Server Extension User’s Guide
SAP Planning and Consolidation 10.0, version for the Microsoft platform
Copyright

© Copyright 2012 SAP AG. All rights reserved.

No part of this publication may be reproduced or transmitted in any form or for any purpose without the express permission of SAP AG. The information contained herein may be changed without prior notice.

Some software products marketed by SAP AG and its distributors contain proprietary software components of other software vendors.

Adobe, the Adobe logo, Acrobat, PostScript, and Reader are either trademarks or registered trademarks of Adobe Systems Incorporated in the United States and/or other countries. Oracle is a registered trademark of Oracle Corporation.

Microsoft, Microsoft SQL Server, SQL Server Analysis Services, SQL Server Management Studio are registered trademarks of Microsoft Corporation.

SAP, R/3, mySAP, mySAP.com, xApps, xApp, SAP NetWeaver, and other SAP products and services mentioned herein as well as their respective logos are trademarks or registered trademarks of SAP AG in Germany and in several other countries all over the world. All other product and service names mentioned are the trademarks of their respective companies. Data contained in this document serves informational purposes only. National product specifications may vary.

These materials are subject to change without notice. These materials are provided by SAP AG and its affiliated companies ("SAP Group") for informational purposes only, without representation or warranty of any kind, and SAP Group shall not be liable for errors or omissions with respect to the materials. The only warranties for SAP Group products and services are those that are set forth in the express warranty statements accompanying such products and services, if any. Nothing herein should be construed as constituting an additional warranty.
## Contents

Icons in Body Text ........................................................................................................................................... 4
Typographic Conventions ............................................................................................................................... 4
Overview ......................................................................................................................................................... 5
  About Server Extensions .............................................................................................................................. 5
  SDK ............................................................................................................................................................. 5
  Deployment ................................................................................................................................................ 6
Step-by-Step Procedure ............................................................................................................................... 7
  Creating a project and referencing the server extension SDK ................................................................. 7
  Adding status messages .............................................................................................................................. 9
  Making a result fail intentionally ............................................................................................................... 10
  Deploying your custom server extension ............................................................................................... 10
  Making a database connection .................................................................................................................. 10
Icons in Body Text

<table>
<thead>
<tr>
<th>Icon</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>🚨</td>
<td>Caution</td>
</tr>
<tr>
<td>📜</td>
<td>Example</td>
</tr>
<tr>
<td>🚼</td>
<td>Note</td>
</tr>
<tr>
<td>🔄</td>
<td>Recommendation</td>
</tr>
<tr>
<td>🔝</td>
<td>Syntax</td>
</tr>
</tbody>
</table>

Additional icons are used in SAP Library documentation to help you identify different types of information at a glance. For more information, see Help on Help → General Information Classes and Information Classes for Business Information Warehouse on the first page of any version of SAP Library.

Typographic Conventions

<table>
<thead>
<tr>
<th>Type Style</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><em>Example text</em></td>
<td>Words or characters quoted from the screen. These include field names, screen titles, pushbuttons labels, menu names, menu paths, and menu options. Cross-references to other documentation.</td>
</tr>
<tr>
<td><em>Example text</em></td>
<td>Emphasized words or phrases in body text, graphic titles, and table titles.</td>
</tr>
<tr>
<td><strong>EXAMPLE TEXT</strong></td>
<td>Technical names of system objects. These include report names, program names, transaction codes, table names, and key concepts of a programming language when they are surrounded by body text, for example, SELECT and INCLUDE.</td>
</tr>
<tr>
<td><em>Example text</em></td>
<td>Output on the screen. This includes file and directory names and their paths, messages, names of variables and options +arguments, source text, and names of installation, upgrade and database tools.</td>
</tr>
<tr>
<td><em>Example text</em></td>
<td>Exact user entry. These are words or characters that you enter in the system exactly as they appear in the documentation.</td>
</tr>
<tr>
<td>&lt;<em>Example text</em>&gt;</td>
<td>Variable user entry. Angle brackets indicate that you replace these words and characters with appropriate entries to make entries in the system.</td>
</tr>
<tr>
<td><strong>EXAMPLE TEXT</strong></td>
<td>Keys on the keyboard, for example, F2 or ENTER.</td>
</tr>
</tbody>
</table>
Overview

SAP Planning and Consolidation 10.0, version for the Microsoft platform provides server extensions that you can use to extend the functionality of processing dimensions, modifying models, and optimizing models. The three extensions have two methods - PRE and POST - and they aid in the manipulation of imported member table data and other data based on custom logic.

This document provides information on extending standard Planning and Consolidation functionality that you can include in all Server Extension implementations.

About Server Extensions

Server-side extensions in SAP Planning and Consolidation are part of a callback-based framework with which you develop your own business logic at particular junctions of a standard workflow. SAP Planning and Consolidation provides a super class and an SDK, while you provide the plug-in logic. Once your codes are registered, SAP Planning and Consolidation is responsible for their management and execution.

Server Extension SDK

The SAP Planning and Consolidation Server Extension SDK is a public DLL that your code needs to reference for the following items:

- ServerExtension base class
  All of your server extensions should be derived from this, and it includes the methods shown in the next topic.
- Data structure and types
  Parameters of each method and return object

Server Extension Methods in SDK

There are both PRE and POST methods for each server extension. The following table shows the methods of the server extensions:

<table>
<thead>
<tr>
<th>Function</th>
<th>Extension</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>All</td>
<td>GetOrder</td>
<td>This returns the execution order when multiple server extensions are present. Without this extension, ServerExtensionManager would arbitrarily determine the order when the orders are the same.</td>
</tr>
<tr>
<td>Process Dimension</td>
<td>PreDimensionProcess</td>
<td>This method is called after the member data has been imported from the client and before executing the dimension processing steps.</td>
</tr>
<tr>
<td></td>
<td>PostDimensionProcess</td>
<td>This method is called after processing the OLAP dimension and before processing the OLAP cube.</td>
</tr>
<tr>
<td>Modify Model</td>
<td>PreModelProcess</td>
<td>This method is called after reassigning the SQL index.</td>
</tr>
<tr>
<td></td>
<td>PostModelProcess</td>
<td>This method is called after processing the OLAP cube.</td>
</tr>
<tr>
<td>Optimize Model</td>
<td>PreModelOptimization</td>
<td>This method is called before copying the master data.</td>
</tr>
<tr>
<td></td>
<td>PostModelOptimization</td>
<td>This method is called at the end of the optimization function.</td>
</tr>
</tbody>
</table>
Parameters of the Server Extension Methods

The following table shows the parameters of each server extension method:

<table>
<thead>
<tr>
<th>Extension Name</th>
<th>Parameter Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>PreDimensionProcess</td>
<td>string env_name</td>
<td>Current environment name.</td>
</tr>
<tr>
<td></td>
<td>string dimension_name</td>
<td>Current dimension name.</td>
</tr>
<tr>
<td></td>
<td>string source_table</td>
<td>Name of the source member table. Name is PreProcess(dimension).</td>
</tr>
<tr>
<td></td>
<td>DimensionProcessOption</td>
<td>process_option Indicates the current processing option: 0 = IncrementalProcess 1 = FullProcess</td>
</tr>
<tr>
<td></td>
<td>WorksheetOption</td>
<td>worksheet_option Indicates whether processing is performed from the member sheet.</td>
</tr>
<tr>
<td>PostDimensionProcess</td>
<td>string env_name</td>
<td>Current environment name.</td>
</tr>
<tr>
<td></td>
<td>string dimension_name</td>
<td>Current dimension name.</td>
</tr>
<tr>
<td></td>
<td>DimensionProcessOption</td>
<td>process_option Indicates the current processing option: 0 = IncrementalProcess 1 = FullProcess</td>
</tr>
<tr>
<td></td>
<td>WorksheetOption</td>
<td>worksheet_option Indicates whether processing is performed from the member sheet.</td>
</tr>
<tr>
<td>PreModelProcess</td>
<td>string env_name</td>
<td>Current environment name.</td>
</tr>
<tr>
<td></td>
<td>string model_name</td>
<td>Current model name.</td>
</tr>
<tr>
<td></td>
<td>bool process_cube</td>
<td>Indicates whether the OLAP cube will be processed; options are True or False</td>
</tr>
<tr>
<td>PostModelProcess</td>
<td>string env_name</td>
<td>Current environment name.</td>
</tr>
<tr>
<td></td>
<td>string model_name</td>
<td>Current model name.</td>
</tr>
<tr>
<td></td>
<td>bool process_cube</td>
<td>Indicates whether the OLAP cube will be processed; options are True or False</td>
</tr>
<tr>
<td>PreModelOptimization</td>
<td>string env_name</td>
<td>Current environment name.</td>
</tr>
<tr>
<td></td>
<td>string model_name</td>
<td>Current model name.</td>
</tr>
<tr>
<td></td>
<td>OptimizeModelOption</td>
<td>optimize_option Indicates the optimization option, such as Full, Incremental, or Lite.</td>
</tr>
<tr>
<td>PostModelOptimization</td>
<td>string env_name</td>
<td>Current environment name.</td>
</tr>
<tr>
<td></td>
<td>string model_name</td>
<td>Current model name.</td>
</tr>
<tr>
<td></td>
<td>OptimizeModelOption</td>
<td>optimize_option Indicates the optimization option, such as Full, Incremental, or Lite.</td>
</tr>
</tbody>
</table>

Deployment

- Server extension directory on the file share
  This is the directory in which you deploy the server extensions you have developed.

  [FileServer]\Data\Webfolders\AdminTemplates\ServerExtension\
- Working directory on the application server
  SAP Planning and Consolidation copies the file from the deployment directory on the file
  server into this local working directory on the application server.
  \[ApplicationServer\]/Websvr\bin\ServerExtension

- Server Extension SDK
  BPCServerExtensionSDK.dll will be distributed under the Server Extension directory. You
  must add a reference to this DLL to develop your own server extension. The DLL includes
  the ServerExtension base class and the data structures that are used as method
  parameters and return object.

**Step-by-Step Procedure**

**Creating a project and referencing the server extension SDK**

You can develop projects to accomplish a custom task. You must develop the custom server
extensions.NET C# or VB.NET, version 4.0.

1. Run Microsoft Visual Studio 10.0 and create a Class Library type project.
2. Right-click on the **References** tree node and choose **Add Reference**.

![Add Reference](image)

3. Add the BPCServerExtensionSDK.dll.
   A. On the **Browse** tab, browse to the `[ApplicationServer]\Websrvr\bin\` folder in which the BPCServerExtensionSDK.dll exists.
   B. Select `BPCServerExtensionSDK.dll` and choose **OK**.

![Add Reference](image)

4. Add a namespace and inherit your class from the BPCServerExtension.
   A. Manually add the `SAP.BPC.Services.Application` namespace manually. This gives you easy access to the BPCServerExtensionSDK and other data structures that are used as method parameters and return object.
   B. Inherit your class from BPCServerExtension.

```csharp
namespace CustomExtension

public class Class1 : BPCServerExtension
``
5. Override methods with the override keyword. In this example, PostDimensionProcess is used.

```csharp
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using SAP.BPC.Services.Application; // add namespace for Server Extension

namespace CustomExtension
{
    public class Class1 : BPCServerExtension
    {
        public override void PostDimensionProcess(string env_name, string dimension_name, DimensionProcessOption process_option, WorksheetOption worksheet_option)
        {
            System.Diagnostics.Debug.WriteLine("PostDimensionProcess() called. value = "+value);
        }
    }
}
```

6. Implement your own custom logic in the derived method.
   A. Create an instance of ServerExtensionResult that is used for the return object.
   B. Implement your own logic in the try { } block as shown in line 14 in the example below.
   C. In the catch { } block, as shown in line 18, set an exception for the SetException property of the return object.

```csharp
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using SAP.BPC.Services.Application; // add namespace for Server Extension

namespace CustomExtension
{
    public class Class1 : BPCServerExtension
    {
        public override ServerExtensionResult PostDimensionProcess(string env_name, string dimension_name, DimensionProcessOption process_option, WorksheetOption worksheet_option)
        {
            System.Diagnostics.Debug.WriteLine("PostDimensionProcess() called. value = " + value);
            ServerExtensionResult ret = new ServerExtensionResult();
            try
            {
                // implement your own custom logic here
            }
            catch (Exception ex)
            {
                ret.Success = false;
                ret.SetException = ex;
            }
            return ret;
        }
    }
}
```

### Adding status messages

You can show your custom messages through the status dialog in the Admin console, which you can do by adding your message with an Add method like the following:

```csharp
ServerExtensionResult ret = new ServerExtensionResult();
try
{
    // implement your own custom logic here
    // Add your custom message,
    // and they will be shown in Status Dialog in admin client
    ret.messages.Add("Your messages here");
}
catch (Exception ex)
{
    ret.Success = false;
    ret.SetException = ex;
}
return ret;
```
Making a result fail intentionally

Since you can create your own logic using a server extension, you may want your method to fail according to your custom logic without an exception occurring. To do so, set the Success property of the return object to `false`.

```csharp
ServerExtensionResult ret = new ServerExtensionResult();
try
{
    // implement your own custom logic here
    ret.Success = false; // make result to fail
}
```

Deploying your custom server extension

After creating your custom server extension DLL, copy it to the server extension folder `\FileServer\Data\Webfolders\AdminTemplates\ServerExtension\` on the file share. SAP Planning and Consolidation does not create the server extension folder by default; you must create it manually on file share.

To apply your changes after copying your DLL to the file share, IISRESET is required of all application servers.

Making a database connection

SAP Planning and Consolidation does not provide any connection object for MSSQL or SSAS. If you want to create your own logic against MSSQL or SSAS, you need to make the connection in your custom server extension code.

Please be careful to release any connection objects when making a connection to MSSQL or SSAS. Not doing so may impact performance or reliability of your data.