	Enable or disable the Add App and Delete App functions.
	Default App	Specify the default app. The deployed apps are listed.
	Full Screen by Default	Whether to display the app in full screen view (with the titlebar hidden) by default when it is opened. In the full screen view, the normal view icon will be available on the top right corner of the window, and when it is tapped, the application will return to the normal view (with the titlebar visible).
Language	Default Language	<p>Click the ellipse button to display the Language box.</p> <p>In the Language box, select the Selected check box for the languages you want to package, and select the Default radio button to specify the default language of your workspace UI.</p> <p>The workspace UI can be displayed in English, French, Simplified Chinese, Traditional Chinese, Japanese, Korean, Italian, and Spanish. To display the workspace UI in other languages, you will need to prepare the language package by following instructions in Prepare the UI language package.</p>

Group	Parameter	Description
		<p>Figure 10.53: Language</p> 
Xcode	App Major Version	The version number of your workspace.
	App Build No.	The build number of your workspace.
	Xcode Version	Select the Xcode version which will be used to compile the application. For now, only Xcode 8 is selectable.
App Icon	iPhone Settings (2x)	Specify the settings icon for the retina display on iPhone on iOS 9/10 (58 x 58 pixels).
	iPhone Settings (3x)	Specify the settings icon for the retina HD display on iPhone on iOS 9/10 (87 x 87 pixels).
	iPhone Spotlight (2x)	Specify the spotlight icon for the retina display on iPhone on iOS 9/10 (80 x 80 pixels).
	iPhone Spotlight (3x)	Specify the spotlight icon for the retina HD display on iPhone on iOS 9/10 (120 x 120 pixels).
	iPhone App (2x)	Specify the app icon for the retina display on iPhone on iOS 9/10 (120 x 120 pixels).
	iPhone App (3x)	Specify the app icon for the retina HD display on iPhone on iOS 9/10 (180 x 180 pixels).
	iPad Settings (1x)	Specify the settings icon for the standard display on iPad on iOS 9/10 (29 x 29 pixels).
	iPad Settings (2x)	Specify the settings icon for the retina display on iPad on iOS 9/10 (58 x 58 pixels).
	iPad Spotlight (1x)	Specify the spotlight icon for the standard display on iPad on iOS 9/10 (40 x 40 pixels).
	iPad Spotlight (2x)	Specify the spotlight icon for the retina display on iPad on iOS 9/10 (80 x 80 pixels).
iPad App (1x)	Specify the app icon for the standard display on iPad on iOS 9/10 (76 x 76 pixels).	

Group	Parameter	Description
	iPad App (2x)	Specify the app icon for the retina display on iPad on iOS 9/10 (152 x 152 pixels).
Launch Image	iPhone Portrait (Retina HD 5.5)	Specify the launch image for the retina HD display 5.5-inch in portrait view on iPhone on iOS 9/10 (1242 x 2208 pixels).
	iPhone Portrait (Retina HD 4.7)	Specify the launch image for the retina HD display 4.7-inch in portrait view on iPhone on iOS 9/10 (750 x 1334 pixels).
	iPhone Landscape (Retina HD 5.5)	Specify the launch image for the retina HD display 5.5-inch in landscape view on iPhone on iOS 9/10 (2208 x 1242 pixels).
	iPhone Portrait (2x)	Specify the launch image for the retina display in portrait view on iPhone on iOS 9/10 (640 x 960 pixels).
	iPhone Portrait (Retina 4)	Specify the launch image for the retina display 4-inch in portrait view on iPhone on iOS 9/10 (640 x 1136 pixels).
	iPad Portrait (1x)	Specify the launch image for the standard display in portrait view on iPad on iOS 9/10 (768 x 1024 pixels).
	iPad Portrait (2x)	Specify the launch image for the retina display in portrait view on iPad on iOS 9/10 (1536 x 2048 pixels).
	iPad Landscape (1x)	Specify the launch image for the standard display in landscape view on iPad on iOS 9/10 (1024 x 768 pixels).
	iPad Landscape (2x)	Specify the launch image for the retina display in landscape view on iPad on iOS 9/10 (2048 x 1536 pixels).

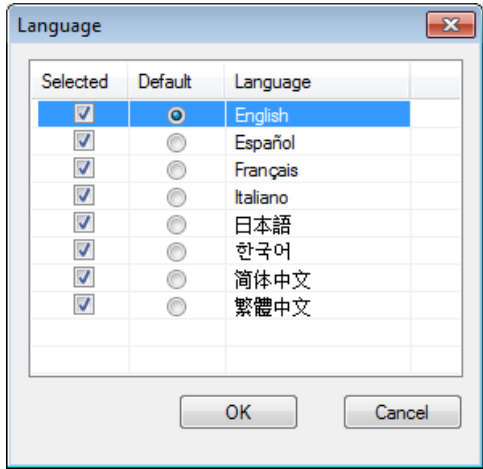
- For apps running on Android, follow instructions in the following table.

Figure 10.54: Workspace parameters for Android**Table 10.6: Workspace settings for Android**

Group	Parameter	Description
General	Display Name (Required)	<p>Due to a limitation of the package tool, the display name cannot contain double-byte characters (such as Chinese, Korean, or Japanese characters).</p> <p>The display name will be used in two areas:</p> <ol style="list-style-type: none"> part of the file name of the generated package. <ul style="list-style-type: none"> For your workspace to have the auto-upgrade feature, you should always use the same display name every time when creating the package for the same workspace. the name of your workspace that will appear in the Android marketplace.
	App Identifier (Required)	<p>Specify the identifier for the app(s).</p> <p>This field is used to identify the app, and when the app is to be installed on a device where another app with the same identifier has already been installed, the installation will fail.</p> <p>The app identifier is also used as part of the file name of the generated package. To help your workspace's auto-upgrade feature to work, you should keep this app identifier unchanged every time when creating the package for the same workspace.</p>

Group	Parameter	Description
		This app identifier must follow the same rules as those for naming packages in the Java programming language. It can only contain lower-case letters, dots, and/or numbers without spaces, and cannot start or end with dots or contain only numbers between dots. It is normally a combination of reversed domain name and the app name (com.companyname.appname), such as, com.appeon.aws, net.abc.appcenter, org.xyz.apploder etc.
	App Version Code	Specify an integer value that uniquely identifies the APK file of your workspace to be uploaded to Google Play. It is used by Google Play for internal purpose and it is invisible to the end user. You can set the value to any integer you want, but each time when you are going to upload an updated APK file for your workspace, make sure you increase it to a greater integer value.
	App Version Name	Specify a string value that represents the release version of your workspace. It will be displayed to the end user.
	Workspace Banner	Whether to display the Workspace Banner in your workspace.
	Workspace Banner Package	<p>If you have created your own workspace banner, you can specify to use your own banner here. For how to create your own workspace banner, refer to Prepare your own workspace banner for details.</p> <p>Click the Workspace Banner Package button to select the folder containing your own HTML Web app. Make sure the default page of your own banner is "index.html". The Package tool will check whether the specified folder contains the index.html file.</p>
	SSL Verify Peer	Enable or disable the SSL peer verification.
	SSL Verify Host	Enable or disable the SSL host name verification.
	Connection Timeout (Seconds)	Specify the timeout seconds for your app connecting to the server.
	Record Logs	Set whether to record app logs.
	Help Path	If you have created your own help file for the workspace, specify the folder that contains your help file. The startup page of your help must be index.html. The Package tool will check whether the specified folder contains the index.html file.
	URL Scheme Name	A URL scheme is a specially formatted URL that allows users to open your app from other Appeon mobile apps or third-party apps. A typical

Group	Parameter	Description
		<p>URL scheme looks like this: <i>schemename</i>://? url=http://ipaddress/appname[&parm1=aaa&parm2=bbb...&parmN=zzz]</p> <p><i>schemename</i> is what you specify in the URL Scheme Name field as the unique scheme identifier for the application to be packaged. The scheme name is highly recommended to be unique; if you use the same scheme name as another app, iOS or Android will not know which app to launch. Scheme name is case sensitive; it can only contain lower-case letters and numbers, and can only start with lower-case letters.</p> <p>[&parm1=aaa&parm2=bbb...&parmN=zzz] is the parameter name and value pair(s) to be passed to the destination app. It is optional. When it is passed to the Appeon mobile app, it can be obtained by the app using APIs. For more about the APIs, refer to the section called “of_geturlscemeparm” in <i>Workarounds & API Guide</i> and the section called “oe_urlschemesucceed” in <i>Workarounds & API Guide</i>.</p> <p>At the time of writing this note, URL scheme is only supported in Safari and Opera Mini. If you want to use the other browser to open an app via URL scheme, please double check that the browser is supported.</p> <p>Examples</p> <p>The Appeon Workspace published in the online app store (e.g. Apple App Store, Google Play) by Appeon has <i>appeonaws</i> as its scheme name, therefore the URL scheme to open this Appeon Workspace is: <i>appeonaws</i>://</p> <p>Suppose you specify <i>ABCaws</i> as the scheme name for your workspace, then the URL scheme to open your workspace will be: <i>ABCaws</i>://</p> <p>And the URL scheme to open the app XXX (suppose its app URL is http://demo.appeon.com/xxx) installed in your workspace will be: <i>ABCaws</i>://?url=http://demo.appeon.com/xxx</p> <p>Suppose you specify <i>ABCyyy</i> as the scheme name for the standalone mobile app YYY, then the URL scheme to open YYY will be: <i>ABCyyy</i>://?url=http://192.0.3.111/yyy&name=jacky age=27</p>
	Support Workspace Auto-Upgrade	<p>If the workspace will be uploaded to the online app store such as Apple App Store, Google Play etc., this option should be disabled, because the workspace's auto-upgrade will be controlled by the online app store. If the workspace will be provided on your own Web site, we recommend you enable this option as well as the "Check for Workspace Updates" option, so that the workspace can be automatically upgraded.</p>

Group	Parameter	Description
		There are other factors affecting the auto-upgrade feature of workspace, refer to Section 3.2.1, “Enabling auto-upgrade of Apeon Workspace” in <i>Apeon Workspace User Guide</i> .
	Check for Workspace Updates	Set whether to automatically check with the server if updates of workspace are available. If updates are detected, the end user will be prompted whether to install the update.
APK Project	Delete project after packaged	Whether to delete the APK project after the app is packaged. If not, there will be a folder named after the Display Name in ...\PowerServerToolkit\AppTemplate\Android\NativeConfig\.
Application	Enable Edit App	Enable or disable the Add App and Delete App functions.
	Default App	Specify the default app. The deployed apps are listed.
	Full Screen by Default	Whether to display the app in full screen view (with the titlebar hidden) by default when it is opened. In the full screen view, the normal view icon will be available on the top right corner of the window, and when it is tapped, the application will return to the normal view (with the titlebar visible).
Language	Default Language	<p>Click the ellipse button to display the Language box.</p> <p>In the Language box, select the Selected check box for the languages you want to package, and select the Default radio button to specify the default language of your workspace UI.</p> <p>The workspace UI can be displayed in English, French, Simplified Chinese, Traditional Chinese, Japanese, Korean, Italian, and Spanish. To display the workspace UI in other languages, you will need to prepare the language package by following instructions in Prepare the UI language package.</p> <p>Figure 10.55: Language</p> 
App Signing	Alias (Required)	Enter the alias name for the key. Only the first 8 characters of the alias name are used.

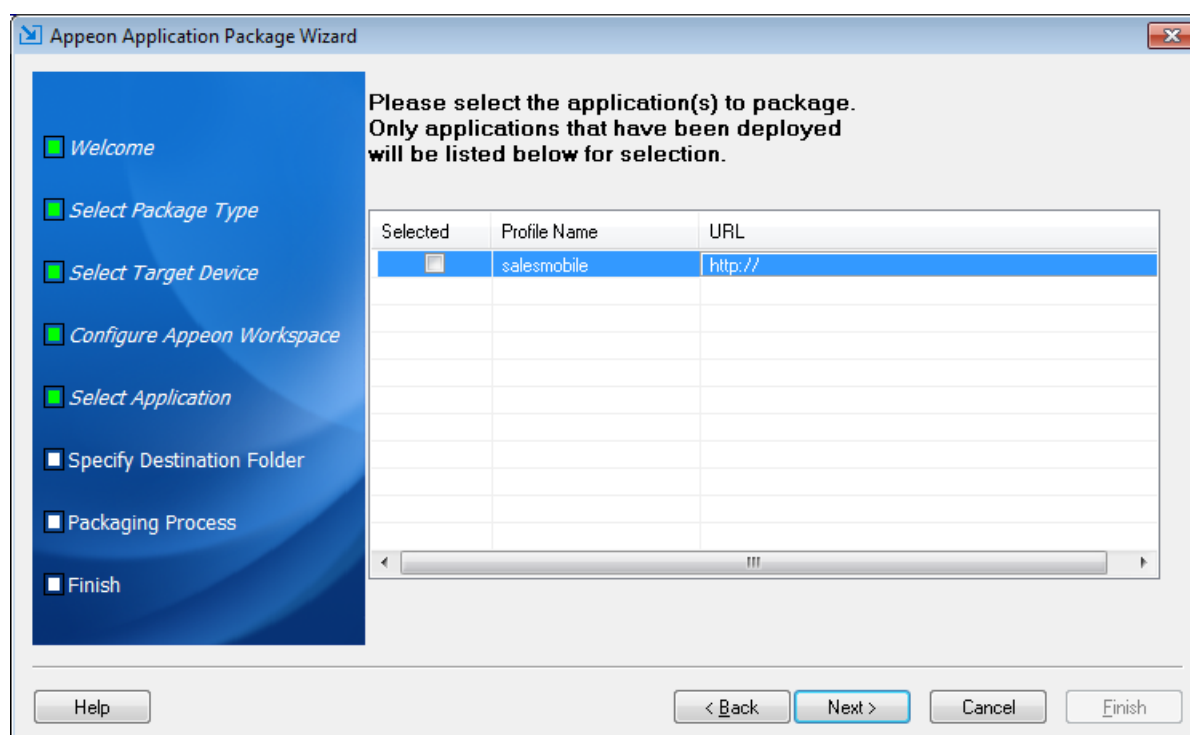
Group	Parameter	Description
		You can use the default alias name generated by Appeon (which is <i>appeon</i>) or use your own name. If you use the default keystore file in the Directory field, you need to keep this field as default.
	Alias Password	Enter the password. You can use the default alias password generated by Appeon (which is <i>appeon</i>) or use your own password. If you use the default keystore file in the Directory field, you need to keep this field as default.
	Directory (Required)	Click the browse button (...) to select the keystore file. You use the default keystore file or click the browse button to select your own one. For how to create the keystore file, refer to Obtain a private key .
	Keystore Password	Enter the keystore password. You can use the default keystore password generated by Appeon (which is <i>appeon</i>) or use your own password. If you use the default keystore file in the Directory field, you need to keep this field as default.
	Map API Key	Specify the Google Map API key. The Map API reads the key value and passes it to the Google Maps server, which then confirms that the app has access to Google Maps data. For how to create the Map API key, refer to Obtain a Google Maps API key .
App Icons (in PNG format)	App Icon in Google Play	Specify the app icon displayed in Google Play (512 x 512 pixels).
	App Icon for MDPI Screen	Specify the app icon for MDPI (~160 DPI) device screens (48 x 48 pixels).
	App Icon for HDPI Screen	Specify the app icon for HDPI (~240 DPI) device screens (72 x 72 pixels).
	App Icon for XHDPI Screen	Specify the app icon for XHDPI (~480 DPI) device screens (96 x 96 pixels).
	App Icon for XXHDPI Screen	Specify the app icon for XXHDPI (~640 DPI) device screens (144 x 144 pixels).
Splash Image	Splash Image in Landscape	Specify the splash image in landscape for the app. The recommended size is 1024 x 768 pixels, and the tool will stretch or compress the images to accommodate various heights and widths.
	Splash Image in Portrait	Specify the splash image in portrait for the app.

Group	Parameter	Description
		The recommended size is 768 x 1024 pixels, and the tool will stretch or compress the images to accommodate various heights and widths..

Step 6 (Optional): Select the apps you want to package into the workspace and specify the URL to the apps. The apps deployed in PowerServer Toolkit are listed by default, and multiple apps can be selected.

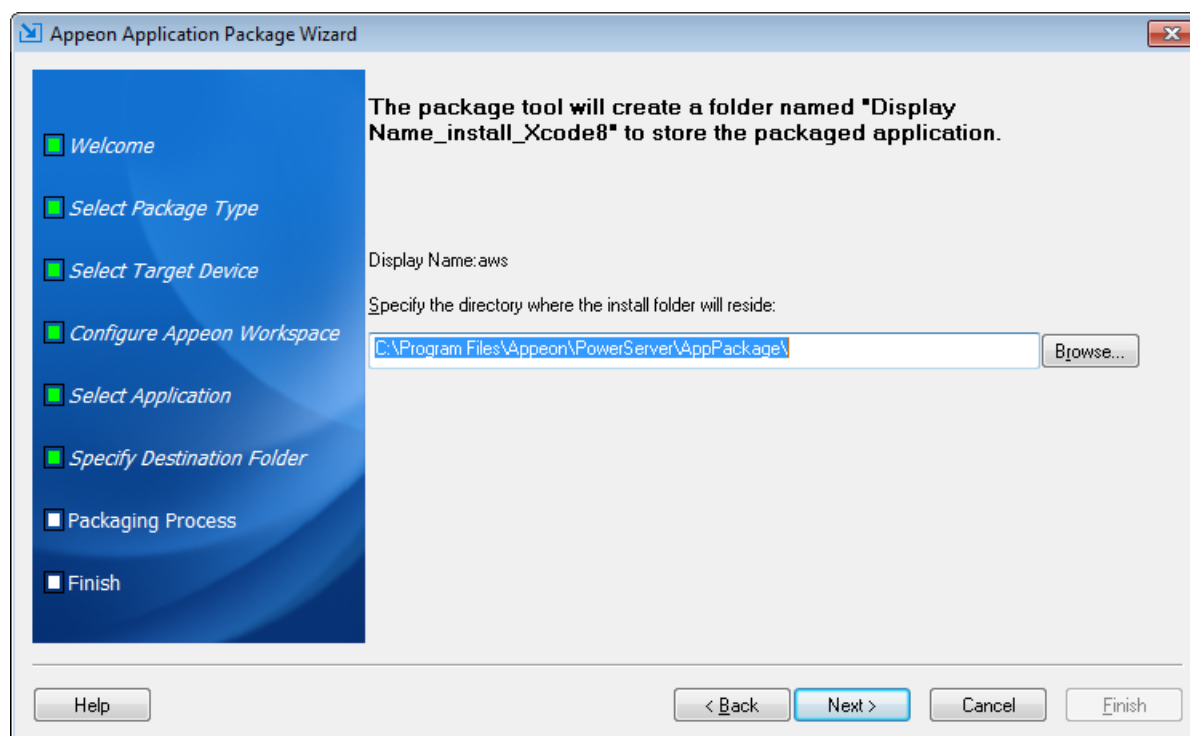
For more information about how to specify the URL, see [URLs of Appeon applications](#).

Figure 10.56: Select apps to be packaged with your Workspace



Step 7: Specify the storage location for the generated package and click **Next**.

- For iOS, the generated package will be stored under a folder named "*Display Name_install_Xcode8*" (for example, *MyWorkspace_install_Xcode8*) under the specified location.
- For Android, the generated package will be stored under a folder named "*Display Name_install_Android*" (for example, *MyWorkspace_install_Android*) under the specified location.

Figure 10.57: Specify the directory

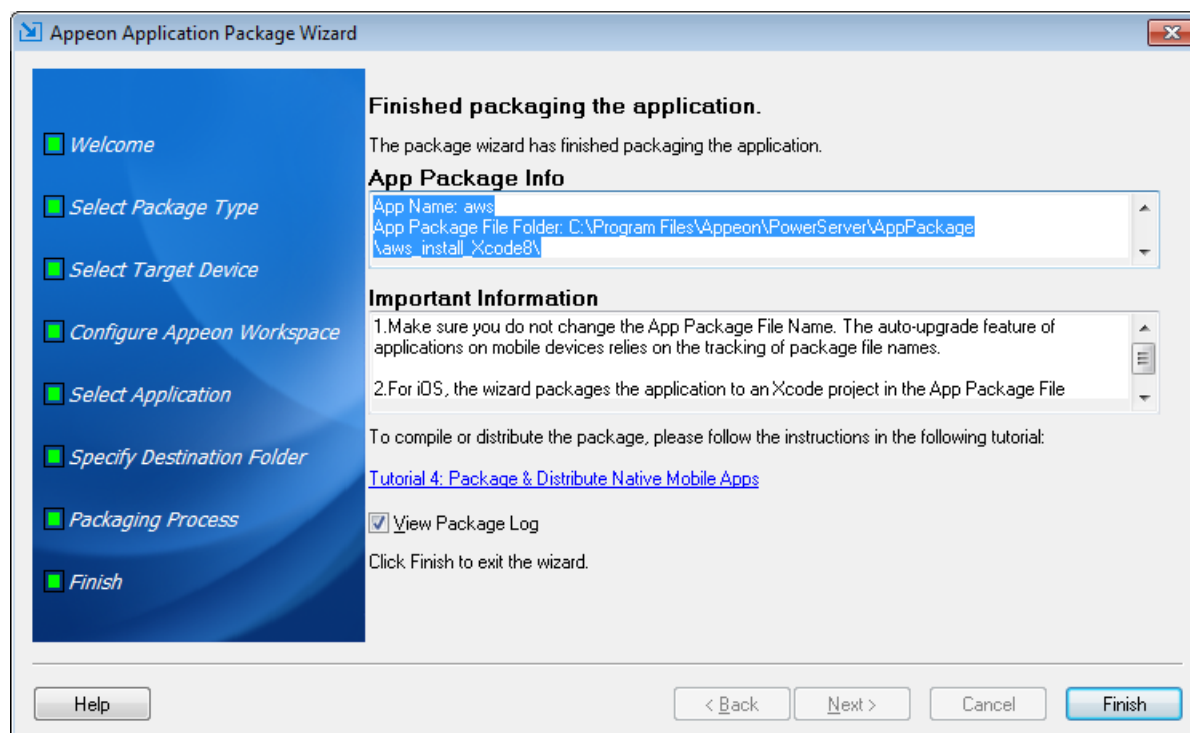
Step 8: Wait until the package process is complete.

Step 9: Click **Finish** when the package process is complete.

If the package log reports the error "Failed to build the native mobile app" when creating the Android APK file, you may try the solution in Section 2.5.1, "Failed to build the native mobile app" error when creating the Android APK package" in *PowerServer Troubleshooting Guide*.

App Package File Name

The **App Package File Name** is automatically generated with a combination of the **Display Name**, **App ID Suffix (Bundle ID)** (for iOS) or **App Identifier** (for Android), and the PowerServer Toolkit version and build number, and you should keep this file name unchanged and also when you upgrade the Appeon product and want to create an updated package of Appeon Workspace, be sure to input the same **Display Name** and **App ID Suffix (Bundle ID)** (for iOS) or **App Identifier** (for Android) as before so that the Appeon Workspace installed on the mobile device can correctly identify the updated package and automatically get upgraded.

Figure 10.58: Package complete

Step 10: Go to the folder "*Display Name_install_Xcode8*" for iOS or "*Display Name_install_Android*" for Android under the specified location.

- For iOS, you will find the following two zip files:
 - i. *Display Name.zip*: the zip package of the workspace
 - ii. *ApeonMobile.framework.zip*: the mobile client library

Then follow the instructions in Section 5.2, “Package & Distribute iOS Apps” in *PowerServer Mobile Tutorials* to compile them into an IPA file and then distribute the IPA file.
- For Android, you will find the following APK file:
 - i. *Display Name_install_Android.apk*: the package of the workspace

Then follow the instructions in Section 5.3, “Package & Distribute Android Apps” in *PowerServer Mobile Tutorials* to distribute the APK file.

11 Undeploying Apeon Applications

The Application Undeployment Wizard removes a deployed application from the associated PowerServer(s) and Web Server(s), including all DataWindows from the PowerServer(s), all the Web or mobile application files from the Web Server(s), and all the Transaction Object mappings from AEM. This is the only way that an Apeon deployed application should be removed. Any other method may not fully remove the application, and may cause errors.

11.1 Undeploying instructions

Step 1: Verify that the PowerServer and the Web Server(s) hosting the Apeon application to be undeployed are running before you proceed with undeployment.


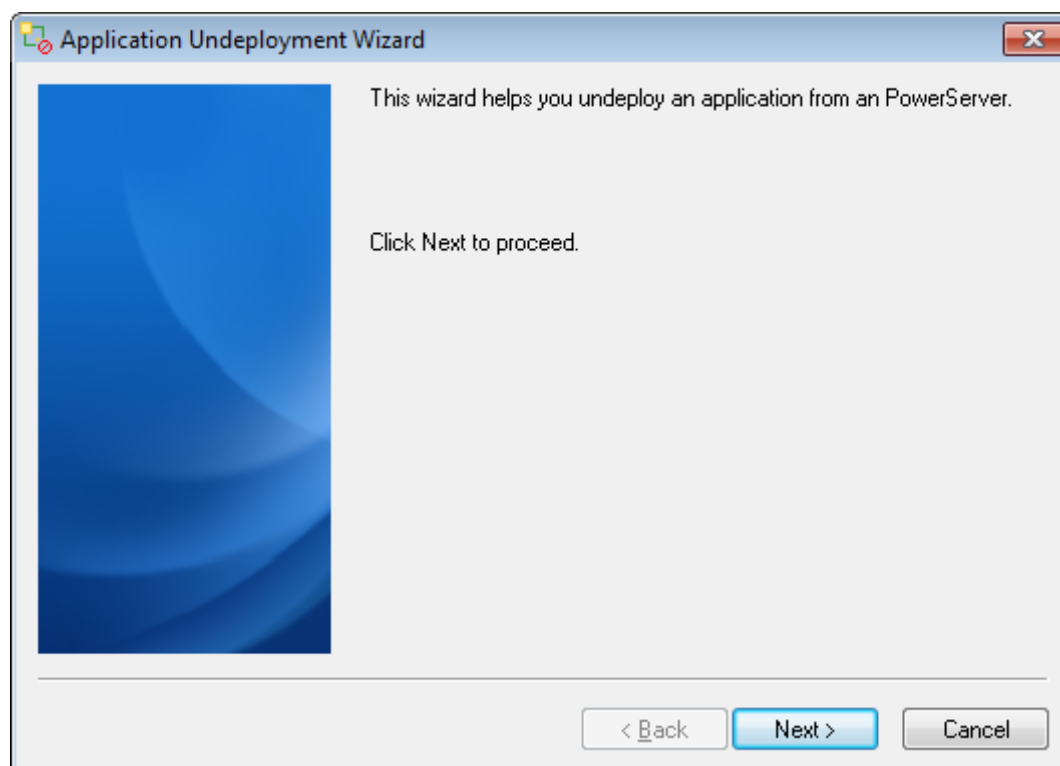
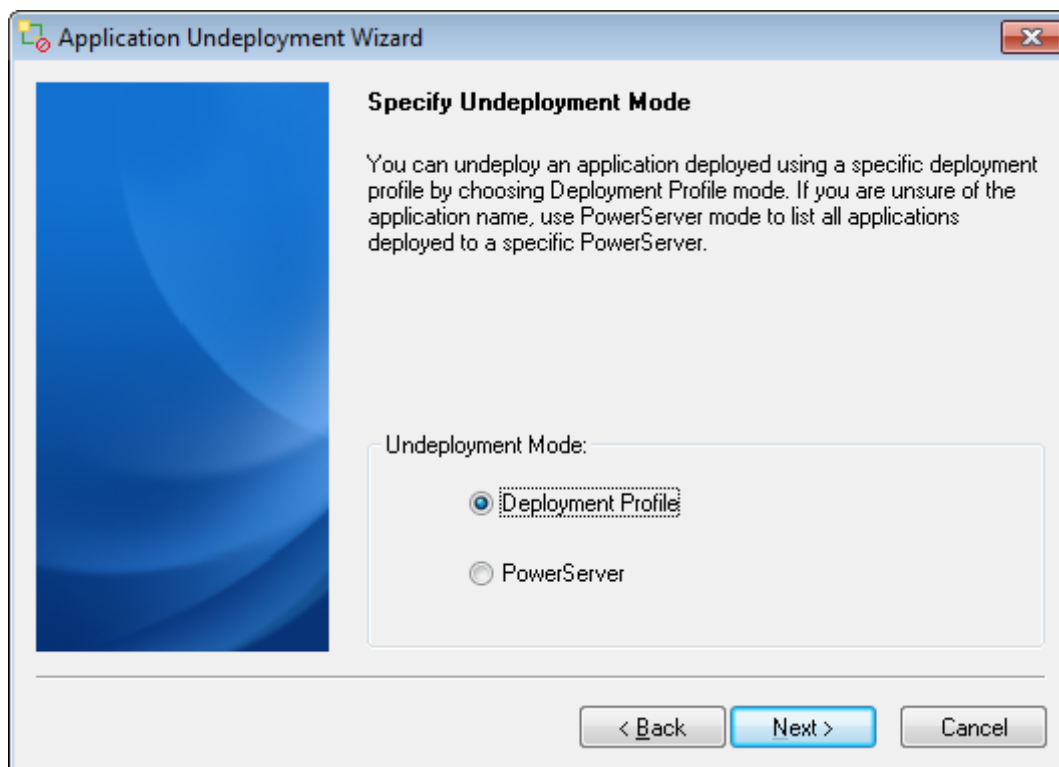
Step 2: Click the **Undeploy** button () in the PowerServer Toolkit. The Application Undeployment Wizard is displayed, as shown in the following figure.

Figure 11.1: Application Undeployment Wizard



Step 3: Click **Next** to continue. The **Specify Undeployment Mode** window is displayed, as shown in the following figure, prompting you to select an undeployment mode.

Figure 11.2: Specify Undeployment Mode window



The following table shows when and how to use each mode.

Table 11.1: Undeployment modes

	When To Use It	How To Use It
Deployment Profile mode	If you clearly know which application you want to delete from which server.	Select an application to be undeployed and its associated deployment profile. For detailed instructions, refer to Section 11.1.1, “Undeploying with the Deployment Profile mode” .
PowerServer mode	When you are uncertain of the name of the application you want to undeploy, or an application profile does not exist for the application you want to undeploy, but you clearly know the PowerServer hosting the Web or mobile application intended for undeployment.	Specify a PowerServer by selecting an PowerServer profile. The selected PowerServer will refer to all the Web or mobile applications that are deployed. Then you can choose one Web or mobile application for undeployment. For detailed instructions, refer to Section 11.1.2, “Undeploying with the PowerServer mode” .

11.1.1 Undeploying with the Deployment Profile mode

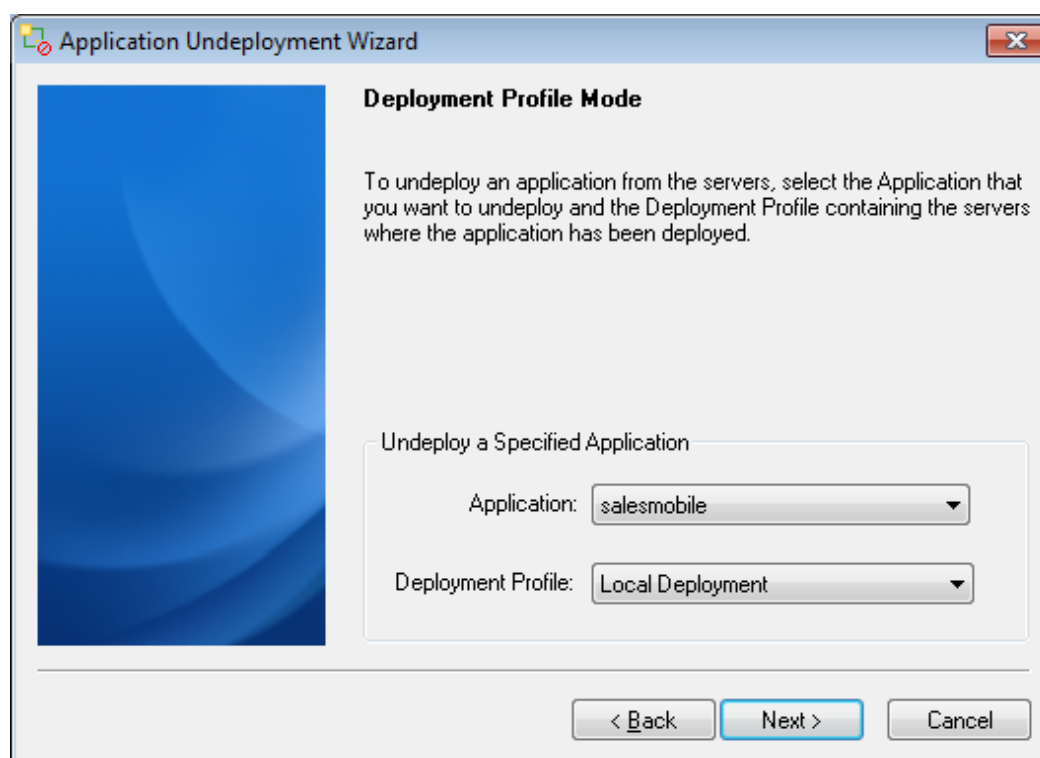
Step 1: Select the **Deployment Profile** radio button in the **Specify Undeployment Mode** window, and click **Next** to continue.

Step 2: Select a Web or mobile application from the Application list box, as shown in the following figure.

The Application list box lists all the application profile names. Be sure to choose one that has been deployed and is intended for undeployment. The application and the deployment profile used in the last deployment will be selected by default.

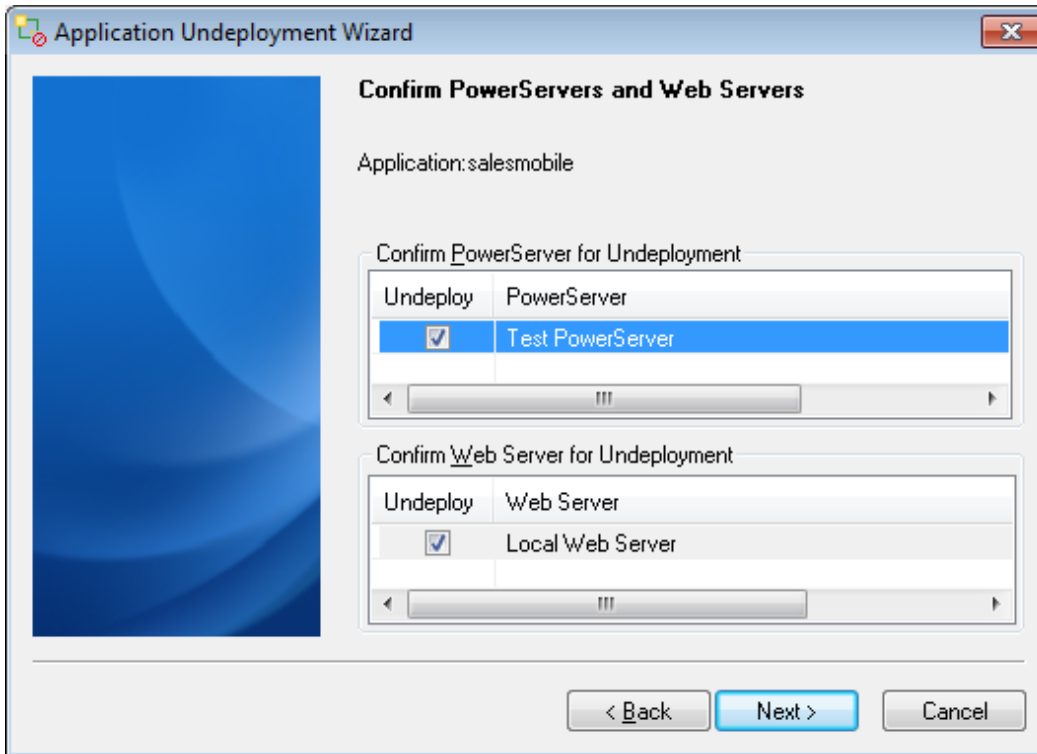
Step 3: Select the deployment profile that is used to deploy the Web or mobile application from the Deployment Profile list box, as shown in the following figure. Click **Next** to continue.

Figure 11.3: Deployment Profile Mode window



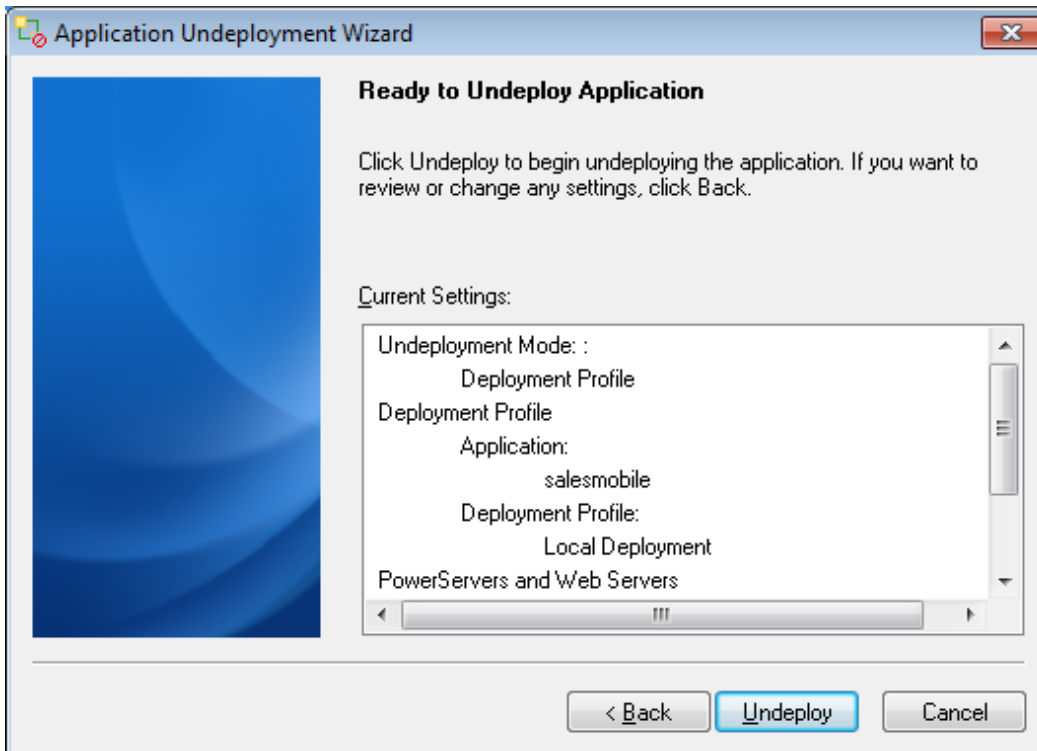
Step 4: Select the PowerServer(s) or Web Server(s) that are intended for undeployment, as shown in the following figure. The PowerServer(s) and Web Server(s) defined in the deployment profile selected in the previous step are listed. Click **Next** to continue.

Figure 11.4: Confirm PowerServer and Web Servers



Step 5: Click **Undeploy** to confirm the undeployment settings and start the undeployment process, as shown in the following figure.

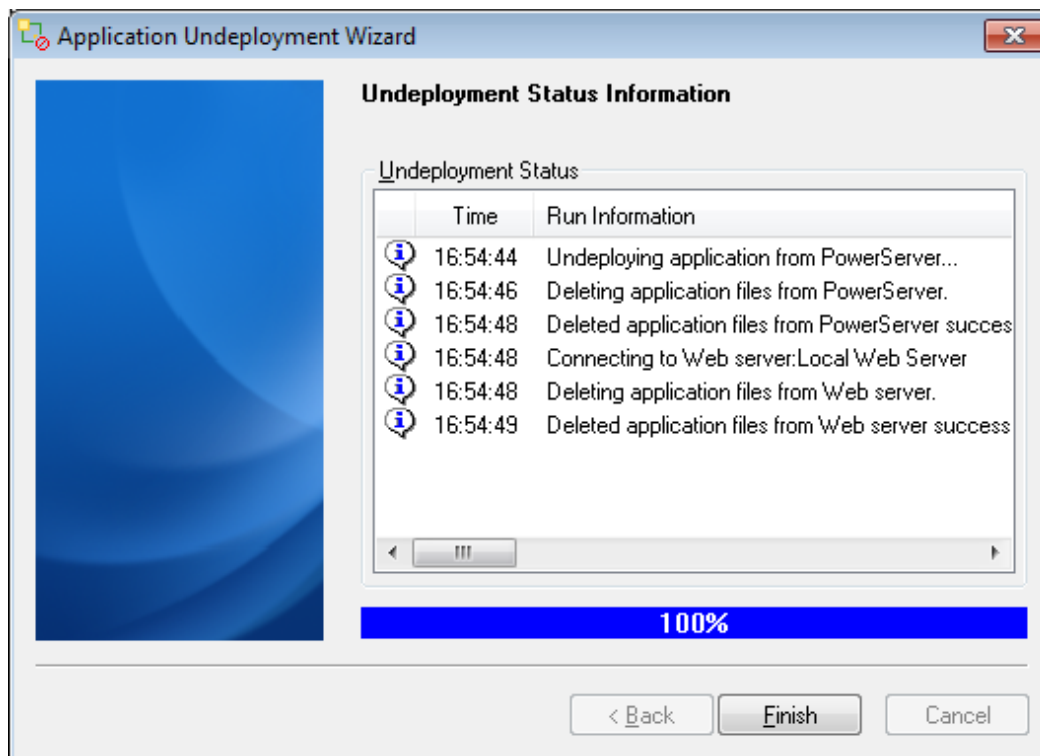
Figure 11.5: Confirm undeployment settings



The undeployment process begins, as shown in the following figure.

Step 6: Click **Finish** to close the dialog box.

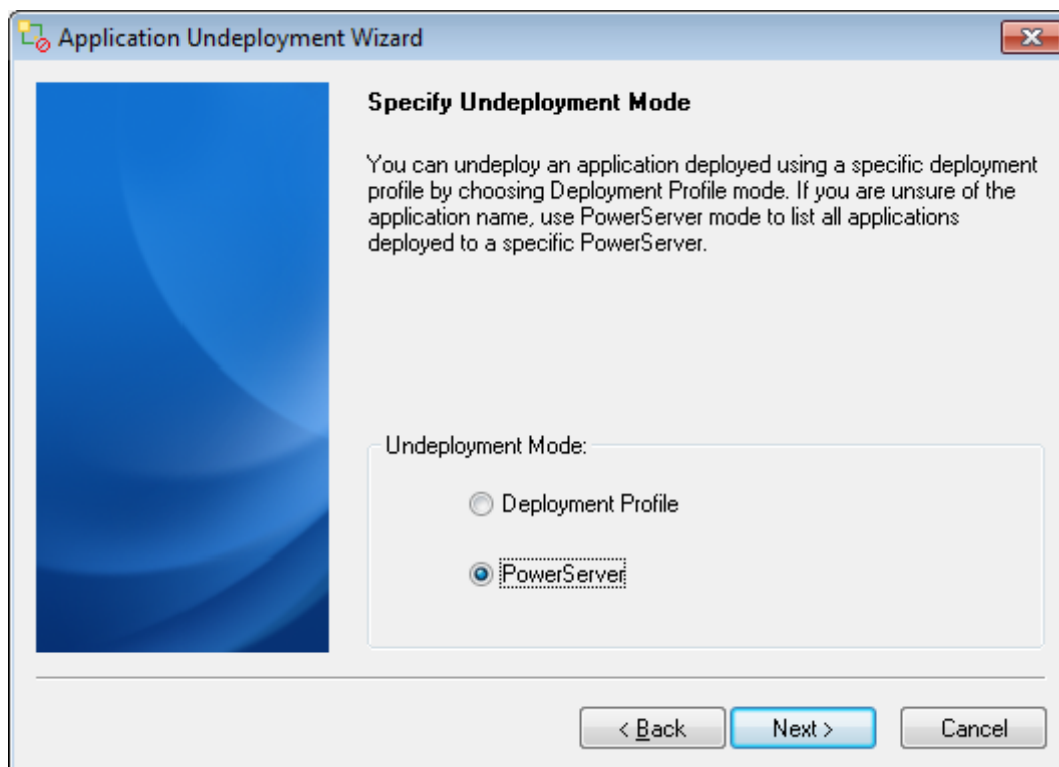
Figure 11.6: Undeployment in Progress



11.1.2 Undeploying with the PowerServer mode

Step 1: Select the **PowerServer** radio button in the **Specify Undeployment Mode** window as shown in the following figure, and click **Next** to continue.

Figure 11.7: Specify Undeployment Mode window



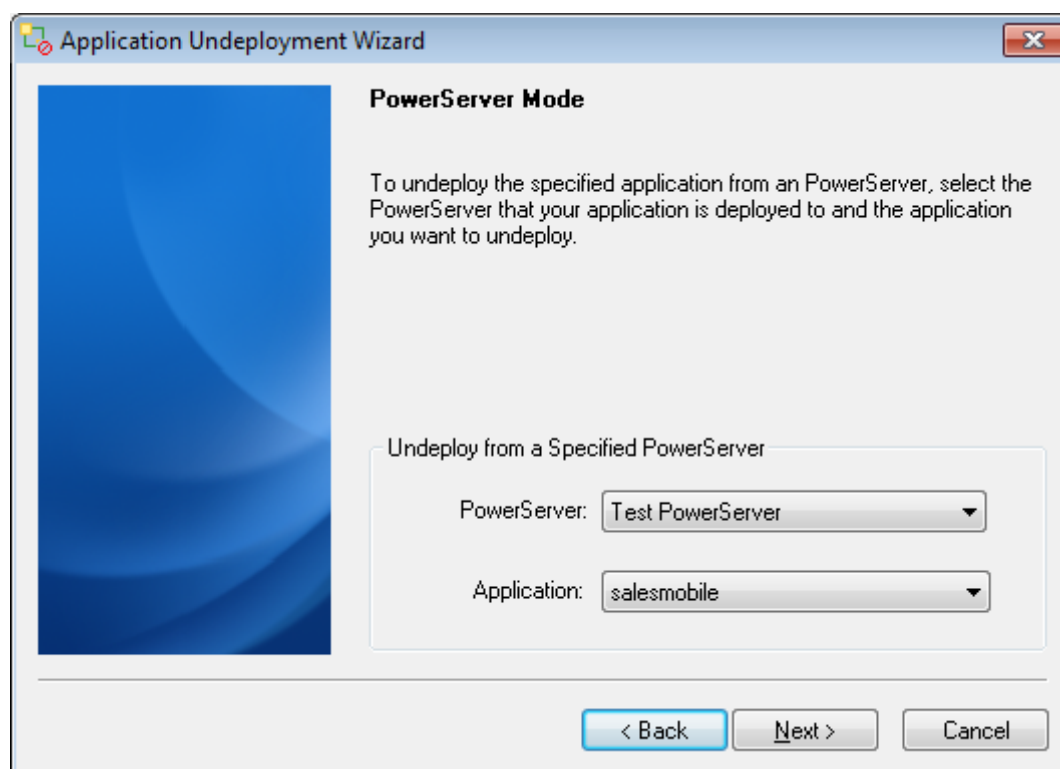
Step 2: Select the PowerServer that contains the Web or mobile application to be undeployed from the PowerServer list box, as shown in the following figure.

The PowerServer list box lists all the PowerServer profile names. Make sure the selected PowerServer is running.

Step 3: Select the application to be undeployed from the Application list box and click **Next**, as shown in the following figure.

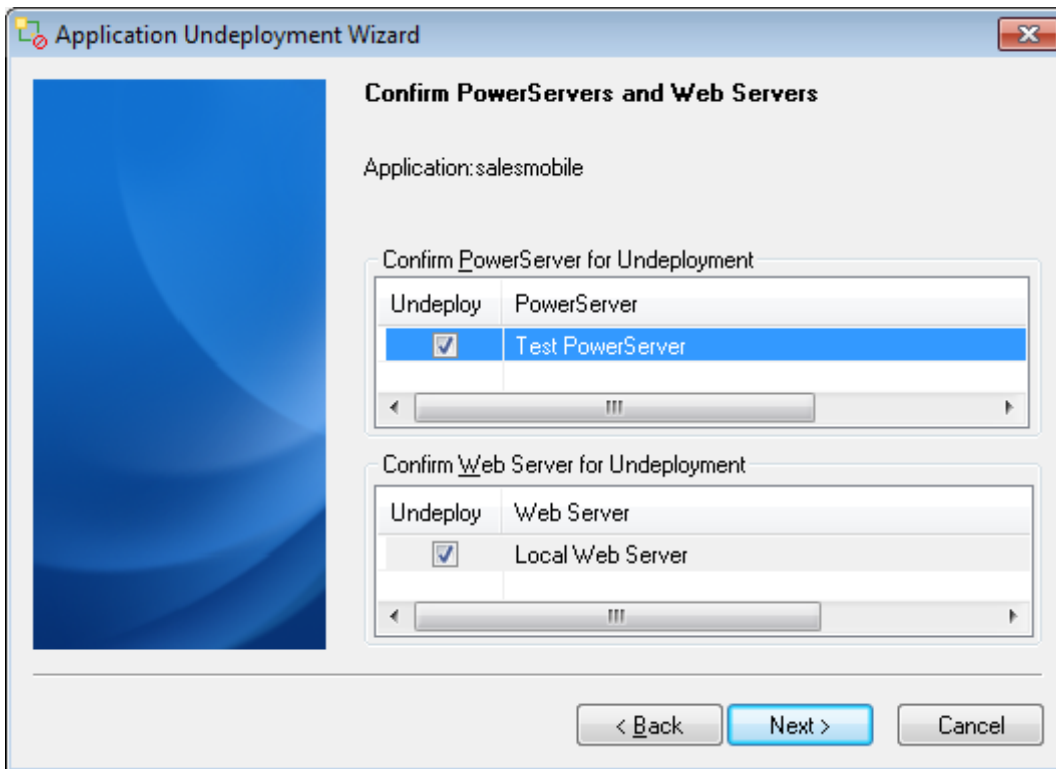
The Application list box lists all the Web or mobile applications deployed to the selected PowerServer.

Figure 11.8: PowerServer Mode window



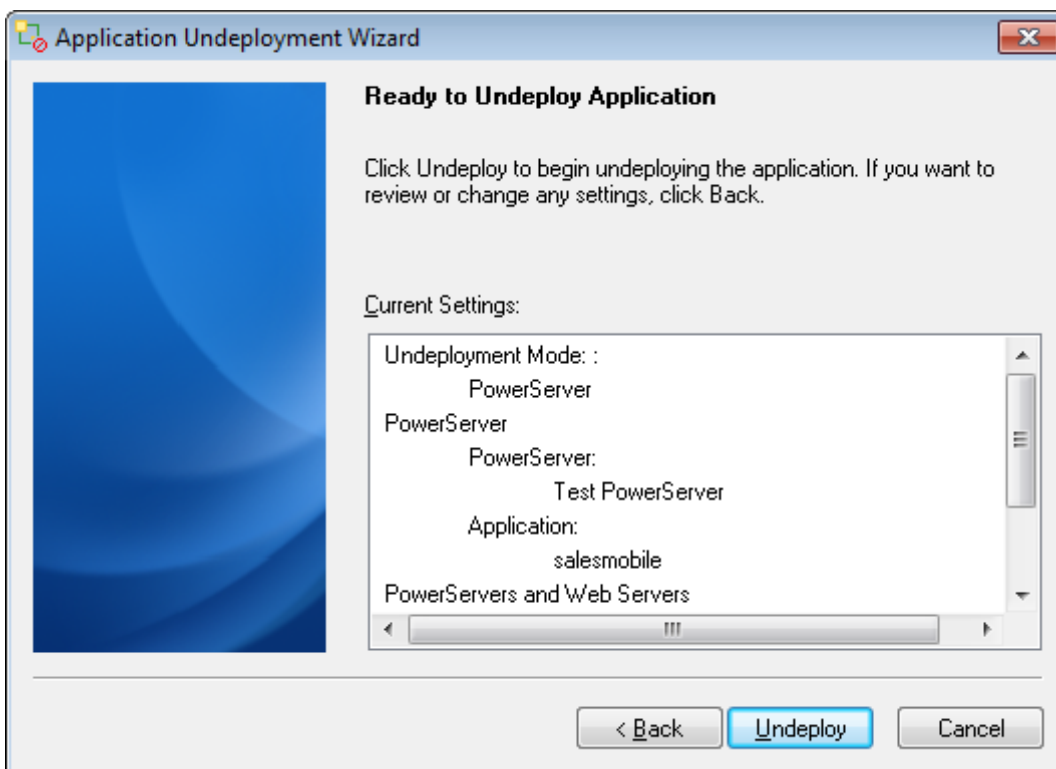
Step 4: Select the PowerServer(s) or Web Server(s) that are intended for undeployment, as shown in the following figure. The PowerServer(s) and Web Server(s) defined in the default deployment profile are listed. Click **Next** to continue.

Figure 11.9: Ready to Undeploy Application window



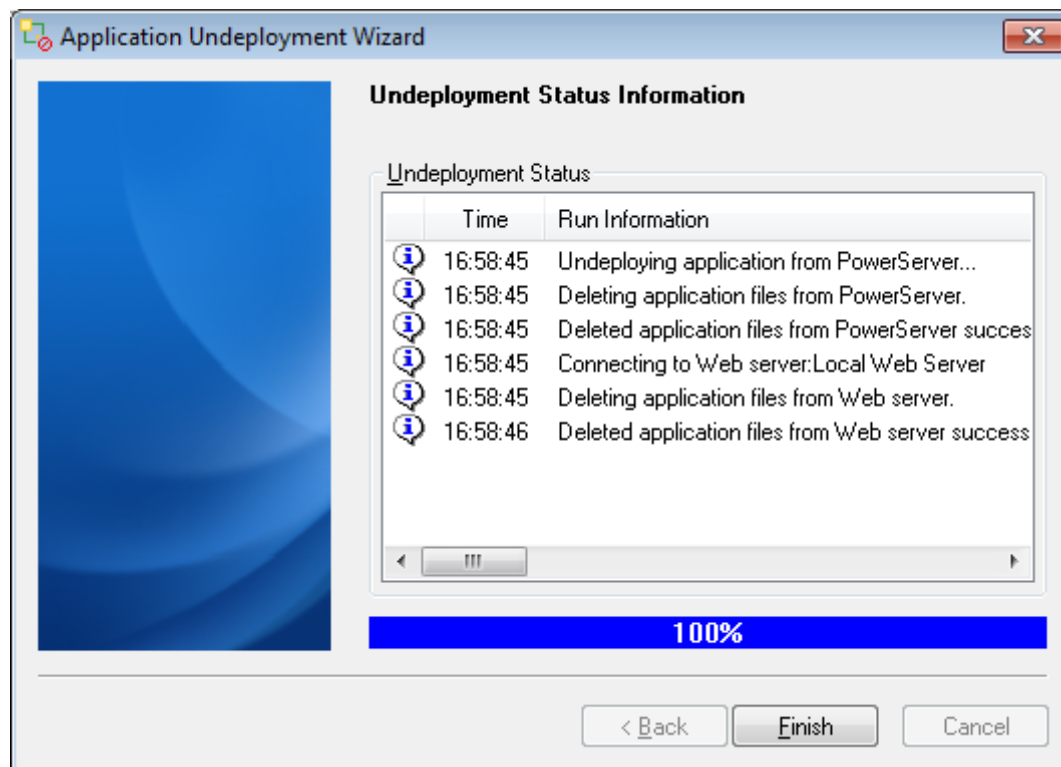
Step 5: Click **Undeploy** to confirm the undeployment settings and start the undeployment process, as shown in the following figure.

Figure 11.10: Undeployment settings



The undeployment process begins, as shown in the following figure.

Figure 11.11: Apeon Application Undeployment



Step 6: Click **Finish** to close the dialog box, as shown in the above figure.

12 Developing with Code Insight

Code Insight is a tool designed to help you write Appeon-supported PowerScript code more quickly by providing a lookup and paste service inside the PowerBuilder Script view. You can use it to develop PowerBuilder applications that are free of Appeon unsupported features and tailored to Web or mobile migration.

The usage of Code Insight is similar to that of PowerBuilder AutoScript. The following table describes the differences between the PowerBuilder AutoScript and Appeon Code Insight.

Table 12.1: Comparison between AutoScript and Code Insight



	PowerBuilder AutoScript	Appeon Code Insight
Effective for	Functions, events, variables, properties, and templates for PowerBuilder DO, FOR, IF, and CHOOSE statements.	Items referred in dot notations
What pops up	A list of properties, variables, methods, or statements.	1) A list of properties, variables, or methods. 2) The Appeon unsupported properties, variables or methods that are marked with a red icon left to them.
Ways to use it	Two ways to use AutoScript: <ul style="list-style-type: none"> • Turn automatic popup on to pop up a list automatically when you pause while typing, or • Invoke AutoScript only once when you need it by selecting the menu items. 	Only one way to use Code Insight: <p>Step 1: Activate Code Insight;</p> <p>Step 2: Configure the default PBT file;</p> <p>Step 3: Enable Code Insight.</p> <p>Refer to the following sections for detailed instructions.</p>
Priority	When Code Insight is activated, AutoScript will be automatically turned off.	




12.1 Activating Code Insight

Click the **Code Insight** button () in the PowerServer Toolkit to launch Code Insight.

At the click of the **Code Insight** button, an icon will appear in the status area of the task bar. Code Insight has five status icons:

Table 12.2: Status of Code Insight

Icon	Status	Description
	Disabled	Code Insight is disabled.
	Loading	Code Insight is analyzing the default PBT and generating the supported and unsupported feature list based on the analysis result.

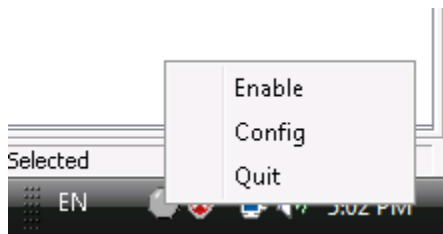
Icon	Status	Description
	Off	Code Insight is enabled but ineffective.
	On	Code Insight is enabled and effective in the current application.
	Gray	This icon appears anytime the PowerBuilder IDE loses focus, even if Code Insight is disabled.

12.1.1 Configuring Code Insight

Before you can enable Code Insight for a PowerBuilder application, you need to configure the PBT file of the desired application as the default PBT file. You can also specify the Apeon feature sets and the feature types for which feature list will be displayed.

Step 1: Click on the icon of Code Insight, and the status-area-icon menu pops up with three items: Enable (or Disable), Config, and Quit, as shown in the following figure.

Figure 12.1: The status-area-icon menu



Step 2: Click **Config** to open the **Code Insight Configuration** window, as shown in the following figure.

Figure 12.2: Code Insight Configuration window - PBT Files tab

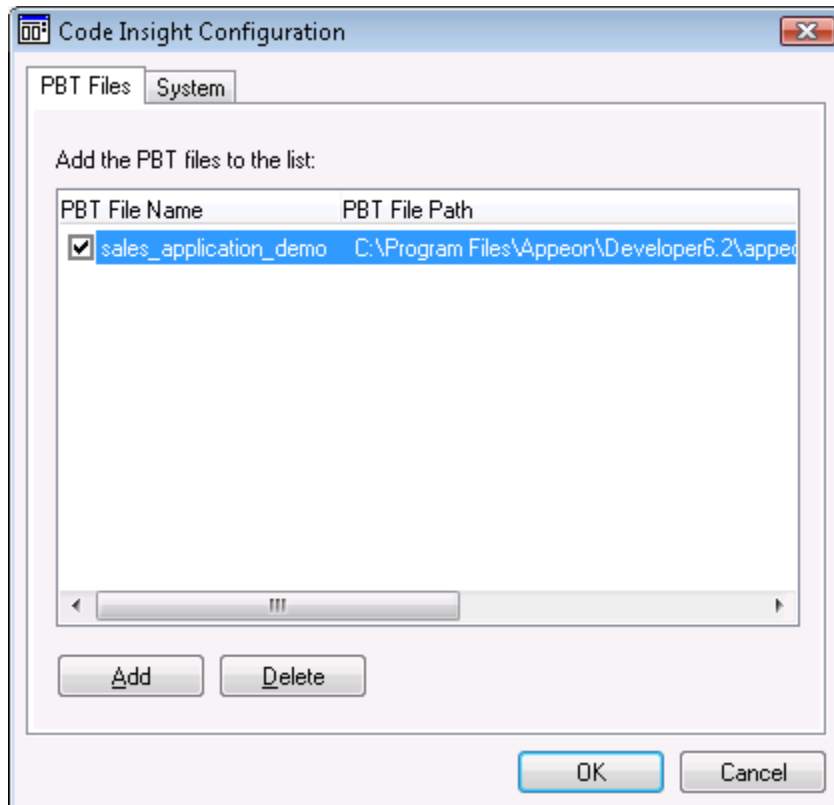
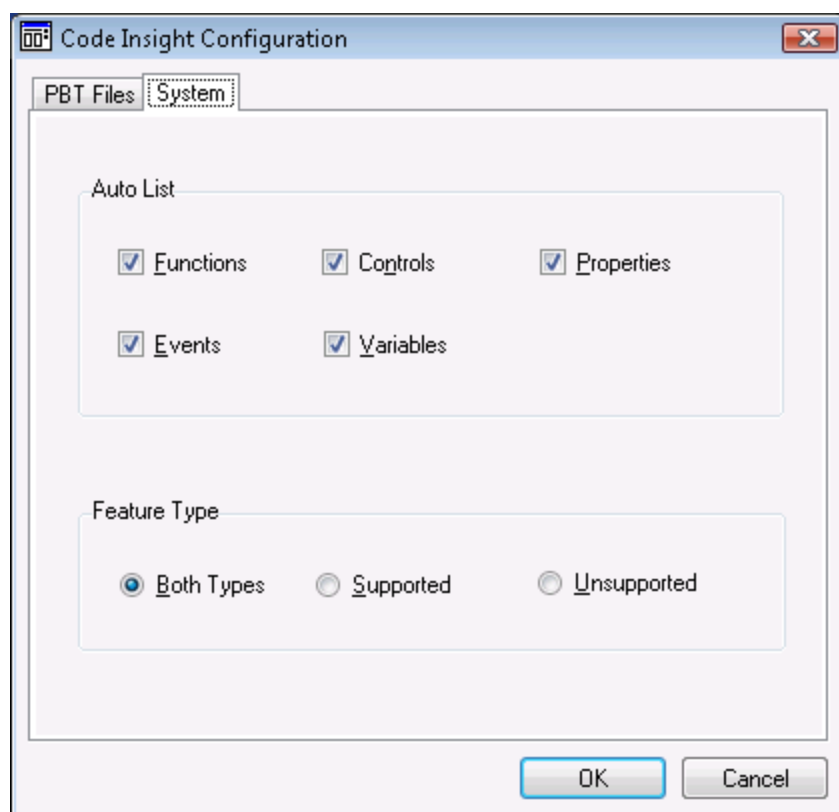


Figure 12.3: Code Insight Configuration window - System tab



The **Code Insight Configuration** window contains two tab pages where you can manage the PBT files and specify what displays in the list.

Table 12.3: Code Insight Configuration window

Select this tab	To do this	Detailed Steps
PBT File tab page	Add the PBT files.	<ol style="list-style-type: none"> 1. Click Add. 2. Search and select the PBT file in the common dialog box. 3. Click Open in the common dialog box. 4. Click OK.
	Delete the PBT files.	<ol style="list-style-type: none"> 1. Select the PBT file you want to delete. 2. Click Delete to delete the selected PBT file. 3. Click OK. <p>NOTE: The default PBT file cannot be deleted.</p>
	Set the default PBT file.	<ol style="list-style-type: none"> 1. Select the check box in front of a PBT file. 2. Click OK.

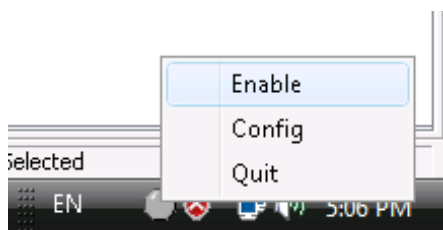
Select this tab	To do this	Detailed Steps
		<p>NOTE: 1) only one PBT file can be set to the default PBT.</p> <p>2) Code Insight can only be effective to the default PBT.</p>
System tab page	Specify what to display in the list.	<ol style="list-style-type: none"> 1. In the Auto List group box, opt to automatically display functions, events, properties, controls and/or variables in the list. 2. In the Application Type group box, opt to display only the unsupported features and/or the supported features for Web applications or/and Mobile applications. 3. In the Feature Type group box, opt to display only the unsupported features and/or the supported features in the list. 4. Click OK.

12.1.2 Enabling Code Insight

After you complete the configuration, you can enable Code Insight and use it to develop the PowerBuilder application.

Click on the icon of Code Insight and select **Enable** from the status-area-icon menu, as shown in the following figure.


Figure 12.4: Enable Code Insight



The Code Insight icon appears in one of the following statuses:

Table 12.4: Enable Code Insight

Status	Detailed Steps
	<p>Code Insight remains disabled.</p> <p>This happens when the PBT File list in the Code Insight Configuration window is empty. Make sure you have added the PBT file to the list.</p>
	<p>Code Insight is enabled but ineffective in the current application. To make it effective, make sure you have</p>

Status	Detailed Steps
	<ol style="list-style-type: none"> 1. Configured the PBT file of the current application as the default PBT file in the Code Insight Configuration window. 2. Opened an object of the current application in a painter (not in the Source editor) by double-clicking it in the Workspace or selecting Edit from the popup menu. <p>After taking the above steps, Code Insight becomes effective (On).</p>
	<p>Code Insight is enabled and effective.</p> <p>You can develop an application with the assistance of Code Insight.</p>

12.2 Coding with Code Insight

When Code Insight is enabled and effective (On) for an application, you can start editing the objects of the application using Code Insight. For example, take the following steps to add a new control and use Code Insight to edit it:

Step 1: Add a button to the current object in the painter.

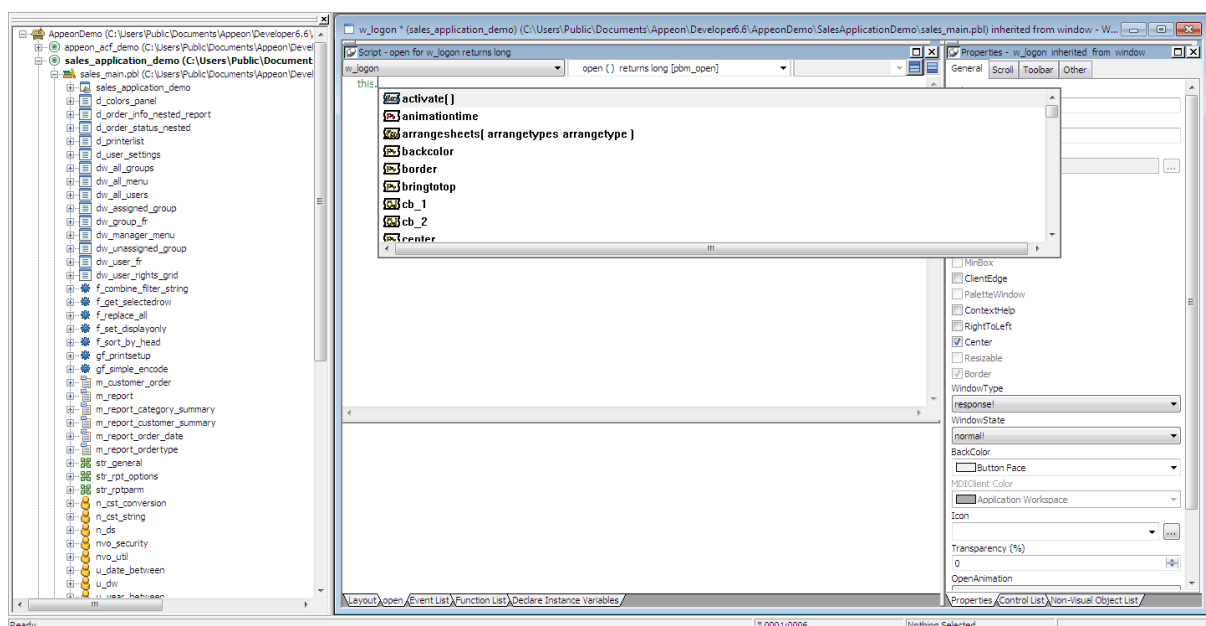
Step 2: Save the object by clicking the **Save** button on the PowerBar.

Note: Saving the object enables Code Insight to generate the supported and unsupported feature list for the new content. Code Insight cannot detect the new content unless the object is saved.

Step 3: Double click the button to open the Script view and pause for a few seconds after typing an identifier followed by a dot (for example, pause after **This.**).

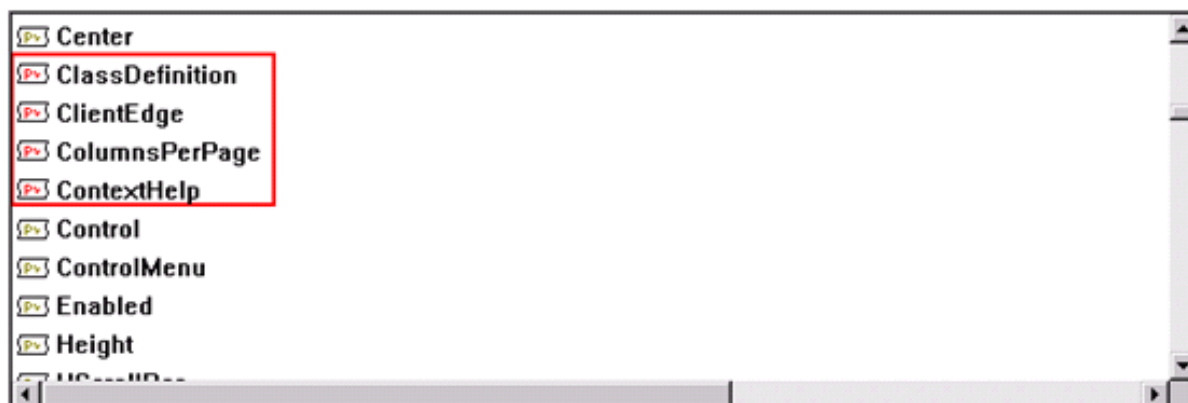
If there is more than one property, variable, method, or statement that could be inserted, the Code Insight feature list pops up for you to select a supported property, variable or method, as shown in the following figure.

Figure 12.5: Develop an application with Code Insight



The unsupported properties, variables or methods in the popup list will have a red icon in front, as shown in the following figure. Avoid using the unsupported features.

Figure 12.6: Apeon unsupported features displayed in popup window



Step 4: Select a supported property, variable or method from the popup list.

You can use Code Insight to help develop an application free of unsupported features and suitable for migration.

13 Launching AEM

AEM is a Web-based application that manages the converted Web or mobile applications and PowerServer over the Internet, an intranet or an extranet. It includes a comprehensive set of easy-to-use tools for system configuration, system maintenance, performance optimization, and application security.

AEM is included when PowerServer is installed.

AEM is accessible from any Client's Web browser or it is possible to access AEM through the **AEM** button that resides on the PowerServer Toolkit. Clicking the **AEM** button will automatically open a Web browser window and load the AEM entry page.

13.1 Requirements

Before launching AEM, verify that:

1. The PowerServer hosting AEM has been started.
2. You have correctly specified the host name and port of the PowerServer in the PowerServer Profile Configuration page.

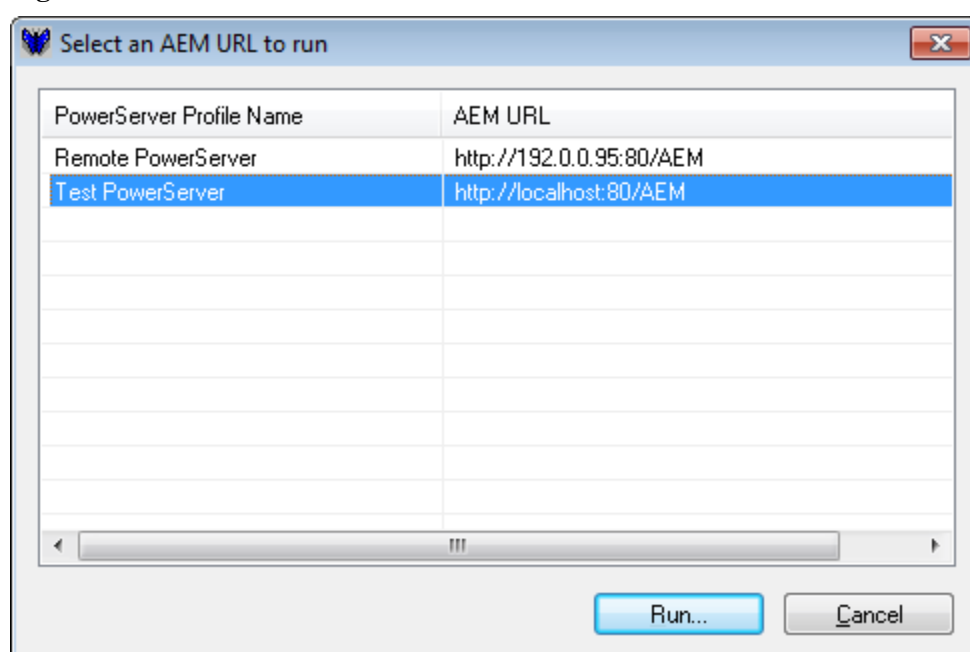
13.2 Launching AEM

Step 1: Click the **AEM** button () on the PowerServer Toolkit.

If more than one PowerServer profile is configured, the **Select a AEM URL to run** page is displayed, as shown in the following figure. All PowerServer profiles and corresponding AEM URLs are listed. The PowerServer configured as the default server is highlighted and selected.

If only one PowerServer profile is configured, the entry page of AEM is displayed.

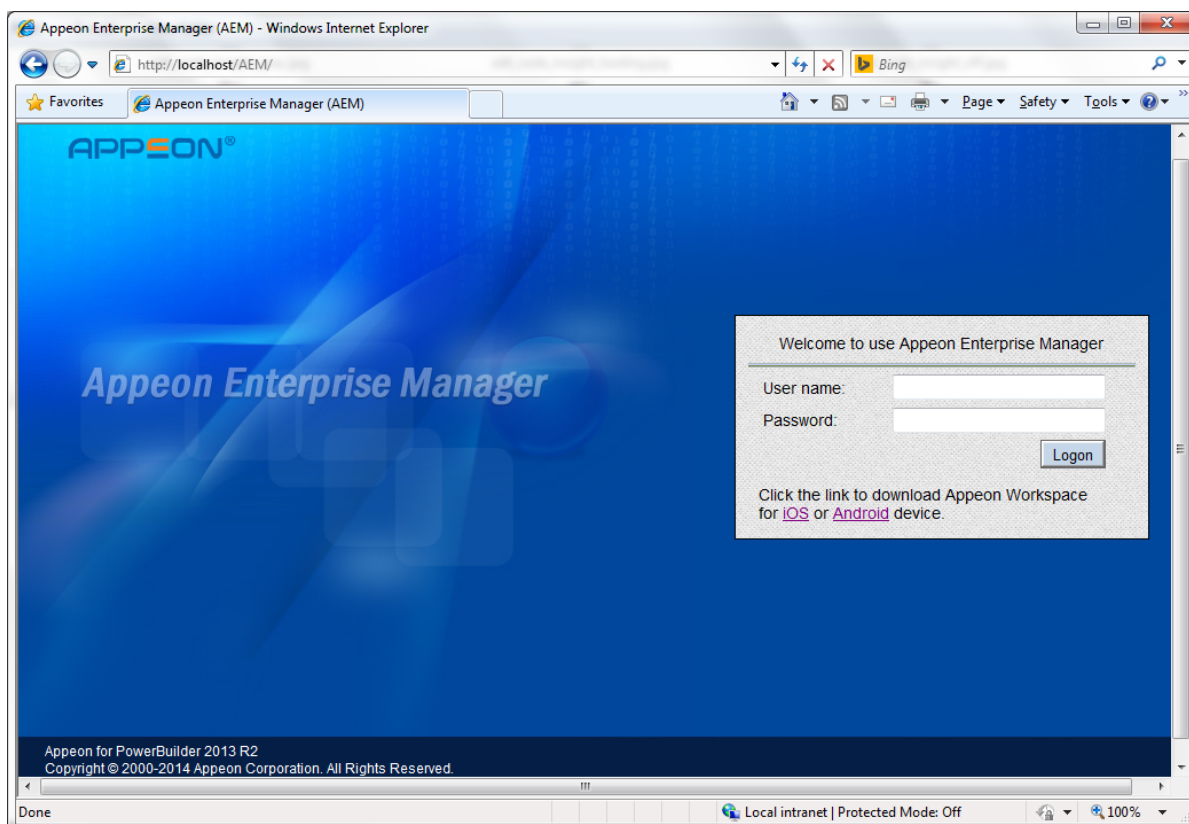
Figure 13.1: Select a AEM URL to run



Step 2: Select a AEM URL and click **Run**, as shown in the following figure.

The entry page of AEM is loaded in the Web browser, as shown in the following figure.

Figure 13.2: AEM entry page



The AEM URL is automatically entered into the Web browser address bar when you click the **Run** button. You can configure AEM's URL and connection method (https or http) in the PowerServer Profile Configuration page.

The default username is *admin* and the default password is *admin*. The username and password settings for AEM can be modified in Security settings in AEM. Refer to Section 5.3.5.1, “AEM login” in *PowerServer Configuration Guide for .NET* or in *PowerServer Configuration Guide for J2EE* for more information.

14 Appeon PowerServer Help

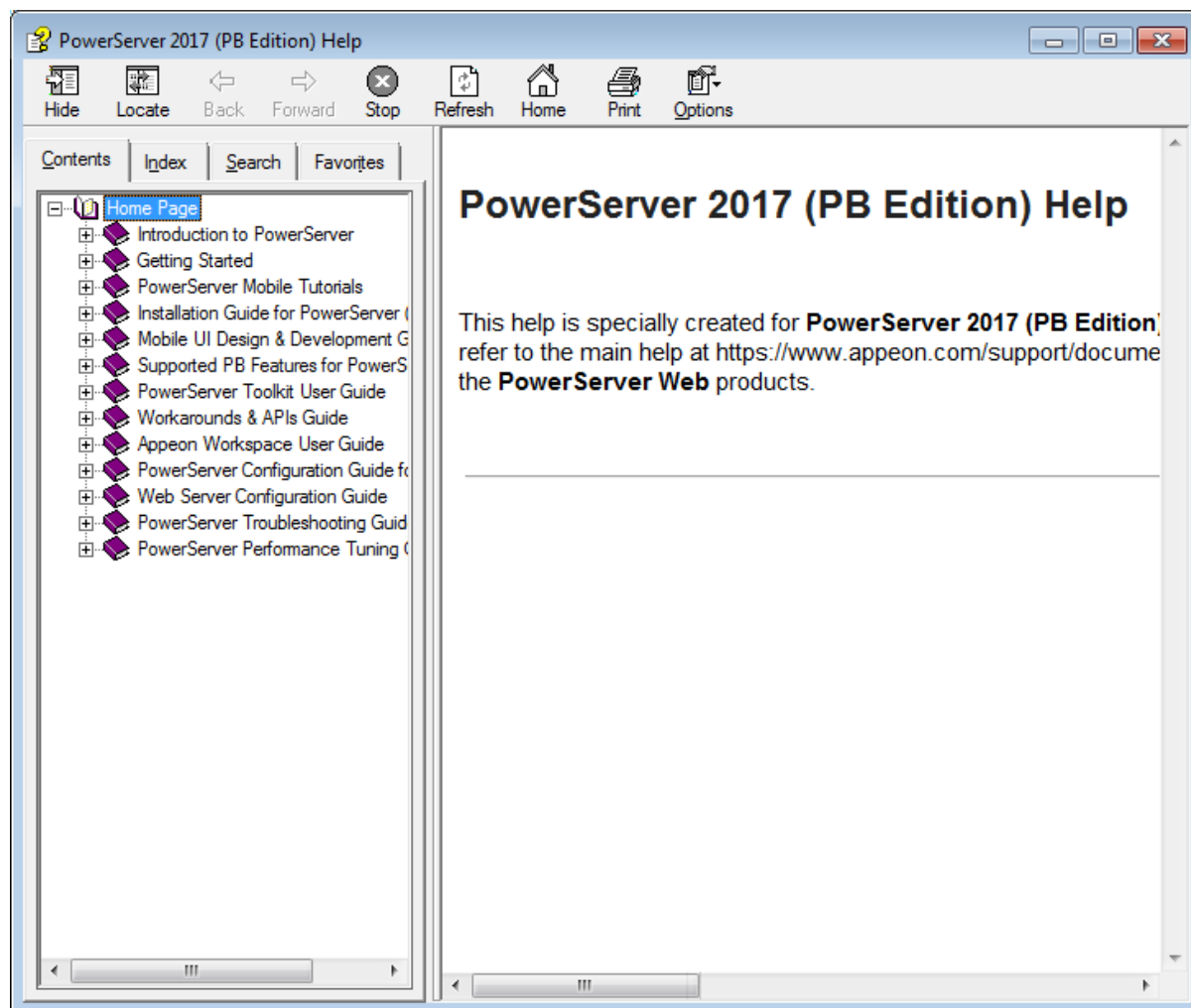
Appeon PowerServer Help is in compiled HTML format, so that you can search by index or by typing keywords, navigate back and forth, or print a specified page.

To access Appeon HTML Help:

Step 1: Click the **Help** button (?) in the PowerServer Toolkit.

Step 2: Click the corresponding help book to access the detailed help information, as shown in the following figure.

Figure 14.1: Appeon PowerServer Help



15 Technical Support

You can get technical support by clicking the **Get Support** icon in the PowerServer Toolkit appearing in the PowerBuilder IDE or launching the PowerServer Help from the Windows start menu.

Get support for Appeon PowerServer 2017

Please send your email to <support@appeon.com> for technical support and be prepared to provide the following files listed in the following table to Appeon Technical Support. This will aid debugging.

Table 15.1: Files required for technical support

File Type	File Name
PowerServer configuration file	For WebLogic\WebSphere\JEUS\JBoss: All files in the %powerserver%\repository\<instancename>\config folder. For .NET IIS: All files in the %powerserver%\AEM\config folder. %powerserver% indicates the PowerServer installation path.
PowerServer log file	For WebLogic\WebSphere\JEUS\JBoss: All files in the %powerserver%\repository\<instancename>\log folder. For .NET IIS: All files in the %powerserver%\AEM\Log folder.
PowerServer Repository DB	For .NET IIS: All files in the %powerserver%\AEM\db folder. For WebLogic\WebSphere\JEUS\JBoss: %powerserver%\db\appeondb.script
Appeon license file	For WebLogic\WebSphere\JEUS\JBoss: %powerserver%\license.appeon For .NET IIS: %powerserver%\AEM\bin\license.appeon
Web Server configuration file	httpd.conf
Web Page file	Web server path\application name>window name.html Web server path\application name>window name.js Web server path\application name\all DataWindows on window.xml Note: If the window named includes inheritance, then also supply all ancestor HTML and JavaScript files.

File Type	File Name
PowerBuilder exported source code file	PowerBuilder application path\exported window name.srw PowerBuilder application path\exported DataWindow names.srd Note: Exported windows should include exported ancestor windows and exported ancestor user objects.

Index

A

access UFA tool, [70](#)
activate Code Insight, [207](#)
add variables or expressions to Watch view, [101](#)
Additional Files, [27](#)
analyze an application, [69](#)
Apeon DataWindow menu, [119](#)
Apeon PowerServer help, [215](#)
application profile settings, [22](#)
 Additional Files, [27](#)
 basic settings, [23](#)
 DB settings, [25](#)
 Misc Settings, [31](#)
 Offline DB Settings, [39](#)
 runtime settings, [44](#)
 summary, [47](#)
 Web service profiles, [38](#)
application profile tab, [21](#)
 delete an application profile, [21](#)
 export or import an application profile, [22](#)
 specify default application profile, [21](#)

B

basic settings, [23](#)

C

change value of a variable or expression, [102](#)
code lines that can set breakpoints, [98](#)
code with Code Insight, [211](#)
configure application database connectivity, [7](#)
configure basic settings, [10](#)
configure Code Insight, [208](#)
configure deployment settings, [13](#)
configure PowerServer Toolkit, [9](#)
configure the project loader, [115](#)
customize and package Apeon Workspace, [182](#)
customize general settings of UFA report, [84](#)

D

DataWindow printing, [124](#)
DB settings, [25](#)
debug Apeon Web applications, [94](#)
declare transaction objects, [14](#)

define priority settings of unsupported features, [83](#)
delete an application profile, [21](#)
deploy PowerBuilder application, [85](#), [90](#)
deployment duration for full deployments, [86](#)
deployment duration for incremental deployments, [86](#)
deployment performance, [85](#)
deployment process, [87](#)
deployment profile settings, [61](#)
deployment profiles tab, [61](#)
develop with Code Insight, [207](#)
disable anti pop-up software, [111](#)

E

enable Apeon DataWindow menu, [120](#)
enable Code Insight, [210](#)
evaluate an expression, [103](#)
examine an application at a breakpoint, [100](#)
examine context in Call Stack view, [103](#)
export or import an application profile, [22](#)

F

filter UFA report items, [79](#)
find, [120](#)
fix code/stop the debug procedure, [103](#)

I

If you need help, [3](#)
install an application, [141](#)
install and uninstall an Apeon application, [140](#)
install Apeon PowerServer, [7](#)
install Apeon Workspace and mobile app, [116](#)
install IWA app, [117](#)
internet explorer settings, [104](#)

L

language setting requirements, [110](#)
launch AEM, [213](#), [213](#)
launch application from Run button, [113](#)

M

manage application profile, [20](#)
manage data source profile, [63](#)
manage database type profiles, [48](#)
manage deployment profiles, [60](#)
manage server profiles, [51](#)
manipulate UFA report, [78](#)

method for setting breakpoint, [99](#)
Misc Settings, [31](#)
modify deploy-config file, [139](#)
modify unsupported features, [76](#)

O

offline settings, [39](#)
open or save UFA report, [78](#)

P

package a server deployment project, [132](#)
package and install Appeon applications, [132](#)
package stand-alone mobile projects, [165](#)
perform feature analysis, [71](#)
PowerServer profile settings, [52](#)
PowerServer Toolkit in PowerBuilder, [4](#)
prepare PowerBuilder application, [87](#)
Preparing for the mobile package, [155](#)

R

Related documents, [2](#)
run Appeon application, [104](#), [113](#)
run application in debug mode, [100](#)
runtime settings, [44](#)

S

SaveAs, [125](#)
search for UFA report items, [79](#)
select a run mode for the Web app, [118](#)
select DB types, [14](#)
select External files, [18](#)
select image files, [17](#)
select INI files, [17](#)
select PBL files, [12](#)
select report view mode, [79](#)
server profile tab, [51](#)
set breakpoint, [98](#)
sort and filter, [121](#)
special variable and expression handlings, [100](#)
specify default application profile, [21](#)
specify deployment settings, [87](#)
speed of deployment process, [85](#)
start Appeon Debugger, [95](#)
start PowerServer, [7](#)
step through application, [103](#)
summary, [19](#), [47](#)

T

technical support, [216](#)

U

undeploy Appeon applications, [199](#)
undeploy instructions, [199](#)
undeploy with Deployment Profile mode, [200](#)
undeploy with PowerServer mode, [203](#)
undetected unsupported features, [73](#)
uninstall an application, [149](#)
URLs of Appeon applications, [112](#)
use Appeon DataWindow menu, [120](#)
 DataWindow printing, [124](#)
 find, [120](#)
 SaveAs, [125](#)
 sort and filter, [121](#)
use Information Manager, [129](#)
use UFA tool, [68](#)

V

view report and log, [129](#)
views in Appeon Debugger, [97](#)

W

Web Server profile settings, [55](#)
Web service profiles, [38](#)