Workarounds & API Guide

Appeon for PowerBuilder 2013

FOR WINDOWS & UNIX & LINUX

DOCUMENT ID: ADC20238-01-0660-01

LAST REVISED: May 08, 2013

Contents

1 About This Book

1.1 Audience

This book is for users who want to get resolutions for issues encountered when using Appeon Web or Appeon Mobile.

1.2 How to use this book

There are four chapters in this book.

Chapter 1: About This Book

A general description of this book.

Chapter 2: Appeon Workarounds PBL Reference

Lists the syntax and code examples of the programming interfaces provided by the Appeon Workarounds PBL, to implement corresponding functionalities.

Chapter 3: Workarounds for Unsupported Features

Provides suggestions to work around the unsupported features that have functional impact on the running of the application.

Chapter 4: FAQ and Workarounds

Lists some frequently asked questions and workaround tips regarding the Appeon application architecture or product features.

1.3 Related documents

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

- Introduction to Appeon:

  Guides you through all the documents included in Appeon for PowerBuilder.

- New Features Guide:

  Introduces new features and changes in Appeon for PowerBuilder.

- Appeon Mobile Tutorials:

  Gives instructions on deploying, running, and debugging Appeon applications, distributing native mobile apps, and configuring Appeon server clusters.

- Appeon Installation Guide:

  Provides instructions on how to install Appeon for PowerBuilder successfully.

- Development Guidelines for Appeon Mobile:

  Introduces general guidelines on developing apps with Appeon Mobile.

- Migration Guidelines for Appeon 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 the user through the entire process of deploying a small PowerBuilder application to the Web.

- Features Help for Appeon:

  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.

- Appeon Developer User Guide:

  Provides instructions on how to use the Appeon Developer toolbar in Appeon for PowerBuilder.

- Workarounds & API Guide:

  Provides resolutions for issues, such as unsupported features, etc,. encountered when using Appeon for PowerBuilder.

- Appeon Workspace User Guide:

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

- Appeon Server Configuration Guide:

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

- Web Server Configuration Guide:

  Describes configuration instructions for Web Servers to work with a single Appeon Server or an Appeon Server cluster.

- Troubleshooting:

  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 QTP:

  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 Appeon Workarounds PBL Reference

This chapter lists the syntax and code examples of the programming interfaces provided by the Appeon Workarounds PBL, to implement the various functionalities in Appeon Web application and Appeon mobile application.

2.1 Introduction to Appeon Workarounds

Appeon for PowerBuilder supports most features of PowerBuilder, which almost fully meets users' needs. However, due to the difference between Client/Server architecture and Browser/Server architecture, some limitations still exist when converting PowerBuilder applications to the Web or Mobile devices. Appeon provides the Appeon workarounds to work around these limitations and provide useful function extensions for the deployed applications.

Appeon workarounds includes one PBL, three DLLs and one XML file to help to work around some PowerBuilder features and extend the usability of the deployed application.

- One PBL: appeon_workarounds.pbl
- Three DLLs: EonAXNVO.dll, EonEmfPic.dll and Eonejbclient.dll
- XML file: ejb_err_info.xml

They are located in the \appeon_workarounds**xxx**e (**xxx** indicates the PowerBuilder version) folder under the Appeon Developer installation directory. For example, C:\Program Files \Appeon\Developer2013\appeon_workarounds125e.

2.2 Important Notes for calling APIs

Not all of the APIs can be used in both the Mobile and Web environment. In the General APIs section, the following functions: of_popmenu, of_popmenuon, of_print2file, of_print2pdf, Getbrowserversion, getiehandle, Getieurl, & appeonisin64browser from the **AppeonExFuncs object** are effective in the Web environment only. In the Mobile APIs section, all listed functions are effective in the Mobile environment. If a function that is effective in the Web environment only is executed in a Mobile environment, or if a function that is effective in the Mobile environment only is executed in a Web environment, the function may return unexpected values, and cause the application to perform abnormally.

Therefore, to avoid the aforementioned problem, you should detect the running environment first before calling the corresponding functions, for example, to call the of_setapprotationlock API, you are recommended to write the scripts in this way:

```
if appeongetclienttype()="MOBILE" then
   eon_mobile_awsex lgnv_aws
   lgnv_aws = CREATE eon_mobile_awsex
   lgnv_aws.of_setAppOrientation(2)
   lgnv_aws.of_setapprotationlock(1)
    destroy lgnv_aws
end if
```

2.3 General API

Most of the following general APIs implement the various functionalities in both Appeon Web applications and Appeon Mobile applications.

2.3.1 AppeonExtFuncs object

AppeonExtfuncs object provides functions to help users manipulate the deployed application. This object is defined in the Appeon Workarounds PBL. If you want to use the functions of this object in your PowerBuilder application, add the library to the Library Search Path of the application.

The function of the AppeonExtfuncs object usually returns different values in the PowerBuilder application and in the deployed application. The values returned in the PowerBuilder application are defined in the AppeonExtfuncs object; while the values returned in the deployed application are defined in the Appeon client library.

The function of AppeonExtfuncs object performs almost identical to the global function of the Appeon client functions (Obsolete). Typically, the function of AppeonExtfuncs object is recommended over the Appeon client functions (Obsolete) for the following three reasons:

- The Appeon client functions are obsolete functions and will be discontinued in a future release.
- The function of the AppeonExtfuncs object delivers a better extensibility and is much easier to manage.
- Calling too much global functions of Appeon client functions (Obsolete) may affect the performance.

Calling functions of AppeonExtFuncs object

The best way to use these functions is to pass their return values into Appeon Server NVO components. Then, in the Appeon Server NVO components, the information (such as browser version, user name etc.) can be utilized to code more application features such as security authentication, auditing, logging, file operation, etc. This means you can write more scripts in NVO components for implementing more application features.

Below lists the function of AppeonExtfuncs object and their corresponding Appeon client functions (Obsolete).

Table 2.1: Functions of AppeonExtFuncs object

Function of AppeonExtFuncs object	Appeon client functions	Description
of_getappeonusername function	AppeonGetAppeonUserName function	Gets the user name that you type into the Appeon Login dialog box.
of_getbrowserversion function	AppeonGetBrowserVersion function	Gets the Internet Explorer version of the client.
of_getcachedir function	AppeonGetCacheDir function	Gets the Cache directory that is used by the current Web application. Generally the cache directory is \Documents and Settings\username\Application Data \appeon\application_name.
of_getclientid function	AppeonGetClientID function	Gets the unique session identifier for the Internet Explorer client.

Function of AppeonExtFuncs object	Appeon client functions	Description
of_getclientip function	AppeonGetClientIP function	Gets the IP address of the Internet Explorer client.
of_getclienttype function	AppeonGetClientType function	Gets the type of an application.
of_gethttpinfo function	AppeonGetHttpInfo function	Gets the HTTP header information from a particular request.
of_getiehandle function	AppeonGetIEHandle function	Gets the Internet Explorer handle for the current application.
of_getieurl function	AppeonGetIEURL function	Gets the URL of the current application.
of_getostype function	AppeonGetOSType function	Gets the type of OS that runs your application (the Appeon Web application, the Appeon mobile application, or the PowerBuilder client application).
NA	AppeonGetRemainingdays function	Gets the remaining day(s) of license or technical support.
of_getservertype function	AppeonGetServerType function	Gets the Appeon Server type where the application runs.
of_getsessioncount function	AppeonGetSessionCount function	Gets the total number of active sessions currently on a cluster. If there is no cluster configured in AEM, it gets all the active sessions for a single named application or all applications in Appeon Server. This function only works in EAServer 5.x.
of_is64browser function	appeonisin64browser function	Detects if the IE browser where the application runs is 64-bit.
of_ldaplogon function	AppeonLDAPLogon Function	Logs in to the LDAP server with the specified user name and password.
of_popmenu function	AppeonPopMenu function	Pops up Appeon DataWindow menu at a specified position in a specified DataWindow control.
of_popmenuon function	AppeonPopMenuOn function	Pops up Appeon DataWindow menu in a specified window when you right click the mouse button.
of_print2file function	AppeonPrint2File function	Saves the specified DataWindow as image files of BMP, JPG or GIF format.

Function of AppeonExtFuncs object	Appeon client functions	Description
of_Print2PDF function (Obsolete)	AppeonPrint2PDF function (Obsolete)	of_Print2PDF is an obsolete function and will be discontinued in a future release. Please replace it with the SaveAs function of the DataWindow or Child DataWindow and set the saveas file type to PDF format.
of_switchRealTimeCalc function	AppeonSwitchRealTimeCalc function	Performs the DataWindow real-time expression calculation in time or performs the calculation for only one time in the whole life-cycle.

2.3.1.1 of_getappeonusername function

Description

Gets the user name that you type into the Appeon Login dialog box.

Syntax

appeonextfuncs.of_getappeonusername( )

Table 2.2:

Argument	Description
appeonextfuncs	A reference to an AppeonExtFuncs object.

Return value

String.

Usage

This function takes effect in the deployed Appeon application, not in the PowerBuilder application.

2.3.1.2 of_getbrowserversion function

Description

Gets the Internet Explorer version of the client.

Unsupported on Mobile client.

Syntax

appeonextfuncs.of_getbrowserversion( )

Table 2.3:

Argument	Description
appeonextfuncs	A reference to an AppeonExtFuncs object.

Return value

String.

Usage

This function takes effect in the deployed Appeon application, not in the PowerBuilder application.

2.3.1.3 of_getcachedir function

Description

Gets the Cache directory that is used by the current application.

Generally, for the Web application, the cache directory is \Documents and Settings\username \Application Data\appeon\application_name.

For the Mobile application, the cache directory is /$AppeonMobile folder$/Documents/ your_application_folder, for example, /var/mobile/Applications/144F5F33-A33F-480D-A3D9-01BBA5410EB2/Documents/4c001b05.

Syntax

appeonextfuncs.of_getcachedir( )

Table 2.4:

Argument	Description
appeonextfuncs	A reference to an AppeonExtFuncs object.

Return value

String.

Usage

This function takes effect in the deployed Appeon application, not in the PowerBuilder application.

2.3.1.4 of_getclientid function

Description

Gets the unique session identifier for the Web or mobile client.

Syntax

appeonextfunc.of_getclientid( )

Table 2.5:

Argument	Description
appeonextfuncs	A reference to an AppeonExtFuncs object.

Return value

String.

2.3.1.5 of_getclientip function

Description

Gets the IP address of the Web or mobile client.

Syntax

appeonextfunc.of_getclientip( )

Table 2.6:

Argument	Description
appeonextfuncs	A reference to an AppeonExtFuncs object.

Return value

String.

2.3.1.6 of_getclienttype function

Description

Gets the type of an application.

Syntax

appeonextfunc.of_getclienttype( )

Table 2.7:

Argument	Description
appeonextfuncs	A reference to an AppeonExtFuncs object.

Return value

Table 2.8:

String.	Returns "WEB" if the application runs on the Web. Returns "PB" if the application runs in PowerBuilder. Returns "MOBILE" if the application runs on a mobile device.

2.3.1.7 of_gethttpinfo function

Description

Gets the HTTP header information from a particular request.

Syntax

appeonextfuncs.of_gethttpinfo(string *attribute*)

Table 2.9:

Argument	Description
appeonextfuncs	A reference to an AppeonExtFuncs object.
attribute	The required HTTP information. For example, "Host", "Cookie", etc.

Return value

String.

Usage

This function takes effect in the deployed Appeon application, not in the original PowerBuilder application.

2.3.1.8 of_getiehandle function

Description

Gets the Internet Explorer handle for the current application.

Unsupported on Mobile client.

Syntax

appeonextfuncs.of_getiehandle( )

Table 2.10:

Argument	Description
appeonextfuncs	A reference to an AppeonExtFuncs object.

Return value

Long.

Usage

This function takes effect in the deployed Appeon application, not in the PowerBuilder application.

2.3.1.9 of_getieurl function

Description

Gets the URL of the current application.

Unsupported on Mobile client, use of_getappinfo instead of_getieurl in the mobile application.

Syntax

appeonextfuncs.of_getieurl( )

Table 2.11:

Argument	Description
appeonextfuncs	A reference to an AppeonExtFuncs object.

Return value

String.

Usage

This function takes effect in the deployed Appeon application, not in the PowerBuilder application.

2.3.1.10 of_getostype function

Description

Gets the type of OS that runs your application (the Appeon Web application, the Appeon mobile application, or the PowerBuilder client application).

Syntax

appeonextfuncs.of_getostype( )

Table 2.12:

Argument	Description
appeonextfuncs	A reference to an AppeonExtFuncs object.

Return value

Table 2.13:

String.	Returns the type of OS that runs the Appeon Web application, the Appeon mobile application, or the PowerBuilder client application.

2.3.1.11 of_getservertype function

Description

Gets the Appeon Server type where the application runs.

Syntax

appeonextfuncs.of_getservertype()

Table 2.14:

Argument	Description
appeonextfuncs	A reference to an AppeonExtFuncs object.

Return value

Table 2.15:

Integer	Returns 1 if the Appeon applications runs on an Appeon Server that is installed to a Java Server (such as EAServer). Returns 2 if the Appeon applications runs on an Appeon Server that is installed to a .NET IIS server.

Usage

This function takes effect in the deployed Appeon application, not in the PowerBuilder application.

2.3.1.12 of_getsessioncount function

Description

Gets the total number of active sessions currently on a cluster. If there is no cluster configured in AEM, it gets all the active sessions for a single named application or all applications in Appeon Server. This function only works in EAServer 5.x.

Syntax

appeonextfuncs.of_getsessioncount (String s*ervername*, String a*ppname*)

Table 2.16:

Argument	**Description**
appeonextfuncs	A reference to an AppeonExtFuncs object.
servername	The name of the Appeon Server that the sessions are created in. If the Appeon server is clustered, input the IP address of the server for this argument.
appname	The name of the application that is deployed to the Appeon Server which is specified in servername.

Return value

Long.

Usage

This function takes effect in the deployed Appeon application, not in the PowerBuilder application.

2.3.1.13 of_is64browser function

Description

Detects if the IE browser where the application runs is 64-bit.

Unsupported on Mobile client.

Syntax

appeonextfuncs.of_is64browser()

Table 2.17:

Argument	**Description**
appeonextfuncs	A reference to an AppeonExtFuncs object.

Return value

Table 2.18:

Boolean.	Returns "true" if the Appeon application runs on a 64-bit IE browser. Returns "false" if the Appeon application runs on a 32-bit IE browser.

Usage

This function takes effect in the deployed Appeon application, not in the PowerBuilder application.

2.3.1.14 of_ldaplogon function

Description

Logs in to the LDAP server with the specified user name and password.

Syntax

appeonextfuncs.of_ldaplogon(String s*ervername*, String a*ppname*)

Table 2.19:

Argument	Description
appeonextfuncs	A reference to an AppeonExtFuncs object.
as_username	User name for logging into the LDAP server.
as_password	Password for logging into the LDAP server.

Return value

Table 2.20:

String.	Returns "" an empty string if the login succeeds. Returns the error information.

Usage

This function takes effect in the deployed Appeon application, not in the PowerBuilder application. To make this function work in the Appeon application, you should set the System Security to ON in AEM, and make sure the LDAP server is correctly set up. For detailed instructions, refer to the "System Security" section in the Server Configuration Guide.

2.3.1.15 of_popmenu function

Description

Pops up Appeon DataWindow menu at a specified position in a specified DataWindow control.

Appeon DataWindow Menu is available in Appeon Web only. For more information about the menu items, refer to **Appeon DataWindow Menu** section in the *Appeon Developer User Guide*.

Supported on Web client only.

Syntax

appeonextfuncs.of_popmenu (datawindow *adw_dw* , Integer *nx* , Integer *ny*)

Table 2.21:

Argument	Description
appeonextfuncs	A reference to an AppeonExtFuncs object.
adw_dw	The DataWindow control on which you want to pop up the Appeon DataWindow menu.

nx	The distance from the left edge of the popup menu.
ny	The distance from the up edge of the popup menu.

Return value

None.

Usage

1. The user customized RMB menu has a higher priority than the Appeon customized menu.
2. The AppeonPopMenu function has a higher priority than the AppeonPopMenuOn function.
3. Defining your RMB menu in RButtonDown event is not recommended because this will cause confusion in the system. To work around this, define your RMB menu in the RButtonUp event.

2.3.1.16 of_popmenuon function

Description

Pops up Appeon DataWindow menu in a specified window when you right click the mouse button.

Appeon DataWindow Menu is available in Appeon Web only. For more information about the menu items, refer to **Appeon DataWindow Menu** section in the *Appeon Developer User Guide*.

Supported on Web client only.

Syntax

appeonextfuncs.of_popmenuon (datawindow *adw _dw*, Boolean *ab_show*)

Table 2.22:

Argument	Description
appeonextfuncs	A reference to an AppeonExtFuncs object.
adw_dw	The DataWindow control on which you want to show the Appeon DataWindow menu.
ab_show	Gives an option whether to display the Appeon DataWindow menu. True - Display the Appeon DataWindow menu. False - Not to display the Appeon DataWindow menu.

Return value

None.

Usage

1. The user customized RMB menu has a higher priority than the Appeon customized menu.
2. The AppeonPopMenu function has a higher priority than the AppeonPopMenuOn function.

3. Defining your RMB menu in RButtonDown event is not recommended because this will cause the confusion of the system. To work around this, define your RMB menu in the RButtonUp event.

2.3.1.17 of_print2file function

Description

Saves the specified DataWindow as image files of BMP, JPG or GIF format.

Supported on Web client only.

Syntax

appeonextfuncs.of_print2file (datawindow adw, string asoutpath, string asoutname, long alouttype)

Table 2.23:

Argument	Description
adw	The DataWindow object to be saved as image files.
asoutpath	The path of image files.
asoutname	The specified name of image files.
alouttype	The format type of files: 1-BMP; 2-JPG; 3-GIF

Return value

Table 2.24:

Integer.	Returns 1 if it succeeds in saving the specified DataWindow as image files. Returns -1 if an unknown error occurs. Returns -2 if alouttype is an unsupported format. Returns -3 if adw is an invalid DataWindow object, DataStore object or DataWindowChild object. Returns -4 if it fails in creating a file, e.g., the specified path does not exist or without access rights. Returns -5 if it fails in creating device context, e.g., user sets a large size when customizing page property. Returns -6~-12 if an internal error occurs.

Usage

1. This function is used to execute saving DataWindow as image files.

2. If the page size is large enough, a DataWindow is saved as one file; if the size of a DataWindow surpasses the page size, the DataWindow is saved as several files.

3. The asoutname is the file name specified by user, for example, the function is appeonprint2file(adw, “c:\”, “appeon”, 1), if a DataWindow is saved as one file, the file is like C:\appeon.bmp; if a DataWindow is saved as several files, the files are like C:\appeon1.bmp, C:\appeon2.bmp, ..., C:\appeonN.bmp. Another example, the function is appeonprint2file(adw, “c:\”, “test.bmp”, 1), if a DataWindow is saved as one file, the file is like C:\ test.bmp.bmp; if a DataWindow is saved as several files, the files are like C:\test.bmp1.bmp, C:\test.bmp2.bmp, ..., C:\test.bmpN.bmp.

4. The upper limit of page size is restrained by the type of operating system. For instance, in Windows Vista, the upper limit of customized page size is 5500*5500 around. However, the page size can also be 1024*10000 by reducing page width and increasing page height.

2.3.1.18 of_Print2PDF function (Obsolete)

Obsolete function

of_Print2PDF is an obsolete function and will be discontinued in a future release. Please replace it with the SaveAs function of the DataWindow or Child DataWindow and set the saveas file type to PDF format.

Supported on Web client only.

2.3.1.19 of_switchRealTimeCalc function

Description

Performs the DataWindow real-time expression calculation in time or performs the calculation for only one time in the whole life-cycle.

Syntax

appeonextfuncs.of_switchRealTimeCalc (powerobject adw, integer para)

Table 2.25:

Argument	Description
adw	The DataWindow/DataStore/DataWindowChild object.

Return value

Table 2.26:

Integer.	Returns 0 (default value) if need to perform real-time calculation. Returns 1 if no need to perform real-time calculation.

2.3.2 Appeon Client Functions (Obsolete)

Appeon also provides a set of PowerBuilder global functions (called Appeon client functions) which perform identical functionalities to the function of AppeonExtFuncs object. However, Appeon client functions are **obsolete** and will be discontinued in a next release, therefore, please replace the Appeon client function with the equivalent fucntion of AppeonExtFuncs object.

2.3.2.1 AppeonGetAppeonUserName function

Description

Gets the user name that you type into the Appeon Login dialog box.

Syntax

AppeonGetAppeonUserName( )

Return value

String.

Usage

This function takes effect in the deployed Appeon application, not in the PowerBuilder application.

2.3.2.2 AppeonGetBrowserVersion function

Description

Gets the Internet Explorer version of the client.

Syntax

AppeonGetBrowserVersion( )

Return value

String.

Usage

This function takes effect in the deployed Appeon application, not in the PowerBuilder application.

2.3.2.3 AppeonGetCacheDir function

Description

Gets the Cache directory that is used by the current Web application. Generally the cache directory is \Documents and Settings\username\Application Data\appeon\application_name.

Syntax

AppeonGetCacheDir( )

Return value

String.

Usage

This function takes effect in the deployed Appeon application, not in the PowerBuilder application.

2.3.2.4 AppeonGetClientID function

Description

Gets the unique session identifier for the Internet Explorer client.

Syntax

AppeonGetClientID( )

Return value

String.

2.3.2.5 AppeonGetClientIP function

Description

Gets the IP address of the Internet Explorer client.

Syntax

AppeonGetClientIP( )

Return value

String.

Usage

This function takes effect in the deployed Appeon application, not in the PowerBuilder application.

2.3.2.6 AppeonGetClientType function

Description

Gets the type of an application.

Syntax

AppeonGetClientType( )

Return value

Table 2.27:

String	Returns "WEB" if the application runs on the Web. Returns "PB" if the application runs in PowerBuilder. Returns "MOBILE" if the application runs on a mobile device.

2.3.2.7 AppeonGetHttpInfo function

Description

Gets the HTTP header information from a particular request.

Syntax

AppeonGetHttpInfo(string attribute)

Table 2.28:

Argument	Description
attribute	The required HTTP information. For example, "Host", "Cookie", etc.

Return value

String

2.3.2.8 AppeonGetIEHandle function

Description

Gets the Internet Explorer handle for the current application.

Syntax

AppeonGetIEHandle( )

Return value

Long.

Usage

This function takes effect in the deployed Appeon application, not in the PowerBuilder application.

2.3.2.9 AppeonGetIEURL function

Description

Gets the URL of the current application.

Syntax

AppeonGetIEURL( )

Return value

String.

Usage

This function takes effect in the deployed Appeon application, not in the PowerBuilder application.

2.3.2.10 AppeonGetOSType function

Description

Gets the type of OS that runs your application (the Appeon Web application, the Appeon mobile application, or the PowerBuilder client application).

Syntax

AppeonGetOSType( )

Return value

Table 2.29:

String.	Returns the type of OS that runs the Appeon Web application, the Appeon mobile application, or the PowerBuilder client application.

2.3.2.11 AppeonGetRemainingdays function

Description

Gets the remaining day(s) of license or technical support.

Syntax

AppeonGetRemainingdays (String as_type, ref string as_error)

Table 2.30:

Argument	Description
as_type	License or technical support that you want to get the remaining day(s). "license" indicates to get remaining day(s) of license. "support" indicates to get remaining day(s) of technical support.

as_error	An empty string or error messages.

Return value

Table 2.31:

Long.	Returns a number >0 if the license or technical support has remaining day(s). Returns 0 if there is no expiration date. Returns -1 if license or technical support has expired, Appeon Server has an exception, or parameter is invalid.

2.3.2.12 AppeonGetServerType function

Description

Gets the Appeon Server type where the application runs.

Syntax

AppeonGetServerType( )

Return value

Table 2.32:

Integer.	Returns 1if the application runs on an Appeon Server that is installed to a Java Server (such as EAServer). Returns 2 if the application runs on an Appeon Server that is installed to a .NET IIS server.

Usage

This function takes effect in the deployed Appeon application, not in the PowerBuilder application.

2.3.2.13 AppeonGetSessionCount function

Description

Gets the total number of active sessions currently on a cluster. If there is no cluster configured in AEM, it gets all the active sessions for a single named application or all applications in Appeon Server. This function only works in EAServer 5.x.

Syntax

AppeonGetSessionCount (String s*ervername* , String a*ppname*)

Table 2.33:

Argument	Description
servername	The name of the Appeon Server that the sessions are created in. If the Appeon server is clustered, input the IP address of the server for this argument.
appname	The name of the application that is deployed to the Appeon Server which is specified in servername argument.

Return value

Long.

Usage

It functions the same as the GetSessionCount interface which is provided by Appeon Server.

2.3.2.14 appeonisin64browser function

Description

Detects if the IE browser where the application runs is 64-bit.

Syntax

appeonisin64browser( )

Return value

Table 2.34:

Boolean.	Returns "true" if the application runs on a 64-bit IE browser. Returns "false" if the application runs on a 32-bit IE browser.

Usage

This function takes effect in the deployed Appeon application, not in the PowerBuilder application.

2.3.2.15 AppeonLDAPLogon function

Description

Logs in to the LDAP server with the specified user name and password.

Syntax

AppeonLDAPLogon ()

Return value

String.

Usage

This function takes effect in the deployed Appeon application, not in the PowerBuilder application. To make this function work in the deployed application, you should set the System Security to ON in AEM, and make sure the LDAP server is correctly set up.

2.3.2.16 AppeonPopMenu function

Description

Pops up Appeon DataWindow menu at a specified position in a specified DataWindow control.

Syntax

AppeonPopMenu (datawindow *adw_dw* , Integer *nx* , Integer *ny*)

Table 2.35:

Argument	**Description**
adw_dw	The DataWindow control on which you want to pop up the Appeon DataWindow menu.
nx	The instance from the left edge of the popup menu.
ny	The instance from the up edge of the popup menu.

Return value

None.

Usage

1. The user customized RMB menu has a higher priority than the Appeon customized menu.
2. The AppeonPopMenu function has a higher priority than the AppeonPopMenuOn function.
3. Defining your RMB menu in RButtonDown event is not recommended because this will cause confusion in the system. To work around this, define your RMB menu in the RButtonUp event.

2.3.2.17 AppeonPopMenuOn function

Description

Pops up Appeon DataWindow menu in a specified window when you right click the mouse button.

Syntax

AppeonPopMenuOn (datawindow *adw _dw*, Boolean *ab_show*)

Table 2.36:

Argument	Description
adw_dw	The DataWindow control on which you want to show the Appeon DataWindow menu.
ab_show	Gives an option whether to display or not display the Appeon DataWindow menu. True - Enables the display of the Appeon DataWindow menu. False - Disables the display of the Appeon DataWindow menu.

Return value

None.

Usage

1. The user customized RMB menu has a higher priority than the Appeon customized menu.
2. The AppeonPopMenu function has a higher priority than the AppeonPopMenuOn function.

3. Defining your RMB menu in RButtonDown event is not recommended because this will cause the confusion of the system. To work around this, define your RMB menu in the RButtonUp event.

2.3.2.18 AppeonPrint2File function

Description

Saves the specified DataWindow as image files of BMP, JPG or GIF format.

Syntax

AppeonPrint2File (datawindow adw, string asoutpath, string asoutname, long alouttype)

Table 2.37:

Argument	**Description**
adw	The DataWindow object to be saved as image files
asoutpath	The path of image files
asoutname	The specified name of image files
alouttype	The format type of files: 1-BMP; 2-JPG; 3-GIF

Return value

Table 2.38:

Integer.	Returns 1 if it succeeds in saving the specified DataWindow as image files. Returns -1 if an unknown error occurs. Returns -2 if alouttype is an unsupported image format. Returns -3 if adw is an invalid DataWindow object, DataStore object or DataWindowChild object. Returns -4 if it fails in creating a file, e.g., the specified path does not exist or without access rights. Returns -5 if it fails in creating device context, e.g., user sets a large size when customizing page property. Returns -6~-12 if an internal error occurs.

Usage

1. This function is used to execute saving DataWindow as image files.

2. If the page size is large enough, a DataWindow is saved as one file; if the size of a DataWindow surpasses the page size, the DataWindow is saved as several files.

3. The asoutname is the file name specified by user, for example, the function is appeonprint2file(adw, “c:\”, “appeon”, 1), if a DataWindow is saved as one file, the file is like C:\appeon.bmp; if a DataWindow is saved as several files, the files are like C:\appeon1.bmp, C:\appeon2.bmp, ..., C:\appeonN.bmp. Another example, the function is appeonprint2file(adw, “c:\”, “test.bmp”, 1), if a DataWindow is saved as one file, the file is like C:\ test.bmp.bmp; if a DataWindow is saved as several files, the files are like C:\test.bmp1.bmp, C:\test.bmp2.bmp, ..., C:\test.bmpN.bmp.

4. The upper limit of page size is restrained by the type of operating system. For instance, in Windows Vista, the upper limit of customized page size is 5500*5500 around. However, the page size can also be 1024*10000 by reducing page width and increasing page height.

2.3.2.19 AppeonPrint2PDF function (Obsolete)

Obsolete function

AppeonPrint2PDF is an obsolete function and will be discontinued in a future release. Please replace it with the SaveAs function of the DataWindow or Child DataWindow and set the saveas file type to PDF format.

2.3.2.20 AppeonSwitchRealTimeCalc function

Description

Performs the DataWindow real-time expression calculation in time or performs the calculation for only 1 time in the whole life-cycle.

Syntax

AppeonSwitchRealTimeCalc (powerobject adw, integer para)

Table 2.39:

Argument	Description
adw	The DataWindow/DataStore/DataWindowChild object.

Return value

Table 2.40:

Integer.	Returns 0 (default value) if need to perform real-time calculation. Returns 1 if no need to perform real-time calculation.

2.3.3 File Upload and Download

Appeon provides a non-visual object, AppeonFileService object, with five functions for uploading files to and downloading files from the file server. This is a web-based solution, it does not work in your client/server application.

You should follow the steps below to implement File Upload and Download on the Web:

Step 1: Configuring and deploying the file server

Step 2: Uploading and Downloading files

2.3.3.1 Configuring and deploying Appeon File Server

Appeon File Server is a standard non-visual Web application running on the back-end for uploading and downloading files. For Appeon Server installed to the .NET IIS, the setup wizard of Appeon File Server (setup.exe) can be found under %IIS_Web_Root%\appeon \plugin\fileservice. For Appeon Server installed to the Java server, the WAR package of Appeon File Server (fileservice.war) can be found under %Appeon_Server%\plugin \fileservice.

You must configure and deploy the file server first in order to use the AppeonFileService object in the Appeon Workarounds PBL. In most cases Appeon File Server is deployed to the machine where the application server is installed. It can also be deployed separately. For example, when the file size is too big, uploading or downloading it will negatively impact the performance of the application server, you can deploy the file server separately to another machine.

In the following section, you will be guided to configure and deploy Appeon File Server to IIS (.NET Framework), WebLogic, WebSphere, JBoss, JEUS, EAServer and NetWeaver. For more about the deploy instructions, refer to related documents of IIS, WebLogic, WebSphere, JBoss, JEUS, EAServer, and NetWeaver.

2.3.3.1.1 Deploying Appeon File Server to IIS

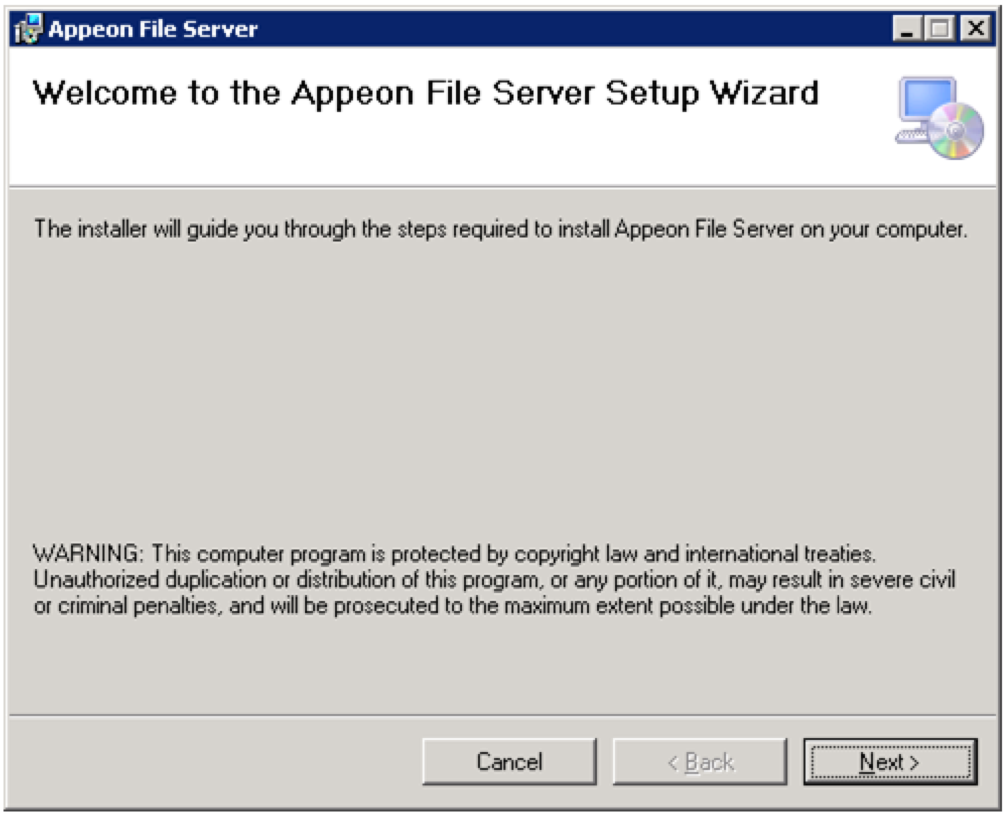
Installing the Appeon File Server

You will need to install first and then configure the Appeon File Server. For installing the file server to IIS, Appeon provides a **setup.exe** file at %IIS_Web_Root%\appeon\plugin \fileservice.

Step 1: Double click the **setup.exe** file to start the setup.

If the "Installation Incomplete" error displays, please try the solution in the "*Appeon File Server Installation Incomplete*" section in the *Appeon Troubleshooting Guide*.

Figure 2.1: Appeon File Service Setup Wizard



Appeon File Server

Welcome to the Appeon File Server Setup Wizard

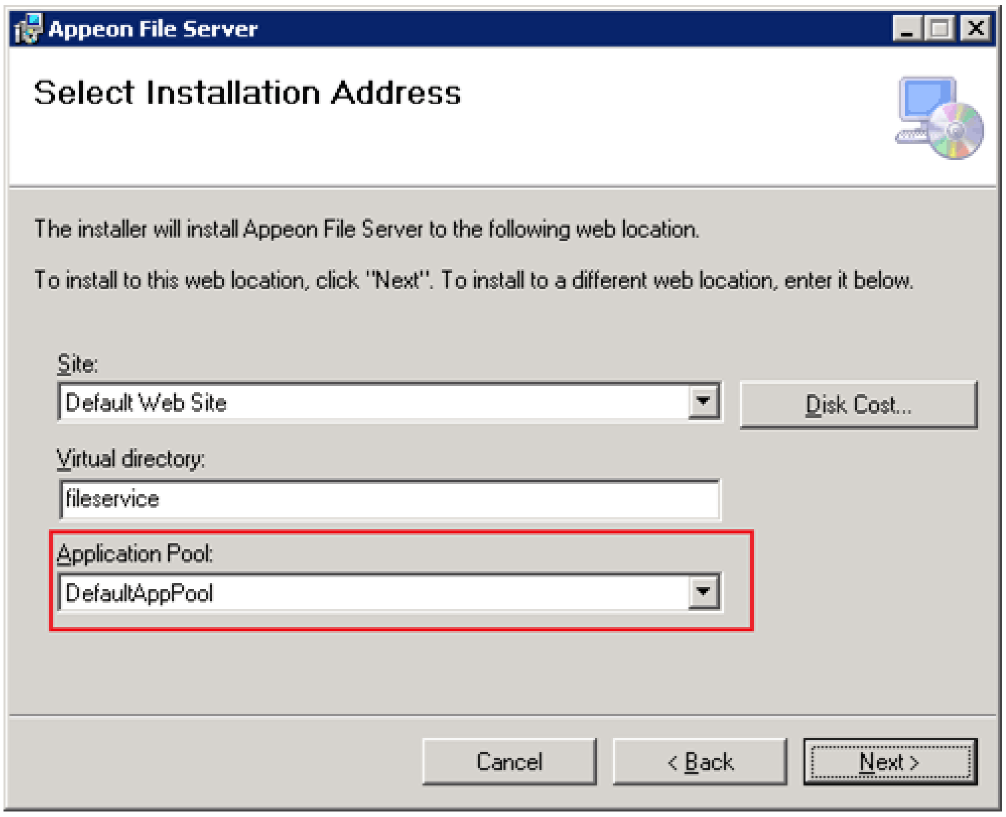
The installer will guide you through the steps required to install Appeon File Server on your computer.

WARNING: This computer program is protected by copyright law and international treaties. Unauthorized duplication or distribution of this program, or any portion of it, may result in severe civil or criminal penalties, and will be prosecuted to the maximum extent possible under the law.

Step 2: Click the **Next** button and then the following window pops up. Select a Web site where the Appeon File Server will be installed.

Note that DO NOT change the name of Virtual directory otherwise Appeon File Server will fail to start.

Figure 2.2: Select Installation Address



The installer will install Appeon File Server to the following web location.

To install to this web location, click "Next". To install to a different web location, enter it below.

Site: Default Web Site

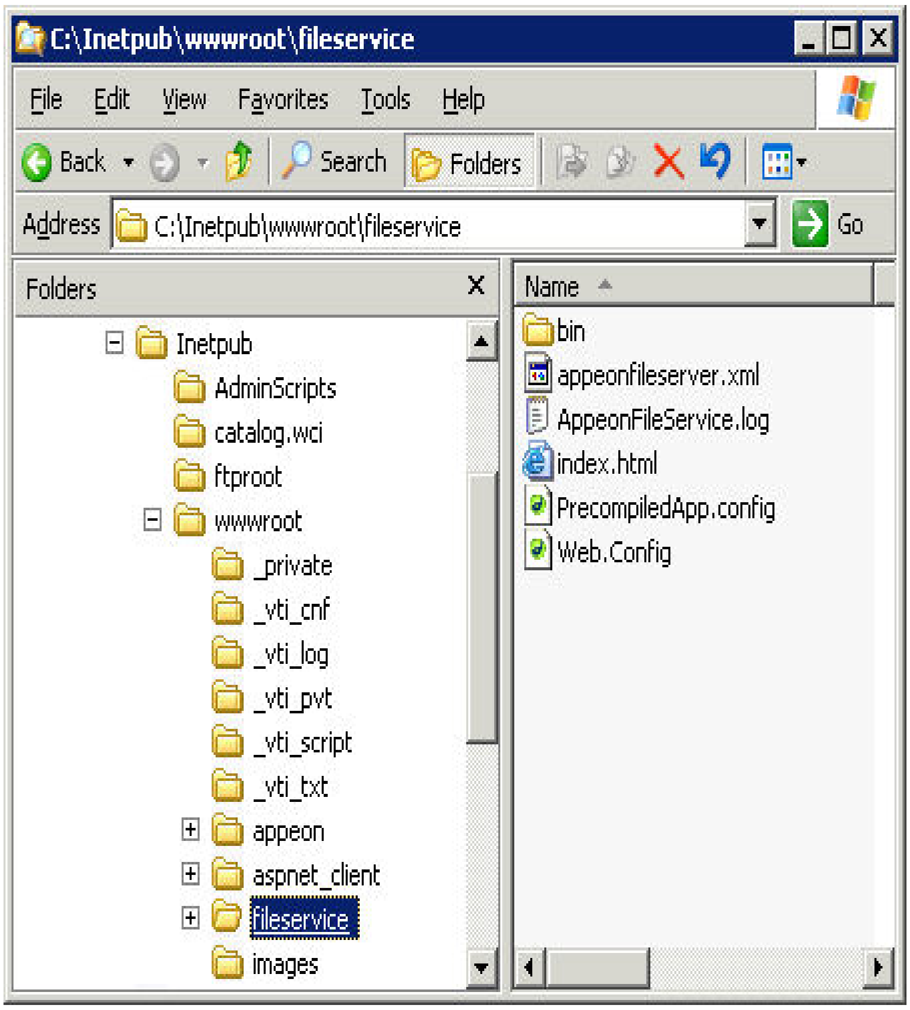
Virtual directory: fileservice

Application Pool: DefaultAppPool

Step 3: Click **Next** until the installation is complete.

Step 4: Go to directory where the Appeon File Server is installed, for example (C:\inetpub \wwwroot\fileservice).

Figure 2.3: Appeon File Server directory

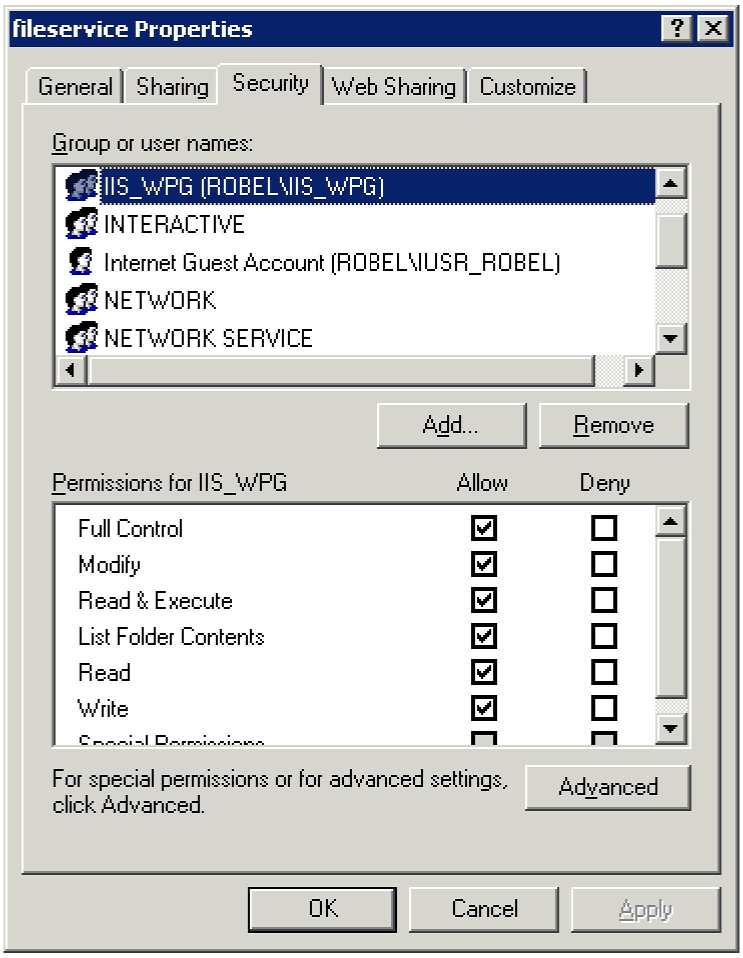


Step 5: Select **Properties** from the right-click menu of the **fileservice** folder.

Step 6: In the **Security** tab page of the **fileservice Properties** window,

For **IIS 6.0**, grant the **IIS_WPG** user with **Full Control** permission to this folder.

Figure 2.4: Grant Full Control (IIS6)



fileservice Properties

General | Sharing | Security | Web Sharing | Customize

Group or user names:

- IIS_WPG (ROBEL\IIS_WPG)
- INTERACTIVE
- Internet Guest Account (ROBEL\IUSR_ROBEL)
- NETWORK
- NETWORK SERVICE

Add... Remove

Permissions for IIS_WPG	Allow	Deny
Full Control	☑	☐
Modify	☑	☐
Read & Execute	☑	☐
List Folder Contents	☑	☐
Read	☑	☐
Write	☑	☐

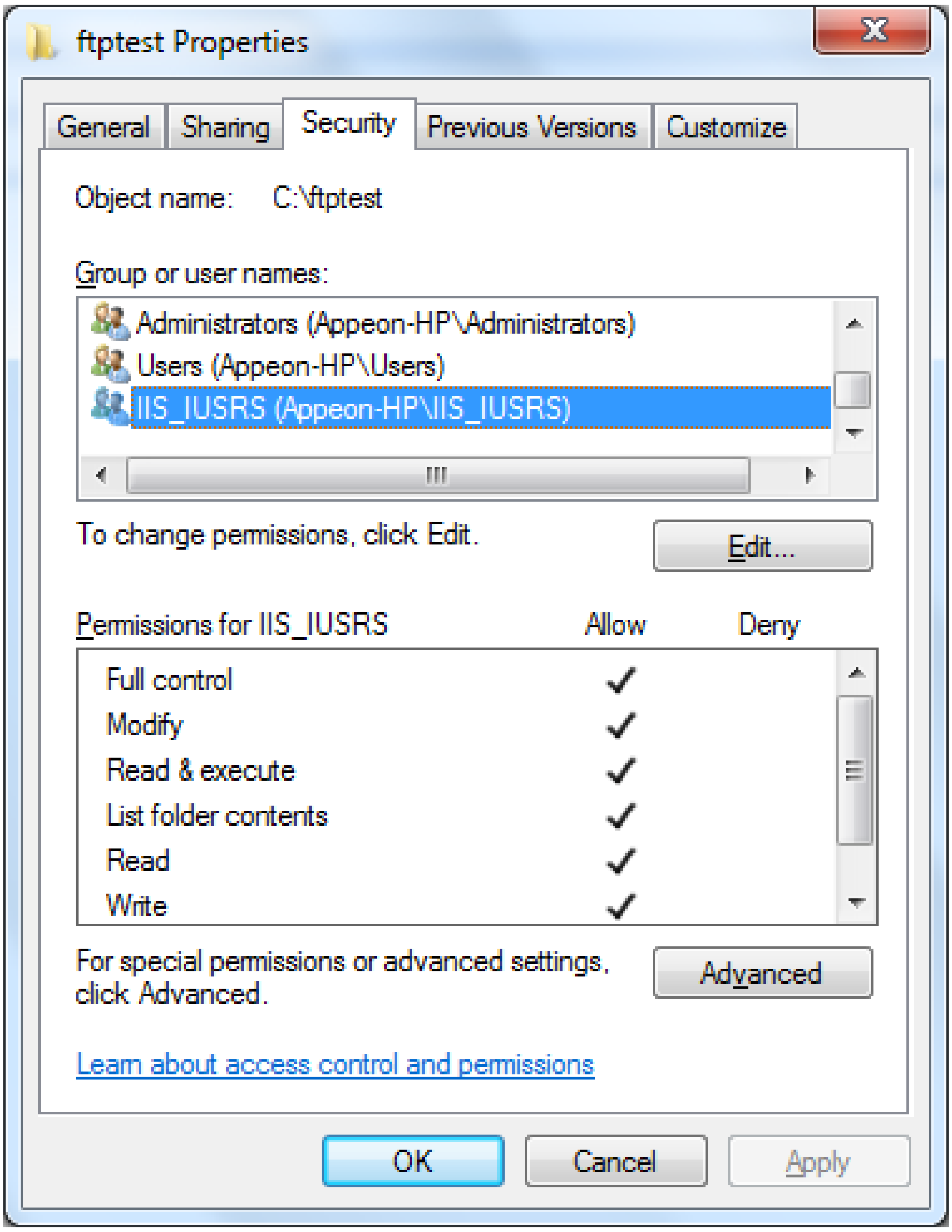
For special permissions or for advanced settings, click Advanced.

Advanced

OK Cancel Apply

For **IIS 7.0 or above**, grant the **IIS_IUSRS** user with **Full Control** permission to this folder.

Figure 2.5: Grant Full Control (IIS7)



Object name: C:\ftptest

Group or user names:

- Administrators (Appeon-HP\Administrators)
- Users (Appeon-HP\Users)
- IIS_IUSRS (Appeon-HP\IIS_IUSRS)

To change permissions, click Edit.

Permissions for IIS_IUSRS	Allow	Deny
Full control	✓	
Modify	✓	
Read & execute	✓	
List folder contents	✓	
Read	✓	
Write	✓	

For special permissions or advanced settings, click Advanced.

Uninstall Appeon File Server

Step 1: Go to **Start** | **Control Panel** | **Add and Remove Programs**.

Step 2: Select **Appeon File Server** and click **Remove** button.

Configuring the Appeon File Server

After deploying the Appeon File Server, follow steps below to configure it.

Open the appeonfileserver.xml in C:\Inetpub\wwwroot\fileservice.

```
<?xml version="1.0" encoding="UTF-8"?>
<webserver>
<!-- The value of attribute "value" must begin with either a single or double quote character. --
 <file-path value="D:\appeon\upload" />
 <log-level value="3" />          <!--0 Error,1 Info,2 Func, 3 Debug-->
 <session-timeout value="3600" /> <!-- the unit is second  -->
 <allowed-file-types value="txt;doc;jpg;mpeg" ignorecase="true" />
 <max-file-size value="20" />     <!-- the unit is M -->
 <users>
  <user name="test" password="password" />
  <user name="userA" password="userA" />
  <user name="userB" password="userB" />
 </users>
</webserver>
```

Modify the file according to your own demands.

Table 2.41: Settings specification value

Settings	The value of the setting specifies...
<file-path>	Specifies the directory to which all uploaded files are saved. Files are automatically saved under the application folder.
<log-level>	Specify the log level. Value: 0 - Error 1 - Information 2 - Function 3 - Debug
<session-timeout>	Specifies the time, in seconds, that passes after the last request is processed before the session times out.
<allow-file-type>	Specifies the file types that can be uploaded. Use a ";" between two file types. 1. "*" - Any file types are allowed. Note: Do not use "*" with other file types because "*" will not take any effect. For example, "*;text" means that only text file is allowed. 2. Using a "-" before the file types - Any file type is allowed, excluding the listed ones. For example, "-text;doc;jpg;mpeg" means that any file type is allowed, excluding text, doc, jpg and mpeg files. 3. "Ignorecase" - The file type is not case sensitive.
<max-file-size>	Specifies the maximum size (MB) of the file that can be uploaded.
<user>	Specifies the User Name and Password that can log on the file server. Multiple users are allowed as shown in the code example above.

2.3.3.1.2 Deploying Appeon File Server to WebLogic

Configuring the Appeon File Server

You will need to configure first and then deploy the Appeon File Server. Follow steps below to configure the Appeon File Server:

Step 1 - On the machine where the Appeon File Server will be deployed, create an XML file wherever you like and name it whatever you wish. In this example, create an XML file named "**appeonfileserver.xml**" under D:\appeon\config directory.

Step 2 - Copy the following code to the XML file.

```
<?xml version="1.0" encoding="UTF-8"?>
<fileserver>
 <file-path value="D:\appeon" />
 <session-timeout value="3600" />
 <allowed-file-types value="txt;doc;jpg;mpeg" ignorecase="true" />
 <max-file-size value="20" />
 <users>
  <user name="userA" password="userA" />
  <user name="userB" password="userB" />
 </users>
</fileserver>
```

Step 3 - Modify the settings in the XML file if necessary.

Table 2.42: The settings specification value

Settings	The value of the setting specifies...
<file-path>	Specifies the directory to which all uploaded files are saved. Files are automatically saved under the application folder.
<session-timeout>	Specifies the time, in seconds, that passes after the last request is processed before the session times out.
<allow-file-type>	Specifies the file types that can be uploaded. Use a ";" between two file types. 1. "*" - Any file types are allowed. Note: Do not use "*" with other file types because "*" will not take any effect. For example, "*;text" means that only text file is allowed. 2. "-" - Exclude the file types listed after "-". For example, "-text;doc;jpg;mpeg" means that any file type is allowed, except for text, doc, jpg and mpeg files. 3. "Ignorecase" - The file type is not case sensitive.
<max-file-size>	Specifies the maximum size (MB) of the file that can be uploaded.
<user>	Specifies the User Name and Password that can log on the file server. Multiple users are allowed as shown in the code example above.

Step 4 - On the machine where the Appeon Server is installed, unzip the f**ileservice.war** (%AppeonServer%\plugin\fileservice) and find the **web.xml** file under the **WEB-INF** folder. Then open **web.xml** with a text editor. Replace the **bold** text with the name and directory of the XML file created in the previous steps.

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE web-app PUBLIC '-//Sun Microsystems, Inc.//DTD Web Application 2.3//EN'
'http://java.sun.com/dtd/web-app_2_3.dtd'>
<web-app>

<servlet>
 <servlet-name>fileservice</servlet-name>
 <servlet-class>com.appeon.fileserver.WebServiceDispatcher</servlet-class>
 <init-param>
  <param-name>config</param-name>
  <param-value>D:\appeon\config\appeonfileserver.xml</param-value>
 </init-param>
 <load-on-startup>1</load-on-startup>
</servlet>
<servlet>
 <servlet-name>uploadfile</servlet-name>
 <servlet-class>com.appeon.fileserver.UploadFile</servlet-class>
 <init-param>
  <param-name>config</param-name>
  <param-value>D:\appeon\config\appeonfileserver.xml</param-value>
 </init-param>
 <load-on-startup>1</load-on-startup>
</servlet>

<servlet-mapping>
 <servlet-name>fileservice</servlet-name>
 <url-pattern>/fileservice</url-pattern>
</servlet-mapping>

</web-app>
```

Step 5 - Save the file and re-compress the **fileservice** folder to **fileservice.war** using WinZip, WinRAR or JDK. Do not use the other compression tools such as 7-zip.

Then follow the next section Deploying the Appeon File Server to deploy fileservice.war.

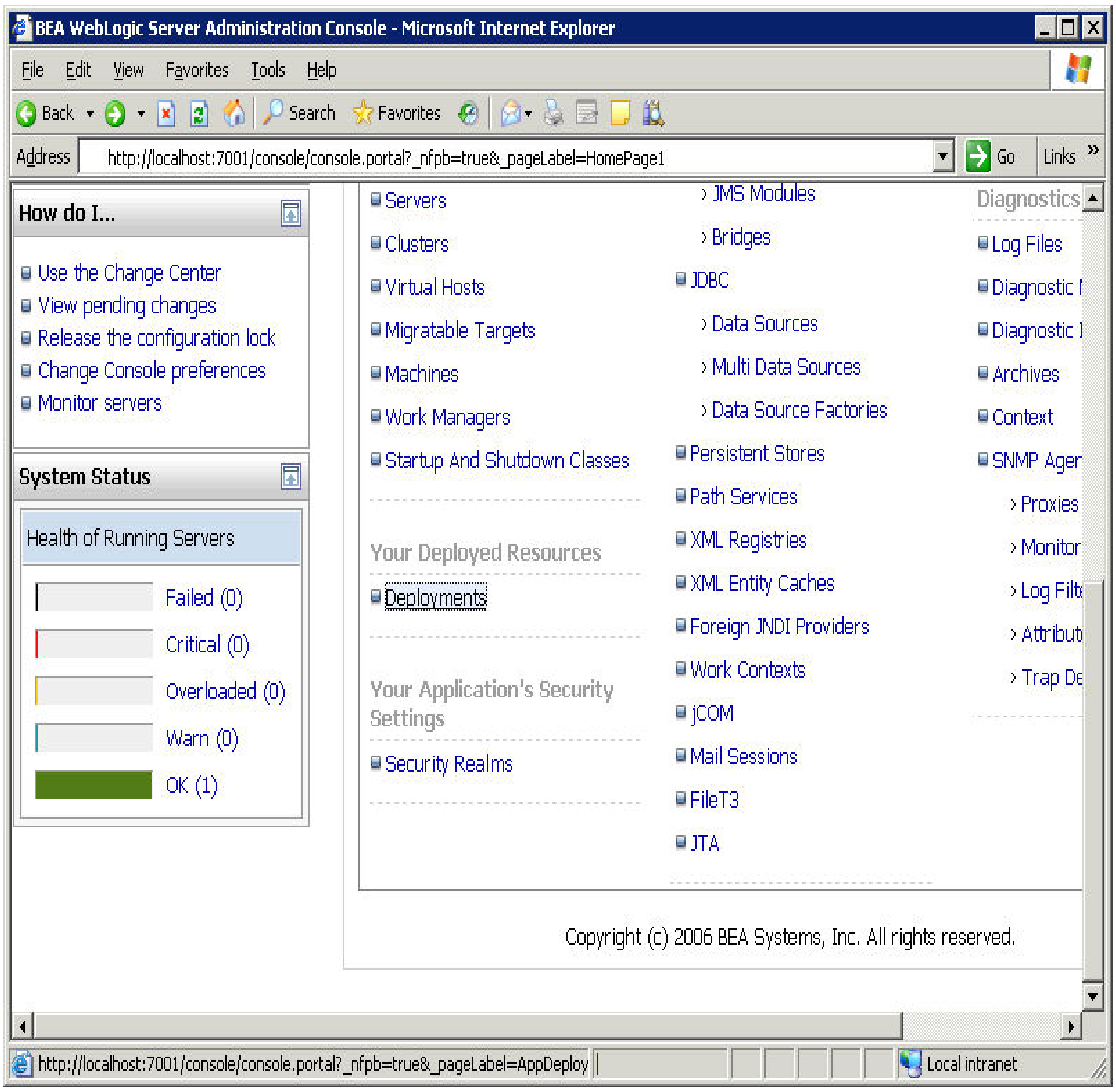
Deploying the Appeon File Server

After configuring the Appeon File Server, follow steps below to deploy it:

Step 1 - On the machine where Appeon Server is installed, access the **WebLogic Server Administration Console** in a Web browser.

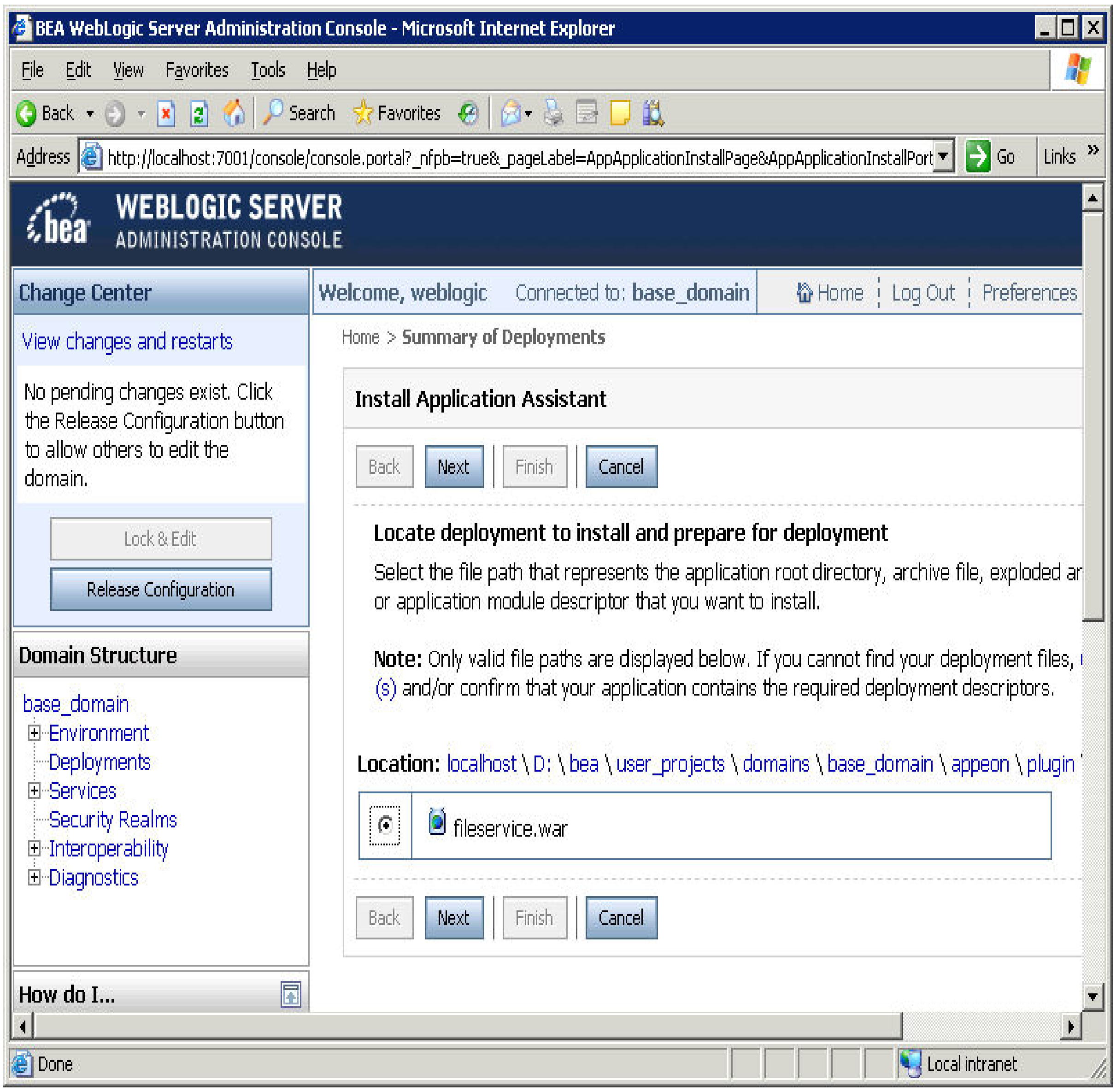
Step 2 - Click the **Deployment** link.

Figure 2.6:



Step 3 - Click the **Install** button and select the **fileservice.war** file at %AppeonServer%\plugin\fileservice.

Figure 2.7:


BEA WebLogic Server Administration Console - Microsoft Internet Explorer

Address http://localhost:7001/console/console.portal?_nfpb=true&_pageLabel=AppApplicationInstallPage&AppApplicationInstallPort

Welcome, weblogic Connected to: base_domain

Home > Summary of Deployments

Install Application Assistant

Locate deployment to install and prepare for deployment

Select the file path that represents the application root directory, archive file, exploded ar
or application module descriptor that you want to install.

Note: Only valid file paths are displayed below. If you cannot find your deployment files,
(s) and/or confirm that your application contains the required deployment descriptors.

Location: localhost \ D: \ bea \ user_projects \ domains \ base_domain \ appeon \ plugin

● fileservice.war

Change Center

View changes and restarts

No pending changes exist. Click the Release Configuration button to allow others to edit the domain.

Step 4 - Click the **Next** button to use the default settings and then click the **Finish** button to finish the deployment of Appeon File Server.

2.3.3.1.3 Deploying Appeon File Server to WebSphere

Configuring the Appeon File Server

Configuring the Appeon File Server in the Java server such as WebLogic, WebSphere, JBoss, JEUS, NetWeaver etc. is the same. Please refer to Configuring the Appeon File Server for WebLogic.

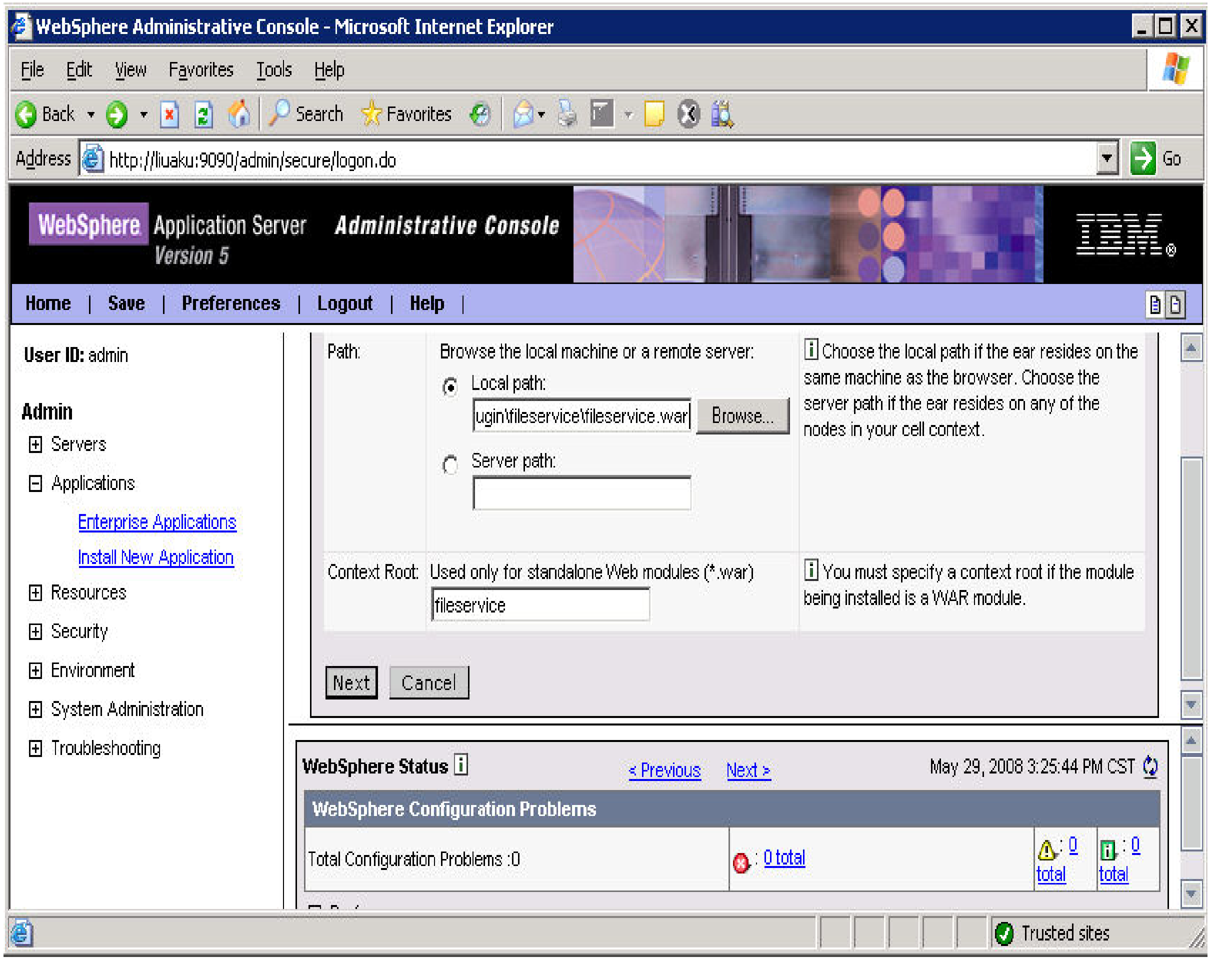
Deploying the Appeon File Server

After configuring the Appeon File Server, follow steps below to deploy it:

Step 1 - On the machine where Appeon Server is installed, access the **WebSphere Administrative Console** in a Web browser.

Step 2 - Select **Applications** | **Install New Applications** from the left tree view. And then in the right page specify the path where the **fileservice.war** is (by default at %AppeonServer%\plugin\fileservice) and **Context Root** for the WAR file and click the **Next** button.

Figure 2.8:


WebSphere Administrative Console - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Search Favorites
Address http://liuaku:9090/admin/secure/logon.do Go
WebSphere Application Server Version 5 Administrative Console
Home | Save | Preferences | Logout | Help |
User ID: admin
Admin
Servers
Applications
Enterprise Applications
Install New Application
Resources
Security
Environment
System Administration
Troubleshooting
Next Cancel
< Previous Next >
Trusted sites

Path: Browse the local machine or a remote server:
Local path: ugin\fileservice\fileservice.war Browse...
Server path:
Choose the local path if the ear resides on the same machine as the browser. Choose the server path if the ear resides on any of the nodes in your cell context.

Context Root: Used only for standalone Web modules (*.war)
fileservice
You must specify a context root if the module being installed is a WAR module.

WebSphere Status May 29, 2008 3:25:44 PM CST

WebSphere Configuration Problems

Total Configuration Problems :0 | 0 total | 0 total | 0 total

Step 3: Click **Next** with default settings until the file server is successfully installed on the WebSphere server.

2.3.3.1.4 Deploying Appeon File Server to JBoss

Configuring the Appeon File Server

Configuring the Appeon File Server in the Java server such as WebLogic, WebSphere, JBoss, JEUS, NetWeaver etc. is the same. Please refer to Configuring the Appeon File Server for WebLogic.

Deploying the Appeon File Server

After configuring the Appeon File Server, follow steps below to deploy it. There are two methods to deploy the Appeon File Server in JBoss:

- Automatic deploy

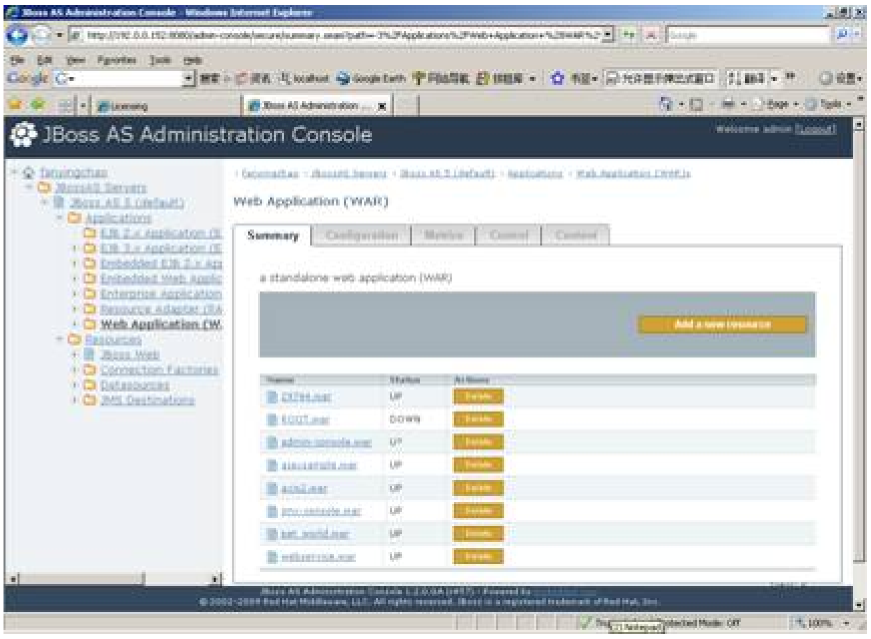
  Copy the file **fileservice.war** to the directory %JBoss installation root directory%\server\<instance>\deploy and then start JBoss.

The fileservice.war file will be deployed automatically.

- Manual deploy

1. Start JBoss and log in to JBoss console.
2. Select **Web Application** in the left tree view and then click **Add a new resource** in the right page.

Figure 2.9: PDF



JBoss AS Administration Console

Web Application (WAR)

Summary | Configuration | Metrics | Control | Content

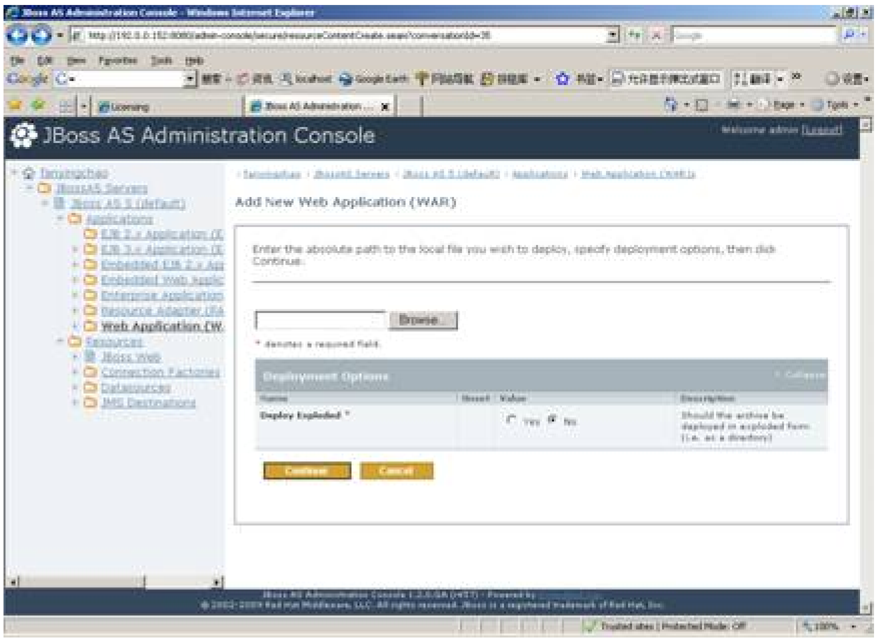
a standalone web application (WAR)

Add a new resource

Name	Status	Actions
[illegible]	UP	Delete
[illegible]	DOWN	Delete
admin-console.war	UP	Delete
[illegible]	UP	Delete
[illegible]	UP	Delete
jmx-console.war	UP	Delete
[illegible]	UP	Delete
[illegible]	UP	Delete

3. Click **Browse** and locate the file **fileservice.war**.

Figure 2.10:



JBoss AS Administration Console

Add New Web Application (WAR)

Enter the absolute path to the local file you wish to deploy, specify deployment options, then click Continue.

Browse...

* denotes a required field.

Deployment Options

Name	Unset	Value	Description
Deploy Exploded *		Yes [✓ No]	Should the archive be deployed in exploded form (i.e. as a directory)

Continue | Cancel

4. Click **Continue** with default settings until the file server is successfully deployed in the JBoss server.

2.3.3.1.5 Deploying Appeon File Server to JEUS

Configuring the Appeon File Server

Configuring the Appeon File Server in the Java server such as WebLogic, WebSphere, JBoss, JEUS, NetWeaver etc. is the same. Please refer to Configuring the Appeon File Server for WebLogic.

Deploying the Appeon File Server

After configuring the Appeon File Server, follow steps below to deploy it:

Copy the **fileservice.war** file to the directory %JEUS installation root directory%\webhome\autodeploy and then start JEUS server. The fileservice.war file will be deployed automatically.

2.3.3.1.6 Deploying Appeon File Server to EAServer

Configuring the Appeon File Server

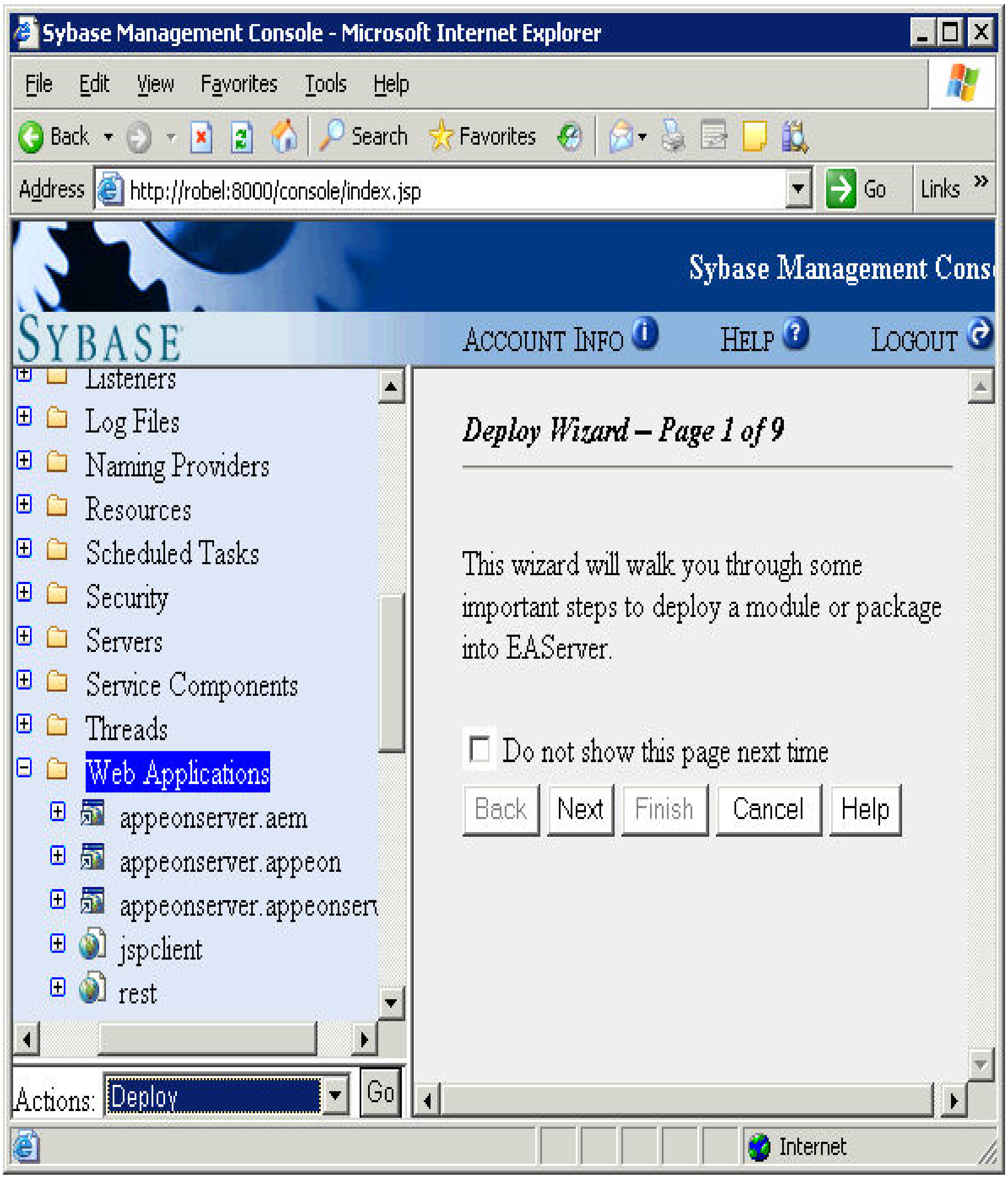
Configuring the Appeon File Server in the Java server such as WebLogic, WebSphere, JBoss, JEUS, NetWeaver etc. is the same. Please refer to Configuring the Appeon File Server for WebLogic.

Deploying the Appeon File Server

After configuring the Appeon File Server, follow steps below to deploy it:

Step 1: Open **Sybase Management Console**, click **Web Applications** and select **Deploy** from the **Actions** list box and click the **Go** button.

Figure 2.11:



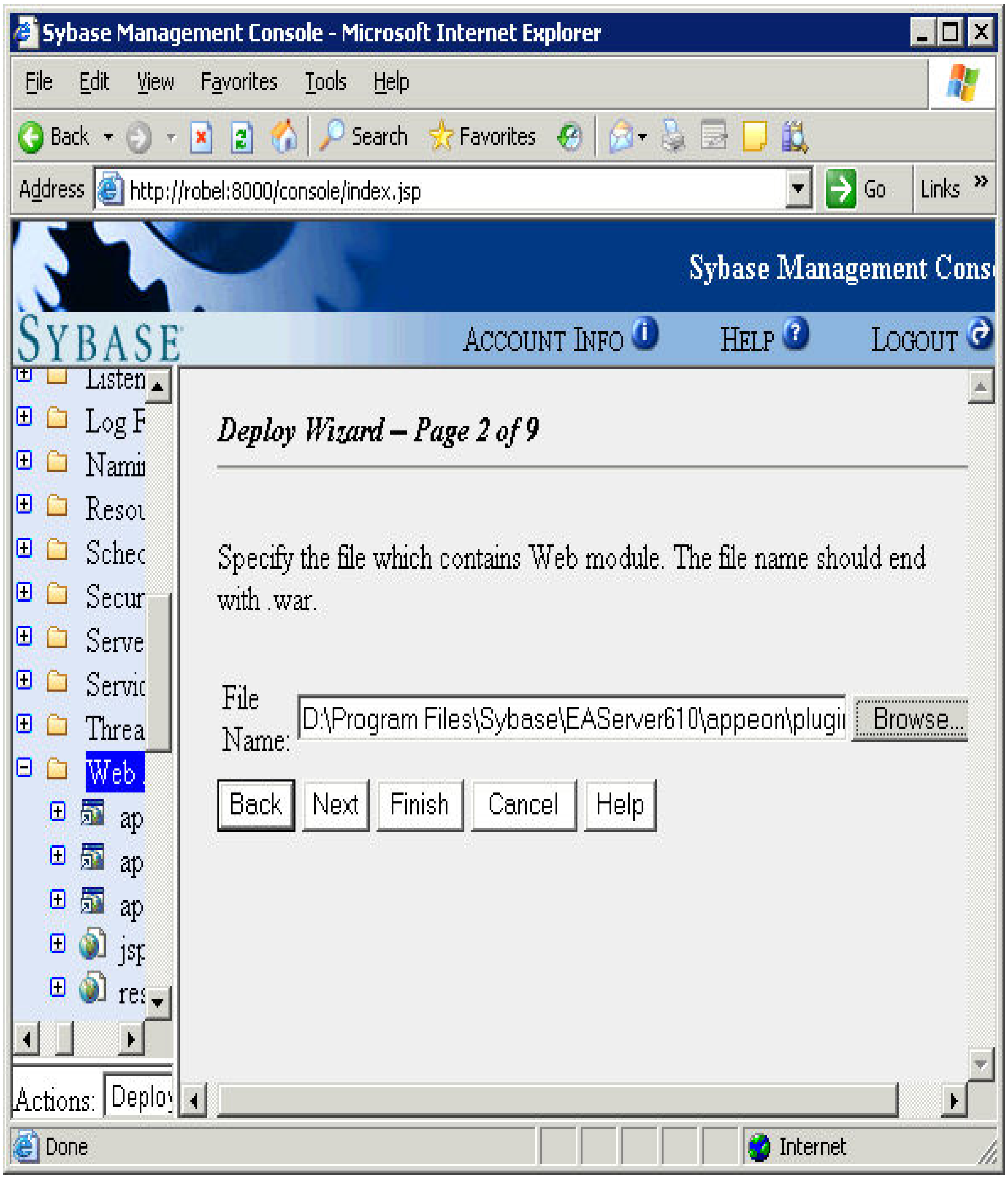
Deploy Wizard – Page 1 of 9

This wizard will walk you through some important steps to deploy a module or package into EAServer.

☐ Do not show this page next time

Step 2: Click ***Browse*** to choose the **fileservice.war** file and then click **Next**.

Figure 2.12:



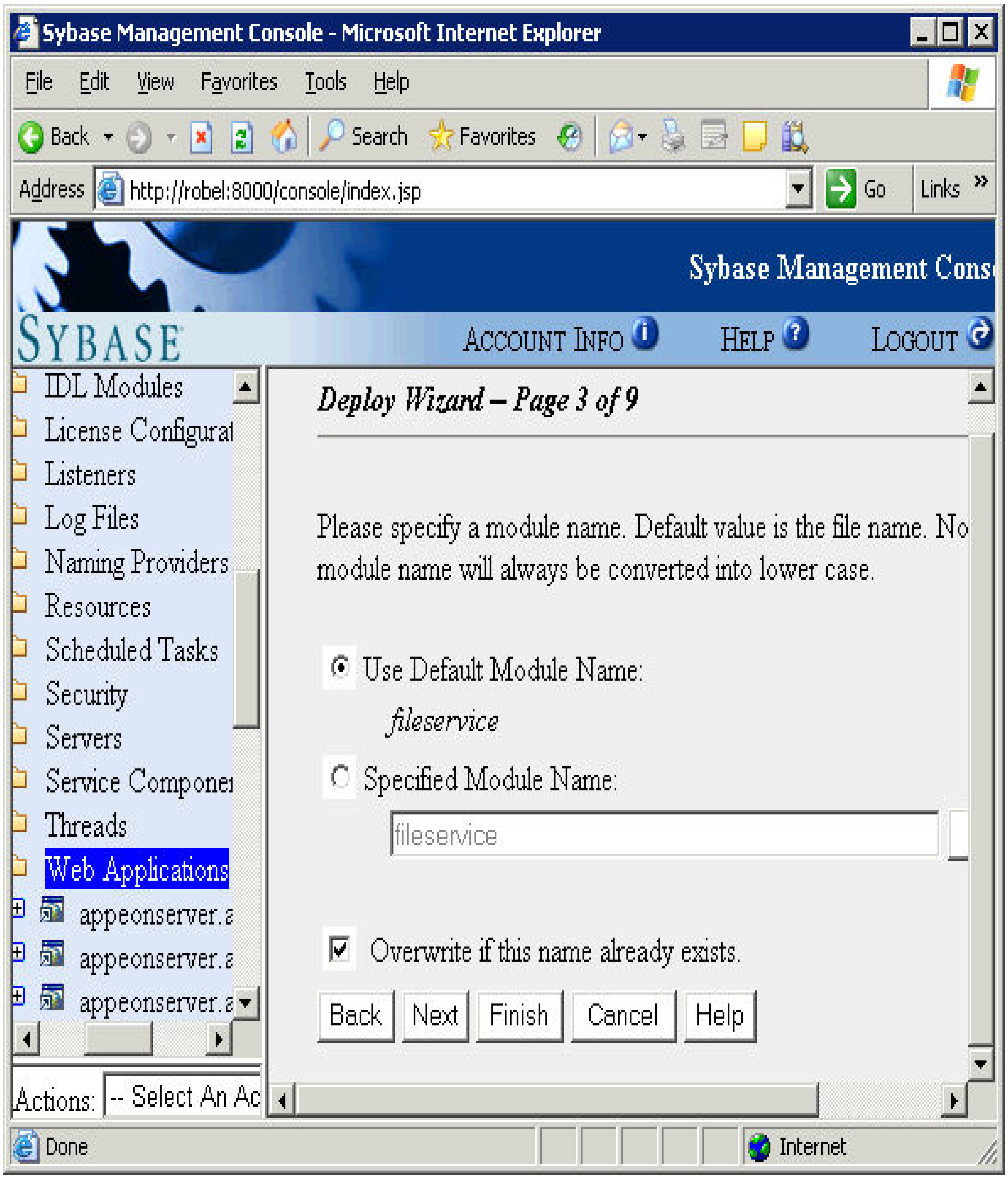
Deploy Wizard – Page 2 of 9

Specify the file which contains Web module. The file name should end with .war.

File Name: D:\Program Files\Sybase\EAServer610\appeon\plugi

Step 3: Choose **Use Default Module Name** and click **Next** to continue the deployment.

Figure 2.13:



Deploy Wizard – Page 3 of 9

Please specify a module name. Default value is the file name. No
module name will always be converted into lower case.

(●) Use Default Module Name:

fileservice

( ) Specified Module Name:

fileservice

☑ Overwrite if this name already exists.

Step 4: Click the **Finish** button to complete the deployment of Appeon File Server.

2.3.3.1.7 Deploying Appeon File Server to NetWeaver

Configuring the Appeon File Server

Configuring the Appeon File Server in the Java server such as WebLogic, WebSphere, JBoss, JEUS, NetWeaver etc. is the same. Please refer to Configuring the Appeon File Server for WebLogic.

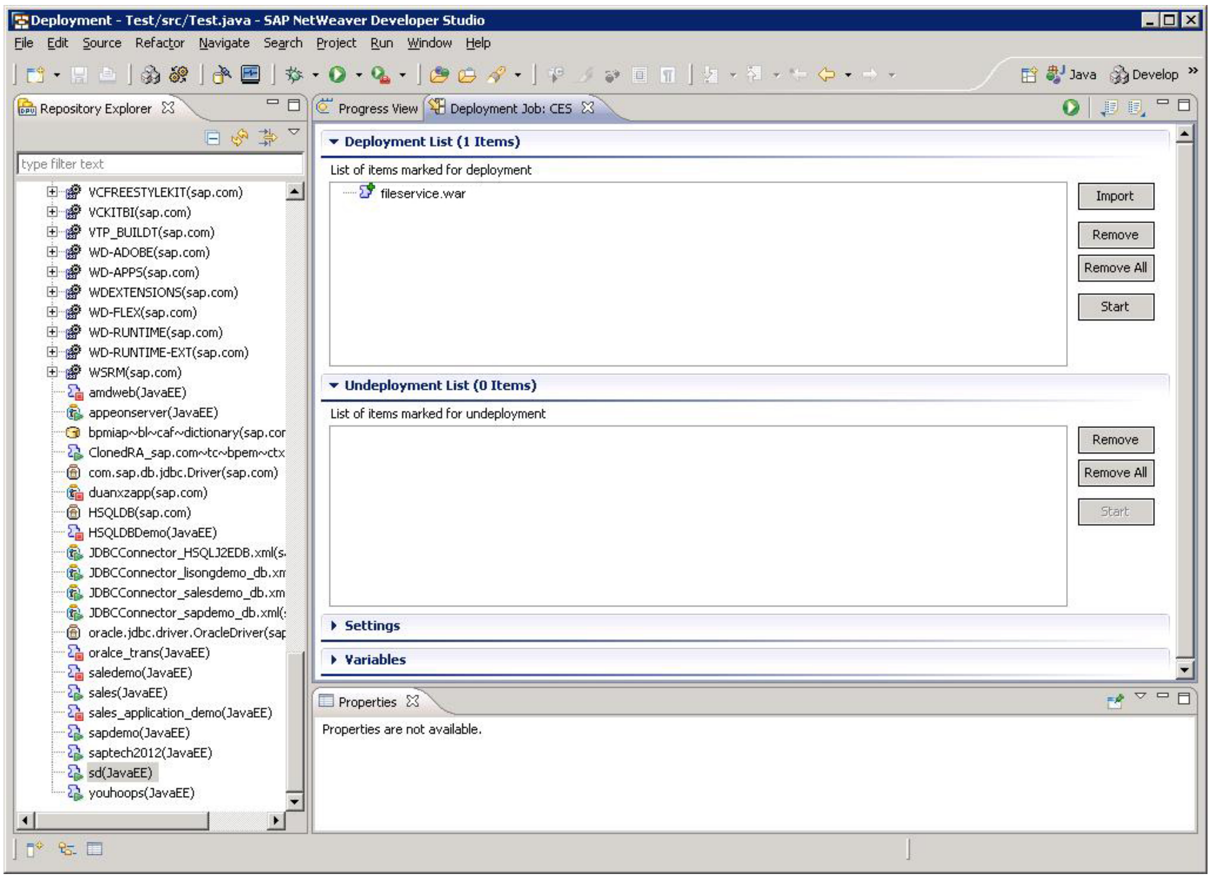
Deploying the Appeon File Server

After configuring the Appeon File Server, follow steps below to deploy it:

Step 1 - Open the deployment perspective in the SAP NetWeaver Developer Studio by clicking the menu **Window** | **Open Perspective** | **Other** and then selecting **deployment** in the pop-up window. The deployment perspective is shown as below.

Step 2 - Click the **Import** button to import the **fileservice.war** file and then click **Start** to deploy the Appeon File Server after the file is imported successfully.

Figure 2.14:


Deployment - Test/src/Test.java - SAP NetWeaver Developer Studio
File Edit Source Refactor Navigate Search Project Run Window Help
Java Develop
Repository Explorer
type filter text
VCFREESTYLEKIT(sap.com)
VCKITBI(sap.com)
VTP_BUILDT(sap.com)
WD-ADOBE(sap.com)
WD-APPS(sap.com)
WDEXTENSIONS(sap.com)
WD-FLEX(sap.com)
WD-RUNTIME(sap.com)
WD-RUNTIME-EXT(sap.com)
WSRM(sap.com)
amdweb(JavaEE)
appeonserver(JavaEE)
bpmiap~bl~caf~dictionary(sap.cor
ClonedRA_sap.com~tc~bpem~ctx
com.sap.db.jdbc.Driver(sap.com)
duanxzapp(sap.com)
HSQLDB(sap.com)
HSQLDBDemo(JavaEE)
JDBCConnector_HSQLJ2EDB.xml(s
JDBCConnector_lisongdemo_db.xm
JDBCConnector_salesdemo_db.xm
JDBCConnector_sapdemo_db.xml(
oracle.jdbc.driver.OracleDriver(sap
oralce_trans(JavaEE)
saledemo(JavaEE)
sales(JavaEE)
sales_application_demo(JavaEE)
sapdemo(JavaEE)
saptech2012(JavaEE)
sd(JavaEE)
youhoops(JavaEE)
Progress View
Deployment Job: CES
Deployment List (1 Items)
List of items marked for deployment
fileservice.war
Import
Remove
Remove All
Start
Undeployment List (0 Items)
List of items marked for undeployment
Remove
Remove All
Start
Settings
Variables
Properties
Properties are not available.

2.3.3.2 Uploading and downloading files

After configuring and deploying the Appeon File Server, you can follow the steps below to upload files to or download files from the Appeon File Server.

Step 1 - Add Appeon Workaround PBL to your PowerBuilder project.

Step 2 - Define a variable and create an instance for the **AppeonFileService** object.

Step 3 - Call of_logonfileserver to connect to the file server.

Step 4 - Call of_appeonupload or of_appeondownload to upload or download files.

Step 5 - Call of_logofffileserver to disconnect from the file server.

2.3.3.2.1 AppeonFileService object

AppeonFileService object provides service about the file uploading and downloading.

Functions for AppeonFileService object

AppeonFileService object provides the following functions to upload and download files:

- of_logonfileserver function: Connects to the file server.
- of_logofffileserver function: Disconnects from the file server.
- of_appeonupload function: Uploads file to the file server. You can specify the path where the source file locates and the path where the file will be uploaded.
- of_appeondownload function: Downloads the specified file from the file server. You can specify the path from which the file is downloaded and the path to which the file will be saved.
- of_FileExists function: Reports whether the specified file exists on the file server.

of_logonfileserver

Description

Connects to the Appeon File Server.

Syntax

appeonfileservice.of_logonfileserver (string fileserveripaddress, long port, string connectstring)

Table 2.43: Syntax

Parameter	Description
appeonfileservice	An instance of an AppeonFileService object
fileserveripaddress	IP address or machine name of the Appeon File Server.
port	HTTP Port of the application where the Appeon File Server is deployed. HTTPS port is unsupported.
connectstring	User name and password for connecting to the Appeon File Server. Format: "username=username;password=password". Use a semicolon ";" to separate the user name and password. The user name and password should be consistent with the specification in the XML.

Return value

Table 2.44: Return value

Long.	Return values are: 0 - The logon is successful. -1 - Connection to the file server fails. -2 - The provided user name or password is invalid.

of_logofffileserver

Description

Disconnects from the Appeon File Server.

Syntax

appeonfileservice.of_logofffileserver()

Table 2.45:

Parameter	Description
appeonfileservice	An instance of an AppeonFileService object

Return value

Table 2.46:

Long.	Return values are: 0 - The logoff is successful. -1 - Connection to the Appeon File Server fails. -2 - The session has timed out or the user has not logged into the file server.

of_appeonupload

Description

Uploads file to the Appeon File Server.

Syntax

appeonfileservice.of_appeonupload (string source, string destination, boolean isrename, ref long errorcode)

appeonfileservice.of_appeonupload (string destination, boolean isrename, ref long errorcode)

appeonfileservice.of_appeonupload (boolean isrename, ref long errorcode)

Table 2.47:

Parameter	Description
appeonfileservice	An instance of an AppeonFileService object.
source	(Optional) The directory of the source file. If no directory is specified here, a dialog box will be displayed prompting the user to select a file to upload.
destination	(Optional) The destination directory where the file is uploaded. The directory specified here is a relative path to the upload file directory on the file server.
isrename	Whether to rename the file if another file with the same name already exists.
errorcode	The error occurred in uploading files. Values are: 0 - Succeeded. -1 - Connection to the file server fails. -2 - ID error. -3 - The format of the source file is forbidden.

	-4 - Destination directory is invalid or the file is too large.

Return value

Table 2.48:

String.	Returns the URL of the uploaded file if it succeeds and returns an empty string if it fails. Returns NULL if any argument is NULL.

Usage

If you want to delete an uploaded file, you must restart the file server after deleting it. Otherwise, you will fail to upload the file with the same name.

of_appeondownload

Description

Downloads the specified file from the Appeon File Server.

Syntax

appeonfileservice.of_appeondownload (string source)

appeonfileservice.of_appeondownload (string source, string target)

Table 2.49:

Parameter	Description
appeonfileservice	An instance of an AppeonFileService object.
source	The directory of the source file. The directory specified here is a relative path to the download file directory on the Appeon File Server.
target	(Optional) The destination directory where the file is saved. If no directory is specified here, a dialog box will be displayed prompting the user to select a location to save the file.

Return value

Table 2.50:

Long.	Return values are: 0 - The file is successfully downloaded. -1 - The file download failed or exceptions occurred. -3 - File name is empty. NULL if any argument is NULL.

of_FileExists

Description

Reports whether the specified file exists on the Appeon File Server.

Syntax

appeonfileservice.of_FileExists (String filename)

Table 2.51:

Parameter	Description
appeonfileservice	An instance of the AppeonFileService object.
filename	A string whose value is the name of a file.

Return value

Long.

1 - The file exists.

0 - The file does not exist on the Appeon File Server.

-1 - Failed to connect to the Appeon File Server.

of_downloadfile (Obsolete)

Obsolete function

of_downloadfile is an obsolete function and will be discontinued in a future release. Please replace it with of_appeondownload.

of_uploadfile (Obsolete)

Obsolete function

of_uploadfile is an obsolete function and will be discontinued in a future release. Please replace it with of_appeonupload.

Properties for AppeonFileService object

AppeonFileService object provides the following properties (private instances) to upload and download files:

- is_FileServerName instance: Indicates the name of the file server.
- is_LogOnParams instance: Indicates the connection string for logging into the file server.
- il_Id instance: Long. Indicates the ID number returned from the file server when logon has succeeded.

is_FileServerName

Description

Indicates the IP address or machine name of the file server. This property is initialized after the of_LogOnFileServer function is called.

Return value

String.

is_LogOnParams

Description

Indicates the connection string for logging into the file server. This property is initialized after the of_LogOnFileServer function is called.

Return value

String.

il_Id

Description

Indicates the ID number returned from the file server when logon has succeeded. This property is initialized after the of_LogOnFileServer function is called. It is used by the file server to validate the client.

Return value

Long.

2.3.4 Distributed DataWindows (EAServer only)

"Distributed DataWindows" refers to the use of DataWindow/DataStore objects in a distributed environment. In a distributed PowerBuilder application, a DataWindow control at the Client can associate with a DataStore object in EAServer. The Client DataWindow control is responsible for the visual representation of data and user operations, while the DataStore object in EAServer is responsible for transactions. The state of the Client DataWindow control is synchronized with the state of the DataStore object in EAServer and vice versa, using relevant DataWindow functions.

There are two benefits to using distributed DataWindow technology with Appeon:

- Provides more scalability by separating user interface and business logic.
- Works around the Appeon-unsupported DataWindow functions by moving the functions to the server DataStore objects.

2.3.4.1 AppeonDataWindow and AppeonDataStore

AppeonDataWindow and AppeonDataStore are two standard user objects provided by Appeon. The appeondatawindow is inherited from the PowerBuilder system DataWindow control, and the appeondatastore is inherited from the PowerBuilder system DataStore object.

Why is workaround required if you use distributing DataWindows in Appeon

PowerBuilder GetFullState, SetFullState, GetChanges and SetChanges functions use BLOB (Binary Large Object) parameters for passing DataWindow or DataStore object specifications.Although Appeon supports BLOB, but it cannot directly interpret the BLOB DataWindow or DataStore object specifications. To work around the unsupported features (the BLOB parameter) in GetFullState, SetFullState, GetChanges and SetChanges, you should use *appeondatawindow* and *appeondatastore*.

Functions for AppeonDataWindow and AppeonDataStore

There are six functions provided by AppeonDataWindow and AppeonDataStore.

1. GetFullState, SetFullState, GetChanges and SetChanges functions are derived from corresponding PowerBuilder functions.
2. AppeonGetFullStateEX and AppeonSetFullStateEX functions.

2.3.4.1.1 AppeonGetFullStateEX

Description

Retrieves the complete state of the main DataWindow into a blob, excluding the information of its DropDownDataWindow.

Syntax

Long dwcontrol.AppeonGetFullStatusEX (blob dwasblob)

Table 2.52: Syntax

Argument	Description
dwcontrol	A reference to an appeondatawindow control and an appeondatastore object.
dwasblob	A variable into which the returned DataWindow will be placed.

Return value

Returns the number of rows in the DataWindow blob if it succeeds and returns -1 if an error occurs. If any argument value is NULL, the method returns NULL.

2.3.4.1.2 AppeonSetFullStateEX

Description

Applies the contents of a DataWindow blob retrieved by AppeonGetFullStateEX to a DataWindow. If the source DataWindow object matches the target DataWindow, DropDownDataWindow information of the target DataWindow will not be changed.

Syntax

Long dwcontrol.AppeonSetFullStatusEX (blob dwasblob)

Table 2.53: Syntax

Argument	Description
dwcontrol	An reference to an appeondatawindow control and an appeondatastore
dwasblob	A variable into which the returned DataWindow will be placed.

Return value

Long. Return value are:

1 - DataWindow objects match; old data and state overwritten.

2 - DataWindow objects do not match; old object, data, and state replaced.

3 - No DataWindow object associated with DataWindow control or DataStore; the DataWindow object associated with the blob is used. The value of the DataObject property remains an empty string.

2.3.4.2 Workaround steps

This section introduces the main workaround steps you should take for using distributed DataWindows in your application.

Step 1 - Add the Workaround PBL and DLLs provided by Appeon to your application.

Step 2 - Derive all distributed DataWindows and DataStores from appeondatawindow and appeondatastore.

appeondatastore and appeondatawindow are built in *appeon_workarounds.pbl*.

Step 4 - Migrate n-Tier DataWindows. In PowerBuilder, deploy the server DataWindows and DataStores that are inherited from appeondatastore to Appeon Server.

You also need to deploy appeondatawindow objects and appeondatastore objects to the Appeon Server.

Step 5 - Generate stubs and skeletons for the server DataWindows and DataStores as well as n-Tier NVOs in the application by following the instructions in How to deploy NVO to EAServer 6.1.

A detailed sample for the workaround

2.3.4.3 Workaround limitations

When using the *appeondatawindow* and *appeondatastore* objects to work around the distributed DataWindow technique, there are some limitations regarding the use of Appeon GetFullState, SetFullState, GetChanges, and SetChanges functions.

Table 2.54:

Limitation/ Difference in ...	Limitation/Difference Description
DataWindow styles	The workaround works with DataWindows or DataStores of all styles, except for OLE and Treeview.
Return value of the functions	The return value of Appeon SetFullState may have a different meaning from that of PowerBuilder system SetFullState function. The Appeon GetChanges function always returns -1 if it fails. In PowerBuilder, the function can return more error numbers (-1, -2 and -3). The Appeon SetChanges function can return -1 and -3, but cannot return 2 and -2.
DataWindow ImportString function	If using the DW ImportString function in a distributed DataWindow environment, keep the date display format the same at the Client and Appeon Server machines. In addition, the date/time format configuration in AEM should be kept the same as that of Appeon Server.
State information initialization of a DataWindow/ DataStore	In PowerBuilder, the state information of a DataWindow/DataStore is initialized whenever you set its DataObject property. However, if using *appeondatawindow* and *appeondatastore*, the state information is initialized only when you change the DataObject property to a different DataWindow object.
Truncation of characters in certain cases	When applying Appeon SetChanges to a target DataWindow/DataStore, if a column of Char type in the source DataWindow/DataStore has defined more characters than its corresponding column in the target DataWindow/DataStore, characters from the source column that exceed the length limit of the target column are truncated, but in PowerBuilder the extra characters are preserved.

Limitation/ Difference in ...	Limitation/Difference Description
Un-modified or modified data	When calling PowerBuilder GetFullState and GetChanges, changed (but not accepted) data in a DataWindow control is treated as un-modified data, but if using the Appeon *appeondatawindow* and *appeondatastore*, changed (but not accepted) data is treated as modified data.

2.3.5 Appeon Labels

Overview

Appeon Labels provided in the Appeon Workarounds PBL can reduce the interactions between the client and Appeon Server, thus boosting the performance of Appeon applications.

Applying the Appeon Labels

The Appeon Labels associated functions are contained in the appeon_nvo_db_update object in appeon_workaround.pbl. If you want to use these functions in your PowerBuilder application, add the library to the Library Search Path of the application. The appeon_workarounds.pbl is located in the \appeon_workarounds**xxx**e folder (**xxx** indicates the version number of PowerBuilder) under the Appeon Developer directory. (For example, C:\Program Files\Appeon\Developer2013\appeon_workarounds125e)

Appeon Labels and associated functions

Table 2.55:

Label Name	Label Associated Function	Description
Appeon Commit/ Rollback Label	of_autocommitrollback	Automatically commits or rolls back the first database operation statement after the label.
Appeon Commit Label	of_autocommit	Automatically commits the first database operation.
Appeon Rollback Label	of_autorollback	Automatically rolls back the first database operation statement if the operation fails.
Appeon Queue Labels	of_startqueue of_commitqueue	The Appeon Queue Labels are designed for use when it is necessary to execute numerous database operation statements on Appeon Server, and the returned values of the statements are not validated or used. Using the pair of labels can dramatically reduce the number of client-server interactions.
Appeon Immediate Call Label	of_imdcall	Immediately commits a database operation.
Appeon Update Label	of_update	Reduces the number of interactions with the server caused by "interrelated updates".

Usage

For detailed instructions on how to take advantages of Appeon Labels to improve performance of the application, please refer to the "*Technique #4: grouping multiple server calls with Appeon Labels*" section in the *Appeon Performance Tuning Guide*.

2.3.5.1 Appeon Commit/Rollback Label

Description

The Appeon Commit/Rollback Label is used to automatically commit or rollback the first database operation statement after the label.

Controls

appeon_nvo_db_update object

Associated functions

of_autocommitrollback

Syntax

objectname.of_autocommitrollback( )

Table 2.56:

Argument	Description
objectname	An instance of the appeon_nvo_db_update object.

Return value

None.

Usage

With the Appeon Commit/Rollback Label, the database operation statement will be sent to the Appeon Server. The server will automatically commit (or roll back) the statement according to the execution result. If the execution succeeds, the result will be committed; if the execution fails, the result will be rolled back.

The first Commit or Rollback statement after the Appeon Commit/Rollback Label will not be submitted to the Appeon Server. Therefore, there must be no more than one database operation statement between the label and the first Commit or Rollback statement. For example, the IF statement should not contain database operation statements, since the executed result will not be committed to the database.

```
gnv_appeonDbLabel.of_autocommitrollback()
UPDATE tab_a ......
IF SQLCA.SQLCODE = 0 THEN
   ......//non-database related bussiness logic
   COMMIT;
ELSE
   ......//non-database related bussiness logic
   ROLLBACK;
   ......
END IF
```

There must be database related operations after the label.

There must be no labels between the Appeon Rollback Label and the first Commit or Rollback statement.

2.3.5.2 Appeon Commit Label

Description

The Appeon Commit Label is used to automatically commit the first database operation.

Controls

appeon_nvo_db_update object

Associated functions

of_autocommit

Syntax

objectname.of_autocommit( )

Table 2.57:

Argument	Description
objectname	An instance of the appeon_nvo_db_update object.

Return value

none.

Usage

After the label there must be database operations.

With Appeon Commit Label, Appeon Server does not validate the execution result of the database operation statement. Instead, the server automatically commits the statement regardless of the execution result.

The first Commit statement after the Appeon Commit Label will not be submitted to the Appeon Server , however, the first Rollback statement will be submitted to the server. Therefore, there should be no more than one database operation between the label and the first Commit statement. For example, the IF statement should not contain database related business logic, since the executed result will not be committed to the database.

```
gnv_appeonDbLabel.of_autocommit()
SELECT.....INTO ......FROM tab_a;
IF SQLCA.SQLCODE = 0 THEN
  ......// non-database related business logic
ELSE
  ...... // non-database related business logic
END IF
COMMIT;
```

There must be no labels between the Appeon Commit Label and the first Commit statement.

2.3.5.3 Appeon Rollback Label

Description

The Appeon Rollback Label is used to automatically roll back the first database operation statement if the operation fails.

Controls

appeon_nvo_db_update object

Associated functions

of_autorollback

Syntax

objectname.of_autorollback( )

Table 2.58:

Argument	Description
objectname	An instance of the appeon_nvo_db_update object.

Return value

None.

Usage

After the label there must be database operations.

With the Appeon Rollback Label, Appeon Server only commits an unsuccessful database operation.

The first Rollback statement after the Appeon Rollback Label will not be submitted to the Appeon Server if the execution fails. Therefore, there should be no more than one database operation between the label and the first Commit or Rollback statement.

There must be no labels between the Appeon Rollback Label and the first Rollback statement.

Code Example

```
gnv_appeonDbLabel.of_autorollback()
IF dw_1.update() <> 1 THEN
  ROLLBACK ;
  ......// non-database related business logic
END IF
```

2.3.5.4 Appeon Queue Labels

Description

Appeon Queue Labels consist of the Appeon Start Queue Label and the Appeon Commit Queue Label. The Appeon Queue Labels are designed for use when it is necessary to execute numerous database operation statements on Appeon Server, and the returned values of the statements are not validated or used. Using the pair of labels can dramatically reduce the number of client-server interactions.

Controls

appeon_nvo_db_update object

Associated functions

of_startqueue, of_commitqueue

Syntax

objectname.of_startqueue ( )

objectname.of_startqueue({integer stopmode})

Table 2.59:

Argument	Description
objectname	An instance of the appeon_nvo_db_update object.
stopmode	0 - continue executing the remaining SQL scripts when an error occurs; 1 - stop executing the remaining SQL scripts when an error occurs. Note: The of_startqueue function without this argument is preserved for compatibility.

objectname.of_commitqueue( )

Table 2.60:

Argument	Description
objectname	An instance of the appeon_nvo_db_update object.

Return value

none.

Usage

Appeon Queue Labels must be used in the same field.

All the database operations in the labels will be submitted to the Appeon Server.

If there are multiple Appeon Commit Queue Labels used together with an Appeon Start Queue Label, only the first Appeon Commit Queue Label that is executed will be effective. Other Appeon Commit Queue Labels will be ignored.

With the stopmode argument, users can choose to continue running or return immediately when an error occurs in the database syntax operation in the queue.

In the Appeon Queue Labels, the SELECT statement cannot be used in the condition statements. The following example is incorrect.

Incorrect Example

```
nv_appeonDbLabel.Of_startqueue()
IF.....THEN
  SELECT STATEMENT 1
else
  SELECT STATEMENT 2
END IF
gnv_appeonDbLabel.Of_commitqueue()
```

In the Appeon Queue Labels, script that stops the execution of another script cannot be included in some events of the DataWindow object, For example, in the following events of DataWindow, the Return statement should not be used: the RetrieveStart event, the RetrieveEnded event, the RowFocusChanged event, the UpdateStart event, the UpdateEnd event, and etc.

For every RETURN statement, there must be an database operation statement or unexpected errors occur.

You can open a cursor in the Appeon Queue Labels.

Using multiple Appeon Queue Labels

Appeon Queue Labels can be embedded in other Appeon Queue Labels. However, only the outer Appeon Queue Labels take effect.

Using non-queue labels together with Appeon Queue Labels

When there are multiple non-queue labels embedded in the Appeon Queue Labels, only the first non-queue label takes effect.

When the other Appeon Labels is embedded in Appeon Queue Labels, the format should be the same as the following code example. Please note that only Commit or Rollback statements are involved in the condition statements.

Code Example

```
nv_appeonDbLabel.of_startqueue()
dw_1.update()
gnv_appeonDbLabel.of_autocommitrollback() // the label takes effect
gnv_appeonDbLabel.of_imdcall() // The label takes no effect
IF dw_2.update() = 1 THEN
  COMMIT;
ELSE
  ROLLBACK;
END if

nv_appeonDbLabel.of_startqueue(1) //Stop immediately when an error occurs
dw_1.update()
gnv_appeonDbLabel.of_autocommitrollback() // the label takes effect
gnv_appeonDbLabel.of_imdcall() // The label takes no effect
if dw_2.update() = 1 THEN
  COMMIT;
ELSE
  ROLLBACK;
END if

nv_appeonDbLabel.of_autocommitrollback()
UPDATE tab_a......
if SQLCA.SQLCODE = 0 THEN
  COMMIT;
ELSE
  ROLLBACK;
END IF
INSERT tab_b......
  COMMIT;
gnv_appeonDbLabel.of_commitqueue()
```

2.3.5.5 Appeon Immediate Call Label

Description

The Appeon Immediate Call Label is used to immediately commit a database operation.

Controls

appeon_nvo_db_update object

Associated functions

of_imdcall

Syntax

objectname.of_imdcall( )

Table 2.61:

Argument	Description
objectname	An instance of the appeon_nvo_db_update object.

Return value

none.

Usage

Appeon Immediate Call Label cannot be used alone, it must be used in Appeon Queue Labels.

With the Appeon Immediate Call Label, the first database operation statement will be sent to the server and executed immediately.

Code Example

```
gnv_appeonDbLabel.of_startretrievequeue()
dw_1.retrieve()
gnv_appeonDbLabel.of_imdcall()
SELECT ......INTO :var_1,:var_2...
IF var_1 > 0 THEN
  para = "ok"
ELSE
  para = "false"
END IF
dw_2.retrieve(para)
gnv_appeonDbLabel.of_endretrievequeue()
```

2.3.5.6 Appeon Update Label

Description

The Appeon Update Label is used to reduce the number of interactions with the server caused by "interrelated updates".

Controls

appeon_nvo_db_update object

Associated functions

of_update

Syntax

objectname.of_update (integer transactionflag, powerobject obj_1, powerobject obj_2)

objectname.of_update (integer transactionflag, powerobject obj_1, powerobject obj_2, powerobject obj_3)

objectname.of_update (integer transactionflag, powerobject obj_1, powerobject obj_2, powerobject obj_3, powerobject obj_4) *objectname.o*f_update (powerobject obj)

objectname.of_update (powerobject obj_1, powerobject obj_2)

objectname.of_update (powerobject obj_1, powerobject obj_2, powerobject obj_3)

objectname.of_update (powerobject obj_1, powerobject obj_2, powerobject obj_3, powerobject obj_4)

Table 2.62:

Argument	Description
objectname	An instance of the appeon_nvo_db_update object.
transactionflag	0 - transaction is automatically committed; 1 - transaction is not automatically committed. Note: the of_update() function without this argument is preserved for compatibility.
obj	The name of the DataWindow, DataStore or DataWindowChild that needs to update.
obj_1	The name of the DataWindow, DataStore or DataWindowChild that needs to update.
obj_2	The name of the DataWindow, DataStore or DataWindowChild that needs to update.
obj_3	(optional) The name of the DataWindow, DataStore or DataWindowChild that needs to update.
obj_4	(optional) The name of the DataWindow, DataStore or DataWindowChild that needs to update.

Return value

Table 2.63:

Integer.	Return values are: 1 - Succeed in update -101 - Fail to update the first DataWindow/DataStore/DataWindowChild -102 - Fail to update the second DataWindow/DataStore/ DataWindowChild -103 - Fail to update the third DataWindow/DataStore/DataWindowChild -104 - Fail to update the fourth DataWindow/DataStore/ DataWindowChild

Usage

The update operations of the DataWindows, DataStores, or DataWindowChild will be submitted to the Appeon Server together. If the operation of a DataWindow, DataStore or DataWindowChile fails, Appeon Server will stop processing the update operation. Users can also use transactionflag argument to control whether to commit or rollback the Database update.

The following script has the same function. However, by using the Appeon Update Label the number of client-server interactions is reduced to one.

Using Appeon Update Label

```
l_rtn = gnv_appeonDb.of_update(0, dw_1,dw_2)
IF l_rtn = 1 THEN
  Messagebox("Success","Update success!")
ELSEIF l_rtn= -102 THEN
  Messagebox("Failure","Update all failure!")
ELSE
  Messagebox("Failure","Update dw_1 failure!")
END IF
```

Without using Appeon Update Label

```
IF dw_1.Update() = 1 THEN
  IF dw_2.Update() = 1 THEN
    COMMIT;
    Messagebox("Success","Update success!")
  ELSE
    ROLLBACK;
    Messagebox("Failure","Update all failure!")
  END IF
ELSE
  ROLLBACK;
  Messagebox("Failure","Update dw_1 failure!")
END IF
```

2.3.6 Calling EJB Component

In order to implement calling EJB components for applications deployed with Appeon, Appeon provides a customized object (EJBObject object), a DLL (EonEJBClient.dll) and a bridge (Appeon Bridge) in Appeon workarounds. With the two features you can call an EJB component either in the PowerBuilder application or in the deployed application. The solution is not suitable for applications that are deployed to Appeon Server for .NET.

Compared to the EJB solution of PowerBuilder, Appeon EJB solution can support more complex parameters such as Structure.

EonEJBClient.dll

Connects to and communicates with the Appeon Bridge. The DLL should exist on the machine on which the Client/Server application runs.

2.3.6.1 Appeon Bridge

Overview

Appeon Bridge is a standard J2EE-compliant Web application that can be deployed to any J2EE compliant application server regardless of whether EJB components exist on the server. It functions as the medium between the clients and EJB components, and can be deployed by installing the appeonbridge.war file, which is located in %AppeonServer%\plugin\appeonbridege.

Bundle EJB Client Proxies and Application Client to Appeon Bridge

Before deploying Appeon Bridge to a J2EE-compliant application server, you must bundle EJB client proxies (JAR file) and Application clients (JAR file) to Appeon Bridge to implement the communication to EJB components. An Application client (for example, websphere.jar for WebSphere) is only required if the J2EE-compliant application server and Java Web server are different (for example, using EAServer as the J2EE-compliant application server and WebSphere as the Java Web Server).

Two ways of bundling the EJB client proxies and Application Clients to Appeon Bridge:

1. Add the proxies and clients to the lib directory in the appeonbridge.war file. Since the appeonbridge.war is a ZIP file, you can check the directory in the WAR file with a third party tools such as WinZip. For more details, take EAServer as an example, shown as below:

   - Locate to %EAServer%\appeon\plugin\appeonbridge folder and open appeonbridge.war with the third party tool such as WinZip.
   - Locate to %EAServer%\deploy\ejbjars\md5generator\com\desta folder.
   - Add full 'md5generator\com\desta' folder to appeonbridge.war\WEB-INF\classes\com directory.
   - Login on EAServer console and locate to EASerer Manager > Local Server > Web Applications.
   - Delete appeonbridge.
   - Deploy %EAServer%\appeon\plugin\appeonbridge\appeonbridge.war.
   - Run your application and test again.

2. Move the EJB Client Proxies and Application Clients to the Lib directory in the Java Web server, and add the directory, where the proxies and clients are stored, to the CLASSPATH environment variable of the machine that hosts the Java Web Server.

Deploy Appeon Bridge to J2EE-compliant application server

Deploying Appeon Bridge (appeonbridge.war) is the same as deploying the file server. Please refer to the corresponding "*Deploying the Appeon File Server*" section under Configuring and deploying the file server. (Or you can refer to documents provided by the corresponding server vendors for the deployment instructions, as deploying Appeon Bridge is the same as deploying any other applications.)

2.3.6.2 EJBObject object

Appeon EJBObject object implements the interaction between client and Appeon Bridge. To use this object, you need to load **appeon_workaround.pbl** into your application and make sure the **EonEJBClient.dll** is stored on the machine where the Client/Server application will be run.

EJBObject object provides the following methods to perform the relevant actions:

Table 2.64:

Method	Description
ConnectServer method	Connecting and disconnecting to Appeon Bridge

Method	Description
DisConnection method	
LookUpJndi method	Obtaining the home interface of an EJB component
CreateRemoteInstance method	Creating and destroying the instance for an EJB component
DestroyRemoteInstance method	
Registering parameter methods	Registering parameters
Invoking component methods	Invoking EJB components
InitLocalLanguage method	Setting the language of the error message

A detail code example of how to call these methods is also provided.

2.3.6.2.1 ConnectServer method

Description

Connects a client application to Appeon Bridge.

Syntax

string EJBObject.ConnectServer (string *url*, string *properties[]*)

Table 2.65:

Parameter	Description
EJBobject	A reference of an EJBobject.
url	URL with port, where Appeon Bridge is installed.
properties[]	Properties of Appeon Bridge.

Return value

String. Returns empty string ("") if it succeeds.

2.3.6.2.2 DisConnection method

Description

Disconnects a client application from Appeon Bridge.

Syntax

string EJBObject.DisConnection ()

Table 2.66:

Parameter	Description

EJBobject	A reference of an EJBobject.

Return value

String. Returns empty string ("") if it succeeds.

2.3.6.2.3 LookUpJndi method

Description

Obtains the home interface of an EJB component in order to create an instance for the component.

Syntax

string EJBObject.lookupjndi (string *jndiname*, ref long *objid*)

Table 2.67:

Parameter	Description
EJBObject	A reference of an EJBobject.
jndiname	The JNDI name of the EJB component.
objid	The handle to the EJB home interface.

Return value

String. Returns empty string ("") if it succeeds.

2.3.6.2.4 InitLocalLanguage method

Description

Sets the language of the error message in PowerBuilder IDE.

Syntax

long EJBObject.initlocallanguage (long *nlocalcode*)

Table 2.68:

Parameter	Description
EJBObject	A reference of an EJBobject.
nlocalcode	A long value representing different languages. 0 English (default) 1 Japanese 2 Korean 3 Simplified Chinese 4 Traditional Chinese

Return value

Long.

2.3.6.2.5 Invoking component methods

Table 2.69:

Invokeretblob	Invokeretblobarray	Invokeretbool	Invokeretboolarray
Invokeretchar	Invokeretchararray	Invokeretdate	Invokeretdatearray
Invokeretdatetime	Invokeretdatetimearray	Invokeretdouble	Invokeretdoublearray
Invokeretint	Invokeretintarray	Invokeretlong	Invokeretlongarray
Invokeretreal	Invokeretrealarray	Invokeretstring	Invokeretstringarray
Invokerettime	Invokerettimearray	Invokeretuint	Invokeretuintarray
Invokeretulong	Invokeretulongarray	Invokeretstru	Invokeretstruarray
Invokeretvoid			

Description

Invoke an EJB component which returns a particular data type. All methods will share the same parameters, syntax and return value.

Syntax

string EJBObject.Invokeblob (long objid, string methodname, boolean autoremove, ref blob retval)

Table 2.70:

Parameter	Description
EJBObject	A reference of an EJBobject.
objid	The handle to the component method.
methodname	The name of the invoking component method.
autoremove	Unsupported. Input false.
retval	The return value of invoking the EJB component method. This parameter does not provided in the Invokeretvoid method. The data type of the reval argument keeps the consistence with the data type used in the invoking methods, except the Invokeretstru and Invokeretstruarray, the data type of the retval arguments for the two methods are both blob.

Return value

String. Returns empty string ("") if it succeeds.

Usage

Variables cannot be null for structure and array.

For a structure to be registered, variables can be :

1. char, string, boolean, int, unit, long, ulong, real, double, datetime, date or time.

2. an array of the above types. The maximum dimension is 3.

3. a structure or a 1-dimensional structure array. And the array must be a fixed array.

For a return value of structure type, variables can be :

1. char, string, boolean, blob, int, unit, long, ulong, real, double, datetime, date or time.

2. a multidimensional array of the above types.

3. a structure or a 1-dimensional structure array. And the array must be a fixed array.

2.3.6.2.6 Registering parameter methods

Register methods are provided to register parameters with different data type. Except RegStruct and RegstructArray, all methods will share the same parameters, syntax, return value.

Table 2.71:

RegChar	RegCharArray	RegDate	RegDateArray
RegDateTime	RegDateTimeArray	RegDouble	RegDoubleArray
RegInt	RegIntArray	RegLong	RegLongArray
RegReal	RegRealArray	RegString	RegStringArray
RegBlob	RegBlobArray	RegBool	RegBoolArray
RegTime	RegTimeArray	RegUInt	RegUIntArray
RegULong	RegULongArray	RegStruct	RegStructArray

Description

Registers a parameter with a certain data type. Syntax below takes RegBlob as an example.

Syntax

string EJBObject.RegBlob (blob data)

Table 2.72:

Parameter	Description
EJBObject	A reference of an EJBobject.
data	The parameter to be registered. Its name can be user-defined, but its type must be consistent with the data type specified in the Register method.

Return value

String. Returns empty string ("") if it succeeds.

RegStruct and RegStructArray methods

Description

Registers a structure or structure array. The two methods will share the same parameters, syntax, return value. Syntax below takes regstructarrary as an example.

Syntax

string EJBObject.regstructarrary (any data [], string javaclassname, readonly classdefinition cdef)

Table 2.73:

Parameter	Description
EJBObject	A reference of an EJBobject.
data	The parameter to be registered and the name is user-defined.
javaclassname	The name of the corresponding Java class in application server.
cdef	ClassDefinition property of structure.

Return value

String. Returns empty string ("") if it succeeds.

Usage

Variables cannot be null for structure and array.

For a structure to be registered, variables can be:

1. char, string, bool, int, unit, long, ulong, real, double, datetime, date or time.
2. an array of the above types. The maximum dimension is 3.
3. a structure or a 1-dimensional structure array. And the array must be a fixed array.
4. (Only for RegStruct method) For the javaclassname parameter, input the full name of the Java class on EJB server corresponding to the structure you defined in PowerBuilder. For example, a.b.c.d.myclassName.

2.3.6.2.7 CreateRemoteInstance method

Description

Creates the instance for an EJB component.

Syntax

EJBObject.CreateRemoteInstance (string *jndiname*, string *homename*, string *methodname*, ref long *beanid*)

Table 2.74:

Parameter	Description
EJBObject	A reference of an EJBobject.
jndiname	The JNDI name of the EJB component
homename	The name of the home interface of an EJB component

methodname	The name of the method
beanid	The handle to the EJB component

Return value

String. Returns empty string ("") if it succeeds.

2.3.6.2.8 DestroyRemoteInstance method

Description

Destroys the instance for an EJB component.

Syntax

EJBObject.DestroyRemoteInstance (long *objid*)

Table 2.75:

Parameter	Description
EJBObject	A reference of an EJBobject.
objid	The handle to the EJB component

Return value

String. Returns empty string ("") if it succeeds.

2.3.6.2.9 Code Example

This section gives you a detail example of how to invoke EJB components in PowerBuilder.

Step 1 - Declare object and connect to Appeon Bridge.

```
ejbobject lo_ejb
long ll_bean1 = 0
long ll_homeHandle = 0
string ls_prop[4]
string ls_serurl,ls_msg
ls_serurl = "http://192.0.2.182:8080/appeonbridge/Dispatch"
ls_prop[1]= "applicationA"
ls_prop[2]= "javax.naming.Context.INITIAL_CONTEXT_FACTORY= 'com.sybase.ejb.InitialContextFactory'
ls_prop[3]= "javax.naming.Context.PROVIDER_URL='iiop://192.0.2.182:9989'"
ls_prop[4]= "username=jagadmin"
ls_prop[5]= "password="
ls_msg = lo_ejb.connectserver(ls_serurl, ls_prop)

if ls_msg = "" then
  MessageBox("connectserver succeed!","srvId ="+string(lo_ejb.il_srvid))
else
  MessageBox("wrong!",ls_msg)
end if
```

Step 2 - Obtain the home interface of an EJB component in a J2EE-compliant server.

```
ls_msg = lo_ejb.lookupjndi ("TestSBeanLess",ref ll_homeHandle)
```

Step 3 - Create instance for EJB component.

```
ls_msg = lo_ejb.createremoteinstance("AllDataType","AllDataType","create", ref ll_bean1)
```

Step 4 - Invoke the EJB components method.

Example one:

```
string ls_msg char c_val c_val = 'a' lo_ejb.regchar(c_val) char retval
  ls_msg = lo_ejb.invokeretchar (ll_bean1, "getChar",true, ref retval)
```

Example two:

```
string ls_msg
boolean b_val
b_val = false
lo_ejb.regbool(b_val)
boolean retval
ls_msg = lo_ejb.invokeretbool (ll_bean1, "getBoolean", true, ref retval)
```

Step 5 - Destroy the instance.

```
ls_msg = lo_ejb.destroyremoteInstance(ll_homeHandle)
If ls_msg <> "" Then
  MessageBox("Destroy Remote Instance Failed",ls_msg)
Return
End If
```

2.3.6.3 Appeon requirements for EJB development

1. Appeon Bridge maps datatypes (except structure) between Java and PowerBuilder is shown as below.

Table 2.76:

char	Char
String	String
boolean	Boolean
short	Int
int	Uint
Int	Long
long	Ulong
long	LongLong
float	Real
double	Double
java.math.BigDecimal	Decimal
double	Number
java.sql.Timestamp	Timestamp
java.sql.Timestamp	DateTime
java.util.Date	Date
java.sql.Time	Time
byte[]	Blob

2. With Appeon EJB solution, Structure data can be passed when invoking EJB components. To implement this, you need to define a Java class in the EJB components. There are two necessary elements in Java Class: 1) **private static String PBMap[]** and 2) **implementing java.io.Serializable interface**. In PBMap array you need to map the members with the identical orderand datatype to a PowerBuilder Structure.

   Following is an example of defining a Java class (please note that the member variables should be in lower case.)

```
package test;
import java.io.Serializable;

public class Simple implements Serializable {
  private short l_int;
  private boolean b_bool;
  private String s_string;

  private static String PBMap[] = {"l_int", "b_bool", "s_string"};

  public String[] getPBMap() {
    return PBMap;
  }

  public boolean isB_bool() {
    return b_bool;
  }

  public short getL_int() {
    return l_int;
  }

  public String getS_string() {
    return s_string;
  }

  public void setB_bool(boolean b_bool) {
    this.b_bool = b_bool;
  }

  public void setL_int(short l_int) {
    this.l_int = l_int;
  }

  public void setS_string(String s_string) {
    this.s_string = s_string;
  }
}
```

2.3.7 Calling .NET/COM server components (.NET only)

Applies to

Appeon Server for .NET.

Supported server component types

- .NET components: All valid .NET components, including executable files (.exe) and DLL files (.dll).

  Supported parameters: primitive type parameters, such as int, vlong char, and boolen. Nonprimitive type parameters, such as class, are unsupported.

Supports reference parameters.

- COM components: COM/COM+ components

  Supported parameters: primitive type parameters, such as byte, int, long, and float.

  Supports reference parameters.

Description

To call .NET/COM components, Appeon provides a non-autoinstantiated NVO - AppeonDotNetComponent - as the proxy object to call the server-side components. The user can either create a local instance of AppeonDotNetComponent for each server component, or directly use an existing instance of AppeonDotNetComponent. The user must specify the properties of the instance, such as the component type, the library name and the class name, to bind the instance with the server component, or change the instance properties during runtime to dynamically bind with a different component.

It provides a universal single interface and a set of parameters which determines which component and methods will be called.

Note:

The script to call AppeonDotNetComponent takes effect only after the PowerBuilder application is deployed, and has no effect when the PowerBuilder application is run.

Register

The COM component must be registered using the regsvr32 tool.

Storage location

The components must reside in the %appeon%/AEM/components folder on the Appeon Server machine. You only need to place the .tlb library files and .dll files of the COM components to the folder. %appeon% indicates the installation directory of Appeon Server.

2.3.7.1 AppeonDotNetComponent object

2.3.7.1.1 Properties

Properties for AppeonDotNetComponent.

Table 2.77:

Properties	Type	Description
ComponentType	String	The type of the component to be called. “1” indicates a .NET Assembly to be called. “2” indicates an unmanaged-code COM component to be called. “3” indicates a managed-code COM component to be called. “4” indicates a built-in Appeon Workaround .NET Assembly to be called.

Properties	Type	Description
TypeLib	String	The name of the component library. Appeon Server uses this name to find the component.
ClassDescript	String	The class name.
ReturnValue	Any	Read-only. The return value of functions. The value and value type varies from function to function.
ErrorText	String	Read-only. The error message of functions. The message varies from function to function. Empty string if no error.

2.3.7.1.2 Methods

of_execinterface

Description

Calls the method in the binding component.

Syntax

long of_execinterface (interfacename, {paralist})

Table 2.78:

Argument	Description
interfacename	The name of the component method.
Paralist	Optional. Arrays of Any type. Specifies the parameter arrays for the component method.

Return value

Long.

0 – Call succeeded. Gets the value from the ReturnValue property of the proxy object.

-1 – Call failed. Gets the error message from the ErrorText property of the proxy object.

Usage

Before calling this method, use the proxy object properties to bind with the target component. If the component method contains no parameters, simply specify the method name. If the component method contains parameters, define an Any type array before the call, then place the argument to the array, finally pass the array as the second parameter of the function.

Examples

Example 1: the interface contains no parameters.

```
AppeonDotNetComponent lu_apf

lu_apf = create AppeonDotNetComponent lu_apf
lu_apf.ComponentType = "2"
lu_apf.TypeLib = "test.dll"
lu_apf.ClassDescript = "testclass"

ll_ret = lu_apf.of_ExecInterface("test")
```

Example 2: the interface contains four parameters, their types are: string, int, long, and string.

```
//Define the array variable
AppeonDotNetComponent lu_apf
any la_1[]

la_1[1] = "Appeon"
la_1[2] = 100
la_1[3] = 256
la_1[4] = "Sybase"

lu_apf = create AppeonDotNetComponent lu_apf
lu_apf.ComponentType = "1"
lu_apf.TypeLib = "testdotnet.dll"
lu_apf.ClassDescript = "interface1"

ll_ret = lu_apf.of_ExecInterface("test_dotnet", la_1)
```

2.3.7.1.3 Events

Constructor

Destructor

Constructor

Description

It will be triggered when you create an instance from a user-defined proxy object inherited from AppeonDotNetComponent.

Event ID

pbm_constructor

Argument

None

Return values

Long

Usage

Do not write scripts to this event directly, because the scripts will be abandoned when the application is deployed. Instead, define and inherit an object from AppeonDotNetComponent, and add the scripts to the Constructor event of the new object. The usage is the same as that of the PowerBuilder system object. For example, you can initialize the property value of this event, or define relevant information objects.

Destructor

Description

It will be triggered when you explicitly call Destroy to destroy the instance of a user-defined proxy object inherited from AppeonDotNetComponent.

Event ID

pbm_destructor

Argument

None

Return value

Long

Usage

Do not write scripts to this event directly, because the scripts will be abandoned when the application is deployed. Instead, define and inherit an object from AppeonDotNetComponent, and add the scripts to the Destructor event of the new object. The usage is the same as that of the PowerBuilder system object. For example, you can add scripts to release the instances related with the proxy object.

2.3.7.1.4 Code Examples

Example 1:

```
long lRet
int iResult
string strError
appeondotnetcomponent comcaller

//create appeondotnetcomponent instance and set properties

comcaller = create appeondotnetcomponent
comcaller.componenttype = '1'
comcaller.typelib = 'DotNetDll.dll'
comcaller.classdescript = 'DotNetClass'

//invoke component method

lRet = comcaller.of_execinterface("GetInt")
if lRet = 0 then
iResult = comcaller.ReturnValue
else
strError = comcaller.ErrorText
end if

// Bind with a component and call the component method

comcaller.componenttype = '2'
comcaller.typelib = 'comfordotnet.dll'
comcaller.classdescript = 'ifdotnet'
comcaller.of_execinterface("getint")
```

Example 2:

```
// Call a method with reference parameters

any paralist[]
long refparam1 = 32764
long refparam2 = 32763

paralist[1] = refparam1
paralist[2] = refparam2

comcaller.componenttype = '1'
comcaller.typelib = 'DotNetDll.dll'
comcaller.classdescript = 'DotNetClass'
comcaller.of_execinterface("GetIntAndRefInt",paralist)

refparam1 = paralist[1]
rafparam2 = paralist[2]
```

2.3.8 Calling Web Service

Description

To call Web services, Appeon provides a non-autoinstantiated NVO – AppeonWebServiceComponent – as the proxy object to call Web services. The user can either create a local instance of AppeonWebServiceComponent for Web service, or directly use an existing instance of AppeonWebServiceComponent. The user must specify the properties of the instance, such as the proxy type, the Web service location and the class name, to bind the instance with the Web service, or change the instance properties during runtime to dynamically bind with a different Web service.

It provides a universal single interface and a set of parameters which determines which Web service and methods will be called.

Note:

The script to call AppeonWebServiceComponent takes effect on Web only, and has no effect in the PowerBuilder application.

2.3.8.1 appeonwebservicecomponent object

2.3.8.1.1 Properties

Properties for appeonwebservicecomponent.

Table 2.79:

Properties	Type	Description
CallType	String	The proxy type of the Web service to be called. " 1" indicates the proxy type is Dynamic Proxy. "2" indicates the proxy type is DLL Proxy. Appeon for J2EE editions support only CallType="1".
ProxyDllOrUrl	String	If CallType="1", it indicates the URL of the Web service to be called; If CallType="2", it indicates the DLL name of the proxy used by the Web service to be called
ClassDescript	String	If CallType="1", it can be null; If CallType="2", it indicates class name.
ReturnValue	Any	Read-only. The return value of functions. The value and value type varies from function to function.
ErrorText	String	Read-only. The error message of functions. The message varies from function to function. Empty string if no error.

2.3.8.1.2 Methods

of_callwebservice

Description

Calls the method in the binding Web service.

Syntax

long of_callwebservice(string methodname, {ref any paralist[]})

Table 2.80:

Argument	Description
methodname	The name of the Web service method.
paralist	Optional. Arrays of Any type. Specifies the parameter arrays for the Web service method.

Return value

Long.

0 – Call succeeded. Gets the value from the ReturnValue property of the proxy object.

-1 – Call failed. Gets the error message from the ErrorText property of the proxy object.

Usage

Before calling this method, use the proxy object properties to bind with the target Web service. If the Web service method contains no parameters, simply specify the method name. If the Web service method contains parameters, define an Any type array before the call, then place the argument to the array, finally pass the array as the second parameter of the function.

Examples

Example 1: the interface contains no parameters.

```
appeonwebservicecomponent caller
caller= create appeonwebservicecomponent
caller.calltype="1"
caller.proxydllorurl="http://localhost/webservice.asmx"
caller.classdescript=""
integer IRet
IRet=caller.of_callwebservice("GetUserName")
```

Example 2: the interface contains two parameters, their types are any.

```
any paralist[]
appeonwebservicecomponent caller
caller=create appeonwebservicecomponent
caller.calltype="1"
caller.proxydllorurl="http://localhost/webservice.asmx"
caller.classdescript=""
paralist[1]="param1"
paralist[2]="param2"
IRet=caller.of_callwebservice("GetUserName",paralist)
```

2.3.8.1.3 Events

Constructor

Destructor

Constructor

Description

It will be triggered when you create an instance from a user-defined proxy object inherited from AppeonWebServiceComponent.

Event ID

pbm_constructor

Argument

None

Return values

Long

Usage

Do not write scripts to this event directly, because the scripts will be abandoned when the application is deployed. Instead, define and inherit an object from AppeonWebServiceComponent, and add the scripts to the Constructor event of the new object. The usage is the same as that of the PowerBuilder system object. For example, you can initialize the property value of this event, or define relevant information objects.

Destructor

Description

It will be triggered when you explicitly call Destroy to destroy the instance of a user-defined proxy object inherited from AppeonWebServiceComponent.

Event ID

pbm_destructor

Argument

None

Return value

Long

Usage

Do not write scripts to this event directly, because the scripts will be abandoned when the application is deployed. Instead, define and inherit an object from AppeonWebServiceComponent, and add the scripts to the Destructor event of the new object. The usage is the same as that of the PowerBuilder system object. For example, you can add scripts to release the instances related with the proxy object.

2.3.8.1.4 Code Examples

Example 1:

```
long IRet
int iResult
string strError
appeonwebservicecomponent caller

//create appeonwebservicecomponent instance and set properties

caller=create appeonwebservicecomponent

//if proxy type is DynamicProxy, the value of calltype is 1;
//if proxy type is DllProxy type, the value of calltype is 2.
//Appeon for J2EE editions support only CallType="1".
//Appeon for .NET edition supports both.

caller.calltype="1"

//since proxy type is DynamicProxy,the value of proxydllorurl is the address of
//the webservice to be called;or else, the value of proxydllorurl is the name
//of DLL.
caller.proxydllorurl="http://localhost/webservice.asmx"
caller.classdescript=""

//Invoke webservice method

IRet=caller.of_callwebservice("GetUserName")
if IRet=0 then
 iResult=caller.ReturnValue
else
 strError=caller.ErrorText
end if
```

Example 2:

```
long IRet
int iResult
string strError
any paralist[]
appeonwebservicecomponent caller

//create appeonwebservice instance and set properties

caller=create appeonwebservicecomponent
caller.calltype="1"
caller.proxydllorurl="http://localhost/webservice.asmx"
caller.classdescript=""
paralist[1]="param1"
paralist[2]="param2"

//invoke webservice method

IRet=caller.of_callwebservice("GetUserName",paralist)
```

Example 3:

```
long IRet
int iResult
string strError
appeonwebservicecomponent caller

//create appeonwebservicecomponent instance and set properties

caller=create appeonwebservicecomponent
caller.calltype="2"
caller.proxydllorurl="DotDllForJava"
caller.classdescript="MyJavaWebService"

//invoke webservice method

IRet=caller.of_callwebservice("GetUserName")
if IRet=0 then
 iResult=caller.ReturnValue
else
 strError=caller.ErrorText
end if
```

2.4 Mobile Device API

Below are the Mobile device APIs which aim to realize the access to the features of multiple mobile devices, and they are called by Mobile SDK to implement the various functionalities in Appeon mobile application.

Note: For each API, there are two objects with the same name. The only difference is that one with the "ex" suffix (e.g. eon_mobile_awsex object) , and the other without the "ex"suffix (e.g. eno_mobile_aws object). The former is the extension of the latter, so you can directly call the corresponding function on the object with the "ex" suffix.

2.4.1 Appeon Workspace

Manipulates the various features of Appeon Workspace, such as the Appeon Workspace orientation, the title bar, the assistive touch bar, the log, the project information, the version, etc.

2.4.1.1 eon_mobile_awsex object

2.4.1.1.1 of_log function

Description

Adds the log information to the Appeon Workspace log, and the default loggin level is info.

Supported on mobile client only.

Syntax

of_log (value integer *ai_level*, value string *as_info*)

Argument	Description
ai_level	1 - INFO level. 3 - ERROR level. 5 - DEBUG level.

Argument	Description
as_info	The log information.

Return value

integer

1 - Success.

-1 - It is called in PowerBuilder or Appeon Web, or there is an error.

2.4.1.1.2 of_clearlog function

Description

Clears the Appeon Workspace log informtion.

Supported on mobile client only.

Syntax

None.

Return value

integer

1 - Success.

-1 - It is called in PowerBuilder or Appeon Web, or there is an error.

2.4.1.1.3 of_getassistivetouchmode function

Description

Gets the current assistive touch mode.

Supported on mobile client only.

Syntax

None.

Return value

integer

0 - The assistive touch mode is Left-click.

1 - The assistive touch mode is Right-click.

2 - The assistive touch mode is Drag.

-1 - It is called in PowerBuilder or Appeon Web, or there is an error.

2.4.1.1.4 of_setassistivetouchmode function

Description

Sets the current assistive touch mode.

Supported on mobile client only.

Syntax

of_setassistivetouchmode (value integer *ai_mode*)

Argument	Description
ai_mode	0 - Sets to the Left-click mode. 1 - Sets to the Right-click mode. 2 - Sets to the Drag mode.

Return value

integer

1 - Success.

-1 - It is called in PowerBuilder or Appeon Web, or there is an error.

2.4.1.1.5 of_getassistivetouchbtnvisible function

Description

Detects if the assistive touch bar is visible.

Supported on mobile client only.

Syntax

None.

Return value

integer

1 - The assistive touch bar is visible.

0 - The assistive touch bar is invisible.

-1 - It is called in PowerBuilder or Appeon Web, or there is an error.

2.4.1.1.6 of_setassistivetouchbtnvisible function

Description

Sets whether the assistive touch bar is visible.

Supported on mobile client only.

Syntax

of_setassistivetouchbtnvisible (value integer *ai_mode*)

Argument	Description
ai_mode	1 - Sets the assistive touch bar to be visible. 0 - Sets the assistive touch bar to be invisible.

Return value

integer

1 - Success.

-1 - It is called in PowerBuilder or Appeon Web, or there is an error.

2.4.1.1.7 of_getapporientation

Description

Detects if the Appeon Workspace screen orientation is Landscape or Portrait.

Supported on mobile client only.

This function is only valid for the current running application.

Syntax

None.

Return value

integer

0 - Unknown.

1 - Portrait.

2 - Landscape.

-1 - It is called in PowerBuilder or Appeon Web, or there is an error.

2.4.1.1.8 of_setapporientation

Description

Sets the screen orientation of the application currently running in Appeon Workspace.

This functionality has no impact on the screen orientation of the device.

If of_setapprotationlock is executed to lock the application screen orientation before of_setapporientation is called,

then of_setapporientation will not take effect and return -1.

Supported on mobile client only.

Syntax

of_setapporientation (value integer *ai_mode*)

Argument	Description
ai_mode	1 - Sets screen orientation to Portrait. 2 - Sets screen orientation to Landscape.

Return value

integer

1 - Success.

-1 - It is called in PowerBuilder or Appeon Web, or there is an error.

2.4.1.1.9 of_getapprotationlock function

Description

Detects if Appeon Workspace screen rotation is locked.

Supported on mobile client only.

Not supported on Emulator, and this function is only valid for the current running application.

Syntax

None.

Return value

integer

1 - Locked.

0 - Unlocked.

-1 - It is called in PowerBuilder or Appeon Web, or there is an error.

2.4.1.1.10 of_setapprotationlock function

Description

Sets whether to lock the screen orientation of the application currently running in Appeon Workspace.

This functionality has no impact on the screen orientation of the device.

If of_setapprotationlock is executed to lock the application screen orientation before of_setapporientation is called,

then of_setapporientation will not take effect and return -1.

Supported on mobile client only.

Syntax

of_setapprotationlock (value integer *ai_mode*)

Argument	Description
ai_mode	1 - Locks the screen orientation of Appeon Workspace, so it will not rotate when the screen orientation of the device changes. 0 - UnLocks the screen orientation of Appeon Workspace.

Return value

integer

1 - Success.

-1 - It is called in PowerBuilder or Appeon Web, or there is an error.

2.4.1.1.11 of_gettitlebarvisible function

Description

Detects if the Appeon Workspace title bar is visible or invisible.

Supported on mobile client only.

Syntax

None.

Return value

integer

1 - The title bar is visible.

0 - The title bar is invisible.

-1 - It is called in PowerBuilder or Appeon Web, or there is an error.

2.4.1.1.12 of_settitlebarvisible function

Description

Sets whether the Appeon Workspace title bar is visible or invisible.

Supported on mobile client only.

Syntax

of_settitlebarvisible (value integer *ai_mode*)

Argument	Description
ai_mode	1 - Sets the title bar to be visible. 0 - Sets the title bar to be invisible.

Return value

integer

1 - Success.

-1 - It is called in PowerBuilder or Appeon Web, or there is an error.

2.4.1.1.13 of_getappinfo function

Description

Gets the information of the current application in Appeon Workspace.

Supported on mobile client only.

Syntax

of_getappinfo (ref string *as_appname*, ref string *as_appurl*)

Argument	Description
as_appname	Returns the name of the current application in Appeon Workspace. Returns empty string if it is called in PowerBuilder or Appeon Web or if there is any error.

Argument	Description
as_appurl	Returns the value of the "App URL" setting of the current application in Appeon Workspace. Returns empty string if it is called in PowerBuilder or Appeon Web or if there is any error.

Return value

integer

1 - Success.

-1 - It is called in PowerBuilder or Appeon Web, or there is an error.

2.4.1.1.14 of_getversion function

Description

Gets the version number of Appeon Workspace.

Supported on mobile client only.

Syntax

of_getversion (ref string *as_version*)

Argument	Description
as_version	Returns the Appeon Workspace version number. Returns empty string if it is called in PowerBuilder or Appeon Web, or if there is any error.

Return value

integer

1 - Success.

-1 - It is called in PowerBuilder or Appeon Web, or there is an error.

2.4.1.1.15 of_sendmail function

Description

Sends a mail.

Supported on mobile client only.

Syntax

of_sendmail (value eon_mobile_str_mailcontent astr_content)

Variable list of the eon_mobile_str_mailcontent struct

Data type	Argument	Description
integer	s_subject	Specifies the subject of the mail.
integer	s_notetext	Specifies the content of the mail body.
boolean	b_allowedit	Sets if the mail format is HTML.
string	as_recipient[ ]	Specifies the recipient list of the mail.
string	as_cc[ ]	Specifies the Cc recipient list of the mail.
string	as_bcc[ ]	Specifies the Bcc recipient list of the mail.
string	as_attachmentfile[ ]	Attachment file path list.

Return value

integer

1 - Sent the mail successfully.

0 - Cancelled sending the mail.

-1 - Failed to send the mail, or It is called in PowerBuilder or Appeon Web, or there is an error.

-2 - The mail account is not configured.

2.4.1.1.16 of_getwindowlisticonvisible function

Description

Detects if the window list icon is visible or not.

Supported on mobile client only.

Syntax

None.

Return value

integer

1 - The window list icon is visible.

0 - The window list icon is invisible.

-1 - It is called in PowerBuilder or Appeon Web, or there is an error.

2.4.1.1.17 of_setwindowlisticonvisible function

Description

Sets whether the window list icon is visible or not.

Supported on mobile client only.

Syntax

of_setwindowlisticonvisible (value integer *ai_mode*)

Argument	Description
ai_mode	1 - Sets the window list icon to be visible. 0 - Sets the window list icon to be invisible.

Return value

integer

1 - Success.

-1 - It is called in PowerBuilder or Appeon Web, or there is an error.

2.4.1.1.18 of_getcloseappiconvisible function

Description

Detects if the close app icon is visible or not.

Supported on mobile client only.

Syntax

None.

Return value

integer

1 - The close app icon is visible.

0 - The close app icon is invisible.

-1 - It is called in PowerBuilder or Appeon Web, or there is an error.

2.4.1.1.19 of_setcloseappiconvisible function

Description

Sets whether the close app icon is visible or not.

Supported on mobile client only.

Syntax

of_setcloseappiconvisible (value integer *ai_mode*, string *as_message*)

Argument	Description
ai_mode	1 - Sets the close app icon to be visible. 0 - Sets the close app icon to be invisible.
as_message	The message that displays in the dialog box which pops up when the Close App icon is tapped. A Yes/No button will be displayed in the dialog box. When Yes is tapped, the app

Argument	Description
	will be closed. When No is tapped, the app will not be closed and will continue running. If as_message is null or blank, then no message box will be displayed and the app will be closed immediately.

Return value

integer

1 - Success.

-1 - It is called in PowerBuilder or Appeon Web, or there is an error.

2.4.1.1.20 of_getdwmousemovemode function

Description

Detects if the pbm_dwnmousemove event ID of DataWindow is supported.

Supported on mobile client only.

Syntax

None.

Return value

integer

1 - Supported.

0 - Unsupported.

-1 - It is called in PowerBuilder or Appeon Web, or there is an error.

2.4.1.1.21 of_setdwmousemovemode function

Description

Sets whether to support the pbm_dwnmousemove event ID of DataWindow.

Supported on mobile client only.

Syntax

of_setdwmousemovemode (value integer *ai_mode*)

Argument	Description
ai_mode	1 - Supported. 0 - Unsupported.

Return value

integer

1 - Success

-1 - It is called in PowerBuilder or Appeon Web, or there is an error.

2.4.2 Barcode

Creates or reads the information of a barcode.

2.4.2.1 eon_mobile_barcodeex object

2.4.2.1.1 of_create function

Description

Generates the image for a barcode.

Supported on mobile client only.

Syntax

of_create (ref string *as_data*, integer *ai_format*, string *as_filepath*)

Argument	Description
as_data	Sets the data of a barcode.
ai_format	Sets the format of barcode.
as_filepath	Returns the path of the generated barcode image.

Following is the list of supported barcode formats:

0	Unknown
1	EAN-2, GS1 2-digit add-on
2	EAN-5, GS1 5-digit add-on
3	EAN-8, EAN-8
4	UPC-E4
5	ISBN-10, from EAN-13
6	UPC-A
7	EAN-13
8	ISBN-13, from EAN-13
9	COMPOSITE, EAN/UPC composite
10	Interleaved 2 of 5
11	CODE 128
12	CODE 93
13	CODE 39
14	ITF
20	QR CODE
21	DataMatrix
22	AZTEC

Return value

integer

1 - Success.

-1 - It is called in PowerBuilder or Appeon Web, or there is an error.

2.4.2.1.2 of_read function

Description

Reads the information of a barcode.

Supported on mobile client only.

Syntax

of_read (ref string *as_data*, integer *ai_format*)

Argument	Description
as_data	Returns the data of a barcode.
ai_format	Returns the format of barcode.

Following is the list of supported barcode formats:

0	Unknown
1	EAN-2, GS1 2-digit add-on
2	EAN-5, GS1 5-digit add-on
3	EAN-8, EAN-8
4	UPC-E4
5	ISBN-10, from EAN-13
6	UPC-A
7	EAN-13
8	ISBN-13, from EAN-13
9	COMPOSITE, EAN/UPC composite
10	Interleaved 2 of 5
11	CODE 128
12	CODE 93
13	CODE 39
14	ITF
20	QR CODE
21	DataMatrix
22	AZTEC

Return value

integer

1 - Success.

0 - Cancel.

-1 - It is called in PowerBuilder or Appeon Web, or there is an error.

2.4.3 Camera

Captures or views a photo using the device's camera.

2.4.3.1 eon_mobile_cameraex object

2.4.3.1.1 of_takefile function

Description

Opens the default camera application and uses it to take photo or video.

Once the photo or video is taken, the camera application automatically closes with all the files saved in the album, and returns to the current application.

The supported video file format is MOV, and the supported photo file format is JPG.

Supported on mobile client only.

Syntax

objectname.of_takefile (ref integer *ai_filetype*, boolean *ab_allowedit*, ref string as_filepath)

Argument	Description
ai_filetype	Sets the type of file to take. The camera application will be set to this type by default when it is opened. Users can also manually switch the type in the camera application. 1 - Takes photo. 2 - Takes video.
ab_allowedit	Sets if the file is editable after picture-taking.
as_filepath	Returns the full path pointing to the file in the sandbox cache directory. The video and photo files will be saved into device's album, and will be copied to the cache directory of current application's sandbox. The files in this cache directory will be removed when the application exits.

objectname.of_takefile (eno_mobile_str_cameraoption astr_option, ref string as_filepath)

value eon_mobile_str_cameraoption astr_option

Sets the camera options.

Variable list of the eon_mobile_str_cameraoption struct

Argument	Description
i_filetype	Sets the type of file to take. The camera application will be set to this type by default when it is opened. Users can also manually switch the type in the camera application. 1 - Takes photo. 2 - Takes video.
b_allowedit	Sets if the file is editable after picture-taking.
as_filepath	Returns the full path of the photo file or video file. Returns empty string if the picture-taking is cancelled, or if there is any error.

objectname.of_takefile (eno_mobile_str_cameraoption astr_option, ref string as_filepath, ref blob ablb_data)

value eon_mobile_str_cameraoption astr_option

Sets the camera options.

Variable list of the eon_mobile_str_cameraoption struct

Argument	Description
i_filetype	Sets the type of file to take. The camera application will be set to this type by default when it is opened. Users can also manually switch the type in the camera application. 1 - Takes photo. 2 - Takes video.
b_allowedit	Sets if the file is editable after picture-taking.
as_filepath	Returns the full path of the photo file or video file. Returns empty string if the picture-taking is cancelled, or if there is any error.
ablb_data	Returns the file data of the photo file or video file.

Return value

integer

1 - Take photo successfully, and *as_filepath* will return the full path of the photo file, and return empty string if the picture-taking is cancelled, or if there is any error.

2 - Take video successfully, and *as_filepath* will return the full path of the video file, and return empty string if the picture-taking is cancelled, or if there is any error.

0 - Cancel taking photo or video.

-1 - It is called in PowerBuilder or Appeon Web, or there is an error.

2.4.3.1.2 of_openalbums function

Description

Opens the album for the user to select a photo or video.

Supported on mobile client only.

Syntax

of_openalbums (ref string *as_filepath*, ref blob *ablb_data*)

Argument	Description
as_filepath	Returns the full path pointing to this file in the sandbox cache directory. The selected video or photo file will be copied to the cache directory of current application's sandbox. The files in this cache directory will be removed when the application exits.
ablb_data	Returns the file data of the photo file or video file.

Return value

integer

1 - Select a photo file successfully, and *as_filepath* will return the full path pointing to the photo file in the sandbox cache directory.

2 - Select a video file successfully, and *as_filepath* will return the full path pointing to the video file in the sandbox cache directory.

0 - Cancel selecting file from the album.

-1 - It is called in PowerBuilder or Appeon Web, or there is an error.

2.4.4 Connection

Obtains the network state and connection type.

2.4.4.1 eon_mobile_connectionex object

2.4.4.1.1 of_getconnectioninfo function

Description

Gets the device's network connection information.

Supported on mobile client only.

Syntax

None.

Return value

integer

0 - No connection.

1 - Cell network.

2 - Wi-Fi network.

-1 - It is called in PowerBuilder or Appeon Web, or there is an error.

2.4.5 Device

Obtains the device specific information, such as the device type, the device DPI/PPI, the OS version, the device's memory, the device's OS, the screen resolution, etc.

2.4.5.1 eon_mobile_deviceex object

2.4.5.1.1 of_getdeviceid function

Description

Gets the device's unique identifier (only supported on iOS 6.0 or later),

which depends on the device manufacturer, product type, and operating system.

Supported on mobile client only.

Syntax

of_getdeviceid (ref string *as_id*)

Argument	Description
as_id	Returns the device unique identifier. Returns empty string if it is called in PowerBuilder or Appeon Web or, if there is any error.

Return value

integer

1 - Success.

-1 - It is called in PowerBuilder or Appeon Web, or there is an error.

2.4.5.1.2 of_getdevicetype function

Description

Gets the device type from the manufacturer.

Supported on mobile client only.

Syntax

of_getdevicetype (ref string *as_type*)

Argument	Description
as_type	Returns the device type; e.g., "iPod touch", "iPhone", or "iPad". The value of device type is set by the manufacturer. Returns empty string if it is called in PowerBuilder or Appeon Web or if there is any error.

Return value

integer

1 - Success.

-1 - It is called in PowerBuilder or Appeon Web, or there is an error.

2.4.5.1.3 of_getdpi function

Description

Gets the device DPI value.

Supported on mobile client only.

Syntax

None.

Return value

integer

>0 - Return the device DPI value.

-1 - It is called in PowerBuilder or Appeon Web, or there is an error.

2.4.5.1.4 of_getfreememory function

Description

Gets the total number (in KB) of the device's current available free memory.

Supported on mobile client only.

Syntax

None.

Return value

long

>0 - Return the total number of the device free memory.

-1 - It is called in PowerBuilder or Appeon Web, or there is an error.

2.4.5.1.5 of_getname function

Description

Gets the device's machine name.

Supported on mobile client only.

Syntax

of_getname (ref string *as_name*)

Argument	Description
as_name	Returns the device's machine name. Returns empty string if it is called in PowerBuilder or Appeon Web, or if there is any error.

Return value

integer

1 - Success.

-1 - It is called in PowerBuilder or Appeon Web, or there is an error.

2.4.5.1.6 of_getorientation function

Description

Detects if the device screen orientation is Landscape or Portrait.

Supported on mobile client only.

Syntax

None.

Return value

integer

0 - Unknown.

1 - Portrait, Home button at the bottom.

2 - Portrait, Home button on the top.

3 - Landscape, Home button to the right.

4 - Landscape, Home button to the left.

-1 - It is called in PowerBuilder or Appeon Web, or there is an error.

2.4.5.1.7 of_getplatform function

Description

Gets the device's OS name.

Supported on mobile client only.

Syntax

of_getplatform (ref string *as_platform*)

Argument	Description
as_platform	Returns the device's OS name; e.g., returns "iPhone OS" if the device is iPhone, iPad, or iPod touch. The value of OS name is obtained from the mobile device. Returns empty string if it is called in PowerBuilder or Appeon Web, or if there is any error.

Return value

integer

1 - Success.

-1 - It is called in PowerBuilder or Appeon Web, or there is an error.

2.4.5.1.8 of_getppi function

Description

Gets the device's PPI value.

Supported on mobile client only.

Syntax

None.

Return value

integer

>0 - Return the device PPI value.

-1 - It is called in PowerBuilder or Appeon Web, or there is an error.

2.4.5.1.9 of_getresolution function

Description

Gets the device's resolution.

Supported on mobile client only.

Syntax

of_getresolution (ref integer *ai_height*, integer *ai_width*)

Argument	Description
ai_height	Returns the device's screen height (in pixels).
ai_width	Returns the device's screen width (in pixels).

Return value

integer

1 - Success.

-1 - It is called in PowerBuilder or Appeon Web, or there is an error.

2.4.5.1.10 of_gettotalmemory function

Description

Gets the total number (in KB) of the device memory.

Supported on mobile client only.

Syntax

None.

Return value

long

>0 - Return the total number of the device memory.

-1 - It is called in PowerBuilder or Appeon Web, or there is an error.

2.4.5.1.11 of_getosversion function

Description

Gets the OS version.

Supported on mobile client only.

Syntax

of_getosversion (ref string *as_version*)

Argument	Description
as_version	Returns the device's OS version.e.g., returns "6.0" if the iOS version is 6.0; returns "6.0.1" if the iOS version is 6.0.1. The value of OS version is obtained from the mobile device. Returns empty string if it is called in PowerBuilder or Appeon Web, or if there is any error.

Return value

integer

1 - Success.

-1 - It is called in PowerBuilder or Appeon Web, or there is an error.

2.4.5.1.12 of_getstatusbarvisible function

Description

Detects if the system status bar is visible.

Supported on mobile client only.

Syntax

None.

Return value

integer

1 - The system status bar is visible.

0 - The system status bar is invisible.

-1 - It is called in PowerBuilder or Appeon Web, or there is an error.

2.4.5.1.13 of_setstatusbarvisible function

Description

Sets whether the system status bar is visible.

Supported on mobile client only.

Syntax

of_setstatusbarvisible (value integer *ai_mode*)

Argument	Description
ai_mode	1 - Sets the system status bar to be visible. 0 - Sets the system status bar to be invisible.

Return value

integer

1 - Success.

-1 - It is called in PowerBuilder or Appeon Web, or there is an error.

2.4.6 Geolocation

Gets the current position and opens the geolocation functionality.

2.4.6.1 eon_mobile_geolocationex object

2.4.6.1.1 of_isnabled function

Description

Detects if the geolocation services can be used.

Supported on mobile client only.

Syntax

None.

Return value

integer

1 - Enabled.

0 - Disabled.

-1 - If it is called in PowerBuilder or Appeon Web, or there is an error.

2.4.6.1.2 of_open function

Description

Gets the current position and opens the geolocation function.

Supported on mobile client only.

Syntax

of_open (value integer *ai_locationaccuracy*, integer *ai_distancefilter*)

Argument	Description
ai_locationaccuracy	Location accuracy update, recommended to use 0 or 1 for the automatic selection. 0 - Automatically selects the optimal accuracy. 1 - Navigates to the optimal accuracy. >2 - Accuracy (in meters).
ai_distancefilter	Location filter, used to control the location update message frequency (in meters). 0 - Notifies by every update. >0 - Updates only when the location change exceeds this value

Return value

integer

1 - Success.

-1 - If it is called in PowerBuilder or Appeon Web, or there is an error.

2.4.6.1.3 of_close function

Description

Closes the geolocation function.

Supported on mobile client only.

Syntax

None.

Return value

integer

1 - Success.

-1 - If it is called in PowerBuilder or Appeon Web, or there is an error.

2.4.6.1.4 of_getcurrentposition

Description

Gets the current position information.

Supported on mobile client only.

Syntax

of_getcurrentposition (ref eon_mobile_str_coordinates astr_coordinates)

Variable list of the eon_mobile_str_coordinates struct

Data type	Argument	description
datetime	dt_timestamp	The timestamp to get the current position information.
decimal{6}	dec_latitude	The latitude value of the current position.
decimal{6}	dec_longitude	The longitude value of the current position.
decimal{2}	dec_accuracy	The latitude and longitude positioning accuracy.
decimal{2}	dec_altitude	The altitude value of the current position.
decimal{2}	dec_altitudeaccuracy	The altitude positioning accuracy.
decimal{2}	dec_heading	The degrees clockwise from true north (0 to 359.99 degrees).
decimal{2}	dec_speed	The displacement velocity (m/sec).

Return value

integer

1 - Success.

-1 - If it is called in PowerBuilder or Appeon Web, or there is an error.

2.4.6.1.5 of_register function

Description

Registers the object and the event to be bound with the oe_error and oe_locationchanged events.

Supported on mobile client only.

Syntax

of_register (powerobject *apb_bind*, string *as_changedevent*, string *as_errorevent*)

Argument	Description
apb_bind	The object to be bound with the oe_error and oe_locationchanged event.
as_changedevent	The event to be bound with the oe_locationchanged event.
as_errorevent	The event to be bound with the oe_error event.

Return value

integer

1 - Success.

-1 - If it is called in PowerBuilder or Appeon Web, or there is an error.

2.4.7 Media

Plays a media file using the device's media player.

2.4.7.1 eon_mobile_mediaex object

2.4.7.1.1 of_play function

Description

Uses the default media player application to play the media files.

The supported audio file formats are .aac, .mp3, .aiff, and .wav; the supported video file formats are .m4v, .mp4, and .mov.

Not all the media files with these suffixes can be played; if some media files cannot be played, they can be transcoded via iTunes.

Supported on mobile client only.

Syntax

of_play (value string *as_filepath*)

Argument	Description
as_filepath	Sets the full path of a media file to be played.

Return value

integer

1 - Play the media file successfully.

0 - Cancel playing the media file.

-1 - It is called in PowerBuilder or Appeon Web, or there is an error.

2.4.8 Notification

Sends a notification to the notification bar.

2.4.8.1 eon_mobile_notificationex object

2.4.8.1.1 of_addmessage function

Description

Sends a notification to the notification bar.

Supported on mobile client only.

Syntax

of_addmessage (value string *as_message*)

Return value

long

>0 - Return the current notification handle if sending notification successfully.

of_removemessage function can use this handle to delete the notification.

-1 - It is called in PowerBuilder or Appeon Web, or there is an error.

2.4.8.1.2 of_removemessage function

Description

Removes a notification from the notification bar.

Supported on mobile client only.

Syntax

of_removemessage (value long *al_handle*)

Argument	Description
al_handle	Specifies the handle of the notification, which is created by the of_addmessage() function.

Return value

integer

1 - Success.

-1 - It is called in PowerBuilder or Appeon Web, or there is an error.

2.4.9 Textchecker

Checks spellings in a text field.

2.4.9.1 eon_mobile_textcheckex object

2.4.9.1.1 of_getignoredwords function

Description

Returns the words that the receiver ignores when spell-checking.

The spell checker excludes ignored words as misspelled words during the current spell-checking session only.

Supported on mobile client only.

Syntax

of_getignoredwords (ref string *as_words[ ]*)

Argument	Description
as_words[ ]	Returns an array of strings, each of which specifies a word the receiver ignores.

Return value

integer

1 - Found any ignored words.

0 - Found no ignored words.

-1 - It is called in PowerBuilder or Appeon Web, or there is an error.

2.4.9.1.2 of_setignoredwords function

Description

Sets the list of words that the receiver should ignore, and the ignored words are not treated as misspelled words.

Supported on mobile client only.

Syntax

of_setignoredwords (value string *as_words[ ]*)

Argument	Description
as_words[ ]	Sets an array of strings, each of which specifies a word the receiver should ignore.

Return value

integer

1 - Success.

-1 - It is called in PowerBuilder or Appeon Web, or there is an error.

2.4.9.1.3 of_getlanguages function

Description

Gets the languages for which the text checker can perform spell-checking.

Supported on mobile client only.

Syntax

of_getlanguages (ref string *as_languages[ ]*)

Argument	Description
as_languages[ ]	Returns an array of strings representing ISO 639-1 language codes or combined ISO 639-1 language codes and ISO 3166-1 regional codes (for example, en_US).

Return value

integer

1 - Success.

-1 - It is called in PowerBuilder or Appeon Web, or there is an error.

2.4.9.1.4 of_getmisspelledword function

Description

Gets the range of the first misspelled word encountered.

Supported on mobile client only.

Syntax

of_getmisspelledword (value eon_mobile_str_textcheckoption astr_textcheckoption)

Sets the check option.

Variable list of the eon_mobile_str_textcheckoption struct

Data type	Argument	Description
string	s_source	The string which you want to check.
long	l_start	A long indicating where the check will begin in s_source.
long	l_length	Starting from the position specified by l_start, the number of characters needed to be checked.
string	s_language	The language of the words to be checked for correct spelling.

of_getmisspelledword (ref long *al_start*, long *al_length*)

Argument	Description
al_start	Returns a long whose value is the starting position of the first misspelled word.
al_length	Returns a long whose value is the length of the first misspelled word.

Return value

integer

1 - Found the misspelled word.

0 - Found no misspelled words.

-1 - It is called in PowerBuilder or Appeon Web, or there is an error.

2.4.9.1.5 of_guessesforword function

Description

Returns an array of strings that are possible completions for a partially entered word.

Supported on mobile client only.

Syntax

of_guessesforword (value eon_mobile_str_textcheckoption of_guessesforword)

Sets the check option.

Variable list of the eon_mobile_str_textcheckoption struct

Data type	Argument	Description
string	s_source	The string which you want to check.
long	l_start	A long indicating where the check will begin in s_source.
long	l_length	Starting from the position specified by l_start, the number of characters needed to be checked.
string	s_language	The language of the words to be checked for correct spelling.

of_guessesforword (ref string *as_guesses[ ])*

Argument	Description
as_guesses[ ]	Returns an array of strings each of which might be a correct substitute (that is, a guess) for a misspelled word in the given range of the string. If no possible guesses are found, the method returns an empty array.

Return value

integer

1 - Found possible guesses.

0 - Found no possible guesses.

-1 - It is called in PowerBuilder or Appeon Web, or there is an error.

2.4.9.1.6 of_haslearnedword function

Description

Detects if the text checker has learned the specified word.

Supported on mobile client only.

Syntax

of_haslearnedword (value string *as_word*)

Argument	Description
as_word	A string representing a word.

Return value

integer

1 - The text check has learned the word.

0 - The text check hasn't learned the word.

-1 - It is called in PowerBuilder or Appeon Web, or there is an error.

2.4.9.1.7 of_learnword function

Description

Tells the text checker to learn the specified word so that it is not evaluated as misspelled.

Supported on mobile client only.

Syntax

of_learnword (value string *as_word*)

Argument	Description
as_word	A string representing the word for the text checker to learn.

Return value

integer

1 - Success.

-1 - It is called in PowerBuilder or Appeon Web, or there is an error.

2.4.9.1.8 of_unlearnword function

Description

Tells the text checker to unlearn the specified word.

Supported on mobile client only.

Syntax

of_unlearnword (value string *as_word*)

Argument	Description
as_word	A string representing the word for the text checker to unlearn.

Return value

integer

1 - Success.

-1 - It is called in PowerBuilder or Appeon Web, or there is an error.

2.4.9.1.9 of_completionsforpartialword function

Description

Returns a list of words that are possible valid replacements for a misspelled word.

Supported on mobile client only.

Syntax

of_completionsforpartialword (value eon_mobile_str_textcheckoption astr_textcheckoption)

Sets the check option.

Variable list of the eon_mobile_str_textcheckoption struct

Data type	Argument	Description
string	s_source	The string which you want to check.
long	l_start	A long indicating where the check will begin in s_source.
long	l_length	Starting from the position specified by l_start, the number of characters needed to be checked.
string	s_language	The language of the words to be checked for correct spelling.

of_completionsforpartialword (ref string *as_completions[ ]*)

Argument	Description
as_completions[ ]	Returns an array of strings, each of which is a completion of a partially entered word represented by range in string. If no possible completions are found, the method returns an empty array.

Return value

integer

1 - Found possible completions.

0 - Found no possible completions.

-1 - It is called in PowerBuilder or Appeon Web, or there is an error.

3 Workarounds for Unsupported Features

Not all of the PowerBuilder features can be supported by Appeon. The unsupported features, if not modified, will be commented out in the generated application files, as a result, the code that contains the unsupported features and other code that is dependent on those unsupported features will stop working.

This chapter provides suggestions to work around the unsupported features that have functional impact on the running of the application. Some cosmetic features, such as BorderStyle=StyleShadowBox! property, can be simply ignored if they will not affect the application.

3.1 Objects & Controls

3.1.1 External user object

Description

The external user object is unsupported.

Workaround

Choose one of the following methods:

Method #1: Encapsulate the functions using DLLs.

Method #2: Encapsulate the functions using a user-defined OCX.

3.1.2 UserObject object

3.1.2.1 AddItem function (EAServer only)

Description

AddItem function is unsupported in a user defined object.

Workaround

Call the AddItem function of controls such as ListBox, DropDownListBox, PictureListBox, DropDownPictureListBox, or ListView.

3.1.2.2 DeleteItem function

Description

DeleteItem function is unsupported in a user defined object.

Workaround

Call the DeleteItem function of controls such as ListBox, DropDownListBox, PictureListBox, DropDownPictureListBox, or ListView instead of calling the DeleteItem function of a user-defined object.

3.1.3 System Objects

3.1.3.1 PipeLine object

Description

The Pipeline object including its properties, functions and events is unsupported.

Workaround

Add a server NVO and deploy it to EAServer to work around this issue. The detailed steps are:

Step 1 - Encapsulate the Pipeline object including its properties, functions and events in a server NVO, and deploy the NVO to EAServer component.

Step 2 - Call the interface function of the component and pass the Pipeline object name and relevant parameters to it.

Step 3 - Call the functions, properties or events of the Pipeline object in the component to implement the same functionalities as the original PipeLine object.

The limitation of this workaround is that only the system Pipeline object can be used in the NVO. That is to say, user-defined objects inherited from Pipeline objects are unsupported in the NVO.

3.1.3.2 Application object

3.1.3.2.1 DWMessageTitle property

Description

The DWMessageTitle is unsupported.

Workaround

Use a global variable to record the value of the DWMessageTitle property. Read and write the global variable instead of using the DWMessageTitle property in the Script.

3.1.3.2.2 FreeDBLibraries property

Description

The FreeDBLibraries property is unsupported.

Workaround

It is a useless property for a Web based application. The property can be simply commented out causing no functionality difference for the deployed application.

3.1.4 System Controls

3.1.4.1 ListView control

3.1.4.1.1 GetItemAtPointer function

Description

GetItemAtPoint is unsupported for ListView control.

Workaround

Replace the use of GetItemAtPointer by using Index argument of RightClicked since they return the same value.

3.1.4.1.2 ItemActivate event

Description

The ItemActivate event for ListView control is unsupported.

Workaround

Copy the script in the ItemActivate event to the Clicked or DoubleClicked event.

3.1.4.2 TreeView control

3.1.4.2.1 SetDropHighlight function

Description

The SetDropHightlight function for TreeView control is unsupported.

Workaround

Use other functions to highlight the item specified in SetDropHighlight as the drop target. For example, change the font of the item or modify the item label.

3.1.4.3 Tab control

3.1.4.3.1 TabPostEvent function

Description

TabPostEvent function of tab control is unsupported.

Workaround

Use TabTriggerEvent instead.

3.2 System Functions

3.2.1 DDE Server functions

Description

The DDE Server functions are unsupported.

Workaround

Use a DLL to work around this issue.

1) Encapsulate the DDE in a DLL.

2) Create a DLL proxy to call the DDE in the DLL.

3.2.2 FileLength64 function

Description

The FileLength64 function is unsupported.

Workaround

Replace the FileLength64 function with FileLength function.

3.2.3 FileReadEx function

The FileReadEx function is supported since Appeon 6.5.

Description

The FileReadEx function is unsupported in Appeon 5.x, 6.0, 6.1 and 6.2.

Workaround

Replace the FileReadEx function with FileRead function.

3.2.4 FileSeek64 function

Description

The FileSeek64 function is unsupported.

Workaround

Replace the FileSeek64 function with FileSeek function.

3.2.5 FileWriteEx function

The FileWriteEx function is supported since Appeon 6.5.

Description

The FileWriteEx function is unsupported in Appeon 5.x, 6.0, 6.1 and 6.2.

Workaround

Replace the FileWriteEx function with FileWrite function.

3.2.6 FromAnsi function

Description

The FromAnsi function is unsupported.

Workaround

Replace the FromAnsi function with String function.

Example

The original code:

```
FromAnsi ( blob )
```

The modified code in PowerBuilder 9:

```
String(blob)
```

The modified code in PowerBuilder 10:

```
String(blob, EncodingANSI!)
```

3.2.7 FromUnicode function

Description

The FromUnicode function is unsupported.

Workaround

Replace the FromUnicode function with String function.

Exmaple

The original code:

```
FromUnicode( blob )
```

The modified code:

```
String(blob)
```

3.2.8 Garbage Collection functions

Description

The Garbage Collection functions (GarbageCollect, GarbageCollectGetTimeLimit & GarbageCollectSetTimeLimit) are unsupported.

Workaround

Appeon has its own mechanism to delete unused objects and classes. This function can be simply commented out, causing no loss to the deployed application.

3.2.9 GetContextService function

Description

The GetContextService is unsupported.

Workaround

If the GetContextService is used to create a reference to an Internet service that calls the HyperlinkToURL function, the workaround is to comment out the call to GetContextService, instead, directly call the HyperlinkToURL function. The reason is, Appeon supports HyperlinkToURL as a system function, and does not require creating an Inet service before calling the HyperlinkToURL function.

Example

The original script:

```
GetContextService("Internet", iinet_base)
iinet_base.HyperlinkToURL("http://www.appeon.com")
```

The modified script:

```
HyperlinkToURL("http://www.appeon.com")
```

3.2.10 GetLibraryList function

Description

The GetLibraryList function is unsupported.

Workaround

Create a new table holding all the library information in the database. Use this table to check the information in the library instead of using the GetLibraryList function.

3.2.11 Help functions

Description

Using the Help functions such as ShowHelp or ShowPopupHelp to display a HLP file or a CHM (HTML Help) file is unsupported.

Workaround

To enable you to display a HLP file or a CHM file, replace the Help functions by applying aStaticHyperLink, PictureHyperLink or HyperLinkToURL:

Step 1 - Upload the Help file to your Web Server.

Step 2 - Create a StaticHyperLink or PictureHyperLink control in the application and set the link of the control to the URL of the Help file. You can also apply the HyperLinkToURL function in the application for accessing the URL of the Help file.

Example

This example uses Appeon Help as the HLP file. First upload the HLP file to the Web Server (in this example Appeon Help is located in http://www.appeon.com/support).

Workaround #1: Open the HLP or CHM file with a StaticHyperLink control.

Create a StaticHyperLink control in the application. Add the Appeon Help URL (http://www.appeon.com/support) to the URL property of StaticHyperLink control.

Workaround #2: Apply the HyperLinkToURL function to access the URL of the Help file.

Add the following code to the relevant location in the application:

```
Inet internet
Internet = create Inet
internet.HyperLinktoURL("http://www.appeon.net/support/manuals")
```

3.2.12 HyperLinkToURL function

Description

While the HyperLinkToURL function is supported, it cannot link to the URL of an Appeon application. This is because an Internet Explorer process only supports one Appeon application at a time. Executing the HyperLinkToURL function from one Appeon application to open another will cause Internet Explorer to shut down.

Workaround

Method 1: Using the Run function to open the second application

Step 1 - Replace the call to the HyperLinkToURL function with a call to the Run system function. Comment out the script in the HyperLinkToURL button and add the following script:

```
run("IEXPLORE.EXE http://192.0.2.189:8080/b237293/",Maximized!);
```

In the code example, "http://192.0.2.189:8080/b237293" can be replaced with the URL of an Appeon application.

Step 2 - At the Web server, copy the IEXPRLORE.EXE to C:\WINNT\system or C:\WINNT \system32. Alternatively, add the directory "C:\Program Files\Internet Explorer" in the PATH variable.

Method 2: Using the PictureHyperLink URL property to open the second application

Step 1 - Add a PictureHyperLink control to the first application, from which the second application will be opened.

Step 2 - Specify the URL of the second application as the URL of the PictureHyperLink control.

3.2.13 LibraryDirectory function

Description

The LibraryDirectory PowerScript function is unsupported.

Workaround

The LibraryDirectory function is used to get a list of all objects or just objects of a specified type in a PowerBuilder library.

Case #1

If the LibraryDirectory function is used in other unsupported Library functions, there is no workaround available. Comment out the script related to the Library functions and the relevant functionality will be lost.

Case #2

If the LibraryDirectory function is used to get the list of all objects in the PowerBuilder Library and check whether a particular object exists, the workaround steps are:

Step 1 - Create a table in the database to store the object names of the PowerBuilder library that is passed to the LibraryDirectory function.

Step 2 - Use a SQL statement to check whether a particular object exists in the table.

3.2.14 PopulateError function

Description

The PopulateError function is unsupported.

Workaround

Appeon has its own error-handling mechanism. This function can be directly commented out, causing no loss to the deployed application.

3.2.15 Registry function

Description

System registry functions can read and write registry entries, keys, and values on a Windows PC. But unlike Windows system, iOS system has no such registry, so these functions cannot be directly executed in iOS system.

Workaround

Appeon offers an option of "Appeon emulation registry" in AEM to allow the mobile application to read and write the mock registry file stored in the Appeon Server database. For detailed information, please refer to **Appeon Server Configuration Guide for .NET | AEM User Guide | Application | PB Features | Registry Mode**.

Note: Mobile applications can only read Appeon emulation registry.

3.2.16 Shared Object functions

Description

Shared Object functions like SharedObjectGet, SharedObjectRegister are unsupported by Appeon, so they cannot be applied to show the process of an event with a progress bar, or control the progress of two different processes.

Workaround

The following is the workaround for showing the process of an event to the users:

Before the execution of the event, prompt a message box at the client to inform the user that the event (such as the retrieve of a large amount of data) is taking place and the event may take relatively long.

3.2.17 SignalError function

Description

The SignalError function is unsupported.

Workaround

Appeon has its own error-handling mechanism. This function can be directly commented out, causing no functionality loss to the deployed application.

3.3 PowerScript Reference

3.3.1 PowerScript Topics

3.3.1.1 Calling functions and events

3.3.1.1.1 Overriding system function

Description

Overriding system functions is unsupported.

Workaround

Create a user-defined function and rewrite the user code in the system function into the user-defined function.

Example

In the original application, the SetTransObject function of the *u_dw* object contains the following user code:

```
trnas_current = atrans_current return super::settransobject(itrans_current)
```

In the converted Web application, remove the preceding user code from the SetTransObject function, then create a user-defined function *uf_settransobject* and rewrite the following code:

```
itrnas_current = atrans_current settransobject(itrans_current)
```

When the SetTransObject function of the *u_dw* object is called in the Web application, the user code is executed in the newly created *uf_settransobject* function.

3.3.1.1.2 Passing arguments to functions and events

Duplicate arguments for a function

Description

Repetitively referring objects as arguments for a function is unsupported. Use the following workaround and example to change it into a supported format.

Workaround

Assign the repetitively referenced object to multiple different variables and pass these variables to a function.

Example

The original script:

```
w_1.wf_1(dw_1,dw_1)
```

Re-write it using the following format:

```
u_dw ldw_1, ldw_2
ldw_1 = dw_1
ldw_2 = dw_1
w_1.wf_1(ldw_1,ldw_2)
```

Passing Menu object as a reference parameter

Description

Passing Menu object as a reference parameter is unsupported.

Workaround

Pass the Menu object by value or as read-only instead of by reference. The reason is, when passing by value or as read-only, if you change the properties of the object by value or as read-only, you are changing the original object, which is the same as passing by reference.

Reference parameter

Description

The return values of functions and properties of objects cannot be directly used as reference parameters for functions.

Workaround

Follow the steps below to work around this issue:

Step 1 - Create a variable and assign the return value of a function or the property of an object to the variable.

Step 2 - Call the variable where the return value or the property is called in the original code.

Step 3 - Assign the return value of calling the variable to the original property or object.

Code example

```
//The original code
poptags(theobject.item[theitem])
//After modification
menu lm_menu
lm_menu = theobject.item[theitem]
poptags(lm_menu)
```

3.3.1.2 Declarations

3.3.1.2.1 Shared variables

Description

Shared variables are unsupported. For example, In the pfcmain.pbl of a PFC application, the pfc_u_dw uses a shared variable snv_property with its type being n_cst_dwsrv_property. The snv_property is unsupported.

Workaround

Follow the steps below to work around this issue.

Step 1 - Change the variable snv_property as an instance variable of n_cst_appmanager (usually declared as gnv).

Step 2 - Open the pfc_u_dw, and replace all the "snv_property" with the "gnv_app.snv_property".

Note

Since the parent code is changed, when deploying a new PFC, the same modification needs to be done again.

Other shared variables can also be worked around in this way.

3.3.1.3 Language basics

3.3.1.3.1 Null values

Description

The Null value calculation in Appeon is quite different from that in PowerBuilder. This is caused by the different calculation methods used in PowerScript and JavaScript.

Workaround

When using expressions with Null value calculations, you should add conditional statements to ensure that you get the correct result.

Example #1

A and/or B is/are likely to carry a Null value in the following assignment:

```
C = A + B
```

Re-write it using the following format:

```
IF IsNull(A) or IsNull(B) THEN
 SetNull(C)
ELSE
 C = A + B
END IF
```

Example #2

Another example for relational operations:

```
IF A = B then
 MessageBox ("Return Value","True")
ELSE
 MessageBox ("Return Value","False")
END IF
```

Re-write it using the following format:

```
IF IsNull(A) or IsNull(B) then
 MessageBox ("Return Value", "False")
ELSEIF A = B then
 MessageBox ("Return Value", "True")
ELSE
 MessageBox ("Return Value", "False")
END IF
```

Note

The following table shows the different return values that PowerBuilder and Appeon will produce when an expression contains at least one null value. In these examples, the values of variables A and B are both null:

Table 3.1:

Expressions	Return Value in PowerBuilder	Return Value in Appeon
A+1	Null	1
A+B	Null	Null
A*B	Null	0
A=1 (relational)	Null	False
A<>1	Null	True
NOT (A=1)	Null	True
A=A (relational)	Null	True
A=B	Null	True
IsNull(A=1)	True	False

3.3.2 PowerScript Statements

3.3.2.1 GOTO statement

Description

PowerScript GOTO statements and Label are unsupported. Using GOTO is not recommended in structured programming.

Workaround

Analyze the code that uses GOTO and re-write the code in a structured way by applying IF ... THEN ... statements.

If the statement that the GOTO label is associated with has a return value, place the statement in a user function, and in place of the GOTO statement, call the user function.

If the statement that the GOTO label is associated with has no return value, place the statement directly in place of the GOTO statement.

Example

The original script:

```
IF sle_1.text = "" THEN GOTO hide_sle_1
sle_1.text = ""
hide_sle_1:
sle_1.visible = false
MessageBox ("","SingleLineEdit sle_1 is cleared and hidden.")
```

Re-write it using IF ... THEN... statement:

```
IF sle_1.text = "" THEN
sle_1.visible = false
MessageBox ("","SingleLineEdit sle_1 is cleared and hidden.")
ELSE
sle_1.text = ""
sle_1.visible = false
MessageBox ("","SingleLineEdit sle_1 is cleared and hidden.")
END IF
```

Re-write it using CHOOSE CASE statement:

```
CHOOSE CASE sle_1.text
CASE ""
sle_1.visible = false
MessageBox ("","SingleLineEdit sle_1 is cleared and hidden.")
CASE ELSE
sle_1.text = ""
sle_1.visible = false
MessageBox ("","SingleLineEdit sle_1 is cleared and hidden.")
END CHOOSE
```

3.3.3 SQL Statements

3.3.3.1 Stored procedure with Null output

Description

It is currently unsupported to use NULL value in an OUTPUT parameter when declaring a stored procedure.

Workaround

Set a variable and set it to NULL. Then use the variable in the OUTPUT parameter to provide the same functionality.

Example

Original code

```
// Declare the procedures
Declare SPgetseniva procedure for SPgetseniva &
@codpes = :l_usp_codpes, &
@nomsis = :s_usp_systemname, &
@staobt = :s_usp_systemstatus,&
@numseniva = null output ;
```

Modified code

```
integer li_return
SetNull(li_return)
// Declare the procedures
Declare SPgetseniva procedure for SPgetseniva &
@codpes = :l_usp_codpes, &
@nomsis = :s_usp_systemname, &
@staobt = :s_usp_systemstatus,&
@numseniva = :li_return output ;
```

3.3.4 PowerScript Events

3.3.4.1 Help event

Description

The Help event is unsupported.

Workaround

Copy the script of the Help event to the Key event. In the Key event, use the KeyDown function to check whether the user has pressed the F1 key and if the KeyDown function returns true, execute the script that is originally in the Help event.

3.3.4.2 Other event

Description

The Other event is unsupported.

Workaround

Move the script in the Other event to the supported events.

3.4 DataWindow Reference

3.4.1 DataWindow operators

3.4.1.1 DataWindow operator precedence

Description

The AND and OR operators in a DataWindow expression have the same precedence in PowerBuilder, but in Appeon, the AND operator has higher precedence.

Workaround

When there are both AND and OR operators in a DataWindow expression, you should use parentheses to get the correct precedence effect.

Example

The following code examples are for a DataWindow expression that sets the column text color. The OR operator will be evaluated first in PowerBuilder, but in Appeon, the following script will evaluate the AND operator first.

Original code:

```
If(Left(GetText(), 1) = 'V' OR Left(GetText(), 1) = 'A' AND Mod(GetRow(), 2) = 1, 236, 243433) //
```

To have the OR operator evaluated first, add a pair of parentheses to the OR expression:

```
If((Left(GetText(), 1) = 'V' OR Left(GetText(), 1) = 'A') AND Mod(GetRow(), 2) = 1, 236, 243433)
```

3.4.2 DataWindow Object Properties

3.4.2.1 Retrieve.AsNeeded

Description

The Retrieve.AsNeeded property of DataWindow object is unsupported.

Workaround

Write script to have the data displayed in batches instead of at one time. The detailed steps are:

Step 1 - Create a user object. The user object contains four Picture buttons. The Picture buttons represents First page, Previous page, Next page and Last page respectively.

Step 2 - Place the user object in the window that contains the DataWindow in which the large quantity of data will be displayed.

Step 3 - Define window functions and modify the original scripts to have them work along with the user object to provide the functionality of displaying data in batches.

3.4.3 DataWindow control

3.4.3.1 Functions

3.4.3.1.1 CanUndo function

Description

CanUndo function is unsupported.

Workaround

Replace the CanUndo function with ModifiedCount function.

Example

The original code:

```
if dw_1.CanUndo() Then
dw_1.Undo()
end if
```

The modified code:

```
if dw_1.ModifiedCount() > 0 Then
dw_1.Undo()
End if
```

3.4.3.1.2 DBCancel function

Description

DBCancel function is unsupported.

Workaround

Appeon has its own mechanism to retrieval data. This function can be simply commented out.

Functionality difference

All the retrieved data will be returned at the same time.

3.4.3.1.3 GenerateResultSet function

Description

The ResultSet system object and GenerateResultSet method are unsupported.

Workaround

To work around the GenerateResultSet method, we have the following two methods:

Method #1: Use GetFullState to retrieve data from a DataWindow and then use SetFullState to apply the blob returned from GetFullState to another DataWindow.

Method #2: Use the datawindow.data property to retrieve data from a DataWindow into a string and insert data into the DataWindow from the string by ImportString.

3.4.3.1.4 GetTrans function

Description

The DataWindow GetTrans function is unsupported.

Workaround

Comment out the unsupported script. Instead use the SetTransObject connection method, to assign a programmer-specified transaction object or a global transaction object called SQLCA to a DataWindow control or DataStore.

To use SetTransObject, write code that performs the following tasks:

1. Set up the transaction object by assigning values to its fields (usually in the application's Open event).

2. Connect to the database using the SQL CONNECT statement and the transaction object (in the Open event for the application or window).

3. Call SetTransObject to associate the transaction object with the DataWindow control or DataStore (usually in the window's Open event).

4. Check the return value from the Update method and follow it with an SQL COMMIT or ROLLBACK statement, as appropriate.

3.4.3.1.5 ReselectRow function

Description

The DataWindow ReselectRow method is unsupported.

Workaround #1

If the DataWindow's source table has a primary key, to work around this issue, follow the steps below:

Step 1 - Use GetItem() to get the value of the primary column in the current row.

Step 2 - Use SQL statement to retrieve data for the current row according to the value of the primary key.

Step 3 - Use SetItem() to assign values to each column in the current row.

Step 4 - Change the status of the current row to "NotModified!"

Workaround #2

Replace the ReselectRow function with the Retrieve function.

3.4.3.1.6 ResetTransObject function

Description

The ResetTransObject function of a DataWindow control or a DataStore is unsupported.

Workaround

Replace the ResetTransObject function with SetTransObject function.

Example

The original script:

```
dw_1.ResetTransObject()
```

The modified script:

```
dw_1.SetTransObject(transaction)
```

3.4.3.1.7 SaveAsAscii function

Description

SaveAsAscii function for DataWindow is unsupported.

Workaround

Use a server NVO to work around the SaveAsAscii function.

Step 1 - Encapsulate the SaveAsAscii function into a NVO and deploy the NVO to EAServer.

Step 2 - Call to the NVO SaveAsAscii function, and store the generated ASCII text file to the Web Server web root.

Step 3 - Send the URL of the generated files to client side using the HyperLinkToURL function of Inet object.

Note

The DataWindow that calls the SaveAsASCII must be deployed to EAServer as well.

All the DataWindow SaveAs types that are unsupported can be worked around in this way.

3.4.3.1.8 Scroll function

Description

The Scroll function of DataWindow control is unsupported.

Workaround

Replace the Scroll function with ScrollToRow, ScrollPriorPage or ScrollNextRow function.

Functionality difference

The event sequence of the corresponding function will be triggered.

Example

The original script:

```
dw_1.Scroll(3)
```

The modified script. Please note that the event sequence of the ScrollToRow function will be triggered.

```
dw_1.ScrollToRow(dw_1.GetRow() + 3)
```

3.4.3.1.9 SetTrans function

Description

Using SetTrans to establish a connection to the transaction object is unsupported.

Workaround

Use the SetTransObject method.

Example

The original script:

```
i = ids_main.SetTrans(itr_sql)
```

Re-write it using the following format:

```
ids_main.setTransObject(itr_sql)
```

3.4.3.2 Events

3.4.3.2.1 ScrollVertical event

Description

ScrollVertical event is unsupported.

Workaround

You can choose either of the following two ways to work around the ScrollVertical event.

To work around the ScrollVertical event for a single DataWindow

Step 1 - Define a user-defined event such as ue_scrollvertical to replace ScrollVertical.

Step 2 - Place the same code that you plan to put in ScrollVertical in the user-defined event.

Step 3 - In the Timer event of the window that hosts the DataWindow, trigger the user-defined event periodically with the following code:

```
long ll_new_firstrow
ll_new_firstrow = long(dw_2.Object.DataWindow.firstRowOnPage) //Assuming the DataWindow is dw_2
if ll_new_firstrow <> il_old_first_row then
 dw_2.trigger event ue_scrollvertical()
 il_old_first_row = ll_new_firstrow
end if
```

Step 4 - Call the Timer event in the Open event of the window with the following code:

```
timer(0.005)
```

To work around the ScrollVertical event for ancestor DataWindow

You can code the workaround once in the ancestor DataWindow and then apply it for as many DataWindow as you want.

Supposing the ancestor window is w_sheet and the ancestor DataWindow is u_dw:

Step 1 - Start the Timer event in the Open event of w_sheet.

```
string ls_timer_interval
ls_timer_interval = &
ProfileString( gnv_app.of_getappinifile( ), "timer", "interval", "0.005" );
timer(Dec(ls_timer_interval));
```

Step 2 - In the pfc_postopen event of w_sheet, add the following code for getting all the DataWindows.

of_sb_get_dwobjects(this.control);

Step 3 - Add a new function called of_sb_get_dwobjects (windowobject awo_control[]) in w_sheet.

```
int        i;
tab        lt_tab;
userobject luo_temp;
u_dw       lu_dw

for i = 1 to upperbound( awo_control )
 if( TypeOf( awo_control[i] ) = Tab! ) then
  lt_tab = awo_control[i];
  of_sb_get_dwobjects( lt_tab.control ); // Recursive call
 elseif( TypeOf( awo_control[i] ) = UserObject! ) then
  luo_temp = awo_control[i];
  of_sb_get_dwobjects( luo_temp.control ); // Recursive call
 elseif( TypeOf( awo_control[i] ) = DataWindow! ) then
  // iu_dw is an instance variable that is an array of u_dw datawindow controls
  iu_dw[ upperbound(iu_dw) + 1 ] = awo_control[i];
 end if
next
```

Step 4 - In the Timer event for w_sheet, add the following code to check row changes happened to all the DataWindows:

```
int i

for i = 1 to upperbound(iu_dw)
 iu_dw[i].of_sb_verticalscroll()
next
```

Step 5 - Add an instance variable for u_dw.

```
long il_old_first_row = -1;
```

Step 6 - Add a new function of_sb_verticalscroll( ) for u_dw.

```
long ll_new_firstrow, ll_counter // Check if only one row per page is being displayed in dw
ll_new_firstrow = long( this.object.DataWindow.FirstRowOnPage );
if (ll_new_firstrow <> il_old_first_row ) then
 il_old_first_row = ll_new_firstrow
 this.trigger event ue_scrollvertical()
end if;
```

Step 7 - Define a user-defined event for u_dw such as ue_scrollvertical to replace ScrollVertical.

Step 8 - Place the same code that you plan to put in ScrollVertical in the user-defined event.

3.4.3.2.2 RetrieveRow event

Description

The RetrieveRow event is unsupported.

Workaround

Move the relevant logic to the RetrieveEnd event.

Example

The original script in the RetrieveRow event:

```
if row < 1 then return
//for every single time, check whether the row should be deleted or not.
if f_find(istr_dwnum.dw,istr_dwnum.id+"='"+this.getitemstring(row,istr_dwnum.id)+"'")>0 then
 this.deleterow(row)
end if
```

The modified RetrieveEnd event (preceding logic is moved to the RetrieveEnd event):

```
long li_row
//loop all rows retrieved from the database and find out which row should be deleted.
for li_row = 1 to rowcount
 if f_find(istr_dwnum.dw,istr_dwnum.id+"='"+this.getitemstring(li_row,istr_dwnum.id
 +"'")>0 then
   this.deleterow(li_row)
     li_row --
 end if
next
```

3.4.3.3 Properties

3.4.3.3.1 LiveScroll property

Description

The LiveScroll property for DataWindow control is unsupported.

Workaround

When the LiveScroll property is enabled, it does not take effect on the Web. A row can only be selected by mouse clicking.

3.4.4 Controls in a DataWindow

3.4.4.1 Large Binary/Text database OLE object

Description

The Large Binary/Text database OLE object for DataWindow is unsupported.

Workaround

This workaround only applies to the BitMap OLE object.

If a column with any large Binary/Text Database OLE object is used for displaying graphs, do the following steps to work around this issue:

Step 1 - Replace the column with a Graph control.

Step 2 - Retrieve the content of the column in the table related with the DataWindow using a SELECTBLOB SQL statement.

Step 3 - Call the SetPicture function of this Graph control.

3.4.4.2 Column control

3.4.4.2.1 Char data type column

Description

When you set the data type of a column as *char* with a specified length, the value of the column data type retrieved by Appeon is different from that in PowerBuilder. The value you get in Appeon is *char*, but the value you get in PowerBuilder is *char(n)*.

Workaround

Add one more condition while using the returned value.

Example

The original script:

```
string ls_datatype
ls_datatype = dw_1.object.group_id.coltype
if(ls_datatype = “char(50)”) then
  ...
else
  ...
end if
```

Add one more conditional statement, as shown in the following modified script.

The modified script:

```
string ls_datatype
ls_datatype = dw_1.object.group_id.coltype
if(ls_datatype = “char(50)” or ls_datatype = "char") then
  ...
else
  ...
end if
```

3.4.4.2.2 Evaluating DataWindow expressions in scripts

Description

When using global functions in DataWindow expressions to dynamically change the attributes of DataWindow objects at run time, this method doesn't work fine with an Appeon application, for the DataWindow expressions are only evaluated once.

Workaround

Modify a computed expression on the DataWindow in order to force the expression to re-evalute.

Note: Generally speaking, DataWindow expressions wil slow-down the initial display or subsequent refresh of DataWndows. As such, Appeon recommend you reduce the usage of DataWindow expressions if possible, especially in the following situations:

- Avoid using DataWindow expressions for computing and setting column properties.

- Avoid setting sort and filter criteria directly for a DataWindow object. Instead, write the sort and filter criteria in the SQL statement of the DataWindow object.As noted previously, it is faster to use SQL statements than DataWindow functionality.

Example

An expression like this will not re-evaluate itself: Expression: f_color()

After making a change that would cause f_color() to return a different value (i.e. selecting a different preferred color from a drop-down), the application has to slightly change the size of the DataWindow in order to force the expression to re-evaluate.

The modified script:

```
integer li_dw_width, li_dw_height
li_dw_width=dw_1.width
li_dw_height=dw_1.height

dw_1.width=li_dw_width-4         //Squeeze the DW
dw_1.height=li_dw_height-4

dw_1.width=li_dw_width         //Restore to original size
dw_1.height=li_dw_heigh
```

3.4.5 Functions of DataStore object

Description

CategoryCount,Clipboard, CopyRTF, DataCount for DataStore, FindCategory, FindSeries, GetBorderStyle, GetData, GetDataPieExplode, GetDataStyle, GetDataValue, GetSeriesStyle, PrintCancel, ResetDataColors, SeriesCount, SeriesName, SetDataPieExplode, SetDataStyle, SetDetailHeight, SetSeriesStyle.

Workaround

Replace the DataStore with a DataWindow control and call the corresponding functions of the DataWindow control.

3.5 DBParm parameters in Database

3.5.1 ConnectString parameter

Description

The ConnecString parameter is unsupported.

Workaround

Connect to the database via JDBC, and specify the Data Source name in AEM or dynamically set up the database connection. For more details, please refer to the "Database Connection Setup" section in *Server Configuration Guide*.

3.6 Workaround Techniques (for EAServer only)

3.6.1 Appeon GetFullState/SetFullState/GetChanges/SetChanges (Windows only)

Description

PowerBuilder provides four functions for synchronizing DataWindows and DataStores in a distributed application:

- GetFullState
- SetFullState
- GetChanges
- SetChanges

These four functions use a BLOB (Binary Large Object) parameter to store the state information of a DataWindow or DataStore, but Web applications that use JavaScript, such as Appeon, do not support the BLOB data type.

Workaround

To work around the unsupported features (the BLOB parameter) in GetFullState, SetFullState, GetChanges and SetChanges, Appeon has provided two standard user objects: the appeondatawindow that inherits from the PowerBuilder system DataWindow control, and the appeondatastore that inherits from the PowerBuilder system DataStore object.

These two objects have user-defined functions (GetFullState, SetFullState, GetChanges and SetChanges) that override the original DataWindow/DataStore GetFullState, SetFullState, GetChanges and SetChanges, with a parameter of the String data type.

When coding distributed DataWindow/DataStores for Appeon Web migration, always use the descendants of appeondatawindow and appeondatastore, and the overriding GetFullState, SetFullState, GetChanges and SetChanges functions that take the String data type as the parameter.

Here are detailed steps on how to perform this workaround.

Example

The following workaround for the GetFullState, SetFullState, GetChanges and SetChanges functions used in the application assumes that the distributed application to be migrated is *application_distribute*.

Step 1 -Build a new application called application_datastore in PowerBuilder to include server DataStores in a*pplication_distribute.*

Step 2 -Add the *appeon_workarounds.pbl* file to the Library Search Path of *application_datastore*.

Step 3-Copy the EonAXNVO.dll from the %AppeonDeveloper%\appeon_workarounds**xxx**e directory (**xxx** indicates the corresponding PowerBuilder version) to the same directory as *application_distribute*.

Step 4 -Search in *application_distribute* for the DataWindow controls that use GetFullState, SetFullState, GetChanges or SetChanges. Change the declarations of these DataWindows to make them inherit from the appeondatawindow user object.

In the following example, the unsupported SetFullState function is called in the Constructor event of the dw_1 DataWindow in the w_main Window:

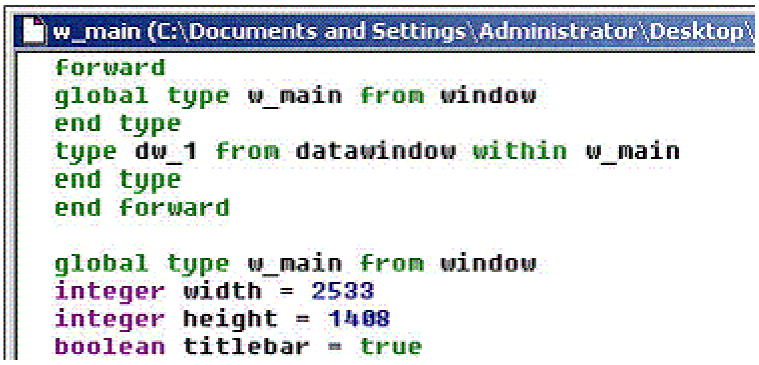
```
long ll_rc
blob lbob_1
nvo_update lnvo_1
ll_rc = myconnect.CreateInstance(lnvo_1,"p_nvo_update1/nvo_update")
IF ll_rc <> 0 THEN
  MessageBox("Create instance failed", ll_rc)
END IF
lbob_1 = lnvo_1.of_getdata1("d_order_master_four")
IF not IsNull(lbob_1) THEN
  dw_1.SetFullState(lbob_1)
ELSE
  MessageBox("", "Getting data failed.")
END IF
```

To re-write the previous unsupported code:

Change the declaration of dw_1 DataWindow by editing the source code of the *w_main* Window.

Right-click on *w_main* in the System Tree and choose Edit Source from the context menu. The Source Editor window opens and displays the source code of the *w_main* Window.

Figure 3.1:



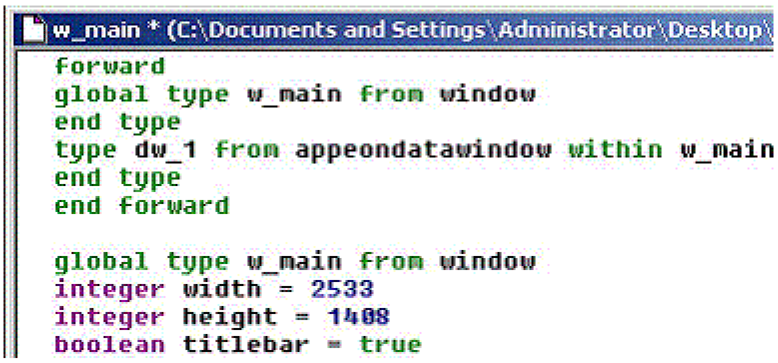
```
forward
global type w_main from window
end type
type dw_1 from datawindow within w_main
end type
end forward

global type w_main from window
integer width = 2533
integer height = 1408
boolean titlebar = true
```

Replace the script “type *dw_1* from datawindow within *w_main*” with “type *dw_1* from appeondatawindow within *w_main*”.

Save the source code and close the Source Editor window for the *w_main* Window object.

Figure 3.2:



```
forward
global type w_main from window
end type
type dw_1 from appeondatawindow within w_main
end type
end forward

global type w_main from window
integer width = 2533
integer height = 1408
boolean titlebar = true
```

Step 5-Search in appeon_datastore for the DataStore objects that use GetFullState, SetFullState, GetChanges or SetChanges. Change the declarations of these DataStores to make them inherit from the appeondatastore user object.

The following is the script for the *of_getdata1* function of the *nvo_update* user object that is called in Step 3. The function creates a DataStore, retrieves data, and uses the unsupported GetFullState function to get and return the state of the DataStore:

```
public function blob of_getdata1 (string as_dataobject);
blob lbob_1
long ll_return
datastore lds_1
lds_1 = Create datastore
lds_1.DataObject = as_dataobject
lds_1.SetTransObject(sqlca)
lds_1.Retrieve()
ll_return = lds_1.GetFullState(lbob_1)
IF IsNull(ll_return) or ll_return < 0 THEN
  SetNull(lbob_1)
END IF
Destroy lds_1
Return lbob_1
end function
```

Modify the function return value to be the String data type, and re-declare the *lds_1* DataStore as appeondatastore. The changed code is shown as follows:

```
public function string of_getdata1 (string as_dataobject);
string ls_fullstate
long ll_return
appeondatastore lds_1
lds_1 = Create appeondatastore
lds_1.DataObject = as_dataobject
lds_1.SetTransObject(sqlca)
lds_1.Retrieve()
ll_return = lds_1.GetFullState(ls_fullstate)
IF IsNull(ll_return) or ll_return < 0 THEN
  SetNull(ls_fullstate)
END IF
Destroy lds_1
Return ls_fullstate
end function
```

After applying this workaround, migrate the entire application to the Web using Appeon. For detailed steps, refer to the "*Migrating distributed applications with distributed DataWindows*" section in the *Appeon Migration Guide*.

3.6.2 Packaging unsupported features in n-Tier NVOs or DLLs

3.6.2.1 Packaging unsupported objects into n-Tier NVOs

Description

The objects shown in the following table are unsupported, but can be supported if they are packaged into n-Tier NVOs.

Table 3.2:

ADOResultSet	ArrayBounds	ClassDefinition
ClassDefinitionObject	ConnectionInfo	ConnectObject
CORBABadTypeContext	CORBABadInvorder	CORBABadOperation
CORBABadParam	CORBABadTypeCode	CORBACommFailure

CORBACurrent	CORBADataConversion	CORBAFreeMem
CORBAImpLimit	CORBAInitialize	CORBAInternal
CORBAIntFrePos	CORBAInvalidTransaction	CORBAInvFlag
CORBAInvIdentInvOBJRef	CORBAMarshal	CORBANoImplement
CORBANoMemory	CORBANoPermission	CORBANoResources
CORBANoResponse	CORBAOBJAdapter	CORBAObjectNoTexist
CORBAPersistStore	CORBASystemException	CORBATransactionRequired
CORBATransactionRolledback	CORBATranslent	CORBAUnion
CORBAUnknown	CORBAUserException	CPlusPlus
DivideByZeroError	ErrorLogging	Exception
JaguarORB	mailFileDescription	mailMessage
mailRecipient	mailSession	OLECustomControl
OLERuntimeError	OLEStorage	OLEStream
OLETxnObject	OMObject	OMStorage
OMStream	ORB	PBTocppObject
Pipeline	ProfileCall	ProfileClass
ProfileLine	ProfileRoutine	Profiling
RemoteObject	ResultSet	ResultSets
RuntimeError	ScriptDefinition	Service
SimpleTypeDefinition	SSLCallBack	SSLServiceProvider
SystemFunctions	Throwable	Timing
TraceActivityNode	TraceBeginEnd	TraceError
TraceESQL	TraceFile	TraceGarbageCollect
TraceLine	TraceObject	TraceRoutine
TraceTree	TraceTreeError	TraceTreeESQL
TraceTreeGarbageCollect	TraceTreeLine	TraceTreeNode
TraceTreeObject	TraceTreeRoutine	TraceTreeUser
TraceUser	TransactionServer	Transport
TypeDefinition	VariableCardinalityDefinition	VariableDefinition

Workaround

Encapsulate the relevant script into non-visual objects (NVOs) and deploy the NVOs to EAServer.

Note

You can also add almost any type of business logic into N-Tier NVOs. Many of the other unsupported features that are discussed in this Workarounds Guide can be supported in N-Tier NVOs, such as DataWindow.RetrieveRow function in a DataStore, etc.

Example

The following steps show how to work around the ArrayBounds object.

Step 1 - Create *NVO n_cst_rs*.

Step 2 - Declare variable instance in *nvo(n_cst_rs)*.

```
resultset irs_resultset
   ADOresultset irs_ADOresultset
```

Step 3 - Define the function *Of_SetResultSet(ref oleobject aole)*.

```
// create ds_source
// generate a result set from an existing DataStore
ds_source.GenerateResultSet(irs_resultset)

// create a new ADOResultSet object and populate it
// from the generated result set
lrs_ADOresultset = CREATE ADOResultSet
lrs_ADOresultset.SetResultSet(irs_resultset)
// pass the data in the ADOResultSet object
// to an OLEObject you can use as an ADO Recordset
irs_ADOresultset.GetRecordSet(aole)
```

Step 4 - Declare variable instance on the Client side.

```
OLEObject ioo_ADOrecordset
N_Cst_RS IN_Cst_RS
```

Step 5 - Write script in the Client side objects.

```
ioo_ADOrecordset = CREATE OLEObject
IN_Cst_RS = CREATE N_CST_RS
IN_Cst_RS. Of_SetResultSet(ioo_ADOrecordset)
// call native ADO Recordset methods on the OLEObject
ioo_ADOrecordset.MoveFirst()
```

Step 6 - Deploy the NVO to EAServer.

3.6.2.2 Placing unsupported functions into n-Tier NVOs

Description

Not all PowerScript functions are supported. For detailed information on all unsupported PowerScript functions, refer to the Functions section of Appeon Features Help.

Workaround

If an unsupported PowerScript function meets the requirements for functions that can be packaged, encapsulate the function into a PowerBuilder non-visual user object (NVO).

Functions can be packaged into NVOs, provided that they do not:

- use the PowerScript MessageBox function. (EAServer limitation)
- use application global variables. (EAServer limitation)
- use visual controls or objects. (EAServer limitation)
- use Any or visual control/object data types as parameters for NVO functions and/or events. (EAServer limitation)

GetEnvironment, Randomize, and File functions are examples of functions that can be packaged.

For more information on how to use NVOs, refer to "*Move unsupported features to Appeon Server as n-Tier NVOs"* section in the *Appeon Migration Guide*, or perform the steps shown in the following example.

Example

The PowerScript GetEnvironment function is used to populate the Environment object that holds information about the computing platform that the PowerBuilder application is running on. The GetEnvironment() and Environment object are unsupported by Appeon. To work around these unsupported features, consider moving them into Appeon Server.

Be aware that the GetEnvironment function, when executed in EAServer, returns the system information of the server machine.

The following code, written in the Clicked event of the *cb_1* CommandButton in the PowerBuilder Client, gets the complete PowerBuilder version number.

```
Environment lenv_obj
Integer li_return
String ls_pbversion = "" //store PB version number
IF GetEnvironment (lenv_obj) = 1 THEN
 li_return = lenv_obj.PBMajorRevision
 ls_pbversion = ls_pbversion + String(li_return)
 li_return = lenv_obj.PBMinorRevision
 ls_pbversion = ls_pbversion + "." + String(li_return)
 li_return = lenv_obj.PBFixesRevision
 ls_pbversion = ls_pbversion + "." + String(li_return)
ELSE
 ls_pbversion = "-1"
END IF
MessageBox ("PB Version", ls_pbversion)
```

To enable the previous unsupported code to run in EAServer, perform the following steps:

Step 1 - Create a PowerBuilder custom class user object in the PowerBuilder Client. Name the custom class user object *nvo_environmen*t.

Step 2 - Add the user function *of_GetPBVersion* to *nvo_environment*. Set the function return value as a string, and add the following code to the function:

```
Environment lenv_obj
Integer li_return
String ls_pbversion = "" //store PB version number
IF GetEnvironment (lenv_obj) = 1 THEN
 li_return = lenv_obj.PBMajorRevision
 ls_pbversion = ls_pbversion + String(li_return)
 li_return = lenv_obj.PBMinorRevision
 ls_pbversion = ls_pbversion + "." + String(li_return)
 li_return = lenv_obj.PBFixesRevision
 ls_pbversion = ls_pbversion + "." + String(li_return)
ELSE
 ls_pbversion = "-1"
END IF
Return ls_pbversion
```

Step 3 - Create an EAServer Component project in the PowerBuilder Client that contains information on deploying *nvo_environment* to EAServer, then deploy *nvo_environment* to EAServer within the PowerBuilder IDE.

Step 4 - Create an EAServer Proxy project in the PowerBuilder Client that acts as the local representation of the deployed *nvo_environment* EAServer NVO component.

Step 5 - When the PowerBuilder Client application starts, create a Connection object named *myconnect*, and connect to the EAServer that hosts the *nvo_environment* NVO component.

Step 6 - In the Clicked event of the *cb_1* CommandButton in the PowerBuilder Client, comment all the original code that gets the PowerBuilder version and add the following code to the event:

```
// Define local variables
long ll_rc
string ls_pbversion
nvo_environment lnvo_1
// Instantiate remote component
ll_rc = myconnect.CreateInstance(lnvo_1, "nvo_env/nvo_environment")
IF ll_rc <> 0 THEN
 MessageBox("Create instance failed", ll_rc)
END IF
// Call component method
ls_pbversion = lnvo_1.of_GetPBVersion()
MessageBox ("PB Version", ls_pbversion)
```

Step 7 - Disconnect from EAServer when the Client application ends.

4 FAQ & Workarounds

This chapter lists some frequently asked questions and workaround tips regarding the Appeon application architecture or product features.

4.1 How to remove the Internet Explorer menu

Description

When a Web application is opened in Internet Explorer, you may see both the Internet Explorer menu and the Web application menu. It is possible to design an HTML file or C++ program to remove the Internet Explorer menu.

Workaround

Method 1: In the application folder under the Web server Web root, create an HTML file named appeon.html for loading the Web application. For example,

```
<html>
 <head>
   <title>Appeon Web Library </title>
 </head>
 <script language="javascript">
    function startApp() {
    g_newWindow = window.open("index.html", "_blank",
    "location=no,titlebar=no,toolbar=no,menubar=no,status=no,resizable=yes",false);
}
 </script>
 <body>
  <script language="javascript">
    startApp()
  </script>
 </body>
</html>
```

Note: In order for sFeatures settings to take effect, the sName argument in the Open method must be “_blank”. This opens a new Internet Explorer window for index.html (the index page of the Web application) and hides the Internet Explorer menu.

Instead of loading Index.html as the entry page for Appeon Web applications, load the appeon.html file using a URL similar to this: http://host:port/appname/appeon.html. When the Open method in the appeon.html file is triggered, the Index.html page will be loaded in a new Internet Explorer window.

Method 2: Create a C++ program that utilizes COM API on the Client side.

The sample code in the C++ program is as follows:

```
// Start a new Internet Explorer as a separate process
 IWebBrowser2* pIE = NULL;
 HRESULT hr;
 hr = CoCreateInstance(CLSID_InternetExplorer, NULL, CLSCTX_SERVER,
    IID_IWebBrowser2, (LPVOID*)&pIE); // if open IE OK
  if (SUCCEEDED(hr)) {
    pIE->put_Visible(TRUE);
    pIE->put_AddressBar(FALSE);
    pIE->put_MenuBar(FALSE);
    pIE->put_StatusBar(TRUE);
    pIE->put_ToolBar(FALSE);
    pIE->put_FullScreen(FALSE);
    COleVariant vtEmpty;
    CString strURL = "http://appeonserver:81"; //The URL to be opened by the program.
    BSTR bstrURL = strURL.AllocSysString();
    pIE->Navigate(bstrURL, &vtEmpty, &vtEmpty, &vtEmpty, &vtEmpty);
    ::SysFreeString(bstrURL); }
```

If the user runs the C++ program on the Client machine, the Web application will be opened in an Internet Explorer browser and the display mode of the Internet Explorer browser is specified in the C++ program.

4.2 How to deploy NVO to EAServer 6.1

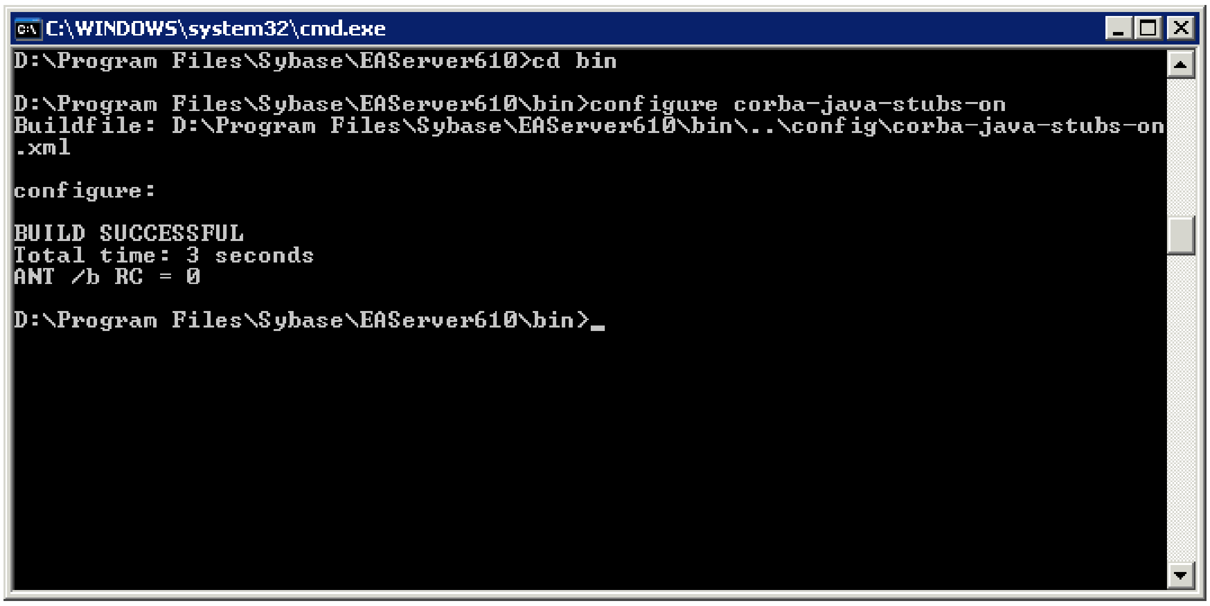
In this section we will demonstrate how to deploy NVO to EAServer 6.1 in PowerBuilder 11. The example NVO used in this section is a simple NVO.

Configuring to Generate Stub and Skeleton

In Windows, run the following command at a prompt window to make sure Stub and Skeleton will be generated during the NVO deployment:

%EAServer%\bin\configure corba-java-stubs-on

Figure 4.1:



```
D:\Program Files\Sybase\EAServer610>cd bin

D:\Program Files\Sybase\EAServer610\bin>configure corba-java-stubs-on
Buildfile: D:\Program Files\Sybase\EAServer610\bin\..\config\corba-java-stubs-on
.xml

configure:

BUILD SUCCESSFUL
Total time: 3 seconds
ANT /b RC = 0

D:\Program Files\Sybase\EAServer610\bin>_
```

Adding EAServer profile in PowerBuilder

Use the PowerBuilder component wizard for deploying an NVO to EAServer. Before using the wizard, create an EAServer profile in PowerBuilder, which will connect to the EAServer that hosts Appeon Server.

An EAServer profile stores information on connection settings used to connect to EAServer. The profile you create is used by wizards that require a connection to EAServer.

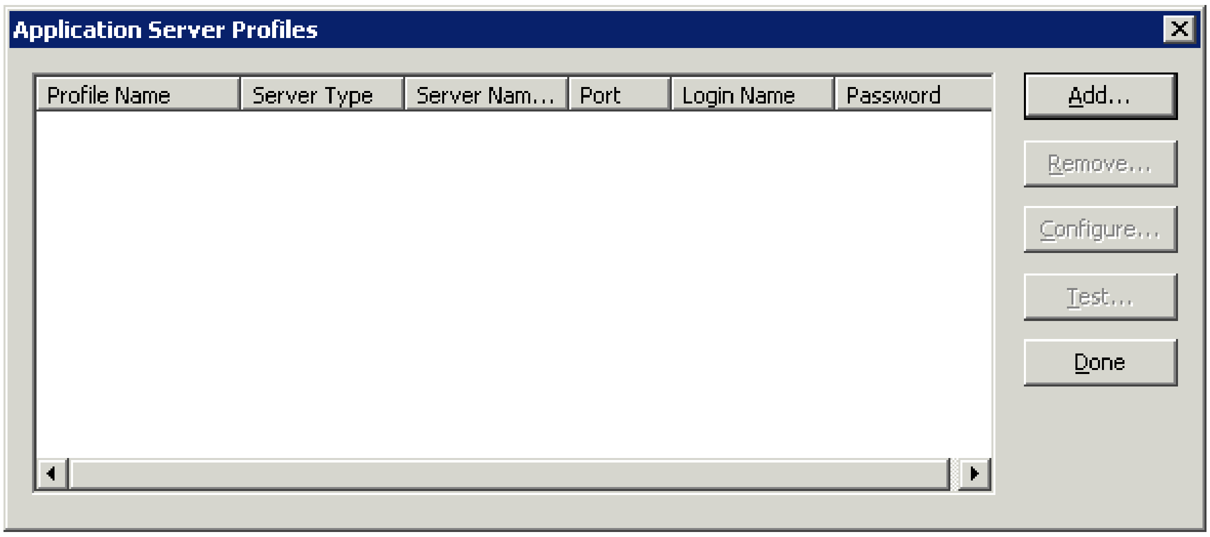
To add an EAServer profile to PowerBuilder:

Step 1 - Start EAServer, if it is not already running. Make sure that EAServer is running during the following steps.

Step 2 - Click the Application Server Profile button in PowerBar1 or choose *Tools* | Application Server Profile from the PowerBuilder menu.

Click *Add* on the Application Server Profile dialog box that is displayed.

Figure 4.2:

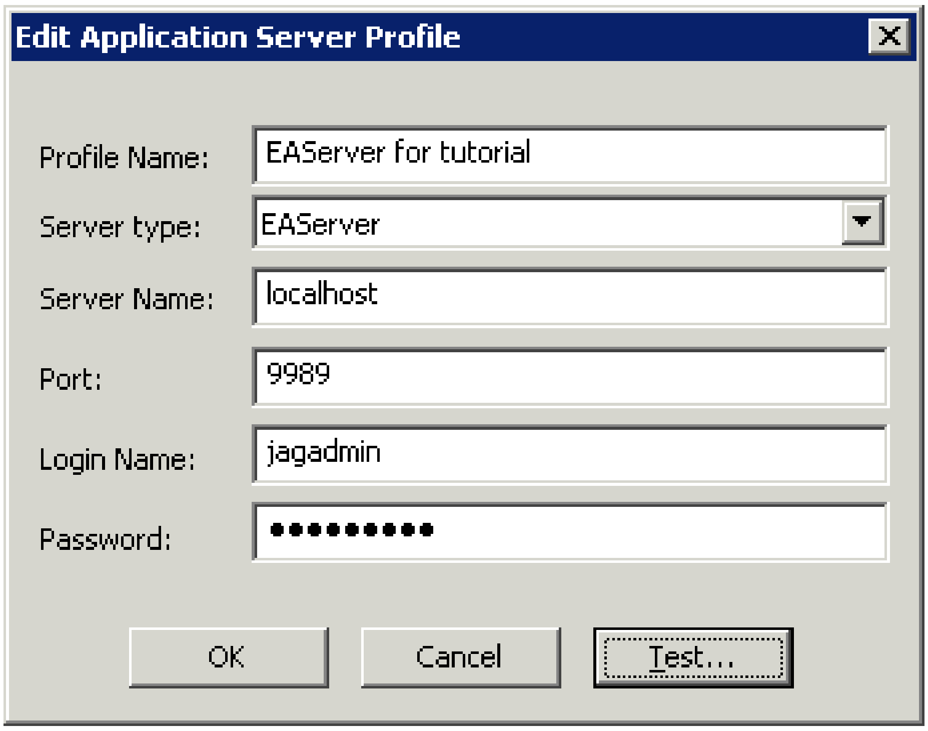


Step 3 - The Edit Application Server Profile dialog box pops up. Type the information contained in the following table into each field. And then verify the EAServer profile by clicking the *Test* button. Note: EAServer must be running for the test to be successful.

Table 4.1:

In this field...	You should...
Profile Name	Type *EAServer for tutorial*
Server Type	Choose EAServer
Server Name	Type *localhost*
Port Number	Type 9989
Login Name	The default login name is *jagadmin*. If you have changed the default, enter the correct login name.
Password	There is no password by default. If you have changed the default, enter the correct password.

Figure 4.3:


Edit Application Server Profile

Profile Name:	EAServer for tutorial
Server type:	EAServer
Server Name:	localhost
Port:	9989
Login Name:	jagadmin
Password:	••••••••••

Step 5 - Make sure that connection testing is successful.

4.2.1 Deploying NVOs to EAServer

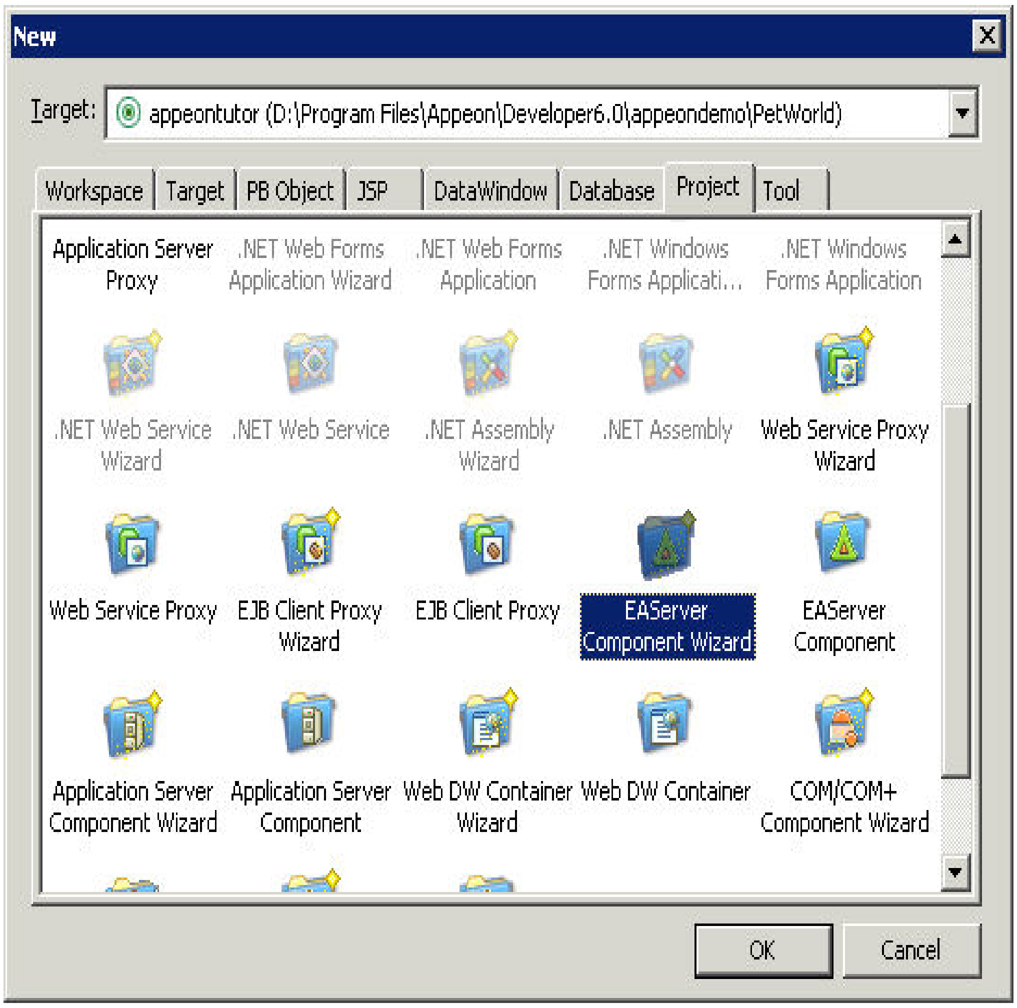
Use the EAServer Component Wizard to deploy EAServer components from PowerBuilder.

To deploy the NVO into EAServer:

Step 1 - Start EAServer, if it is not already started.

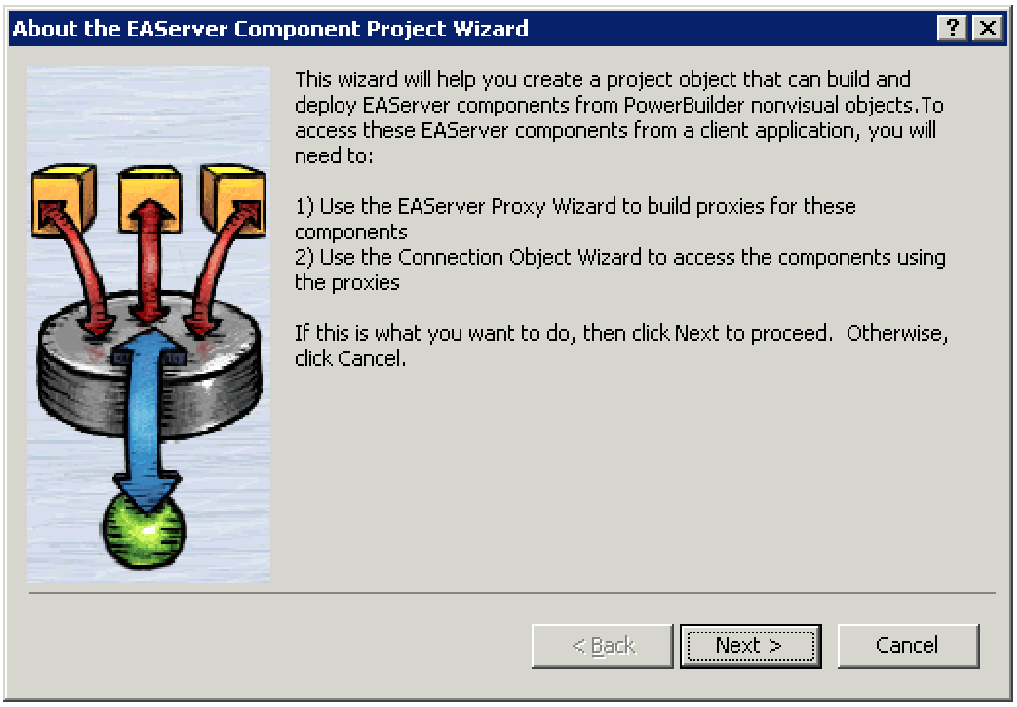
Step 2 - Choose *File* | *New* from the PowerBuilder menu. Then select the *EAServer Component Wizard* icon under the Project tab in the New dialog box, and click *OK*.

Figure 4.4:


New
Target: appeontutor (D:\Program Files\Appeon\Developer6.0\appeondemo\PetWorld)
Workspace
Target
PB Object
JSP
DataWindow
Database
Project
Tool
Application Server Proxy
.NET Web Forms Application Wizard
.NET Web Forms Application
.NET Windows Forms Applicati...
.NET Windows Forms Application
.NET Web Service Wizard
.NET Web Service
.NET Assembly Wizard
.NET Assembly
Web Service Proxy Wizard
Web Service Proxy
EJB Client Proxy Wizard
EJB Client Proxy
EAServer Component Wizard
EAServer Component
Application Server Component Wizard
Application Server Component
Web DW Container Wizard
Web DW Container
COM/COM+ Component Wizard
OK
Cancel

Step 3 - The EAServer Component Project Wizard starts. Click *Next*.

Figure 4.5:


About the EAServer Component Project Wizard

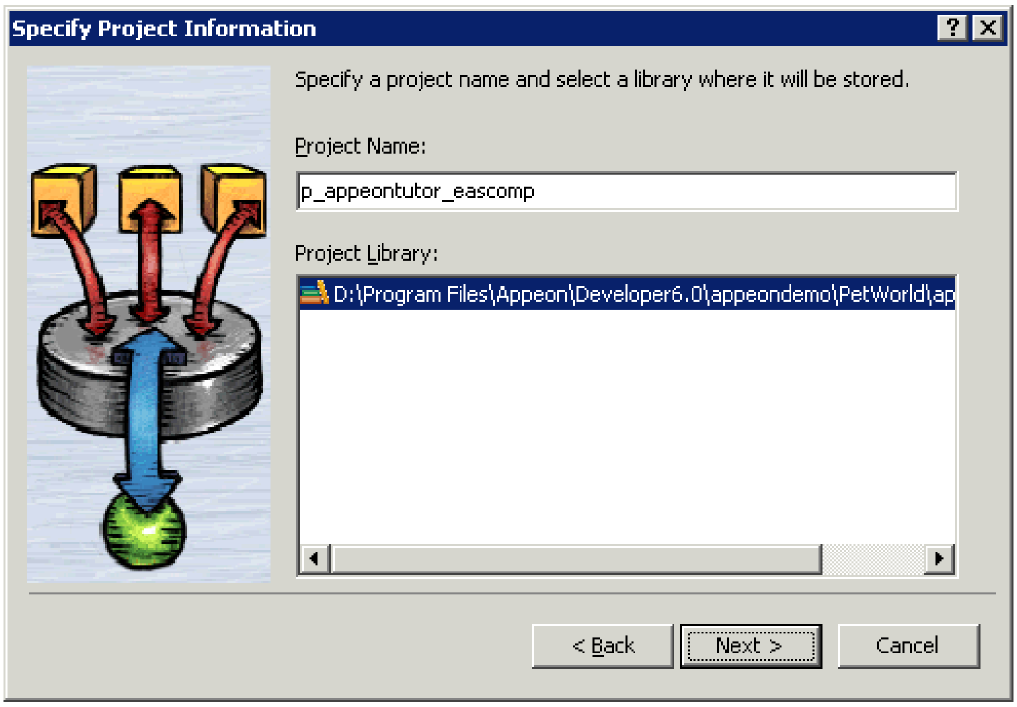
This wizard will help you create a project object that can build and deploy EAServer components from PowerBuilder nonvisual objects.To access these EAServer components from a client application, you will need to:

1) Use the EAServer Proxy Wizard to build proxies for these components
2) Use the Connection Object Wizard to access the components using the proxies

If this is what you want to do, then click Next to proceed. Otherwise, click Cancel.

Step 4 - Leave the project name as p_appeontutor_eascomps and Click *Next*.

Figure 4.6:


Specify Project Information

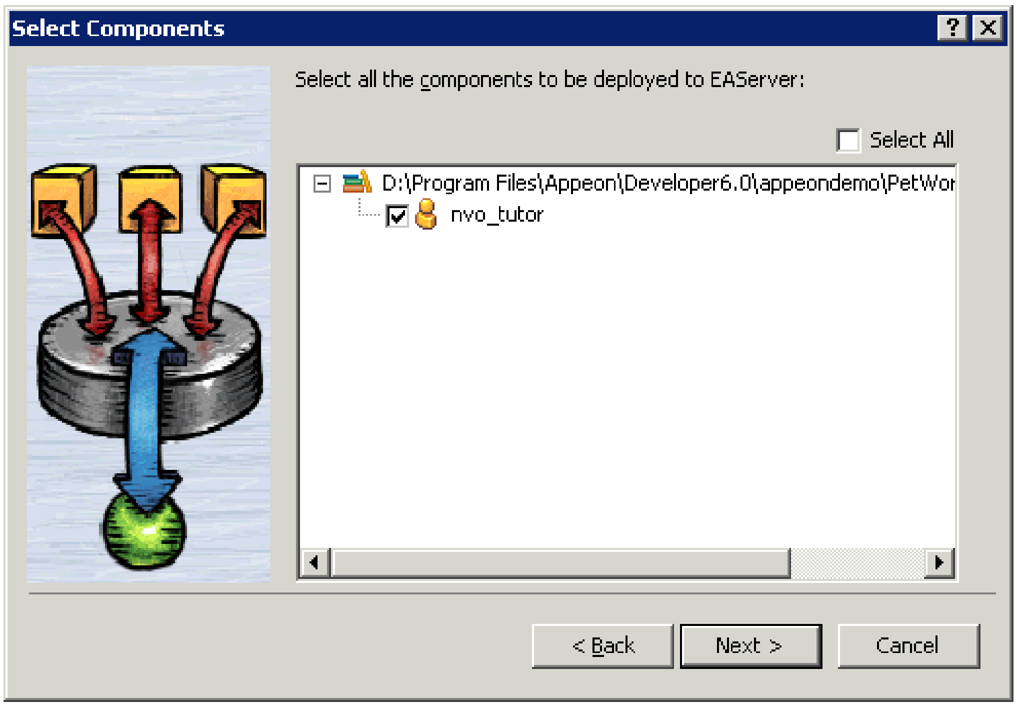
Specify a project name and select a library where it will be stored.

Project Name: p_appeontutor_eascomp

Project Library: D:\Program Files\Appeon\Developer6.0\appeondemo\PetWorld\ap

Step 6 - Select the *nvo_tutor* NVO that is to be deployed to EAServer. Click *Next*.

Figure 4.7:

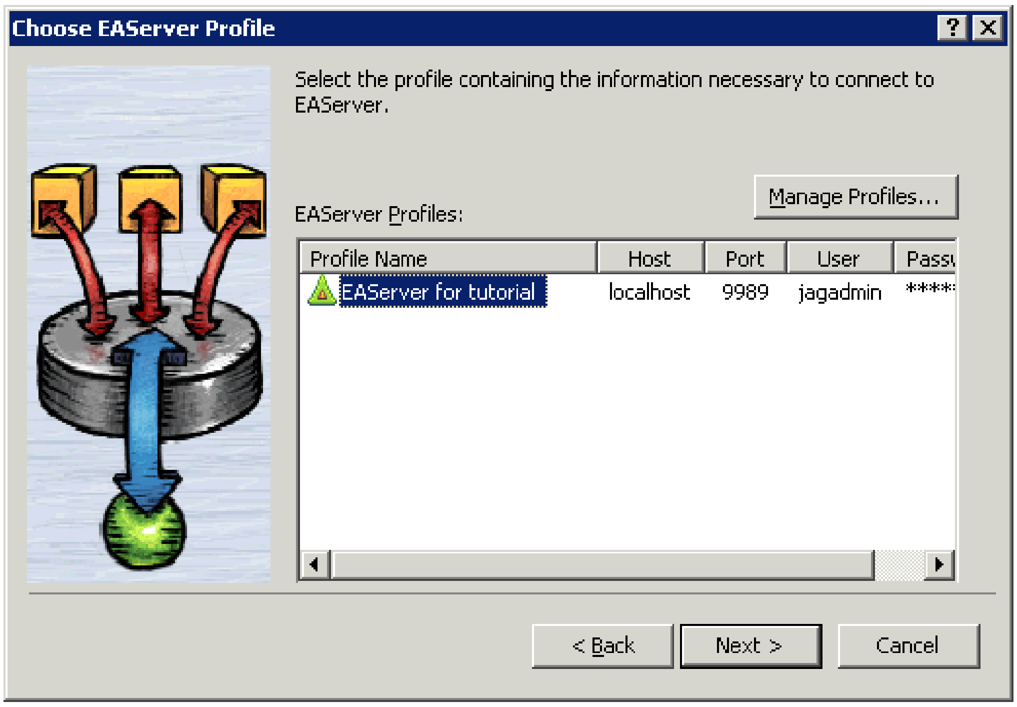


Select all the components to be deployed to EAServer:

D:\Program Files\Appeon\Developer6.0\appeondemo\PetWor
[✓] nvo_tutor

Step 7 - Select the *EAServer for tutorial* profile. You created this profile in a previous step. If you have set up profiles outside of the tutorial for other EAServers, there will be more profiles in the list.

Figure 4.8:



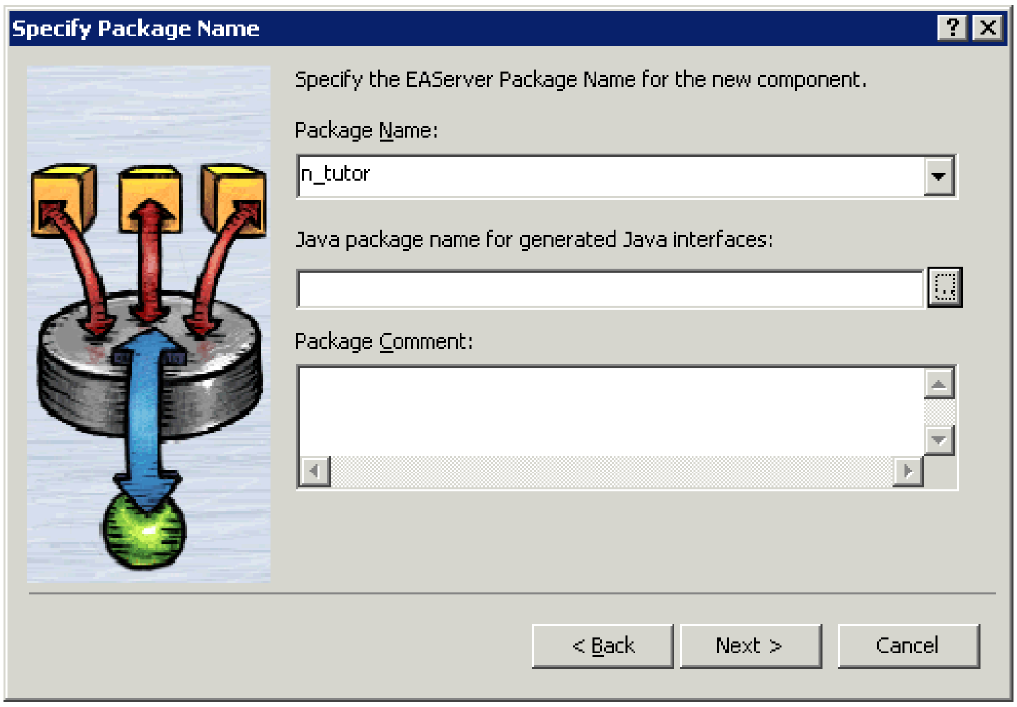
Select the profile containing the information necessary to connect to EAServer.

EAServer Profiles:

Profile Name	Host	Port	User	Passi
EAServer for tutorial	localhost	9989	jagadmin	*****

Step 8 - Type *n_tutor* into the Package Name dropdown listbox. If necessary type the Java package name according to your own needs. Click *Next*.

Figure 4.9:


Specify Package Name

Specify the EAServer Package Name for the new component.

Package Name:

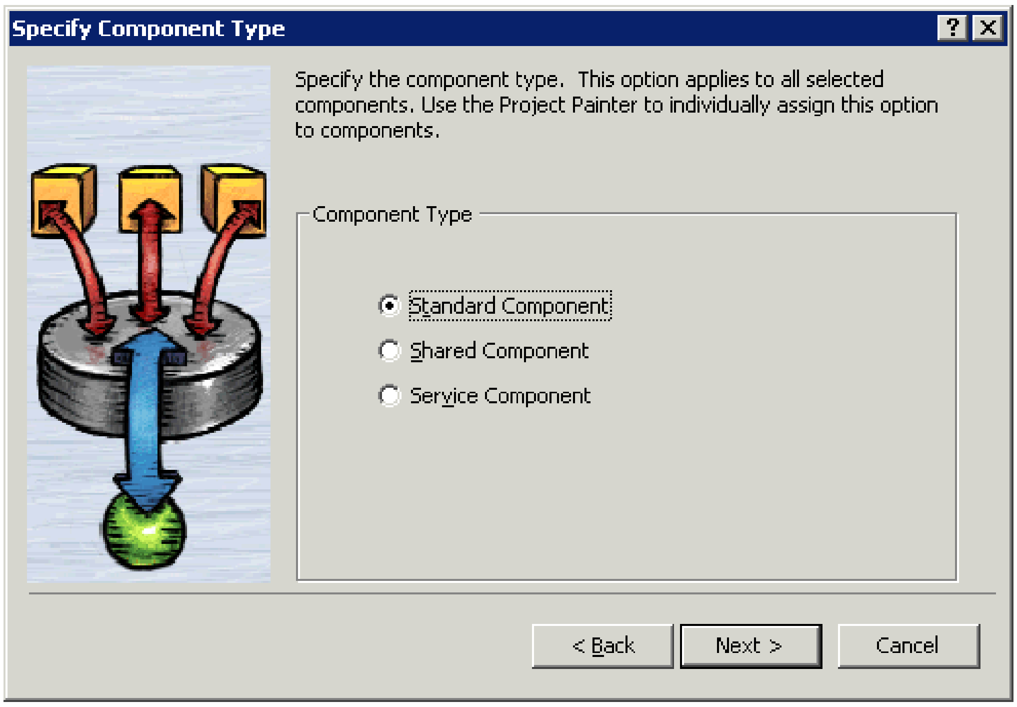
n_tutor

Java package name for generated Java interfaces:

Package Comment:

Step 9 - Select *Standard Component* on the Specify Component Type dialog box. Click *Next*.

Figure 4.10: PDF


Specify Component Type

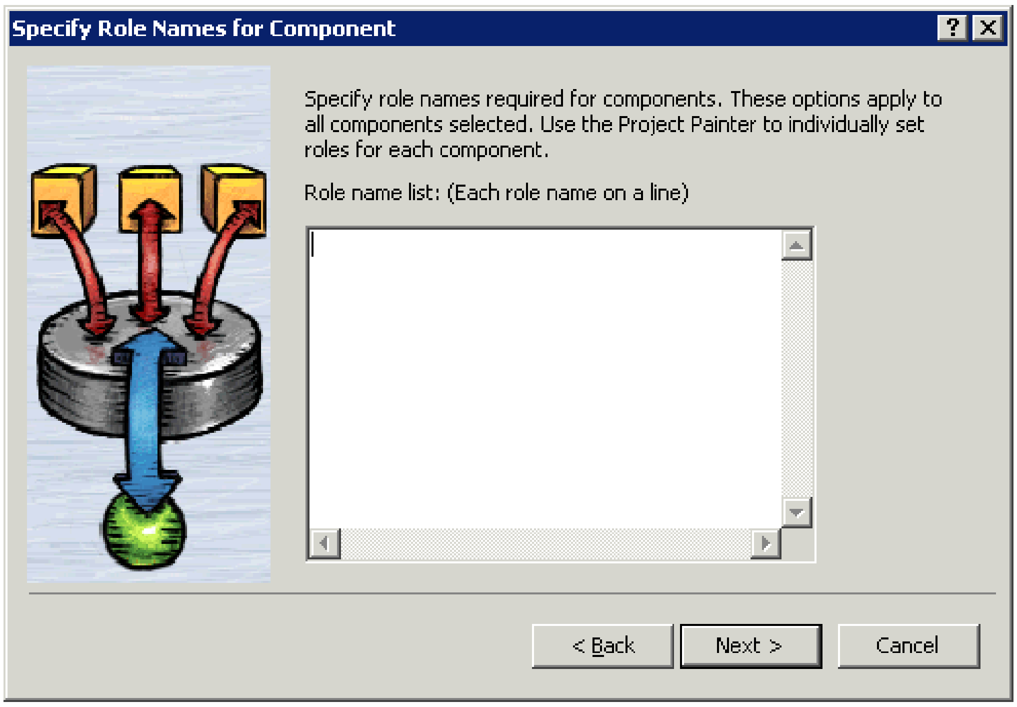
Specify the component type. This option applies to all selected components. Use the Project Painter to individually assign this option to components.

Component Type

[✓ Standard Component]
Shared Component
Service Component

Step 10 - Specify the role name if necessary. Click Next.

Figure 4.11:



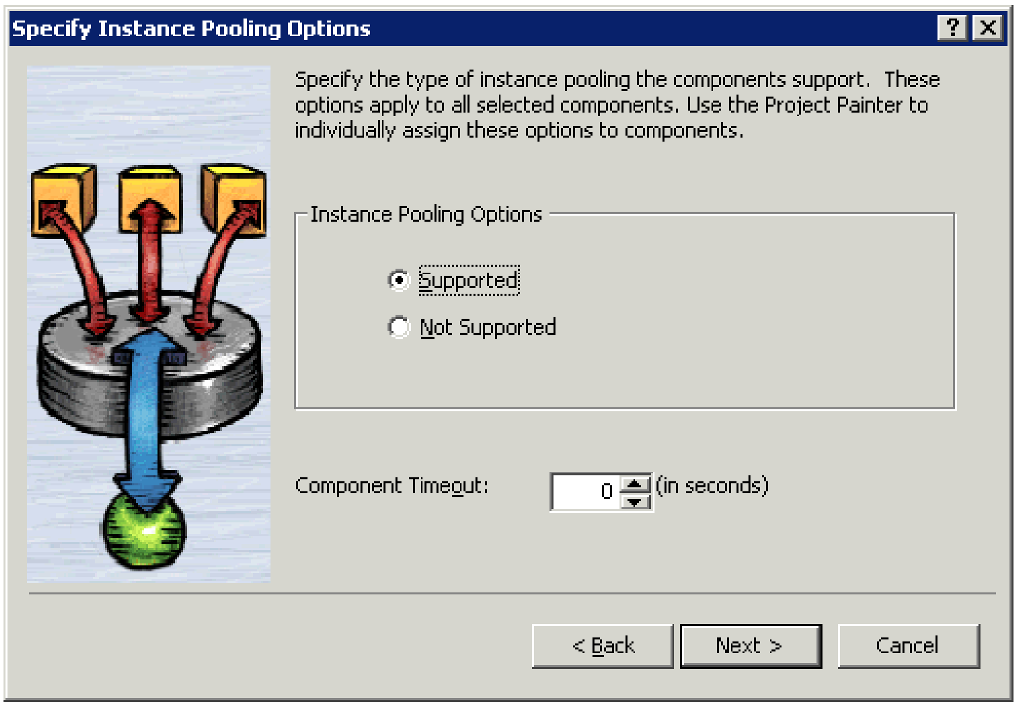
Specify Role Names for Component

Specify role names required for components. These options apply to all components selected. Use the Project Painter to individually set roles for each component.

Role name list: (Each role name on a line)

Step 11 - Leave the *Supported* option checked for the *Instance Pooling Options* field. Click *Next*.

Figure 4.12:



Specify Instance Pooling Options

Specify the type of instance pooling the components support. These options apply to all selected components. Use the Project Painter to individually assign these options to components.

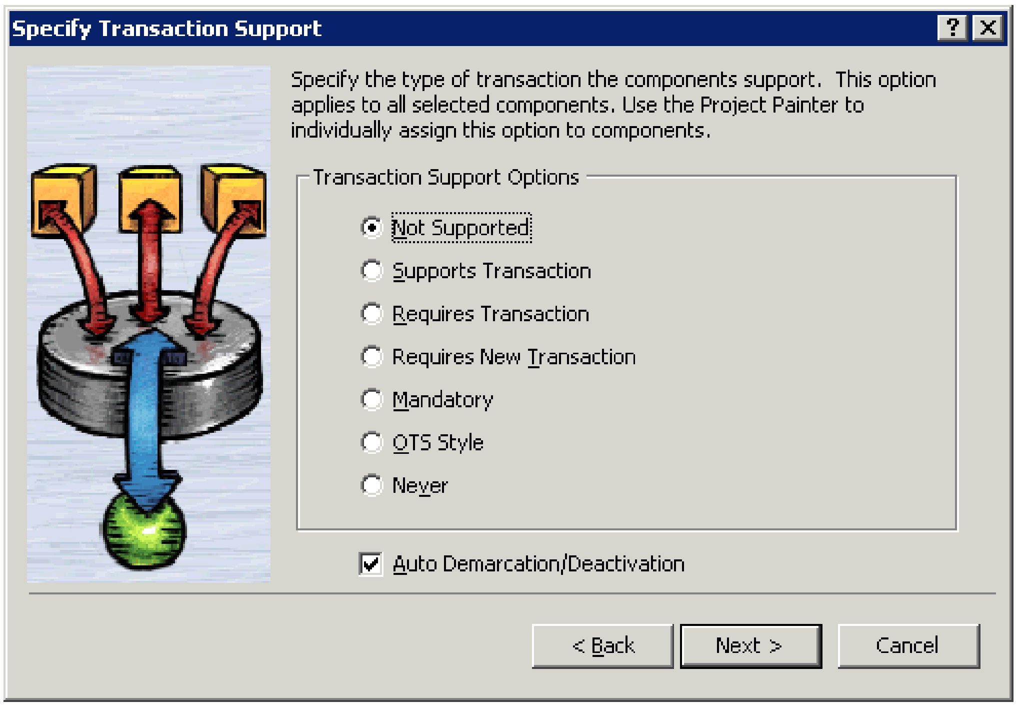
Instance Pooling Options

[✓ Supported] Not Supported

Component Timeout: 0 (in seconds)

Step 11 - In the *Transaction Support Options* field leave the default option as *Not Supported*. Click *Next*.

Figure 4.13:



Specify Transaction Support

Specify the type of transaction the components support. This option applies to all selected components. Use the Project Painter to individually assign this option to components.

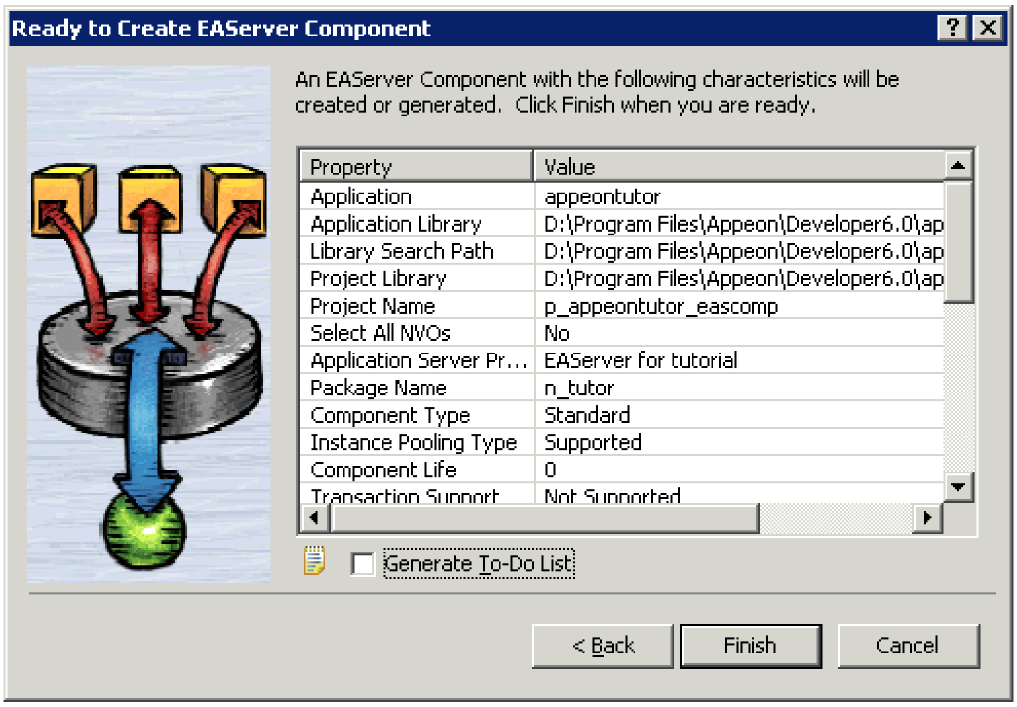
Transaction Support Options

[✓ Not Supported]
Supports Transaction
Requires Transaction
Requires New Transaction
Mandatory
OTS Style
Never

[✓] Auto Demarcation/Deactivation

Step 12 - Leave all options at default. Click Next until the wizard now comes to its final stage. Click Generate To-Do List and *Finish*.

Figure 4.14:



Ready to Create EAServer Component

An EAServer Component with the following characteristics will be created or generated. Click Finish when you are ready.

Property	Value
Application	appeontutor
Application Library	D:\Program Files\Appeon\Developer6.0\ap
Library Search Path	D:\Program Files\Appeon\Developer6.0\ap
Project Library	D:\Program Files\Appeon\Developer6.0\ap
Project Name	p_appeontutor_eascomp
Select All NVOs	No
Application Server Pr...	EAServer for tutorial
Package Name	n_tutor
Component Type	Standard
Instance Pooling Type	Supported
Component Life	0
Transaction Support	Not Supported

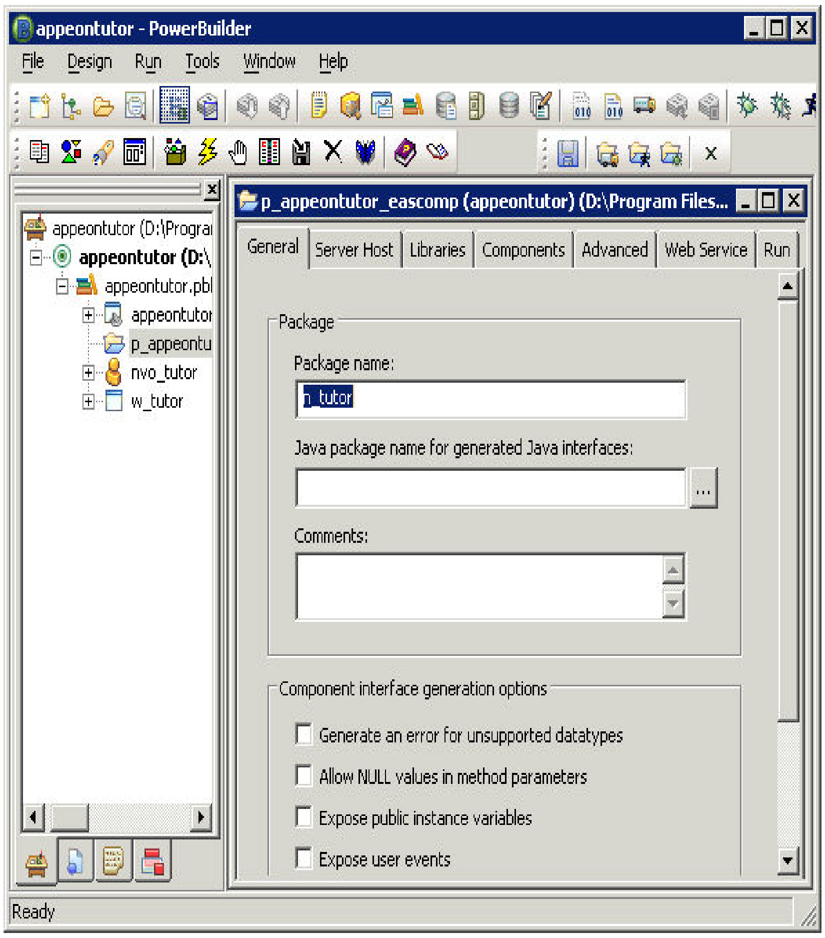
[ ] Generate To-Do List

Step 13 - The wizard now creates a component project. The Project Painter for the component opens automatically in PowerBuilder, and the project name (p_appeontutor_eascomps) appears in the left system tree list.

Click the *Deploy* button in PowerBuilder PainterBar1, or choose *Design | Deploy Project* from the PowerBuilder menu to start deployment of the component.

Note: Ensure that EAServer is running.

Figure 4.15:



appeontutor - PowerBuilder
File Design Run Tools Window Help
appeontutor (D:\Progra
appeontutor (D:\
appeontutor.pbl
appeontutor
p_appeontu
nvo_tutor
w_tutor
p_appeontutor_eascomp (appeontutor) (D:\Program Files...
General Server Host Libraries Components Advanced Web Service Run
Ready

Package

Package name: n_tutor

Java package name for generated Java interfaces:

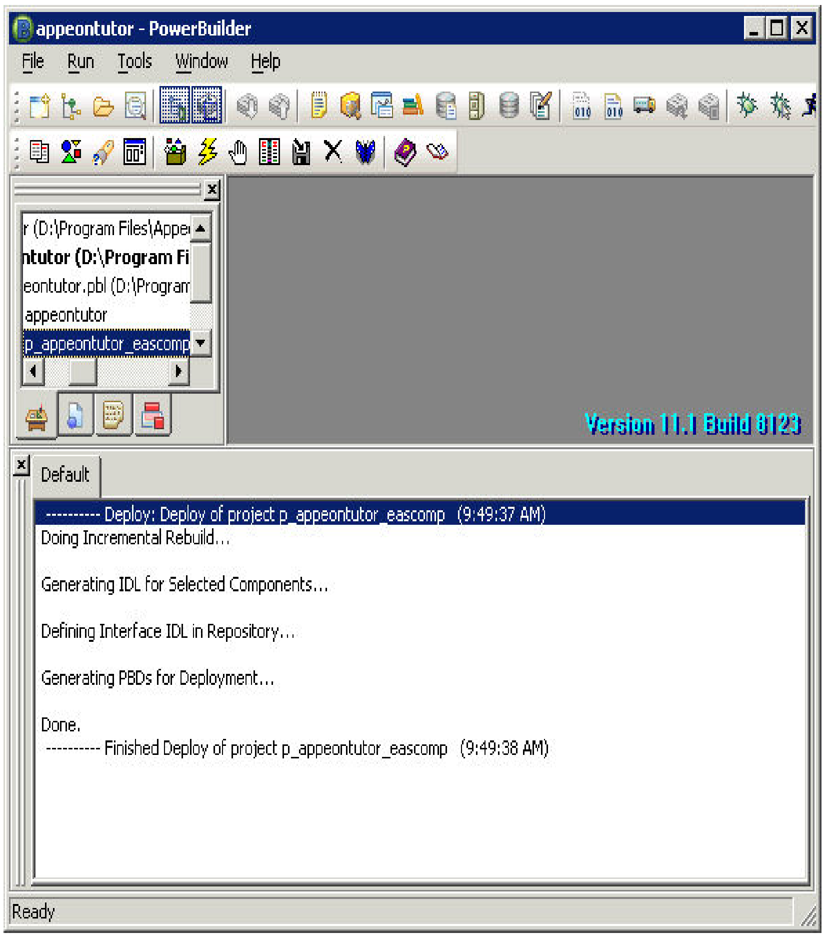
Comments:

Component interface generation options

- [ ] Generate an error for unsupported datatypes
- [ ] Allow NULL values in method parameters
- [ ] Expose public instance variables
- [ ] Expose user events

Step 14 - The deployment starts and relevant information is displayed in the Output window. When the process is complete, the Output window will display "Finished Deploy of p_appeontutor_eascomps".

Figure 4.16:



```
--------- Deploy: Deploy of project p_appeontutor_eascomp   (9:49:37 AM)
Doing Incremental Rebuild...

Generating IDL for Selected Components...

Defining Interface IDL in Repository...

Generating PBDs for Deployment...

Done.
 --------- Finished Deploy of project p_appeontutor_eascomp   (9:49:38 AM)
```

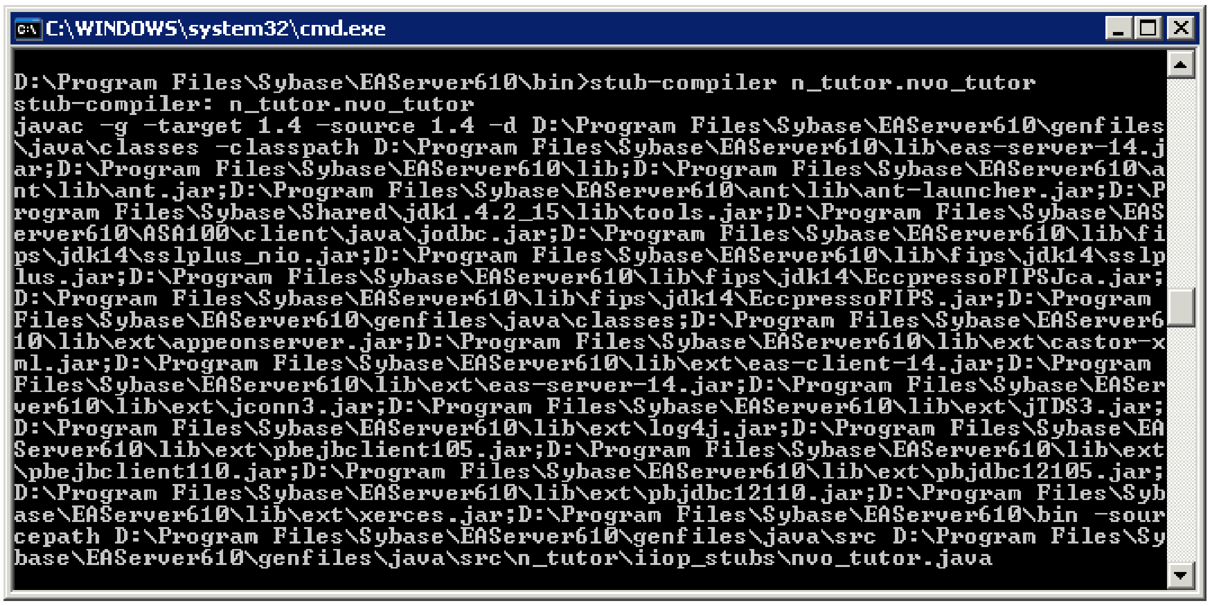
Step 15 - Close the *p_appeontutor_eascomps* Project Painter.

Step 16 - Check if the Stub is generated in the folder ${EAServer}\genfiles\java\classes. If the stub is not generated manually compile the Stub by executing the following command at a prompt window:

${EAServer}\bin\stub-compiler package_name.component_name.

In the deployment mentioned in this section the package name is n_tutor, the component name is nvo_tutor.

Figure 4.17:


C:\WINDOWS\system32\cmd.exe

```
D:\Program Files\Sybase\EAServer610\bin>stub-compiler n_tutor.nvo_tutor
stub-compiler: n_tutor.nvo_tutor
javac -g -target 1.4 -source 1.4 -d D:\Program Files\Sybase\EAServer610\genfiles
\java\classes -classpath D:\Program Files\Sybase\EAServer610\lib\eas-server-14.j
ar;D:\Program Files\Sybase\EAServer610\lib;D:\Program Files\Sybase\EAServer610\a
nt\lib\ant.jar;D:\Program Files\Sybase\EAServer610\ant\lib\ant-launcher.jar;D:\P
rogram Files\Sybase\Shared\jdk1.4.2_15\lib\tools.jar;D:\Program Files\Sybase\EAS
erver610\ASA100\client\java\jodbc.jar;D:\Program Files\Sybase\EAServer610\lib\fi
ps\jdk14\sslplus_nio.jar;D:\Program Files\Sybase\EAServer610\lib\fips\jdk14\sslp
lus.jar;D:\Program Files\Sybase\EAServer610\lib\fips\jdk14\EccpressoFIPSJca.jar;
D:\Program Files\Sybase\EAServer610\lib\fips\jdk14\EccpressoFIPS.jar;D:\Program
Files\Sybase\EAServer610\genfiles\java\classes;D:\Program Files\Sybase\EAServer6
10\lib\ext\appeonserver.jar;D:\Program Files\Sybase\EAServer610\lib\ext\castor-x
ml.jar;D:\Program Files\Sybase\EAServer610\lib\ext\eas-client-14.jar;D:\Program
Files\Sybase\EAServer610\lib\ext\eas-server-14.jar;D:\Program Files\Sybase\EASer
ver610\lib\ext\jconn3.jar;D:\Program Files\Sybase\EAServer610\lib\ext\jTDS3.jar;
D:\Program Files\Sybase\EAServer610\lib\ext\log4j.jar;D:\Program Files\Sybase\EA
Server610\lib\ext\pbejbclient105.jar;D:\Program Files\Sybase\EAServer610\lib\ext
\pbejbclient110.jar;D:\Program Files\Sybase\EAServer610\lib\ext\pbjdbc12105.jar;
D:\Program Files\Sybase\EAServer610\lib\ext\pbjdbc12110.jar;D:\Program Files\Syb
ase\EAServer610\lib\ext\xerces.jar;D:\Program Files\Sybase\EAServer610\bin -sour
cepath D:\Program Files\Sybase\EAServer610\genfiles\java\src D:\Program Files\Sy
base\EAServer610\genfiles\java\src\n_tutor\iiop_stubs\nvo_tutor.java
```

4.3 How to deploy NVOs to EAServer without PowerBuilder

Description

You can deploy NVOs to EAServer without PowerBuilder.

Workaround

Step 1 - Copy the following files and sub-directories about the n-Tier NVO from %JAGUAR %\Repository\Component\PackageName.

- The component property files (*.props)
- The Component PBD files
- The component sub-directories

Step 2 - Paste the files and sub-directories obtained in Step 1 to %JAGUAR%\Repository \Component\PackageName in the new EAServer.

Step 3 - Go to EAServer Manager and generate stubs and skeletons for the components. For detailed instructions, refer to How to deploy NVO to EAServer 6.1.

4.4 How to use Client resources in Appeon Web applications

Description

Appeon Web applications cannot directly interface with external resources from PowerBuilder code on the Client.

Workaround

Transfer Client resources to the Server, do all manipulations on the Server, and then return the result to the Client.

Example

The following example shows you how to use an n-tier NVO and a JSP page to work around a PowerBuilder application whose main functionality is updating a database using a text file at the Client.

The original PowerBuilder application

The main logic

Step 1 - In a DataWindow control, save the specified column values as a text file on the Client.

Step 2 - Modify the column values stored in the text file.

Step 3 - Update the latest value stored in the text file to the database.

After modification

The main logic

Step 1 - Encapsulate the following logic to an NVO.

- Logic to save the specified column values as a text file.
- Logic to update the latest value stored in the text file to the database.

Step 2 - Deploy the NVO to EAServer as a Jaguar component.

Step 3 - Call Jaguar component to save the column values as a text file on the Server.

Step 4 - Read the contents of the text file and save it to the Client using IE browser.

Step 5 - Modify the column values in the text file on the Client.

Step 6 - Upload the text file to the Server using JSP.

Step 7 - Use the functionality encapsulated in the Jaguar component to update the database.

4.5 How to replace Appeon image that displays at the running of applications

Description

When you launch an application in Internet Explorer, an image () shows in the browser for a moment, indicating the start of the loading process. The image is pre-defined in Appeon, but you can change it to any other image.

Workaround

Once you change the Appeon-defined image to your own image, all applications deployed to the server will show the new image at the beginning of application-loading process.

Step 1 - Prepare an image that you want to display at the beginning of application-loading process. Name it to "awl_loading.gif".

Step 2 - Go to the folder where the Appeon-defined image is stored, and replace the old awl_loading.gif with the new awl_loading.gif. You need to replace awl_loading.gif in the following two folders: %AppeonServer%\weblibrary_ax\debug\image\ and %AppeonServer%\weblibrary_ax\release\image\.

4.6 How to deploy an Appeon application without Appeon Developer

Description

Appeon Developer is needed to deploy the application to at least one Appeon Server and Web Server. Once deployed to an Appeon Server and a Web Server, you can replicate the deployed application to other Appeon Servers and Web Servers without using Appeon Developer.

Workaround

Use the Package Wizard provided in the Appeon Developer toolbar to generate a portable installation package for your Appeon deployed application and install the package to the other Appeon Servers and Web Servers. For step-by-step instructions on how to package and install applications, please refer to the "*Packaging and Installing Appeon Applications*" chapter in *Appeon Developer User Guide*.

4.7 How to log in the Web application with single sign-on

Method 1: Use a server component to manage user information.

Step 1 - Create an EAServer shared component (either a PowerBuilder or Java component) for storing user information. This shared component is shared between the Appeon application and other applications such as COBOL apps, Web Services, SOA, etc.

Step 2 - Add the following logic to the Appeon application: when a user logs in, the user information (user ID and password) is passed to the EAServer shared component.

Step 3 - When the user accesses another application (COBOL apps, Web Services, SOA, etc), the other application gets the user information from the shared component and authenticates the user.

The same method can be used for the user to first access a non-Appeon application and then access an Appeon application with single sign-on.

Method 2: Apply command line argument.

Appeon supports the CommandParm function and the command line argument in the Open event of a PowerBuilder application. These features can be applied for implementing single sign-on.

The command line argument can be passed to an Appeon application in the following way:

http://192.0.1.94:8080/MyTest/index.htm?user=appeon&password=appeon

This attaches the string "*index.htm?arguments*" to the end of the original application URL ("*index.htm*" must be included in the string).

If the user wants to launch an Appeon application after logging on to an LDAP based application, the LDAP based application passes the user information via the URL of the Appeon application, and the user starts the Appeon application without further login procedures.

It is also possible to pass the session ID only. A table is created in the database for keeping the session information of the LDAP based application, with a session ID assigned to each session, and the session information containing user information. When the Appeon application is launched with the session ID as its command line argument, the application reads from the database table the user information and authenticates the user. The user can start the Appeon application without further login procedures.

4.8 How to add headers & footers to a Web application

Description

Use this solution to add headers and footers to a corporate website with an Appeon application in a frame.

Workaround

Create a new frame page with a header and footer. Set the URL that is used to access the deployed application to the initial page of the content frame.

Step 1 - Prepare two HTML files: *header.html* and *footer.html*.

Step 2 - In Microsoft FrontPage, create a new frame page that contains header, footer and content frames. To create a new frame page, click *File* | *New* | *Page*. In the *Frames Page* tab, select *Header, Footer and Contents Template*.

Step 3 - Set *header.html* as the initial page of the header frame, and set *footer.html* as the initial page of the footer frame.

Step 4 - Set the URL that is used to access the deployed application as the initial page of the content frame.

Step 5 - Put the frame page in the same Web root as the original deployed application and provide the frame page name to the user.

Now the user can directly open the frame page which opens the deployed application. Headers and footers have already been added to the deployed application.

4.9 How to get the user name and password of the operating system

To get the user name and password of the operating system, follow the two steps:

Step 1 - Package the GetUserName function into a DLL then calling the DLL in PowerBuilder.

Step 2 - Deploy the DLL to Appeon server and download the DLL to the client needed. You can get the detail information of the GetUserName function from the following Website: http://msdn.microsoft.com/en-us/library/windows/desktop/ms724432(v=vs.85).aspx.

4.10 How to set limited connections for an Appeon application using ASA database

Cannot limit connections with only Limited Seat(s) property

The Limited Seat(s) property of ASA databases cannot directly limit connections to the database of an Appeon deployed application, although it works well in the PowerBuilder application.

The reason is that the way the database server identifies users is changed in the Appeon architecture. In the traditional Client/Server application, the clients interact with the database directly; the database can obtain the correct information on how many users have been connected to the database by using the Limited Seat(s) property. While in the Appeon application, the clients do not connect to the database directly, instead, all clients connect to Appeon Server and it is Appeon Server that interacts with the database. As a result, the database always considers that there is only one user ("Appeon Server") connected.

How to limit connections in Appeon architecture

To limit connections for the Appeon application, perform the following steps:

Step 1 - Set the limited connections in the ASA database by the Limited Seat(s) property.

Step 2 - Use SELECT property('LicenseCount') SQL syntax to get the value of the Limited Seat(s) property in the ASA database.

Step 3 - Use the AppeonGetSessionCount function provided by Appeon to get the total number of the clients connected to Appeon Server.

Step 4 - Write script to compare the two values obtained in Step 2 and Step 3. When the number of connections to Appeon Server is smaller than the number set as the limited connections for the ASA database, the clients can still log to the Appeon application.

4.11 How to modify the storage location of Web application files in Appeon Developer

Appeon Developer will generate and store Web application files on the local machine before deploying it to Appeon Server. You can configure to store these files to other location rather than the default one. By default the Web files will be stored in %Appeon_Developer %\Project\%application_name% (e.g., C:\Program Files\Appeon\Developer6.5\Project \PetWorld). %Appeon_Developer% indicates the installation path of Appeon Developer.

How to configure the location for Web files

Step 1 - Open the ADTConfig.xml file in the directory %Appeon_Developer% \Developer2013.

Step 2 - Find the following lines:

```
<PathCfg>
<Project value="%Appeon_Developer%\Developer2013\Project\"/>
</PathCfg>
```

Step 3 - Replace the value in bold with another location where you want to store the Web files.

4.12 How to integrate Appeon Web applications with JSP/ASP

4.12.1 Applying Appeon CommandParm and Hyperlink features

Description

If your application needs to pass parameters to a JSP/ASP application, use the following method.

Workaround for passing parameters from a JSP/ASP application to an Appeon application

Apply the JSP/ASP programming method to add parameters to the URL of the Appeon application. Based upon the parameters, any functionality can be built in the Appeon Web application, such as opening windows and retrieving data to provide client-side integration.

The Appeon application receives the parameters using the CommandParm function and CommandLine parameter of the Open even.

Workaround for passing parameters from an Appeon application to a JSP/ASP application

Appeon supports PictureHyperLink and StaticHyperLink window controls. For example, you can statically or dynamically assign the URL of the JSP/ASP Web application to *http://www.x.x/index.asp?aid=x&bid=y&cid=z* in the Clicked event of a PictureHyperLink or StaticHyperLink window control and send the parameter from the Appeon application to a JSP/ASP application.

Appeon also supports the HyperLinkToURL PowerScript function. The developers can also apply this function to pass the parameter to a JSP/ASP application through automatic code rather than being user-initiated. The Web application needs to be refreshed while receiving the parameter.

4.12.2 Using Internet Explorer Frame

Description

Appeon applications can be accessed in an Internet Explorer frame. It is possible to set the Appeon application and the ASP/JSP application in two different frames of the same browser.

Workaround

Take the following steps to build up integration using IE frames:

Step 1 - Divide the IE browser into two frames; one for running the Appeon application and one for running the ASP/JSP application.

Step 2 - Apply the Appeon CommandParm feature for launching the Appeon application from the ASP/JSP application.

Step 3 - Apply the Appeon Hyperlink feature for launching the ASP/JSP application from the Appeon application.

4.12.3 Integration through intermediate n-Tier Server-level solutions

Description

It is possible to pass parameters between Appeon applications and JSP/ASP applications by applying server-level integration. The information that is passed can be stored at any of the tiers in the n-Tier environment, including the Client PC, Application Server, or the Database Server.

Workaround

- The information can be stored and read on the Client PC operating system through the signed and secure Appeon ActiveX.

  Both Appeon and JSP/ASP application read and write a normal Client PC operating system file. The JSP/ASP application needs the ability to access the Client PC operating system file through ActiveX or a Plug-in, etc.

  See the following example for how to store and read the information on the Client PC operating system DLL. Appeon calls to a Client PC operating system DLL file in the same way as in PowerBuilder using the Appeon ActiveX:

```
Var objForm
  set objForm=Server.CreateObject("Scripting.Dictionary")
  set tStream = Server.CreateObject("adodb.stream")
```

- The intermediate information can be stored in a Database Server table.

  Both applications can read and write a normal RDBMS database table. Information such as orders, products, customers, or loans can be stored in a database table. After the information is stored in the database table, other applications can trigger a user-initiated event or simple automatic timer event to get updated information.

- The intermediate information can be stored in a file on the Application Server.

If both the Appeon and JSP/ASP applications call to the same DLL, the developer can make use of functionality provided by DLLs for setting up communication between the applications.

4.13 No Workarounds

Not every unsupported feature can be worked around or need to be worked around. Some features are small or trivial or may not necessarily cause functinoality loss even if they are not modified or worked around.

If the unsupported feature is flagged as "Have to modify" in the Unsupported Feature Aanalysis report while you cannot find a workaround for it, please send a test case to `<support@appeon.com>` for help.

Index