Repository Services Component
(BC-FES-AIT)

Release 4.6C
Copyright

© Copyright 2001 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®, NT®, EXCEL®, Word®, PowerPoint® and SQL Server® are registered trademarks of Microsoft Corporation.

IBM®, DB2®, OS/2®, DB2/6000®, Parallel Sysplex®, MVS/ESA®, RS/6000®, AIX®, S/390®, AS/400®, OS/390®, and OS/400® are registered trademarks of IBM Corporation.

ORACLE® is a registered trademark of ORACLE Corporation.

INFORMIX®-OnLine for SAP and Informix® Dynamic Server™ are registered trademarks of Informix Software Incorporated.

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

HTML, DHTML, XML, XHTML 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.

SAP, SAP Logo, R/2, R/3, R/2, ABAP, SAP ArchiveLink, SAP Business Workflow, WebFlow, SAP EarlyWatch, BAPI, SAPPHIRE, Management Cockpit, mySAP.com Logo and mySAP.com are trademarks or registered trademarks of SAP AG in Germany and in several other countries all over the world. All other products mentioned are trademarks or registered trademarks of their respective companies.
## Icons

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

Repository Services Component (BC-FES-AIT)................................. 8
What’s New in Release 4.6A?........................................................................ 9
What’s New in Release 4.6B?...................................................................... 10
System Requirements.................................................................................. 11
Repository Services Object Hierarchy....................................................... 12
Working Online ....................................................................................... 14
The Local Repository: Working Offline.................................................... 18
Reading from the Local Repository ......................................................... 20
Copying Data from R/3 to the Local Repository ...................................... 22
Deleting Objects from the Local Repository .......................................... 26
Refreshing Local Repository Data ............................................................ 27
Component Interface Reference .............................................................. 28
How to Use this Documentation .............................................................. 29
IRepositoryServices .................................................................................. 30
  Unique Properties .................................................................................. 32
  Connection (Get) ................................................................................ 33
  Connection (Set) ................................................................................ 34
  IsOffline .............................................................................................. 35
  IsOnline ............................................................................................... 36
  IsR3Connected .................................................................................. 37
  Unique Methods .................................................................................. 38
  ApplicationHierarchies ....................................................................... 39
  BusinessObjects .................................................................................. 40
  CompareSAPSystems ......................................................................... 41
  CreateSAPSystem ............................................................................. 42
  DeleteSAPSystem ............................................................................. 43
  FunctionGroups .................................................................................. 44
  Functions ............................................................................................ 45
  Offline ................................................................................................. 46
  Online ................................................................................................. 47
  SAPSystems ....................................................................................... 48
  SetSAPSystem .................................................................................... 49
  WriteAppHierarchies ........................................................................ 50
  WriteAppHierarchy ........................................................................... 51
  WriteBO............................................................................................. 53
  WriteBOTree ....................................................................................... 54
  WriteFunctionGroup ........................................................................ 56
  WriteFunctionGroups ....................................................................... 58
  WriteRFC ............................................................................................. 59
ISAPSystems ............................................................................................. 61
ISAPSystem ............................................................................................ 62
  Unique Properties .............................................................................. 63
<table>
<thead>
<tr>
<th>Class</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>IFIELD</td>
<td>114</td>
</tr>
<tr>
<td>IFIELDS</td>
<td>113</td>
</tr>
<tr>
<td>IFIELD</td>
<td>112</td>
</tr>
<tr>
<td>ITable</td>
<td>111</td>
</tr>
<tr>
<td>IStructure</td>
<td>110</td>
</tr>
<tr>
<td>IStructure</td>
<td>109</td>
</tr>
<tr>
<td>ITable</td>
<td>108</td>
</tr>
<tr>
<td>IFields</td>
<td>107</td>
</tr>
<tr>
<td>IField</td>
<td>106</td>
</tr>
</tbody>
</table>

**Unique Properties**

- CheckTable: 127
- DataElement: 128
- Decimals: 129
- Domain: 130
- HasFixedValues: 131
- HelpValues: 132
- HelpValuesCount: 133
- InternalLength: 134
- IsCaseSensitive: 135
- IsConversionExit: 136
- Length: 137
- LongScreenText: 138
- Offset: 139
- Position: 140
- ReferenceField: 141
- ReferenceTable: 142
- ReportHeader: 143
- ShortScreenText: 144
- ValuesForField: 145
- ValuesForFieldCount: 146

**Common Properties**

- ABAPType: 147
- ApplicationArea: 148
- Count: 149
- Description: 150
- DictionaryType: 151
- Documentation: 152
- InternalName: 153

---

April 2001
Repository Services Component (BC-FES-AIT)

Purpose
The Repository Services Component provides read access to SAP R/3 business object and RFC function module metadata to external programs and applications.

The Repository Services Component provides a standard access interface to R/3 objects and encapsulates the mechanism required to physically access the underlying R/3 system and database.

Implementation Considerations
You can use the Repository Services component in programs that follow the Microsoft Component Object Model (COM).

Integration
The Repository Services component allows you to create a Repository Services object on a client computer, which holds the metadata of an R/3 business object or a function module.

The following diagram illustrates, as an example, a Repository Services object representing a SAP business object.

![Diagram of Repository Services Component](image)

The details of the metadata of the SAP business object or of the RFC function module are contained in a hierarchy of objects under the Repository Services object [Page 12].

Features
The Repository Services component allows you read SAP business object and SAP function module metadata.

The Repository Services component allows you to access this metadata using a live connection to an R/3 system. We refer to this as working online [Page 14].

The Repository Services component also allows you to save metadata in a local repository database [Page 18]. This enables access to the metadata offline, that is, without a connection to the R/3 system.
What's New in Release 4.6A?

Error handling has changed in Release 4.6A to use the COM error handling. As a result, the various methods of the Repository Services component now return either True or False, instead of Success or Failure. In addition, the LastError property of IRepositoryServices is no longer in use.
What's New in Release 4.6B?

The Repository Services Component now works with R/3 Releases 3.0 A and greater.
System Requirements

Development Requirements
To create applications using the Repository Services component you need the following:

- Windows NT 4.0, Windows 95, or the Windows 98 operating system
- The SAP DCOM Connector product installed, which in turn requires:
  - Visual C++ version 5.0 or higher
  - Windows NT 4.0 Option Pack including the MTS development environment
- SAP R/3 System Release 3.0 A or higher.

Run-time Requirements
The end user of an applications using the Repository Services component needs the same components as listed above.
Repository Services Object Hierarchy

Use

The Repository Services component allows you to create a Repository Services object, representing the metadata of an SAP business object or of an RFC function module.

The Repository Services component also allows you to create other objects, all related to the Repository Services object, to hold the details of the metadata of the SAP business object or function module.

Using the various methods and properties of these objects you can find out the exact details of the parameters of a function module, for example, including their data type and documentation. For a business object, as another example, you can find out what are its methods (BAPIs), and for each method, what are its parameters.

Structure

The following diagram shows the hierarchy relationship among the Repository Services objects.
Working Online

Use
Using the Repository Services component you can work in online mode, that is, you can read metadata information using a live connection to an R/3 system.

Prerequisites
Before you can access any metadata in online mode through the Repository Services, you must set up the necessary connection information in the Repository Services object.

Why You Need to Set up The Connection Property?
The Repository Services object represents an SAP business object or an RFC function module. When working online, the Repository Services component needs to be connected the R/3 system containing the SAP business object in order to get its metadata.

The Repository Services component uses services from the SAP DCOM Connector product to handle connections to an R/3 system when a connection is necessary. It does not actively establish a connection.

To allow for a connection to be established when needed, each object requiring a connection must contain connection information in their properties.

The Repository Services object therefore contains a Connection property to hold this required information.

The Connection property points to a Connection object, whose properties may hold the various parameters of logon information.

Programs using the Repository Services component can use the SAP Automation DCOM Connector Logon Component [Ext.] to create a Connection object.

With the SAP Automation DCOM Connector Logon Component you can present a Logon dialog to your end user, which then fills out the various properties of the Connection object.

You then assign this connection object to the Connection property of the Repository Services object.
Procedure

1. Set up at least one destination system with the SAP DCOM Connector Destination editor. This creates a destination entry in the Windows Registry. This destination entry may include client, user ID, and even password.

   See the SAP DCOM Connector notes, or the online help for the SAP Automation DCOM Connector Logon Component [Ext.].

2. Create a Connection object with the SAP Automation DCOM Connector Logon Component [Ext.].

3. Set up the properties of the Connection object to the values of the system you wish to use.

   You can set up the values programatically, or you can use the Logon method of the SAP Automation DCOM Connector Logon Component to display a Logon dialog to your end user. When the user chooses OK at this dialog, the various properties of the Connection object are filled with the values entered at the dialog.

   If you set up the properties of the Connection object, then their values override any logon values that exist in the Windows Registry when the connection is established.

4. Assign the Connection object to the Connection property of the Repository Services object by either setting the Connection property [Page 34] or by using the Online method [Page 47].

The following illustration summarizes the process of setting up an online mode.
After setting the Connection property of the Repository Services object you are ready to use the various Repository Services objects, their methods and properties. Whenever an object or method you are using requires a connection to an R/3 system, the connection is established automatically for you.

**Example**

The following example is a VB function for setting up the connection parameters. It uses the SAP Automation DCOM Connector Logon Component to create a Connection object, and to display a Logon dialog to the end user.

```vbnet
Private Function R3Logon() As Object
    Dim oConn As Connection
    ' Create the Connection object
    Set oConn = New SAPLogonLib.Connection
    ' use the Logon method of the
    ' SAP Automation DCOM Connector Logon Component
    ' to display a Logon dialog to the user
    oConn.Logon
    Set R3Logon = oConn
    Set oConn = Nothing
End Function
```

Dim oRep As New RepositoryServices
Dim oConn As Object
'Log onto R/3 using the R3Logon function shown above
Set oConn = R3Logon()
oRep.online (oConn)

Note that although the example above does not actively establish a connection to R/3 (a call to a BAPI or an RFC that requires a connection automatically establishes such a connection), it is called R3Logon for compatibility with older sample code, which was using the Logon Control for establishing a connection.

Previous versions of the Repository Services component (for R/3 release 4.5A and earlier) used the SAP Automation Logon Control. The following code for the R3Logon function uses the Logon Control to create the Connection object. It also actively establishes a connection to R/3. Use this code if you are using earlier versions of the Repository Services component.

Private Function R3Logon() As Object
    Dim oLogonControl As Object
    Dim oLogon As Object
    Set oLogonControl = CreateObject("SAP.LogonControl.1")
    If Not oLogonControl Is Nothing Then
        Set oLogon = oLogonControl.NewConnection
        If oLogon.logon() Then
            Set R3Logon = oLogon
        End If
    Set oLogonControl = Nothing
    Set oLogon = Nothing
End Function

You can use the sample code in other sections of this Help document without any changes regardless of whether you use the code using the SAP Automation Logon Control or the code using the SAP Automation DCOM Connector Logon Component.

See Also
What's New? [Page 10], SAP Automation DCOM Connector Logon Component [Ext.], Working offline with a local repository [Page 18]
The Local Repository: Working Offline

Use
The Repository Services product allows you to copy metadata you obtain from an R/3 system into a local repository. You can then work offline, that is, you can access the desired metadata from the local repository, instead of from the original R/3 system.

Working offline allows you to work with the metadata when the R/3 system is not available, or when connecting to the R/3 system would be too slow.

Since metadata usually does not change very often, working offline is a good alternative to connecting to a live R/3 system for retrieving metadata.

Integration
The local repository is a Microsoft Access database. Its file type is MDB.

Features
You can store metadata from one or more R/3 systems in a single local repository.

You can create one or more local repositories, but you always work with one of them at a time.

Repository Services provides special methods for working with local repositories:

- Connecting to a local repository
- Writing data into the local repository
- Deleting all the data that belongs to a specific R/3 system from the local repository

Reading data from the local repository is done with the same methods used for reading data from an R/3 database. The Repository Services automatically uses the local repository if you are working in an offline mode.

Activities
You start using the local repository by using the Offline method, in which you specify the name of the local repository.

The Repository Services creates the local repository database automatically the first time you use the Offline method with a database, if that database does not exist yet.

The following table lists the tasks you can perform when working with the local repository. It lists the method or methods of the \texttt{IRepositoryServices[Page 30]} you can use to perform each of these tasks.

<table>
<thead>
<tr>
<th>Task</th>
<th>\texttt{IRepositoryServices Method(s)}</th>
</tr>
</thead>
</table>

April 2001
### The Local Repository: Working Offline

<table>
<thead>
<tr>
<th>Action</th>
<th>Reference</th>
</tr>
</thead>
<tbody>
<tr>
<td>Connect to the local repository and work in offline mode.</td>
<td>Offline [Page 46]</td>
</tr>
<tr>
<td>Note that if you are also connected to a live R/3 database, that connection takes precedence, meaning that if you use a method for reading data, the Repository Services reads that data from the R/3 database, not from the local repository.</td>
<td></td>
</tr>
<tr>
<td>Store metadata, which you have read from an R/3 system, into the local repository</td>
<td>The various Write methods: WriteAppHierarchies [Page 50], WriteAppHierarchy [Page 51], WriteBO [Page 53], WriteBOTree [Page 54], WriteFunctionGroup [Page 56], WriteFunctionGroups [Page 58], WriteRFC [Page 59]</td>
</tr>
<tr>
<td>Get the list of SAP R/3 systems whose objects exist in the local repository</td>
<td>SAPSystems [Page 48]</td>
</tr>
<tr>
<td>Specify which R/3 system objects to work with, when reading from the local repository</td>
<td>SetSAPSystem [Page 49]</td>
</tr>
<tr>
<td>Read metadata from the local repository</td>
<td>ApplicationHierarchies [Page 39], BusinessObjects [Page 40], FunctionGroups [Page 44], Functions [Page 45] (The same functions that are used for reading from an R/3 system)</td>
</tr>
<tr>
<td>Delete all the data belonging to the R/3 system you specify from the repository.</td>
<td>DeleteSAPSystem [Page 43]</td>
</tr>
</tbody>
</table>
Reading from the Local Repository

Use
Reading data from the local repository allows you to retrieve metadata offline, that is, retrieve metadata without being connected to the R/3 system containing that metadata.

Procedure
When reading metadata with the Repository Services you access one R/3 system at a time. When reading data in offline mode you also need to work with objects that belong to one R/3 system at a time.

Since metadata originating from different systems can be stored in the same local repository, you need to specify which R/3 system’s objects you are going to work with.

The following table summarizes the steps for reading data from the local repository, and the methods to use for performing these steps.

<table>
<thead>
<tr>
<th>Step</th>
<th>Method(s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.</td>
<td>Connect to the local repository.</td>
</tr>
<tr>
<td>2.</td>
<td>(Optional) Get the list of SAP R/3 systems contained in the local repository. This is the list of all the systems to which the objects in the local repository belong. You do not need to perform this step if you know the R/3 system to which the objects you wish to work with belong.</td>
</tr>
<tr>
<td>3.</td>
<td>Specify which of these systems to work with.</td>
</tr>
<tr>
<td>4.</td>
<td>Read metadata from the local repository using the same functions that are used for reading from an R/3 system. Note that if you are also connected to a live R/3 database, the live R/3 connection takes precedence, meaning that the Repository Services reads the data from the R/3 database, not from the local repository.</td>
</tr>
</tbody>
</table>

Example
The following VB code example looks for the objects belonging to a specific SAP system in the local repository, and then gets the list of business objects belonging to that system:

```vbnet
Dim oRep As New RepositoryServices
Dim oAppHiers As IApplicationHierarchies
Dim oAppHier As IApplicationHierarchy
Dim oBusObjs As IBusinessObjects
Dim oBusObj As IBusinessObject
Dim oSyss As ISapSystems
Dim Count As Integer

If oRep.offline("c:\sapreps.mdb") Then
```
Set oSyss = oRep.sapsystems()
For Count = 1 To oSyss.Count
    ' (Your code for determining which system to look at)
    If StrComp(oSyss(Count).Name, "ESS", vbtextcompreal) = 0 Then _
        oRep.SetSAPSystem (oSyss(Count))
Next
End If
'Getting list of ApplicationHierarchy objects
Set oAppHiers = oRep.ApplicationHierarchies()
For Count = 1 To oAppHiers.Count
    Set oAppHier = oAppHiers.Item(Count)
    ' Your code for using the application hierarchy object,
    ' for example:
    'Msgbox oappHier.description
'Get the list of Business Objects
Set oBusObjs = oAppHier.BusinessObjects()
For Count = 1 To oBusObjs.Count
    Set oBusObj = oBusObjs.Item(Count)
    ' Your code for using the business object
    ' for example:
    'Msgbox oBusObj.name
Next
Next
Copying Data from R/3 to the Local Repository

Use
To work with a local repository you must first populate the local repository database with metadata from the R/3 database.

You can copy all of the metadata of all business objects and RFC functions from an R/3 system, or you can copy a subset of this metadata.

Prerequisites
To copy data from an R/3 system to the local repository you must be connected to both the R/3 system and the local repository.

To connect to the R/3 system from which you wish to copy data you use the SAP Logon Control outside of the Repository Services product.

Procedure
1. Use the `Online` method to point the `IRepositoryServices` object to the R/3 connection you have established through the Logon Control. The Online method assigns the logon object to the `IRepositoryServices` you are working with.

   You can also use the `Connection property` of the `IRepositoryServices` object to perform this task.

2. Use the `Offline` method to connect to the local repository, specifying the full path of the local database location and name.

   If the local repository exists, the Repository Services opens it. If the local repository database does not exist, the Repository Services creates it first.

   The Repository Services also verifies the integrity of the local repository database, to check that it was not manipulated manually.

3. Get the desired metadata from the R/3 system. Use the read methods, such as `ApplicationHierarchies`, `BusinessObjects`, `FunctionGroups`, and `Functions`.

   Since the connection to the R/3 takes precedence, the Repository Services reads the data from R/3 even though you are connected to both the local repository and the R/3 database at the same time.

4. Use the `CreateSAPSystem` method to create the `ISAPSystem` object (representing the R/3 system you are copying from) in the local repository.
Use the CreateSAPSystem even if the ISAPSystem object for that R/3 system exists. If that ISAPSystem already exists, the method returns a pointer to it, allowing you to use it in the methods for writing the data into the local repository, as in the next step.

5. Write the object or collection of objects to the local repository using the appropriate Write method, such as WriteAppHierarchies [Page 50], WriteAppHierarchy [Page 51], WriteBO [Page 53], WriteBOTree [Page 54], WriteFunctionGroup [Page 56], WriteFunctionGroups [Page 58], WriteRFC [Page 59].

For example, if you retrieved a business object from the R/3 system with the BusinessObjects method, then you can add that business object to the local repository with the WriteBO method. To add all the children objects of that business object to the local repository, as well as the business object itself, use the WriteBOTree method.

Examples

The following example copies business objects from an R/3 system into the local repository. It shows two alternative codes for selecting business objects: one section of the code shows how to save all of the application hierarchies and their child objects, and one section of the code shows how to save only a specific application hierarchy, and its child objects:

Dim oRep As New RepositoryServices
Dim oAppHiers As IApplicationHierarchies
Dim oAppHier As IApplicationHierarchy
Dim oSys As ISAPSystem
Dim oConn As Object
Dim Count As Integer
Dim Count1 As Integer

' Log onto R/3 using the function shown in the example for the Working Online topic
Set oConn = R3Logon()
If Not oConn Is Nothing Then
    oRep.online (oConn)
' Connect offline
    ' This creates the local database file if it does not exist
    oRep.offline ("c:\sapreps.mdb")
    Set oSys = oRep.CreateSAPSystem(oConn.destination, oConn.HostName, oConn.System, oConn.systemnumber, oConn.applicationserver, oConn.saprelease)

    'Get a list of ApplicationHierarchy objects
    Set oAppHiers = oRep.ApplicationHierarchies()
    'Save all the application hierarchy objects and their child objects, such as BusinessObjects, their methods, parameters etc
    oRep.writeapphierarchies oSys, oAppHiers
    For Count = 1 To oAppHiers.Count
        Set oAppHier = oAppHier.Item(Count)
        ' Your code for using the application hierarchy object for example: MsgBox oapphier.description
        ' As an alternative to saving all of the application hierarchies, as in the code above
Copying Data from R/3 to the Local Repository

' you can save only a single application hierarchy object
' (the current application hierarchy object),
' and all of its child objects, such as
' BusinessObjects, their methods, parameters etc.
' Use the following code:
' oRep.writeappHierarchies oSys, oAppHier
Set oBusObjs = oAppHier.BusinessObjects
For Count1 = 1 To oBusObjs.Count
    Set oBusObj = oBusObjs.Item(Count1)
    ' Your code for using the current business object
    ' MsgBox oBusObj.name
Next
Next
End If

The following example copies all of the business objects of a particular R/3 system into the local repository:

Dim oRep As New RepositoryServices
Dim oBusObjs As IBusinessObjects
Dim oBusObj As IBusinessObject
Dim oSys As ISAPSystem
Dim oConn As Object
Dim Count As Integer

' Log onto R/3 using the function shown in the example
'   for the Working Online topic
Set oConn = R3Logon()
If Not oConn Is Nothing Then
    oRep.online (oConn)
    ' Connect offline
    ' This creates the local database file
    ' if it does not exist
    oRep.offline ("c:\sapreps.mdb")
Set oSys = oRep.CreateSAPSystem(oConn.destination, _
    oConn.HostName, oConn.System, oConn.systemnumber, _
    oConn.applicationserver, oConn.saprelease)
' Get the list of all the Business objects
Set oBusObjs = oRep.BusinessObjects
For Count = 1 To oBusObjs.Count
    Set oBusObj = oBusObjs.Item(Count)
    ' Your code for using the business object, for example:
    ' MsgBox oBusObj.name
    ' To save the current Business Object object,
    ' and all of its child objects
    ' such as methods, parameters etc:
    oRep.writeBOTree oSys, oBusObj
    ' To save only the current business object
    ' without its child objects:
    ' oRep.writeBO oSys, oBusObj
Next
End If

**See Also**

*Working Online [Page 14].*

You may need to refresh the metadata in the local repository [Page 27] occasionally, if the metadata the R/3 database changes after you have copied it.
Deleting Objects from the Local Repository

Use
You can delete all the objects originating from a single R/3 system from the local repository.
You cannot delete individual objects from the local repository.

Procedure
1. Connect to the local repository with the Offline method [Page 46], specifying the local repository database from which you wish to delete objects.
2. (Optional) Get a list of the SAP Systems in the local repository, by calling SAPSystems. Identify the SAP system to be deleted.
3. Use the DeleteSAPSystem method [Page 43] to delete all of the objects belonging to the ISAPSystem object you specify.

Result
The objects are deleted from the Microsoft Access database file (MDB).

The size of the MDB file does not change after deleting objects. If you delete objects often, you may want to release the space on your hard disk by using the Microsoft Acces command: Tools ➔ Compact Database.
Refreshing Local Repository Data

Use

If your program or users read data from the local repository, and if the business objects and RFC functions metadata in your R/3 system changes, you should update the local repository with the new version of the metadata from R/3.

The frequency of updating the local repository depends on how often the metadata in your R/3 system changes.

Procedure

To refresh the local copy of the metadata of a particular R/3 system:

1. Delete the data belonging to that R/3 system from the local repository database [Page 26].
2. Copy the desired metadata from the R/3 system to the new local repository [Page 22].

Since you can create and work with more than one local repository database, as an alternative, you can first copy the desired metadata into a new local repository database, and only then delete the old copy of the data from the existing local repository.

1. Create a new local repository database by using the Offline method [Page 46], and specifying a new database name.
2. Copy the desired data from R/3 to the new local repository [Page 22].
3. Delete the data from the existing local repository database [Page 26].

Do not simply re-write the same objects into the local repository without deleting the old data first.

If an object you are writing into the local repository already exists in the local repository, the Repository Services does not write it again. If you are trying to write a newer version of the objects, which you have just copied from a live R/3 system, then the object is not updated in the local repository.
Component Interface Reference
How to Use this Documentation

This reference documentation describes the various interfaces to the objects in the Repository Services hierarchy.

The properties and methods of each of the interfaces is listed in the interface description. A separate topic describes each property or methods in more details.

The properties and methods that are unique to each interface appears under that interface. Properties and methods that are common to multiple interfaces are grouped together under the Common Properties and Common Methods titles, respectively. To see the full description of a property or method, follow the link from the list in the interface description.

Syntax Conventions

The format used for describing the syntax of the various properties in this reference documentation is as follows:

PropertyName : PropertyType = InitialValue

For example, the following is the syntax for the Count property:

Count : Integer = 0

The format used for describing method syntax is as follows:

MethodName () : MethodType

For example, the following is the syntax for the Fields method:

Fields () : IFields

When a property or a method has one or more parameters, the parameters and their type is included in the syntax. For example, the two parameters of the WriteAppHierarchy method and their type are described in the following syntax:

WriteAppHierarchy(aSAPSystem : ISAPSystem, aApplicationHierarchy : IApplicationHierarchy) : Boolean
IRepositoryServices

Purpose
An Interface for a repository services object. It retrieves the metadata of R/3 objects and manages the connection to R/3 system and the local repository. This is the entry point of the Repository Services component.

Properties

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Connection</td>
<td>Sets [Page 34] or gets [Page 33] connection to an R/3 System</td>
</tr>
<tr>
<td>IsOffline</td>
<td>Finds out if a user connects to the local repository</td>
</tr>
<tr>
<td>IsOnline</td>
<td>Finds out if the user is connected to the R/3 system</td>
</tr>
<tr>
<td>IsR3Connected</td>
<td>Finds out if a user's attempt to connect to R/3 System has worked</td>
</tr>
</tbody>
</table>

Methods

<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ApplicationHierarchies</td>
<td>Gets an application hierarchy collection containing all application hierarchies</td>
</tr>
<tr>
<td>BusinessObjects</td>
<td>Gets a business object collection containing one or more business objects as specified by the search criteria</td>
</tr>
<tr>
<td>CompareSAPSystems</td>
<td>Compares the system information between the current R/3 System and other R/3 System</td>
</tr>
<tr>
<td>CreateSAPSystem</td>
<td>Create a new ISAPSystem object in the local repository whose properties are the parameters you specify</td>
</tr>
<tr>
<td>DeleteSAPSystem</td>
<td>Deletes all the R/3 objects associated with the R/3 system you specify from the local repository</td>
</tr>
<tr>
<td>FunctionGroups</td>
<td>Gets one or more function groups as specified by the search criteria</td>
</tr>
<tr>
<td>Functions</td>
<td>Gets a function collection containing one or more functions as specified by the search criteria</td>
</tr>
<tr>
<td>Offline</td>
<td>Sets the connection to the local repository</td>
</tr>
<tr>
<td>Online</td>
<td>Sets the connection to a R/3 System</td>
</tr>
<tr>
<td>SAPSystems</td>
<td>Gets a list of R/3 Systems, which are stored in the local repository, to which a user has the authorization to log on</td>
</tr>
<tr>
<td>SetSAPSystem</td>
<td>Connects to one R/3 System in the local repository</td>
</tr>
<tr>
<td>WriteAppHierarchies</td>
<td>Downloads an application hierarchies collection into the local repository</td>
</tr>
<tr>
<td>WriteAppHierarchy</td>
<td>Downloads an application hierarchy object into the local repository</td>
</tr>
<tr>
<td>WriteBO</td>
<td>Downloads a business object into the local repository</td>
</tr>
<tr>
<td>Method</td>
<td>Description</td>
</tr>
<tr>
<td>------------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td><strong>WriteBOTree [Page 54]</strong></td>
<td>Downloads a business object and all of its children objects into the local repository</td>
</tr>
<tr>
<td><strong>WriteFunctionGroup [Page 56]</strong></td>
<td>Downloads a function group object into the local repository</td>
</tr>
<tr>
<td><strong>WriteFunctionGroups [Page 58]</strong></td>
<td>Downloads a function group collection into the local repository</td>
</tr>
<tr>
<td><strong>WriteRFC [Page 59]</strong></td>
<td>Downloads a function object into the local repository</td>
</tr>
</tbody>
</table>
Unique Properties
Connection (Get)

Purpose
Returns the Connection object associated with Connection property of the Repository Services object.
This is for online use only.

Syntax
Connection : Object = Null

Parameters
None.

Return Value
Returns a connection object, or null if it fails.

See Also
Online [Page 47], Connection (Set) [Page 34], Working online [Page 14]
Connection (Set)

Purpose
Assigns the Connection object from the SAP Automation DCOM Connector Logon Component (containing all the necessary R/3 connection information) to the IRepositoryServices object.
This is for online use only.

Syntax
Connection(aConnection : Object) : HRESULT

Parameters

| aConnection | A Connection object from the SAP logon control |

Return Value
Returns HRESULT.

Comments
Using the connection property to set the connection object is the same as using the Online method [Page 47] of IRepositoryServices.

See Also
Online [Page 47], Connection (Get) [Page 33], Working online [Page 14]
IsOffline

Purpose
Finds out if a user connects to the local repository.

Syntax
IsOffline : Boolean = False

Parameters
None.

Return Value
Returns true, if the user connects to the local repository, and false if the user does not connect.
IsOnline

**Purpose**
Finds out if the user is connected to the R/3 system.

**Syntax**
IsOnline : Boolean = False

**Parameters**
None

**Return Value**
Returns true if the user is connected to the R/3 system, and false if not.
IsR3Connected

Purpose
Indicates whether the user is in online mode.

Syntax
IsR3Connected : Boolean = False

Parameters
None.

Comments
You can obtain the same information with the IsOnline property [Page 36]. The IsR3Connected is kept for compatibility with previous versions of the Repository Services component. In previous versions of Repository Services this property indicated if a user's attempt to connect to the R/3 system has succeeded.

Return Value
Returns true, if the user connected to the R/3 system, and false if the user does not connect.
Unique Methods
ApplicationHierarchies

Purpose

Gets all application hierarchies of a particular R/3 system.

- When working online, objects are always retrieved directly from R/3 system.
- When working offline, objects are retrieved from the local repository.

Syntax

```java
ApplicationHierarchies () : IApplicationHierarchies
```

Parameters

None.

Return Value

Returns an application hierarchy object collection containing all the application hierarchy objects, or null, if failure.
BusinessObjects

Purpose
Gets one or more business objects as specified by the search criteria.

- When working online, business objects are always retrieved directly from R/3 system.
- When working offline, business objects are retrieved from the local repository.

Syntax
BusinessObjects (aSearchCriterion : String) : IBusinessObjects

Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>aSearchCriterion</td>
<td>A search criterion to retrieve Business objects</td>
</tr>
</tbody>
</table>

If the parameter is null, then all business objects are retrieved. If it is not null, then the requested business object is retrieved, based on the search criterion.

Return Value
Returns a business object collection containing all the business objects that satisfy the selection criteria, or null, if failure.
CompareSAPSystems

Purpose
Compares the system information between the current SAP R/3 system and another SAP R/3 system.

Syntax
CompareSAPSystems (aSAPSystem : ISAPSystem) : Boolean

Parameters
| aSAPSystem | a ISAPSystem object |

Return Value
Returns true, if the two SAP R/3 systems are the same, and false if they are different.
CreateSAPSystem

Purpose
Create a new ISAPSystem object whose properties are the parameters you specify.
You can use this ISAPSystem object to create objects in the local repository that belong to the
R/3 system identified by the ISAPSystem.
If the ISAPSystem already exists, this method returns a pointer to it.

Syntax
CreateSAPSystem (aDestination : String, aHostName : String, aSystem :String, aSystemNumber : String, aApplicationServer : String, aSAPRelease : String) : ISAPSystem

Parameters
<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>aDestination</td>
<td>the SAP R/3 system name</td>
</tr>
<tr>
<td>aHostName</td>
<td>the hostname of the SAP R/3 system</td>
</tr>
<tr>
<td>aSystem</td>
<td>SAP R/3 system description</td>
</tr>
<tr>
<td>aSystemNumber</td>
<td>the system number of the SAP R/3 system</td>
</tr>
<tr>
<td>aApplicationServer</td>
<td>the SAP R/3 application server name</td>
</tr>
<tr>
<td>aSAPRelease</td>
<td>the SAP R/3 system release</td>
</tr>
</tbody>
</table>

Return Value
Returns an ISAPSystem object, or null if failure.
DeleteSAPSystem

Purpose
Deletes all the R/3 objects associated with the R/3 system you specify from the local repository.

Syntax
DeleteSAPSystem (aSAPSystem : ISAPSystem) : Boolean

Parameters

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>aSAPSystem</td>
<td>An ISAPSystem object, identifying the R/3 system to which the objects belongs</td>
</tr>
</tbody>
</table>

Return Value
Returns True if the deletion succeeds, and False if it fails.

See Also
CreateSAPSystem [Page 42]
FunctionGroups

**Purpose**

Gets one or more function groups as specified by the search criteria.

- When working online, function group objects are always retrieved directly from R/3 System.
- When working offline, the function group objects are retrieved from the local repository.

**Syntax**

```
FunctionGroups (aSearchCriterion : String) : IFunctionGroups
```

**Parameters**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>aSearchCriterion</td>
<td>Search criterion to retrieve the function groups</td>
</tr>
</tbody>
</table>

If the parameter is null, then all function groups are retrieved. If it is not null, then the requested function groups based upon the search criterion are retrieved. The wildcard expression can be used in the aSearchCriterion parameter.

**Return Value**

Returns a function group object collection containing all the function group objects that satisfy the selection criteria, or null, if failure.
Functions

Purpose
Gets one or more functions as specified by the search criteria.

- When working online, function objects are always retrieved directly from R/3 System.
- When working offline, the function objects are retrieved from the local repository.

Syntax
Functions (aSearchCriterion : String) : IFunctions

Parameters

| aSearchCriterion | A search criterion to retrieve the functions |

If the parameter is null, then all functions are retrieved. If it is not null, then the requested functions are retrieved based upon the search criterion. A wildcard expression can be used in the aSearchCriterion parameter.

Return Value

Returns a function object collection containing all the function objects that satisfy the selection criteria, or null, if failure.
Offline

Purpose
Sets a connection to the local repository [Page 18].

Note that if you are connected to both the local repository and to a live R/3 database, the live R/3 connection takes precedence. That means that if you use a method for reading data, the Repository Services reads that data from the R/3 database, not from the local repository.

Syntax
Offline (aDatabase : String) : Boolean

Parameters

| aDatabase | Full path of the local database location and name |

Result
- If the local repository database exists, this method opens it.
- If the local repository database file does not exist, the method first creates it, and then opens it.
- If the local database file is invalid, meaning that it does not follow the correct structure for the Repository Services (for example, if it was created manually), then the method fails.

Return Value
Returns true if the local database has been opened successfully, or false if failure.
**Online**

**Purpose**
Assigns the SAP Automation DCOM Connector Logon Component's Connection object to the Connection property of the IRepositoryServices you are working with.

**Syntax**
```plaintext
Online (aConnection : Object) : Boolean
```

**Parameters**
- `aConnection` A connection object of the SAP logon control

**Comments**
Using the Online method is the same as using the [Connection property][Page 34] of IRepositoryServices.

Note that if you are connected to both the [local repository][Page 18] and to a live R/3 database, the live R/3 connection takes precedence. That means that if you use a method for reading data, the Repository Services reads that data from the R/3 database, not from the local repository.

**Return Value**
Returns true if successful, or false if failure.

**See Also**
- [Working Online][Page 14]
SAPSystems

**Purpose**

Gets a list of SAP R/3 Systems:

- In online mode, this method returns a single ISAPSystem object, representing the SAP R/3 system to which the user is currently logged on.
- In offline mode, this method gets the list of ISAPSystem objects stored in the local repository.

**Syntax**

`SAPSystems () : ISAPSystems`

**Parameters**

None.

**Return Value**

Returns a collection of SAP R/3 Systems or null, if failure.
SetSAPSystem

Purpose
Specifies the SAP system whose objects you are going to access when working with the local repository.

Syntax
SetSAPSystem (aSAPSystem : ISAPSystem) : Boolean

Parameters

| aSAPSystem | An ISAPSystem object |

Return Value
Return True if successful, or False if failure.
WriteAppHierarchies

Purpose
Downloads an application hierarchies collection into the local repository.

Syntax
WriteAppHierarchies(aSAPSystem : ISAPSystem, aApplicationHierarchies : IApplicationHierarchies) : Boolean

Parameters
<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>aSAPSystem</td>
<td>An ISAPSystem object, identifying the R/3 system to which the application hierarchies collection belongs</td>
</tr>
<tr>
<td>aApplicationHierarchies</td>
<td>The IApplicationHierarchies object that will be stored in the local repository</td>
</tr>
</tbody>
</table>

Return Value
Return True if successful, or False if failure.
WriteAppHierarchy

Purpose
Downloads an application hierarchy and all of its children objects into the local repository. The following diagram shows the objects that are downloaded into the local repository.
WriteAppHierarchy

**Syntax**

\[
\text{WriteAppHierarchy}(\text{aSAPSystem} : \text{ISAPSystem}, \text{aApplicationHierarchy} : \text{IApplicationHierarchy}) : \text{Boolean}
\]

**Parameters**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>aSAPSystem</td>
<td>An ISAPSystem object, identifying the R/3 system to which the application hierarchy object belongs</td>
</tr>
<tr>
<td>aApplicationHierarchy</td>
<td>The IApplicationHierarchy object that will be stored in the local repository</td>
</tr>
</tbody>
</table>

**Return Value**

Return True if successful, or False if failure.

**See Also**

[WriteAppHierarchies][1]  

---

[1]: Page 50
WriteBO

Purpose
Downloads a business object into the local repository.

Syntax
WriteBO(aSAPSystem : ISAPSystem, aBusinessObject : IBusinessObject) : Boolean

Parameters
<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>aSAPSystem</td>
<td>An ISAPSystem object, identifying the R/3 system to which the business object belongs</td>
</tr>
<tr>
<td>aBusinessObject</td>
<td>The IBusinessObject object that will be stored in the local repository</td>
</tr>
</tbody>
</table>

Return Value
Return True if successful, or False if failure.

See Also
WriteBOTree [Page 54]
WriteBOTree

Purpose

Downloads a business object and all of its children objects into the local repository.

The following diagram shows the objects that are downloaded as part of a business object's tree.
Syntax
WriteBOTree(aSAPSystem : ISAPSystem, aBusinessObject : IBusinessObject) : Boolean

Parameters
<table>
<thead>
<tr>
<th>aSAPSystem</th>
<th>An ISAPSystem object, identifying the R/3 system to which the business object tree belongs</th>
</tr>
</thead>
<tbody>
<tr>
<td>aBusinessObject</td>
<td>The IBusinessObject object whose tree will be stored in the local repository</td>
</tr>
</tbody>
</table>

Return Value
Return True if successful, or False if failure.

See Also
WriteBO [Page 53]
WriteFunctionGroup

Purpose
Downloads a function group object and all of its children objects into the local repository.
The following diagram shows the objects that are downloaded into the local repository.

Syntax
WriteFunctionGroups(aSAPSystem : ISAPSystem, aFunctionGroup : IFunctionGroup) : Boolean
Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>aSAPSystem</td>
<td>An ISAPSystem object, identifying the R/3 system to which the function group belongs</td>
</tr>
<tr>
<td>aFunctionGroup</td>
<td>The IFunctionGroup object that will be stored in the local repository</td>
</tr>
</tbody>
</table>

Return Value
Return True if successful, or False if failure.

See Also
WriteFunctionGroups [Page 58], WriteRFC [Page 59]
WriteFunctionGroups

Purpose
Downloads a function group collection into the local repository.

Syntax
WriteFunctionGroups(aSAPSystem : ISAPSystem, aFunctionGroups : IFunctionGroups) : Boolean

Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>aSAPSystem</td>
<td>An ISAPSystem object, identifying the R/3 system to which the business object belongs</td>
</tr>
<tr>
<td>aFunctionGroups</td>
<td>The IFunctionGroups object that will be stored in the local repository</td>
</tr>
</tbody>
</table>

Return Value
Return True if successful, or False if failure.

See Also
WriteFunctionGroup [Page 56], WriteRFC [Page 59]
**WriteRFC**

**Purpose**
Downloads a function object and all of its children into the local repository.
The following diagram shows the objects that are downloaded into the local repository.

![Diagram showing objects downloaded into local repository]

**Syntax**

```plaintext
WriteRFC(aSAPSystem : ISAPSystem, aFunction : Ifunction) : Boolean
```
WriteRFC

**Parameters**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>aSAPSystem</td>
<td>An ISAPSystem object, identifying the R/3 system to which the function object belongs</td>
</tr>
<tr>
<td>aFunction</td>
<td>The IFunction object that will be stored in the local repository</td>
</tr>
</tbody>
</table>

**Return Value**

Return True if successful, or False if failure.

**See Also**

[WriteFunctionGroup](#), [WriteFunctionGroups](#)
ISAPSystems

Purpose
An Interface for a SAP R/3 system collection that contains the SAP R/3 system objects.

Properties

<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Count [Page 152]</td>
<td>Gets the number of items in a SAP R/3 system collection</td>
</tr>
<tr>
<td>Item [Page 157]</td>
<td>Given an index or a name, returns an item in the collection</td>
</tr>
<tr>
<td>ItemByIndex [Page 158]</td>
<td>Given an index, returns an item in the collection</td>
</tr>
<tr>
<td>ItemByName [Page 159]</td>
<td>Given a name, returns an item in the collection</td>
</tr>
<tr>
<td>Parent [Page 162]</td>
<td>Gets the parent of a SAP R/3 system collection</td>
</tr>
<tr>
<td>ParentType [Page 163]</td>
<td>Gets the parent type of a SAP R/3 system collection</td>
</tr>
<tr>
<td>Root [Page 164]</td>
<td>Gets the IRepositoryServices object at the root of the ISAPSystems' hierarchy</td>
</tr>
</tbody>
</table>
ISAPSystem

Purpose
An Interface for an SAP R/3 system that contains the metadata of an R/3 System.

Properties

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ApplicationServer [Page 64]</td>
<td>Gets the SAP R/3 application server name</td>
</tr>
<tr>
<td>HostName [Page 65]</td>
<td>Gets the hostname of the SAP R/3 system</td>
</tr>
<tr>
<td>Name [Page 160]</td>
<td>Gets the SAP R/3 system name</td>
</tr>
<tr>
<td>Parent [Page 162]</td>
<td>Gets the parent of the SAP R/3 system object</td>
</tr>
<tr>
<td>ParentType [Page 163]</td>
<td>Gets the parent type of a SAP R/3 system object</td>
</tr>
<tr>
<td>Root [Page 164]</td>
<td>Gets the IRepositoryServices object at the root of the ISAPSystem's hierarchy</td>
</tr>
<tr>
<td>SAPRelease [Page 66]</td>
<td>Gets the SAP R/3 system release</td>
</tr>
<tr>
<td>System [Page 67]</td>
<td>Gets SAP R/3 system description</td>
</tr>
<tr>
<td>SystemNumber [Page 68]</td>
<td>Gets the system number of the SAP R/3 system</td>
</tr>
</tbody>
</table>
Unique Properties
**ApplicationServer**

**Purpose**
Gets the SAP R/3 application server name.

**Syntax**
ApplicationServer : String = Null

**Parameters**
None.

**Return Value**
Returns the application server name, or null if failure.
HostName

Purpose
Gets the hostname of the SAP R/3 system.

Syntax
HostName : String = Null

Parameters
None.

Return Value
Returns the host name, or null if failure.
**SAPRelease**

**Purpose**
Gets the SAP R/3 system release.

**Syntax**

```plaintext
SAPRelease : String = Null
```

**Parameters**

None.

**Return Value**

Returns the SAP R/3 system release, or null if failure.
**System**

**Purpose**
Gets an SAP R/3 system description.

**Syntax**
System : String = Null

**Parameters**
None.

**Return Value**
Returns the SAP R/3 system description, or null if failure.
SystemNumber

Purpose
Gets the system number of the SAP R/3 system.

Syntax
SystemNumber : Long = 0

Parameters
None.

Return Value
Returns the system number.
IApplicationHierarchies

**Purpose**
An Interface for an application hierarchy collection that contains application hierarchy objects.

**Properties**

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Count</td>
<td>Gets the number of items in an application hierarchy collection</td>
</tr>
<tr>
<td>Item</td>
<td>Given an index or a name, return an item in the collection</td>
</tr>
<tr>
<td>ItemByIndex</td>
<td>Given an index, return an item in the collection</td>
</tr>
<tr>
<td>ItemByName</td>
<td>Given a name, return an item in the collection</td>
</tr>
<tr>
<td>Parent</td>
<td>Gets the parent of an application hierarchy collection</td>
</tr>
<tr>
<td>ParentType</td>
<td>Gets the parent type of an application hierarchy collection</td>
</tr>
<tr>
<td>Root</td>
<td>Gets the IRepositoryServices object at the root of the IApplicationHierarchies' hierarchy</td>
</tr>
</tbody>
</table>

April 2001
IApplicationHierarchy

Purpose
An Interface for an application hierarchy that contains metadata of an application hierarchy.

Properties

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ApplicationArea</td>
<td>Gets the application area of an application hierarchy</td>
</tr>
<tr>
<td>ApplicationAreaDescription</td>
<td>Gets the application description of an application hierarchy</td>
</tr>
<tr>
<td>Parent</td>
<td>Gets the parent of an application hierarchy</td>
</tr>
<tr>
<td>ParentType</td>
<td>Gets the parent type of an application hierarchy</td>
</tr>
<tr>
<td>Root</td>
<td>Gets the IRepositoryServices object at the root of the IApplicationHierarchy's hierarchy</td>
</tr>
</tbody>
</table>

Methods

<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>BusinessObjects</td>
<td>Gets a business object collection containing the business objects of an application hierarchy</td>
</tr>
</tbody>
</table>
Unique Properties
ApplicationAreaDescription

Purpose
Gets the application description of an application hierarchy.

Syntax
ApplicationAreaDescription : String = Null

Parameters
None.

Return Value
Returns the application description, or null if failure.
Unique Methods
BusinessObjects

Purpose
Gets the business objects of an application hierarchy.

Syntax
BusinessObjects () : IBusinessObjects

Parameters
None.

Return Value
Returns a business object collection, or null if failure.
**IBusinessObjects**

**Purpose**
An Interface for a business object collection that contains business objects.

**Properties**

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Count [Page 152]</td>
<td>Gets the number of items in a business object collection</td>
</tr>
<tr>
<td>Item [Page 157]</td>
<td>Given an index or a name, return an item in the collection</td>
</tr>
<tr>
<td>ItemByIndex [Page 158]</td>
<td>Given an index, return an item in the collection</td>
</tr>
<tr>
<td>ItemByName [Page 159]</td>
<td>Given a name, return an item in the collection</td>
</tr>
<tr>
<td>Parent [Page 162]</td>
<td>Gets the parent of a business object collection</td>
</tr>
<tr>
<td>ParentType [Page 163]</td>
<td>Gets the parent type of a business object collection</td>
</tr>
<tr>
<td>Root [Page 164]</td>
<td>Gets the IRepositoryServices object at the root of the IBusinessObjects' hierarchy</td>
</tr>
</tbody>
</table>
**IBusinessObject**

**Purpose**
An Interface for a business object that contains metadata of a business object.

**Properties**

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ApplicationArea [Page 151]</td>
<td>Gets the application area of a business object</td>
</tr>
<tr>
<td>Description [Page 153]</td>
<td>Gets the description of a business object</td>
</tr>
<tr>
<td>DevelopmentClass [Page 78]</td>
<td>Gets the development class a business object</td>
</tr>
<tr>
<td>Documentation [Page 155]</td>
<td>Gets the documentation of a business object</td>
</tr>
<tr>
<td>InternalName [Page 156]</td>
<td>Gets the object type of a business object</td>
</tr>
<tr>
<td>LastChangeDate [Page 79]</td>
<td>Gets the date on which the business object is changed</td>
</tr>
<tr>
<td>LastChangeRelease [Page 80]</td>
<td>Gets the R/3 release of the last change on a business object</td>
</tr>
<tr>
<td>LastChangeTime [Page 81]</td>
<td>Gets the time at which a business object is changed</td>
</tr>
<tr>
<td>Name [Page 160]</td>
<td>Gets the name of a business object</td>
</tr>
<tr>
<td>Obsolete [Page 161]</td>
<td>Gets release as of which the object type has been marked as obsolete</td>
</tr>
<tr>
<td>Parent [Page 162]</td>
<td>Gets the parent of a business object</td>
</tr>
<tr>
<td>ParentType [Page 163]</td>
<td>Gets the parent type of a business object</td>
</tr>
<tr>
<td>Root [Page 164]</td>
<td>Gets the IRepositoryServices object at the root of the IBusinessObject's hierarchy</td>
</tr>
</tbody>
</table>

**Methods**

<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>KeyFields [Page 83]</td>
<td>Gets the key fields of a business object</td>
</tr>
<tr>
<td>Methods [Page 84]</td>
<td>Gets the methods of a business object</td>
</tr>
</tbody>
</table>
Unique Properties
DevelopmentClass

**Purpose**

Gets the development class of a business object.

**Syntax**

DevelopmentClass : String = Null

**Parameters**

None.

**Return Value**

Returns the development class or null, if failure.
**LastChangeDate**

**Purpose**
Gets the date on which the business object has last changed.

**Syntax**

```plaintext
LastChangeDate : String = Null
```

**Parameters**

None.

**Return Value**

Returns the changed date or null, if failure.
LastChangeRelease

Purpose
Gets the R/3 release of the last change to a business object.

Syntax
LastChangeRelease : String = Null

Parameters
None.

Return Value
Returns the R/3 release of the last change or null, if failure.
LastChangeTime

Purpose
Gets the time at which the business object has last changed.

Syntax
LastChangeTime : String = Null

Parameters
None.

Return Value
Returns the changed time or null, if failure.
Unique Methods
KeyFields

Purpose
Gets the key fields of a business object.

Syntax
KeyFields () : IFields

Parameters
None.

Return Value
Returns a collection of fields or null, if not found.
Methods

Purpose
Gets the methods of a business object.

Syntax
Methods () : IMethods

Parameters
None.

Return Value
Returns a collection of methods or null, if not found.
IMethods

Purpose
An Interface for a method collection that contains method objects.

Properties

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Count [Page 152]</td>
<td>Gets the number of items in a method collection</td>
</tr>
<tr>
<td>Item [Page 157]</td>
<td>Given an index or a name, returns an item in the collection</td>
</tr>
<tr>
<td>ItemByIndex [Page 158]</td>
<td>Given an index, returns an item in the collection</td>
</tr>
<tr>
<td>ItemByName [Page 159]</td>
<td>Given a name, returns an item in the collection</td>
</tr>
<tr>
<td>Parent [Page 162]</td>
<td>Gets the parent of a method collection</td>
</tr>
<tr>
<td>ParentType [Page 163]</td>
<td>Gets the parent type of a method collection</td>
</tr>
<tr>
<td>Root [Page 164]</td>
<td>Gets the IRepositoryServices object at the root of the IMethods' hierarchy</td>
</tr>
</tbody>
</table>
**IMethod**

**Purpose**
An Interface for a method that contains metadata of a method.

**Properties**

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Description [Page 153]</td>
<td>Gets the description of a method</td>
</tr>
<tr>
<td>Documentation [Page 155]</td>
<td>Gets the documentation of a method</td>
</tr>
<tr>
<td>InternalName [Page 156]</td>
<td>Gets the object type of a method</td>
</tr>
<tr>
<td>IsClassMethod [Page 88]</td>
<td>Determines if the method is a class method</td>
</tr>
<tr>
<td>IsFactoryMethod [Page 89]</td>
<td>Decides if the method is a factory method</td>
</tr>
<tr>
<td>IsFrozen [Page 90]</td>
<td>Determines if the DDIC structure is frozen</td>
</tr>
<tr>
<td>IsModelOnly [Page 91]</td>
<td>Determines if the method is modeled only</td>
</tr>
<tr>
<td>IsSynchronousMethod [Page 92]</td>
<td>Decides if the method is a synchronous method</td>
</tr>
<tr>
<td>MethodType [Page 93]</td>
<td>Gets the type of a method</td>
</tr>
<tr>
<td>Name [Page 160]</td>
<td>Gets the name of a method</td>
</tr>
<tr>
<td>Obsolete [Page 161]</td>
<td>Gets the release as of which the object type has been marked as obsolete</td>
</tr>
<tr>
<td>Parent [Page 162]</td>
<td>Gets the parent of a method</td>
</tr>
<tr>
<td>ParentType [Page 163]</td>
<td>Gets the parent type of a method</td>
</tr>
<tr>
<td>Root [Page 164]</td>
<td>Gets the IRepositoryServices object at the root of the IMethod's hierarchy</td>
</tr>
</tbody>
</table>

**Methods**

<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Exceptions [Page 167]</td>
<td>Gets the exceptions of a method</td>
</tr>
<tr>
<td>Parameters [Page 169]</td>
<td>Gets the parameters of a method</td>
</tr>
</tbody>
</table>
Unique Properties
IsClassMethod

**Purpose**
Determines if the method is a class method.

**Syntax**
IsClassMethod : Boolean = False

**Parameters**
None.

**Return Value**
Returns true if it is a class method, and false if it is not.

**Description**
If a method is neither a class nor a factory, then it is an instance.
IsFactoryMethod

Purpose
Determines if the method is a factory method.

Syntax
IsFactoryMethod : Boolean = False

Parameters
None.

Return Value
Returns true if it is a factory method, and false if it is not.

Remarks
If a method is neither a class nor a factory, then it is an instance.
IsFrozen

**Purpose**
Determines if the DDIC structure is frozen.

**Syntax**
IsFrozen : Boolean = False

**Parameters**
None.

**Return Value**
Returns true if it is, and false if it is not.

**Remarks**
For 4.0A release and higher
IsModelIndex

Purpose
Determines if the method is modeled only.

Syntax
IsModelIndex : Boolean = False

Parameters
None.

Return Value
Returns true if it is modeled, and false if it is not.
IsSynchronousMethod

**Purpose**
Decides if the method is a synchronous method.

**Syntax**
IsSynchronousMethod : Boolean = False

**Parameters**
None.

**Return Value**
Returns true if the method is synchronous, and false if it is not.
**MethodType**

**Purpose**
Gets the type of a method.

**Syntax**

```
MethodType : BO_METHOD_TYPE = UNKNOWN
```

**Parameters**
None.

**Return Value**
See the [BO_METHOD_TYPE][171] topic for a list of its values.

**Remarks**
If a method is neither a class nor a factory, then it is an instance.
IFunctionGroups

Purpose
An Interface, which is collection that contains function group objects.

Properties

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Count</strong></td>
<td>Gets the number of items in a function group collection</td>
</tr>
<tr>
<td><strong>Item</strong></td>
<td>Given an index or a name, returns an item in the collection</td>
</tr>
<tr>
<td><strong>ItemByIndex</strong></td>
<td>Given an index, returns an item in the collection</td>
</tr>
<tr>
<td><strong>ItemByName</strong></td>
<td>Given a name, returns an item in the collection</td>
</tr>
<tr>
<td><strong>Parent</strong></td>
<td>Gets the parent of a function group collection</td>
</tr>
<tr>
<td><strong>ParentType</strong></td>
<td>Gets the parent type of a function group collection</td>
</tr>
<tr>
<td><strong>Root</strong></td>
<td>Gets the <code>IRepositoryServices</code> object at the root of the IFunctionGroups' hierarchy</td>
</tr>
</tbody>
</table>
**IFunctionGroup**

**Purpose**
An Interface for a function group and contains the metadata of a function group.

**Properties**

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Description</td>
<td>Gets the description of a function group</td>
</tr>
<tr>
<td>Documentation</td>
<td>Gets the documentation of a function group</td>
</tr>
<tr>
<td>Name</td>
<td>Gets the name of a function group</td>
</tr>
<tr>
<td>Parent</td>
<td>Gets the parent of a function group</td>
</tr>
<tr>
<td>ParentType</td>
<td>Gets the parent type of a function group</td>
</tr>
<tr>
<td>Root</td>
<td>Gets the IRepositoryServices object at the root of the IFunctionGroup's hierarchy</td>
</tr>
</tbody>
</table>

**Methods**

<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Functions</td>
<td>Gets the functions of a function group</td>
</tr>
</tbody>
</table>
Unique Methods
Functions

Purpose
Gets the functions of a function group.

Syntax
Functions () : IFunctions

Parameters
None.

Return Value
Returns a function collection, or null if failure.
IFunctions

Purpose
An Interface for a function collection that contains function objects.

Properties

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Count [Page 152]</td>
<td>Gets the number of items in a function collection</td>
</tr>
<tr>
<td>Item [Page 157]</td>
<td>Given an index or a name, returns an item in the collection</td>
</tr>
<tr>
<td>ItemByIndex [Page 158]</td>
<td>Given an index, returns an item in the collection</td>
</tr>
<tr>
<td>ItemByName [Page 159]</td>
<td>Given a name, returns an item in the collection</td>
</tr>
<tr>
<td>Parent [Page 162]</td>
<td>Gets the parent of a function collection</td>
</tr>
<tr>
<td>ParentType [Page 163]</td>
<td>Gets the parent type of a function collection</td>
</tr>
<tr>
<td>Root [Page 164]</td>
<td>Gets the IRepositoryServices object at the root of the IFunctions' hierarchy</td>
</tr>
</tbody>
</table>
IFunction

Purpose
An Interface for a function that contains the metadata of a function.

Properties

<table>
<thead>
<tr>
<th>Documentation [Page 155]</th>
<th>Gets the documentation of a function</th>
</tr>
</thead>
<tbody>
<tr>
<td>GroupName [Page 101]</td>
<td>Gets the group name of a function</td>
</tr>
<tr>
<td>Name [Page 160]</td>
<td>Gets the name of a function</td>
</tr>
<tr>
<td>Parent [Page 162]</td>
<td>Gets the parent of a function</td>
</tr>
<tr>
<td>ParentType [Page 163]</td>
<td>Gets the parent type of a function</td>
</tr>
<tr>
<td>Root [Page 164]</td>
<td>Gets the IRepositoryServices object at the root of the IFunction's hierarchy</td>
</tr>
</tbody>
</table>

Methods

<table>
<thead>
<tr>
<th>Exceptions [Page 167]</th>
<th>Gets all exception messages of a function</th>
</tr>
</thead>
<tbody>
<tr>
<td>Parameters [Page 169]</td>
<td>Gets the parameters of a function</td>
</tr>
</tbody>
</table>
Unique Properties
**GroupName**

**Purpose**
Gets the group name of a function.

**Syntax**

```java
GroupName : String = Null
```

**Parameters**
None.

**Return Value**
Returns the function group name, or null if failure.
IExceptions

Purpose
An Interface for an exception collection that contains exception objects.

Properties

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Count [Page 152]</td>
<td>Gets the number of items in an exception collection</td>
</tr>
<tr>
<td>Item [Page 157]</td>
<td>Given an index or a name, return an item in the collection</td>
</tr>
<tr>
<td>ItemByIndex [Page 158]</td>
<td>Given an index, return an item in the collection</td>
</tr>
<tr>
<td>ItemByName [Page 159]</td>
<td>Given a name, return an item in the collection</td>
</tr>
<tr>
<td>Parent [Page 162]</td>
<td>Gets the parent of an exception collection</td>
</tr>
<tr>
<td>ParentType [Page 163]</td>
<td>Gets the parent type an exception collection</td>
</tr>
<tr>
<td>Root [Page 164]</td>
<td>Gets the IRepositoryServices object at the root of the IExceptions' hierarchy</td>
</tr>
</tbody>
</table>
IException

Purpose
An Interface for an exception that contains the metadata of an exception.

Properties

<table>
<thead>
<tr>
<th>Description [Page 153]</th>
<th>Gets the description of an exception</th>
</tr>
</thead>
<tbody>
<tr>
<td>ID [Page 105]</td>
<td>Gets the ID of an exception</td>
</tr>
<tr>
<td>Parent [Page 162]</td>
<td>Gets the parent of an exception</td>
</tr>
<tr>
<td>ParentType [Page 163]</td>
<td>Gets the parent type of an exception</td>
</tr>
<tr>
<td>Root [Page 164]</td>
<td>Gets the IRepositoryServices object at the root of the IException's hierarchy</td>
</tr>
</tbody>
</table>
Unique Properties
ID

Purpose
Gets the ID of an exception.

Syntax
ID : String = Null

Parameters
None.

Return Value
Returns the exception ID, or null if failure.
**IParameters**

**Purpose**
An Interface for a parameter collection that contains parameter objects.

**Properties**

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Count</td>
<td>Gets the number of items in a parameter collection</td>
</tr>
<tr>
<td>Item</td>
<td>Given an index or a name, returns an item in the collection</td>
</tr>
<tr>
<td>ItemByIndex</td>
<td>Given an index, returns an item in the collection</td>
</tr>
<tr>
<td>ItemByName</td>
<td>Given a name, returns an item in the collection</td>
</tr>
<tr>
<td>Parent</td>
<td>Gets the parent of a parameter collection</td>
</tr>
<tr>
<td>ParentType</td>
<td>Gets the parent type of a parameter collection</td>
</tr>
<tr>
<td>Root</td>
<td>Gets the IRepositoryServices object at the root of the IParameters' hierarchy</td>
</tr>
</tbody>
</table>
IParameter

Purpose
An Interface for a parameter that contains the metadata of a parameter.

Properties

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ABAPName [Page 110]</td>
<td>Gets the ABAP name of a parameter, which is not applicable to RFC</td>
</tr>
<tr>
<td>ABAPType [Page 150]</td>
<td>Gets the ABAP data type of a parameter</td>
</tr>
<tr>
<td>Default [Page 111]</td>
<td>Gets the default value of a parameter</td>
</tr>
<tr>
<td>Description [Page 153]</td>
<td>Gets the description of a parameter</td>
</tr>
<tr>
<td>DictionaryType [Page 154]</td>
<td>Gets the screen data type of a parameter for the Screen Painter tool</td>
</tr>
<tr>
<td>Documentation [Page 155]</td>
<td>Gets the documentation of a parameter</td>
</tr>
<tr>
<td>InternalName [Page 156]</td>
<td>Gets the reference object type of a parameter</td>
</tr>
<tr>
<td>IsExport [Page 113]</td>
<td>Determines if the parameter is an export parameter</td>
</tr>
<tr>
<td>IsField [Page 114]</td>
<td>Determines if the parameter is a field parameter</td>
</tr>
<tr>
<td>IsImport [Page 115]</td>
<td>Determines if the parameter is an import parameter</td>
</tr>
<tr>
<td>IsMandatory [Page 116]</td>
<td>Determines if the parameter is mandatory</td>
</tr>
<tr>
<td>IsStructure [Page 112]</td>
<td>Determines if the parameter is a structure parameter</td>
</tr>
<tr>
<td>IsTable [Page 117]</td>
<td>Determines if the parameter is a table parameter</td>
</tr>
<tr>
<td>Name [Page 160]</td>
<td>Gets the name of a parameter</td>
</tr>
<tr>
<td>Parent [Page 162]</td>
<td>Gets the parent of a parameter</td>
</tr>
<tr>
<td>ParentType [Page 163]</td>
<td>Gets the parent type of a parameter</td>
</tr>
<tr>
<td>Root [Page 164]</td>
<td>Gets the IRepositoryServices object at the root of the IParameter's hierarchy</td>
</tr>
</tbody>
</table>

Methods

<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Field [Page 119]</td>
<td>Gets the field of a parameter</td>
</tr>
<tr>
<td>Structure [Page 120]</td>
<td>Gets the structure of a parameter</td>
</tr>
</tbody>
</table>
### IPARAMETER

| Table [Page 121] | Gets the table of a parameter |
Unique Properties
ABAPName

Purpose
Gets the ABAP name of a parameter, which is not applicable to RFC.

Syntax
ABAPName : String = Null

Parameters
None.

Return Value
Returns one of the following values:
- The ABAP name, if the parameter is from BAPI.
- Null, if the parameter is from RFC.
- Null, if failure.
Default

Purpose
Gets the default value of a parameter.

Applies To
This property is only for a simple parameter (field) and does not apply to Table or Structure.

Syntax
Default : String = Null

Parameters
None.

Return Value
Returns the default value, or null if failure.
IsStructure

**Purpose**
Determines if the parameter is a structure parameter.

**Syntax**
IsStructure : Boolean = False

**Parameters**
None.

**Return Value**
Returns true if the parameter is a structure, and false if it is not.
**IsExport**

**Purpose**
Determines if the parameter is an export parameter.

**Syntax**

IsExport : Boolean = False

**Parameters**
None.

**Return Value**
Returns true, if the parameter is an export parameter, and false if it is not.
IsField

**Purpose**
Determines if the parameter is a field parameter.

**Syntax**

```
IsField : Boolean = false
```

**Parameters**
None.

**Return Value**
Returns true if the parameter is a field, and false if it is not.
**IsImport**

**Purpose**
Determines if the parameter is an import parameter.

**Syntax**

\[
\text{IsImport : Boolean = False}
\]

**Parameters**

None.

**Return Value**

Returns true, if the parameter is an import parameter, and false if it is not.
IsMandatory

**Purpose**
Determines if the parameter is mandatory.

**Syntax**
IsMandatory : Boolean = False

**Parameters**
None.

**Return Value**
Returns true if the parameter is mandatory, and false if it is not.
**IsTable**

**Purpose**
Determines if the parameter is a table parameter.

**Syntax**

```
IsTable : Boolean = False
```

**Parameters**
None.

**Return Value**
Returns true, if the parameter is a table, and false if it is not.
Unique Methods
Field

Purpose
Gets the field of a parameter.

Syntax
Field () : IField

Parameters
None.

Return Value
Returns the field, or null if failure.
Structure

Purpose
Gets the structure of a parameter.

Syntax
Structure () : IStructure

Parameters
None.

Return Value
Returns the structure, or null if failure.
Table

Purpose
Gets the table of a parameter.

Syntax
Table () : ITable

Parameters
None.

Return Value
Returns the table, or null if failure.
IStructure

Purpose
An Interface for a structure that contains the metadata of a structure.

Properties

| Name [Page 160] | Gets the name of a structure |
| Parent [Page 162] | Gets the parent of a structure |
| ParentType [Page 163] | Gets the parent type of a structure |
| Root [Page 164] | Gets the IRepositoryServices object at the root of the IStructure's hierarchy |
| RowLength [Page 165] | Gets the length of a structure |

Methods

| Fields [Page 168] | Gets the fields of a structure |
ITable

Purpose
An Interface for a table that contains the metadata of a table.

Properties

<table>
<thead>
<tr>
<th>Name [Page 160]</th>
<th>Gets the name of a table</th>
</tr>
</thead>
<tbody>
<tr>
<td>Parent [Page 162]</td>
<td>Gets the parent of a table</td>
</tr>
<tr>
<td>ParentType [Page 163]</td>
<td>Gets the parent type of a table</td>
</tr>
<tr>
<td>Root [Page 164]</td>
<td>Gets the IRepositoryServices object at the root of the ITable's hierarchy</td>
</tr>
<tr>
<td>RowLength [Page 165]</td>
<td>Gets the row length of a table</td>
</tr>
</tbody>
</table>

Methods

| Fields [Page 168] | Gets the fields in a table |
**IFields**

**Purpose**
An Interface for a field collection that contains field objects.

**Properties**

<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Count</strong></td>
<td>Gets the number of items in a field collection</td>
</tr>
<tr>
<td><strong>Item</strong></td>
<td>Given an index or a name, return an item in the collection</td>
</tr>
<tr>
<td><strong>ItemByIndex</strong></td>
<td>Given an index, return an item in the collection</td>
</tr>
<tr>
<td><strong>ItemByName</strong></td>
<td>Given a name, return an item in the collection</td>
</tr>
<tr>
<td><strong>Parent</strong></td>
<td>Gets the parent of a field collection</td>
</tr>
<tr>
<td><strong>ParentType</strong></td>
<td>Gets the parent type of a field collection</td>
</tr>
<tr>
<td><strong>Root</strong></td>
<td>Gets the <code>IRepositoryServices</code> object at the root of the IFields' hierarchy</td>
</tr>
</tbody>
</table>
IField

An Interface for a field that contains the metadata of a field.

Properties

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ABAPType [Page 150]</td>
<td>Gets the ABAP data type of a field</td>
</tr>
<tr>
<td>CheckTable [Page 128]</td>
<td>Gets the check table of a field</td>
</tr>
<tr>
<td>DataElement [Page 129]</td>
<td>Gets the data element (semantic domain) of a field</td>
</tr>
<tr>
<td>Decimals [Page 130]</td>
<td>Gets the number of decimals of a field</td>
</tr>
<tr>
<td>Description [Page 153]</td>
<td>Gets the description of a field</td>
</tr>
<tr>
<td>DictionaryType [Page 154]</td>
<td>Gets the screen data type of a field for the Screen Painter tool</td>
</tr>
<tr>
<td>Domain [Page 131]</td>
<td>Gets the domain of a field</td>
</tr>
<tr>
<td>HasFixedValues [Page 132]</td>
<td>Determines if the field contains fixed values</td>
</tr>
<tr>
<td>HelpValues [Page 133]</td>
<td>Given an index, return an item in the help value collection</td>
</tr>
<tr>
<td>HelpValuesCount [Page 134]</td>
<td>Gets the number of help values of a field</td>
</tr>
<tr>
<td>InternalLength [Page 135]</td>
<td>Gets the internal length of a field in bytes</td>
</tr>
<tr>
<td>InternalName [Page 156]</td>
<td>Gets the type name of a field</td>
</tr>
<tr>
<td>IsCaseSensitive [Page 136]</td>
<td>Gets the case sensitivity status of a field</td>
</tr>
<tr>
<td>IsConversionExit [Page 137]</td>
<td>Finds the field's conversion exit status</td>
</tr>
<tr>
<td>Length [Page 138]</td>
<td>Gets the width of a field</td>
</tr>
<tr>
<td>LongScreenText [Page 139]</td>
<td>Gets the long screen text of a field</td>
</tr>
<tr>
<td>MediumScreenText [Page 140]</td>
<td>Gets the medium screen text of a field</td>
</tr>
<tr>
<td>Name [Page 160]</td>
<td>Gets the name of a field</td>
</tr>
<tr>
<td>Offset [Page 141]</td>
<td>Gets the internal start address of a field</td>
</tr>
<tr>
<td>Parent [Page 162]</td>
<td>Gets the parent of a field</td>
</tr>
<tr>
<td>ParentType [Page 163]</td>
<td>Gets the parent type of a field</td>
</tr>
<tr>
<td>Position [Page 142]</td>
<td>Gets the position of a field in a table</td>
</tr>
<tr>
<td>ReferenceField [Page 143]</td>
<td>Gets the reference field for currency and quantity fields</td>
</tr>
<tr>
<td>Method</td>
<td>Description</td>
</tr>
<tr>
<td>--------------------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>ReferenceTable [Page 144]</td>
<td>Gets the reference table of a field</td>
</tr>
<tr>
<td>ReportHeader [Page 145]</td>
<td>Gets the report header of a field</td>
</tr>
<tr>
<td>Root [Page 164]</td>
<td>Gets the IRepositoryServices object at the root of the IField's hierarchy</td>
</tr>
<tr>
<td>ShortScreenText [Page 146]</td>
<td>Gets the short screen text of a field</td>
</tr>
<tr>
<td>ValuesForField [Page 147]</td>
<td>Given an index, return an item in the values for field collection</td>
</tr>
<tr>
<td>ValuesForFieldCount [Page 148]</td>
<td>Gets the number of values of a field</td>
</tr>
</tbody>
</table>
Unique Properties
CheckTable

Purpose
Gets the check table of a field.

Syntax
CheckTable : String = Null

Parameters
None.

Return Value
Returns the field's check table, or null if failure.
**DataElement**

**Purpose**
Gets the data element (semantic domain) of a field.

**Syntax**

```
DataElement : String = Null
```

**Parameters**
None.

**Return Value**

Returns the field data element (semantic domain), or null if failure.
Decimals

Purpose
Gets the number of decimals of a field.

Syntax
Decimals : Long = 0

Parameters
None.

Return Value
Returns the number of decimals in the field.
Domain

Purpose
Gets the domain of a field.

Syntax
Domain : String = Null

Parameters
None.

Return Value
Returns the field's domain, or null if failure.
HasFixedValues

**Purpose**
Determines if the field contains fixed values.

**Syntax**
HasFixedValues : Boolean = False

**Parameters**
None.

**Return Value**
Returns true, if there are fixed values, and false if there are no fixed values.
HelpValues

Purpose
Given an index, returns an item in the help value collection.

Syntax
HelpValues(aIndex : Long) : String = Null

Parameters

| aIndex | An index |

Return Value
Returns the requested help value, or null if failure.

See also:
HelpValuesCount [Page 134]
HelpValuesCount

**Purpose**
Gets the number of help values of a field.

**Syntax**
```
HelpValuesCount : Long = 0
```

**Parameters**
None.

**Return Value**
Returns the number of field help values.
**InternalLength**

**Purpose**
Gets the internal length of a field in bytes.

**Syntax**

```
InternalLength : Long = 0
```

**Parameters**

None.

**Return Value**

Returns the field's internal length.
IsCaseSensitive

**Purpose**
Gets the case sensitivity status of a field.

**Syntax**
IsCaseSensitive : Boolean = False

**Parameters**
None.

**Return Value**
Returns true, if the field is case sensitive and false, if it is not.
IsConversionExit

**Purpose**
Gets the field's conversion exit status.

**Syntax**
```plaintext
sConversionExit : Boolean = False
```

**Parameters**
None.

**Return Value**
Returns true, if there is a conversion exit, or false if there is no exit.
Length

Purpose
Gets the width of a field.

Syntax
Length : Long = 0

Parameters
None.

Return Value
Returns the field's width.
**LongScreenText**

**Purpose**
Gets the long screen text of a field.

**Syntax**

```
LongScreenText : String = Null
```

**Parameters**
None.

**Return Value**

Returns the long screen text, or null if failure.
MediumScreenText

**Purpose**
Gets the medium screen text of a field.

**Syntax**
MediumScreenText : String = Null

**Parameters**
None.

**Return Value**
Returns the medium screen text, or null if failure.
Offset

Purpose
Gets the internal start address of a field.

Syntax
Offset : Long = 0

Parameters
None.

Return Value
Returns the field's internal start address.
Position

**Purpose**
Gets the position of a field in a table.

**Syntax**
Position : Long = 0

**Parameters**
None.

**Return Value**
Returns the field's position in a table.
**ReferenceField**

**Purpose**
Gets the reference field for currency and quantity fields.

**Syntax**

```
ReferenceField : String = Null
```

**Parameters**

None.

**Return Value**

Returns the field's reference field, or null if failure.
ReferenceTable

**Purpose**

Gets the reference table of a field.

**Syntax**

ReferenceTable : String = Null

**Parameters**

None.

**Return Value**

Returns the field reference table, or null if failure.
ReportHeader

Purpose
Gets the report header of a field.

Syntax
ReportHeader : String = Null

Parameters
None.

Return Value
Returns the field report header, or null if failure.
ShortScreenText

**Purpose**
Gets the short screen text of a field.

**Syntax**
ShortScreenText : String = Null

**Parameters**
None.

**Return Value**
Returns the short screen text, or null if failure.
**ValuesForField**

**Purpose**
Given an index, returns an item in the values for field collection.

**Syntax**
ValuesForField(aIndex : Long) : String = Null

**Parameters**
- `aIndex` : An index

**Return Value**
Returns the requested value for field, or null if failure.

**See also:**
ValuesForFieldCount [Page 148]
ValuesForFieldCount

**Purpose**

Gets the number of values of a field.

**Syntax**

ValuesForFieldCount : Long = 0

**Parameters**

None.

**Return Value**

Returns the number of values for a field.
Common Properties
**ABAPType**

**Purpose**
Gets the ABAP data type of a field or a parameter.

**Applies To**
[IField [Page 125]], [IParameter [Page 107]]

**Syntax**

```
ABAPType : String = Null
```

**Parameters**
None.

**Return Value**
Returns the ABAP data type of the field or parameter, or null if failure.
ApplicationArea

Purpose
Gets the application area of an application hierarchy or a business object.

Applies To
IAplicationHierarchy [Page 70], IBusinessObject [Page 76]

Syntax
ApplicationArea : String = Null

Parameters
None.

Return Value
Returns the application area, or null if failure.
Count

**Purpose**

Gets the number of items in a collection of objects.

**Applies To**


**Syntax**

Count : Integer = 0

**Parameters**

None.

**Return Value**

Returns the number of objects in the collection.
Description

Purpose
 Gets the description of the object.

Applies To
 IFункцияГруппа [Page 95], IBusinessObject [Page 76], IМетод [Page 86], IПараметр [Page 107], IОшибка [Page 103], IПоле [Page 125]

Syntax
 Description : String = Null

Parameters
 None.

Return Value
 Returns the object's description, or null if failure.
DictionaryType

**Purpose**
Gets the screen data type of the object for the Screen Painter tool.

**Applies To**
IField [Page 125], IParameter [Page 107].

**Syntax**
DictionaryType : String = Null

**Parameters**
None.

**Return Value**
Returns the field's data type, or null if failure.
Documentation

Purpose
Gets the documentation of the object.

Applies To
 IBusinessObject [Page 76], IFunction [Page 99], IFunctionGroup [Page 95], IMethod [Page 86], IParameter [Page 107].

Syntax
Documentation : String = Null

Parameters
None.

Return Value
Returns the object's documentation, or null if failure.
InternalName

Purpose
Gets the object type of a business object, a method, a parameter, or a field.

Applies To
IBusinessObject [Page 76], IMethod [Page 86], IParameter [Page 107], IField [Page 125].

Syntax
InternalName : String = Null

Parameters
None.

Return Value
Returns the object type or null, if failure.
Item

Purpose
Given an index or a name, returns an item in the collection.

Applies To

Syntax
Item (Index : Variant) : Object = Null

Parameters

| Index        | A name or an index |

Return Value
Returns the requested object, or null if failure.

The type of the returned object (Object in the above syntax) depends on the collection. For an ISAPSystems collection for example, the returned object is an ISAPSystem object.

See Also
Count() [Page 152]
ItemByIndex

Purpose
Given an index, returns an item in the collection.

Applies To

Syntax
ItemByIndex (aIndex : Integer) : Object = Null

Where Object is the item in the collection, for example, for an SAPSystems collection the syntax is:
ItemByIndex (aIndex : Integer) : ISAPSystem = Null

Parameters

\[\text{aIndex} \quad \text{An index}\]

Return Value
Returns the requested object, or null if failure.

The type of the returned object (Object in the above syntax) depends on the collection. For an SAPSystems collection for example, the returned object is an ISAPSystem object.

See also
Count() [Page 152]
**ItemByName**

**Purpose**
Given an object name, returns the item in the collection. For example, given a SAP R/3 system name, it returns an item in the ISAPSystems collection.

**Applies To**

**Syntax**
ItemByName (aName : String) : Object = Null

**Parameters**

| aName          | Object name, for example, an SAP R/3 system name |

**Return Value**
Returns the requested SAP R/3 system, or null if failure.

The type of the returned object (Object in the above syntax) depends on the collection. For an ISAPSystems collection for example, the returned object is an ISAPSystem object.
Name

Purpose
Gets the name of the object.

Applies To
IFunction [Page 99], IFunctionGroup [Page 95], IBusinessObject [Page 76], IMethod [Page 86], IParameter [Page 107], ISAPSystem [Page 62], IStructure [Page 122], ITable [Page 123], IField [Page 125].

Syntax
Name : String = Null

Parameters
None.

Return Value
Returns the object's name, or null if failure.
Obsolete

Purpose
Gets the release as of which the object type has been marked as obsolete.

Applies To
IBusinessObject [Page 76], IMethod [Page 86].

Syntax
Obsolete : String = Null

Parameters
None.

Return Value
Returns the release as of which the object type has been marked as obsolete, or null if failure.

Remarks
For Release 4.0A and higher.
Parent

Purpose
Gets the parent of the collection or object.

Prerequisites
Before using this property, use the ParentType property to find out the parent type.

Applies To
All interfaces, except IRepositoryServices.

Syntax
Parent : Object = Null

Parameters
None.

Return Value
Returns the parent object of the object or collection, or null if the parent is not found.

Examples
For an ISAPSystems collection, this property returns one of the following values:
- An IRepositoryService object, if the parent is a repository services
- Null, if the parent is not found
For ISAPSystem object, this property returns one of the following values:
- An ISAPSystems object, if the parent is a SAP R/3 system collection
- Null, if the parent is not found
For an IFields collection, this property returns one of the following values:
- An ITable object, if the parent is a table
- An IStructure object, if the parent is a structure
- An IBusinessObject object, if the parent is a business object
- Null, if the parent is not found

See Also
ParentType
ParentType

Purpose
Gets the parent type of a collection or object. Use this property before using the Parent [Page 162] property.

Applies To
All interfaces, except IRepositoryServices.

Syntax
ParentType : OBJECT_TYPE = CUNKNOWN

Parameters
None.

Return Value
See OBJECT_TYPE [Page 172] for the list of values that may be returned. The returned value may either be CUNKNOWN, if the parent is not found, or it is the type of the parent object.

For example, for a field collection, this property returns one of the following values:

- CTABLE, if the parent is a table.
- CSTRUCTURE, if the parent is a structure.
- CBUSINESSOBJECT, if the parent is a business object.
- CUNKNOWN, if the parent is not found.
**Root**

**Purpose**
Gets the `IRepositoryServices` object at the root of the object's hierarchy.

**Applies To**
All interfaces, except `IRepositoryServices` itself.

**Syntax**
```
Root : IRepositoryServices = Null
```

**Parameters**
None.

**Return Value**
Returns an `IRepositoryServices` object, or null if failure.
RowLength

Purpose
Gets the length of a structure or one row in a table.

Applies To
IStructure [Page 122], ITable [Page 123].

Syntax
RowLength : Long = 0

Parameters
None.

Return Value
Returns the length of the structure or the row in the table.
Common Methods
Exceptions

Purpose
Gets the exception messages of a function or a method.

Applies To
IFunction [Page 99], IMethod [Page 86].

Syntax
Exceptions () : IExceptions

Parameters
None.

Return Value
Returns a set of messages, or null if failure.
Fields

Purpose
Gets all of the fields in a table or a structure.

Applies To
ITable [Page 123], IStructure [Page 122].

Syntax
Fields () : IFields

Parameters
None.

Return Value
Returns a field collection, or null if failure.
Parameters

Purpose
Gets the parameters of a function or a method.

Applies To
IFunction [Page 99], IMethod [Page 86].

Syntax
Parameters () : IParameters

Parameters
None.

Return Value
Returns a parameter collection, or null if there is no parameter.
Common Returned Values
BO_METHOD_TYPE

The BO_METHOD_TYPE is a returned value of an enumeration type.

<table>
<thead>
<tr>
<th>Attribute and Value</th>
<th>Indicates that the Method is…</th>
</tr>
</thead>
<tbody>
<tr>
<td>UNKNOWN = -1</td>
<td>unknown</td>
</tr>
<tr>
<td>CLASSM = 0</td>
<td>a class method</td>
</tr>
<tr>
<td>INSTANCENM = 1</td>
<td>an instance method</td>
</tr>
<tr>
<td>FACTORYM = 2</td>
<td>a factory method</td>
</tr>
</tbody>
</table>
The Object_Type returned value is of an enumeration type.

<table>
<thead>
<tr>
<th>Attribute and Value</th>
<th>Indicates that the Object is…</th>
</tr>
</thead>
<tbody>
<tr>
<td>CUNKNOWN = -1</td>
<td>unknown</td>
</tr>
<tr>
<td>CREPOSITORYSERVICES = 0</td>
<td>a repository services object</td>
</tr>
<tr>
<td>CBUSINESSOBJECTS = 1</td>
<td>a business object collection</td>
</tr>
<tr>
<td>CBUSINESSOBJECT = 2</td>
<td>a business object</td>
</tr>
<tr>
<td>CMETHODS = 3</td>
<td>a method collection</td>
</tr>
<tr>
<td>CMETHOD = 4</td>
<td>a method</td>
</tr>
<tr>
<td>CPARAMETERS = 5</td>
<td>a parameter collection</td>
</tr>
<tr>
<td>CPARAMETER = 6</td>
<td>a parameter</td>
</tr>
<tr>
<td>CSTRUCTURE = 7</td>
<td>a structure</td>
</tr>
<tr>
<td>CTABLE = 8</td>
<td>a table</td>
</tr>
<tr>
<td>CFIELDS = 9</td>
<td>a field collection</td>
</tr>
<tr>
<td>CFIELD = 10</td>
<td>a field</td>
</tr>
<tr>
<td>CAPPLICATIONHIERARCHIES = 11</td>
<td>an application hierarchy collection</td>
</tr>
<tr>
<td>CAPPLICATIONHIERARCHY = 12</td>
<td>an application hierarchy</td>
</tr>
<tr>
<td>CFUNCTIONGROUPS = 13</td>
<td>a function group collection</td>
</tr>
<tr>
<td>CFUNCTIONGROUP = 14</td>
<td>a function group</td>
</tr>
<tr>
<td>CFUNCTIONS = 15</td>
<td>a function collection</td>
</tr>
<tr>
<td>CFUNCTION = 16</td>
<td>a function</td>
</tr>
<tr>
<td>CEXCEPTIONS = 17</td>
<td>an exception collection</td>
</tr>
<tr>
<td>CEXCEPTION = 18</td>
<td>an exception</td>
</tr>
<tr>
<td>CSAPSYSTEMS = 25</td>
<td>a SAP system collection</td>
</tr>
<tr>
<td>CSAPSYSTEM = 26</td>
<td>a SAP system</td>
</tr>
</tbody>
</table>