Creating a Callable Object in Group: Miscellaneous
© Copyright 2005 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.

Microsoft, Windows, Outlook, and PowerPoint are registered trademarks of Microsoft Corporation.

IBM, DB2, DB2 Universal Database, OS/2, Parallel Sysplex, MVS/ESA, AIX, S/390, AS/400, OS/390, OS/400, iSeries, pSeries, xSeries, zSeries, z/OS, AFP, Intelligent Miner, WebSphere, Netfinity, Tivoli, and Informix are trademarks or registered trademarks of IBM Corporation in the United States and/or other countries.

Oracle is a registered trademark of Oracle Corporation.

UNIX, X/Open, OSF/1, and Motif are registered trademarks of the Open Group.

Citrix, ICA, Program Neighborhood, MetaFrame, WinFrame, VideoFrame, and MultiWin are trademarks or registered trademarks of Citrix Systems, Inc.

HTML, XML, XHTML and W3C are trademarks or registered trademarks of W3C®. World Wide Web Consortium, Massachusetts Institute of Technology.

Java is a registered trademark of Sun Microsystems, Inc.

JavaScript is a registered trademark of Sun Microsystems, Inc., used under license for technology invented and implemented by Netscape.

MaxDB is a trademark of MySQL AB, Sweden.

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.

Disclaimer
Some components of this product are based on Java™. Any code change in these components may cause unpredictable and severe malfunctions and is therefore expressively prohibited, as is any decompilation of these components.

Any Java™ Source Code delivered with this product is only to be used by SAP’s Support Services and may not be modified or altered in any way.
### Typographic Conventions

<table>
<thead>
<tr>
<th>Type Style</th>
<th>Represents</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Example Text</strong></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>Example text</td>
<td>Emphasized words or phrases in body text, graphic titles, and table titles.</td>
</tr>
<tr>
<td>EXAMPLE TEXT</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>Example text</td>
<td>Output on the screen. This includes file and directory names and their paths, messages, names of variables and parameters, source text, and names of installation, upgrade and database tools.</td>
</tr>
<tr>
<td>Example text</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;Example text&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>EXAMPLE TEXT</td>
<td>Keys on the keyboard, for example, F2 or ENTER.</td>
</tr>
</tbody>
</table>

### Icons

<table>
<thead>
<tr>
<th>Icon</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="image" alt="Caution" /></td>
<td>Caution</td>
</tr>
<tr>
<td><img src="image" alt="Example" /></td>
<td>Example</td>
</tr>
<tr>
<td><img src="image" alt="Note" /></td>
<td>Note</td>
</tr>
<tr>
<td><img src="image" alt="Recommendation" /></td>
<td>Recommendation</td>
</tr>
<tr>
<td><img src="image" alt="Syntax" /></td>
<td>Syntax</td>
</tr>
</tbody>
</table>
**Scenario**

Guided Procedures (GP) provides a set of callable object implementations, which you can use by simply configuring them in the GP Design Time.

There are six types of callable objects in the *Miscellaneous* group.

The table below shows the types with a short description of their functionality, parameters and special configuration aspects.

<table>
<thead>
<tr>
<th>Callable Object</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Read System Properties</td>
<td><strong>Type:</strong> Java callable object for background execution</td>
</tr>
<tr>
<td></td>
<td><strong>Functionality:</strong> Returns a list of system properties</td>
</tr>
<tr>
<td></td>
<td><strong>Class:</strong> com.sap.caf.eu.gp.callobj.system.SystemPropertiesCO</td>
</tr>
<tr>
<td></td>
<td><strong>Container:</strong> caf<del>eu</del>gp~actions</td>
</tr>
<tr>
<td></td>
<td><strong>Output parameters:</strong></td>
</tr>
<tr>
<td></td>
<td>1. <em>sysprops</em> – System properties are returned in this parameter</td>
</tr>
<tr>
<td></td>
<td><strong>Configuration parameters:</strong></td>
</tr>
<tr>
<td></td>
<td>1. <em>prefix</em> – Returns all properties that start with this property prefix</td>
</tr>
<tr>
<td></td>
<td><strong>Result states:</strong></td>
</tr>
<tr>
<td></td>
<td>1. COMPLETED – Reached on successful completion</td>
</tr>
</tbody>
</table>

| Write to System Log          | **Type:** Java callable object for background execution                     |
|                              | **Functionality:** Writes information text to the system log               |
|                              | **Class:** com.sap.caf.eu.gp.callobj.system.SystemLogCO                     |
|                              | **Container:** caf~eu~gp~actions                                           |
|                              | **Input parameters:**                                                     |
|                              | 1. *text* – Text to be added to the system log                            |
|                              | **Configuration parameters:**                                              |
|                              | 1. *severity* – Level of severity. It can be one of the following:         |
|                              |   📝 Information message                                                   |
|                              |   🚸 Warning message                                                       |
|                              |   🚸 Error message                                                         |
|                              | 2. *app_prefix* – Prefix for the information text. This helps to identify  |
|                              |   the log entries of the various applications.                             |
|                              | **Result states:**                                                        |
|                              | COMPLETED – Reached on successful completion                                |
Current State Information (Date and Time, for example)

**Type:** Java callable object for background execution  
**Functionality:** Returns information about the current system state (date, time, language code)  
**Class:** com.sap.caf.eu.gp.callobj.misc.CurrentStateInfoCO  
**Container:** caf-eu-gp-actions  
**Output parameters:**
- CURRENT_DATE - Current date  
- CURRENT_TIME - Current time  
- CURRENTDATETIME - Time stamp, consisting of current date and time  
- CURRENT_DATE_TEXT - Current date, available as string  
- CURRENT_TIME_TEXT - Current time, available as string  
- CURRENTDATETIME_TEXT - Time stamp, consisting of current date and time, available as string  
- CURRENT_TIMESTAMP/rfc1123 - Time stamp (RFC 1123), available as string  
- CURRENT_TIMESTAMP/rfc850 - Time stamp (RFC 850), available as string  
- CURRENT_TIMESTAMP_ANSI - Time stamp (ANSI), available as string  
- CURRENT_TIMESTAMP_R3 - Time stamp (R/3 format), available as string  
- CURRENT_TIMESTAMP_ISO8601 - Time stamp (ISO 8601), available as string  
- LANG_SYS_FALLBACK - System fallback language code in lowercase  
- LANG_SYS_FALLBACK_UCASE - System fallback language code in uppercase  
**Result states:**
- COMPLETED – Reached on successful completion.

Send an E-Mail in Background

**Type:** Java callable object for background execution  
**Functionality:** Sends an e-mail to a specified address  
**Class:** com.sap.caf.eu.gp.callobj.mail.SendMailCO  
**Container:** caf-eu-gp-actions  
**Input parameters:**
- email_adress – The recipient address for the e-mail to be sent  
You can also add any input parameters.  
**Result states:** No result state is set
| Send Notification | **Type:** Java callable object for background execution  
**Functionality:** Sends a notification e-mail to specified process contributors  
**Class:** com.sap.caf.eu.gp.callobj.mail.NotificationCO  
**Container:** caf-eu-gp-actions  
**Input parameters:**  
  - `email_address` – The recipient address for the e-mail to be sent  
You can also add any input parameters.  
**Configuration parameters:**  
  - **MIME** - Technical name of the e-mail template to send. You can choose the same options as in Visual Approval.  
  - **CONTRIBUTOR** - Process contributors who are going to receive notification. You can choose one of the following:  
    - *Current processors* – The members of the current process role receive a message (default)  
    - *Initiator* – Only the initiator receives a message  
    - *Owner* – Only the owner receives a message  
    - *Overseer* – Everyone who has an overseer role receives a message  
    - *Administrator* – Only the administrator receives a message  
    - *All Contributors* – Everyone who has already contributed to the process receives a message  
    - *E-mail Address passed as Context Parameter* – The message is delivered to the address specified in the `email_address` input parameter.  
**Result states:**  
  - **COMPLETED** – Reached on successful completion. |

| Jump Between Callable Objects | **Type:** Java callable object for background execution  
**Functionality:** Used to model the process flow. At runtime, you can define targets for the result states defined by the object.  
**Class:** com.sap.caf.eu.gp.callobj.misc.JumpCO  
**Container:** caf-eu-gp-actions  
**Result states:**  
1. **JUMP_TARGET** – You can define a target for this result state; when it is reached, the process flow continues with the specified action. This result state is set when the object is executed.  
2. **EMPTY_TARGET** – This result state remains open to enable exit from the process flow in case of errors. |

**About This Document**

This tutorial describes how to create the six types of callable objects in the *Miscellaneous* group.
General Prerequisites

Related Documents

[1]. Overview GP Designtime

[2]. <link - Editing SAP System Properties>

[3]. <link – Setting Up Mail Templates>
   http://help.sap.com/saphelp_nw2004s/helpdata/en/ec/591342bc45dd2ce10000000a1550b0/content.htm>

Applicable Release

This tutorial is compatible with the following releases: Beginning with SAP NetWeaver '04s SPS6.

Disclaimer

Any software coding and/or code lines / strings ("Code") included in this documentation are only examples the correctness and completeness of the Code given herein, and SAP shall not be liable for errors or and are not intended to be used in a productive system environment. The Code is only intended better explain and visualize the syntax and phrasing rules of certain coding. SAP does not warrant damages caused by the usage of the Code, except if such damages were caused by SAP intentionally or grossly negligent.

Approaches to Building the Model

You can choose the top-down approach to create the process, block, and action and then insert a new callable object; alternatively you can use the bottom-up approach to start with the callable object without having any other model item.

Top-Down Approach

In this case you create all design time objects (process, block, action) that will make use of this callable object.

For further details on this top-down approach see [1].
1. Choose (Create New) to embed a new callable object in the existing Action.

**Bottom-Up Approach**

In this case you create the callable object first and insert it in the actions and blocks you create later.
The Step-By-Step Solution


You reach the callable object creation screen with either approach. Now select the type of callable object you wish to create. The following sections describe different callable objects in the group Miscellaneous.

2. In the contextual panel (You Can), choose Create Callable Object to start the Callable Object design time.
Create a Read System Properties Callable Object

With this callable object you can query the values of WAS system properties.

3. Choose Miscellaneous→Read System Properties from the Type list.

Enter the following data:

- Name, for example Read System Properties CO
- Description
- Language
- Define a location for your callable object (Folder)

Choose Next.

4. The output parameter is a structure that contains the system properties.

Choose Next.

5. Enter the prefix of the system property to be retrieved and choose Next.

For further information on available system properties see [2].

Go to step 2.10 to finish the tutorial.
Create a Write to System Log Callable Object

This callable object writes a prefixed string to the log of the WAS.

1. Choose Miscellaneous → Write to System Log from the Type list.

   Enter the following data:
   - Name, for example Write to System Log CO
   - Description
   - Language
   - Define a location for the callable object (Folder)

   Choose Next.

2. The pre-defined input parameter Log text contains the text to be logged when the callable object is called.

   Click Next.

3. Define the severity level of the log entry. You can also define a prefix for each log entry.

   Choose Next.

   Go to step 2.10 to finish the tutorial.
Create a Current State Information Callable Object

The current state information callable object returns information about the current system state (date, time, language code).

1. Choose Miscellaneous → Current State Information from the Type list.

   Enter the following data:
   - **Name**, for example **Current State Information CO**
   - **Description**
   - **Language**
   - Define a location for the callable object (Folder)

   Choose Next.

2. The output parameters of the callable object are displayed.

   Choose Next.

   Go to step 2.10 to finish the tutorial.
Create a Send an E-mail in Background Callable Object

This callable object is for testing purposes. It sends a test e-mail message to the given e-mail address.

3. Choose Miscellaneous → Send an e-mail in background from the Type list.

Enter the following data:

- **Name**, for example **Send an e-mail in background CO**
- **Description**
- **Language**
- Define a location for the callable object (Folder)

Choose Next.

4. The only input parameter is the e-mail address of the recipient.

Choose Next.

Go to step 2.10 to finish the tutorial.
Create a Send an Notification Callable Object

This callable object sends a notification by e-mail to a specified recipient. It uses a predefined e-mail template.

5. Choose Miscellaneous → Send an notification from the Type list.
   Enter the following data:
   - Name, for example Send an notification CO
   - Description
   - Language
   - Define a location for the callable object (Folder)
   Choose Next.

6. Input parameter email_address specifies the recipient of the e-mail sent by this callable object.
   You can define further input parameters (with Insert New…), which can be referenced later from the e-mail template.
   Choose Next.

For further information on e-mail templates see document [3].

7. Now select the e-mail template to be used and the mode of recipient determination. To use the address provided by the input parameter email_address, select E-Mail Address passed as Context Parameter.
   Go to step 2.10 to finish the tutorial.
Create a Jump Between Callable Objects Callable Object

You can use this callable object to model the process flow. At runtime, you can define targets for the result states defined by the object.

1. Choose Miscellaneous → Jump between callable objects from the Type list.

Enter the following data:

- **Name**, for example *Jump Between Callable Objects CO*
- **Description**
- **Language**
- Define a location for the callable object (Folder)

Choose **Next**.

Go to step 2.10 to finish the tutorial.

Finishing Callable Object Creation

1. Your callable object is now complete. Choose **Finish and Open** to save and open it. If you started the tutorial using the top-down approach, choose **Finish**.

2. By choosing **Finish and Open** you open the callable object design time.

Click the **Test** tab at the bottom of the page.
3. Enter your values for the input parameters of the callable object.

   Choose Execute.

4. There is no user interface for the Callable Object Execution step, as all six COs are Java classes that are executed in the background.

5. The callable object has been executed and the test results are displayed here.

   If execution was successful, the Result details are displayed. This is the result state described in the table in section “Scenario.”

   The Output Parameters are also displayed.

6. You can activate your callable object by choosing (Activate) at the top of the page or implicitly by activating the process in which this service method is used.