Perceptive Document Composition

Web Services

Version: 5.3

perceptivesoftware

Written by: Product Documentation, R&D
Date: May 2013

Table of Contents

Introduction

The Web Services (MWS) require a fully functional Perceptive Document Composition Studio installation.

They provide third-party applications with various functions of a correspondence system via standard SOAP (Simple Object Access Protocol) messages.

The Web services enable third-party applications to:

- Open and close server-side sessions
- Retrieve information from a repository
  For more detailed information, see the chapter on Repository.
- Initiate and manage correspondence processes
  For more detailed information, see the chapter on Correspondence Processes.

Both conventional Web applications and thin clients (i.e. Windows applications that access the SOAP interface directly) can be developed on the basis of Web services.

For more detailed information, see the chapter on Integration.

Architecture

From a technical point of view, the Web services can be split into three layers:

- The interface layer is responsible for both publishing the Web services and controlling access to them.
  Communication with the outside world is done exclusively over SOAP messages.
- The business logic of correspondence processes is situated in the business layer.
  This is where access to the repository and actual document creation takes place.
  Communication with the data server is also the responsibility of the business layer.
  As in Composition Studio, it is possible to run data selections defined in Xdata or import XML files.
- The state layer is responsible for saving all repository, status, and process-related data, thus making it the uniform basis for text administration and business user functionality.

Modules

WS Process

Definitions

Process Identifier

The **PID** (Process Identifier) is a unique process number.

All process data is stored under this number on the server. As such, it must be a unique system-wide number.

The PID can have a maximum of 40 characters and must be usable as a file name.

Please note:

A process must always be deleted with the function **Process_Delete** in order to free up the resources on the server.

After calling **Process_Delete**, the PID can no longer be used.

Process Options

Options are used to define various settings required for an individual process. They must always be passed as XML wide string.

The structure of the XML string resembles the result XML of the function Process_GetInfo and reflects the internal object structure of the MWS.

See also:

Info XML

Please note:

When setting options, only attributes that are explicitly declared as attributes in this section can be set.

As an XML wide string is used, it is possible to include a number of different settings at once.

These are essentially:

- Set the start command
  See also:
  Function Process_Create, element **mwsprocessmngr**, attribute **onstart**.
  Possible attribute values: Actions, commands and their results
- Set process title and description
  See also:
  Function Process_Close or Options When Closing a Process.
- Set general settings of actions
  - Every action of a process can be configured.
  - There are actions that require a configuration and actions that do not necessarily have to be configured.
  - For all actions (element **mwsaction**), the attributes state and configured can be set.
  - Setting the action property **state** from **finished** to the value **ready**, means the action can be repeated.
    Please note that, in doing so, any already existing results of this action (e.g., documents) and the results of all subsequent actions are deleted.
  - Setting the action property **configured** to true stipulates that configuration of this action is complete.
- Set parameters for data retrieval
  - Parameters for data retrieval belong to the action **DATASELECTION** and must therefore be defined as a child element of this action.
  - The name of the set element for all data retrieval parameters is **selparams**.
  - Parameters are addressed by the attribute **name**. The value must be passed as CDATA.

    Example:

```
...
<selparams>
   <selparam name="Partner-Nr">0815</selparam>
</selparams>
```

```
...
```

- Set manual variables
  - Manual variables belong to the action **CREATEDOCUMENTS** and must therefore be defined as a child element of this action.
  - The name of the set element for all manual variables of a bundle is **manvars**.
  - The name of the set element for all manual variables of a document is **docref**.
  - The document must be addressed by the attribute reference (see: Address of an Object in a Bundle
  - The actual manual variable is described by the name **manvar** in the element and must also be addressed by the attribute **reference**. The value must be passed as CDATA.

  Example:

```
...
<manvars>
   <docref reference="790B817A-B2C3-475E-86E5-08118150EA94">
      <manvar reference="DocumentCollection.790B817A-B2C3-475E-86E5-
 08118150EA94:308DFBC5-F520-414E-ACBD-0A4495C33301_E4437CCD-E71A-4181-
9C6E-0691B26E0C8F">Value 1</manvar>
      <manvar reference="DocumentCollection.790B817A-B2C3-475E-86E5-
08118150EA94:308DFBC5-F520-414E-ACBD-0A4495C33301_EAAC39D9-81BA-476B-
A003-039ABF3618AA" >Value 2</manvar>
   </docref>
</manvars>
...
```

- Set parameters for output management
  Parameters for output management belong to the action **PRINTANDARCHIVE** and must therefore be defined as attributes of this action.

  See also:
  Special Attributes of the Action PRINTDOCUMENTS
- Set variables in the pool SYSTEMPOOL
  Pool variables are independent of actions and can therefore be defined directly below the element **process**.

  See also:
  Get and Set Pool Variables

  As a general rule, the following attributes cannot be set:
  - process_status
  - lastaction
  - lastactionname
  - lasterrorcode
  - lasterrormessage

Address of an Object in a Bundle

Each object in a bundle has a unique reference.

This reference is defined in the attribute **reference**. A reference is comprised of the child IDs of the parent objects and of the object itself.

Example:

790B817A-B2C3-475E-86E5-08118150EA94:308DFBC5-F520-414E-ACBD-0A4495C33301_EAAC39D9-81BA-476B-A003-039ABF3618AA

Actions, Commands and Their Results

A Web service process is essentially defined by its actions.

An action can have a predecessor and can require configuration.

In addition, an action has a status that provides information on action progress.

The MwsActivityStatus process components in a MWS process serve to represent actions.

Web Service Actions when Using the MWS Standard Process

OBJECTSELECTION	Object selection
DATASELECTION	Data retrieval
COMPLETESTRUCTURE	Complete bundle structure, for example, dynamic bundle
CREATEDOCUMENTS	Document generation
FORWARD	Forward process
PRINTDOCUMENTS	Print bundle documents
PRINTANDARCHIVE	Pass bundle documents to Odin

This sequence also defines the dependencies between individual actions:

- Printing is only possible after documents have been created.
- Documents can only be created when it is clear which documents are to be created.
- It is only clear which documents are to be created after data has been retrieved.

In addition to these fixed dependencies between actions, there are other dynamic dependencies that are not always present and, quite frequently, can only be resolved by user interaction.

In other words, in certain cases, actions can only then be executed when the action has been configured.

Dynamic Dependencies

- Data can only be retrieved when selection parameters have been defined.
  The necessity of entering parameters, however, is only then given, when these parameters actually exist.

- Documents can only be created when their manual variables have been set.
  The necessity of entering manual variables, however, is only then given, when manual variables actually exist. This can, of course, change very quickly, depending on which text blocks have been selected.
- A bundle can only be printed directly when a direct printer has been defined.
  The necessity of entering a direct printer, however, is only then given, when direct print is required.
- A bundle can only be forwarded when either the user and/or a user group, where it is to be forwarded to, has been defined.
  If one or more of these dynamic dependencies are present, the respective action can only be successfully started when configuration is complete.
  An action is considered to be configured when the property (see Action Property configured) has been set accordingly.
  Action properties can be passed in the options (see Process Options) on calling the functions **Process_Start** or **Process_Create**.

Process Control

The process control of the Web services stipulates that, basically, every action can always be started.

If the predecessors to the action to be started have not yet been executed, the process control tries to start them. The predecessor actions check always starts as the first action of the process.

If, for example, straight after creating a process **PRINTDOCUMENTS** is started, the process control tries to execute all outstanding actions starting with the first action of the process.

If one of these actions is not configured, a corresponding configuration warning is issued. Otherwise, the action is executed and the next action of the sequence is checked.

The process progresses until the first configuration warning or ends with the execution of the started target action, in this case, **PRINTDOCUMENTS**.

Error Handling

If manual input is required, starting a command results in error code 6000 being returned.

The MWS stores the command that was originally started, so that it can be continued after the missing information has been entered. On the next command start, the stored command can be re-enabled by the fictional command **CONTINUE_LAST_ACTION**.

Example Pseudocode:

```
...
res := Process.Process_Start(FSID, FPID, 'PRINT_AND_ARCHIVE', options, resxml
);
if res = 6000 then
begin
   ok := LetUserConfigActions(options, resxml);
   if ok then
      res := Process.Process_Start(FSID, FPID, 'CONTINUE_LAST_ACTION', option
s, resxml);
end;
...
```

Commands and Their Results

As seen in the last example, the content of the parameter **command** is not consistent with the name of the process actions.

Apart from the two static commands **SET_PARAMETERS** and **CONTINUE_LAST_ACTION**, the commands are derived from the names of the script execution process blocks contained in the MWS process.

Here, the target activity is set by specifying the command of the first script execution process block, whose object name matches the command passed.

If no such script execution process block is found, a corresponding error is thrown to this effect.

The following list applies to the MWS standard process.

- **SET_PARAMETERS**
  This command evaluates and sets the parameters in the options. An action is not started. An empty string is returned if successful.
- **SELECT_OBJECT**
  The action OBJECTSELECTION is started. If successful, the result XML is similar to that of the command **SELECT_DATA**.
  If no bundle is loaded and the corresponding parameters to load the object (bundle) are required, the command returns the error code 6000.
  A description of the structure of the XML can be found in the chapter on Example XML of a Configuration Warning for the Action **FORWARD**.
- **SELECT_DATA**
  The action **DATASELECTION** (data retrieval) is started. If successful, an XML is returned representing the structure of the bundle. This XML corresponds to the result of the procedure **Obj_GetStructure** if an empty string is passed in the parameter RootRef and the value 1 is passed as the parameter MaxLevel.
  The command returns the error code 6000 when parameters are required for starting the selection. In this case, the result XML contains the parameters required.
  A description of the structure of the XML can be found in the chapter on Example XML for **SELPARAMS**.
- **COMPLETE_STRUCTURE**
  The event **AfterDataRetrieval** is triggered and the respective script (if assigned) is executed. The result XML corresponds to that of the command **SELECT_DATA**.
- **CREATE_DOCUMENTS**
  Document creation is started.
  Here, like for the command **SELECT_DATA**, the structure of the bundle is returned when successful.
  The command returns the error code 6000 when manual variables are still required. In this case, the result XML contains the manual variables required.
  A description of the structure of the XML can be found in the chapter on Example XML for MANVARS.
- **FORWARD_DOCUMENTS**
  The action **FORWARD** is executed. If successful, the result XML is similar to that of the command **SELECT_DATA.** The command returns the error code 6000 when the bundle is blocked (the BlockBundle flag in the assigned **AfterCreatingDocuments** script is set to true and the current user is the same user who created the process). In this case, the result XML contains the parameters

required for forwarding. A description of the structure of the XML can be found in the chapter on Example XML for SELPARAMS.

- **PRINT_DOCUMENTS**
  Executes the action PRINTDOCUMENTS**.**
  Prepares transfer to output management. **If successful, the result XML is similar to that of the command** SELECT_DATA**.**
  The command returns the error code 6000 if not all documents have a valid print definition. In this case, the result XML contains the documents whose print definitions are still to be completed by the method Obj_SetStructure.
  More detailed information on the structure of the XML can be found in the chapter on OUTPUTPARAMS.
- **PRINT_AND_ARCHIVE**
  Transfer to output management is performed by the odinProcessImport **component. If successful, the result XML is similar to that of the command** SELECT_DATA**. Otherwise, a corresponding error XML is returned.**
- **CONTINUE_LAST_ACTION**
  If a command was interrupted by the error code 6000, it can be re-enabled, i.e. restarted using this fictional command.
  If no stored command is found, a corresponding error XML is returned.

Note:

To ensure that all commands are executed without interruption, the required configuration parameters can, or respectively, must be passed in the options and the corresponding actions set to **configured**.

For more information, see the chapter on Process Options.

Process Information

A Web service process is comprised of a number of actions, a process manager that controls the processes, and variable pools.

Each of these objects has properties that can change during the course of a process.

An application that integrates the MWS can access information on a process to be able to display it or, respectively, to be able to decide which subsequent actions have to be executed.

To keep network data traffic to a minimum, it is possible to request this information selectively.

To do this, the type of information must be specified.

The properties of the process manager are always returned, irrespective of the information type requested.

The element **mwsprocessmngr** is located below the element **process**.

Example

```
...
   <mwsprocessmngr lasterrorcode="0" lasterrormessage="" lastactionname="" on
start="">
...
```

Attribute	Description

lasterrorcode	Number of the last error
lasterrormessage	Description of the last error
onstart	Name of the action to be started directly after a process is created. **Note** Setting this property at a point in time after the process was created has no effect!
lastactionname	Name of the last action executed

Another important piece of information is the last error that occurred. This information is always returned when an error was thrown, irrespective of the information type requested. The information is located directly below the element **mws**.

Example

```
...
   <error errorcode="6000">error text</error>
...
```

The following information types are currently supported:

PROCESSINFO

Attribute	Description
Title	Title of the process
Description	Description of the process

Actions are listed directly below the process manager in the element **mwsaction**.

Each action has the following attributes:

Attribute	Description
name	Name of the action
lasterrorcode	Number of the last error
lasterrormessage	Description of the last error
state	See Action Property state
configured	See Action Property configured

Extended attributes of the action **DATASELECTION**:

See Special Attributes of the Action DATASELECTION

Extended attributes of the action **PRINTANDARCHIVE**:

See Special Attributes of the Action PRINTDOCUMENTS

POOLVARS

Returns **SYSTEMPOOL** variables: See: Get and Set Pool Variables

MANVARS

Returns manual variables of a bundle.

As manual variables belong to the action **CREATEDOCUMENTS**, they are listed below the corresponding element.

Example

```
...
<mwsaction name="CREATEDOCUMENTS" lasterrorcode="0" lasterrormessage="" state
="0" configured="1">
   <manvars>
      <docref name="Partner_Application_ManVar" title="Partner_Application_Ma
nVar" reference="790B817A-B2C3-475E-86E5-08118150EA94">
         <manvar name="Agent_Forename" reference="DocumentCollection.790B817A
-B2C3-475E-86E5-08118150EA94:308DFBC5-F520-414E-ACBD-0A4495C33301_E4437CCD-
E71A-4181-9C6E-
0691B26E0C8F" title="Agent_Forename" mask="!90/90/0000;1;_" vh_name="" vh_id=
"" vh_system="" input_enabled="Y" data_link="MODUSUSER" data_attribute="Foren
ame" input_required="0" >Valu1</manvar>
         <manvar name="Agent_Surname" reference="DocumentCollection.790B817A-
B2C3-475E-86E5-08118150EA94:308DFBC5-F520-414E-ACBD-0A4495C33301_EAAC39D9-
81BA-476B-A003-
039ABF3618AA" title="Agent_Surname" mask="90/90/0000;1;_," vh_id="" vh_system
="" input_enabled="Y" data_link="MODUSUSER" data_attribute="Surname" input_re
quired="0" >Value2</manvar>
      </docref>
</mwsaction>
...
```

The element **manvars** is a collector node for all manual variables. The manual variables of a document are listed under the collector node docref.

This element has the following attributes:

Attribute	Description
name	Name of the document
title	Document title
reference	See Address of an Object in a Bundle

The element manvar describes a manual variable as follows:

Attribute	Description
name	Name of the manual variable
reference	See Address of an Object in a Bundle
title	Title for the prompt
mask	Input mask (see PDC.Studio documentation)
vh_name	Name of the value help (optional)
vh_id	Object ID of the value help (optional)
vh_system	System ID of the value help (optional)
input_enabled	0 when no manual input should be possible (e.g., if only value selection should be possible). This attribute only exists if no manual input should be possible.
Data	Current value of the manual variable

SELPARAMS

Returns parameters for data retrieval. As these parameters belong to the action DATASELECTION, they are listed below the corresponding element.

Example

```
...
<mwsaction name="DATASELECTION" lasterrorcode="0" lasterrormessage="" state="
0" configured="0">
   <selparams>
      <selections>
         <selection name="Partner_Letter" title="Partner_Letter" active="True
">
            <selparam name="Partnernumber" displayname="PartnerNumberTest" ma
sk=""/>0815</selparam>
         </selection>
      </selections>
   </selparams>
</mwsaction>
...
```

A data retrieval parameter is described by the element selparam:

Attribute	Description
name	Name of the parameter
displayname	Name used for display. If empty, name is used.
mask	Input mask for this parameter. If empty, no input mask s used.
data	Parameter value

OUTPARAMS

Returns parameters for output management. Since these parameters belong to the action **PRINTDOCUMENTS**, they are listed below the corresponding element.

Example

```
...
<mwsaction name="PRINTDOCUMENTS" lasterrorcode="0" lasterrormessage="" state=
"0" configured="0">
   <objects>
      <object title="ApplicationPack" type="D" name="ApplicationPack" index="
07102009-289-tdr-dm1" reference="933110B7-0485-4C62-A50C-0B9D7FC62DB1"/>
   </objects>
</mwsaction>
...
```

All objects are listed where print definitions are still to be completed by the method Obj_SetStructure.

Information on Process Forwarding

This XML is returned from the function **Process_Start** when the **PrintAndArchive** command is executed on a blocked bundle.

A bundle is blocked when the **BlockBundle** flag in the assigned **AfterCreatingDocuments** script is set to true and the current user is the same user who created the process.

To enable a Client to perform automatic forwarding, a destination target can be set in the **AfterCreatingDocuments** script (variables DestUser and/or DestUsergroup).

The destination targets set here are returned from the corresponding attributes so that they can be used as parameters for the function call **Process_Forward**. At least one destination target is required.

If no destination targets are defined in the **AfterCreatingDocuments** script, the Client has to implement the respective target selection process itself.

Example

```
<mws type="FORWARD" version="2">
   <error errorcode="6000">Configuration for action FORWARD required!</error>
   <process id="120de98f-9ec4-469a-b7ad-376651986aa9" systemoid="dm1">
      <title>BasicBindingTest</title>
```

```
      <mwsprocessmngr lasterrorcode="6000" lasterrormessage="Configuration fo
r action FORWARD required!" lastactionname="PRINT_AND_ARCHIVE" onstart="">
         <mwsaction name="FORWARD" lasterrorcode="0" lasterrormessage="" stat
e="0" configured="0"><forwardparams destuser="MyDestUser" destusergroup="MyDe
stGroup"/></mwsaction>
      </mwsprocessmngr>
   </process>
</mws>
```

Attribute	Description
destuser	The user specified in the **AfterCreatingDocuments** script to whom the process is to be forwarded to.
destusergroup	The user group specified in the **AfterCreatingDocuments** script to which the process is to be forwarded to.

Action Property state

Possible values:

Attribute	Description
0	unknown (no status information available)
1	ready (ready to execute)
2	busy (is being executed)
3	finished (execution finished)

Action Property configured

Possible values:

Attribute	Description
0	false (not configured)
1	true (configured)

Special Attributes of the Action PRINTDOCUMENTS

Using the method Obj_SetStructure allows you to set or alter the print definition of a document.

Attribute	Value	Description
print_asktime	0 1	Query print time. Do not query print time.
form		Form name
printer		Printer name
printer_type	OEP OP	Email printer Online printer
printtime	0 1 2	unknown immediately shifted
print_changeable	0 1	Changes to print definition allowed Changes to print definition not allowed

Example

```
<object reference="790B817A-B2C3-475E-86E5-
08118150EA94" printer="OnlinePrinter" printer_type="OP" form="Form1" copycoun
t="0" copytext="" printtime="2" />
```

Special Attributes of the Action DATASELECTION

Attribute	Description
selection	Name of the selection

Example

```
<selections>
   <selection name="Partner_Letter" title="Partner_Letter" active="True">
      <selparam name="Partnernumber" displayname="PartnerNumberTest" mask=""/
>0815</selparam>
   </selection>
   <selection name="Partner_V2" title="Partner_V2" active="False"/>
</selections>
```

Element selections	Collection element of all existing data references/selections in a bundle		
Element selection	Description of a data reference with the attributes:		
	name	Name of the data reference	
	title	Title of the data reference	
	active	true	Active selection
		false	XML selection (data passed to MWS via the method SetData)
Element selparams	Parameter description for active selection with the attributes:		
	name	Name of the parameter	
	displayname	Display name of the parameter	
	mask	Input mask	
	DATA	Current value of the parameter, if already set	

Get and Set Pool Variables

Pool variables can be queried in the context of information retrieval by using the method **Process_GetInfo** (infoType **POOLVARS**).

The **SYSTEMPOOL** and **MODUSUSER** pools are returned.

Only variables of the **SYSTEMPOOL** pool can be set.

Pool variables of these pools are set with the method **Process_Start**.

If only pool variables are to be set and no other action taken, the command **SET_PARAMETERS** must be used.

Example XML

```
<pools>
   <pool name="SYSTEMPOOL"/>
      <poolvar name="MyVariable" type="S">MyVariableValue</poolvar>
   </pool>
</pools>
...
```

Element	Collection node for all pools

pools			
Element pool	Collection node for all pool variables of a specific pool. The attribute **name** contains the name of the pool.		
Element poolvar	Description of a pool variable with the attributes:		
	name	Name of the pool variable	
	type	Type of pool variable	B: Boolean (TRUE or FALSE) L: Longinteger (Number) S: Widestring (any string) F: Float (floating-point number, decimal separator according to regional settings)
	data	Content of the pool variable	

Note

When setting variables, the attribute **type** is optional.

If the attribute is not defined, the type **Widestring** is assumed.

If a type is defined, the content specified in the data area must match the data type.

XML Examples

Options When Closing a Process

```
 <mws version="2">
   <title>Title of the process</title>
   <description>Description of the process</description>
</mws>
```

Options When Forwarding a Process

```
 <mws version="2">
   <title>Title of the process</title>
   <description>Description of the process</description>
</mws>
```

Processlist Options XML

This Options XML can be used to specify which processes should be included in the retrieved process list.

Possible settings are:

1. Returns the processes created by the user (default when empty string passed).

2. Returns the processes forwarded to the user directly or to a group the user belongs to.
3. Returns own processes and forwarded processes. If an empty string is passed for this parameter, only a user's own processes are returned (see 1. above).

```
<mws>
   <processlist>
      <item name="forwarded" include="1" />
      <item name="nonforwarded" include="0"/>
   </processlist>
</mws>
```

The items of the element **processlist** specify which processes are included in the process list returned.

The attributes have the following meaning:

Attribute	Value	Description
name	forwarded	Returns all forwarded processes. These are the processes forwarded either to the user directly, or to a group the user belongs to.
include	0 1	The respective processes are not included in the process list returned. The processes specified by the attribute **name** are included in the list.
name	nonforwarded	Returns all processes that were not forwarded. These are processes that the user created.
include	0 1	The respective processes are not included in the process list returned. The processes specified by the attribute **name** are included in the list.

Special features

If an empty string is passed for the options, or both attributes are disabled (include="0"), the default setting of only processes created by the user is returned.

Extended Optional XML Attributes

The following attributes can be optionally defined to further specify the process list returned:

```
<mws>
   <processlist>
      ...
      <item name="listuser" include="Admin" />
      <item name="shortprocdesc" include="10"/>
      <item name="additionalCondition" include=" AND (MWS_STATUS = 2)" />
      <item name="replacecondition" include="(MWS_STATUS = 2)" />
      ...
   </processlist>
```

```
</mws>
```

The attributes have the following meaning:

Attribute	Value	Description
name	listuser	If this attribute is defined, a respective process list for the specified user is returned. In the list of forwarded processes, only processes that were forwarded to the specified user directly are listed.
include	Username	This attribute is ignored if it contains an empty string.
name	shortprocdesc	If this attribute is specified and a process has a process description, the process element inside the ProcessList XML returned, contains the attribute **shortprocdesc**. This attribute contains the first X characters of the process description. If no process description is found, the corresponding attribute is not included. **Example** `<process id="ade9e10d-ccfb-490f-a5b4-6fb1ac69bfd8" user="Admin" system="dm1" type="" objectname="TestBundle" title="My Title" state="0" created_at="27012010140555" lastsaved="04022010173422" created_by="Admin" shortprocdesc="ShortDescription" />`
include	Number of characters	Number of characters to be returned from the process description. This attribute is ignored if it contains the value 0.
name	additionalcondition	If this attribute is defined, the condition of the select statement is expanded to include the specified condition. In this way, the conditions that apply as a result of other process list options can be individually expanded.
include	The condition to be added	This attribute is ignored if it contains an empty string.
name	replacecondition	If this attribute is defined, the condition of the select statement is completely replaced by the condition specified here. All other specified process list options are overwritten. In this way, it is possible to return exactly those processes in the process list that match the specified condition.

include	New condition	This attribute is ignored if it contains an empty string.

Example of Use:

In this example, all processes that are either finished (status = 2) or locked (status = 1) and have not been edited for at least a day are deleted.

To do this, the process list option **replacecondition** is used to completely redefine the condition for the select statement and, subsequently, to delete all processes returned in the process list.

```
;******************** MLMwsClient ************************************
GetObject("MwsClient","MLMwsClient")
MwsClient.MwsUrl = "http://localhost:8011/mws/mwsprocess"
GetObject("date", "MLDate")
;
; Current date
CurrDateAsInt = date.CurrentDate()
;
; current date minus 1 day
SearchDateInt = date.IncDate(CurrDateAsInt, -1, 0, 0)
;
;SearchDate as string
SearchDateStr = date.Int64ToDateString("dd.MM.yyyy", SearchDateInt)
FreeObject("date")
Protocol(SearchDateStr, 8)
;
; replacecondition --> Status = 1 and last edit less than SearchDate
options = FormatStr("<mws><processlist><item name=""replacecondition"" includ
e=""(MWS_STATUS = 2) OR ((MWS_STATUS = 1) AND (MWS_LASTSAVED &lt;= '%s'))"" /
></processlist></mws>", SearchDateStr)
myXmlResult = ""
res = MwsClient.Process_GetList(0, 100, options)
;
Protocol(res, 8)
myXmlResult = MwsClient.LastXmlResult
Protocol(result, 8)
;
if (myXmlResult <> "")
   GetObject("xml", "MLXmlDocument")
   xml.LoadXml(myXmlResult)
   root = xml.DocumentElement ;rootElement holen
   nodeList = root.SelectNodes("/mws/processlist/process") ;get all nodes of
the process list
   Protocol("Number of processes:  {0}", 8, nodeList.Count)
   nodeListEnumerator = nodeList.GetEnumerator() ; Get an enumerator for the
nodes
   ;
```

```
   ; Iterate over all items...
   readNext = nodeListEnumerator.MoveNext()
   while (readNext)
      node = nodeListEnumerator.Current ; get current item node
      ;
      ;read processid
      attrColl = node.Attributes
      attr = attrColl.GetNamedItem("id")
      processId = attr.Value
      Protocol("ProcessId: {0}", 8, processId)
      if (processId <> "")
         MwsClient.Process_Delete(processId) ; Delete process
         Protocol("Process [{0}] deleted", 8, processId)
      end-if
      readNext = nodeListEnumerator.MoveNext() ; get next item node
   end-while
   FreeObject("xml")
end-if
FreeObject("MwsClient")
;
```

Processlist XML

```
 <mws xmlns:xsi="http://www.w3.org/2001/XMLSchema-
instance" xmlns:xsd="http://www.w3.org/2001/XMLSchema" type="PROCESSLIST" ver
sion="2.0" eof="true" from="0" to="100">
   <processlist>
      <process id="5002f2fb-0d6d-4c30-96f4-
2015785854e8" user="MMueller" system="dm1" type="mmtype" objectname="Applicat
ionPack_Perf" state="1" created_at="13102009114113" lastsaved="13102009114119
" created_by="MMueller" />
      <process id="d381cb40-aa53-4443-9683-
542b8cc557cf" user="MMueller" system="dm1" objectname="ApplicationPack" title
="Adminstrator_Role" state="0" created_at="13102009114006" lastsaved="1310200
9114015" created_by="MMueller" />
   </processlist>
</mws>
```

Description of the element 'process':

Attribute	Description
id	Process-Id
user	User name

system	System OID of the system the bundle was loaded from
type	Specified process type (only contained if type is defined)
objectname	Name of the object loaded
title	Specified process title (only contained if title is defined)
state	Process status
created_at	Date the process was created. Date format: DDMMYYYYHHMMNN
lastsaved	Date of the last user activity for this process. Date format: DDMMYYYYHHMMNN
created_by	Name of the user who created the process.

Object XML

```
<mws type="OBJECT">
   <process id="52a8ce36-0885-4bed-89e6-9ec6357a6af5">
      <object title="ApplicationPack" type="P" name="ApplicationPack" index="02062008-822-
tdc-itl1" reference="">
         <object title="Partner_ApplicationPack" type="D" name="Partner_ApplicationPack" in
dex="02062008-823-tdr-itl1" reference="10DE6C09-79CF-4547-97AA-
6302CF976185" state="1" enabled="1" print_asktime="0" form="Formular1" printer="" printer_t
ype="" printtime="2" print_changeable="1" copycount="0" copy="0" password="modus" doctype="
1" dataid="Partner_V2" filename="Partner_ApplicationPack_10DE6C09-79CF-4547-97AA-
6302CF976185_O.docx" isprotected="1">
         <object title="Confirmation_NewCustomer" type="B" name="Confirmation_NewCustomer"
index="02062008-759-tc-itl1" reference="10DE6C09-79CF-4547-97AA-6302CF976185:7D4AD928-68CD-
4775-A471-35235B9C0B16" state="2" enabled="1" childcount="0"/>
         <object title="If_you_have_more_questions" type="B" name="If_you_have_more_questio
ns" index="02062008-739-tc-itl1" reference="10DE6C09-79CF-4547-97AA-6302CF976185:308DFBC5-
F520-414E-ACBD-0A4495C33301" state="2" enabled="1" childcount="2"/>
         <object title="List_Accounts" type="B" name="List_Accounts" index="02062008-773-
tc-itl1" reference="10DE6C09-79CF-4547-97AA-6302CF976185:D6A6A212-0B4F-4308-A835-
815678A8CC6B" state="2" enabled="1" childcount="1" />
         <object title="Salutation" type="B" name="Salutation" index="02062008-776-tc-
itl1" reference="10DE6C09-79CF-4547-97AA-6302CF976185:81D0CC66-8A1E-4219-BF17-
29668FFA6B32" state="1" enabled="1" childcount="2"/>
      </object>
      <object title="RepCopy" type="D" name="Partner_ApplicationPack" index="02062008-824-
tcr-itl1" reference="3C331B86-8552-4AD6-A242-
29106BFBCA21" state="0" enabled="1" print_asktime="0" form="Formular1" printer="" printer_t
ype="" printtime="2" print_changeable="1" copycount="0" copy="1" password="modus" copytext=
```

```
"RepCopy" originalid"10DE6C09-79CF-4547-97AA-6302CF976185:81D0CC66-8A1E-4219-BF17-
29668FFA6B32"=/>
      </object>
   </process>
</mws>
```

Description of the common attributes of the element 'object':

Attribute	Description
type	Object type P = Bundle D = Document Template B = Text Block
name	Object name
Index	Object index
title	Title (replaces the name displayed to business users)
reference	Reference of the Document in the Bundle See: Address of an Object in a Bundle

Additional attributes for documents, copies and text blocks

Attribute	Value	Description
state	0 1 2 3	Deactivated (object is deactivated, but can be activated) Required (object is activated and cannot be deactivated) Activated (object is activated, but can be deactivated) Disabled (object is deactivated and cannot be activated)
enabled	0	• for a document: when state is deactivated or disabled • for a document copy: when the respective original is not activated (i.e. original has the state deactivated or disabled)
	1	• for a document: when the document is activated (state is activated or required) • for a document copy: when the original is activated and the copy has the state activated or required

Additional attributes for documents / copies

Attribute	Value	Description
print_asktime	0 1	Query print time. Do not query print time.
form		Form name
printer		Printer name
printer_type	OEP OP	Email printer Online printer
printtime	0 1 2	unknown immediately shifted
print_changeable	0 1	Changes to print definition allowed Changes to print definition not allowed
copyCount		Number of real copies
copy	0 1	Original Copy
password		Document password in plain text
doctype	1 2 3 4 5	DynamicWord StaticWord StaticPdf StaticTiff StaticXps

The following attribute is only included for originals:

Attribute	Description
dataid	Name of the assigned TextDataReference that references the selection to be used.

The following attributes are only included for copies:

Attribute	Description

copytext	Text on copy
originalid	Child ID of the original. The value corresponds to the value of the attribute **reference** of the respective original document

The following attributes are only included if the document or copy has already been created:

Attribute	Value	Description
filename		Name of the document created
isprotected	0 1	Write protection is not activated Write protection is activated

Additional text block attributes:

Attribute	Value	Description
childcount	0...n	Number of child objects in the text block

The following attribute is only included for documents / text blocks, if an external document has been inserted into the object from the Client:

Attribute	Value	Description
isexternaldocument	1	An external document is assigned

Embedded Actions in Object XML

If specific object properties are set with the command Set_Structure, then bundle and document actions can also be executed.

Note

For dynamically inserted documents and text blocks, the automatically assigned reference string on insert corresponds to a negative number.

Attribute description

Attribute	Value	Description

Action	includeDoc	Insert a new document into the bundle. (Business User document)
	includeBlock	Dynamic text block insertion into a bundle document
	move	Move existing object (business user documents and text blocks).
	delete	Delete inserted object (business user documents and text blocks).
	loadexternaldocument	Load / remove external document
target_reference		Object reference index in the tree determining the insert position. If this value is not specified, the object is inserted at the end.
position	after before	Object is inserted after the target_reference. Object is inserted before the target_reference.
name		Name of the object to be inserted, or for loadexternaldocument the unique name of the file that has already been uploaded or an empty string to remove the external document again.
reference		For the actions Move/Delete, the reference of the object to be moved/deleted.

		For the action loadexternaldocument, the reference of the object for which the external document is to be loaded/removed.
doc_reference		Reference of the document in which the text block is to be dynamically inserted.

Note

Please ensure that the specified file for the action **loadexternaldocument** with the attribute **name** has already been uploaded to the server via the method **Doc_SetFile_Mime**.

Furthermore, the Client must specify a unique name for the file to prevent an existing file with the same name from being overwritten.

If an external file should be removed again, the action **loadexternaldocument** must be executed using an empty string as value for the attribute name.

Example of Using Actions

Insert a document

```
<mws type="OBJECT">
  <process>
    <object action="includedoc" name="TestDocument" target_reference="" position="after"/>
  </process>
</mws>
```

Move a document

```
<mws type="OBJECT">
  <process>
    <object action="move" reference="-3" target_reference="-1" position="before"/>
  </process>
</mws>
```

Delete a document

```
<mws type="OBJECT">
  <process>
    <object action="delete" reference="-1"/>
  </process>
```

```
</mws>
```

Insert text block (no target object)

```
<mws type="OBJECT">
  <process>
    <object action="includeblock" name="DynBst1" doc_reference="-1" target_reference="-
1" position="unknown" />
  </process>
</mws>
```

Insert text block (before target object)

```
<mws type="OBJECT">
   <process>
      <object action="includeblock" name="DynBlock1" doc_reference="-
1" target_reference="-1" position="unknown" />
   </process>
</mws>
```

Move a text block

```
<mws type="OBJECT">
   <process>
      <object action="move" reference="-1:-2" target_reference="-1:-
1" position="after"/>
   </process>
</mws>
```

Delete a text block

```
<mws type="OBJECT">
   <process>
      <object action="delete" reference="-1:-2"/>
   </process>
</mws>
```

Set external document for a text block

The specified file must have already been uploaded to the server by the Client (method: Doc_SetFile_Mime).

```
<mws type="OBJECT">
  <process>
```

```
     <object action="loadexternaldocument" name="MyExternalBlockDocument.docx
" reference="9BD927E4-39CC-4217-90D1-38C3B15F4CED:35E5A161-D291-4244-84FF-
6CB30F14D1A0" />
  </process>
</mws>
```

Remove external document from a text block

```
<mws type="OBJECT">
   <process>
      <object action="loadexternaldocument" name="" reference="9BD927E4-39CC-
4217-90D1-38C3B15F4CED:35E5A161-D291-4244-84FF-6CB30F14D1A0" />
   </process>
</mws>
```

Setting an external document for a document

The specified file must have already been uploaded to the server by the Client (method: Doc_SetFile_Mime).

```
<mws type="OBJECT">
   <process>
      <object action="loadexternaldocument" name=”MyExternalStaticDocument.pd
f" reference="C6D5FD76-516C-482E-AAA6-9E51BE8BD2E2" />
   </process>
</mws>
```

Removing an external document from a document

```
<mws type="OBJECT">
   <process>
      <object action="loadexternaldocument" name=”" reference="C6D5FD76-516C-
482E-AAA6-9E51BE8BD2E2" />
   </process>
</mws>
```

Info XML

```
<mws type="PROCESSINFO,SELPARAMS,MANVARS,POOLVARS" version="2">
   <process id="a0deab2e-7c05-498e-8c92-0648cff1b3da" systemoid="dm1">
      <title>BasicBindingTest</title>
      <mwsprocessmngr lasterrorcode="0" lasterrormessage="" lastactionname="" onstart="">
         <mwsaction name="FORWARD" lasterrorcode="0" lasterrormessage="" state="0" configur
ed="0"/>
         <mwsaction name="PRINTANDARCHIVE" lasterrorcode="0" lasterrormessage="" state="0"
configured="0"/>
         <mwsaction name="PRINTDOCUMENTS" lasterrorcode="0" lasterrormessage="" state="0" c
onfigured="0"/>
```

```
         <mwsaction name="OBJECTSELECTION" lasterrorcode="0" lasterrormessage="" state="0"
configured="0"/>
         <mwsaction name="DATASELECTION" lasterrorcode="0" lasterrormessage="" state="0" co
nfigured="0"/>
            <selparams>
               <selections>
                  <selection name="Partner_Letter" title="Partner_Letter" active="True">
                     <selparam name="Partnernumber" displayname="PartnerNumberTest" mask=""
/>
                  </selection>
                  <selection name="Partner_V2" title="Partner_V2" active="False"/>
               </selections>
            </selparams>
         </mwsaction>
         <mwsaction name="COMPLETESTRUCTURE" lasterrorcode="0" lasterrormessage="" state="0
" configured="0"/>
         <mwsaction name="CREATEDOCUMENTS" lasterrorcode="0" lasterrormessage="" state="0"
configured="0">
            <manvars>
               <docref name="Partner_Application_ManVar" title="Partner_Application_ManVar"
 reference="790B817A-B2C3-475E-86E5-08118150EA94">
                  <manvar name="Agent_Forename" reference="DocumentCollection.790B817A-
B2C3-475E-86E5-08118150EA94:308DFBC5-F520-414E-ACBD-0A4495C33301_E4437CCD-E71A-4181-9C6E-
0691B26E0C8F" title="Agent_Forename" mask="" vh_name="" vh_id="" vh_system="" input_enabled
="Y" data_link="MODUSUSER" data_attribute="Forename" input_required="0"/>
                  <manvar name="Agent_Surname" reference="DocumentCollection.790B817A-B2C3-
475E-86E5-08118150EA94:308DFBC5-F520-414E-ACBD-0A4495C33301_EAAC39D9-81BA-476B-A003-
039ABF3618AA" title="Agent_Surname" mask="" vh_name="" vh_id="" vh_system="" input_enabled=
"Y" data_link="MODUSUSER" data_attribute="Surname" input_required="0"/>
               </docref>
               <docref name="Partner_Application" title="Partner_Application" reference="93
3110B7-0485-4C62-A50C-0B9D7FC62DB1"/>
            </manvars>
         </mwsaction>
      </mwsprocessmngr>
      <pools>
         <pool name="SYSTEMPOOL"/>
         <pool name="MODUSUSER">
            <poolvar name="USERID" type="S">MMueller</poolvar>
            <poolvar name="USERROLENAMES" type="S">Administrator_Role,User_Role</poolvar>
            <poolvar name="EMAIL" type="S">n.v. mail</poolvar>
            <poolvar name="TELEPHONE" type="S">n.v. telephonenumber</poolvar>
            <poolvar name="USERPRINCIPALNAME" type="S">n.v. userpricipalname</poolvar>
            <poolvar name="DISPLAYNAME" type="S">n.v.
DisplayName</poolvar> DisplayName</poolvar>
         </pool>
      </pools>
   </process>
```

```
</mws>
```

Login XML

```
 <?xml version="1.0" encoding="utf-16"?>
<mws>
   <loginparams>
      <loginparam name=""alias"">USERNAME</loginparam>
   </loginparams>
</mws>
```

The user name and password passed at login are used for authentication.

If an alias is passed, the user information (profile attributes/rights) of the user are loaded and used in the MWS session.

UserInfo XML

```
 <mws xmlns:xsi="http://www.w3.org/2001/XMLSchema-
instance" xmlns:xsd="http://www.w3.org/2001/XMLSchema" type="USERINFO" versio
n="2.0">
   <userinfo>
      <attributes>
         <attribute name="UserId" value="TestUser" />
         <attribute name="UserRoleNames" value="Administrator_Role" />
      </attributes>
      <objectpermissions>
         <objectpermission objecttype="DataProviderDefinition" objectactionpe
rmission="Create Read Update Delete" />
         <objectpermission objecttype="AliasTable" objectactionpermission="Cr
eate Read Update Delete" />
         <objectpermission objecttype="TextComponent" objectactionpermission=
"Create Read Update Delete" />
         <objectpermission objecttype="TextComponent" objectactionpermission=
"Create Read Update Delete" />
         <objectpermission objecttype="Enclosure" objectactionpermission="Cre
ate Read Update Delete" />
         <objectpermission objecttype="TextDocumentCollection" objectactionpe
rmission="Create Read Update Delete" />
         <objectpermission objecttype="DataObject" objectactionpermission="Cr
eate Read Update Delete" />
         <objectpermission objecttype="TextDocument" objectactionpermission="
Create Read Update Delete" />
         <objectpermission objecttype="EMailPrinter" objectactionpermission="
Create Read Update Delete" />
         <objectpermission objecttype="Form" objectactionpermission="Create R
ead Update Delete" />
         <objectpermission objecttype="Envelope" objectactionpermission="Crea
te Read Update Delete" />
```

```
         <objectpermission objecttype="DataObjectQuery" objectactionpermissio
n="Create Read Update Delete" />
         <objectpermission objecttype="DataObjectScript" objectactionpermissi
on="Create Read Update Delete" />
         <objectpermission objecttype="MwsProcess" objectactionpermission="Cr
eate Read Update Delete" />
         <objectpermission objecttype="LookupTable" objectactionpermission="C
reate Read Update Delete" />
         <objectpermission objecttype="Printer" objectactionpermission="Creat
e Read Update Delete" />
         <objectpermission objecttype="Folder" objectactionpermission="Create
 Read Update Delete" />
         <objectpermission objecttype="OutsourcingPrinter" objectactionpermis
sion="Create Read Update Delete" />
         <objectpermission objecttype="PostageDefinition" objectactionpermiss
ion="Create Read Update Delete" />
         <objectpermission objecttype="Profile" objectactionpermission="Creat
e Read Update Delete" />
         <objectpermission objecttype="Process" objectactionpermission="Creat
e Read Update Delete" />
         <objectpermission objecttype="Role" objectactionpermission="Create R
ead Update Delete" />
         <objectpermission objecttype="RoleMapping" objectactionpermission="C
reate Read Update Delete" />
         <objectpermission objecttype="Selection" objectactionpermission="Cre
ate Read Update Delete" />
         <objectpermission objecttype="Script" objectactionpermission="Create
 Read Update Delete" />
         <objectpermission objecttype="StandardProcess" objectactionpermissio
n="Create Read Update Delete" />
         <objectpermission objecttype="SimpleTextContainer" objectactionpermi
ssion="Create Read Update Delete" />
         <objectpermission objecttype="TransferJob" objectactionpermission="C
reate Read Update Delete" />
         <objectpermission objecttype="ValueHelp" objectactionpermission="Cre
ate Read Update Delete" />
         <objectpermission objecttype="CustomUI" objectactionpermission="Crea
te Read Update Delete" />
         <objectpermission objecttype="XsdSelection" objectactionpermission="
Create Read Update Delete" />
      </objectpermissions>
      <permissions>
         <permission>Administrate</permission>
         <permission>CommonSettings</permission>
         <permission>MwsInsertBlock</permission>
         <permission>MwsClearCache</permission>
         <permission>MwsDataSelection</permission>
         <permission>MwsInsertDocument</permission>
         <permission>MwsCreateDocuments</permission>
```

```
<permission>CanPrintInPreview</permission>
<permission>MwsPrintAndArchive</permission>
<permission>MwsSetBundlePrinter</permission>
<permission>MwsSetDocumentPrinter</permission>
<permission>MwsSetDocumentPrinterWizzard</permission>
<permission>MwsDownloadDocument</permission>
<permission>MwsLoadExternalBlockFile</permission>
<permission>MwsLoadExternalDocumentFile</permission>
<permission>MwsAdministrate</permission>
<permission>MwsViewOpenEnvelope</permission>
<permission>MwsViewProcess</permission>
<permission>MwsViewStack</permission>
<permission>MwsViewOpenJob</permission>
<permission>MwsShowNavigator</permission>
<permission>MwsChangeFolder</permission>
<permission>MwsChangeSystem</permission>
<permission>ExecuteModusStudio</permission>
<permission>ShowNavigator</permission>
<permission>ShowTypeFilter</permission>
<permission>OdinJobRemove</permission>
<permission>OdinJobSetPrinter</permission>
<permission>OdinJobSetState</permission>
<permission>OdinStackEnvelopeRemove</permission>
<permission>OdinProcessSuspend</permission>
<permission>OdinProcessUnlock</permission>
<permission>OdinProcessDelete</permission>
<permission>OdinProcessSetBack</permission>
<permission>OdinStackSetBackToStreaming</permission>
<permission>OdinStackEdit</permission>
<permission>OdinStackUnlock</permission>
<permission>OdinStackFree</permission>
<permission>OdinStackDelete</permission>
<permission>OdinStackLock</permission>
<permission>OdinOpenJobCreateStack</permission>
<permission>OdinProcessCreateStack</permission>
<permission>OdinStackSetBack</permission>
<permission>OdinStackSetBackTo</permission>
<permission>ChangeFolder</permission>
<permission>MwsManualForwardProcess</permission>
<permission>MwsProcessList</permission>
<permission>MwsProcessListFromDefinedUser</permission>
<permission>ModifyFontFace</permission>
<permission>ModifyFontColour</permission>
<permission>ModifyFontStyleBold</permission>
<permission>ModifyFontStyleItalic</permission>
<permission>ModifyFontStyleUnderline</permission>
```

```
         <permission>ModifyFontSize</permission>
         <permission>Search</permission>
         <permission>ChangeSystem</permission>
         <permission>SynchronizeSystem</permission>
         <permission>TransferSystem</permission>
         <permission>Test</permission>
         <permission>ModifyFontAlignment</permission>
         <permission>MwsForwaredProcessList</permission>
      </permissions>
   </userinfo>
</mws>
```

Error XML

```
<?xml version="1.0" encoding="utf-16"?>
<mws xmlns:xsi="http://www.w3.org/2001/XMLSchema-
instance" xmlns:xsd="http://www.w3.org/2001/XMLSchema" type="ERROR" version="
2.0">
   <error location="functionname" errorcode="numeric Code">error message</err
or>
</mws>
```

Functions

Clear Cache

Clears the cache of the RepositoryRuntimeService.

Syntax

ClearCache(string sessionId, out string xmlResult)

Parameter

sessionId	Valid session identifier (SessionId)
xmlResult	Empty when okay, otherwise MwsError

Return value

0	OK
<>0	Error

Doc_GetFile_Mime

Enables the download of a document already created on the server. The document is returned as a base64 coded string.

Syntax

```
Doc_GetFile_Mime(string sessionId, string processId,
                 string fileName, out string doc,
                 out string xmlResult )
```

Parameter

sessionId	Valid session identifier (SessionId)
processId	ID of the process whose business object is to be managed.
fileName	Name of the file to be returned.
doc	Base64 coded string, representing the content of the file.
xmlResult	Empty when okay

Return value

0	OK
<>0	Error

Doc_SetFile_Mime

Uploads a file (base64 coded string) to the server.

Syntax

```
Doc_SetFile_Mime(string sessionId, string processId,
                 string fileName, string doc,
                 out string xmlResult
```

Parameter

sessionId	Valid session identifier (SessionId)
processId	ID of the process whose business object is to be managed.
fileName	Name of the file whose content is to be written.
doc	Base64 coded string, representing the content of the file.

xmlResult	Empty when okay

Return value

0	OK
<>0	Error

Login

This function is used to log in to the server. The login function must be executed once when commencing a server session and is valid for the duration of the session.

The session ID (Session Identifier) returned must be used for all subsequent calls.

Syntax

```
Login(string userName, string password, string code,
      string options, out string sessionID,
      out string xmlResult )
```

Parameter

userName	User name
password	Password in plain text. Only used when the parameter **code** contains an empty string
code	Encrypted password (created previously using Encoder.exe)
options	Login Xml, see: Login XML
sessionID	Returns the session ID to be used for subsequent calls
xmlResult	The respective UserInfo Xml when login execution was successful, otherwise Mws Error XML

Return value

0	OK
<>0	Error

Logout

This function is used to log out of the session Please note, a session must always be ended with a logout in order to delete it from the server. If this rule is not observed, the datasets are left in the respective table.

Syntax

Logout(string sessionId, out string xmlResult)

Parameter

sessionId	Valid session ID of the session to be logged out of.
xmlResult	Empty when okay, otherwise MwsError

Return value

0	OK
<>0	Error

Obj_GetStructure

Returns the structure of the referenced object.

Syntax

```
int Obj_GetStructure(string sessionId, string processId,
                     string rootRef,int maxLevel,
                     out string xmlResult)
```

Parameter

sessionId	Valid session identifier (SessionId)	
processId	ID of the process of which the object is being referenced.	
rootRef	Object reference within the process	
maxLevel	-1	everything
	0	just the object
	<> 0	the structure including all child elements at the n-th level
xmlResult	Object XML depending on the component	

Return value

0	OK
<>0	Error

Obj_SetStructure

Allows you to set properties of already existing objects or use commands to execute bundle and document actions. For more information, see the chapter on Embedded Actions in Object XML.

Syntax

```
Obj_SetStructure(string sessionId,string processId,
                 string content,string rootRef,int maxLevel,
                 out string xmlResult)
```

Parameter

sessionId	Valid session identifier (SessionId)
processId	ID of the process of which the object is being referenced.
content	The business object information to be set.
rootRef	Object reference whose structure is to be returned after the call
maxLevel	Determines how many levels are to be returned (-1 = all)
xmlResult	If okay, the structure of the business object

Return value

0	OK
<>0	Error

Obj_Toggle

Switches the status of an object and returns the modified structure. Please note that toggling objects based on groups and actions can lead to other objects being automatically toggled as well.

The function returns the modified structure after all respective dependent toggle actions have been executed.

Syntax

```
Obj_Toggle(string sessionId,string processId,string objRef,
```

```
string rootRef,int maxLevel,out string xmlResult)
```

Parameter

sessionId	Valid session identifier (SessionId)
processId	ID of the process of which the object is being referenced.
objRef	Reference of the object to be toggled.
rootRef	Object reference whose structure is to be returned after the call
maxLevel	Determines how many levels are to be returned (-1 = all)
xmlResult	The structure of the object, if successful.

Return value

0	OK
<>0	Error

Ping

Returns a standard message for test purposes.

Syntax

```
Ping()
```

Parameter

-

Return value

Standard message **MWS Process Service**.

Process_Close

Closes a process with the passed in status.

Syntax

```
Process_Close(string sessionId,string processId,int status,string options,out
 string xmlResult)
```

Parameter

<table>
<tr><td>sessionId</td><td>Valid session identifier (SessionId)</td></tr>
<tr><td>processId</td><td>ID of the process to be closed.</td></tr>
<tr><td>status</td><td>Status to be set:<table>
<tr><td>0</td><td>Save process
Handling of processes with status 0 can be resumed.</td></tr>
<tr><td>1</td><td>The process is currently being edited.
The value 1 is used internally and should not be passed in, as it causes an error.</td></tr>
<tr><td>2</td><td>The process was completed successfully.
The service MWS_Assistant automatically deletes processes saved with the status 2.</td></tr>
<tr><td>3</td><td>The process was faulty, for example, user abort.
Defective or faulty processes are not automatically deleted by the service MWS_Assistant.
The Document Composition Client explicitly deletes the processes aborted by a user.</td></tr>
<tr><td>Other values</td><td>Process is closed.</td></tr>
</table></td></tr>
<tr><td>options</td><td>Options XML for closing a process with specification of title and description.
Example XML for setting title and description:
<?xml version="1.0" encoding="utf-16"?>
<mws>
<process>
<title>My Title</title>
<description>My description</description>
</process>
</mws></td></tr>
<tr><td>xmlResult</td><td>Empty when okay</td></tr>
</table>

Return value

0	OK

<>0	Error

Process_Create

Creates a new MWS process.

To create a new process, the following actions are executed on the server:

- Load the configured process that defines the sequence of MWS process steps from the repository (if not already done).
- Create a directory for the data of the respective document creation process.
- Create a dataset for the process in the database table MWS_Processes.
- Evaluate options, if an Options XML was passed (parameter **options**).
- Execute the **OnSetOptionsScript** of the MWS process, if assigned.
- Execute the **OnNewProcess** script of the MWS process, if assigned.
- Execute a start command if passed to the Options XML.

If an error occurs during the execution of this method, the process data created (files and DB dataset) is deleted.

Syntax

```
Process_Create(string sessionId,ref string processId,
               string mSystem,string mType,string mTitle,
               string objectIndex,string objectName,
               string objectType,string objectData,
               string options,out string xmlResult)
```

Parameter

sessionId	Valid session identifier (SessionId)
processId	If the process ID is not passed in, a new one is created.
mSystem	System OID of the system the object to be loaded resides in.
mType	Free text to define the process type.
mTitle	Free text to define a title for the process.
objectIndex	Index of the object to be loaded (if this parameter is used, the name is ignored).
objectName	Name of the object to be loaded (is ignored, if the index is specified).
objectType	Type of object to be loaded (currently only P = bundle)

objectData	Optional XML data for a bundle. The data will be assigned to the first passive selection in the script OnNewStart.
options	Options XML for the MWS process.
xmlResult	If the process was created successfully, either the bundle structure or process information is returned. The bundle structure is returned if a start command is passed to the options.

Return value

0	OK
<>0	Error

Process_Delete

Deletes an open process and frees its server resources.

Syntax

```
Process_Delete(string sessionId,string processId,
               out string xmlResult)
```

Parameter

sessionId	Valid session identifier (SessionId)
processId	ID of the process to be deleted.
xmlResult	Empty when OK, otherwise MwsError

Return value

0	OK
<>0	Error

Process_Forward

Forwards a process.

Syntax

```
Process_Forward(string sessionId,string processId,
                string destUser,string destUsergroup,
                string options,out string xmlResult)
```

Parameter

sessionId	Valid session identifier (SessionId)
processId	ID of the process to be forwarded.
destUser	Name of the target user the process is to be forwarded to.
destUsergroup	Name of the target group the process is to be forwarded to.
options	Close options XML (title and description)
xmlResult	Empty when okay

Return value

0	OK
<>0	Error

Process_GetInfo

Returns information about the process.

Syntax

```
int Process_GetInfo(string sessionId,string processId,
                    string infoType,string options,
                    out string xmlResult)
```

Parameter

sessionId	Valid session identifier (SessionId)
processId	ID of the process, for which the information is to be returned.
infoType	Comma separated string containing the information to be retrieved. This information is, firstly, the predefined types: PROCESSINFO POOLVARS and, secondly, MWS process dependent and as such is loaded and evaluated by the OnGetInfo script.
options	Is not currently supported.

xmlResult	ProcessInfo XML, if OK

Return value

0	OK
<>0	Error

Process_GetLastError

Returns the last error of a process.

Syntax

```
Process_GetLastError(string sessionId,string processId,
                     out string xmlResult)
```

Parameter

sessionId	Valid session identifier (SessionId)
processId	ID of the process, for which the last error is to be retrieved.
xmlResult	If successful, returns the last error XML.

Return value

0	OK
<>0	Error

Process_GetList

Returns a list of all currently existing processes.

Syntax

```
Process_GetList(string sessionId,int startAt,int max,
                string options,out string xmlResult)
```

Parameter

sessionId	Valid session identifier (SessionId)

startAt	Start index
max	Maximum number of processes to be returned.
options	This parameter can be used to specify which processes should be included in the retrieved process list. **Possible settings are:** 1. Returns the processes created by the user (default when empty string passed). 2. Returns the processes forwarded to the user directly or to a group the user belongs to. 3. Returns own processes and forwarded processes. If an empty string is passed for this parameter, only a user's own processes are returned (see 1. above).
xmlResult	An MwsProcessList XML when okay, otherwise an MwsError.

Return value

0	OK
<>0	Error

Process_Open

Resumes processing of a process and sets the user name to that of the current user.

Syntax

```
Process_Open(string sessionId,string processId,string options,
             out string xmlResult)
```

Parameter

sessionId	Valid session identifier (SessionId)
processId	Process ID to be opened.
options	Not currently used
xmlResult	If OK, the object structure of the business object

Return value

0	OK

<>0	Error

Process_SetData

Sets XML data for the business object.

Syntax

```
Process_SetData(string sessionId,string processId,string dataType,
                string data,out string xmlResult)
```

Parameter

sessionId	Valid session identifier (SessionId)
processId	ID of the process, for which data is to be set.
dataType	Name to be assigned to the data. Currently refers to the name of the data reference.
data	The XML data to be set.
xmlResult	Empty when okay

Return value

0	OK
<>0	Error

Process_Start

Starts a process action.

Syntax

```
Process_Start(string sessionId,string processId,string command,
              string options,out string xmlResult)
```

Parameter

sessionId	Valid session identifier (SessionId)

processId	Process ID of the process where the action is to be started.
command	This parameter contains the command to be passed. The names of the commands are derived from the Invoke Activities script of the MWS process.
options	This parameter can be used to set various options of the process. Can be used to set process activities, pass in variables, etc.
xmlResult	The structure of the bundle, if successful. **Example** `<?xml version="1.0" encoding="utf-16"?>` `<mws type="OBJECT">` `<process id="b9187257-812a-4e88-afa6-facb621dd2eb">` `<object title="ApplicationPack" type="P" name="ApplicationPack" index="02062008-822-tdc-itl1" reference="">` `<object title="Partner_ApplicationPack" type="D" name="Partner_ApplicationPack" index="02062008-823-tdr-itl1" reference="10DE6C09-79CF-4547-97AA-6302CF976185" state="1" enabled="1" print_asktime="0" form="DIN_A4_80g/qm" printer="" printer_type="" printtime="2" print_changeable="0" copycount="0" copy="0" password="modus" filename="Partner_ApplicationPack_10DE6C09-79CF-4547-97AA-6302CF976185_O.docx" />` `<object title="pagenum" type="D" name="pagenum" index="23062009-591-tdr-dm" reference="A7DE4DF9-00A4-41FC-8A84-05BB1D9D893D" state="2" enabled="1" print_asktime="0" form="DIN_A4_80g/qm" printer="" printer_type="" printtime="2" print_changeable="1" copycount="0" copy="0" password="" filename="pagenum_A7DE4DF9-00A4-41FC-8A84-05BB1D9D893D_O.docx" />` `<object title="RepCopy" type="D" name="Partner_ApplicationPack" index="02062008-824-tcr-itl1" reference="3C331B86-8552-4AD6-A242-29106BFBCA21" state="2" enabled="1" print_asktime="0" form="DIN_A4_80g/qm" printer="" printer_type="" printtime="2" print_changeable="0" copycount="0" copy="1" password="modus" copytext="Copy for Representative" />` `<object title="AgentCopy" type="D" name="Partner_ApplicationPack" index="02062008-825-tcr-itl1" reference="2572D52B-D2EC-4864-81A3-A4017CD16F53" state="2" enabled="1" print_asktime="0" form="DIN_A4_80g/qm" printer="FinePrint" printer_type="OP" printtime="1" print_changeable="1" copycount="0" copy="1" password="modus" copytext="Copy for Agent" />` `</object>` `</process>` `</mws>`

Return value

0	OK

<>0	Error

WS Repository

XML Examples

Systemdef XML

```
<?xml version="1.0" encoding="UTF-8"?>
<mws xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema" type="SYSTEMDEF" version="2.0">
   <systemdef name="Odin_Standard" description="" systemoid="odv"/>
</mws>
```

MwsItemInfo XML

```
<mws xmlns:xsi="http://www.w3.org/2001/XMLSchema-
instance" xmlns:xsd="http://www.w3.org/2001/XMLSchema" type="ITEMINFO" versio
n="2.0">
   <iteminfo>
      <IsProtected>false</IsProtected>
      <IsReadOnly>false</IsReadOnly>
      <ObjectName>Partner_Application</ObjectName>
      <ObjectId>02062008-805-td-itl1</ObjectId>
      <DatabaseId>294</DatabaseId>
      <ObjectType>TextDocument</ObjectType>
      <LastModifiedAt>2010-09-01T11:28:26.617</LastModifiedAt>
      <LastModifiedBy>MEGATRON\Administrator</LastModifiedBy>
      <CreatedAt>2008-06-02T13:05:51.937</CreatedAt>
      <CreatedBy>DT_MMUELLER\MMueller</CreatedBy>
      <LockedBy />
      <LockedAt xsi:nil="true" />
      <Context>None</Context>
   </iteminfo>
</mws>
```

MwsSystemList XML

```
<mws xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema" type="SYSTEMLIST" version="2.0">
   <systems>
      <system name="MWS_Standard" description="" systemoid="mws"/>
      <system name="Batch_Standard" description="" systemoid="modb"/>
      <system name="User_Standard" description="" systemoid="user"/>
      <system name="Odin_Standard" description="" systemoid="odv"/>
      <system name="TestSystem" description="" systemoid="ts"/>
```

```
      <system name="Test" description="" systemoid="test"/>
   </systems>
</mws>
```

MwsNavigTree XML

```
<mws xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema" type="NAVIGTREE"
version="2.0" systemname="it2">
   <folder name="ImportTest_2" type="A" path="/" dbid="0">
      <folders>
         <folder name="Test1" type="A" path="/Test1/" dbid="126">
            <folders>
               <folder name="Test1_1" type="A" path="/Test1/Test1_1/" dbid="2
371"/>
            </folders>
         </folder>
         <folder name="Test2" type="A" path="/Test2/" dbid="897"/>
      </folders>
   </folder>
</mws>
```

MwsNavigList XML

This XML is used to navigate through the repository.

```
<mws xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema" type="NAVIGLIST" version="2.0">
   <foldercontent path="" filter="A,P,D,B" systemname="itl2">
      <items>
         <item name="Paragraph1" objectid="14102009-1768-tc-itl2"
         type="B" />
         <item name="Paragraph2" objectid="14102009-1769-tc-itl2"
         type="B" />
         <item name="KeepParagraphsTogether" objectid=
         "14102009-1767-tc-itl2" type="B" />
         <item name="KeepParagraphsTogetherTables"
         "14102009-1770-tc-itl2" type="B" />
         <item name="Sender_Date" objectid="14102009-1759-tc-itl2"
         type="B" />
         <item name="Partner_Application" objectid="14102009-1790-td-itl2"
         type="D" />
         <item name="Partner_Application_XDOC" objectid="14102009-1799-td-
itl2"
         type="D" />
         <item name="XML-Testdocument" objectid="14102009-1798-td-
itl2" type="D" />
```

```
         <item name="Application" objectid="14102009-1807-tdc-
itl2" type="P" />
         <item name="Application_XDOC" objectid="14102009-1812-tdc-
itl2" type="P" />
      <items>
</mws>
```

Attributes of the element foldercontent

parentpath	Parent path
path	Path for which objects are to be listed
filter	Filter used
user	ID of the current user
systemname	Name of the system

Attributes of the element item

type	Object type
name	Object name

MwsFolderContent XML

```
<mws xmlns:xsi="http://www.w3.org/2001/XMLSchema-
instance" xmlns:xsd="http://www.w3.org/2001/XMLSchema" type="FOLDERCONTENT" v
ersion="2.0">
<foldercontent systemname="it2">
   <items>
      <item name="Test1_1" objectid="19102009-1230-f-it2" type="A"/>
      <item name="TextBlock" objectid="it2-tc-17" type="B"/>
      <item name="TextBlockWithVariable" objectid="it2-tc-18" type="B"/>
   <items>
</foldercontent>
</mws>
```

MwsLookupResult XML

For Rep_GetLookupValue (contains MwsLookupValue xml):

```
<mws xmlns:xsi="http://www.w3.org/2001/XMLSchema-
instance" xmlns:xsd="http://www.w3.org/2001/XMLSchema" type="LOOKUPRES" versi
on="2.0" reference="VariablenVorgabe|15042008-666-lt-
it2|447|LookupTable|None|dm1">
```

```
   <lookup aliastable="DefaultVariables" alias="Var1">Wert1|"\(999 99\) 99999
;1;_"</lookup>
</mws>
```

For Rep_GetLookUpObjects (contains MwsFolderContent xml):

```
<mws xmlns:xsi="http://www.w3.org/2001/XMLSchema-
instance" xmlns:xsd="http://www.w3.org/2001/XMLSchema" type="LOOKUPRES" versi
on="2.0" reference="SystemAliasTable|15042008-665-at-
it2|446|AliasTable|None|dm1">
   <foldercontent systemname="dm1">
      <items>
         <item name="DefaulVariables" objectid="15042008-666-lt-
it2" type=""/>
      <items>
   </foldercontent>
</mws>
```

MwsDataProviderDefinition XML

```
<mws xmlns:xsi="http://www.w3.org/2001/XMLSchema-
instance" xmlns:xsd="http://www.w3.org/2001/XMLSchema" type="DATAPROVIDERDEF" version="2.0"
>
   <providerxml>&lt;?xml version="1.0" encoding="utf-
16"?&gt;&lt;ms:dataproviderdefinitiontree xmlns:ms="http://www.modus-
suite.de/2006/XMLSchema"&gt;&lt;dataproviderdefinition objectid="09072009-28-dpd-
mws" objectname="All_SQL_MWS_Sessions" description="" xml="" category="" rowsperpage="50" t
ables="MWS_SESSION" dbalias="MWS|21012008-1-dba-
mws|2|DbAlias|DatabaseAlias|mws" protected="False"&gt;&lt;sqlcolumns objectname="SqlColumnC
ollection" description=""&gt;&lt;sqlcolumn objectname="MWSS_SESSION_ID" description="" titl
e="" dbcolumnname="MWSS_SESSION_ID" size="-
1" isvisible="True" /&gt;&lt;sqlcolumn objectname="MWSS_CREATED" description="" title="" db
columnname="MWSS_CREATED" size="-
1" isvisible="True" /&gt;&lt;sqlcolumn objectname="MWSS_USERNAME" description="" title="" d
bcolumnname="MWSS_USERNAME" size="-
1" isvisible="True" /&gt;&lt;sqlcolumn objectname="MWSS_MODUSUSERINFO" description="" title
="" dbcolumnname="MWSS_MODUSUSERINFO" size="-
1" isvisible="True" /&gt;&lt;/sqlcolumns&gt;&lt;/dataproviderdefinition&gt;&lt;/ms:dataprov
iderdefinitiontree&gt;
   </providerxml>
</mws>
```

MwsValueHelpDefinition XML

```
<mws type="VALUEHELP" version="2.0" valuehelptype="3"
reference="Date_Time|27102008-182-vhl-dm|15|ValueHelp|None|odv">
   <definition>&lt;datetimepicker xmlns:xsi="http://www.w3.org/2001/XMLSchema
-
instance" xmlns:xsd=http://www.w3.org/2001/XMLSchema customformat="dd.MM.yyyy
 HH:mm:ss" format="Custom" /&gt;
```

```
   </definition>
</mws>
```

MwsNavigTree Options XML

```
<mws>
   <navigtree>
      <item name="startfoldername" value="[OrdnerName]" />
   </navigtree>
</mws>"
```

Defining a folder name specifies the folder from which to start listing a directory tree / folder hierarchy.

If an empty string is passed, the specification is ignored, i.e. the full hierarchy starting from the root folder of the system is returned.

If the folder specified does not exist in the system, a corresponding Error XML is returned.

Functions

Ping

Returns a standard message for test purposes.

Syntax

```
Ping()
```

Parameter

-

Return value

Standard message: **MWS Repository Service**

Rep_GetBinFile_Mime

Returns the BLOB data of an object.

Syntax

```
Rep_GetBinFile_Mime(string systemName, string options,
                    string objectId, string objectName,
                    string objectType, out string doc,
                    out string xmlResult)
```

Parameter

systemName	SystemOId of the system
options	Parameter is currently ignored and should be passed as an empty string.

objectId	Object ID of the object whose BLOB data is to be retrieved. If not set, the parameter ObjectName and ObjectType must be set.
objectName	Name of the object. If specified, then the type must also be specified.
objectType	The type of the object data to be retrieved (e.g., D, B)
doc	The content of the BLOB field as Base64-coded string.
xmlResult	Empty when OK, otherwise Mws Error XML

Return value

0	OK
<>0	Error

Rep_GetDataProviderDefinition

Reads a DataProviderDefinition from the repository.

Syntax

```
Rep_GetDataProviderDefinition(
    string systemName,string dataProviderDefinitionId,
    string dataProviderDefinitionName,
    out string xmlResult)
```

Parameter

systemName	SystemOId
dataProviderDefinitionId	Object ID of the DataproviderDefinition. If not defined, the name of the DataProviderDefinition must be defined instead
dataProviderDefinitionName	Name of the DataProviderDefinition
xmlResult	XML serialized MwsDataProviderDefinition object

Return value

0	OK
<>0	Error

Rep_GetFolderContent

Returns the content of a folder.

Syntax

```
Rep_GetFolderContent(string systemName,string options,
                     string filter,string folderName,
                     out string xmlResult)
```

Parameter

systemName	SystemOId of the system
options	Not currently used
filter	Filter settings for the query (D,P,B,A)
folderName	Folder DBID or folder name
xmlResult	MwsFolderContent XML serialized, or MwsError in the event of an error

Return value

0	OK
<>0	Error

Rep_GetForms

Returns a list of forms from the repository.

If the name or object ID of a printer is specified, only the forms used by the printer are listed.

If the object ID of the printer is specified, the parameter **PrinterName** is ignored.

Syntax

```
Rep_GetForms(string systemName,string usedByPrinterId,
             string usedByPrinterName,string printerType,
             out string xmlResult)
```

Parameter

systemName	SystemOId of the system

usedByPrinterID	Optional: PrinterOID
usedByPrinterName	Optional: PrinterName is only used, when PrinterOID is not set
printerType	Optional: required if usedByPrinterID is set. Possible values are "OEP" for email printer, "OP" for online printer Possible values are "OEP" for email printer, "OP" for online printer
xmlResult	Forms list serialized as MwsFolderContent XML, or MwsError in the event of an error

Return value

0	OK
<>0	Error

Rep_GetItemInfo

Reads an ItemInfo from the repository.

Syntax

```
Rep_GetItemInfo(string systemName, string objectId,
                string objectName, string objectType,
                out string xmlResult)
```

Parameter

systemName	SystemOId of the system
objectId	Object ID of the object, for which the ItemInfo is to be read. If this parameter is not set, the parameter ObjectName and ObjectType must be set.
objectName	Name of the object, for which the ItemInfo is to be read. If the parameter ObjectName is set, then the ObjectType must also be set.
objectType	Type of object, for which the ItemInfo is to be read.
xmlResult	XML serialized MwsItemInfo object, or MwsError in the event of an error. Empty if the specified object is not found.

Return value

0	OK
<>0	Error

Rep_GetLookupObjects

Looks up the values to an alias in an alias table and interprets the results as object references (RepositoryReference(s)).

Syntax

```
Rep_GetLookupObjects(string systemName,string options,
                     string aliasTableId,string aliasTableName,
                     string alias,out string xmlResult)
```

Parameter

systemName	SystemOId
options	Not currently evaluated
aliasTableId	Object ID of the alias table. If the object ID is not defined, the object name and type must be defined instead
aliasTableName	Name of alias table in which the lookup is to be executed.
alias	Alias whose values are to be read
xmlResult	Returns the object references in the form of a MwsLookupResult Xml (contains MwsFolderContent Xml)

Return value

0	OK
<>0	Error

Rep_GetLookupValue

Returns the value of a specified key from the lookup table defined.

Syntax

```
Rep_GetLookupValue(string systemName,string options,
                   string lookupTableId,string lookupTableName,
```

```
                    string key,out string xmlResult)
```

Parameter

systemName	SystemOId
options	Not currently evaluated
lookupTableId	Object ID of the lookup table. If not defined, the object name and type must be defined instead
lookupTableName	Name of the lookup table
key	Key
xmlResult	Returns the object references in the form of a MwsLookupResult Xml (contains MwsLookupValue Xml)

Return value

0	OK
<>0	Error

Rep_GetNavigList

Returns a list of objects in a folder in a system.

This function requires a path definition and takes into account a defined start folder for a user.

Internally, this is automatically appended to the beginning of the path definition.

Syntax

```
Rep_GetNavigList(string systemName,string options,
                 string filter,string path,
                 out string xmlResult)
```

Parameter

systemName	SystemOId
options	Parameter is currently ignored and should be passed as an empty string.
filter	Comma-separated list with type identifiers of the objects contained in the result list. Example: "A,P" (folder and bundle)

path	DB ID of the folder, or a fully qualified path definition (/Name/Name1/Name2)
xmlResult	An MwsNavigList Xml when execution was successful, otherwise Mws Error XML

Return value

0	OK
<>0	Error

Rep_GetNavigTree

Returns the navigator tree of a system.

Syntax

```
Rep_GetNavigTree(string systemName,string options,
                 out string xmlResult)
```

Parameter

systemName	SystemOId of the system
options	Options Xml, see: MwsNavigTree Options XML
xmlResult	MwsNavigTree-XML when OK, otherwise Mws Error XML

Return value

0	OK
<>0	Error

Rep_GetPrinters

Returns a list of online printers and email printers from the repository.

If the name or object ID of a form is specified, only the printers using this form are listed.

If object ID of the form is specified, the parameter **FormName** is ignored.

Syntax

```
Rep_GetPrinters(string systemName,string containingFormId,
                string containingFormName,
```

```
                out string xmlResult)
```

Parameter

systemName	SystemOId of the system
containingFormId	Optional: Object ID of the form.
containingFormName	Optional: Optional: Name of the form (only used when form ID is not specified)
xmlResult	Printer list serialized as MwsFolderContent XML, or MwsError in the event of an error

Return value

0	OK
<>0	Error

Rep_GetSystem

Returns system information from the repository.

Syntax

```
Rep_GetSystem(string systemName,out string xmlResult)
```

Parameter

systemName	SystemOId of the system for which information is to be returned
xmlResult	SystemDef XML when OK, otherwise Mws Error XML

Return value

0	OK
<>0	Error

Rep_GetSystemList

Returns the system list.

Syntax

```
Rep_GetSystemList(string options,out string xmlResult)
```

Parameter

options	Not currently used
xmlResult	MwsSystemList xml

Return value

0	OK
<>0	Error

Rep_GetValueHelpDefinition

Reads a value help from the repository.

Syntax

```
Rep_GetValueHelpDefinition(string systemName,
                           string valueHelpId,
                           string valueHelpName,
                           out string xmlResult)
```

Parameter

systemName	SystemOId of the system
valueHelpId	Object ID of the value help. If not defined, the name of the value help must be defined instead.
valueHelpName	Name of the value help
xmlResult	XML serialized MwsValueHelpDefinition object

Return value

0	OK
<>0	Error

WS UserRepository

XML Examples

RoleList XML

```
 <GetRolesResult xmlns:a="http://schemas.microsoft.com/2003/10/Serialization/
Arrays" xmlns:i="http://www.w3.org/2001/XMLSchema-instance">
   <a:string>Admin_Role</a:string>
   <a:string>Administrator_Role</a:string>
   <a:string>User_Role</a:string>
   <a:string>MainUser_Role</a:string>
   <a:string>ProdDev_Role</a:string>
   <a:string>BusinessUser_Role</a:string>
   <a:string>TextAdmin_Role</a:string>
   <a:string>OtherUser_Role</a:string>
</GetRolesResult>
```

UserList XML for GetUsers

```
<GetUsersResult xmlns:a="http://schemas.microsoft.com/2003/10/Serialization/Arrays" xmlns:i="http://
www.w3.org/2001/XMLSchema-instance">
   <a:string>DepartmentHead</a:string>
   <a:string>ACTUser</a:string>
   <a:string>Admin</a:string>
   <a:string>Administrator</a:string>
   <a:string>ASPNET</a:string>
   <a:string>Mike</a:string>
   <a:string>MMueller</a:string>
   <a:string>BusinessUser</a:string>
   <a:string>SQLDebugger</a:string>
</GetUsersResult>
```

UserList XML for GetUsersOfRole

```
 <GetUsersOfRoleResult xmlns:a="http://schemas.microsoft.com/2003/10/Serializ
ation/Arrays" xmlns:i="http://www.w3.org/2001/XMLSchema-instance">
   <a:string>Admin</a:string>
   <a:string>Administrator</a:string>
   <a:string>ASPNET</a:string>
   <a:string>Mike</a:string>
   <a:string>MMueller</a:string>
</GetUsersOfRoleResult>
```

Functions

GetRoles

Returns all roles from the configured RoleMapper.

Syntax

```
GetRoles()
```

Parameter

-

Return value

String array of all existing roles (RoleList XML)

GetUsersOfRole

Returns all users that belong to the specified role.

Syntax

```
GetUsersOfRole(string roleName)
```

Parameter

roleName	Name of the role, for which the users are to be returned.

Return value

String array of all user names belonging to the specified role (UserList XML).

GetUsers

Returns all users that belong to any role in the RoleMapper.

Syntax

```
GetUsers()
```

Parameter

-

Return value

String array of all user names that belong to any role in the RoleMapper (UserList XML)

Ping

Returns a standard message for test purposes.

Syntax

```
Ping()
```

Parameter

-

Return value

Standard message: **UserRepository Service**

Return Values

Return Values of MWS Functions

As a rule, all MWS server functions return a number.

This number specifies whether the function was executed successfully.

Basically, for a code <> 0 in the XML, an error XML is returned. This XML describes the error in more detail.

Result codes can be grouped into three categories:

- Result of Successful Execution
- Error Codes
- Warnings

Result of Successful Execution

0	Call was successfully processed.

Error Codes

-1	Error cannot be identified exactly, details in the XML error message
1	Error during function call, details in the XML error message
302	Error while setting options passed
303	Error while setting bundle structure
304	Error while closing bundle
305	Error while initializing data retrieval
309	Maximum number of client licenses exceeded. **This return code is generated when:** • The maximum number of clients defined in the server license exceeds the number of

	sessions in the sessions table. • No license found for the server. In this case only one session is allowed.

Warnings

Warnings are assigned to the range 6000 to 6999.

6000	Action cannot be executed, further configuration required.

Additional Information

Integration

The Web services approach makes implementation in a virtually unlimited number of integration scenarios possible.

The Web service architecture provides highly flexible options for technical and business-side integration of correspondence processes in an enterprise.

Technical Integration

The standard interface of the Web services is the SOAP interface layer.

This provides the deepest possible integration option for Web services. Any programming language that can process SOAP can be used to develop native Clients.

Both server-side and client-side integration can be implemented on the basis of this interface.

Integration in a Web application is also possible, for example, by direct embedding in Active Server Pages or by integrating calls to Web services as Internet links.

Business Integration

The greatest advantage for business-side integration of Web service correspondence processes is the enormous flexibility it provides.

Correspondence generation with Document Composition consists of a number of complementary and collaboratively interlinked activities. These activities can either be set to run automatically, or can be manually started and handled by a user.

The sequence in which these activities are preformed is defined by their dependency on results from preceding activities.

The Web services enable granular access to these individual activities.

This makes it easy to eliminate any unnecessary steps or to repeat one or more activities as individually required. The whole concept is geared toward flexible adaptation of technical procedures to specific business requirements.

Consider the following scenario as an example. The parameterization of data retrieval is very complex and implemented over a number of interdependent Web pages. Document selection, manual variable

input and editing of documents is not desired. Only control documents should be displayed and, if correct, then archived directly.

With Web services, implementing a scenario like this turns out to be quite simple.

Repository

The repository serves as the basic foundation of the Web services.

This is where all definitions are stored and retrieved from. These definitions can be split into different categories:

- Data relevant to correspondence processes, such as bundles, text blocks and document templates.
- Data from the area of output management, such as printers and forms.
- System-relevant data

Information can be obtained from these areas by using a special interface. This makes it possible for a third-party application to execute specific functions, for example, system selection or bundle selection including folder hierarchy.

Correspondence Processes

A correspondence process consists of a number of complementary and collaboratively interlinked activities. These activities can either be set to run automatically, or can be started and handled by a user or third-party application.

Activities can also be set to run in parallel.

Managing a Correspondence Process

A correspondence process is managed in a database table and file system and identified by a unique key. The key itself can be defined by a third-party application.

Normally, the life cycle of a correspondence process is limited to creating and printing documents. The term printing is understood here to mean passing on documents to an output management system such as Odin.

It is also possible to save a correspondence process to resume handling at a later point in time. This is done by assigning an internal status level to a process.

The Status of a Correspondence Process

A new process is always created with the status 1 (in progress).

When a process is closed, the status can either be set to 0 (saved but open for further processing) or 2 (processing is complete and, in principle, can be deleted).

Actions of a Correspondence Process

A correspondence process is comprised of the following actions:

- Create a Correspondence Process
- Specify Parameters for Data Retrieval (optional)
- Data Retrieval (optional)
- Enhance Bundle Structure (dynamic bundle, optional)

- Configure Documents (optional)
- Set Manual Variables (optional)
- Generate Documents (optional)
- Edit Documents (optional)
- Print and Archive (optional)
- Close a Correspondence Process

Index