BCT Enterprise Information Management

Corsa 7.2

Web Service Technical Description

Information creates.

Document versioning

Version	Date	Description
1.0.60	23-11-2018	Added AddFolderAction, AddFolderAction2 and DeleteFolderAction.
1.0.59	24-01-2018	Added element 'document name' (Docname) to output (GetFileVersionList.xml) of GetFileVersionList.
1.0.58	06-11-2017	Added ConnectAs
1.0.57	04-04-2017	Added CreateMetaAgendaItem and GenerateID

Table of contents

1. General information

The Corsa 7.2 Web Service is a FIPS-compliant Web Service based on the Microsoft .NET 4.0 Framework (as from update 2016-02 with use of an "ASP.NET v4.0" application pool and projects targetFramework="4.0".) and can be used to connect an application to Corsa.

This document describes the Corsa 7.2 Web Service which is an improved version of the Corsa 7.1 Web Service. When the Corsa 7.2 Web Service is installed the 'old' functionality of the Corsa 7.1 Web Service is still available, it is however highly recommended to upgrade the Corsa Web Service consumer to use the Corsa 7.2 Web Service.

Improvements of the Corsa 7.2 Web Service are:

- WS-I Basic Profile 1.1 Compliance: The Corsa 7.2 Web Service (Corsa72WS.wsdl and Corsa72WS4j.wsdl) is WS-I Basic Profile 1.1 Compliant.
- MTOM support: Corsa 7.2 Web Service supports MTOM (http://www.w3.org/TR/2005/REC-soap12-mtom-20050125/) based on Microsoft Web Services Enhancements 3.0.
- Most methods have a new output parameter LastError which makes the use of extra calls to the LastErrorMessage method redundant.

When an existing solution needs to be upgraded to use the Corsa 7.2 Web Service (Corsa72WS.wsdl or Corsa72WS4j.wsdl instead of Corsa71WS.wsdl) the following steps are necessary:

- Import the new wsdl and use the new available classes
- Change the code for the methods which now have a LastError output parameter to handle the new output parameter
- Change calls to QueryExecute, which now has an extra input parameter (the default value is an empty string)
- Verify the output of the methods GetConfidentialities, GetObjectKinds, QueryExecute, GetObjectRelationTypes, GetDocTodoList, GetCaseTodoList which has changed to comply with the WS-I Basic Profile 1.1

1.1 Structure

To use the Corsa Web Service, a fully configured Corsa environment should be installed. The Corsa Web Service uses standard Corsa functionality like Corsa/DS and the Corsa AppServer.

The Corsa Web Service is not a default component of a Corsa environment.

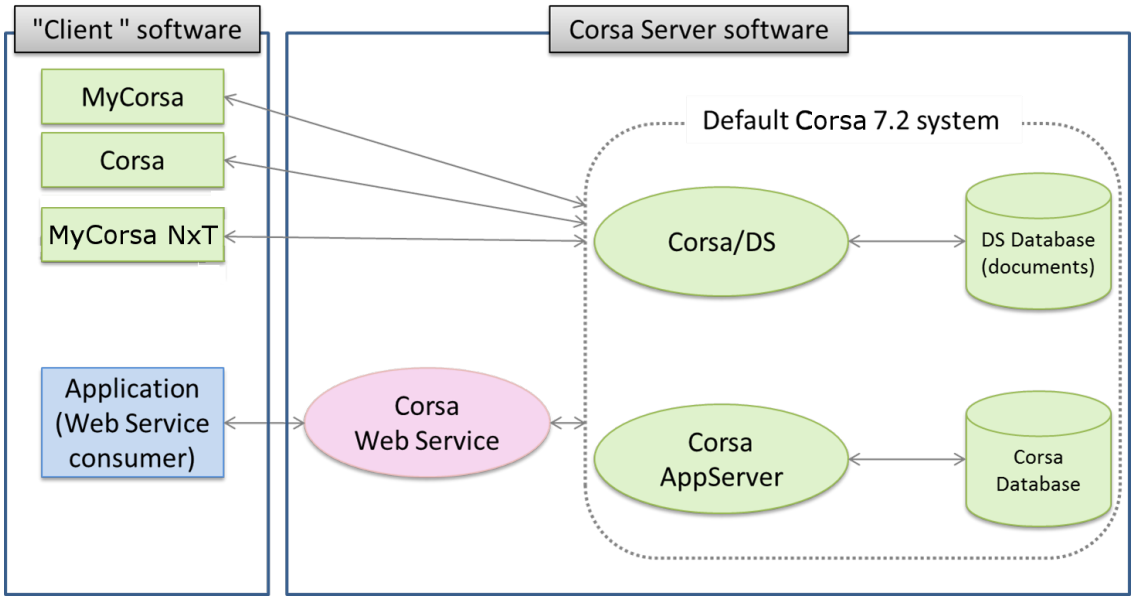


Figure 1: Structure of Corsa and the Corsa Web Service

1.2 .NET Framework

The Corsa Web Service is based on the Microsoft .NET 4.0 Framework. For communicating with the Corsa AppServer the Microsoft .NET 1.1 Framework is required.

1.3 SOAP Message Format

The Corsa Web Service can be accessed using SOAP messages. The SOAP message format for the Corsa Web Service is Document/Literal.

1.4 Sessions

To be able to use Corsa functionality correctly, the Corsa Web Service uses sessions for maintaining state between method calls. A consumer application therefore must support HTTP cookies.

1.5 Deployment and configuration

Information about deployment and configuration can be found in the Corsa Web Service installation documentation.

2. Consuming the Corsa Web Service

To create a consumer for the Corsa Web Service, the Web Service contract, the WSDL document, can be used to generate proxy classes. After installing and configuring the Corsa Web Service the WSDL is available by using the following URL (the first part of the URL's depends on the installation of the Corsa Web Service):

```
http://localhost/wsCorsa7/Corsa72WS.asmx?WSDL
```

2.1 Basic usage

To be able to use the Corsa Web Service methods, first make a call to the Connect method.

Without a successful call to the Connect method all other functions will fail!

Before using the Connect method the setLicenseKey method must be used to specify the license key.

If a method returns false or null, either the LastError output parameter or the LastErrorMessage method can be used to get detailed information about a possible error.

To end a session, call the Disconnect method.

Since the Corsa Web Service uses sessions, the consumer application needs to support cookies.

2.2 .NET consumer

When using a .NET application create a System.Net.CookieContainer and assign it to the CookieContainer property of the Web Service proxy object. For example:

```
CORSA72WS ws = new CORSA72WS();
if (ws != null)
{
       ws.CookieContainer = new System.Net.CookieContainer();
}
```

2.3 JAVA consumer

Since the Corsa72WS.wsdl contains some classes which are .NET specific some JAVA toolkits are unable to parse the wsdl file. Therefor a special wsdl file is provided for JAVA consumers: Corsa72WS4j.wsdl.

The WSDL is available by using the following URL (the first part of the URL's depends on the installation of the Corsa Web Service):

```
http://localhost/wsCorsa7/Corsa72WS4j.asmx?WSDL
```

2.3.1 Oracle JDeveloper

When using Oracle JDeveloper use setMaintainSession(true) with the generated proxy object to maintain session state.

2.3.2 Apache Axis2

When using Apache Axis2 use the following settings to maintain session state:

```
//Get initialized Endpoint
Options options = getCorsaConnector()._getServiceClient().getOptions();

//Disable "Chunked" when communicating with.NET server
options.setProperty(HTTPConstants.CHUNKED, Boolean.FALSE);

//Supress "HttpMethodBase [INFO] Discarding unexpected response: HTTP/1.1 100
// Continue"
options.setProperty(HTTPConstants.HTTP_PROTOCOL_VERSION,
  HTTPConstants.HEADER_PROTOCOL_10);

//Maintain server sessions
options.setProperty(HTTPConstants.REUSE_HTTP_CLIENT, true);
options.setCallTransportCleanup(true);

//Start session
getCorsaConnector().startSession();
```

2.3.3 Apache CXF (JAXB)

When using Apache CXF (JAXB) use the following settings to maintain session state:

```
// Maintain session state between calls which is needed since we call a
functional connect

((BindingProvider)
stub).getRequestContext().put(BindingProvider.SESSION_MAINTAIN_PROPERTY,Boolean.
TRUE);
```

2.4 Delphi Win32 consumer

To be able to use the Corsa Web Service from a Delphi Win32 application created with a Delphi version earlier than Delphi 2006, the SOAP interface unit, generated by importing the WSDL, needs to be modified manually. The WSDL importer of older Delphi versions does not recognize the binding method from the ASP.NET 2.0 WSDL file correctly. This is due to the fact that Microsoft changed some of the ways WSDL was published compared to .NET 1.1.

Since the WSDL importer did not recognize the Document/Literal style, Delphi will use RPC instead (Delphi's default). This will result in losing values for parameters of method calls to the Corsa Web Service. The solution for this is to manually specify the ioDocument as InvokeOptions for the SOAP interface type, as follows:

```
InvRegistry.RegisterInvokeOptions(TypeInfo(xxx), ioDocument);
```

Where xxx is the name of your SOAP interface type. This line of code needs to be added to the initialization section of the generated Win32 import unit.

See also
http://www.bobswart.nl/weblog/Blog.aspx?RootId=5:798

2.5 MTOM (Message Transmission Optimization Mechanism)

The Corsa Web Service supports MTOM (http://www.w3.org/TR/2005/REC-soap12-mtom-20050125/). When a consumer also supports MTOM this will increase performance when sending binary data to and from the Corsa Web Service.

MTOM support is added by using Microsoft's Web Services Enhancements 3.0 (WSE 3.0) (http://msdn.microsoft.com/en-us/library/aa139619.aspx).

MTOM Usage in .NET consumer:

- Make sure that WSE 3.0 is installed in Visual Studio (http://www.microsoft.com/downloads/details.aspx?familyid=018a09fd-3a74-43c5-8ec1-8d789091255d&displaylang=en)
- Enable the project for Web Service Enhacements (right click the poject and choose "WSE Settings 3.0" and enable the checkbox 'Enable this project for Web Services Enhancements')
- Add a Web Reference to Corsa72WS.wsdl

After adding the Web Reference two proxy classes are generated:

1. CORSA72WS: this is the proxy class that does not use WSE 3.0. It inherits from the System.Web.Services.Protocols.SoapHttpClientProtocol class.
2. CORSA72WSWse: this is the proxy class that makes use of WSE 3.0. It inherits from the Microsoft.Web.Services3.WebServicesClientProtocol class.

To use MTOM the CORSA72WSWse class needs to be used. This class has a property called "RequireMtom" which should be set to true to enable MTOM:

```
CORSA72WSWse ws = new CORSA72WSWse();
ws.CookieContainer = new System.Net.CookieContainer();
ws.RequireMtom = true;
```

3. Methods

This section describes the methods and their parameters which are available in the Corsa Web Service. These methods are available in the Corsa72WS.wsdl.

3.1 About

Returns information about the Corsa 7.2 Web Service.

Parameters:

None

Output:

Name	Type	Description
Result	String	Information about the Corsa Web Service

3.2 AddFolderAction

Adds a specific action to a specific folder-object (registration) or modifies an existing action.

Input parameters

Name	Type	Description
FolderID	String	Corsa ObjectID of the folder.
ActionID	String	ActionID of the action to add to the specific folder.
Date	String	Optional parameter. Parameter “Date” or “PeriodID” (one of both) should be provided. ..
Period	String	Optional parameter. Parameter “PeriodID” or “Date” (one of both) should be provided. ..

Output parameters

Name	Type	Description
• Result	Boolean	True when successful, otherwise false.
• LastError	String	Error information in case of unsuccessful execution.

3.3 AddFolderAction2

Adds a specific action to a specific folder-object (registration) or modifies an existing action.

Input parameters

Name	Type	Description
FolderID	String	Corsa ObjectID of the folder.
ActionID	String	ActionID of the action to add to the specific folder.
Date	String	Optional parameter. Parameter “Date” or “PeriodID” (one of both) should be provided. ..
PeriodID	String	Optional parameter. Parameter “PeriodID” or “Date” (one of both) should be provided. ..
ReminderDate	String	Optional parameter. ..
ActionDescription	String	Optional parameter. Alternative description for the action.
Note	String	Optional parameter.
CompletionDate	String	Optional parameter. ..

Output parameters

Name	Type	Description
• Result	Boolean	True when successful, otherwise false.
• LastError	String	Error information in case of unsuccessful execution.

3.4 AddPermit

The AddPermit method adds a permit or modifies an existing permit of an object (document, case, person, organisation, etc.). If no permit for a user exists yet, a permit is created, otherwise the existing permit of the user is modified.

Parameters:

Name	Type	Description
ObjectType	String	Object type

ObjectID	String	Object ID
UserID	String	Corsa User ID
Modify	Boolean	User is allowed to modify data
DateTo	String	Permit is valid until this date, format is "DD/MM/YYYY"

Output:

Name	Type	Description
Result	Boolean	True when successful, otherwise false
LastError	String	Error information in case of unsuccessful execution

3.5 AddRECity

Adds a new Real Estate city to table Cities [VPL] .

Parameters:

Name	Type	Description
City	String	Name of the real estate city in Corsa

Output:

Name	Type	Description
Result	Boolean	True when successful, otherwise false
LastError	String	Error information in case of unsuccessful execution

As from update 2016-05 this methods returns True when the city being added already exists (instead of an error message "City already exists.").

3.6 AddREStreet

Adds a new Real Estate street to table Streets [VSTR] for an existing Real Estate city in table Cities [VPL] .

Parameters:

Name	Type	Description
City	String	Name of the real estate city in Corsa
Street	String	Name of the real estate street

Output:

Name	Type	Description
Result	Boolean	True when successful, otherwise false
LastError	String	Error information in case of unsuccessful execution

As from update 2016-05 this methods returns True when the street being added already exists (instead of an error message "Street already exists.").

3.7 AddSenderAddr

The AddSenderAddr method can be used to add a Sender/Addr. to a document registration.

Parameters:

Name	Type	Description
ObjectID	String	ID of a document registration
AfzAanRelType	String	Contact type for the Sender/Addr
AfzAanRelID	String	Contact id for the Sender/Addr
AfzAanExtRelID	String	External Contact Id (only valid for AfzAanRelType = 'K')
RelationType	String	Kind of relations

Output:

Name	Type	Description
Result	Boolean	True when successful, otherwise false
LastError	String	Error information in case of unsuccessful execution

3.8 AppendChunk

Append a chunk of bytes to a file. The client should ensure that all messages are sent in sequence. This method always overwrites any existing file with the same name.

Parameters:

Name	Type	Description
FileName	String	The name of the file that this chunk belongs to
Buffer	Byte array	The byte array, i.e. the chunk being transferred
Offset	Long	The offset at which to write the buffer to

Improvement: As from update 2016-06 a session check is executed for this method.

See also
AppendChunk2

3.9 AppendChunk2

Append a chunk of bytes to a file. The client should ensure that all messages are sent in sequence. This method always overwrites any existing file with the same name.

Parameters:

Name	Type	Description
FileName	String	The name of the file that this chunk belongs to
Buffer	Byte array	The byte array, i.e. the chunk being transferred
Offset	Long	The offset at which to write the buffer to

Output:

Name	Type	Description
Result	Boolean	True when successful, otherwise false
LastError	String	Error information in case of unsuccessful execution

Method **AppendChunk2** has the same functionality as **AppendChunk**. However AppendChunk2 returns a boolean Result value and a LastError string (as from update 2013-wk51).

Improvement: As from update 2016-06 a session check is executed for this method.

3.10 CaseAdhocAction

This method is related to Corsa/Case (workflow). Please contact BCT for more information regarding the use of this method.

3.11 CaseCalcInitialValues

This method is related to Corsa/Case (workflow). Please contact BCT for more information regarding the use of this method.

3.12 CaseChangeDocCustomer

Returns the contact from an object to another object.

The CaseChangeDocCustomer method changes the CASE customer from a document.

Parameters:

Name	Type	Description
DocumentId	String	The documentId you want to change the customer
RelSoortId	String	The id of the relation kind
CaseCustomerType	String	The ObjectType of the customer
CaseCustomerId	String	The ObjectId of the customer

Output:

Name	Type	Description
Result	Boolean	True when successful, otherwise false
LastError	String	Error information in case of unsuccessful execution

3.13 CaseCompleteStep

This method is related to Corsa/Case (workflow). Please contact BCT for more information regarding the use of this method.

3.14 CaseConnectDocumentType

From update 2014-03 on this method is generally available (originally only custom-made)

CaseConnectDocumentType connects a DocumentType to a document

```
public bool CaseConnectDocumentType(string DocumentID, string DocTypeID, string
  CaseID, string ObjectType, string ObjectID, bool ProcessSysSteps, out string
  StartedCaseID, out String LastError)
```

Parameters:

Name	Type	Default	Description
DocumentID	string		DocumentID that has to be linked to the DocumentType
DocType	string		DocumentTypeID
CaseID	string	“”	(Optional) If the DocumentID is linked to the CaseID
ObjectType	string	“”	(Optional) Customer's Object type (E=Organization, P=Person)
ObjectID	string	“”	(Optional) Customer's ObjectID If ObjectType + ObjectID are valid, the current customer of the case will be overwritten.
ProcessSysSteps	boolean		If True the system steps will be executed immediately, otherwise they won't

Output:

Name	Type	Default	Description
StartedCaseID	string		This output parameter is filled with the ID when a CaseId was created or connected. In some cases StartedCaseID is empty because an user interface has to be displayed on which is decided what case or procedure has to be created/started. If the CaseID is empty this doesn’t mean that the API function has failed. The case will be displayed in the user's to-do-list.
LastError	String	“”	Error information if the result is False.
Result	Boolean	False	True if successful else false

3.15 CaseEvaluate

The **CASEEvaluate** function can be used to evaluate a specific CASE. If there is one or more ‘system step’ then the Corsa/Case engine will try to complete that step.

Parameters:

Name	Type	Description
CaseID	String	CaseID of a CASE to “evaluate”

Output:

Name	Type	Description

Result	Boolean	True when successful, otherwise false
LastError	String	Error information in case of unsuccessful execution

3.16 CaseGetDocumentTypes

Method CaseGetDocumentTypes can be used to retrieve a list of available Corsa/Case Document Types.

Parameters:

Name	Type	Description
StartersOnly	Boolean	Retrieve only Document Types which are CASE starters when TRUE

Output:

Name	Type	Description
Result	Boolean	True when successful, otherwise false
DocumentTypes	DataTable	Null when failed, otherwise a DataTable containing a table with the Document Types
LastError	String	Error information in case of unsuccessful execution

Corsa72WS4j: the result DataTable will be returned as a byte array containing an XML representation of the DataTable.

3.17 CaseGetInfo

The CaseGetInfo method retrieves information about a specific Case.
(As from update 2013-wk39)

Parameters:

Name	Type	Description
CaseID	String	Case ID

Output:

Name	Type	Description
Result	Boolean	True when successful, otherwise false
CaseInfo	NameValue[]	Case details
LastError	String	Error information in case of unsuccessful execution

3.18 CaseGetProcessList

Method CaseGetProcessList can be used to retrieve a list of available Corsa/Case processes.

Parameters:

None

Output:

Name	Type	Description
Result	Boolean	True when successful, otherwise false
ProcessList	DataTable	Null when failed, otherwise a DataTable containing a table with the processes
LastError	String	Error information in case of unsuccessful execution

Corsa72WS4j: the result DataTable will be returned as a byte array containing an XML representation of the DataTable.

3.19 CaseGetQueue

This method is related to Corsa/Case (workflow). Please contact BCT for more information regarding the use of this method.

3.20 CaseGetQueueData

This method is related to Corsa/Case (workflow). Please contact BCT for more information regarding the use of this method.

3.21 CaseGetReferences

The CaseGetReferences method can be used to retrieve a list of references that are used with a specific Case.

Parameters:

Name	Type	Description
CaseID	String	Case ID

Output:

Name	Type	Description
Result	Boolean	True when successful, otherwise false

References	DataTable	Null when failed, otherwise a DataTable containing a table with the Case references
LastError	String	Error information in case of unsuccessful execution

Corsa72WS4j: the result DataTable will be returned as a byte array containing an XML representation of the DataTable.

3.22 CaseGetStepInfo

The CaseGetStepInfo method retrieves information about a specific Case step.
(As from update 2013-wk39)

Parameters:

Name	Type	Description
CaseID	String	Case ID
StepID	String	Case step ID

Output:

Name	Type	Description
Result	Boolean	True when successful, otherwise false
StepInfo	NameValue[]	Case step details
LastError	String	Error information in case of unsuccessful execution

3.23 CaseStartPrcd

Method CaseStartPrcd can be used to start a Corsa/Case procedure.

Parameters:

Name	Type	Description
ProcedureID	String	Procedure ID
ProcedureVersion	String	Procedure version
ObjectType	String	Object type
ObjectID	String	Object ID

Output:

Name	Type	Description
CaseID	String	The started Case ID

StepID	Integer	The active Step ID for the started Case
Result	Boolean	True when successful, otherwise false
LastError	String	Error information in case of unsuccessful execution

3.24 CaseSubmitQueueData

This method is related to Corsa/Case (workflow). Please contact BCT for more information regarding the use of this method.

3.25 CaseWFVFlowCase

This method is related to Corsa/Case (workflow). Please contact BCT for more information regarding the use of this method.

3.26 CaseWFVFlowPrcd

This method is related to Corsa/Case (workflow). Please contact BCT for more information regarding the use of this method.

3.27 ChangeObjectKind

With the ChangeObjectKind method it is possible to change the object kind from an object.

Parameters:

Name	Type	Description
ObjectType	String	Object type from the object you want to change.
ObjectID	String	ObjectID from the object you want to change.
NewObjectKindID	String	The objectkind id you want to change to.
Options	NameValue[]	There are three options for changing the objectkind of an object: Name: ModifyObjectId Value=bool If the value is true you allow that the Object ID changes. False if it shouldn't be possible. Name: DeleteCurrentConf Value=bool If the value is true the confidentiality is deleted when the objectkind change is successful. If the value is false it won't be deleted. Name: LinkNewConf Value=bool If the value is true the confidentialities are added from the new objectkind when the objectkind change is successful. If the value is false they won't be added.

Output:

Name	Type	Description
NewObjectID	String	Returns the same or a new objectId
LastError	String	Error information in case of unsuccessful execution
Result	Boolean	True when successful, otherwise false

3.28 CheckFileHash

As from update 2016-02 a FIPS-compliant version of the **CheckFileHash** method is available in **CheckFileHashSHA256** and is preferred over **CheckFileHash**.
Be aware that the **CheckFileHash** method returns a MD5 hash, whereas the **CheckFileHashSHA256** method returns a SHA-256 hash.
Consumers switching to **CheckFileHashSHA256** should of course also change the locally computed hash to match the hash of the server.
The **CheckFileHash** method is still supported for backward compatibility but will not work in set-ups that have FIPS enabled.

The **CheckFileHash** method is used after an upload (AppendChunk) or download (DownloadChunk) to calculate and compare a hash.

To enable the processing of huge files you can use the **AppendChunk** and **DownloadChunk** methods. By using these methods a file is not uploaded/downloaded as a whole but in chunks (small portions). After sending all chunks it can be verified whether the file was sent/received properly by calculating a hash for that file. The returned hash by **CheckFileHash** from the Webservice then can be compared by the hash that was calculated locally. Only if both hashes match, the file transfer was successful.

Parameters:

Name	Type	Description
FileName	String	The name of the file on the server

Output:

Name	Type	Description
Result	String	MD5 hash string when successful, otherwise an empty string

3.29 CheckFileHashSHA256

As from update 2016-02 the **CheckFileHashSHA256** method is used after an upload (AppendChunk) or download (DownloadChunk) to calculate and compare a hash. **CheckFileHashSHA256** is a FIPS-compliant version of the CheckFileHash method.

To enable the processing of huge files you can use the **AppendChunk** and **DownloadChunk** methods. By using these methods a file is not uploaded/downloaded as a whole but in chunks (small portions). After sending all chunks it can be verified whether the file was sent/received properly by calculating a hash for that file. The returned hash by **CheckFileHashSHA256** from the Webservice then can be compared by the hash that was calculated locally. Only if both hashes match, the file transfer was successful.

Parameters:

Name	Type	Description
FileName	String	The name of the file on the server

Output:

Name	Type	Description
Result	String	SHA-256 hash string when successful, otherwise an empty string

3.30 Connect

Start a session and connect the Corsa Web Service to a specific Corsa database.

To start a session and connect to the desired Corsa database the Connect method must be used.

Without a successful call to the Connect method all other functions will fail!

Before using the Connect method the setLicenseKey method must be used to specify the license key.

Note: Before using the Connect the consumer must be enabled to maintain the session state as described in "Basic Usage"

Parameters:

Name	Type	Description
ApplicationID	String	Identifier for the application calling the CORSA71WS
ConnectionID	String	Identifier for connection to use. The connection must be defined in the Web.config file. See Deployment and configuration for details
UserName	String	Corsa username to connect with
Password	String	Corsa password
EncodedPass	Boolean	Indicates whether the password is Progress encoded password (default false)

Output:

Name	Type	Description
Result	Boolean	True when successful, otherwise false
LastError	String	Error information in case of unsuccessful execution

Use the following steps to use the Corsa auto login mechanism (login with username only):

Application manager

- Start Corsa WebService consumer application
- Type a new password which you want to use for auto login in the 'New Autologin Pswrd' field
- Press button 'Encrypt'
- Copy/paste new encrypted password in AppSettings section of web.config file

End user

- Specify a valid Corsa network user name as value for the UserName parameter
- Specify a valid auto login password as value for the Password parameter. Default is 'CORSAWSAUTOLOGIN'
- Make sure that in Corsa parameter 'fastws' is set to 'yes'

See also
Basic Usage
SetLicenseKey
Disconnect

3.31 ConnectAs

Start a session and connect the Corsa Web Service to a specific Corsa database, but be finally connected as another user.

To start a session and connect as another user to the desired Corsa database the **ConnectAs** method must be used.

Without a successful call to the ConnectAs method all other functions will fail!

Before using the ConnectAs method the setLicenseKey method must be used to specify the license key.

Note: Before using the ConnectAs the consumer must be enabled to maintain the session state as described in "Basic Usage"

Required: Corsa parameter "LoginAsUser", which contains a comma separated list of usernames.

Parameters:

Name	Type	Description
ApplicationID	String	Identifier for the application calling the CORSA71WS
ConnectionID	String	Identifier for connection to use. The connection must be defined in the Web.config file. See Deployment and configuration for details
UserName	String	Corsa username to connect with
Password	String	Corsa password
EncodedPass	Boolean	Indicates whether the password is Progress encoded password (default false)
LoginAsUser	String	Corsa username to be finally connected as

Output:

Name	Type	Description
Result	Boolean	True when successful, otherwise false
LastError	String	Error information in case of unsuccessful execution

See also
Basic Usage
SetLicenseKey
Disconnect

3.32 Connected

Check if session is connected.

Parameters:

None

Output:

Name	Type	Description
Result	Boolean	True when successful, otherwise false

See also
Basic Usage
Connect
Disconnect

3.33 CreateFileVersion

Creates a new or updates an existing file version of a specific (existing) object.

The CreateFileVersion method stores a local file as a new version in Corsa.

Parameters:

Name	Type	Description
ObjectType	String	Corsa ObjectType of the object for the file version
ObjectID	String	Corsa ObjectID of the object for the file version
FileBytes	Byte Array	Byte Array of the file data
FileType	DSFileType	The DS file type, either Native, Archive or OCRtext
FileVersion	Integer	Version for the file. To create a new version, specify a value < 1. To overwrite an existing version, specify the existing version number. When using a FileType other than Native, an existing version must be specified
FileExtension	String	Extension of the file, including a preceding “.”
DocumentName	String	Name of the document. DocumentName is the same for all versions

VersionDescription	String	Description of the version.
Action	DSAction Array	Action(s) to perform on the file. Default null (let Corsa decide what actions to perform).

Output:

Name	Type	Description
Result	Boolean	True when successful, otherwise false
VersionInfo	VersionInfo	VersionInfo class containing information about the file version
LastError	String	Error information in case of unsuccessful execution

Improvement: As from update 2018-01 the document name is automatically filled with the document id. if parameter DocumentName is empty.
*CRS-3615

Improvement: As from update 2017-03 the issue is solved that for FileType 'Archive' and FileVersion '1' the Result was 'True' even if there was no document version available yet. In that case the archive- or text file was written unintentionally to the Document Server.
*CRS-1482

Improvement: As from update 2016-06 the issue is solved that resulted in a wrong extension when changing an existing version and not passing a file extension (the old extension was preserved in the Corsa database while the file on Corsa DS was without extension).
*HD.042004
Improvement: As from update 2013-wk51 option PDF/OCR is now checked in Corsa when file type archive/OCRtext is passed. And from update 2014-04: When passing an archive file (FileType = DSFileType.ftArchive), no actions on the original file are performed anymore.

3.34 CreateFileVersion2

Creates a new or updates an existing file version of specific (existing) object. The file must first be transferred to the server.

The CreateFileVersion2 method stores a local file as a new version in Corsa.

Parameters:

Name	Type	Description
ObjectType	String	Corsa ObjectType of the object for the file version
ObjectID	String	Corsa ObjectID of the object for the file version
FileName	String	Name of the file on the server
FileType	DSFileType	The DS file type, either Native, Archive or OCRtext

FileVersion	Integer	Version for the file. To create a new version, specify a value < 1. To overwrite an existing version, specify the existing version number. When using a FileType other than Native, an existing version must be specified
FileExtension	String	Extension of the file, including a preceding “.”
DocumentName	String	Name of the document. DocumentName is the same for all versions
VersionDescription	String	Description of the version.
Action	DSAction Array	Action(s) to perform on the file. Default null (let Corsa decide what actions to perform).

Output:

Name	Type	Description
Result	Boolean	True when successful, otherwise false
VersionInfo	VersionInfo	VersionInfo class containing information about the file version
LastError	String	Error information in case of unsuccessful execution

Improvement: As from update 2018-01 the document name is automatically filled with the document id. if parameter DocumentName is empty.
*CRS-3615

Improvement: As from update 2017-03 the issue is solved that for FileType 'Archive' and FileVersion '1' the Result was 'True' even if there was no document version available yet. In that case the archive- or text file was written unintentionally to the Document Server.
*CRS-1482

Improvement: As from update 2016-06 the issue is solved that resulted in a wrong extension when changing an existing version and not passing a file extension (the old extension was preserved in the Corsa database while the file on Corsa DS was without extension).
*HD.042004
Improvement: As from update 2013-wk51 option PDF/OCR is now checked in Corsa when file type archive/OCRtext is passed. And from update 2014-04: When passing an archive file (FileType = DSFileType.ftArchive), no actions on the original file are performed anymore.

3.35 CreateMetaAgendaItem

Creates a new or updates an existing Agenda Item registration.

For parameter FieldValues both fields bi-agpunt.agenda_dat and bi-agpunt.orgaan_id are mandatory.

Parameters:

Name	Type	Description
ObjectID	String	Corsa ObjectID for the registration. If the ObjectID contains a valid ObjectID then that registration will be updated. Default “”
ObjectKind	String	Corsa object kind to use for the registration. See GetObjectKinds for information about retrieving a list of object kinds
FieldValues	NameValue array	Array of NameValue items with fields and their values. Mandatory NameValue items are: • bi-agpunt.agenda_dat (Agenda date) • bi-agpunt.orgaan_id (Organ Id.)
ReferenceValues	NameValue array	Array of NameValue items with references and their values.
CheckValues	Boolean	Determines what happens if an invalid value was specified. In this context ‘value’ refers to the Value property of a NameValue object that was passed to FieldValues or ReferenceValues. If true, the method will return an error if an invalid value was specified. The registration will not be created or updated. If false, the method will not return an error if an invalid value was specified. The registration will be created or updated.

Output:

Name	Type	Description
Result	Boolean	True when successful, otherwise false
NewObjectID	String	Corsa ObjectID for the created AgendaItem registration
LastError	String	Error informative in case of unsuccessful execution

3.36 CreateMetaDocument

Creates a new or updates an existing Document (meta-) registration.

With the CreateMetaDocument method a Corsa document registration can be created or updated with specific values and references.

As from update 08/02/2013 an empty string is allowed for mandatory parameter ObjectKind when modifying a document (parameter ObjectID is not empty).
*HD 034983
As from update 2014-12 the issue is solved whereby a SOAP fault occurred (System.ArgumentNullException) when passing a null value to FieldValues.
*HD 040774

As from update 2015-2 **CreateMetaDocument** no longer returns a question ("Do you want to link all linked documents of document '@' with folder '@' as well?") when setting document field **Folder id**. [stuk_doss.dossier_id] while parameter **ststds** is set to '1'. No linked documents will be linked to the folder.
*HD 040734

Parameters:

Name	Type	Description
ObjectID	String	Corsa ObjectID for the registration. If the ObjectID contains a valid ObjectID then that registration will be updated. Default “”
ObjectKind	String	Corsa object kind to use for the registration. See GetObjectKinds for information about retrieving a list of object kinds
FieldValues	NameValue Array	Array of NameValue items with fields and their values
ReferenceValues	NameValue Array	Array of NameValue items with references and their values
DuplicateID	String	Corsa ObjectID to use to copy registration data from
DSPTemplateID	String	Corsa DSP-Subject TemplateID to use as registration template

For a list of possible values for the FieldValues parameter select menu option 'System | Installation | Programs' in Corsa and execute ontw/d-dspfields.r Select Object kind 'Document' and Type 'Registration window fields' in window 'Fields per type'.

For a list of possible values for the ReferenceValues parameter use Corsa and go to 'System | Settings | References'. Only the fields for object type Document can be used.

For references that are defined with 

, you must pass multiple values in a single NameValue, with values separated by chr(10): value 1 + chr(10) + value 2 + chr(10) value n.
When updating an existing object, existing values in Corsa that are not passed in NameValue are deleted.
For example, if a reference of an object in Corsa has values "A" and "B" and in NameValue "B" + chr(10) + "D" is passed, the result in Corsa will be that the reference has values "B" and "D" (value "A" is deleted).

Output:

Name	Type	Description
Result	Boolean	True when successful, otherwise false
NewObjectID	String	Corsa ObjectID for the created registration
LastError	String	Error information in case of unsuccessful execution

 A call to CreateMetaDocument is the same as a call to CreateMetaDocument2 with parameters WhoObjectType = "" and WhoSearchValue = "".
See also

CreateMetaDocument2
CreateMetaDocument3
CreateMetaDocument4

3.37 CreateMetaDocument2

Creates a new or updates an existing Document (meta-) registration.

With the CreateMetaDocument2 method a Corsa document registration can be created or updated with specific values and references.

As from update 08/02/2013 an empty string is allowed for mandatory parameter ObjectKind when modifying a document (parameter ObjectID is not empty).
* HD.034983
As from update 2014-12 the issue is solved whereby a SOAP fault occurred (System.ArgumentNullException) when passing a null value to FieldValues.
* HD.040774
As from update 2015-2 **CreateMetaDocument** no longer returns a question ("Do you want to link all linked documents of document '@' with folder '@' as well?") when setting document field **Folder id**. [stuk_doss.dossier_id] while parameter **ststds** is set to '1'. No linked documents will be linked to the folder.
* HD.040734
Parameters:

Name	Type	Description
ObjectID	String	Corsa ObjectID for the registration. If the ObjectID contains a valid ObjectID then that registration will be updated. Default ''
ObjectKind	String	Corsa object kind to use for the registration. See GetObjectKinds for information about retrieving a list of object kinds
FieldValues	NameValue Array	Array of NameValue items with fields and their values
ReferenceValues	NameValue Array	Array of NameValue items with references and their values
DuplicateID	String	Corsa ObjectID to use to copy registration data from

DSPTemplateID	String	Corsa DSP-Subject TemplateID to use as registration template
WhoObjectType	String	Corsa object type of the Who? contact ('E' for organisation, 'P' for person). Default ''
WhoSearchValue	String	The (unique) value to find the Who? contact with. Default ''

For a list of possible values for the FieldValues parameter select menu option 'System | Installation | Programs' in Corsa and execute ontw/d-dspfields.r Select Object kind 'Document' and Type 'Registration window fields' in window 'Fields per type'.

For a list of possible values for the ReferenceValues parameter use Corsa and go to 'System | Settings | References'. Only the fields for object type Document can be used.

For references that are defined with 

, you must pass multiple values in a single NameValue, with values separated by chr(10): value 1 + chr(10) + value 2 + chr(10) value n.
When updating an existing object, existing values in Corsa that are not passed in NameValue are deleted.
For example, if a reference of an object in Corsa has values "A" and "B" and in NameValue "B" + chr(10) + "D" is passed, the result in Corsa will be that the reference has values "B" and "D" (value "A" is deleted).

When WhoObjectType and WhoSearchValue are specified, a WHO contact will be searched/created and used as the sender for the Document registration. The WHO contact will also be used as the case customer if a case document type ID (ws-ps_ds.dsrt_id) is specified in FieldValues or when a DSP-Subject TemplateID is specified in DSPTemplateID.

The reference specified in a parameter ('zkkme' when WhoObjectType is 'E' and 'zkkmp' when WhoObjectType is 'P') is used to find the WHO contact. The reference of the contact must match the value in WhoSearchValue.

- If one contact was found then this is used as the sender and/or case customer.
- If multiple contacts were found, the Result will be false and no create/update of a Document registration will take place.
- If, however, no contact is found, a (broader) general query using WhoSearchValue is performed automatically to find the contact.
  In that case
    - If one contact was found, then this is used as the sender and/or case customer.
    - If multiple contacts were found, a new contact is created and will be used as the sender and/or case customer. This contact is created with the default organisation/person type that is specified in parameter 'defsrte' (for organisations) or 'defsrtp' (for persons).

If FieldValues already contains a sender the specified value will not be overwritten. The same applies to case customer.

Output:

Name	Type	Description
Result	Boolean	True when successful, otherwise false
NewObjectID	String	Corsa ObjectID for the created registration
LastError	String	Error information in case of unsuccessful execution

 A call to CreateMetaDocument2 is the same as a call to CreateMetaDocument3 with parameters WhoFieldValues = null and WhoReferenceValues = null.

See also

CreateMetaDocument3
CreateMetaDocument4

3.38 CreateMetaDocument3

Creates a new or updates an existing Document (meta-) registration.

With the CreateMetaDocument3 method a Corsa document registration can be created or updated with specific values and references.

As from update 08/02/2013 an empty string is allowed for mandatory parameter ObjectKind when modifying a document (parameter ObjectID is not empty).
* HD.034983
As from update 2014-12 the issue is solved whereby a SOAP fault occurred (System.ArgumentNullException) when passing a null value to FieldValues.
* HD.040774
As from update 2015-2 **CreateMetaDocument** no longer returns a question ("Do you want to link all linked documents of document '@' with folder '@' as well?") when setting document field **Folder id**. [stuk_doss.dossier_id] while parameter **ststds** is set to '1'. No linked documents will be linked to the folder.
* HD.040734
Parameters:

Name	Type	Description
ObjectID	String	Corsa ObjectID for the registration. If the ObjectID contains a valid ObjectID then that registration will be updated. Default ''
ObjectKind	String	Corsa object kind to use for the registration. See GetObjectKinds for information about retrieving a list of object kinds
FieldValues	NameValue Array	Array of NameValue items with fields and their values
ReferenceValues	NameValue Array	Array of NameValue items with references and their values
DuplicateID	String	Corsa ObjectID to use to copy registration data from

DSPTemplateID	String	Corsa DSP-Subject TemplateID to use as registration template
WhoObjectType	String	Corsa object type of the WHO contact ('E' for organisation, 'P' for person). Default ''
WhoSearchValue	String	The (unique) value to find the WHO contact with. Default ''
WhoFieldValues	NameValue Array	Array of CORSA/WHO? NameValue items with fields and their values
WhoReferenceValues	NameValue Array	Array of CORSA/WHO?NameValue items with references and their values

For a list of possible values for the FieldValues parameter select menu option 'System | Installation | Programs' in Corsa and execute ontw/d-dspfields.r Select Object kind 'Document' and Type 'Registration window fields' in window 'Fields per type'.

For WhoFieldValues, only the fields for Object type 'Person' (when WhoObjectType is 'P') or Object type 'Organisation' (when WhoObjectType is 'E') can be used.

For a list of possible values for the ReferenceValues and WhoReferenceValues parameters use Corsa and go to 'System | Settings | References'. For ReferenceValues, only the fields for object type Document can be used. For WhoReferenceValues, only the fields for object type Person (when WhoObjectType is 'P') or Organisation (when WhoObjectType is 'E') can be used.

For references that are defined with 

, you must pass multiple values in a single NameValue, with values separated by chr(10): value 1 + chr(10) + value 2 + chr(10) value n.
When updating an existing object, existing values in Corsa that are not passed in NameValue are deleted.
For example, if a reference of an object in Corsa has values "A" and "B" and in NameValue "B" + chr(10) + "D" is passed, the result in Corsa will be that the reference has values "B" and "D" (value "A" is deleted).

When WhoObjectType and WhoSearchValue are specified, a WHO contact will be searched/created and used as the sender for the Document registration. The WHO contact will also be used as the case customer if a case document type ID (ws-ps_ds.dsrt_id) is specified in FieldValues or when a DSP-Subject TemplateID is specified in DSPTemplateID. When a WHO contact is created, additional values (like address data) for the registration can be specified in WhoFieldValues and WhoReferenceValues.

The reference specified in a parameter ('zkkme' when WhoObjectType is 'E' and 'zkkmp' when WhoObjectType is 'P') is used to find the WHO contact. The reference of the contact must match the value in WhoSearchValue. Note that confidentialities are taken into account when searching WHO contacts.

- If one contact was found then this is used as the sender and/or case customer.
- If multiple contacts were found, the Result will be false and no create/update of a Document registration will take place.

- If, however, no contact is found, a (broader) general query using WhoSearchValue is performed automatically to find the contact.
  In that case
    - If one contact was found, then this contact is used as the sender and/or case customer.
    - If multiple contacts were found, a new contact is created and will be used as the sender and/or case customer. This contact is created with the default organisation/person type that is specified in parameter 'defsrte' (for organisations) or 'defsrtp' (for persons).

If FieldValues already contains a sender the specified value will not be overwritten. The same applies to case customer.

Output:

Name	Type	Description
Result	Boolean	True when successful, otherwise false
NewObjectID	String	Corsa ObjectID for the created registration
LastError	String	Error information in case of unsuccessful execution

A call to CreateMetaDocument3 is the same as a call to CreateMetaDocument4 with parameter CheckValues = false.

See also
CreateMetaDocument4

3.39 CreateMetaDocument4

Creates a new or updates an existing Document (meta-) registration.

With the CreateMetaDocument4 method a Corsa document registration can be created or updated with specific values and references.

As from update 2014-12 the issue is solved whereby a SOAP fault occurred (System.ArgumentNullException) when passing a null value to FieldValues.
*HD 040774
As from update 2015-02 **CreateMetaDocument** no longer returns a question ("Do you want to link all linked documents of document '@' with folder '@' as well?") when setting document field **Folder id**. [stuk_doss.dossier_id] while parameter **ststds** is set to '1'. No linked documents will be linked to the folder.
*HD 040734
As from update 2015-04 the check of the entered **Characteristic formats** and **Corsa fields** is expanded.
To create an object registration using the Webservice as with method **CreateMetaDocument** the check of **Characteristic formats** and **Corsa fields** is expanded for simple formats such as "X (10)" and "-..> >>> >> 9.99999" (not for input masks like "9 (2) - (3) -9" and "9999 XX").

For **CheckValues** = true the checks below are executed:

Data type	Check
Date	length (max. 10 digits) and validity
Character	length
Logical	for references: 1, 0, yes, no, true or false for fields : yes, no, true or false
Memo	length
Integer	number of digits, sign (negative allowed) and validity
Numerical	number of digits before and after the comma, sign (negative allowed) and validity

*HD 040405

As from update 2018-03 a **Sender id**. retrieved by BSN-number is also applied to field **Case Customer** of the Document registration.
(Previously field **Case Customer** contained the BSN-number)

Parameters:

Name	Type	Description
ObjectID	String	Corsa ObjectID for the registration. If the ObjectID contains a valid ObjectID then that registration will be updated. Default ''
ObjectKind	String	Corsa object kind to use for the registration. See GetObjectKinds for information about retrieving a list of object kinds
FieldValues	NameValue Array	Array of NameValue items with fields and their values
ReferenceValues	NameValue Array	Array of NameValue items with references and their values
DuplicateID	String	Corsa ObjectID to use to copy registration data from
DSPTemplateID	String	Corsa DSP-Subject TemplateID to use as registration template
WhoObjectType	String	Corsa object type of the WHO contact ('E' for organisation, 'P' for person). Default ''
WhoSearchValue	String	The (unique) value to find the WHO contact with. Default ''
WhoFieldValues	NameValue Array	Array of CORSA/WHO? NameValue items with fields and their values

WhoReferenceValues	NameValue Array	Array of CORSA/WHO?NameValue items with references and their values
CheckValues	Boolean	(As from update 2013-wk35) Determines what happens if an invalid value was specified. In this context 'value' refers to the Value property of a NameValue object that was passed to FieldValues, ReferenceValues, WhoFieldValues or WhoReferenceValues. If true, the method will return an error if an invalid value was specified. The registration will not be created or updated. If false, the method will not return an error if an invalid value was specified. The registration will be created or updated. (As from update 2013-wk45 some checks are added to this method in order to display a message if an update of a field value is not allowed) If CheckValues is true, checks are performed for the fields • "poststuk.v_plaats_id" - **administrator** of the document's route step • "obj_vert.vertrouw_id" - if multiple **confidentialities** are linked • "stuk_doss.dossier_id" - if multiple **folders** are linked

For a list of possible values for the FieldValues parameter select menu option 'System | Installation | Programs' in Corsa and execute ontw/d-dspfields.r Select Object kind: 'Document' and Type: 'Registration window fields' in window 'Fields per type'.

For WhoFieldValues, only the fields for Object type 'Person' (when WhoObjectType is 'P') or Object type 'Organisation' (when WhoObjectType is 'E') can be used.

For a list of possible values for the ReferenceValues and WhoReferenceValues parameters use Corsa and go to 'System | Settings | References'. For ReferenceValues, only the fields for object type Document can be used. For WhoReferenceValues, only the fields for object type Person (when WhoObjectType is 'P') or Organisation (when WhoObjectType is 'E') can be used.

For references that are defined with 

, you must pass multiple values in a single NameValue, with values separated by chr(10): value 1 + chr(10) + value 2 + chr(10) value n.
When updating an existing object, existing values in Corsa that are not passed in NameValue are deleted.
For example, if a reference of an object in Corsa has values "A" and "B" and in NameValue "B" + chr(10) + "D" is passed, the result in Corsa will be that the reference has values "B" and "D" (value "A" is deleted).

When WhoObjectType and WhoSearchValue are specified, a WHO contact will be searched/created and used as the sender for the Document registration. The WHO contact will also be used as the case customer if a case document type ID (ws-ps_ds.dsrt_id) is specified in FieldValues or when a DSP-Subject TemplateID is specified in DSPTemplateID. When a WHO contact is created, additional values (like address data) for the registration can be specified in WhoFieldValues and WhoReferenceValues.

The reference specified in a parameter ('zkkme' when WhoObjectType is 'E' and 'zkkmp' when WhoObjectType is 'P') is used to find the WHO contact. The reference of the contact must match the value in WhoSearchValue. Note that confidentialities are taken into account when searching WHO contacts.

- If one contact was found then this is used as the sender and/or case customer.
- If multiple contacts were found, the Result will be false and no create/update of a Document registration will take place.
- If, however, no contact is found, a (broader) general query using WhoSearchValue is performed automatically to find the contact.
  In that case
    - If one contact was found, then this is used as the sender and/or case customer.
    - If multiple contacts were found, a new contact is created and will be used as the sender and/or case customer. This contact is created with the default organisation/person type that is specified in parameter 'defsrte' (for organisations) or 'defsrtp' (for persons).

If FieldValues already contains a sender the specified value will not be overwritten. The same applies to case customer.

Output:

Name	Type	Description
Result	Boolean	True when successful, otherwise false
NewObjectID	String	Corsa ObjectID for the created registration
LastError	String	Error information in case of unsuccessful execution

3.40 CreateMetaFolder

Creates a new or updates an existing Folder (meta-) registration

With the CreateMetaFolder method a Corsa folder registration can be created or updated with specific values and references.

As from update 2014-05 an empty string is allowed for mandatory parameter ObjectKind when modifying a folder (parameter ObjectID is not empty).
*HD 039629

Parameters:

Name	Type	Description

ObjectID	String	Corsa ObjectID for the registration. If the ObjectID contains a valid ObjectID then that registration will be updated. Default “”.
ObjectKind	String	CORSA object kind to use for the registration. See GetObjectKinds for information about retrieving a list of object kinds
FieldValues	NameValue Array	Array of NameValue items with fields and their values
ReferenceValues	NameValue Array	Array of NameValue items with references and their values

For a list of possible values for the FieldValues parameter select menu option 'System | Installation | Programs' in Corsa and execute ontw/d-dspfields.r Select Object kind: 'Folder' and Type: 'Registration window fields' in window 'Fields per type'.

For a list of possible values for the ReferenceValues parameter use Corsa and go to 'System | Settings | References'. Only the fields for object type Folder can be used.

For references that are defined with Multiple, you must pass multiple values in a single NameValue, with values separated by chr(10): value 1 + chr(10) + value 2 + chr(10) value n.
When updating an existing object, existing values in Corsa that are not passed in NameValue are deleted.
For example, if a reference of an object in Corsa has values "A" and "B" and in NameValue "B" + chr(10) + "D" is passed, the result in Corsa will be that the reference has values "B" and "D" (value "A" is deleted).

Output:

Name	Type	Description
Result	Boolean	True when successful, otherwise false
NewObjectID	String	Corsa ObjectID for the created registration
LastError	String	Error information in case of unsuccessful execution

A call to CreateMetaFolder is the same as a call to CreateMetaFolder2 with parameter FolderStructureId = "".

See also

- CreateMetaFolder2
- CreateMetaFolder3
- CreateMetaFolder4
- CreateMetaFolder5

3.41 CreateMetaFolder2

Creates a new or updates an existing Folder (meta-) registration

With the CreateMetaFolder2 method a Corsa folder registration can be created or updated with specific values and references.

With method **CreateMetaFolder(2-5)** you can modify a folder using a folder template (FolderStructureID) with subfolders.
However, these template subfolders are also added as new subfolders each time a folder is modified.
With the new method **CreateMetaFolder6** by using parameter **MapStructureOnlyIfNew** is True, you can define that new subfolders are created for new folders only. (Update 2017-03 for releases 2015-1 and 2016)

As from update 2014-05 an empty string is allowed for mandatory parameter ObjectKind when modifying a folder (parameter ObjectID is not empty).
*HD 039629

Parameters:

Name	Type	Description
ObjectID	String	Corsa ObjectID for the registration. If the ObjectID contains a valid ObjectID then that registration will be updated. Default “”.
ObjectKind	String	CORSA object kind to use for the registration. See GetObjectKinds for information about retrieving a list of object kinds
FieldValues	NameValue Array	Array of NameValue items with fields and their values
ReferenceValues	NameValue Array	Array of NameValue items with references and their values
FolderStructureID	String	CORSA FolderID from which the folder structure will be copied

For a list of possible values for the FieldValues parameter select menu option 'System | Installation | Programs' in Corsa and execute ontw/d-dspfields.r Select Object kind: 'Folder' and Type: 'Registration window fields' in window 'Fields per type'.

For a list of possible values for the ReferenceValues parameter use Corsa and go to 'System | Settings | References'. Only the fields for object type Folder can be used.

For references that are defined with ☑ Multiple, you must pass multiple values in a single NameValue, with values separated by chr(10): value 1 + chr(10) + value 2 + chr(10) value n.
When updating an existing object, existing values in Corsa that are not passed in NameValue are deleted.

For example, if a reference of an object in Corsa has values "A" and "B" and in NameValue "B" + chr(10) + "D" is passed, the result in Corsa will be that the reference has values "B" and "D" (value "A" is deleted).

Output:

Name	Type	Description
Result	Boolean	True when successful, otherwise false
NewObjectID	String	Corsa ObjectID for the created registration
LastError	String	Error information in case of unsuccessful execution

 A call to CreateMetaFolder2 is the same as a call to CreateMetaFolder3 with parameters WhoObjectType = "" and WhoSearchValue = "".

See also

CreateMetaFolder3
CreateMetaFolder4
CreateMetaFolder5

3.42 CreateMetaFolder3

Creates a new or updates an existing Folder (meta-) registration.

With the CreateMetaFolder3 method a Corsa folder registration can be created or updated with specific values and references.

With method **CreateMetaFolder(2-5)** you can modify a folder using a folder template (FolderStructureID) with subfolders.
However, these template subfolders are also added as new subfolders each time a folder is modified.
With the new method **CreateMetaFolder6** by using parameter **MapStructureOnlyIfNew** is True, you can define that new subfolders are created for new folders only. (Update 2017-03 for releases 2015-1 and 2016)

As from update 2014-05 an empty string is allowed for mandatory parameter ObjectKind when modifying a folder (parameter ObjectID is not empty).
*HD 039629

Parameters:

Name	Type	Description
ObjectID	String	Corsa ObjectID for the registration. If the ObjectID contains a valid ObjectID then that registration will be updated. Default ''.

ObjectKind	String	Corsa object kind to use for the registration. See GetObjectKinds for information about retrieving a list of object kinds
FieldValues	NameValue Array	Array of NameValue items with fields and their values
ReferenceValues	NameValue Array	Array of NameValue items with references and their values
FolderStructureID	String	Corsa FolderID from which the folder structure will be copied
WhoObjectType	String	Corsa object type of the WHO contact ('E' for organisation, 'P' for person). Default ''
WhoSearchValue	String	The (unique) value to find the WHO contact with. Default ''

For a list of possible values for the FieldValues parameter select menu option 'System | Installation | Programs' in Corsa and execute ontw/d-dspfields.r Select Object kind: 'Folder' and Type: 'Registration window fields' in window 'Fields per type'.
For WhoFieldValues, only the fields for Object type 'Person' (when WhoObjectType is 'P') or Object type 'Organisation' (when WhoObjectType is 'E') can be used.

For a list of possible values for the ReferenceValues parameter use Corsa and go to 'System | Settings | References'. Only the fields for object type Folder can be used.

For references that are defined with 

, you must pass multiple values in a single NameValue, with values separated by chr(10): value 1 + chr(10) + value 2 + chr(10) value n.
When updating an existing object, existing values in Corsa that are not passed in NameValue are deleted.
For example, if a reference of an object in Corsa has values "A" and "B" and in NameValue "B" + chr(10) + "D" is passed, the result in Corsa will be that the reference has values "B" and "D" (value "A" is deleted).

When WhoObjectType and WhoSearchValue are specified, a WHO contact will be searched/created and used as the main contact for the Folder registration.

The reference specified in a parameter ('zkkme' when WhoObjectType is 'E' and 'zkkmp' when WhoObjectType is 'P') is used to find the WHO contact. The reference of the contact must match the value in WhoSearchValue.

- If one contact was found then this is used as the main contact. .
- If multiple contacts were found, the Result will be false and no create/update of a Folder registration will take place.
- If, however, no contact is found, a (broader) general query using WhoSearchValue is performed automatically to find the contact.
  In that case
    - If one contact was found, then this is used as the main contact..

- If multiple contacts were found, a new contact is created and will be used as the main contact.
  This contact is created with the default organisation/person type that is specified in parameter 'defsrte' (for organisations) or 'defsrtp' (for persons).

If FieldValues already contains a main contact the specified value will not be overwritten.

Output:

Name	Type	Description
Result	Boolean	True when successful, otherwise false
NewObjectID	String	Corsa ObjectID for the created registration
LastError	String	Error information in case of unsuccessful execution

 A call to CreateMetaFolder3 is the same as a call to CreateMetaFolder4 with parameters WhoFieldValues = null and WhoReferenceValues = null.

See also

CreateMetaFolder4
CreateMetaFolder5

3.43 CreateMetaFolder4

Creates a new or updates an existing Folder (meta-) registration.

With the CreateMetaFolder4 method a Corsa folder registration can be created or updated with specific values and references.

With method **CreateMetaFolder(2-5)** you can modify a folder using a folder template (FolderStructureID) with subfolders.
However, these template subfolders are also added as new subfolders each time a folder is modified.
With the new method **CreateMetaFolder6** by using parameter **MapStructureOnlyIfNew** is True, you can define that new subfolders are created for new folders only. (Update 2017-03 for releases 2015-1 and 2016)

As from update 2014-05 an empty string is allowed for mandatory parameter ObjectKind when modifying a folder (parameter ObjectID is not empty).
*HD 039629

Parameters:

Name	Type	Description

ObjectID	String	Corsa ObjectID for the registration. If the ObjectID contains a valid ObjectID then that registration will be updated. Default ''.
ObjectKind	String	Corsa object kind to use for the registration. See GetObjectKinds for information about retrieving a list of object kinds
FieldValues	NameValue Array	Array of NameValue items with fields and their values
ReferenceValues	NameValue Array	Array of NameValue items with references and their values
FolderStructureID	String	Corsa FolderID from which the folder structure will be copied
WhoObjectType	String	Corsa object type of the WHO contact ('E' for organisation, 'P' for person). Default ''
WhoSearchValue	String	The (unique) value to find the WHO contact with. Default ''
WhoFieldValues	NameValue Array	Array of CORSA/WHO? NameValue items with fields and their values
WhoReferenceValues	NameValue Array	Array of CORSA/WHO?NameValue items with references and their values

For a list of possible values for the FieldValues parameter select menu option 'System | Installation | Programs' in Corsa and execute ontw/d-dspfields.r Select Object kind: 'Folder' and Type: 'Registration window fields' in window 'Fields per type'.
For WhoFieldValues, only the fields for Object type 'Person' (when WhoObjectType is 'P') or Object type 'Organisation' (when WhoObjectType is 'E') can be used.

For a list of possible values for the ReferenceValues and WhoReferenceValues parameters use Corsa and go to 'System | Settings | References'. For ReferenceValues, only the fields for object type Folder can be used. For WhoReferenceValues, only the fields for object type Person (when WhoObjectType is 'P') or Organisation (when WhoObjectType is 'E') can be used.

For references that are defined with 

, you must pass multiple values in a single NameValue, with values separated by chr(10): value 1 + chr(10) + value 2 + chr(10) value n.
When updating an existing object, existing values in Corsa that are not passed in NameValue are deleted.
For example, if a reference of an object in Corsa has values "A" and "B" and in NameValue "B" + chr(10) + "D" is passed, the result in Corsa will be that the reference has values "B" and "D" (value "A" is deleted).

When WhoObjectType and WhoSearchValue are specified, a WHO contact will be searched/created and used as the main contact for the Folder registration. When a WHO contact is created, additional values (like address data) for the registration can be specified in WhoFieldValues and WhoReferenceValues.

The reference specified in a parameter ('zkkme' when WhoObjectType is 'E' and 'zkkmp' when WhoObjectType is 'P') is used to find the WHO contact. The reference of the contact must match the value in WhoSearchValue. Note that confidentialities are taken into account when searching WHO contacts.

- If one contact was found then this is used as the main contact. .
- If multiple contacts were found, the Result will be false and no create/update of a Folder registration will take place.
- If, however, no contact is found, a (broader) general query using WhoSearchValue is performed automatically to find the contact.
  In that case
    - If one contact was found, then this is used as the main contact..
    - If multiple contacts were found, a new contact is created and will be used as the main contact.
      This contact is created with the default organisation/person type that is specified in parameter 'defsrte' (for organisations) or 'defsrtp' (for persons).

If FieldValues already contains a main contact the specified value will not be overwritten.

Output:

Name	Type	Description
Result	Boolean	True when successful, otherwise false
NewObjectID	String	Corsa ObjectID for the created registration
LastError	String	Error information in case of unsuccessful execution

 A call to CreateMetaFolder4 is the same as a call to CreateMetaFolder5 with parameter CheckValues = false.

See also
CreateMetaFolder5

3.44 CreateMetaFolder5

Creates a new or updates an existing Folder (meta-) registration.

With the CreateMetaFolder5 method a Corsa folder registration can be created or updated with specific values and references.

With method **CreateMetaFolder(2-5)** you can modify a folder using a folder template (FolderStructureID) with subfolders.
However, these template subfolders are also added as new subfolders each time a folder is modified.

With the new method **CreateMetaFolder6** by using parameter **MapStructureOnlyIfNew** is True, you can define that new subfolders are created for new folders only. (Update 2017-03 for releases 2015-1 and 2016)

As from update 2014-05 an empty string is allowed for mandatory parameter ObjectKind when modifying a folder (parameter ObjectID is not empty).
*HD.039629

Parameters:

Name	Type	Description
ObjectID	String	Corsa ObjectID for the registration. If the ObjectID contains a valid ObjectID then that registration will be updated. Default ''.
ObjectKind	String	Corsa object kind to use for the registration. See GetObjectKinds for information about retrieving a list of object kinds
FieldValues	NameValue Array	Array of NameValue items with fields and their values
ReferenceValues	NameValue Array	Array of NameValue items with references and their values
FolderStructureID	String	Corsa FolderID from which the folder structure will be copied
WhoObjectType	String	Corsa object type of the WHO contact ('E' for organisation, 'P' for person). Default ''
WhoSearchValue	String	The (unique) value to find the WHO contact with. Default ''
WhoFieldValues	NameValue Array	Array of CORSA/WHO? NameValue items with fields and their values
WhoReferenceValues	NameValue Array	Array of CORSA/WHO?NameValue items with references and their values
CheckValues	Boolean	(As from update 2013-wk35) Determines what happens if an invalid value was specified. In this context 'value' refers to the Value property of a NameValue object that was passed to FieldValues, ReferenceValues, WhoFieldValues or WhoReferenceValues. If true, the method will return an error if an invalid value was specified. The registration will not be created or updated. If false, the method will not return an error if an invalid value was specified. The registration will be created or updated.

For a list of possible values for the FieldValues parameter select menu option 'System | Installation | Programs' in Corsa and execute ontw/d-dspfields.r Select Object kind: 'Folder' and Type: 'Registration window fields' in window 'Fields per type'.
For WhoFieldValues, only the fields for Object type 'Person' (when WhoObjectType is 'P') or Object type 'Organisation' (when WhoObjectType is 'E') can be used.

For a list of possible values for the ReferenceValues and WhoReferenceValues parameters use Corsa and go to 'System | Settings | References'. For ReferenceValues, only the fields for object type Folder can be used. For WhoReferenceValues, only the fields for object type Person (when WhoObjectType is 'P') or Organisation (when WhoObjectType is 'E') can be used.

For references that are defined with 

, you must pass multiple values in a single NameValue, with values separated by chr(10): value 1 + chr(10) + value 2 + chr(10) value n.
When updating an existing object, existing values in Corsa that are not passed in NameValue are deleted.
For example, if a reference of an object in Corsa has values "A" and "B" and in NameValue "B" + chr(10) + "D" is passed, the result in Corsa will be that the reference has values "B" and "D" (value "A" is deleted).

When WhoObjectType and WhoSearchValue are specified, a WHO contact will be searched/created and used as the main contact for the Folder registration. When a WHO contact is created, additional values (like address data) for the registration can be specified in WhoFieldValues and WhoReferenceValues.

The reference specified in a parameter ('zkkme' when WhoObjectType is 'E' and 'zkkmp' when WhoObjectType is 'P') is used to find the WHO contact. The reference of the contact must match the value in WhoSearchValue. Note that confidentialities are taken into account when searching WHO contacts.

- If one contact was found then this is used as the main contact..
- If multiple contacts were found, the Result will be false and no create/update of a Folder registration will take place.
- If, however, no contact is found, a (broader) general query using WhoSearchValue is performed automatically to find the contact.
  In that case
    - If one contact was found, then this is used as the main contact..
    - If multiple contacts were found, a new contact is created and will be used as the main contact.
      This contact is created with the default organisation/person type that is specified in parameter 'defsrte' (for organisations) or 'defsrtp' (for persons).

If FieldValues already contains a main contact the specified value will not be overwritten.

Output:

Name	Type	Description
Result	Boolean	True when successful, otherwise false

NewObjectID	String	Corsa ObjectID for the created registration
LastError	String	Error information in case of unsuccessful execution

3.45 CreateMetaFolder6

Creates a new or updates an existing Folder (meta-) registration.

With the CreateMetaFolder6 method a Corsa folder registration can be created or updated with specific values and references.

With method **CreateMetaFolder(2-5)** you can modify a folder using a folder template (FolderStructureID) with subfolders.
However, these template subfolders are also added as new subfolders each time a folder is modified.
With the new method **CreateMetaFolder6** by using parameter **MapStructureOnlyIfNew** is True, you can define that new subfolders are created for new folders only. (Update 2017-03 for releases 2015-1 and 2016)

As from update 2014-05 an empty string is allowed for mandatory parameter ObjectKind when modifying a folder (parameter ObjectID is not empty).
*HD 039629

Parameters:

Name	Type	Description
ObjectID	String	Corsa ObjectID for the registration. If the ObjectID contains a valid ObjectID then that registration will be updated. Default ''.
ObjectKind	String	Corsa object kind to use for the registration. See GetObjectKinds for information about retrieving a list of object kinds
FieldValues	NameValue Array	Array of NameValue items with fields and their values
ReferenceValues	NameValue Array	Array of NameValue items with references and their values
FolderStructureID	String	Corsa FolderID from which the folder structure will be copied
WhoObjectType	String	Corsa object type of the WHO contact ('E' for organisation, 'P' for person). Default ''
WhoSearchValue	String	The (unique) value to find the WHO contact with. Default ''
WhoFieldValues	NameValue Array	Array of CORSA/WHO? NameValue items with fields and their values

WhoReferenceValues	NameValue Array	Array of CORSA/WHO?NameValue items with references and their values
CheckValues	Boolean	(As from update 2013-wk35) Determines what happens if an invalid value was specified. In this context 'value' refers to the Value property of a NameValue object that was passed to FieldValues, ReferenceValues, WhoFieldValues or WhoReferenceValues. If true, the method will return an error if an invalid value was specified. The registration will not be created or updated. If false, the method will not return an error if an invalid value was specified. The registration will be created or updated.
MapStructureOnlyIfNew	Boolean	(As from update 2017-03 for releases 2015-1 and 2016) If true, new subfolders (from folder template FolderStructureID) are created for new folders only. If false, template subfolders are also added as new subfolders each time a folder is modified (which is the default behaviour of method CreateMetaFolder(1-5)).

For a list of possible values for the FieldValues parameter select menu option 'System | Installation | Programs' in Corsa and execute ontw/d-dspfields.r Select Object kind: 'Folder' and Type: 'Registration window fields' in window 'Fields per type'.
For WhoFieldValues, only the fields for Object type 'Person' (when WhoObjectType is 'P') or Object type 'Organisation' (when WhoObjectType is 'E') can be used.

For a list of possible values for the ReferenceValues and WhoReferenceValues parameters use Corsa and go to 'System | Settings | References'. For ReferenceValues, only the fields for object type Folder can be used. For WhoReferenceValues, only the fields for object type Person (when WhoObjectType is 'P') or Organisation (when WhoObjectType is 'E') can be used.

For references that are defined with 

, you must pass multiple values in a single NameValue, with values separated by chr(10): value 1 + chr(10) + value 2 + chr(10) value n.
When updating an existing object, existing values in Corsa that are not passed in NameValue are deleted.
For example, if a reference of an object in Corsa has values "A" and "B" and in NameValue "B" + chr(10) + "D" is passed, the result in Corsa will be that the reference has values "B" and "D" (value "A" is deleted).

When WhoObjectType and WhoSearchValue are specified, a WHO contact will be searched/created and used as the main contact for the Folder registration. When a WHO contact is created, additional values (like address data) for the registration can be specified in WhoFieldValues and WhoReferenceValues.

The reference specified in a parameter ('zkkme' when WhoObjectType is 'E' and 'zkkmp' when WhoObjectType is 'P') is used to find the WHO contact. The reference of the contact must match the value in WhoSearchValue. Note that confidentialities are taken into account when searching WHO contacts.

- If one contact was found then this is used as the main contact..
- If multiple contacts were found, the Result will be false and no create/update of a Folder registration will take place.
- If, however, no contact is found, a (broader) general query using WhoSearchValue is performed automatically to find the contact.
  In that case
    - If one contact was found, then this is used as the main contact..
    - If multiple contacts were found, a new contact is created and will be used as the main contact.
      This contact is created with the default organisation/person type that is specified in parameter 'defsrte' (for organisations) or 'defsrtp' (for persons).

If FieldValues already contains a main contact the specified value will not be overwritten.

Output:

Name	Type	Description
Result	Boolean	True when successful, otherwise false
NewObjectID	String	Corsa ObjectID for the created registration
LastError	String	Error information in case of unsuccessful execution

3.46 CreateMetaOrganisation

Creates a new or updates an existing Organisation (meta-) registration

With the CreateMetaOrganisation method a Corsa organisation registration can be created or updated with specific values and references.

As from update 2014-05 an empty string is allowed for mandatory parameter ObjectKind when modifying organization data (parameter ObjectID is not empty).
*HD.039629

Parameters:

Name	Type	Description
ObjectID	String	Corsa ObjectID for the registration. If the ObjectID contains a valid ObjectID then that registration will be updated. Default “”.
ObjectKind	String	Corsa object kind to use for the registration. See GetObjectKinds for information about retrieving a list of object kinds

FieldValues	NameValue Array	Array of NameValue items with fields and their values
ReferenceValues	NameValue Array	Array of NameValue items with references and their values

For a list of possible values for the FieldValues parameter select menu option 'System | Installation | Programs' in Corsa and execute ontw/d-dspfields.r Select Object kind: 'Organisation' and Type: 'Registration window fields' in window 'Fields per type'.

For a list of possible values for the ReferenceValues parameter use Corsa and go to 'System | Settings | References'. Only the fields for object type Organisation can be used.

For references that are defined with 

, you must pass multiple values in a single NameValue, with values separated by chr(10): value 1 + chr(10) + value 2 + chr(10) value n.
When updating an existing object, existing values in Corsa that are not passed in NameValue are deleted.
For example, if a reference of an object in Corsa has values "A" and "B" and in NameValue "B" + chr(10) + "D" is passed, the result in Corsa will be that the reference has values "B" and "D" (value "A" is deleted).

Output:

Name	Type	Description
Result	Boolean	True when successful, otherwise false
NewObjectID	String	Corsa ObjectID for the created registration
LastError	String	Error information in case of unsuccessful execution

A call to CreateMetaOrganisation is the same as a call to CreateMetaOrganisation2 with parameter CheckValues = false.

3.47 CreateMetaOrganisation2

Creates a new or updates an existing Organisation (meta-) registration.

With the CreateMetaOrganisation2 method a Corsa organisation registration can be created or updated with specific values and references.

As from update 2014-05 an empty string is allowed for mandatory parameter ObjectKind when modifying organization data (parameter ObjectID is not empty).
*HD 039629

Parameters:

Name	Type	Description

ObjectID	String	Corsa ObjectID for the registration. If the ObjectID contains a valid ObjectID then that registration will be updated. Default “”.
ObjectKind	String	Corsa object kind to use for the registration. See GetObjectKinds for information about retrieving a list of object kinds
FieldValues	NameValue Array	Array of NameValue items with fields and their values
ReferenceValues	NameValue Array	Array of NameValue items with references and their values
CheckValues	Boolean	(As from update 2013-wk35) Determines what happens if an invalid value was specified. In this context 'value' refers to the Value property of a NameValue object that was passed to FieldValues or ReferenceValues. If true, the method will return an error if an invalid value was specified. The registration will not be created or updated. If false, the method will not return an error if an invalid value was specified. The registration will be created or updated.

For a list of possible values for the FieldValues parameter select menu option 'System | Installation | Programs' in Corsa and execute ontw/d-dspfields.r Select Object kind: 'Organisation' and Type: 'Registration window fields' in window 'Fields per type'.

For a list of possible values for the ReferenceValues parameter use Corsa and go to 'System | Settings | References'. Only the fields for object type Organisation can be used.

For references that are defined with 

, you must pass multiple values in a single NameValue, with values separated by chr(10): value 1 + chr(10) + value 2 + chr(10) value n.
When updating an existing object, existing values in Corsa that are not passed in NameValue are deleted.
For example, if a reference of an object in Corsa has values "A" and "B" and in NameValue "B" + chr(10) + "D" is passed, the result in Corsa will be that the reference has values "B" and "D" (value "A" is deleted).

Output:

Name	Type	Description
Result	Boolean	True when successful, otherwise false
NewObjectID	String	Corsa ObjectID for the created registration
LastError	String	Error information in case of unsuccessful execution

3.48 CreateMetaPerson

Creates a new or updates an existing Person (meta-) registration

With the CreateMetaPerson method a Corsa person registration can be created or updated with specific values and references.

As from update 2014-05 an empty string is allowed for mandatory parameter ObjectKind when modifying person data (parameter ObjectID is not empty).
*HD.039629

Parameters:

Name	Type	Description
ObjectID	String	Corsa ObjectID for the registration. If the ObjectID contains a valid ObjectID then that registration will be updated. Default “”.
ObjectKind	String	Corsa object kind to use for the registration. See GetObjectKinds for information about retrieving a list of object kinds
FieldValues	NameValue Array	Array of NameValue items with fields and their values
ReferenceValues	NameValue Array	Array of NameValue items with references and their values

For a list of possible values for the FieldValues parameter select menu option 'System | Installation | Programs' in Corsa and execute ontw/d-dspfields.r Select Object kind: 'Person' and Type: 'Registration window fields' in window 'Fields per type'.

For a list of possible values for the ReferenceValues parameter use Corsa and go to 'System | Settings | References'. Only the fields for object type Person can be used.

For references that are defined with ☑ Multiple, you must pass multiple values in a single NameValue, with values separated by chr(10): value 1 + chr(10) + value 2 + chr(10) value n.
When updating an existing object, existing values in Corsa that are not passed in NameValue are deleted.
For example, if a reference of an object in Corsa has values "A" and "B" and in NameValue "B" + chr(10) + "D" is passed, the result in Corsa will be that the reference has values "B" and "D" (value "A” is deleted).

Output:

Name	Type	Description

Result	Boolean	True when successful, otherwise false
NewObjectID	String	Corsa ObjectID for the created registration
LastError	String	Error information in case of unsuccessful execution

As from update 2013-wk24: After a Person (meta-) registration is created /modified with method CreateMetaPerson, the address of the accompanying organization is automatically displayed in Corsa /Enter value if field "Persoon.ext_rel_id" field is given.

 A call to CreateMetaPerson is the same as a call to CreateMetaPerson2 with parameter CheckValues = false.

3.49 CreateMetaPerson2

Creates a new or updates an existing Person (meta-) registration

With the CreateMetaPerson2 method a Corsa person registration can be created or updated with specific values and references.

As from update 2014-05 an empty string is allowed for mandatory parameter ObjectKind when modifying person data (parameter ObjectID is not empty).
*HD.039629

Parameters:

Name	Type	Description
ObjectID	String	Corsa ObjectID for the registration. If the ObjectID contains a valid ObjectID then that registration will be updated. Default “”.
ObjectKind	String	Corsa object kind to use for the registration. See GetObjectKinds for information about retrieving a list of object kinds
FieldValues	NameValue Array	Array of NameValue items with fields and their values
ReferenceValues	NameValue Array	Array of NameValue items with references and their values
CheckValues	Boolean	(As from update 2013-wk35) Determines what happens if an invalid value was specified. In this context 'value' refers to the Value property of a NameValue object that was passed to FieldValues or ReferenceValues. If true, the method will return an error if an invalid value was specified. The registration will not be created or updated.

		If false, the method will not return an error if an invalid value was specified. The registration will be created or updated.

For a list of possible values for the FieldValues parameter select menu option 'System | Installation | Programs' in Corsa and execute ontw/d-dspfields.r Select Object kind: 'Person' and Type: 'Registration window fields' in window 'Fields per type'.

For a list of possible values for the ReferenceValues parameter use Corsa and go to 'System | Settings | References'. Only the fields for object type Person can be used.

For references that are defined with 

, you must pass multiple values in a single NameValue, with values separated by chr(10): value 1 + chr(10) + value 2 + chr(10) value n.
When updating an existing object, existing values in Corsa that are not passed in NameValue are deleted.
For example, if a reference of an object in Corsa has values "A" and "B" and in NameValue "B" + chr(10) + "D" is passed, the result in Corsa will be that the reference has values "B" and "D" (value "A" is deleted).

Output:

Name	Type	Description
Result	Boolean	True when successful, otherwise false
NewObjectID	String	Corsa ObjectID for the created registration
LastError	String	Error information in case of unsuccessful execution

As from update 2013-wk24: After a Person (meta-) registration is created /modified with method CreateMetaPerson, the address of the accompanying organization is automatically displayed in Corsa /Enter value if field "Persoon.ext_rel_id" field is given.

3.50 CreateMetaRealEstate

Creates a new or updates an existing Real Estate (meta-) registration.

With the CreateMetaRealEstate method a Corsa real estate registration can be created or updated with specific values and references.

Parameters:

Name	Type	Description
ObjectID	String	Corsa ObjectID for the registration. If the ObjectID contains a valid ObjectID then that registration will be updated. Default “”.

ObjectKind	String	Corsa object kind to use for the registration. See GetObjectKinds for information about retrieving a list of object kinds
FieldValues	NameValue Array	Array of NameValue items with fields and their values
ReferenceValues	NameValue Array	Array of NameValue items with references and their values
CheckValues	Boolean	Determines what happens if an invalid value was specified. In this context 'value' refers to the Value property of a NameValue object that was passed to FieldValues or ReferenceValues. If true, the method will return an error if an invalid value was specified. The registration will not be created or updated. If false, the method will not return an error if an invalid value was specified. The registration will be created or updated.

For a list of possible values for the FieldValues parameter select menu option 'System | Installation | Programs' in Corsa and execute ontw/d-dspfields.r Select Object kind: 'Real estate' and Type: 'Registration window fields' in window 'Fields per type'.

For a list of possible values for the ReferenceValues parameter use Corsa and go to 'System | Settings | References'. Only the fields for object type Real Estate can be used.

For references that are defined with 

, you must pass multiple values in a single NameValue, with values separated by chr(10): value 1 + chr(10) + value 2 + chr(10) value n.
When updating an existing object, existing values in Corsa that are not passed in NameValue are deleted.
For example, if a reference of an object in Corsa has values "A" and "B" and in NameValue "B" + chr(10) + "D" is passed, the result in Corsa will be that the reference has values "B" and "D" (value "A" is deleted).

Output:

Name	Type	Description
Result	Boolean	True when successful, otherwise false
NewObjectID	String	Corsa ObjectID for the created registration
LastError	String	Error information in case of unsuccessful execution

3.51 DeleteCopyHolder

The DeleteCopyHolder function can be used to remove a copyholder of a document registration.

Parameters:

Name	Type	Description
ObjectID	String	ID of a document registration

RelationType	String	Relation type for the copyholder
RelationID	String	Relation id for the copyholder

Output:

Name	Type	Description
Result	Boolean	True when successful otherwise false
LastError	String	Error information in case of unsuccessful execution

3.52 DeleteFileVersion

Deletes the last file version of a specific object.

With the DeleteFileVersion method the last file version of a specific object can be deleted.

Parameters:

Name	Type	Description
ObjectType	String	Corsa ObjectType of the object for the file version to delete
ObjectID	String	Corsa ObjectID of the object for the file version to delete

Output:

Name	Type	Description
Result	Boolean	True when successful, otherwise false
LastError	String	Error information in case of unsuccessful execution

3.53 DeleteFolderAction

Deletes a specific action from a specific folder-object (registration).

Input parameters

Name	Type	Description
FolderID	String	Corsa ObjectID of the folder.
ActionID	String	ActionID of the action to add to the specific folder.

Output parameters

Name	Type	Description
• Result	Boolean	True when successful, otherwise false.

• LastError	String	Error information in case of unsuccessful execution.

3.54 DeleteObject

Deletes a specific object (registration).

A call to DeleteObject is the same as a call to DeleteObject2 with the following parameters:

Parameters:

Name	Type	Description
ObjectType	String	Corsa ObjectType of the object to delete
ObjectID	String	Corsa ObjectID of the object to delete
Options	NameValue[]	null

See also
DeleteObject2
DeleteObject3

3.55 DeleteObject2

Deletes a specific object (registration).

A call to DeleteObject2 is the same as a call to DeleteObject3 with the following parameters:

Parameters:

Name	Type	Description
ObjectType	String	Corsa ObjectType of the object to delete
ObjectID	String	Corsa ObjectID of the object to delete
DeleteLogically	Boolean	False
Options	NameValue[]	Additional options. Default null.

See also
DeleteObject3

3.56 DeleteObject3

Physically or logically (= to recycle bin) deletes a specific object (registration).

With the DeleteObject method a specific Corsa object (registration) can be physically or logically (= to recycle bin) deleted. When an object is deleted physically the associated file versions are also deleted.

Parameters:

Name	Type	Description
ObjectType	String	Corsa ObjectType of the object to delete
ObjectID	String	Corsa ObjectID of the object to delete
DeleteLogically	Boolean	True to logically delete (= to recycle bin), False to physically delete the object.
Options	NameValue[]	Additional options. Default null.

Use the Options parameter to specify what must happen if certain conditions are met by adding one or more NameValues.
Specifying these options prevents possible questions from being returned in LastError. Specify “true” or “false” as value.
When deleting a document (ObjectType = “S”), the following options can be used:

- “DelRelDoss”: specifies whether to delete the document in case of related folders. Related Corsa parameter is “chkpd”.
- “DelRelCase”: specifies whether to delete the document in case of related cases. Related Corsa parameter is “chkpc”.

When deleting a folder (ObjectType = “D”), the following option can be used:

- “DelRelPost”: specifies whether related documents must be deleted (as from release 2014).

Output:

Name	Type	Description
Result	Boolean	True when successful, otherwise false
LastError	String	Error information in case of unsuccessful execution

(as from update 2015-01)

3.57 DeletePermit

The DeletePermit method deletes a permit of an object (document, case, person, organisation, etc.)

Note: this method is supported as from release 2010-4.

Parameters:

Name	Type	Description
ObjectType	String	Object type
ObjectID	String	Object ID
UserID	String	Corsa User ID

Output:

Name	Type	Description
Result	Boolean	True when successful, otherwise false
LastError	String	Error information in case of unsuccessful execution

3.58 DelSenderAddr

The DelSenderAddr method can be used to delete Sender/Addr. of a document registration.

Parameters:

Name	Type	Description
ObjectID	String	ID of document registration
AfzAanRelId	String	Relation id for the Sender/Addr

Output:

Name	Type	Description
Result	Boolean	True when successful, otherwise false
LastError	String	Error information in case of unsuccessful execution

3.59 Disconnect

Disconnect session.

Abandon the session and disconnect from the Corsa database.

Parameters:

None

Output:

Name	Type	Description
Result	Boolean	True when successful, otherwise false

3.60 DownloadChunk

Download a chunk of a file from the upload folder on the server.

Parameters:

Name	Type	Description

FileName	String	The filename to download
Offset	Long	The offset at which to fetch the next chunk
BufferSize	Integer	The size of the chunk

Output:

Name	Type	Description
	Byte array	The chunk as a byte array

3.61 ExportGetObjectList

Returns an XML file based on Corsa/EXPORT.

Parameters:

Name	Type	Description
AppID	String	Should always be “”, reserved for internal usage
XMLCommandFile	Byte Array	Byte Array with contents of input XML file (see below)

The XMLCommandFile parameter should contain a valid XML file and the following parmeters:

```
<?xml version="1.0" ?>
<Corsa>
        <Application>Appliction ID provided by BCT</Application>
        <Environment>Code of the export environment</Environment>
        <InclFiles>yes/no</InclFiles>
        <MaxDocs>100</MaxDocs>
</Corsa>
```

Output:

Name	Type	Description
Result	Boolean	True when successful, otherwise false
XMLExport	Byte Array	XML file with export data
LastError	String	Error information in case of unsuccessful execution

Auditing:
This method supports object auditing. Auditing is applied as follows:

	Folders	Agenda items	Documents	Native files	Archive files
Audit option	Find	MsWord	MsWord	Native file	Archive file
Audit details	-	-	-	-	-

3.62 ExportObjectListSetStatus

Sets export status for a specified list of objects.

Parameters:

Name	Type	Description
AppID	String	Should always be “”, reserved for internal usage
XMLObjStatusList	Byte Array	Byte Array with contents of input XML file (see below)

The XMLObjStatusList parameter should contain a valid XML file and should contain the following parameters:

```
<?xml version="1.0" ?>
<Corsa>
  <Application>Appliction ID provided by BCT</Application>
  <Environment>Code of the export environment</Environment>
  <Status>
      <Object>
          <ObjectType>S</ObjectType>
          <ObjectId>INK123ABC</ObjectId>
          <StatusOk>1</StatusOk>
      </Object>
      <Object>
          <ObjectType>A</ObjectType>
          <ObjectId>AGP001</ObjectId>
          <StatusOk>0</StatusOk>
      </Object>
      <Object>
          <ObjectType>S</ObjectType>
          <ObjectId>DH.0007</ObjectId>
          <StatusOk>1</StatusOk>
      </Object>
  </Status>
</Corsa>
```

Output:

Name	Type	Description
Result	Boolean	True when successful, otherwise false
LastError	String	Error information in case of unsuccessful execution

3.63 FileVersionCheckIn

Set the 'checked in' status for a file version for a specific object.

Parameters:

Name	Type	Description
ObjectType	String	Corsa ObjectType of the object
ObjectID	String	Corsa ObjectID of the object

Output:

Name	Type	Description
Result	Boolean	True when successful, otherwise false
LastError	String	Error information in case of unsuccessful execution

3.64 FileVersionCheckOut

Set the 'checked out' status for a file version for a specific object.

Parameters:

Name	Type	Description
ObjectType	String	Corsa ObjectType of the object
ObjectID	String	Corsa ObjectID of the object
FileVersion	Integer	Corsa File Version to check out
CheckedOutPath	String	Path and filename where the file is checked out (optional)

Output:

Name	Type	Description
Result	Boolean	True when successful, otherwise false
LastError	String	Error information in case of unsuccessful execution

3.65 GenerateId

Generates a unique id for a given type, Document or Zaak.

Required: Corsa parameter “gemcode”, which must contain a 4 character code.
The id is created as follows: the value of “gemcode” followed by a unique sequential number per type.

Parameters:

Name	Type	Description
TypeId	CMIS_IdType	The type, either Document or Zaak

Output:

Name	Type	Description
Result	Boolean	True when successful, otherwise false
Id	String	Generated Id for the provided type.
LastError	String	Error information in case of unsuccessful execution.

Example:
If gemcode is “ABCD”, the id could be “ABCD000000000010”.

3.66 GetCaseTodoList

Retrieves the to do list for cases.

With the GetCaseTodoList method a DataTable containing the to do list for cases can be retrieved for the currently connected user or a specific employee, role or organization unit. When the Id parameter is set to an empty string then the employee id for the connected user will be used.

Parameters:

Name	Type	Description
TodoListType	CaseTodoListType	The to do list type: Employee, Role or Organisation-unit
Id	String	The Employee, Role or Organisation-unit ID to retrieve the todo list for
Status	CaseTodoListStatus	The status: All, Completed, InProcess, InProgress, Reminder

Output:

Name	Type	Description
Result	DataTable	Null when failed, otherwise a DataTable containing a table with the todo list
LastError	String	Error information in case of unsuccessful execution

Corsa72WS4j: The result DataTable will be returned as a byte array containing an XML representation of the DataTable

Auditing:
This method supports object auditing. Auditing is applied as follows:

Audit option	Find
Audit details	-

 A call to GetCaseTodoList is the same as a call to GetCaseTodoList2 with parameter MaxResult = 0.

3.67 GetCaseTodoList2

Retrieves the to do list for cases.

With the GetCaseTodoList method a DataTable containing the to do list for cases can be retrieved for the currently connected user or a specific employee, role or organization unit. When the Id parameter is set to an empty string then the employee id for the connected user will be used.

Parameters:

Name	Type	Description
TodoListType	CaseTodoListType	The to do list type: Employee, Role or Organisation-unit
Id	String	The Employee, Role or Organisation-unit ID to retrieve the todo list for
Status	CaseTodoListStatus	The status: All, Completed, InProcess, InProgress, Reminder
MaxResult	Integer	Maximum number of items to retrieve. Use 0 to not limit the result

Output:

Name	Type	Description
Result	DataTable	Null when failed, otherwise a DataTable containing a table with the todo list
LastError	String	Error information in case of unsuccessful execution

Auditing:
This method supports object auditing. Auditing is applied as follows:

Audit option	Find
Audit details	-

Corsa72WS4j: The result DataTable will be returned as a byte array containing an XML representation of the DataTable

3.68 GetChunkSize

Get the value of Web Service configuration parameter “ChunkSize”.

Parameters:

None

Output:

Name	Type	Description
Result	String	Returns the chunk size for file transfer in KB

3.69 GetConfidentialities

Returns confidentialities.

With the GetConfidentialities method a list of confidentialities which are defined in Corsa can be retrieved.

Parameters:

None

Output:

Name	Type	Description
Result	IdDescription Array	Null when failed, otherwise an array of IdDescription items containing confidentialities
LastError	String	Error information in case of unsuccessful execution

3.70 GetCopyHolderRelationTypes

The GetCopyHolderRelationTypes function can be used to retrieve a list of possible relation types for manipulating copyholder data of a document registration. The output parameter Result contains a table with possible relation types. Each record consists of an id and a description.

Parameters:

None

Output:

Name	Type	Description
Result	DataTable	Null when failed, otherwise a DataTable containing a table with the relation types
LastError	String	Error information in case of unsuccessful execution

Corsa72WS4j: The result DataTable will be returned as a byte array containing an XML representation of the DataTable.

3.71 GetCopyHolders

The GetCopyHolders function can be used to retrieve a list with copyholder data of a document registration.

Parameters:

Name	Type	Description
ObjectID	String	ID of a document registration

Output:

Name	Type	Description
Result	DataTable	Null when failed, otherwise a DataTable containing a table with the relation types
LastError	String	Error information in case of unsuccessful execution

Auditing:
This method supports object auditing. Auditing is applied as follows:

Audit option	Detail
Audit details	GetAllKopieHouders

Corsa72WS4j: The result DataTable will be returned as a byte array containing an XML representation of the DataTable.

3.72 GetDocTodoList

Retrieves the to do list for documents.

With the GetDocTodoList method a DataTable containing the to do list for documents can be retrieved for the currently connected user or a specific employee or organization unit. When EmployeeId and OrganisationUnitId are set to an empty string then the Employee ID for the connected user will be used.

Parameters:

Name	Type	Description
EmployeeId	String	The Corsa Employee ID to retrieve the todo list for
OrganisationUnitId	String	The Corsa Organisation-unit ID to retrieve the todo list for
ProcessType	TodoListProcessType	The process type, either Administrative or Managerial
Status	TodoListStatus	The status: ToBeReceived, InProcess, Reminder, Completed or Copies
DateFrom	String	Date filter, format is “DD/MM/YYYY”
DateTo	String	Date filter, format is “DD/MM/YYYY”

Output:

Name	Type	Description
Result	DataTable	Null when failed, otherwise a DataTable containing a table with the todo list
LastError	String	Error information in case of unsuccessful execution

Corsa72WS4j: The result DataTable will be returned as a byte array containing an XML representation of the DataTable

Auditing:
This method supports object auditing. Auditing is applied as follows:

Audit option	Find
Audit details	-


A call to GetDocTodoList is the same as a call to GetDocTodoList2 with parameter MaxResult = 0.

3.73 GetDocTodoList2

Retrieves the to do list for documents.

With the GetDocTodoList method a DataTable containing the to do list for documents can be retrieved for the currently connected user or a specific employee or organization unit. When EmployeeId and OrganisationUnitId are set to an empty string then the Employee ID for the connected user will be used.

Parameters:

Name	Type	Description
EmployeeId	String	The Corsa Employee ID to retrieve the todo list for
OrganisationUnitId	String	The Corsa Organisation-unit ID to retrieve the todo list for
ProcessType	TodoListProcessType	The process type, either Administrative or Managerial
Status	TodoListStatus	The status: ToBeReceived, InProcess, Reminder, Completed or Copies
DateFrom	String	Date filter, format is “DD/MM/YYYY”
DateTo	String	Date filter, format is “DD/MM/YYYY”
MaxResult	Integer	Maximum number of items to retrieve. Use 0 to not limit the result

Output:

Name	Type	Description
Result	DataTable	Null when failed, otherwise a DataTable containing a table with the todo list
LastError	String	Error information in case of unsuccessful execution

Auditing:
This method supports object auditing. Auditing is applied as follows:

Audit option	Find
Audit details	-

Corsa72WS4j: The result DataTable will be returned as a byte array containing an XML representation of the DataTable

3.74 GetDSFileExtensions

Returns the extension for files generated by the CORSA/Document server.

Parameters:

None

Output:

Name	Type	Description
Result	Boolean	True when successful, otherwise false
ExtensionPDF	String	Extension for archive (PDF) files generated by the CORSA/Document server
ExtensionOCR	String	Extension for OCR files generated by the CORSA/Document server
ExtensionTMB	String	Extension for thumbnail files generated by the CORSA/Document server
LastError	String	Error information in case of unsuccessful execution

3.75 GetDSPInfo

With the **GetDSPInfo** method you can retrieve the ID and description of a DSP subject.
To retrieve the ID and description pass either the TemplateID or the SubjectID.

Parameters:

Name	Type	Description
TemplateID	String	Corsa DSP-Subject TemplateID
or		
SubjectID	String	Corsa DSP-SubjectID

Output:

Name	Type	Description
Result	Boolean	True when successful, otherwise false
ID	String	ID of the Corsa DSP-Subject
Description	String	Description of the Corsa DSP-Subject
LastError	String	Error information in case of unsuccessful execution

3.76 GetDSPTemplateObjectInfo

Methode GetDSPTemplateObjectInfo is custom made only.
Parameters:

Name	Type	Description
TemplateID	String	DSP template ID

Output:

Name	Type	Description
Result	DataTable	Null when failed, otherwise a DataTable containing a table with the DSP templates
LastError	String	Error information in case of unsuccessful execution

Corsa72WS4j: The result DataTable will be returned as a byte array containing an XML representation of the DataTable.

Based on a template id. this method produces the Contents and Confidentialities of the document template as a DataTable. This DataTable consists of two columns **Id** and **Value**. The **Id** column contains the field name Inhoud (Contents) or Vertrouwld (Confidentiality Id). The **Value** column contains the value of the concerning field. To each confidentiality there is a record with **Id** = **Confidentiality Id** .

3.77 GetDSPTemplatesForWPR

Retrieves a list of available DSP templates for a specific workprocess.

A call to GetDSPTemplatesForWPR is the same as a call to GetDSPTemplatesForWPR2 with the following parameters:

Parameters:

Name	Type	Description
WPRCode	String	Workprocess code
IncludeExpired	Boolean	True

See also
GetDSPTemplatesForWPR2

3.78 GetDSPTemplatesForWPR2

Retrieves a list of available DSP templates for a specific workprocess.

Parameters:

Name	Type	Description
WPRCode	String	Workprocess code
IncludeExpired	Boolean	True to include expired templates, otherwise false.

Output:

Name	Type	Description
Result	DataTable	Null when failed, otherwise a DataTable containing a table with the DSP templates
LastError	String	Error information in case of unsuccessful execution

Corsa72WS4j: The result DataTable will be returned as a byte array containing an XML representation of the DataTable.

(as from update 2015-02 the parameter IncludeExpired value may also be 'false' to return all DSP templates that are **not** expired)
*PG20023

3.79 GetDSPTemplatesWithWord

A call to GetDSPTemplatesWithWord is the same as a call to GetDSPTemplatesWithWord2 with the following parameters:

Parameters:

Name	Type	Description
SearchWords	String	Keywords separated by space
IncludeExpired	Boolean	True

See also
GetDSPTemplatesWithWord2

3.80 GetDSPTemplatesWithWord2

Retrieves a list of available DSP templates containing specific word(s). The word occurs in subject description or DSP template description.

Parameters:

Name	Type	Description
SearchWords	String	Keywords separated by space
IncludeExpired	Boolean	True to include expired templates, otherwise false.

Output:

Name	Type	Description
Result	DataTable	Null when failed, otherwise a DataTable containing a table with the DSP templates
LastError	String	Error information in case of unsuccessful execution

Corsa72WS4j: The result DataTable will be returned as a byte array containing an XML representation of the DataTable.

(as from update 2015-02 the parameter IncludeExpired value may also be 'false' to return all DSP templates that are **not** expired)
*PG20023

3.81 GetEmployeeByEmail

Retrieves the employee or employees with the specified e-mail address. The result can be 0, 1 or more employees.

Parameters:

Name	Type	Description
Email	String	E-mail address

Output:

Name	Type	Description
Result	Boolean	True when successful, otherwise false
Employees	NameValue[]	Employees
LastError	String	Error information in case of unsuccessful execution

3.82 GetEmployeeId

Retrieves an employee id for a specific user id.

Parameters:

Name	Type	Description
UserId	String	CORSA User ID

Output:

Name	Type	Description
Result	Boolean	True when successful, otherwise false
EmployeeId	String	Employee ID for the specified user ID
LastError	String	Error information in case of unsuccessful execution

3.83 GetEmployeeInformation

(Custom-made!)

GetEmployeeInformation returns information about an employee (same as formulas EmployeeName() and EmployeeInfo())

```
public bool GetEmployeeInformation(string EmployeeID, string[] Fields, out NameValue[]
  EmployeeInformation, out String LastError)
```

Parameters:

Name	Type	Default	Description
EmployeeID	String		Employee id.
Fields	String[]		One or more of FullName,Name,Initials,Prefix,Title,Salutation,FormOfAdress,UserId,Engaged,Resigned,OrgUnit,OrgUnitName,Function,Building,Room,Phone,Email,Role,FirstName,Note)

Output:

Name	Type	Default	Description
EmployeeInformation		NameValue[]	Employee details
LastError	String	“”	Error information if the result is False.
Result	Boolean	False	True if successful else false

3.84 GetEmployees

Returns the employees.

Parameters:

Name	Type	Description
Status	GetEmployees_Status	Status of the employees (engaged, resigned or all)

Output:

Name	Type	Description
Result	Data Table	Null when failed, otherwise a DataTable containing a table with the employees
LastError	String	Error information in case of unsuccessful execution

Corsa72WS4j: The result DataTable will be returned as a byte array containing an XML representation of the DataTable.

3.85 GetFavouriteContents

Retrieves the contents (e.g. documents) of a favourites tree node.

Parameters:

Name	Type	Description
ObjectType	String	Corsa object type of the node for which the contents should be retrieved.
NodeID	String	ID of the node for which the contents should be retrieved.
ReturnIDsOnly	Boolean	True to retrieve only Object IDs as result, false to use the ResultItems parameter to determine the returned values.
MaxResult	Integer	Maximum number of items to retrieve. Use 0 to not limit the result.
ResultItems	TypeValue Array	Array of TypeValue items to specify the returned values. Use null to retrieve the values as defined by the query window layout, otherwise specify one or more items to be retrieved. See below for more detailed information.
QueryLayoutId	String	Function ID for (result-)layout defined in Corsa.

To retrieve the contents of a favourites tree node, pass the object type and Id columns of the tree node to the parameters ObjectType and NodeID of this method.

The ResultItems parameter is used to define the desired result. When the value for the ResultItems parameter is null, then the result will contain the values as defined by the query window layout in Corsa. To specify specific result items, add one or more TypeValue items to the array. The Type should be set to the field type, the Value to the field id.

For a list of possible field types and ids use Corsa and go to 'System | Installation | Programs' and execute the program std/w-brubr.r.

When the ReturnIDsOnly parameter is set to true, then the ResultItems parameter will be ignored.

Output:

Name	Type	Description
Result	Boolean	True when successful, otherwise false
FavouritesResult	DataTable	Null when failed, otherwise a DataTable containing a table with the contents
LastError	String	Error information in case of unsuccessful execution

Auditing:
This method supports object auditing. Auditing is applied as follows:

Audit option	Find
Audit details	-

Corsa72WS4j: The result DataTable will be returned as a byte array containing an XML representation of the DataTable

3.86 GetFavourites

Retrieves the nodes of the favourites tree (personal or public).

Parameters:

Name	Type	Description
Type	GetFavourites_type	The type of favourites (personal or public)

Output:

Name	Type	Description
Result	Boolean	True when successful, otherwise false
FavouritesResult	DataTable	Null when failed, otherwise a DataTable containing a table with the favourites tree
LastError	String	Error information in case of unsuccessful execution

This method retrieves all the nodes (including child nodes) of the personal or public favourites tree. To build the first (top) level of the tree, get all the rows where the ParentId column is empty. To build the child nodes of the tree, recursively get all the rows where the ParentId column equals the Id column of the parent. The HasChildren column indicates whether a node has child nodes. The Label column contains the label for a node.

Corsa72WS4j: The result DataTable will be returned as a byte array containing an XML representation of the DataTable

3.87 GetFields

Retrieves a list of available fields.

With the GetFields method either Query Fields (rubrics) or Registration Fields can be retrieved.

Parameters:

Name	Type	Description
gfType	GetFields_Type	Type of the fields to retrieve (gftQueryFields, gftRegistrationFields)
ObjectType	String	CORSA ObjectType for which the field list should be retrieved

Output:

Name	Type	Description
Result	Boolean	True when successful, otherwise false
FieldInfoData	FieldInfo Array	Array of FieldInfo items containing the field information
LastError	String	Error information in case of unsuccessful execution

3.88 GetFieldValues

Retrieves the value for one or more fields and references for a specific object.

Parameters:

Name	Type	Description

ObjectType	String	CORSA ObjectType of the object
ObjectID	String	CORSA ObjectID of the object
Fields	String Array	String Array with fields
References	String Array	String Array with references

For a list of possible values for the Fields parameter select menu option 'System | Installation | Programs' in Corsa and execute ontw/d-dspfields.r Select the concerned ObjectType and Type: 'Registration window fields' in window 'Fields per type'.

For a list of possible values for the References parameter use Corsa and go to 'System | Settings | References'.

Output:

Name	Type	Description
Result	Boolean	True when successful, otherwise false
Values	NameValue Array	Array of NameValue items with fields/references and their values
LastError	String	Error information in case of unsuccessful execution

Auditing:
This method supports object auditing. Auditing is applied as follows:

Audit option	Detail
Audit details	GetWaarden

3.89 GetFileSize

Get the number of bytes in a file in the Upload folder on the server. The client needs to know this to know when to stop downloading.

Parameters:

Name	Type	Description
FileName	String	Filename on the server

Output:

Result	Type	Description
Number of bytes in the file on the server	Long	

3.90 GetFileVersion

Returns an existing file version of a specific object.

The GetFileVersion method retrieves a file version from Corsa.

Parameters:

Name	Type	Description
ObjectType	String	Corsa ObjectType of the object for the file version
ObjectID	String	Corsa ObjectID of the object for the file version
FileType	DSFileType	The DS file type, either Native, Archive or OCRtext
FileVersion	Integer	Version for the file. To retrieve the latest version, specify a value < 1. To retrieve an existing version, specify the existing version number

Output:

Name	Type	Description
Result	Boolean	True when successful, otherwise false
FileBytes	Byte Array	Byte Array of the file data
VersionInfo	VersionInfo	VersionInfo class containing information about the file version
LastError	String	Error information in case of unsuccessful execution

Auditing:
This method supports object auditing. Auditing is applied as follows:

	Native files	Archive files
Audit option	Native file	Archive file
Audit details	-	-

3.91 GetFileVersion2

Returns an existing file version of a specific object. This file will be saved to a location on the server.

Parameters:

Name	Type	Description
ObjectType	String	Corsa ObjectType of the object for the file version

ObjectID	String	Corsa ObjectID of the object for the file version
FileType	DSFileType	The DS file type, either Native, Archive or OCRtext
FileVersion	Integer	Version of the file. To retrieve the latest version, specify a value < 1. To retrieve an existing version, specify the existing version number

Output:

Name	Type	Description
Result	Boolean	True when successful, otherwise false
FileName	String	File will be saved to a location on the server. This is the filename on the server.
LastError	String	Error information in case of unsuccessful execution

Auditing:
This method supports object auditing. Auditing is applied as follows:

Audit option	Detail
Audit details	CheckOutInfo GetCheckOutPath

3.92 GetFileVersionCheckOutInfo

Returns information about the 'check out' and ‘final’ status of a file version for a specific object.

Parameters:

Name	Type	Description
ObjectType	String	Corsa ObjectType of the object
ObjectID	String	Corsa ObjectID of the object

Output:

Name	Type	Description
Result	Boolean	True when successful, otherwise false
VersionStatusInfo	VersionStatusInfo	VersionStatusInfo class containing information about the file status
LastError	String	Error information in case of unsuccessful execution

Auditing:
This method supports object auditing. Auditing is applied as follows:

Audit option	Detail
Audit details	CheckOutInfo GetCheckOutPath

3.93 GetFileVersionList

Gets a list of file versions of a specific object.

Parameters:

Name	Type	Description
ObjectType	String	Object type
ObjectID	String	ObjectID

Output:

Name	Type	Description
Result	DataTable	Null when failed, otherwise a DataTable containing a table with the file version list
LastError	String	Error information in case of unsuccessful execution

Auditing:
This method supports object auditing. Auditing is applied as follows:

Audit option	Detail
Audit details	GetVersion

Corsa72WS4j: The result DataTable will be returned as a byte array containing an XML representation of the DataTable

As from update 2018-02 for releases 2015-1, 2016 and 2017 the fields below are included in the Result's DataTable
"ArcDSCode"
"OcrDSCode"
"TmbDSCode"
"IdxDSCode"

In these fields a numeric value represents the status of the action:
0 - Unknown
1 - Pending
2 - Succeed
3 - Failed
4 - Blocked by rule
99 - Resend (=sent back)

3.94 GetModules

The GetModules method can be used to retrieve a list of available modules. The result is a comma-separated string containing the module characters.

Parameters:

None

Output:

Name	Type	Description
Result	String	Comma-separated string of available modules
LastError	String	Error information in case of unsuccessful execution

Module character	Module name
A	Documents
B	Arch
C	Library
D	Who
E	Case
F	Scan
G	General
H	List
I	Capa
J	WebXs
K	Abis
L	MyCORSA
M	Mail
N	Audiodoc
O	Development
P	Pmes
Q	Deva
R	Lite
S	Dsp
T	Audit

U	DLC
V	Real estate
W	Vartab
X	Xtractor
Y	BPR
Z	Custom-made
1	Export
2	Capture
3	DSP Process
4	Extract Transform Load
5	xView
6	DigSign
7	Task service
8	Procedure model
9	Minutes manager

3.95 GetObjectConfidentialities

Returns the confidentialities of a specific object.

Parameters:

Name	Type	Description
ObjectType	String	Object type
ObjectID	String	Object ID

Output:

Name	Type	Description
Result	Boolean	True when successful, otherwise false.
Confidentialities	NameVakue[]	Confidentialities
LastError	String	Error information in case of unsuccessful execution

3.96 GetObjectGuid

Returns the GUID of an object. This GUID is used by the Office integration of MyCorsa NxT.

Note: this method is supported as from release 2015-1.

Parameters:

Name	Type	Description
ObjectType	String	Corsa ObjectType
ObjectId	String	Corsa ObjectId

Output:

Name	Type	Description
Result	String	GUID when successful, otherwise null or an empty string
LastError	String	Error information in case of unsuccessful execution

3.97 GetObjectKinds

Returns object kinds which can be used for registration and/or requesting data.

With the GetObjectKinds method a list of object kinds which are defined in Corsa can be retrieved.

Parameters:

Name	Type	Description
ObjectType	String	Corsa ObjectType for the object kinds to retrieve
RegistrationOnly	Boolean	True to retrieve object kinds which can be used for registration purporses, false to retrieve all object kinds
DSSupportOnly	Boolean	True to retrieve object kinds which have a device link (and thus can be used to create file versions), false to retrieve all object kinds. This parameter is only effective when RegistrationOnly is true

Output:

Name	Type	Description
Result	IdDescription Array	Null when failed, otherwise an array of IdDescription items containing object kinds
LastError	String	Error information in case of unsuccessful execution

3.98 GetObjectMemo

With GetObjectMemo the memo of an object (document, case, person, organisation, etc.) is retrieved. (As from update 2013-wk35)

Parameters:

Name	Type	Description
ObjectType	String	Object type
ObjectID	String	Object ID

Output:

Name	Type	Description
Result	Boolean	True when successful, otherwise false
MemoText	String	Memo text
SecureMemo	Boolean	True when memo is protected, otherwise false As from update 2017-04 for release 2017 default True
LastError	String	Error information in case of unsuccessful execution

3.99 GetObjectRelationTypes

Returns relation types which can be used for relating objects.

With the GetObjectRelationTypes method a list of relation types which are defined in Corsa can be retrieved.

Parameters:

None

Output:

Name	Type	Description
Result	IdDescription Array	Null when failed, otherwise an array of IdDescription items containing relation types
LastError	String	Error information in case of unsuccessful execution

3.100 GetObjectToObjectRelation

Returns the relation from an object to another object.

With the GetObjectToObjectRelation method a list of relation between objects can be retrieved.

Parameters:

Name	Type	Description
ObjectType1	String	Object type from
ObjectId1	String	ObjectId from
ObjectType2	String	Object type to
ObjectId2	String	ObjectId to

Output:

Name	Type	Description
Result	IdDescription Array	Null when failed, otherwise an array of IdDescription items containing the relation
LastError	String	Error information in case of unsuccessful execution

Auditing:
This method supports object auditing. Auditing is applied as follows:

Audit option	Detail
Audit details	GetRelatieSoort (<ObjectType2>, <ObjectId2>)

3.101 GetODMADocName

The GetODMADocName function can be used to retrieve the document name that can be used to open a file via ODMA.

Parameters:

Name	Type	Description
ObjectType	String	Corsa object type for the object (Usually “S”)
ObjectID	String	Corsa object id for the object
Version	String	Version of the object
Extension	String	Extension of the object

Output:

Name	Type	Description
Result	Byte array	Byte array containing an ODMA Document name string
LastError	String	Error information in case of unsuccessful execution

3.102 GetParamValue

Retrieves the value for a CORSA parameter.

Parameters:

Name	Type	Description
ParamId	String	CORSA parameter ID for which the value should be retrieved

Output:

Name	Type	Description
Result	Boolean	True when successful, otherwise false
ParamValue	String	Value for the specified parameter
LastError	String	Error information in case of unsuccessful execution

3.103 GetPersonsInOrganisation

Returns the persons in an organisation.

Parameters:

Name	Type	Description
ObjectID	String	Object ID of the organisation

Output:

Name	Type	Description
Result	Data Table	Null when failed, otherwise a DataTable containing a table with the persons
LastError	String	Error information in case of unsuccessful execution

Corsa72WS4j: The result DataTable will be returned as a byte array containing an XML representation of the DataTable.

3.104 GetReferences

Retrieves a list of available references for a specific object type.

Parameters:

Name	Type	Description
ObjectType	String	CORSA ObjectType for which the field list should be retrieved

Output:

Name	Type	Description
Result	Boolean	True when successful, otherwise false
ReferenceTable	DataTable	Null when failed, otherwise a DataTable containing a table with the references
LastError	String	Error information in case of unsuccessful execution

Corsa72WS4j: The result DataTable will be returned as a byte array containing an XML representation of the DataTable

3.105 GetReferenceValues

Returns reference values for a specific reference.

Parameters:

Name	Type	Description
ReferenceId	String	Reference Id to retrieve the values for

Output:

Name	Type	Description
Result	Boolean	True when successful, otherwise false.
ReferenceValueTable	DataTable	Null when failed, otherwise a DataTable containing a table with the reference values
LastError	String	Error information in case of unsuccessful execution

Corsa72WS4j: The result DataTable will be returned as a byte array containing an XML representation of the DataTable

3.106 GetSignature

(Custom-made!)

Get the signature of an employee

```
public enum GetSignature_type { gstParaphCopy, gstParaphOriginal, gstSignatureCopy,
gstSignatureOriginal };

public bool GetSignature(String EmployeeID, GetSignature_type SigType, out byte[]
  Signature, out string Extension, out String LastError)
```

Parameters:

Name	Type	Default	Description
EmployeeID	String		EmployeeID
SigType	GetSignature_type	gstSignatureOriginal	Signature type

Output:

Name	Type	Default	Description
Signature	Byte[]	Null	Signature
Extension	String	“”	Signature extension
LastError	String	“”	Error information if the result is False.
Result	Boolean	False	True if successful else false

3.107 GetTableContent

Please contact BCT for more information regarding the use of this method.

3.108 GetUserName

Gets the Corsa user name. When connecting with the network user, this method will return the Corsa user that corresponds to that network user.

Parameters:

None

Output:

Name	Type	Description
Result	String	The Corsa user

3.109 GetWhoData

Retrieves Corsa/Who? address data for a specific object.

Parameters:

Name	Type	Description
ObjectType	String	Corsa ObjectType of the object. Possible values are “P” (person) and “E” (organisation)
ObjectID	String	Corsa ObjectID of the object
AddressType	Who_AddressType	Address type to change (Contact or Private)
DataSetCode	String	A code for a dataset (defined in Corsa). Use an empty string for the default (auto-generated) dataset

Output:

Name	Type	Description
Result	Boolean	True when successful, otherwise false
WhoData	NameValue Array	Array of NameValue items with dataset fields and their values
LastError	String	Error information in case of unsuccessful execution

Auditing:
This method supports object auditing. Auditing is applied as follows:

Audit option	MsWord
Audit details	-

For a list of possible **FieldNames** in the Result use Corsa and go to menu option - **Develop|Development|DataSetFields** and choose type **Person** or - **System|Installation|Programs** and execute the program **std/w-bdatru.r**. The default dataset code is @MLITE. If other data is needed then create a new dataset in Corsa.

3.110 GetWPRList

Retrieves a list of available workprocesses, personal or all.

Parameters:

Name	Type	Description

isPersonal	Boolean	True for personal workprocesses, false for all workprocesses

Output:

Name	Type	Description
Result	DataTable	Null when failed, otherwise a DataTable containing a table with the workprocesses
LastError	String	Error information in case of unsuccessful execution

Corsa72WS4j: The result DataTable will be returned as a byte array containing an XML representation of the Data.

3.111 LastErrorMessage

Returns the last occurred error, if any.

Parameters:

None

Output:

Name	Type	Description
Result	String	An empty String when no error occurred, otherwise a String containing the last occurred error.

3.112 MergeOrganisation

With the MergeOrganisation method the data of two Corsa/Who? organisations can be merged.

Parameters:

Name	Type	Description
ObjectID1	String	Corsa ObjectID of first organisation
ObjectID2	String	Corsa ObjectID of second organisation

The organisation in ObjectID1 will be merged with the organisation in ObjectID2. The merge is identical to a merge through the Corsa user interface. The organisation in ObjectID1 will be deleted, the organisation in ObjectID2 will remain.

Output:

Name	Type	Description

Result	Boolean	True when successful, otherwise false
LastError	String	Error information in case of unsuccessful execution

3.113 MergePerson

With the MergePerson method the data of two Corsa/Who? persons can be merged.

Parameters:

Name	Type	Description
PersData1	String	Data of first person
PersData2	String	Data of second person

Output:

Name	Type	Description
Result	Boolean	True when successful, otherwise false
LastError	String	Error information in case of unsuccessful execution

The **PersData1** and **PersData2** parameters are chr(1) separated strings of person data:

Person_id + chr(1) +
Initials + + chr(1) +
Prefix + chr(1) +
Name + chr(1) +
Street + chr(1) +
Housenr. + chr(1) +
Housenr.annex + chr(1) +
Postal code + chr(1) +
City + chr(1) +
Phone + chr(1) +
[Yes/No]

The last entry is optional and indicates whether existence of the person in Corsa should be checked.

3.114 MergePerson2

With the MergePerson2 method the data of two Corsa/Who? persons can be merged.

Parameters:

Name	Type	Description
ObjectID1	String	Corsa ObjectID of the first person (FROM)
ObjectID2	String	Corsa ObjectID of the second person (TO)

The person in ObjectID1 will be merged with the person in ObjectID2. The merge is identical to a merge through the Corsa user interface. The person in ObjectID1 will be deleted, the person in ObjectID2 will remain.

Output:

Name	Type	Description
Result	Boolean	True when successful, otherwise false
LastError	String	Error information in case of unsuccessful execution

3.115 ModConfidentialities

Add or delete confidentialities for a specific object.

Parameters:

Name	Type	Description
Action	ModConfidentialities_Action	Action to perform (MC_Add, MC_Delete)
ObjectType	String	Corsa ObjectType of the object
ObjectID	String	Corsa ObjectID of the object
Confidentialities	String Array	String Array with Confidentialities

Output:

Name	Type	Description
Result	Boolean	True when successful, otherwise false.
LastError	String	Error information in case of unsuccessful execution

See also
GetConfidentialities

3.116 ModCopyHolder

The ModCopyHolder function can be used to add or modify copyholder data of a document registration.

Parameters:

Name	Type	Description
ObjectID	String	ID of a document registration
RelationType	String	Relation type for the copyholder
RelationID	String	Relation id for the copyholder
MarkAsRead	Boolean	Set the status to “Not Read” or “Read”
SendEmail	Boolean	Send email to copyholder

Output:

Name	Type	Description
Result	Boolean	True when successful otherwise false
LastError	String	Error information in case of unsuccessful execution

The SendEmail parameter will only be used when the module “Mail” is installed in Corsa and configured to use SMTP and the value of Corsa parameter autoemail is set to 1.

3.117 ModObjectRelation

Create or delete a relation between Corsa objects.

With the ModObjectRelation method a relation between two Corsa objects can be created or deleted.

Parameters:

Name	Type	Description
Action	ModObjectRelation_Action	Action to execute (create or delete)
ObjectType_1	String	Corsa ObjectType of the first object
ObjectID_1	String	Corsa ObjectID of the first object
ObjectType_2	String	Corsa ObjectType of the second object
ObjectID_2	String	Corsa ObjectID of the second object
RelationTypeID	String	Relation ID of the relation type to use. See GetObjectRelationTypes for information about retrieving a list of relation types.

		If left empty, a standard relation type will be used, depending on the selected object. Standard relation types can be set using installation parameters (e.g. stdrelps in case of a document). A complete list of installation parameters (stdrel**) can be found in CORSA via System | Installation | Parameters.

Output:

Name	Type	Description
Result	Boolean	True when successful, otherwise false.
LastError	String	Error information in case of unsuccessful execution

See also
ModObjectRelation2

3.118 ModObjectRelation2

Create or delete a relation between Corsa objects.
Note: this method is supported as from release 2014.

With the ModObjectRelation2 method a relation between two Corsa objects can be created or deleted.

Parameters:

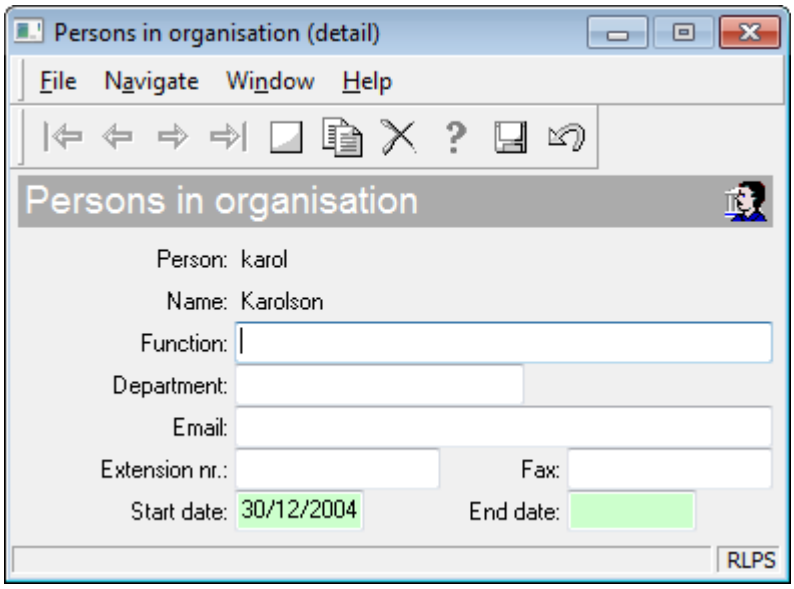
Name	Type	Description
Action	ModObjectRelation_Action	Action to execute (create or delete)
ObjectType_1	String	Corsa ObjectType of the first object
ObjectID_1	String	Corsa ObjectID of the first object
ObjectType_2	String	Corsa ObjectType of the second object
ObjectID_2	String	Corsa ObjectID of the second object
RelationTypeID	String	Relation ID of the relation type to use. See GetObjectRelationTypes for information about retrieving a list of relation types.

		If left empty, a standard relation type will be used, depending on the selected object. Standard relation types can be set using installation parameters (e.g. stdrelps in case of a document). A complete list of installation parameters (stdrel**) can be found in CORSA via System | Installation | Parameters.
Options	NameValue[]	Additional options. Default null.

Relations between two persons are supported only as from release 2014.

Use the Options parameter to pass additional fields by adding one or more NameValues.

Person <-> Organisation
When creating a relation between a person (ObjectType = “P”) and an organisation (ObjectType = ”E”), the fields of the **Persons in organisation** window can be used:



The fields 'Function', 'Department', 'Email', 'Phone' (= used for **Extension nr.**) and 'Fax' are string fields. The date field format of the fields 'DateFrom' (= used for **Start date**) and 'DateUntil' (= used for **End date**) is DD/MM/YYYY.

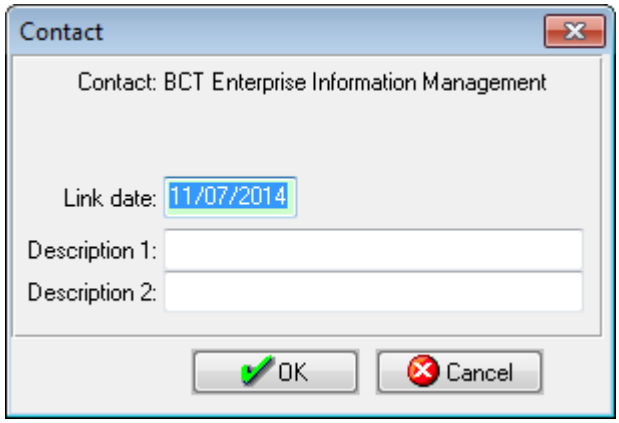
Options parameter example for **Person <-> Organisation**

```
<bct:Options>
  <bct:NameValue>
    <bct:Name>DateFrom</bct:Name>
    <bct:Value>23/11/2018<</bct:Value>
  </bct:NameValue>
  <bct:NameValue>
    <bct:Name>DateUntil</bct:Name>
```

```
        <bct:Value>31/12/2017</bct:Value>
      </bct:NameValue>
      <bct:NameValue>
        <bct:Name>Department</bct:Name>
        <bct:Value>Development</bct:Value>
      </bct:NameValue>
      <bct:NameValue>
        <bct:Name>Email</bct:Name>
        <bct:Value>abc@bct.nl</bct:Value>
      </bct:NameValue>
      <bct:NameValue>
        <bct:Name>Fax</bct:Name>
        <bct:Value>-</bct:Value>
      </bct:NameValue>
      <bct:NameValue>
        <bct:Name>Function</bct:Name>
        <bct:Value>Sr. Developer</bct:Value>
      </bct:NameValue>
      <bct:NameValue>
        <bct:Name>Phone</bct:Name>
        <bct:Value>324</bct:Value>
      </bct:NameValue>
    </bct:Options>
```

Organisation <-> Organisation

When creating a link between two organizations (ObjectType = “E”), the fields below can be used:



The date field format of the field 'DateRelation' (= used for **Link date**)is DD/MM/YYYY.
The field 'Desc1' (= used for **Description 1**) and 'Desc2' (= used for **Description 2**) are string fields.

Options parameter example for **Organisation <-> Organisation**

```
    <bct:Options>
      <bct:NameValue>
        <bct:Name>DateRelation</bct:Name>
        <bct:Value>23/11/2018</bct:Value>
```

```
        </bct:NameValue>
        <bct:NameValue>
          <bct:Name>Desc1</bct:Name>
          <bct:Value>Test description 1</bct:Value>
        </bct:NameValue>
        <bct:NameValue>
          <bct:Name>Desc2</bct:Name>
          <bct:Value>Test description 2</bct:Value>
        </bct:NameValue>
      </bct:Options>
```

Output:

Name	Type	Description
Result	Boolean	True when successful, otherwise false.
LastError	String	Error information in case of unsuccessful execution

3.119 ModPassword

Change the password for the connected user.

With the ModPassword method the Corsa password for the connected user can be modified.

Parameters:

Name	Type	Description
NewPassword	String	New Corsa password for the connected user

Output:

Name	Type	Description
Result	Boolean	True when successful, otherwise false
LastError	String	Error information in case of unsuccessful execution

3.120 ModRECity

Modifies the name of an existing Real Estate city in table Cities [VPL] .

Parameters:

Name	Type	Description
OldCity	String	Name of the existing real estate city in Corsa
NewCity	String	Name of the new real estate city

Output:

Name	Type	Description
Result	Boolean	True when successful, otherwise false
LastError	String	Error information in case of unsuccessful execution

3.121 ModREStreet

Modifies the name of an existing Real Estate street in table Streets [VSTR] of an existing Real Estate city in table Cities [VPL] .

Parameters:

Name	Type	Description
City	String	Name of the existing real estate city in Corsa
OldStreet	String	Name of the existing real estate street
NewStreet	String	Name of the new real estate street

Output:

Name	Type	Description
Result	Boolean	True when successful, otherwise false
LastError	String	Error information in case of unsuccessful execution

3.122 ProcessStufDCRResult

For internal use (BCT) only!

This method processes the result of a StUF-DCR request.

Parameters:

Name	Type	Description
ObjectID	String	Object ID
JobID	String	Job ID
FieldValues	NameValue Array	Array of NameValue items with fields and their values With the DCR-request a set of fields and values is passed. When the ProcessStufDCRRequest is called, these fields/values may have been changed since making the DCR-request and should be returned in this parameter.
JobStatus	String	Job status
JobMessage	String	Job (error) message

Output:

Name	Type	Description
Result	Boolean	True when successful, otherwise false
LastError	String	Error information in case of unsuccessful execution

3.123 QueryExecute

Executes a query and returns the results.

The QueryExecute method can be used to search objects available in the Corsa database.

Parameters:

Name	Type	Description
ObjectType	String	Corsa ObjectType for the objects to search
QueryItems	QueryItem Array	Array of QueryItem items which define the query to execute. See below for more detailed information
SkipEmptyConditions	Boolean	True to skip empty query conditions, otherwise false. Default true
SearchByQueryOrder	Boolean	True to execute the query according to the query order, otherwise false. Default false
SearchSubFolders	Boolean	Default false, only used when ObjectType is D (= folder). True to additionally return subfolders of the folders found (1st level down only). Set parameter SearchRecursive to True to return all sub folder levels (including subfolders of subfolders) .

SearchRelated	Boolean	Default false. True to additionally return related objects (not recursively). To recursively return all related objects (related objects of related objects), set parameter SearchRecursive to True.
SearchRecursive	Boolean	Default false. True to search through related objects recursively.
ReturnIDsOnly	Boolean	True to retrieve only ObjectID's as result, false to use the ResultItems parameter to determine the returned values
MaxResult	Integer	Maximum number of items to retrieve. Use 0 to not limit the result
ResultItems	TypeValue Array	Array of TypeValue items to specify the returned values. Use null to retrieve the values as defined by the query window layout, otherwise specify one or more items to be retrieved. See below for more detailed information
QueryLayoutId	String	Function code for (result-)layout defined in Corsa.

The QueryItems parameter is used to define the query to execute. For a list of possible rubrics use Corsa and go to 'System | Installation | Programs' and execute the program std/w-brubr.r.

The QueryItem object has the following properties:

Name	Type	Description
RubricType	QueryRubricType	The rubric type
RubricID	String	ID of the rubric
Condition	QueryCondition	Specify the condition (And,Or, Not)
Operator	QueryOperator	Specify the operation
Value	String	The value to search for

Parameter SearchSubFolders had no effect till update 2014-05.

The ResultItems parameter is used to define the desired result. When the value for the ResultItems parameter is null, then the result will contain the values as defined by the query window layout in Corsa. To specify specific result items, add one or more TypeValue items to the array. The Type should be set to the field type the Value to the field id. All values returned by the method are truncated to 500 characters by default. In case a value is longer than 500 characters and the complete value is needed, use method **GetFieldValues**.

For a list of possible field types and id's use Corsa and go to 'System | Installation | Programs' and execute the program std/w-brubr.r.

When the ReturnIDsOnly parameter is set to true, then the ResultItems parameter will be ignored.

Output:

Name	Type	Description
Result	DataTable	Null when failed, otherwise a DataTable containing a table with the objects found by the query
LastError	String	Error information in case of unsuccessful execution

Auditing:
This method supports object auditing. Auditing is applied as follows:

Audit option	Find
Audit details	-

Corsa72WS4j: The result DataTable will be returned as a byte array containing an XML representation of the DataTable

 A call to QueryExecute is the same as a call to QueryExecute2 with parameter SearchMostRecent = false.

3.124 QueryExecute2

Executes a query and returns the results.

The QueryExecute method can be used to search objects available in the Corsa database.

Parameters:

Name	Type	Description
ObjectType	String	Corsa ObjectType for the objects to search
QueryItems	QueryItem Array	Array of QueryItem items which define the query to execute. See below for more detailed information
SkipEmptyConditions	Boolean	True to skip empty query conditions, otherwise false. Default true
SearchByQueryOrder	Boolean	True to execute the query according to the query order, otherwise false. Default false
SearchSubFolders	Boolean	Default false, only used when ObjectType is D (= folder). True to additionally return subfolders of the folders found (1st level down only). Set parameter SearchRecursive to True to return all sub folder levels (including subfolders of subfolders) .

Name	Type	Description
SearchRelated	Boolean	Default false. True to additionally return related objects (not recursively). To recursively return all related objects (related objects of related objects), set parameter SearchRecursive to True.
SearchRecursive	Boolean	Default false.True to search through related object recursively.
ReturnIDsOnly	Boolean	True to retrieve only ObjectID's as result, false to use the ResultItems parameter to determine the returned values
SearchMostRecent	Boolean	Default false, only used when ObjectType is S. True to retrieve more recent objects first. Useful if the query has many hits and the maximum number of hits will be reached.
MaxResult	Integer	Maximum number of items to retrieve. Use 0 to not limit the result.
ResultItems	TypeValue Array	Array of TypeValue items to specify the returned values. Use null to retrieve the values as defined by the query window layout, otherwise specify one or more items to be retrieved. See below for more detailed information
QueryLayoutId	String	Function code for (result-)layout defined in Corsa.

The QueryItems parameter is used to define the query to execute. For a list of possible rubrics use Corsa and go to 'System | Installation | Programs' and execute the program std/w-brubr.r.

The QueryItem object has the following properties:

Name	Type	Description
RubricType	QueryRubricType	The rubric type
RubricID	String	ID of the rubric
Condition	QueryCondition	Specify the condition (And,Or, Not)
Operator	QueryOperator	Specify the operation
Value	String	The value to search for

Parameter SearchSubFolders had no effect till update 2014-05.

The ResultItems parameter is used to define the desired result. When the value for the ResultItems parameter is null, then the result will contain the values as defined by the query window layout in Corsa. To specify specific result items, add one or more TypeValue items to the array. The Type should be set to the field type the Value to the field id. All values returned by the method are truncated to 500 characters by default. In case a value is longer than 500 characters and the complete value is needed, use method **GetFieldValues**.

For a list of possible field types and id's use Corsa and go to 'System | Installation | Programs' and execute the program std/w-brubr.r.

When the ReturnIDsOnly parameter is set to true, then the ResultItems parameter will be ignored.

Output:

Name	Type	Description
Result	DataTable	Null when failed, otherwise a DataTable containing a table with the objects found by the query
LastError	String	Error information in case of unsuccessful execution

Auditing:
This method supports object auditing. Auditing is applied as follows:

Audit option	Find
Audit details	-

Corsa72WS4j: The result DataTable will be returned as a byte array containing an XML representation of the DataTable

3.125 SetFinalStatus

Set the 'final' status for a specific object.

Parameters:

Name	Type	Description
ObjectType	String	Corsa ObjectType of the object
ObjectID	String	Corsa ObjectID of the object
FinalStatus	Boolean	Boolean indicating the value to set

Output:

Name	Type	Description
Result	Boolean	True when successful, otherwise false
LastError	String	Error information in case of unsuccessful execution

3.126 SetLicenseKey

Sets the license key to use.

If the license key is not set to a valid license key and the 'Named User' licensing model is activated in the Corsa environment then the Connect function will return an error. You will then not be able to connect to the CORSA71WS.

To obtain a license key please contact BCT.

Parameters:

Name	Type	Description
LicenseKey	String	License key to use

3.127 SetObjectMemo

SetObjectMemo sets the memo of an object (document, case, person, organisation, etc.).
(As from update 2013-wk35)

If the memo is protected (default from release 2017 onwards) the memo text is added.
Otherwise the memo is overwritten.
Use GetObjectMemo to get the current memo text and to check whether the memo is protected or not.

Parameters:

Name	Type	Description
ObjectType	String	Object type
ObjectID	String	Object ID
MemoText	String	Memo text

Output:

Name	Type	Description
Result	Boolean	True when successful, otherwise false
SecureMemo	Boolean	As from update 2017-04 for release 2017 default True
LastError	String	Error information in case of unsuccessful execution

3.128 SetReferences

Add or delete one or more reference values for a specific object.

Parameters:

Name	Type	Description
Action	SetReference_Action	Action to execute (add or delete)
ObjectType	String	Corsa ObjectType of the object
ObjectID	String	Corsa ObjectID of the object
ReferenceValues	NameValue Array	Array of NameValue items with references and their values

Output:

Name	Type	Description
Result	Boolean	True when successful, otherwise false
ObjectList	TypeValue Array	Array of TypeValue items which can contain zero, one or more ObjectType and ObjectID values if due to adding a reference value a new object was created in Corsa
LastError	String	Error information in case of unsuccessful execution

3.129 SetWhoStUFAddress

 For internal use by BCT only!

SetWhoStUFAddress processes (StUF) changes to an address. This method changes the address itself as well as address related (StUF) references.

Parameters:

Name	Type	Description
ObjectType	String	Corsa ObjectType of the object
ObjectID	String	Corsa ObjectID of the object
AddressType	Who_AddressType	Address type to process (Contact or Private)
FieldValues	NameValue Array	Array of NameValue items with fields and their values

These fields can be passed as FieldValues:

- "begindatumRelatie"
- "einddatumRelatie"
- "woonplaatsnaam"

- "postcode"
- "straatnaam"
- "huisnummer"
- "huisletter"
- "huisnummertoevoeging"
- "locatieomschrijving"
- "postbusnummer"
- "adresBuitenland1"
- "adresBuitenland2"
- "adresBuitenland3"
- "landcode"
- "geheimAdres"

Dates must be passed in format 99/99/9999.

Output:

Name	Type	Description
Result	Boolean	True when successful, otherwise false
LastError	String	Error information in case of unsuccessful execution

 For internal use by BCT only!

3.130 VarTabGet

The method VarTabGet can be used to retrieve a table for a specific object.

Parameters:

Name	Type	Description
ObjectType	String	Corsa object type for the object
ObjectId	String	Corsa object id for the object
TableId	String	TableId for the variable table
ScreenId	Integer	Id of a variable window (default -1)

Output:

Name	Type	Description
TableXML	Byte Array	Byte array containing XML String with the table definition and data
NumLines	Integer	The number of rows in the table
LockStamp	String	Unique 'timestamp' for the table

LastError	String	Error information in case of unsuccessful execution

Auditing:
This method supports object auditing. Auditing is applied as follows:

Audit option	Detail
Audit details	AsVartabGet (<TableId>, <ScreenId>)

The ScreenId parameter can be set to -1 or a specific Corsa variable window id. Based on the variable window the definition for the variable table can differ.

3.131 VarTabSet

The method VarTabSet can be used to modify (save) a table for a specific object. Only tables retrieved using VarTabGet can be used.

Parameters:

Name	Type	Description
TableXML	Byte Array	Byte array containing XML String with the table definition and data

Output:

Name	Type	Description
LineId	String	Reserved for future usage
ColumnId	String	Reserved for future usage
LastError	String	Error information in case of unsuccessful execution

Usage
Basically the two API calls must be combined to be able to retrieve and update variable tables. The basic usage is as follows:

1. Retrieve a TableXML using VarTabGet. The XML which is returned contains both the table definition and data (if available).
2. For displaying the table in for example a grid the table definition can be used. Among other information the definition contains ColumnName, ColumnLabel and Visibility information.
3. For updating the table the original XML must be edited and then saved using VarTabSet
   a. **Changing a row**
      - Set the value for the 'UpdateMode' column to 'U'
      - Set the new values for the columns. Only columns with 'Editable=True' can be modified. The new values must be correct values.
   b. **Adding a row**
      - Set the value for the 'UpdateMode' column to 'A'
      - Add a new <row> element

- Specify the values for the columns. Only columns with 'Editable=True' have to be provided. The new values must be correct values.

c. **Removing a row**

- Set the value for the 'UpdateMode' column to 'D'

After using VarTabSet a new XML must be retrieved using VarTabGet if any further changes are necessary because of the unique 'LockStamp' in the XML.

3.132 VerifyConfidentialities

Verifies read or write access based on confidentialities and permits for a list of objects.

The VerifyConfidentialities method can be used to check whether a user has read or write acces for a list of objects.

Parameters:

Name	Type	Description
ObjectType	String	Corsa ObjectType of the objects to verify
ObjectIdList	String Array	Array with Corsa ObjectID's to verify
VerifyMode	ConfVerifyMode	Access level to verify, either ReadAccess or WriteAccess

Output:

Name	Type	Description
Result	Boolean	True when successful, otherwise false
FilteredObjectIdList	String Array	String Array with Corsa ObjectID's for which access is allowed
LastError	String	Error information in case of unsuccessful execution

 As from update 2014-04 verification of permits is added to the VerifyConfidentialities method.
*HD 039691

3.133 WhoRemoval

Changes Corsa/Who? address data for a specific object (removal).
Parameters:

Name	Type	Description
ObjectType	String	Corsa ObjectType of the object. Possible values are "P" (person) and "E" (organisation).
ObjectID	String	Corsa ObjectID of the object
AddressType	Who_AddressType	Address type to change (Contact or Private)

DateBegin	String	Start date for the address. Format is “DD/MM/YYYY”
DateEnd	String	End date for the address. Format is “DD/MM/YYYY”
Country	String	Country for the address
PostcalCode	String	Postcal code for the address
HouseNr	String	House number for the address
HouseNrAddition	String	House number addition for the address
City	String	City for the address
Street	String	Street for the address
District	String	District for the address

Output:

Name	Type	Description
Result	Boolean	True when successful, otherwise false
LastError	String	Error information in case of unsuccessful execution

4. Attachments

4.1 Change history

Version	Date	Description
1.0.59	24-01-2018	Added element 'document name' (Docname) to output (GetFileVersionList.xml) of GetFileVersionList.
1.0.58	06-11-2017	Added ConnectAs
1.0.57	04-04-2017	Added CreateMetaAgendaItem and GenerateID
1.0.56	14-03-2017	Updated CreateFileVersion and CreateFileVersion2
1.0.55	28-02-2016	Added CreateMetaFolder6
1.0."55"	14-09-2016	Added methods GetContracts and DownloadContractImage.
1.0.54	15-06-2016	Updated CreateFileVersion and CreateFileVersion2 Added a session check to CreateFileVersion2, AppendChunk and AppendChunk2
1.0.53	18-05-2016	Updated AddRECity and AddREStreet,
1.0.52	27-01-2016	Added CheckFileHashSHA256
1.0.51	08-01-2016	Added ProcessStufDCRResult
1.0.50	21-09-2015	Added MergeOrganisation and SetWhoStUFAddress Added Change history as attachment
1.0.49	20-05-2015	Added GetObjectGuid
1.0.48	21-04-2015	The check of entered **Characteristic formats** and **Corsa fields** is expanded in methods to create an object registration like **CreateMetaDocument**
1.0.47	19-03-2015	New settings available for the web.config installation file to activate event log categories
1.0.46	06-03-2015	Added GetEmployeeByEmail
1.0.45	27-02-2015	Added GetUserName, GetObjectConfidentialities, GetPersonsInOrganisation and GetEmployees
1.0.44	09-02-2015	Changed CreateMetaDocument4 for parameter ststds=1
1.0.43	15-01-2015	Added GetDSPTemplatesWithWord2 and GetDSPTemplatesForWPR2
1.0.42	07-01-2015	Added DeleteObject3
1.0.41	14-10-2014	Added return value length info to QueryExecute and QueryExecute2

Version	Date	Description
1.0.40	09-07-2014	Added ModObjectRelation2 and Added DeleteObject2
1.0.39	12-05-2014	Modified CreateMetaFolder(1-5), CreateMetaOrganisation(1-2) and CreateMetaPerson(1-2) Improved QueryExecute and QueryExecute2 Added explanation about differences between these and other multi version methods
1.0.38	28-04-2014	Added verification of permits to VerifyConfidentialities
1.0.37	03-03-2014	Added CreateMetaRealEstate, AddRECity, AddREStreet, ModRECity and ModREStreet CaseConnectDocumentType is now available generally (originally custom-made) * [illegible] * [illegible]
1.0.36	17-12-2013	Added AppendChunk2
1.0.35	08-11-2013	Added 3 checks to CreateMetaDocument4
1.0.34	27-09-2013	Added CaseGetInfo and CaseGetStepInfo
1.0.33	30-08-2013	Added CreateMetaDocument4, CreateMetaFolder5, CreateMetaPerson2 and CreateMetaOrganisation2 Added GetObjectMemo and SetObjectMemo
1.0.32	28-05-2013	Added GetDSPInfo
1.0.31	29-03-2013	Added MergePerson2
1.0.30	26-03-2013	Auditing is added to several methods
1.0.29	05-03-2013	New look Added (Custom-made) GetDSPTemplateObjectInfo
1.0.28	17 September 2012	Added (Custom-made) GetSignature Added (Custom-made) CaseConnectDocumentType Added (Custom-made) GetEmployeeInformation
1.0.27	10 May 2012	Added ChangeObjectKind
1.0.26	30 March 2012	Added CaseChangeDocCustomer Added GetObjectToObjectRelation
1.0.25	3 February 2012	Changed CreateMetaDocument2 Added CreateMetaDocument3 Added CreateMetaFolder4
1.0.24	13 January 2012	Added GetFileVersionList
1.0.23	15 December 2011	Added AddPermit Added DeletePermit
1.0.22	18 July 2011	Added GetFavourites Added GetFavouriteContents
1.0.21	8 June 2011	Added CreateMetaDocument2

Version	Date	Description
		Added CreateMetaFolder3
1.0.20	23 May 2011	Added VarTabGet Added VarTabSet Changed Connect
1.0.19	21 April 2011	Added GetDSFileExtensions
1.0.18	9 February 2011	Added CreateFileVersion2 Added GetFileVersion2 Added GetFileSize Added AppendChunk Added DownloadChunk Added CheckFileHash Added GetChunkSize Added GetDSPTemplatesWithWord Added GetDSPTemplatesForWPR Added GetWPRList Added GetCopyHolderRelationTypes Added ModCopyHolder Added DeleteCopyHolder Added GetCopyHolders Added GetODMADocName Added GetTableContent Added AddSenderAddr Added DelSenderAddr Added CaseGetDocumentTypes Added CaseGetProcessList Added CaseStartProd Added CaseGetReferences Added GetModules Added MergePerson
1.0.17	12 October 2010	Added a additional info for ModObjectRelation
1.0.16	27 April 2010	Added CreateMetaFolder2
1.0.15	15 February 2010	Added ModConfidentialities Added GetParamValue Added GetReferences Added GetFields Added GetEmployeeId
1.0.14	24 December 2009	Added QueryExecute2 Added GetDocTodoList2 Added GetCaseTodoList2 Added GetFieldValues
1.0.13	20 November 2009	Added Corsa72WS4j information
1.0.12	5 November 2009	Added CaseEvaluate Added SetReferences

Version	Date	Description
		Added WhoRemoval Added GetWhoData
1.0.11	12 August 2009	Added VerifyConfidentialities
1.0.10	02 July 2009	Added GetFileVersionCheckOutInfo Added FileVersionCheckIn Added FileVersionCheckOut Added SetFinalStatus Added GetReferenceValues
1.0.09	25 June 2009	Updated documentation with new features and Corsa72WS information
1.0.08	25 August 2008	Added several methods for Corsa/Case
1.0.07	15 April 2008	Changed 7.1 to 7.2 Added SetLicenseKey method and additional info Added GetDocTodoList Added GetCaseTodoList
1.0.06	27 November 2007	Added additional info for Connect and Disconnect method
1.0.05	8 November 2007	Added ExportObjectListSetStatus
1.0.04	6 November 2007	Added ExportGetObjectList
1.0.03	22 August 2007	Added CreateMetaOrganisation and CreateMetaPerson
1.0.02	17 July 2007	Added DeleteFileVersion and DeleteObject
1.0.01	6 July 2007	Added additional info about session state
1.0.00	7 May 2007	First version of the Corsa 7.1 Web Service description

BCT Enterprise Information Management

BCT, trade name of BCT BV P.O. box 59, 6130 AB Sittard, Hub Dassenplein 3, 6131 LB Sittard
T +31 (0)46 442 45 45, F +31 (0)46 442 47 30, E info@bct.nl, I www.bct.nl, Chamber of Commerce: 14043652
Servicedesk: T +31 (0)46 442 50 20, F +31 (0)46 442 59 79, E servicedesk@bct.nl