BCC Architect

BCC Architect Developer Guide

Canada

Version June 2019

201903260331

Contents

Welcome to BCC Architect CA

Welcome to BCC Architect for Canada.

This software offers large-scale integrations greater speed and flexibility for address cleansing and postal presorting solutions.

Installing BCC Architect

BCC Architect is installed using an installer wizard. For help with installation, please see the Installation Guide.

Product Updates

Every two months, you will receive a new issue of BCC Architect. To learn more about new features, refer to the Release Notes made available with the release of each issue.

About this Guide

The Developer Guide is a comprehensive resource for developers using the BCC Architect API.

This manual describes the functions available through BCC Architect and serves as your primary API reference guide.

The first section of this guide contains a high-level overview of how BCC Architect works for some of the common tasks for which it is used.

The latter portion of this guide includes the complete API reference for COM,.NET, and Web Services.

Additional Resources

The BCC Architect installer includes the Developer Center, which includes sample projects that demonstrate how to use each object with a variety of development technologies.

Available Technologies

BCC Architect supports integration and development with technologies including:

- COM
- DLL
- .NET
- Web Services

Processing Modes

BCC Architect Server

The BCC Architect Server is a Win32 application that currently accepts single address verification requests via a TCP/IP connection for the following countries: United States, Canada, Australia, and United Kingdom (with the purchase of these countries).

The clients of the server are very simple and can exist on any system. The BCC Architect installer can install sample client code in the following programming languages:

- JAVA
- C/C++
- ASP

See the Sample Code section of the Developer Center. One of the sample projects is a Visual C++ application that will check addresses from any of the countries listed above.

By making the BCC Architect Server client very simple, you can have a Unix Web Server that contains a JAVA applet, which connects to the BCC Architect Server to process an address correction request.

BCC Architect Server Request Scheme Format

For a detailed description of the BCC Architect Server requests, please refer to the Client/Server Implementation Guide in the Documentation section of the Developer Center.

Using BCC Architect CA for Common Tasks

BCC Architect has capabilities that allow you to cleanse address records, update moved addresses, presort mailings, and integrate with other environments. Depending on your licensing agreement for BCC Architect, you can use BCC Architect for:

- Correcting Single Addresses in Real-Time
- Correcting Batch Addresses Asynchronously
- Presorting Postal Mailings
- Creating Postal Reports

Initiating BCC Architect

Your registration keys are entered when you install BCC Architect. Whenever you use the BCC Architect API, the registration keys are checked automatically. If you try to create an object that is not part of your license agreement, you will receive an error notifying you that your registration key is invalid, expired, or missing.

Correcting Single Addresses in Real-Time

You can use BCC Architect to correct individual addresses on a transactional basis, in real time. This address correction uses the BCC Software SERP Certified address correction engine.

The general process is to:

1. Create the object.
2. Set the address and properties.
3. Check the address.
4. Retrieve the corrected address record.

Interfaces for Correcting Single Addresses

You can use any of the available interfaces to correct single addresses.

Please see the Developer Center for additional examples and sample code.

.NET

1. Create a POSTALCODEAssembly object.
2. Set the address and configuration properties.
3. Call CheckAddress to process the address.
4. Retrieve the corrected and certified address from the POSTALCODEAssembly object.

See the .NET POSTALCODEAssembly reference for more information about POSTALCODEAssembly.

COM

1. Create a PostalCodeTask COM object.
2. Set the address and configuration properties.
3. Call CheckAddress to process the address.
4. Retrieve the corrected and certified address from the PostalCodeTask object.

See the COM PostalCodeTask reference for more information about PostalCodeTask.

Web Services

1. A POSTALCODEService reference is created, then a client-side web service interface object is created.
2. The address and configuration properties are set, processing is done and the processed address is retrieved from the web service.

See the Web Services POSTALCODEService reference for more information about POSTALCODEService .

Correcting Batch Addresses Asynchronously

You can use BCC Architect to correct batches of addresses on a scheduled basis. This address correction uses the BCC Architect SERP Certified address correction engine.

Batch address correction works by packaging multiple addresses and passing them along to the processing routines on the local machine or on a semi-dedicated server. After the processing of address correction is done, reports can be printed for use with a mailing.

The general process is to:

1. Create the object.
2. Set the address and properties.
3. Create objects for each address, and combine into an address block.
4. Process the address block.
5. Retrieve the corrected address records.

Interfaces for Correcting Batch Addresses

You can use .NET or COM interfaces to correct batch addresses.

Please see the Developer Center for additional examples and sample code.

.NET

1. A BATCHAssembly object is created.
2. The address and configuration properties are set.
3. CAAddressRecord objects are created for each address and combined into a CAAddressRecordBlock object.
4. The CAAddressRecordBlock object is processed using the BATCHAssembly object.

5. Use the resulting CAAddressRecordBlock object and its contained CAAddressRecord objects to retrieve your results.

See the .NET BATCHAssembly reference for more information about BATCHAssembly.

COM

1. A BatchTask COM object is created.
2. The address and configuration properties are set.
3. The processing is done.
4. The corrected and certified addresses are retrieved from the COM object.
5. If selected, the reports can be previewed or printed.

See the COM BatchTask reference for more information about BatchTask.

Sorting Postal Mailings

You can use BCC Architect to sort and prepare your address records for a Canada Post mailing.

The general process is to:

1. Create the object.
2. Set the sorting and configuration properties.
3. Create objects for each address, and combine into an address block.
4. Submit the blocks for SORT processing.
5. Retrieve the sorted address block.
6. Create or view reports, if needed.

Interfaces for Presorting Postal Mailings

You can use .NET or COM interfaces to presort postal mailings.

Please see the Developer Center for additional examples and sample code.

.NET

1. A SORTAssembly object is created.
2. The sorting and general configuration properties are set.
3. CAAddressRecord objects are created for each address within the sort and are added to CAAddressRecordBlock objects.
4. Each CAAddressRecordBlock is submitted for sorting using the SORTAssembly object.
5. Once sorting is complete CAAddressRecordBlock objects are retrieve from the SORTAssembly object, each containing a portion of the sort results.
6. Reports can then be printed or previewed.

See the .NET SORTAssembly reference for more information about SORTAssembly.

COM

1. A CASortTask COM object is created.
2. The address and configuration properties are set.
3. The processing is done.
4. The corrected and sorted addresses are retrieved from the COM object.
5. If selected, the reports can be previewed or printed.

See the COM CASortTask reference for more information about CASortTask .

Creating Postal Reports

When you have completed a postal sort, you can use BCC Architect to generate postal reports.

Interfaces for Creating Postal Reports

You can use the COM interface to create postal reports.

See the COM CASOAReportTask reference for more information about CASOAReportTask.

COM API Reference

CONTENTS

The COM Object Factory

Configuring Object Factory Tasks

Creating Task interface objects in BCC Architect begins with the MRTKObjFactory object. All BCC Architect Task objects should be created through the MRTKObjFactory. The MRTKObjFactory performs the proper initialization of interface objects and handles such features as "AutoUpdate." The AutoUpdate feature automatically updates clients to make sure they have the most current Task interfaces and data files. Future releases of BCC Architect will include more advanced updating features, which you can take advantage of through the MRTKObjFactory object. In addition, any future Task interfaces will be exposed via the MRTKObjFactory. The BCC Architect object factory is contained in the MRTKFact.dll.

BCC Architect CA provides a simple and consistent interface for postal mail sorting and address cleansing. Each step of the mailing process is encapsulated in a Task object. The following objects are currently provided: CAPostalCodeTask, CABatchTask, CASOAReportTask, CASortTask.

Task objects require the following paradigm to be used correctly:

1. Create an object through the MRTK object Factory.
2. Once an object has been created, call its PrepareTask function.
3. Now set the Task object's properties through calls to SetProperty. See the tables at the end of this document for descriptions of the available properties for each Task object.
4. After setting the relevant properties for a Task, call ValidateProperties.
5. Perform the work of a particular task. If you were using the CASortTask, for instance, you would call DoSort to perform a CA mail sort.
6. Call EndTask once you have finished using a Task object.

For a detailed description of each Task object, we recommended that you read the introduction to each object to find the specific function that it performs. The following is a brief description of each Task's main function:

CAPostalCodeTask – Single address correction. This is ideal for web pages, order entry screens or customer maintenance screens. The object could be used in any environment that needs single record correction. Can also be used for batch address correction.

CABatchTask – Batch address certification. This Task is used to certify a set of records. It can be set up to run through your database at a specified time and clean all the uncoded or newly added records.

CASOAReportTask – Prints the Statement of Accuracy report.

CASortTask– Performs a Canada Post mail sort on your mailing list. This Task will perform mail sorts based on the parameters specified within the Presort Wizard.

Object Factory Functions

CreateObject

Create a MRTK object.

Syntax

VB:

```
CreateObject(eMRTKObject)
```

C++:

```
IDispatch CreateObject(tagMRTKOBJECTS eMRTKObject)
```

Arguments

VB:

eMRTKObject as tagMRTKOBJECTS

eMRTKObject is the type of object to create

C++:

eMRTKObject

eMRTKObject is the type of object to create

Returns

Newly created interface of type eMRTKObject, otherwise nothing

Notes

It is very important that all interfaces are created using this method for a few reasons. The first and most important is that the CreateObject function is responsible for running the “AutoUpdate” feature of MRTK. If CreateObject is not used to initialize an interface, then the AutoUpdate feature must be implemented by calling IsUpdateAvailable, and if IsUpdateAvailable returns true, then calling RunUpdate.

VB Example

```
Dim obj As CAPostalCodeTask
Dim MRTKFactory As New MRTKFACTORYLib.MRTKObjFactory
Set obj = _
MRTKFactory.CreateObject(ENTERPRISE_CAPOSTCODETASK)
' obj is a newly created MRTK object
…
```

See also

See the tagMRTKOBJECTS Properties table for a list of possible objects

MRTKObjFactory.IsObjectExpired

Checks if the MRTK object has expired

Syntax

VB:

```
IsObjectExpired(eMRTKObject)
```

C++:

```
long IsObjectExpired(tagMRTKOBJECTS eMRTKObject)
```

Arguments

VB:

eMRTKObject as TagMRTKOBJECTS

eMRTKObject is the type of object to check for expiration

C++:

eMRTKObject

eMRTKObject is the type of object to check for expiration

Returns

VB:

0 if object has not expired, 1 if object has expired (as Long). The VB Err object will contain the MRTK result code if an error occurs.

C++:

0 if object has not expired, 1 if object has expired, or an MRTK result code if an error occurred

Notes

VB Example

```
…
Dim nObjExpired as Long
nObjExpired = _  MRTKFactory.IsObjectExpired(ENTERPRISE_CAPOSTCODETASK)
If nObjExpired = 1 Then
MsgBox "CAPostalCodeTask has expired - please update"
End If
…
```

See also

See the tagMRTKOBJECTS Properties table for a list of possible objects

IsUpdateAvailable

Checks if an update is available

Syntax

VB:

```
IsUpdateAvailable
```

C++:

```
long IsUpdateAvailable(long *nUpdateAvail)
```

Arguments

VB:

None

C++:

nUpdateAvail

Returns 1 if update available, 0 if not available

Returns

VB:

1 if update available, 0 if not available (as Long). The VB Err object will contain the MRTK result code if an error occurs.

C++:

0 if successful, otherwise an MRTK result code

Notes

Checks the data file folder containing the current location of the Datafile.dat file for an updated version of MRTK. If the most current update has been previously run then this function will return 0. The current Datafile.dat file location can be retrieved using the MRTK Global property caDATAFILE_LOCATION.

VB Example

```
…
Dim bUpdate As Boolean
bUpdate = MRTKFactory.IsUpdateAvailable
If bUpdate Then
       MRTKFactory.RunUpdate
End If
…
```

See also

MRTKObjFactory.RunUpdate

MRTK Global Properties table for the definition of:

caDATAFILE_LOCATION

RunUpdate

Update the BCC Architect controls to the latest version

Syntax

VB:

```
RunUpdate
```

C++:

```
long RunUpdate()
```

Arguments

None

Returns

VB:

None. The VB Err object will contain the MRTK result code if an error occurs.

C++:

0 if successful, otherwise an MRTK result code

Notes

Only call RunUpdate if IsUpdateAvailable returns True. Running the updater will sometimes require the computer to be rebooted because a file that needs to be updated may be locked by the operating system. In this event, RunUpdate returns a result code indicating that a system reboot is necessary to complete the update. These files will be freed after a reboot is performed.

VB Example

```
Public Sub RunMRTKUpdate()
  On Error GoTo errRunMRTKUpdate
  Dim MRTKObjectFactory As Object
  Set MRTKObjectFactory = _
CreateObject("MRTKFactory.MRTKObjFactory.1")
  If (MRTKObjectFactory.IsUpdateAvailable) Then
    MRTKObjectFactory.RunUpdate
  End If
  Set MRTKObjectFactory = Nothing
  Exit Sub
errRunMRTKUpdate:
  If Err.Number = &H80040417 then
MsgBox "It is necessary to reboot your computer _ to complete the update."
  ElseIf Err.Number = &H80040418 then
MsgBox "The autoupdater failed to update _ Architect."
  Else
       MsgBox "Some other error occurred."
  End If
  Set MRTKObjectFactory = Nothing
```

```
End Sub
…
' A good place to test for updates is within the
' Form.Load event handler
Private Sub Form_Load()
  RunMRTKUpdate
End Sub
```

See also

MRTKObjFactory.IsUpdateAvailable

Object Factory Properties

The MRTKObjFactory properties are defined below. After you have added a reference to the MRTKFact.dll, you can begin using these properties in your project.

DataFilePath

Set/retrieve the location of the data files

Syntax

VB:

```
DataFilePath(nCountryID)
```

C++:

```
long DataFilePath(tagMRTKCOUNTRYIDS nCountryID,
BSTR *pVal) /* when retrieving */
long DataFilePath(tagMRTKCOUNTRYIDS nCountryID,
BSTR newVal) /* when setting */
```

Data Type

VB:

String

C++:

BSTR

Arguments

VB:

nCountryID as tagMRTKCOUNTRYIDS

The country whose data file path you want to set or retrieve

C++:

nCountryID

The country whose data file path you want to set or retrieve

pVal

Returns property value

newVal

Value to assign to the

Returns

VB:

None. The VB Err object will contain the MRTK result code if an error occurs.

C++:

0 if successful, otherwise an MRTK result code

Notes

Can read and set this property. This property acts the same as the MRTKObjFactory.FileSearchPath property when the nCountryID is UNITED_STATES. This property is to be used for US, CANADA, and UK datasets.

When setting this property, you can use any of the following formats:

- `"C:\BCC Architect\Datafile"`
- `"C:\BCC Architect\Datafile\"`
- `"C:\BCC Architect\Datafile\UKAddr.paf" (United Kingdom)`
- `"C:\BCC Architect\Datafile\Address.cas" (United States)`
- `"C:\BCC Architect\Datafile\Datafile.dat" (Canada)`

When retrieving this property, it will always return the path to the data file folder in upper case letters followed by a back-slash, regardless of the format used to set it:

- `"C:\QBCC Architect\DATAFILE\"`

VB Example

```
…
Dim szDataPath As String
szDataPath = "C:\BCC Architect\Datafile"
…
MRTKFactory.DataFilePath(CANADA) = szDataPath
…
szDataPath = MRTKFactory.DataFilePath(CANADA)
…
```

See also

See the tagMTRKCOUNTRYIDS Properties table for a list of available countries

FileSearchPath

Set/retrieve the location of the data files (united states only)

Syntax

VB:

```
FileSearchPath
```

C++:

```
long FileSearchPath(BSTR *pVal) /* when retrieving */
long FileSearchPath(BSTR newVal) /* when setting */
```

Data Type

VB:

String

C++:

BSTR

Arguments

VB:

None

C++:

pVal

Returns property value

newVal

Value to assign to the

Returns

VB:

None. The VB Err object will contain the MRTK result code if an error occurs.

C++:

0 if successful, otherwise an MRTK result code

Notes

Can read and set this property. To be used only with MRTK United States. For other countries, use the DataFilePath property.

VB Example

```
…
Dim szDataPath As String
szDataPath = "C:\BCC Architect\Datafile"
…
MRTKFactory.FileSearchPath = szDataPath
…
szDataPath = MRTKFactory.FileSearchPath
…
```

See also

MRTKObjFactory.DataFilePath

MRTKRegistrationKey

Set/retrieve the BCC Architect registration serial number

Syntax

VB:

```
MRTKRegistrationKey
```

C++:

```
long MRTKRegistrationKey(BSTR *pVal) /*when retrieving*/
long MRTKRegistrationKey(BSTR newVal) /*when setting*/
```

Data Type

VB:

String

C++:

BSTR

Arguments

VB:

None

C++:

pVal

Returns property value

newVal

Value to assign to the

Returns

VB:

None. The VB Err object will contain the MRTK result code if an error occurs.

C++:

0 if successful, otherwise an MRTK result code

Notes

Can read and set this property. Setting this property will change the registration serial number in the registry.

VB Example

```
…
Dim szRegKey as String
szRegKey = "XXXXXXDEFGHDDDDPP9U8"
…
MRTKFactory.MRTKRegistrationKey = szRegKey
…
szRegKey = MRTKFactory.MRTKRegistrationKey
…
```

See also

MRTK Object Names Summary Table

Below are the string ids of each MRTK object

> NOTE It is recommended that all Task objects be created through the MRTKObjFactory object.

String Object Name	Description
MRTKFact-ory.MRTKObjFactory.1	Name of the MRTKObjFactory object
MRTKTask.PostalCodeTask.1	Name of the PostalCodeTask object (United States single address corrections)
MRTKTask.BatchTask.1	Name of the BatchTask object (United States batch address corrections)
MRTKTask.ReviewErrorsTask.1	Name of the ReviewErrorsTask object (United States)
MRTKTask.CASOARe-portTask.1	Name of the CASOAReportTask object (United States)
MRTKTask.CASortTask .1	Name of the CASortTask object (United States)

MRTKUK.UKPostCodeTask.1	Name of the UKPostCodeTask object (United Kingdom single address corrections)
MRTKUK.UKBatchTask.1	Name of the UKBatchTask object (United Kingdom batch address corrections)
MRTKUK.UKBatchReportTask.1	Name of the UKBatchReportTask object (United Kingdom)
MRTKUK.UKSortTask.1	Name of the UKSortTask object (United Kingdom)
MRTKCA.CAPostalCodeTask.1	Name of the CAPostalCodeTask object (Canada single address corrections)
MRTKCA.CABatchTask.1	Name of the CABatchTask object (Canada batch address corrections)
MRTKCA.CASOAReportTask.1	Name of the CASOAReportTask object (Canada)
MRTKCA.CASortTask.1	Name of the CASortTaskobject (Canada)

Object Factory MRTKFACTORYLib Enums Summary Table

Below are all the MRTKfactorylib property names.

tagMRTKOBJECTS Properties	Enum Value	Description
ENTERPRISE_ CAPOSTALCODETASK	12	Create the Architect version of Canada ZIP Agent
ENTERPRISE_CABATCHTASK	15	Create the Architect version of Canada Batch Agent
ENTERPRISE_CASOAREPORTTASK	16	Create the Architect version of Canada Statement of Accuracy Report
ENTERPRISE_CASORTTASK	14	Create the Architect version of Canada Sort Agent
ENTERPRISE_CASOAREPORTTASK	4	Create the Architect version of CASS Agent Reporting
ENTERPRISE_BATCHTASK	3	Create the Architect version of CASS Agent
ENTERPRISE_DATABROWSER	8	Create the Architect version of ZIP Browser
ENTERPRISE_CASORTTASK	5	Create the Architect version of PRESORT Agent
ENTERPRISE_REVIEWERRORSTASK	9	Create the Architect version of ReviewErrors Agent
ENTERPRISE_ UKBATCHREPORTTASK	17	Create the Architect version of UK Batch Report
ENTERPRISE_UKBATCHTASK	13	Create the Architect version of UK Batch Agent
ENTERPRISE_UKPOSTCODETASK	11	Create the Architect version of UK ZIP Agent

ENTERPRISE_UKSORTTASK	10	Create the Architect version of UK Sort Agent
ENTERPRISE_POSTALCODETASK	6	Create the Architect version of ZIP Agent
SMALLBUSINESS_CASSAGENT	0	Create CASS Agent interface
SMALLBUSINESS_UKBATCHAGENT	20	Create Batch Agent interface
SMALLBUSINESS_ UKPOSTCODEAGENT	19	Create Postcode Agent interface
SMALLBUSINESS_UKSORTAGENT	18	Create Mailsort Agent interface
SMALLBUSINESS_ZIPAGENT	1	Create ZIP Agent interface
SMALLBUSINESS_ZIPBROWSER	2	Create ZIP Browser interface

tagMRTKCOUNTRYIDS Properties	Enum Value	Description
CANADA	2	Canada
UNITED_KINGDOM	1	United Kingdom
UNITED_STATES	0	United States of America

Object Factory Global Properties Summary Table

MRTK Global Properties	Enum Value	Data Type	Default Value	Description
caDATAFILE_ LOCATION	100	String	EMPTY	Location of Datafile.dat data file
caDELIMITER_ FIELD	175	String	'\t' , Chr(9) (tab)	Field delimiting character
caDELIMITER_ RECORD	176	String	'\n', Chr(10) (line feed)	Record delimiting character
caFIELD_LIST_ IN	200	String	EMPTY	Delimited list specifying input fields
caFIELD_LIST_ OUT	201	String	EMPTY	Delimited list specifying desired output fields

caINPUT_BLOCK_RECORD_COUNT	202	Long	1	Count of records within record block
caMAILROOM_SERVER_LIST	203	String	EMPTY	Sets and retrieves the location of the BCC Architect Server. Set this property to create a TCP/IP connection to the BCC Architect Server, which can reside on the local network or virtually anywhere. The format is: Server Name or IP Address:Port.
caSETTINGS_INI_FILE_NAME	150	String	“MRTKca.ini”	Full name and path of ini file to use
caTEMPLATE_NAME_TO_USE	151	String	EMPTY	The template is a section in the ini file that stores presort settings, printer names, etc.

The COM PostalCodeTask Object for Correcting Single Addresses

The BCC Architectt CAPostalCodeTask object certifies a single address. The CAPostalCodeTask object is designed for use within address entry forms, web pages, and any other environment where single address verification is needed. After calling CAPostalCodeTask.CheckAddress, you can retrieve the corrected address information. This includes the correct postal code and all the other individual elements of an address, if the address was certified. Returning the individual elements is a powerful tool that can be used to test if certain elements are missing. For instance, with just a few lines of code you can tell whether or not a user entered a municipality or unit number. If the CheckAddress method was unable to certify a record, an error code and its associated description can be retrieved.

PostalCodeTask Functions

The PostalCodeTask functions are defined below. Once you have added a reference to the BCC Architect files, you can begin using these functions in your project.

PrepareTask

Initialize and prepare the object

Syntax

VB:

```
PrepareTask
```

C++:

```
long PrepareTask()
```

Arguments

None

Returns

VB:

None. The VB Err object will contain the MRTK result code if an error occurs.

C++:

0 if successful, otherwise an MRTK result code

Notes

PrepareTask should be called only once, when the CAPostalCodeTask object is first created. This function must be called before any of the other functions/properties of CAPostalCodeTask. Failing to do so will cause subsequent function calls to fail. The one exception to this rule is setting the path of the Datafile.dat file with the DatafileLocation property. It is preferable, however, to set the data file path using the MRTKObjFactory.DataFilePath property.

VB Example

```
Dim MRTKFactory As New MRTKFACTORYLib.MRTKObjFactory
Dim objCAPostalCodeTask As CAPostalCodeTask
Set objCAPostalCodeTask = _ MRTKFactory.CreateObject(ENTERPRISE_CAPOSTCODETASK)
objCAPostalCodeTask.PrepareTask
…
```

See also

PostalCodeTask Properties

The PostalCodeTask properties are defined below. Once you have added a reference to the BCC Architect files, you can begin using these properties in your project.

AddressLine1

Sets/retrieves Address Line 1

Syntax

VB:

```
AddressLine1
```

C++:

```
long AddressLine1 (BSTR *pVal) /* when retrieving */
long AddressLine1 (BSTR newVal) /* when setting */
```

Data Type

VB:

String

C++:

BSTR

Arguments

VB:

None

C++:

pVal

Returns property value

newVal

Value to assign to the

Returns

VB:

None. The VB Err object will contain the MRTK result code if an error occurs.

C++:

0 if successful, otherwise an MRTK result code

Notes

Can read and set this property. Address line 1 is always the line directly above the municipality in the address. After calling CheckAddress, the address line 1 returned from this property is correctly formatted to the specifications of Canada Post.

VB Example

```
…
Dim szAddressLine1 As String
objCAPostalCodeTask.CheckAddress
szAddressLine1 = objCAPostalCodeTask.AddressLine1
…
```

See also

CAPostalCodeTask.CheckAddress

AddressLine2

Sets/retrieves Address Line 2

Syntax

VB:

```
AddressLine2
```

C++:

```
long AddressLine2 (BSTR *pVal) /* when retrieving */
long AddressLine2 (BSTR newVal) /* when setting */
```

Data Type

VB:

String

C++:

BSTR

Arguments

VB:

None

C++:

pVal

Returns property value

newVal

Value to assign to the

Returns

VB:

None. The VB Err object will contain the MRTK result code if an error occurs.

C++:

0 if successful, otherwise an MRTK result code

Notes

Can read and set this property. Address line 2 is always the line directly above address line 1 in the address. After calling CheckAddress, the address line 2 returned from this property is correctly formatted to the specifications of Canada Post.

VB Example

```
…
Dim szAddressLine2 As String
objCAPostalCodeTask.CheckAddress
szAddressLine2 = objCAPostalCodeTask.AddressLine2
…
```

See also

CAPostalCodeTask.CheckAddress

BrowseAddress

Launches the postal code browser dialog

Syntax

VB:

```
BrowseAddress
```

C++:

```
long BrowseAddress(long *pUpdateRequested)
```

Arguments

VB:

None

C++:

pUpdateRequested

Returns 0 if user clicks Cancel, 1 if user clicks Update

Returns

VB:

0 if user clicks Cancel, 1 if user clicks Update (as Long). The VB Err object will contain the MRTK result code if an error occurs.

C++:

0 if successful, otherwise an MRTK result code

Notes

This method will launch the Postal Code Browser and try to "lookup" the address contained within the CAPostalCodeTask object. The Postal Code Browser is a tool that will allow the user to search through all the addresses contained in the address files. If you want to launch the Postal Code Browser and not look up an address then call CAPostalCodeTask.ClearAddress before calling this method. If the return value is True, then the CAPostalCodeTask object will contain the address elements for the address that the user chose to keep within the Postal Code Browser window.

By checking the ErrorCode property after calling the CheckAddress function, the Postal Code Browser can be automatically launched for an address that does not certify.

As of now, the CAPostalCodeTask object must be able to access the MRTK data files in order to use this method. In the future the BCC Architect Server will be able to act as a data feed for the Postal Code Browser.

VB Example

```
Dim nErrorCode As Long
…
objCAPostalCodeTask.AddressLine2 = "UPPER FLOOR"
objCAPostalCodeTask.AddressLine1 = "350 MERRILL ST"
objCAPostalCodeTask.Municipality = "THUNDER BAY"
objCAPostalCodeTask.Province = "ON"
objCAPostalCodeTask.Postcode = "P7A 1E6"
```

```
…
objCAPostalCodeTask.CheckAddress
nErrorCode = objCAPostalCodeTask.ErrorCode
' Check if record did not certify
If (nErrorCode >= 100) And (nErrorCode < 500) Then
' Launch Postal Code Browser to find out exactly
' what was wrong with the address
       If (objCAPostalCodeTask.BrowseAddress) Then
       ' Save address elements
       …
       End If
Else
       ' Save address elements
…
End If
```

See also

CAPostalCodeTask.CheckAddress

CAPostalCodeTask.ClearAddress

CAPostalCodeTask.ErrorCode

BusinessName

Sets/retrieves the business name of an address

Syntax

VB:

```
BusinessName
```

C++:

```
long BusinessName(BSTR *pVal) /* when retrieving */
long BusinessName(BSTR newVal) /* when setting */
```

Data Type

VB:

String

C++:

BSTR

Arguments

VB:

None

C++:

pVal

Returns property value

newVal

Value to assign to the

Returns

VB:

None. The VB Err object will contain the MRTK result code if an error occurs.

C++:

0 if successful, otherwise an MRTK result code

Notes

Can read or set this property. This property will be updated only if the address is certified. Otherwise, the original business name is returned.

VB Example

```
…
Dim szBusinessName As String
objCAPostalCodeTask.BusinessName = "Mini Market"
szBusinessName = objCAPostalCodeTask.BusinessName
…
```

See also

Capitalize

Determines if returned address elements will be capitalized

Syntax

VB:

```
Capitalize
```

C++:

```
long Capitalize(long *pVal) /* when retrieving */
long Capitalize(long newVal) /* when setting */
```

Data Type

VB:

Long

C++:

long

Arguments

VB:

None

C++:

pVal

Returns 0 if False, 1 if True

newVal

Set to 0 for False, 1 for True

Returns

VB:

0 if False, 1 if True (as Long). The VB Err object will contain the MRTK result code if an error occurs.

C++:

0 if successful, otherwise an MRTK result code

Notes

Can read or set this property. You must call CheckAddress before the address elements in the CAPostalCodeTask object will be capitalized. The default value for this property is False.

VB Example

```
…
Dim szAddressBlock As String
objCAPostalCodeTask.Capitalize = 1
objCAPostalCodeTask.CheckAddress
szAddressBlock = objCAPostalCodeTask.AddressBlock
…
```

See also

CAPostalCodeTask.CheckAddress

CASSDate

Sets/retrieves the last certification date of an address

Syntax

VB:

```
CASSDate
```

C++:

```
long CASSDate(long *pVal) /* when retrieving */
long CASSDate(long newVal) /* when setting */
```

Data Type

VB:

Long

C++:

long

Arguments

VB:

None

C++:

pVal

Returns property value

newVal

Value to assign to the

Returns

VB:

None. The VB Err object will contain the MRTK result code if an error occurs.

C++:

0 if successful, otherwise an MRTK result code

Notes

The CASS date is a binary field that contains information about the last results from the address matching engine. One of the fields within the CASSDate is the DVD issue number that was used to certify the address. If you are going to batch process it is strongly recommended that you save this field because it will allow a batch process (via CAPostalCodeTask, for instance) to skip this record if it was previously certified, based on the CertifyFlag property. Can read or set this property.

VB Example

```
…
Dim nCASSDate As Long
nCASSDate = objCAPostalCodeTask.CASSDate
…
```

See also

CAPostalCodeTask.CertifyFlag

CertifyFlag

Sets/retrieves the flag that determines if an address will be processed

Syntax

VB:

```
CertifyFlag
```

C++:

```
long CertifyFlag(long *pVal) /* when retrieving */
long CertifyFlag(long newVal) /* when setting */
```

Data Type

VB:

Long

C++:

long

Arguments

VB:

None

C++:

pVal

Returns property value

newVal

Value to assign to the

Returns

VB:

None. The VB Err object will contain the MRTK result code if an error occurs.

C++:

0 if successful, otherwise an MRTK result code

Notes

Indicate whether to skip re-certification for records that have been previously certified based on the CASSDate field:

- -2 =Check only records not certified with this CD, the last DVD or the DVD prior to the last DVD
- -1 = Check only records not certified with this DVD or the last CD
- 0 = Check only records not certified with this CD
- 1 = Check every record

Indicate records to include when building the Summary Report based on the CASSDate field:

- 2 = Certified with this CD, the last DVD or the DVD prior to the last CD
- 3 = Certified with this DVD or the last CD
- 4 = Certified with this CD

All records passed in will be checked to see if they "qualify" to be added to the totals for the Summary Report.

> NOTE If a record is coded but does not qualify, it will show up in the errors section of the report.

VB Example

```
…
Dim nCertifyFlag As Long
nCertifyFlag = objCAPostalCodeTask.CertifyFlag
…
```

See also

CAPostalCodeTask.CASSDate

CheckAddress

Perform CA matching of address

Syntax

VB:

```
CheckAddress
```

C++:

```
long CheckAddress()
```

Arguments

None

Returns

VB:

None. The VB Err object will contain the MRTK result code if an error occurs.

C++:

0 if successful, otherwise an MRTK result code

Notes

This function will try to certify the address contained in the CAPostalCodeTask object. After a call to CheckAddress, you can get the results of the certification by looking at the ErrorCode property. In some situations you may want to call BrowseAddress if ErrorCode represents an error.

VB Example

```
Dim nErrorCode As Long
…
objCAPostalCodeTask.AddressLine2 = "UPPER FLOOR"
objCAPostalCodeTask.AddressLine1 = "350 MERRILL ST"
objCAPostalCodeTask.Municipality = "THUNDER BAY"
objCAPostalCodeTask.Province = "ON"
objCAPostalCodeTask.Postcode = "P7A 1E6"
…
objCAPostalCodeTask.CheckAddress
nErrorCode = objCAPostalCodeTask.ErrorCode
' Check if record did not certify
If (nErrorCode >= 100) And (nErrorCode < 500) Then
' Launch Postal Code Browser to find out exactly
' what was wrong with the address
       If (objCAPostalCodeTask.BrowseAddress) Then
       ' Save address elements
       …
       End If
Else
       ' Save address elements
…
End If
```

See also

CAPostalCodeTask.BrowseAddress function

CAPostalCodeTask.AddressBlock

CAPostalCodeTask.ErrorCode

CivicNumber

Sets/retrieves the civic number of an address

Syntax

VB:

```
CivicNumber
```

C++:

```
long CivicNumber(BSTR *pVal) /* when retrieving */
long CivicNumber(BSTR newVal) /* when setting */
```

Data Type

VB:

String

C++:

BSTR

Arguments

VB:

None

C++:

pVal

Returns property value

newVal

Value to assign to the

Returns

VB:

None. The VB Err object will contain the MRTK result code if an error occurs.

C++:

0 if successful, otherwise an MRTK result code

Notes

Can read or set this property.

VB Example

```
…
Dim szCivicNumber As String
objCAPostalCodeTask.CivicNumber = "36"
szCivicNumber = objCAPostalCodeTask.CivicNumber
…
```

See also

CivicNumberSuffix

Sets/retrieves the civic number suffix of an address

Syntax

VB:

```
CivicNumberSuffix
```

C++:

```
long CivicNumberSuffix(BSTR *pVal) /* when retrieving */
long CivicNumberSuffix(BSTR newVal) /* when setting */
```

Data Type

VB:

String

C++:

BSTR

Arguments

VB:

None

C++:

pVal

Returns property value

newVal

Value to assign to the

Returns

VB:

None. The VB Err object will contain the MRTK result code if an error occurs.

C++:

0 if successful, otherwise an MRTK result code

Notes

Can read or set this property.

VB Example

```
…
Dim szCivicNumberSuffix As String
objCAPostalCodeTask.CivicNumberSuffix = "A"
```

```
szCivicNumberSuffix = _
objCAPostalCodeTask.CivicNumberSuffix
…
```

See also

ClearAddress

Reset the address information to nulls

Syntax

VB:

```
ClearAddress
```

C++:

```
Long ClearAddress()
```

Arguments

None

Returns

VB:

None. The VB Err object will contain the MRTK result code if an error occurs.

C++:

0 if successful, otherwise an MRTK result code

Notes

This function can be used to quickly clear the address information from a CAPostalCodeTask object. This function can also be used to bring the CAPostalCodeTask object into a known state (i.e., no address, error code, etc.) before setting the object's properties.

VB Example

```
…
objCAPostalCodeTask.ClearAddress
objCAPostalCodeTask.AddressLine2 = "UPPER FLOOR"
objCAPostalCodeTask.AddressLine1 = "350 MERRILL ST"
…
objCAPostalCodeTask.Postcode = "P7A 1E6"
…
objCAPostalCodeTask.CheckAddress
…
```

See also

Country

Retrieve the country of a foreign address

Syntax

VB:

```
Country
```

C++:

```
long Country(BSTR *pVal) /* when retrieving */
```

Data Type

VB:

String

C++:

BSTR

Arguments

VB:

None

C++:

pVal

Returns property value

Returns

VB:

None. The VB Err object will contain the MRTK result code if an error occurs.

C++:

0 if successful, otherwise an MRTK result code

Notes

Can only read this property. This property is not currently implemented.

VB Example

```
…
Dim szCountry As String
```

```
szCountry = objCAPostalCodeTask.Country
…
```

See also

DatafileLocation

Sets/retrieves the data file location

Syntax

VB:

```
DatafileLocation
```

C++:

```
long DatafileLocation(BSTR *pVal) /* when retrieving */
long DatafileLocation(BSTR newVal) /* when setting */
```

Data Type

VB:

String

C++:

BSTR

Arguments

VB:

None

C++:

pVal

Returns property value

newVal

Value to assign to the

Returns

VB:

None. The VB Err object will contain the MRTK result code if an error occurs.

C++:

0 if successful, otherwise an MRTK result code

Notes

Can read or set this property. The Datafile.dat file is required to perform address matching and its location must be specified by the user. It is important that this location be set prior to calling PrepareTask. It is preferable, however, to set the data file path using the MRTKObjFactory.DataFilePath property.

VB Example

```
…
Dim szDatafileLocation As String
objCAPostalCodeTask.DatafileLocation = _
"C:\Program Files\Satori Architect\Datafile"
szDatafileLocation = objCAPostalCodeTask.DatafileLocation
…
```

See also

ErrorCode

Retrieves the error code from an address matching operation

Syntax

VB:

```
ErrorCode
```

C++:

```
long ErrorCode(long *pVal) /* when retrieving */
```

Data Type

VB:

Long

C++:

long

Arguments

VB:

None

C++:

pVal

Returns error code value

Returns

VB:

None. The VB Err object will contain the MRTK result code if an error occurs.

C++:

0 if successful, otherwise an MRTK result code

Notes

Can only read this property. You should call CheckAddress before attempting to retrieve this property.

VB Example

```
…
Dim nErrorCode As Long
objCAPostalCodeTask.CheckAddress
nErrorCode = objCAPostalCodeTask.ErrorCode
…
```

See also

CAPostalCodeTask.CheckAddress

See the MRTKCALib Error Codes Table for a list of error code values and their descriptions

ErrorCodeString

Retrieves the error code string associated with an error code

Syntax

VB:

```
ErrorCodeString(nExtendedError)
```

C++:

```
long ErrorCodeString(long nExtendedError, BSTR *pVal)
/* when retrieving */
```

Data Type

VB:

String

C++:

BSTR

Arguments

VB:

nExtendedError

0 for standard description, 1 for extended description (as Long)

C++:

nExtendedError

0 for standard description, 1 for extended description

pVal

Returns property value

Returns

VB:

None. The VB Err object will contain the MRTK result code if an error occurs.

C++:

0 if successful, otherwise an MRTK result code

Notes

Can only read this property. You should call CheckAddress before attempting to retrieve this property.

VB Example

```
…
Dim szErrorCode As String
objCAPostalCodeTask.CheckAddress
szErrorCode = objCAPostalCodeTask.ErrorCodeString(False)
…
```

See also

CAPostalCodeTask.CheckAddress

GDType

Sets/retrieves the general delivery type of an address

Syntax

VB:

```
GDType
```

C++:

```
long GDType(BSTR *pVal) /* when retrieving */
long GDType(BSTR newVal) /* when setting */
```

Data Type

VB:

String

C++:

BSTR

Arguments

VB:

None

C++:

pVal

Returns property value

newVal

Value to assign to the

Returns

VB:

None. The VB Err object will contain the MRTK result code if an error occurs.

C++:

0 if successful, otherwise an MRTK result code

Notes

Can read or set this property.

VB Example

```
…
Dim szGDType As String
objCAPostalCodeTask.GDType = "POSTE RESTANTE"
szGDType = objCAPostalCodeTask.GDType
…
```

See also

MailRoomServer

Set/retrieve the BCC Architect Server IP address and port number

Syntax

VB:

```
MailRoomServer
```

C++:

```
long MailRoomServer(BSTR *pVal) /* when retrieving */
long MailRoomServer(BSTR newVal) /* when setting */
```

Data Type

VB:

String

C++:

BSTR

Arguments

VB:

None

C++:

pVal

Returns property value

newVal

Value to assign to the

Returns

VB:

None. The VB Err object will contain the MRTK result code if an error occurs.

C++:

0 if successful, otherwise an MRTK result code

Notes

Can read and set this property. Use this property to set and retrieve the location of the BCC Architect Server. Setting this property creates a TCP/IP connection to the BCC Architect Server, which can reside on the local network or virtually anywhere. It is recommended that the BCC Architect Server be used when certifying addresses from a web site. The format is: Server Name or IP Address:Port. Currently, going outside of the proxy server might not be supported. Also, using the BCC Architect Server with the Postal Code Browser is not currently supported.

VB Example

```
…
Dim szMRServerLoc as String
objCAPostalCodeTask.MailRoomServer = _
"MailRoom1.SatoriSoftware.com:5150"
…
szMRServerLoc = objCAPostalCodeTask.MailRoomServer
…
```

See also

BCC Architect Server

Municipality

Sets/retrieves the municipality of an address

Syntax

VB:

```
Municipality
```

C++:

```
long Municipality(BSTR *pVal) /* when retrieving */
long Municipality(BSTR newVal) /* when setting */
```

Data Type

VB:

String

C++:

BSTR

Arguments

VB:

None

C++:

pVal

Returns property value

newVal

Value to assign to the

Returns

VB:

None. The VB Err object will contain the MRTK result code if an error occurs.

C++:

0 if successful, otherwise an MRTK result code

Notes

Can read or set this property.

VB Example

```
…
Dim szMunicipality As String
objCAPostalCodeTask.Municipality = "THUNDER BAY"
szMunicipality = objCAPostalCodeTask.Municipality
…
```

See also

POBoxNumber

Sets/retrieves the PO Box number of an address

Syntax

VB:

```
POBoxNumber
```

C++:

```
long POBoxNumber(BSTR *pVal) /* when retrieving */
long POBoxNumber(BSTR newVal) /* when setting */
```

Data Type

VB:

String

C++:

BSTR

Arguments

VB:

None

C++:

pVal

Returns property value

newVal

Value to assign to the

Returns

VB:

None. The VB Err object will contain the MRTK result code if an error occurs.

C++:

0 if successful, otherwise an MRTK result code

Notes

Can read or set this property.

VB Example

```
…
Dim szPOBoxNumber As String
objCAPostalCodeTask.POBoxNumber = "26003"
szPOBoxNumber = objCAPostalCodeTask.POBoxNumber
…
```

See also

POBoxType

Sets/retrieves the PO Box type of an address

Syntax

VB:

```
POBoxType
```

C++:

```
long POBoxType(BSTR *pVal) /* when retrieving */
long POBoxType(BSTR newVal) /* when setting */
```

Data Type

VB:

String

C++:

BSTR

Arguments

VB:

None

C++:

pVal

Returns property value

newVal

Value to assign to the

Returns

VB:

None. The VB Err object will contain the MRTK result code if an error occurs.

C++:

0 if successful, otherwise an MRTK result code

Notes

Can read or set this property.

VB Example

```
…
Dim szPOBoxType As String
objCAPostalCodeTask.POBoxType = "PO BOX"
szPOBoxType = objCAPostalCodeTask.POBoxType
…
```

See also

Postcode

Sets/retrieves the postal code of an address

Syntax

VB:

```
Postcode
```

C++:

```
long Postcode(BSTR *pVal) /* when retrieving */
```

```
long Postcode(BSTR newVal) /* when setting */
```

Data Type

VB:

String

C++:

BSTR

Arguments

VB:

None

C++:

pVal

Returns property value

newVal

Value to assign to the

Returns

VB:

None. The VB Err object will contain the MRTK result code if an error occurs.

C++:

0 if successful, otherwise an MRTK result code

Notes

Can read or set this property. The PostCode property is always formatted with a space separating the Forward Sortation Area (first three characters) and the Local Delivery Unit (last three characters).

VB Example

```
…
Dim szPostCode As String
objCAPostalCodeTask.Postcode = "N1R 8E8"
Postcode = objCAPostalCodeTask.Postcode
…
```

See also

Province

Sets/retrieves the province of an address

Syntax

VB:

```
Province
```

C++:

```
long Province(BSTR *pVal) /* when retrieving */
long Province(BSTR newVal) /* when setting */
```

Data Type

VB:

String

C++:

BSTR

Arguments

VB:

None

C++:

pVal

Returns property value

newVal

Value to assign to the

Returns

VB:

None. The VB Err object will contain the MRTK result code if an error occurs.

C++:

0 if successful, otherwise an MRTK result code

Notes

Can read or set this property.

VB Example

```
…
Dim szProvince As String
objCAPostalCodeTask.Province = "BC"
```

```
szProvince = objCAPostalCodeTask.Province
…
```

See also

RouteNumber

Sets/retrieves the route number of an address

Syntax

VB:

```
RouteNumber
```

C++:

```
long RouteNumber(BSTR *pVal) /* when retrieving */
long RouteNumber(BSTR newVal) /* when setting */
```

Data Type

VB:

String

C++:

BSTR

Arguments

VB:

None

C++:

pVal

Returns property value

newVal

Value to assign to the

Returns

VB:

None. The VB Err object will contain the MRTK result code if an error occurs.

C++:

0 if successful, otherwise an MRTK result code

Notes

Can read or set this property

VB Example

```
…
Dim szRouteNumber As String
objCAPostalCodeTask.RouteNumber = "6"
szRouteNumber = objCAPostalCodeTask.RouteNumber
…
```

See also

RouteType

Sets/retrieves the route type of an address

Syntax

VB:

```
RouteType
```

C++:

```
long RouteType(BSTR *pVal) /* when retrieving */
long RouteType(BSTR newVal) /* when setting */
```

Data Type

VB:

String

C++:

BSTR

Arguments

VB:

None

C++:

pVal

Returns property value

newVal

Value to assign to the

Returns

VB:

None. The VB Err object will contain the MRTK result code if an error occurs.

C++:

0 if successful, otherwise an MRTK result code

Notes

Can read or set this property.

This information may be contained in the address line properties.

Contains one of the following:

- RR Rural Route
- SS Suburban Service
- MR Mobile Route
- GD General Delivery

CheckAddress should be called before attempting to retrieve the value of this property.

VB Example

```
…
Dim szRouteType As String
objCAPostalCodeTask.RouteType = "RR"
szRouteType = objCAPostalCodeTask.RouteType
…
```

See also

StationName

Sets/retrieves the station name of an address

Syntax

VB:

```
StationName
```

C++:

```
long StationName(BSTR *pVal) /* when retrieving */
long StationName(BSTR newVal) /* when setting */
```

Data Type

VB:

String

C++:

BSTR

Arguments

VB:

None

C++:

pVal

Returns property value

newVal

Value to assign to the

Returns

VB:

None. The VB Err object will contain the MRTK result code if an error occurs.

C++:

0 if successful, otherwise an MRTK result code

Notes

Can read or set this property.

VB Example

```
…
Dim szStationName As String
objCAPostalCodeTask.StationName = "SCARBOROUGH"
szStationName = objCAPostalCodeTask.StationName
…
```

See also

StationQualifier

Sets/retrieves the station qualifier of an address

Syntax

VB:

```
StationQualifier
```

C++:

```
long StationQualifier(BSTR *pVal) /* when retrieving */
long StationQualifier(BSTR newVal) /* when setting */
```

Data Type

VB:

String

C++:

BSTR

Arguments

VB:

None

C++:

pVal

Returns property value

newVal

Value to assign to the

Returns

VB:

None. The VB Err object will contain the MRTK result code if an error occurs.

C++:

0 if successful, otherwise an MRTK result code

Notes

Can read or set this property.

VB Example

```
…
Dim szStationQualifier As String
objCAPostalCodeTask.StationQualifier = "FULFORD HARBOUR"
```

```
szStationQualifier = objCAPostalCodeTask.StationQualifier
…
```

See also

StationType

Sets/retrieves the station type of an address

Syntax

VB:

```
StationType
```

C++:

```
long StationType(BSTR *pVal) /* when retrieving */
long StationType(BSTR newVal) /* when setting */
```

Data Type

VB:

String

C++:

BSTR

Arguments

VB:

None

C++:

pVal

Returns property value

newVal

Value to assign to the

Returns

VB:

None. The VB Err object will contain the MRTK result code if an error occurs.

C++:

0 if successful, otherwise an MRTK result code

Notes

Can read or set this property.

VB Example

```
…
Dim szStationType As String
objCAPostalCodeTask.StationType = "SUCC"
szStationType = objCAPostalCodeTask.StationType
…
```

See also

StreetDirection

Sets/retrieves the street direction of an address

Syntax

VB:

```
StreetDirection
```

C++:

```
long StreetDirection(BSTR *pVal) /* when retrieving */
long StreetDirection(BSTR newVal) /* when setting */
```

Data Type

VB:

String

C++:

BSTR

Arguments

VB:

None

C++:

pVal

Returns property value

newVal

Value to assign to the

Returns

VB:

None. The VB Err object will contain the MRTK result code if an error occurs.

C++:

0 if successful, otherwise an MRTK result code

Notes

Can read or set this property.

VB Example

```
…
Dim szStreetDirection As String
objCAPostalCodeTask.StreetDirection = "NW"
szStreetDirection = objCAPostalCodeTask.StreetDirection
…
```

See also

StreetName

Sets/retrieves the street name of an address

Syntax

VB:

```
StreetName
```

C++:

```
long StreetName(BSTR *pVal) /* when retrieving */
long StreetName(BSTR newVal) /* when setting */
```

Data Type

VB:

String

C++:

BSTR

Arguments

VB:

None

C++:

pVal

Returns property value

newVal

Value to assign to the

Returns

VB:

None. The VB Err object will contain the MRTK result code if an error occurs.

C++:

0 if successful, otherwise an MRTK result code

Notes

Can read or set this property.

VB Example

```
…
Dim szStreetName As String
objCAPostalCodeTask.StreetName = "LONDONDERRY"
szStreetName = objCAPostalCodeTask.StreetName
…
```

See also

StreetType

Sets/retrieves the street type of an address

Syntax

VB:

```
StreetType
```

C++:

```
long StreetType(BSTR *pVal) /* when retrieving */
long StreetType(BSTR newVal) /* when setting */
```

Data Type

VB:

String

C++:

BSTR

Arguments

VB:

None

C++:

pVal

Returns property value

newVal

Value to assign to the

Returns

VB:

None. The VB Err object will contain the MRTK result code if an error occurs.

C++:

0 if successful, otherwise an MRTK result code

Notes

Can read or set this property.

VB Example

```
…
Dim szStreetType As String
objCAPostalCodeTask.StreetType = "RUE"
szStreetType = objCAPostalCodeTask.StreetType
…
```

See also

UnitDesignator

Sets/retrieves the unit designator of an address

Syntax

VB:

```
UnitDesignator
```

C++:

```
long UnitDesignator(BSTR *pVal) /* when retrieving */
```

```
long UnitDesignator(BSTR newVal) /* when setting */
```

Data Type

VB:

String

C++:

BSTR

Arguments

VB:

None

C++:

pVal

Returns property value

newVal

Value to assign to the

Returns

VB:

None. The VB Err object will contain the MRTK result code if an error occurs.

C++:

0 if successful, otherwise an MRTK result code

Notes

Can read or set this property.

VB Example

```
…
Dim szUnitDesignator As String
objCAPostalCodeTask.UnitDesignator = "SUITE"
szUnitDesignator = objCAPostalCodeTask.UnitDesignator
…
```

See also

UnitNumber

Sets/retrieves the unit number of an address

Syntax

VB:

```
UnitNumber
```

C++:

```
long UnitNumber(BSTR *pVal) /* when retrieving */
long UnitNumber(BSTR newVal) /* when setting */
```

Data Type

VB:

String

C++:

BSTR

Arguments

VB:

None

C++:

pVal

Returns property value

newVal

Value to assign to the

Returns

VB:

None. The VB Err object will contain the MRTK result code if an error occurs.

C++:

0 if successful, otherwise an MRTK result code

Notes

Can read or set this property.

VB Example

```
…
Dim szUnitNumber As String
objCAPostalCodeTask.UnitNumber = "E210"
```

```
szUnitNumber = objCAPostalCodeTask.UnitNumber
…
```

See also

EndTask

When done with the task, clean up and release

Syntax

VB:

```
EndTask
```

C++:

```
long EndTask()
```

Arguments

None

Returns

VB:

None. The VB Err object will contain the MRTK result code if an error occurs.

C++:

0 if successful, otherwise an MRTK result code

Notes

CAPostalCodeTask.EndTask releases the matching engine. It is recommended that you call EndTask when you are done with the CAPostalCodeTask object. It is not required to call EndTask after each call to CheckAddress.

VB Example

```
…
objCAPostalCodeTask.EndTaskSet
objCAPostalCodeTask = Nothing
…
```

See also

The COM BatchTask Object for Correcting Batch Addresses

The BCC Architect CABatchTask object performs address certification on a batch of addresses, returning addresses correctly formatted to the specifications of Canada Post. CABatchTask provides a flexible interface through which you can control the amount of information returned for each address as well as the number of records returned during each batch process. In addition to data flow control, CABatchTask can also be set to indicate if a record was processed successfully on a previous occasion (see FLD_SKIPPED_CERTIFY in the MRTKCALib Field Names table). By checking this flag, a program can avoid a costly write to a database, potentially speeding up processing. As with all the BCC Architect Task objects, CABatchTask should be created through the MRTKObjFactory.

BatchTask Functions

PrepareTask

Initialize and prepare the object

Syntax

VB:

```
PrepareTask
```

C++:

```
long PrepareTask()
```

Arguments

None

Returns

VB:

None. The VB Err object will contain the MRTK result code if an error occurs.

C++:

0 if successful, otherwise an MRTK result code

Notes

This function must be called before any of the other functions/properties of CABatchTask. Failing to do so will cause subsequent function calls to fail. The one exception to this rule is calling SetProperty to set the path of the Datafile.dat file, which must be defined prior to calling PrepareTask. It is preferable, however, to set the data file path using the MRTKObjFactory.DataFilePath property.

VB Example

```
Dim MRTKFactory As New MRTKFACTORYLib.MRTKObjFactory
```

```
Dim objCABatchTask As CABatchTask
Set objCABatchTask = _
MRTKFactory.CreateObject(ENTERPRISE_CABATCHTASK)
objCABatchTask.PrepareTaskobjCABatchTask.SetProperty(…)
…
```

See also

BCC Architect Factory Object

GetProperty

Get a cabatchtask property value

Syntax

VB:

```
GetProperty(MRTKPropertyID)
```

C++:

```
long GetProperty(long MRTKPropertyID, VARIANT *pVal)
```

Arguments

VB:

MRTKPropertyID as Long

The property ID of the property to get

C++:

MRTKPropertyID

The property ID of the property to get

pVal

Returns the value of the

Returns

VB:

The value of the property (as Variant). The VB Err object will contain the MRTK result code if an error occurs.

C++:

0 if successful, otherwise an MRTK result code

Notes

VB Example

```
…
Dim szCompanyName as String
' Get the name of company performing mailing
szCompanyName = _
objCABatchTask.GetProperty(caBATCH_CUSTOMER_NAME)
…
```

See also

See the CABatchTask Properties table for a list of property IDs

SetProperty

Set a CABatchTask property

Syntax

VB:

```
SetProperty(MRTKPropertyID, value)
```

C++:

```
long SetProperty(long MRTKPropertyID, VARIANT value)
```

Arguments

VB:

MRTKPropertyID as Long

The property ID of the property to set

value as Variant

The value of the property to set

C++:

MRTKPropertyID

The property ID of the property to set

value

The value of the property to set

Returns

VB:

None. The VB Err object will contain the MRTK result code if an error occurs.

C++:

0 if successful, otherwise an MRTK result code

Notes

VB Example

```
Dim szCompanyName as String
szCompanyName = "XYZ Corp"
objCABatchTask.PrepareTask
…
objCABatchTask.SetProperty caBATCH_CUSTOMER_NAME, _ szCompanyName
…
objCABatchTask.ValidateProperties
…
```

See also

See the CABatchTask Properties table for a list of property IDs

ValidateProperties

Verify that the task is set up correctly and ready to run

Syntax

VB:

```
ValidateProperties
```

C++:

```
long ValidateProperties()
```

Arguments

None

Returns

VB:

None. The VB Err object will contain the MRTK result code if an error occurs.

C++:

0 if successful, otherwise an MRTK result code

Notes

The basic requirements of the CABatchTask are as follows:

The address matching engine is loaded and able to run.

The input field list consists of the minimum set of fields, namely, Postal Code and Address Line 1.

This function needs to be called before you call CABatchTask.Update.

VB Example

```
…
On Error Goto errValidateProperties
objCABatchTask.ValidateProperties
errValidateProperties:
' Msgbox Err.Description
…
…
```

See also

Update

Certify the addresses contained in the record block

Syntax

VB:

```
Update(bstrInBlock)
```

C++:

```
long Update(BSTR bstrInBlock, BSTR *pbstrOutBlock)
```

Arguments

VB:

bstrInBlock as String

A string that contains calNPUT_BLOCK_RECORD_COUNT addresses that are separated by caDELIMITER_FIELD and caDELIMITER_RECORD

C++:

bstrInBlock

A string that contains calNPUT_BLOCK_RECORD_COUNT addresses that are separated by caDELIMITER_FIELD and caDELIMITER_RECORD

pbstrOutBlock

Returns block of certified addresses that contains calNPUT_BLOCK_RECORD_COUNT addresses that are separated by caDELIMITER_FIELD and caDELIMITER_RECORD

Returns

VB:

Block of certified addresses that contains calNPUT_BLOCK_RECORD_COUNT addresses that are separated by caDELIMITER_FIELD and caDELIMITER_RECORD (as String). The VB Err object will contain the MRTK result code if an error occurs.

C++:

0 if successful, otherwise an MRTK result code

Notes

You may want to experiment with the caINPUT_BLOCK_RECORD_COUNT property. In preliminary tests, we have found the optimal setting to be around 25-50. The number of fields that you want returned (see caFIELD_LIST_OUT property) greatly affects this number. Also, for optimal performance, only ask for fields that you want. The reason for this is that the extra information requires additional lookups that slow processing.

VB Example

```
…
For nIdx = 1 To nLoops
' Get nRecordBlockCount addresses from recordset
   szAddressInBlock = GetAddressBlock
szAddressOutBlock = _ objCABatchTask.Update(szAddressInBlock)
       ' Save updated addresses
   SetAddressBlock szAddressOutBlock
Next nIdx
…
```

See also

See the MRTK Global Properties table for definition of:

caINPUT_BLOCK_RECORD_COUNT

caDELIMITER_FIELD

caDELIMITER_RECORD

caFIELD_LIST_IN

caFIELD_LIST_OUT

See the MRTKCALib Field Names table for definition of:

FLD_SKIPPED_CERTIFY

ReviewErrors

Display review errors window

Syntax

VB:

```
ReviewErrors
```

C++:

```
long ReviewErrors()
```

Arguments

None

Returns

VB:

None. The VB Err object will contain the MRTK result code if an error occurs.

C++:

0 if successful, otherwise an MRTK result code

Notes

When the review errors window is displayed, any addresses that are kept will be returned by the RetrieveReviewed function. Before you can successfully call this function you must set the caBATCH_REVIEW_ERRORS property to True. This function is not currently implemented.

VB Example

```
…
For nIdx = 1 To nLoops
' Get nRecordBlockCount addresses from recordset
   szAddressInBlock = GetAddressBlock
szAddressOutBlock = _ objCABatchTask.Update(szAddressInBlock)
' Save updated addresses
   SetAddressBlock szAddressOutBlock
Next nIdx
' Finished updating addresses so show review
objCABatchTask.ReviewErrors
…
```

See also

See the CABatchTask Properties table for definition of:

caBATCH _REVIEW_ERRORS

RetrieveReviewed

Retrieve the addresses kept by the user in the review errors window

Syntax

VB:

```
RetrieveReviewed
```

C++:

```
long RetrieveReviewed(BSTR *pbstrAddressBlock)
```

Arguments

VB:

None

C++:

pbstrAddressBlock

Returns a string that contains caINPUT_BLOCK_RECORD_COUNT addresses that are separated by caDELIMITER_FIELD and caDELIMITER_RECORD

Returns

VB:

A string that contains caINPUT_BLOCK_RECORD_COUNT addresses that are separated by caDELIMITER_FIELD and caDELIMITER_RECORD (as String). The VB Err object will contain the MRTK result code if an error occurs.

C++:

0 if successful, otherwise an MRTK result code

Notes

Before you can successfully call this function you must set the caBATCH_REVIEW_ERRORS property to True. This function is not currently implemented.

VB Example

```
…
While (bContinue)
szAddressBlock = ""
' Get the addresses kept
szAddressBlock = objCABatchTask.RetrieveReviewed
' Save the addresses
nRecordsReceived = SetAddressBlock(szAddressBlock)

       ' End loop if the number of addresses in
       ' szAddressBlock is less than the number specified
       ' in the SetProperty
If (nRecordsReceived < nReviewBlockCount) Then
' Not a full block so we must be done
bContinue = False
End If
Wend
…
```

See also

See the CABatchTask Properties table for definition of:

caBATCH_REVIEW_ERRORS

ShowBatchWizard

Display the batch wizard

Syntax

VB:

```
ShowBatchWizard
```

C++:

```
long ShowBatchWizard()
```

Arguments

None

Returns

VB:

None. The VB Err object will contain the MRTK result code if an error occurs.

C++:

0 if successful, otherwise an MRTK result code

Notes

Call this function if you want to display the pre-designed Batch Wizard. The Batch Wizard provides a graphical interface that leads a user through the steps necessary to process a mailing list. This function is not currently implemented.

VB Example

```
...
objCABatchTask.ShowBatchWizard
...
```

See also

GetPropertySummary

Returns the display string for a property

Syntax

VB:

```
String GetPropertySummary(MRTKPropertyID)
```

C++:

```
long GetPropertySummary(long MRTKPropertyID, BSTR *pVal)
```

Arguments

VB:

MRTKPropertyID as Long

The property ID of the property for which you want summary information

C++:

MRTKPropertyID

The property ID of the property for which you want summary information

pVal

Returns a BSTR that contains a description of the property specified by MRTKPropertyID

Returns

VB:

A description of the property specified by MRTKPropertyID (as String). The VB Err object will contain the MRTK result code if an error occurs.

C++:

0 if successful, otherwise an MRTK result code

Notes

This function can be used to display the description of a property on screen to the user.

VB Example

```
…
Dim szDescription As String
szDescription = _
objCABatchTask.GetPropertySummary(caBATCH_CERTIFY_FLAG)
…
```

See also

See the CABatchTask Properties table for a list of property IDs

PrintSOA

Prints the Statement of Accuracy report

Syntax

VB:

```
PrintSOA
```

VB:

```
long PrintSOA()
```

Arguments

None

Returns

VB:

None. The VB Err object will contain the MRTK result code if an error occurs.

C++:

0 if successful, otherwise an MRTK result code

Notes

Call this function after CABatchTask.Update.

VB Example

```
…
objCABatchTask.Update
objCABatchTask.SaveReportAsPDF "C:\Reports\SOA.PDF"
objCABatchTask.EndTask
…
```

See also

CABatchTask.PrintPreviewSOA

CABatchTask.SaveSOAAsPDF

PrintPreviewSOA

Displays the Statement of Accuracy report on monitor

Syntax

VB:

```
PrintPreviewSOA
```

C++:

```
long PrintPreviewSOA()
```

Arguments

None

Returns

VB:

None. The VB Err object will contain the MRTK result code if an error occurs.

C++:

0 if successful, otherwise an MRTK result code

Notes

Call this function after CABatchTask.Update.

VB Example

```
…
objCABatchTask.Update
objCABatchTask.SaveReportAsPDF "C:\Reports\SOA.PDF"
objCABatchTask.EndTask
…
```

See also

CABatchTask.PrintSOA

CABatchTask.SaveSOAAsPDF

SaveSOAAsPDF

Save the Statement of Accuracy report as a PDF file

Syntax

VB:

```
SaveSOAAsPDF(bstrFileName)
```

C++:

```
long SaveSOAAsPDF(BSTR bstrFileName,
 BSTR *pbstrOutputFileName)
```

ParametersVB:

bstrFileName as String

The name of the file to save

C++:

bstrFileName

The name of the file to save

pbstrOutputFileName

Returns the name of the saved file

Returns

VB:

The name of the saved file. The VB Err object will contain the MRTK result code if an error occurs.

C++:

0 if successful, otherwise an MRTK result code

Notes

Call this function after CABatchTask.Update.

VB Example

```
…
objCABatchTask.Update
objCABatchTask.SaveReportAsPDF "C:\Reports\SOA.PDF"
objCABatchTask.EndTask
…
```

See also

CABatchTask.PrintSOA

CABatchTask.PrintPreviewSOA

AbortTask

Exit the task early

Syntax

VB:

```
AbortTask
```

C++:

```
long AbortTask()
```

Arguments

None

Returns

VB:

None. The VB Err object will contain the MRTK result code if an error occurs.

C++:

0 if successful, otherwise an MRTK result code

Notes

VB Example

```
For nIdx = 1 To nLoops
' Get nRecordBlockCount addresses from recordset
   szAddressInBlock = GetAddressBlock
szAddressOutBlock = _ objCABatchTask.Update(szAddressInBlock)
       If (bSomeEventHappened = True) Then
               objCABatchTask.AbortTask
               Exit Sub
```

```
        End If
        …
Next nIdx
```

See also

EndTask

Done with the task, clean up and release

Syntax

VB:

```
EndTask
```

C++:

```
long EndTask()
```

Arguments

None

Returns

VB:

None. The VB Err object will contain the MRTK result code if an error occurs.

C++:

0 if successful, otherwise an MRTK result code

Notes

EndTask clears up and releases used resources and should be called when done with the CABatchTask object. It is very important to call EndTask after processing the last batch of addresses before calling CASOAReportTask.ValidateProperties.

VB Example

```
…
objCABatchTask.EndTask
…
```

See also

CABatchTask Properties Summary Table

CABatchTask Properties	Enum Value	Data Type	Default Value	Description
caBATCH_ALL_CAPS	2080	BOOL	FALSE	TRUE = Return all caps FALSE = Return mixed case
caBATCH_CERTIFY_FLAG	2046	Long	0	Indicate whether to skip re-certification for records that have been previously certified based on the CASSDate field: 0 = Check only records not certified with this CD 1 = Check every record Indicate records to include when building the Summary Report based on the CASSDate field: 4 = Certified with this CD All records passed in will be checked to see if they "qualify" to be added to the totals for the Summary Report. NOTE If a record is coded but does not qualify, it will show up in the errors section of the report.
caBATCH_HIDE_PROGRESS_DLG_WHEN_DONE	2062	BOOL	FALSE	TRUE to hide progress dialog after the completion of batch processing
caBATCH_INPUT_RECORD_COUNT	2063	Long	0	The total number of records to be processed
caBATCH_LIST_NAME	2067	String	EMPTY	Mailing list name to show in progress dialog
caBATCH_SHOW_PROGRESS_DLG	2061	BOOL	FALSE	TRUE to show progress dialog during batch processing

The COM CASOAReportTask Object for Creating Postal Reports

The BCC Architect CASOAReportTask object prints the Statement of Accuracy report for the last batch of addresses processed using the CABatchTask object. Therefore, if you wish to print a Statement of Accuracy report you should process a mailing list using the CABatchTask object before calling the CASOAReportTask object. The CASOAReportTask object should be created through the BCC Architect object factory MRTKObjFactory.

CASOAReportTask Functions

PrepareTask

Initialize and prepare the object

Syntax

VB:

```
PrepareTask
```

C++:

```
long PrepareTask()
```

Arguments

None

Returns

VB:

None. The VB Err object will contain the MRTK result code if an error occurs.

C++:

0 if successful, otherwise an MRTK result code

Notes

This function must be called before any of the other functions/properties of CASOAReportTask. Failing to do so will cause subsequent function calls to fail. The one exception to this rule is calling SetProperty to set the path of the Data-file.dat file. It is preferable, however, to set the data file path using the MRTKObjFactory.DataFilePath property.

VB Example

```
Dim MRTKFactory As New MRTKFACTORYLib.MRTKObjFactory
Dim objCASOAReportTask As CASOAReportTask
Set objCASOAReportTask = _
  MRTKFactory.CreateObject(ENTERPRISE_CASOAREPORTTASK)
objCASOAReportTask.PrepareTask
objCASOAReportTask.SetProperty(…)
…
```

See also

BCC Architect Factory Object

CABatchTask.PrepareTask

GetProperty

Get a CASOAReportTask property value

Syntax

VB:

```
GetProperty(MRTKPropertyID)
```

C++:

```
long GetProperty(long MRTKPropertyID, VARIANT *pVal)
```

Arguments

VB:

MRTKPropertyID as Long

The property ID of the property to get

C++:

MRTKPropertyID

The property ID of the property to get

pVal

Returns the value of the

Returns

VB:

The value of the property (as Variant). The VB Err object will contain the MRTK result code if an error occurs.

C++:

0 if successful, otherwise an MRTK result code

Notes

VB Example

```
…
Dim szPrinterName as String
' Get printer used to print form
szPrinterName = _
  objCASOAReportTask.GetProperty(caBATCH_PRINTER_NAME)
…
```

See also

See the CASOAReportTask Properties table for a list of property ID's

SetProperty

Set a CASOAReportTask property

Syntax

VB:

```
SetProperty(MRTKPropertyID, value)
```

C++:

```
long SetProperty(long MRTKPropertyID, VARIANT value)
```

Arguments

VB:

MRTKPropertyID as Long

The property ID of the property to set

value as Variant

The value of the property to set

C++:

MRTKPropertyID

The property ID of the property to set

value

The value of the property to set

Returns

VB:

None. The VB Err object will contain the MRTK result code if an error occurs.

C++:

0 if successful, otherwise an MRTK result code

Notes

You must set the customer name and list name properties with this function.

VB Example

```
Dim szPrinterName As String
szPrinterName = "\\Server\HP LaserJet 5P/5MP (HP)"
objCASOAReportTask.PrepareTask
objCASOAReportTask.SetProperty caBATCH_PRINTER_NAME, _
 szPrinterName
…
objCASOAReportTask.ValidateProperties
```

See also

See the CASOAReportTask Properties table for a list of property ID's

ValidateProperties

Verify that the task is set up correctly and ready to run

Syntax

VB:

```
ValidateProperties
```

C++:

```
long ValidateProperties()
```

Arguments

None

Returns

VB:

None. The VB Err object will contain the MRTK result code if an error occurs.

C++:

0 if successful, otherwise an MRTK result code

Notes

This function needs to be called before calling CASOAReportTask.PrintSOA or CASOAReportTask.PrintPreviewSOA. It is very important to call CABatchTask.EndTask, after processing the last batch of addresses, before calling CASOAReportTask.ValidateProperties.

VB Example

```
…
On Error GoTo errValidateProperties
objCASOAReportTask.ValidateProperties
errValidateProperties:
' Handle errors
…
```

See also

PrintSOA

Prints the Statement of Accuracy report

Syntax

VB:

```
PrintSOA
```

C++:

```
long PrintSOA()
```

Arguments

None

Returns

VB:

None. The VB Err object will contain the MRTK result code if an error occurs.

C++:

0 if successful, otherwise an MRTK result code

Notes

This function should be called after CASOAReportTask.ValidateProperties. It is very important to call CABatchTask.EndTask, after processing the last batch of addresses, before calling CASOAReportTask.ValidateProperties.

VB Example

```
…
On Error GoTo errValidateProperties
objCASOAReportTask.ValidateProperties
' Print Statement of Accuracy Report
objCASOAReportTask.PrintSOA
errValidateProperties:
' Handle error
…
```

See also

CASOAReportTask.ValidateProperties

PrintPreviewSOA

Displays the Statement of Accuracy report on monitor

Syntax

VB:

```
PrintPreviewSOA
```

C++:

```
long PrintPreviewSOA()
```

Arguments

None

Returns

VB:

None. The VB Err object will contain the MRTK result code if an error occurs.

C++:

0 if successful, otherwise an MRTK result code

Notes

This function should be called after CASOAReportTask.ValidateProperties. It is very important to call CABatchTask.EndTask, after processing the last batch of addresses, before calling CASOAReportTask.ValidateProperties.

VB Example

```
…
On Error GoTo errValidateProperties
objCASOAReportTask.ValidateProperties
' Print preview Statement of Accuracy Report
objCASOAReportTask.PrintPreviewSOA
errValidateProperties:
' Handle error
…
```

See also

CASOAReportTask.ValidateProperties

EndTask

Done with the task, clean up and release

Syntax

VB:

```
EndTask
```

C++:

```
long EndTask()
```

Arguments

None

Returns

VB:

None. The VB Err object will contain the MRTK result code if an error occurs.

C++:

0 if successful, otherwise an MRTK result code

Notes

EndTask clears up and releases used resources and should be called when done with the CASOAReportTask object.

VB Example

```
...
objCASOAReportTask.EndTask
...
```

See also

CASOAReportTask Properties Summary Table

CASOAReportTask Properties	Enum Value	Data Type	Default Value	Description
caBATCH_CUSTOMER_ADDRESS	2002	String	EMPTY	Customer address to print on SOA
caBATCH_CUSTOMER_COMPANY	2006	String	EMPTY	Customer company to print on SOA
caBATCH_CUSTOMER_CPC_NUMBER	2001	String	EMPTY	Customer CPC number to print on SOA
caBATCH_CUSTOMER_MUNICIPALITY	2003	String	EMPTY	Customer municipality to print on SOA
caBATCH_CUSTOMER_NAME	2000	String	EMPTY	Customer name to print on SOA
caBATCH_CUSTOMER_POSTALCODE	2005	String	EMPTY	Customer postal code to print on SOA
caBATCH_CUSTOMER_PROVINCE	2004	String	EMPTY	Customer province to print on SOA

caBATCH_PRINTER_NAME	2050	String	Default Printer	Name of printer to send the SOA report to. Note: If the printer is a network printer the name must be the full name (e.g., \\Server\HP Laser Jet 5). To find the full printer name, you could print a test page from the print driver.
caBATCH_SHOW_PRINT_DIALOG	2049	BOOL	FALSE	TRUE to show the standard print dialog

The COM CASortTaskObject for Presorting Mailings

The CASortTask allows BCC Architect to provide sortation for Canadian postal addresses. This feature expands the BCC Architect repertoire from only USPS presorting to include Canadian postal sorting as well. The CASortTaskoffers a flexible interface for Canadian mail sorting. A programmer can specify the number of records sent and returned, how user interface windows are displayed, and several other properties (see the CASortTaskProperties table for more information). In addition to mail sorting, CASortTaskalso offers a Presort Wizard that gathers data relevant to a user's particular mail sort.

CASortTaskObject Functions

PrepareTask

Initialize and prepare the object

Syntax

VB:

```
PrepareTask
```

C++:

```
long PrepareTask()
```

Arguments

None

Returns

VB:

None. The VB Err object will contain the MRTK result code if an error occurs.

C++:

0 if successful, otherwise an MRTK result code

Notes

This function must be called before any of the other functions/properties of CASortTask. Failing to do so will cause subsequent function calls to fail.

VB Example

```
Dim MRTKFactory As New MRTKFACTORYLib.MRTKObjFactory
Dim objCASortTaskAs CASortTask
Set objCASortTask= _ MRTKFactory.CreateObject(ENTERPRISE_CASORTTASK)
objCASortTask.PrepareTask
objCASortTask.SetProperty(…)
…
```

See also

GetProperty

Get a CASortTaskproperty value

Syntax

VB:

```
GetProperty(MRTKPropertyID)
```

C++:

```
long GetProperty(long MRTKPropertyID, VARIANT *pVal)
```

Arguments

VB:

MRTKPropertyID as Long

The property ID of the property to get

C++:

MRTKPropertyID

The property ID of the property to get

pVal

Returns the value of the

Returns

VB:

The value of the property (as Variant). The VB Err object will contain the MRTK result code if an error occurs.

C++:

0 if successful, otherwise an MRTK result code

Notes

VB Example

```
…
Dim nPieceWeight As Long
' Get the piece weight
nPieceWeight = _
 objCASortTask.GetProperty(caSORT_PIECE_WEIGHT)
…
…
```

See also

See the CASortTaskProperties table for a list of property ID's

SetProperty

Set a CASortTaskproperty

Syntax

VB:

```
SetProperty(MRTKPropertyID, value)
```

C++:

```
long SetProperty(long MRTKPropertyID, VARIANT value)
```

Arguments

VB:

MRTKPropertyID as Long

The property ID of the property to set

value as Variant

The value of the property to set

C++:

MRTKPropertyID

The property ID of the property to set

value

The value of the property to set

Returns

VB:

None. The VB Err object will contain the MRTK result code if an error occurs.

C++:

0 if successful, otherwise an MRTK result code

Notes

VB Example

```
objCASortTask.PrepareTask
…
objCASortTask.SetProperty caSORT_CERTIFY_ADDRESSES_FIRST, True
…
objCASortTask.ValidateProperties
…
```

See also

See the CASortTaskProperties table for a list of property ID's

ValidateProperties

Verify that the task is set up correctly and ready to run

Syntax

VB:

```
ValidateProperties
```

C++:

```
long ValidateProperties()
```

Arguments

None

Returns

VB:

None. The VB Err object will contain the MRTK result code if an error occurs.

C++:

0 if successful, otherwise an MRTK result code

Notes

As a minimum, the input field list for CASortTaskmust contain the postal code of the address to sort.

VB Example

```
…
```

```
On Error GoTo errValidateProperties
objCASortTask.ValidateProperties
errValidateProperties:
' Handle error
MsgBox Err.Description
…
```

See also

See the MRTKCALib Field Names table for definition of:

FLD_POSTALCODE

Send

Send address block to CASortTask

Syntax

VB:

```
Send(strAddressBlock)
```

C++:

```
long Send(BSTR *pbstrAddressBlock)
```

Arguments

VB:

strAddressBlock as String

A string that contains calNPUT_BLOCK_RECORD_COUNT addresses that are separated by caDELIMITER_FIELD and caDELIMITER_RECORD

C++:

pbstrAddressBlock

A BSTR that contains calNPUT_BLOCK_RECORD_COUNT addresses that are separated by caDELIMITER_FIELD and caDELIMITER_RECORD

Returns

VB:

None. The VB Err object will contain the MRTK result code if an error occurs.

C++:

0 if successful, otherwise an MRTK result code

Notes

You may want to experiment with the number of records in an input block as specified by the calNPUT_BLOCK_RECORD_COUNT property. In preliminary tests, we have found the optimal setting to be around 25-50. You should call

CASortTask.DoSort after you have finished sending all of your records to the CASortTaskobject.

VB Example

```
…
On Error GoTo errCASort
For nIdx = 1 To nLoops
       ' Get nRecordBlockCount addresses from recordset
   szAddressBlock = GetAddressBlock
objCASortTask.Send szAddressBlock
Next nIdx
…
objCASortTask.DoSort
…
errCASort:
       MsgBox Err.Description
```

See also

CASortTask.DoSort, CASortTask.ShowPresortWizard functions

DoSort

Perform chosen presort on records sent to CASortTask

Syntax

VB:

```
DoSort
```

C++:

```
long DoSort()
```

Arguments

None

Returns

VB:

None. The VB Err object will contain the MRTK result code if an error occurs.

C++:

0 if successful, otherwise an MRTK result code

Notes

You should call this function after your final call to CASortTask.Send.

VB Example

```
…
For nIdx = 1 To nLoops
' Get nRecordBlockCount addresses from recordset
   szAddressBlock = GetAddressBlock
   objCASortTask.Send szAddressBlock
Next nIdx
…
objCASortTask.DoSort
…
```

See also

CASortTask.Send

CASortTask.ShowPresortWizard

See the CASortTaskProperties table for definition of:

caSORT_SHOW_PROGRESS_DLG

Retrieve

Retrieve sorted address block from CASortTask

Syntax

VB:

```
Retrieve(strAddressBlock)
```

C++:

long Retrieve(BSTR *pbstrAddressBlock)

Arguments

VB:

StrAddressBlock as String

A string that contains calNPUT_BLOCK_RECORD_COUNT addresses that are separated by caDELIMITER_FIELD and caDELIMITER_RECORD

C++:

pbstrAddressBlock

A string that contains calNPUT_BLOCK_RECORD_COUNT addresses that are separated by caDELIMITER_FIELD and caDELIMITER_RECORD

Returns

VB:

None. The VB Err object will contain the MRTK result code if an error occurs.

C++:

0 if successful, otherwise an MRTK result code

Notes

Retrieve fields specified by the caFIELD_LIST_OUT property. Addresses will be certified prior to sorting if the caSORT_CERTIFY_ADDRESSES_FIRST property is set to True with the SetProperty function.

VB Example

```
…
For nIdx = 1 To nLoops
objCASortTask.Retrieve szAddressBlock
' Save presort info
       SetAddressBlock szAddressBlock
Next nIdx
…
```

See also

See the MRTK Global Properties table for definition of:

caFIELD_LIST_OUT

See the CASortTaskProperties table for definition of:

caSORT_CERTIFY_ADDRESSES_FIRST

ShowPresortWizard

Display dialog to step user through presort setup

Syntax

VB:

```
ShowPresortWizard
```

C++:

```
long ShowPresortWizard()
```

Arguments

None

Returns

VB:

None. The VB Err object will contain the MRTK result code if an error occurs.

C++:

0 if successful, otherwise an MRTK result code

Notes

This function should be called before calling CASortTask.Send. The wizard needs to be called only the first time a mail sort is performed. Using the wizard, the user can specify mail sort options and save these options as a template. This template can then be accessed with the MRTK global property caTEMPLATE_NAME_TO_USE. The caSORT_ENABLE_BATCH_PROCESSING property must be set to True if you want to print or preview the Statement of Accuracy report.

VB Example

```
…
objCASortTask.ShowPresortWizard
…
```

See also

See the MRTK Global Properties table for definition of:

- caTEMPLATE_NAME_TO_USE

See the CASortTaskProperties table for definition of:

- caSORT_ENABLE_BATCH_PROCESSING

GetPropertySummary

Returns the display string for a

Syntax

VB:

```
GetPropertySummary(MRTKPropertyID)
```

C++:

```
long GetPropertySummary(long MRTKPropertyID, BSTR *pVal)
```

Arguments

VB:

MRTKPropertyID as Long

- The property ID of the property for which you want summary information

C++:

MRTKPropertyID

- The property ID of the property for which you want summary Information

pVal

Returns a BSTR that contains a description of the property specified by MRTKPropertyID

Returns

VB:

A description of the property specified by MRTKPropertyID (as String). The VB Err object will contain the MRTK result code if an error occurs.

C++:

0 if successful, otherwise an MRTK result code

Notes

This function can be used to display the description of a property on screen to the user.

VB Example

```
…
Dim szCompanyName As String
szCompanyName = _
objCASortTask.GetPropertySummary(caSORT_CUSTOMER_NAME)
…
```

See also

PrintReport

Prints a postal report

Syntax

VB:

```
PrintReport(nCAReportID, bstrPrinterName,
bShowPrintSetupDlg)
```

C++:

```
long PrintReport(long nCAReportID,
BSTR bstrPrinterName, long bShowPrintSetupDlg)
```

Arguments

VB:

nCAReportID as Long

- The ID of the report you wish to print

bstrPrinterName as String

- The name of the printer you wish to print a report to

bShowPrintSetupDlg as Long

- 1 to show the printer setup dialog or 0 to hide this dialog

C++:

nCAReportID

- The ID of the report you wish to print

bstrPrinterName

- The name of the printer you wish to print a report to

bShowPrintSetupDlg

- 1 to show the printer setup dialog or 0 to hide this dialog

Returns

VB:

None. The VB Err object will contain the MRTK result code if an error occurs.

C++:

0 if successful, otherwise an MRTK result code

Notes

If “bstrPrinterName” is set to an empty string (“”) then the default printer will be used to print the indicated form. Also, if “bShowPrintSetupDlg” is set to 1 then “bstrPrinterName” will be ignored. Setting "nUKReportID" equal to caSORT_PRINT_ALL_REPORTS will tell CASortTask.PrintReport to print all of the postal reports. Calling this function will override the selections made within the Presort Wizard. To print the reports specified in the Presort Wizard, call PrintPostalReports.

VB Example

```
…
If objCASortTask.GetProperty(caSORT_PRINT_TIEONTAGS)Then
   objCASortTask.PrintReport caSORT_PRINT_TIEONTAGS, _
"", False
End If…
```

See also

CASortTask.PrintPostalReports

See the CASortTaskProperties table for definition of:

- caSORT_PRINT_ACCURACY_REPORT
- caSORT_PRINT_BUNDLE_SLIPS
- caSORT_PRINT_CONTAINER_LABELS
- caSORT_PRINT_PACKING_REPORT
- caSORT_PRINT_SUMMARY_REPORT

- caSORT_PRINT_TIEONTAGS
- caSORT_PRINT_ALL_REPORTS

PreviewReport

Displays a postal form on monitor

Syntax

VB:

```
PreviewReport(nCAPresortID)
```

C++:

```
long PreviewReport(long nCAPresortID)
```

Arguments

VB:

nCAPresortID as Long

- The ID of the report you wish to preview

C++:

nCAPresortID

- The ID of the report you wish to preview

Returns

VB:

None. The VB Err object will contain the MRTK result code if an error occurs.

C++:

0 if successful, otherwise an MRTK result code

Notes

Each call to this function will create a new preview window for the specified report. Hence, multiple calls to PreviewReport will result in multiple preview windows being displayed. Calling this function will override the selections made within the Presort Wizard. To preview the reports specified in the Presort Wizard, call PreviewPostalReports.

VB Example

```
…
objCASortTask.PreviewReport caSORT_PRINT_TIEONTAGS
…
```

See also

CASortTask.PreviewPostalReports

See the CASortTaskProperties table for definition of:

- caSORT_PRINT_ACCURACY_REPORT
- caSORT_PRINT_BUNDLE_SLIPS
- caSORT_PRINT_CONTAINER_LABELS
- caSORT_PRINT_PACKING_REPORT
- caSORT_PRINT_SUMMARY_REPORT
- caSORT_PRINT_TIEONTAGS
- caSORT_PRINT_ALL_REPORTS

SaveReportAsPDF

Save a postal presort report as a PDF file

Syntax

VB:

```
SaveReportAsPDF(MRTKReportID, bstrFileName)
```

C++:

```
long SaveReportAsPDF(long MRTKReportID,
BSTR bstrFileName, BSTR *pbstrOutputFileName)
```

ParametersVB:

MRTKReportID as Long

- The ID of the report you wish to preview

bstrFileName as String

- The name of the file to save

C++:

MRTKReportID

- The ID of the report you wish to preview

bstrFileName

- The name of the file to save

pbstrOutputFileName

- Returns the name of the saved file

Returns

VB:

The name of the saved file. The VB Err object will contain the MRTK result code if an error occurs.

C++:

0 if successful, otherwise an MRTK result code

Notes

Calling this function will override the selections made within the Presort Wizard. To save the reports specified in the Presort Wizard, call SaveReportsAsPDF.

VB Example

```
…
If objCASortTask.GetProperty(caSORT_PRINT_TIEONTAGS)Then
objCASortTask.SaveReportAsPDF caSORT_PRINT_TIEONTAGS,
 "C:\Reports\Presort Reports.PDF"
End If
…
```

See also

CASortTask.PreviewPostalReports

See the CASortTaskProperties table for definition of:

- caSORT_PRINT_ACCURACY_REPORT
- caSORT_PRINT_BUNDLE_SLIPS
- caSORT_PRINT_CONTAINER_LABELS
- caSORT_PRINT_PACKING_REPORT
- caSORT_PRINT_SUMMARY_REPORT
- caSORT_PRINT_TIEONTAGS
- caSORT_PRINT_ALL_REPORTS

PrintPresortReports

Prints the reports specified in the CASortTask properties

Syntax

VB:

```
PrintPresortReports
```

C++:

```
long PrintPresortReports()
```

Arguments

None

Returns

VB:

None. The VB Err object will contain the MRTK result code if an error occurs.

C++:

0 if successful, otherwise an MRTK result code

Notes

This function prints the reports selected within the Presort Wizard. Unless the programmer changes the relevant properties, by calling PrintReport, a call to this function will print the reports that were selected the last time the Presort Wizard was run.

VB Example

```
…
' Set reports we want to print
objCASortTask.SetProperty caSORT_PRINT_BUNDLE_SLIPS, True
objCASortTask.SetProperty caSORT_PRINT_TIEONTAGS, True
objCASortTask.PrintPresortReports
…
```

See also

CASortTask.PrintReport

PreviewPresortReports

Preview the reports specified in the CASortTask properties

Syntax

VB:

```
PreviewPresortReports
```

C++:

```
long PreviewPresortReports()
```

Arguments

None

Returns

VB:

None. The VB Err object will contain the MRTK result code if an error occurs.

C++:

0 if successful, otherwise an MRTK result code

Notes

This function previews the reports selected within the Presort Wizard. Unless the programmer changes the relevant properties, by calling PreviewReport, a call to this function will preview the reports that were selected the last time the Presort Wizard was run. If the SOA report was selected, it will be shown in a second preview window after the presort reports.

VB Example

```
…
' Set reports we want to preview
objCASortTask.SetProperty caSORT_PRINT_BUNDLE_SLIPS, True
objCASortTask.SetProperty caSORT_PRINT_TIEONTAGS, True
objCASortTask.PreviewPresortReports
…
```

See also

CASortTask.PreviewReport

SaveReportsAsPDF

Save the reports specified in the presort task properties

Syntax

VB:

```
SaveReportsAsPDF(bstrFileName)
```

C++:

```
long SaveReportsAsPDF(BSTR bstrFileName, BSTR *pbstrOutputFileName)
```

ParametersVB:

bstrFileName as String

The name of the file to save

C++:

bstrFileName

The name of the file to save

pbstrOutputFileName

Returns the name of the saved file

Returns

VB:

The name of the saved file. The VB Err object will contain the MRTK result code if an error occurs.

C++:

0 if successful, otherwise an MRTK result code

Notes

This function saves the reports selected within the Presort Wizard or whose properties have been set to True with SetProperty. Unless the programmer changes the relevant properties, for example, by calling PreviewReport, a call to this function will save all reports that were selected the last time the Presort Wizard was run or whose properties were last set to True. Setting caSORT_PRINT_ALL_REPORTS equal to True will set the properties of all reports to True and hence a call to this function will save all reports. The reports will all be saved as a single file.

VB Example

```
…
' Set print properties to known state, in this case False
objCASortTask.SetProperty caSORT_PRINT_ALL_REPORTS, False
' Set reports we want to save
objCASortTask.SetProperty caSORT_PRINT_ACCURACY_REPORT, True
objCASortTask.SetProperty caSORT_PRINT_SUMMARY_REPORT, True
objCASortTask.SaveReportsAsPDF "C:\Reports\Presort Reports.PDF"
…
```

See also

SaveReportAsPDF

See the CASortTaskProperties table for definition of:

- caSORT_PRINT_ACCURACY_REPORT
- caSORT_PRINT_BUNDLE_SLIPS
- caSORT_PRINT_CONTAINER_LABELS
- caSORT_PRINT_PACKING_REPORT
- caSORT_PRINT_SUMMARY_REPORT
- caSORT_PRINT_TIEONTAGS
- caSORT_PRINT_ALL_REPORTS

EndTask

Done with the task, clean up and release

Syntax

VB:

```
EndTask
```

C++:

```
long EndTask()
```

Arguments

None

Returns

VB:

None. The VB Err object will contain the MRTK result code if an error occurs.

C++:

0 if successful, otherwise an MRTK result code

Notes

EndTask clears up and releases used resources and should be called when done with the CASortTaskobject.

VB Example

```
...
objCASortTask.EndTask
...
```

See also

CASortTask Properties Summary Table

CASortTaskProperties	Enum Value	Data Type	Default Value	Description
caSORT_BUNDLESLIPS_ BOTTOM_ MARGIN	2058	Long	0	Vertical space between bundle slips in mm
caSORT_BUNDLESLIPS_ COLUMNS	2053	Long	0	Number of bundle slip columns per page
caSORT_BUNDLESLIPS_ CONTINUOUS	2059	BOOL	FALSE	TRUE if using continuous paper (dot matrix printer) for bundle slips
caSORT_BUNDLESLIPS_H_ PAGEOFFSET	2055	Long	0	Horizontal page offset for bundle slips in mm
caSORT_BUNDLESLIPS_ RIGHT_MARGIN	2057	Long	0	Horizontal space between bundle slips in mm
caSORT_BUNDLESLIPS_ROWS	2054	Long	0	Number of bundle slip rows per page
caSORT_BUNDLESLIPS_V_ PAGEOFFSET	2056	Long	0	Vertical page offset for bundle slips in mm
caSORT_CERTIFY_ ADDRESSES_FIRST	2048	BOOL	FALSE	TRUE to perform address correction prior to mail sort

caSORT_CONTAINERLABELS_ BOTTOM_MARGIN	2034	Long	0	Vertical space between labels in mm
caSORT_CONTAINERLABELS_ COLUMNS	2036	Long	0	Number of label columns per page
caSORT_CONTAINERLABELS _ CONTINUOUS	2042	BOOL	FALSE	TRUE if using continuous paper (dot matrix printer) for labels
caSORT_CONTAINERLABELS_ H_ PAGEOFFSET	2040	Long	0	Horizontal page offset for labels in mm
caSORT_CONTAINERLABELS_ RIGHT_ MARGIN	2035	Long	0	Horizontal space between labels in mm
caSORT_CONTAINERLABELS_ ROWS	2037	Long	0	Number of label rows per page
caSORT_CONTAINERLABELS_ V_ PAGEOFFSET	2041	Long	0	Vertical page offset for labels in mm
caSORT_CONTAINER_TYPE	2008	String	NULL	Type of container
caSORT_CUSTOMER_ ADDRESS	2030	String	NULL	Address of company performing the mailing
caSORT_CUSTOMER_ COMPANY	2047	String	NULL	Name of company performing the mailing
caSORT_CUSTOMER_ID	2028	String	NULL	ID number of company performing the mailing
caSORT_CUSTOMER_NAME	2029	String	NULL	Name of customer performing the mailing
caSORT_CUSTOMER_ POSTALCODE	2079	String	NULL	Postal code of company performing the mailing
caSORT_CUSTOMER_ TELEPHONE	2031	String	NULL	Telephone number of company performing the mailing
caSORT_DELIVERY_OFFICE	2032	String	NULL	Delivery office
caSORT_ENABLE_BATCH_ PROCESSING	2077	BOOL	FALSE	TRUE to enable SOA report in Presort Wizard
caSORT_FOLDER_NAME_ ELECTRONIC_PLAN	2105	String	EMPTY	Path to the folder in which to save the Electronic Mailing Plan file
caSORT_HIDE_PROGRESS_ DLG_WHEN_DONE	2065	BOOL	FALSE	TRUE to hide progress dialog after the presort is complete
caSORT_MAILING_DATE	2007	String	EMPTY	Date of mailing
caSORT_MAILING_TYPE	2044	String	EMPTY	Type of mailing
caSORT_PIECE_DEPTH	2011	Float	0	Thickness of the mail piece in mm
caSORT_PIECE_HEIGHT	2009	Long	0	Length of the mail piece in mm

caSORT_PIECE_WEIGHT	2012	Long	0	Weight of the mail piece in grams
caSORT_PIECE_WIDTH	2010	Long	0	Width of the mail piece in mm
caSORT_PRESORT_WIZARD_ CAPTION	2045	String	"Presort Wizard"	Caption that appears on each of the Presort Wizard screens
caSORT_PRINT_ACCURACY_ REPORT	2023	BOOL	FALSE	TRUE if SOA report should be printed
caSORT_PRINT_ALL_REPORTS	2078	BOOL	FALSE	TRUE if all reports should be printed
caSORT_PRINT_BUNDLE_SLIPS	2051	BOOL	FALSE	TRUE if bundle slips should be printed
caSORT_PRINT_ CONTAINERLABELS	2020	BOOL	FALSE	TRUE to print container labels
caSORT_PRINT_PACKING_ REPORT	2021	BOOL	FALSE	TRUE if packing report should be printed
caSORT_PRINT_SUMMARY_ REPORT	2022	BOOL	FALSE	TRUE if summary report should be printed
caSORT_PRINT_TIEONTAGS	2075	BOOL	FALSE	TRUE to print tie-on tags
caSORT_PRINTER_ ACCURACY_REPORT	2027	String	NULL	Name of printer for SOA report
caSORT_PRINTER_BUNDLE_ SLIPS	2052	String	NULL	Name of printer for bundle slips
caSORT_PRINTER_ CONTAINER_LABELS	2024	String	EMPTY	Name of printer for container labels
caSORT_PRINTER_PACKING_ REPORT	2025	String	NULL	Name of printer for packing report
caSORT_PRINTER_SUMMARY_ REPORT	2026	String	NULL	Name of printer for summary report
caSORT_PRINTER_TIEONTAGS	2076	String	EMPTY	Name of printer for tie-on tags
caSORT_PUB_MAIL_DEPOSIT_ POSTAL_ CODE	2103	String	EMPTY	Postal code of the office of deposit for a Public-ations Mail sort
caSORT_SAVE_ELECTRONIC_ PLAN	2104	BOOL	FALSE	TRUE to save the Electronic Mailing Plan file. You need to call PrintPresortReports, PreviewPresortReports or SaveReportsAsPDF to actually create the Electronic Mailing Plan file.
caSORT_SHOW_PRINT_ SETUP_DIALOG	2060	BOOL	FALSE	TRUE to show the standard print dialog
caSORT_SHOW_PROGRESS_ DLG	2064	BOOL	FALSE	TRUE to show the presort progress dialog

caSORT_SOM_ID	2033	String	EMPTY	SOM ID
caSORT_TIEONTAGS_ BOTTOM_MARGIN	2068	Long	0	Vertical space between tie-on tags in mm
caSORT_TIEONTAGS_ COLUMNS	2070	Long	0	Number of tie-on tag columns per page
caSORT_TIEONTAGS_ CONTINUOUS	2074	BOOL	FALSE	TRUE if using continuous paper (dot matrix printer) for tie-on tags
caSORT_TIEONTAGS_H_ PAGEOFFSET	2072	Long	0	Horizontal page offset for tie-on tags in mm
caSORT_TIEONTAGS_RIGHT_ MARGIN	2069	Long	0	Horizontal space between tie-on tags in mm
caSORT_TIEONTAGS_ROWS	2071	Long	0	Number of tie-on tag rows per page
caSORT_TIEONTAGS_V_ PAGEOFFSET	2073	Long	0	Vertical page offset for tie-on tags in mm

COM ReportIDs

CAReportIDs	Enum Value	Description
caREPORT_ACCURACY_REPORT	2023	Specifies printing the Accuracy Report.
caREPORT_BUNDLE_SLIPS	2051	Specifies printing the Bundle Facing Slips.
caREPORT_CARDS	2088	Specifies printing the cards.
caREPORT_PRINTER_CARDS	2089	Specifies the printer for the cards.
caREPORT_CONTAINERLABELS	2020	Specifies printing the Container Labels.
caREPORT_FILE_ELECTRONIC_ PLAN	2106	Specifies saving the Electronic Mailing Plan file to the specified folder.
caREPORT_KEEPERTAGS	2097	Specifies printing the Keeper Tags.
caREPORT_PRINTER_ KEEPERTAGS	2098	Specifies the printer for the Keeper Tags.
caREPORT_PACKING_REPORT	2021	Specifies printing the Packing Report.
caREPORT_SUMMARY_REPORT	2022	Specifies printing the Summary Report.
caREPORT_TIEONTAGS	2075	Specifies printing the Tie-on Tags.

COM Field Names

Below are all the field names for an input or output record

Notes

The Field IDs below are concatenated together with a delimiter and passed to the SetProperty function. These Field IDs specify which fields will be passed in or returned from a particular task.

Field ID	Data Direction	Data Type Returned	Description
FLD_ADDRESS_LANGUAGE	In/Out	String	E for English or F for French
FLD_ADDRESSLINE1	In/Out	String	The first address line directly above Municipality; this is a required input field
FLD_ADDRESSLINE2	In/Out	String	The second address line
FLD_BUNDLE_ID	Out	Long	Bundle number
FLD_BUSINESS	In/Out	String	Business name
FLD_CASSDATE	In/Out	Long	Date address was last certified
FLD_CONTAINER_ID	Out	Long	Container number
FLD_COUNTRY	In/Out	String	Country
FLD_DELETE_RECORD_FLAG	Out	Bool	Not currently implemented
FLD_DELIVERY_MODE	In/Out	String	Code indicating type of delivery and delivery route
FLD_ERRORCODE	Out	Long	Error code returned from address certification process
FLD_FIRST_NAME	In/Out	String	First name
FLD_GD_KEYWORD	In/Out	String	General delivery type
FLD_LAST_NAME	In/Out	String	Last name
FLD_LONG_ERROR_STRING	Out	String	Extended description of error code
FLD_LVR_BRANCH_NAME	In/Out	String	Large volume receiver branch name
FLD_LVR_NAME	In/Out	String	Large volume receiver name
FLD_MUNICIPALITY	In/Out	String	Municipality
FLD_NONSTANDARD_EXTRA	In/Out	String	Nonstandard extra information
FLD_PIECE_POSTAGE	In/Out	Float	Piece postage amount
FLD_POBOX_KEYWORD	In/Out	String	PO Box type

FLD_POBOX_ NUMBER	In/Out	String	PO Box number
FLD_POST_ DIRECTIONAL	In/Out	String	Street direction
FLD_POSTALCODE	In/Out	String	Postal code - this is a required input field
FLD_PRESORT_ID	Out	String	The rank order of a mail piece in a particular sort
FLD_PRIMARY_ NUMBER	In/Out	String	Civic number
FLD_PROVINCE	In/Out	String	Province
FLD_PUB_MAIL_ DISTANCE_CODE	Out	String	Returns a single character that indicates the distance for publication mailings. Returns one of the following: R – Regional N – National L – Local
FLD_RECORD_ID	In/Out	String	User field that can contain the input record's index or ID
FLD_RECORD_TYPE	Out	String	Address correction result code. Returns one of the following: C—Corrected N—Not correctable V—Valid
FLD_RURAL_ KEYWORD	In/Out	String	Route type
FLD_RURAL_ NUMBER	In/Out	String	Route number
FLD_SHORT_ ERROR_STRING	Out	String	Standard description of error code
FLD_SKIPPED_ CERTIFY	Out	Bool	True if record was skipped during certification
FLD_STANDARD_ EXTRA	In/Out	String	Standard extra information
FLD_STN_NAME	In/Out	String	Station name
FLD_STN_QUALIFIER	In/Out	String	Station qualifier
FLD_STN_TYPE	In/Out	String	Station type
FLD_STREET_NAME	In/Out	String	Street name
FLD_STREET_TYPE	In/Out	String	Street type
FLD_SUFFIX	In/Out	String	Civic number suffix

FLD_UNIT_ DESIGNATOR	In/Out	String	Unit designator
FLD_UNIT_NUMBER	In/Out	String	Unit number
FLD_USER_ DEFINED_1	In/Out	String	A user defined field
FLD_USER_ DEFINED_10	In/Out	String	A user defined field
FLD_USER_ DEFINED_11	In/Out	String	A user defined field
FLD_USER_ DEFINED_12	In/Out	String	A user defined field
FLD_USER_ DEFINED_13	In/Out	String	A user defined field
FLD_USER_ DEFINED_14	In/Out	String	A user defined field
FLD_USER_ DEFINED_15	In/Out	String	A user defined field
FLD_USER_ DEFINED_2	In/Out	String	A user defined field
FLD_USER_ DEFINED_3	In/Out	String	A user defined field
FLD_USER_ DEFINED_4	In/Out	String	A user defined field
FLD_USER_ DEFINED_5	In/Out	String	A user defined field
FLD_USER_ DEFINED_6	In/Out	String	A user defined field
FLD_USER_ DEFINED_7	In/Out	String	A user defined field
FLD_USER_ DEFINED_8	In/Out	String	A user defined field
FLD_USER_ DEFINED_9	In/Out	String	A user defined field

COM Result Codes

Below are the result codes currently returned from each task

Notes

The Result Codes below indicate the cause of an error encountered when performing a specific Task's operations. The codes are associated with a string that gives a more detailed explanation of the error. There is a C++ header file within the `C:\Program Files (x86)\Satori Software\Satori Architect\Developer Center (Canada)\Sample Code\client\VC++ 6.0 Samples\Canada\CAPostCodeTask Sample` folder if you would like to include these Result Codes within your C\C++ project. The VB Err object will contain the MRTK result code if an error occurs in a Visual Basic project.

Result Code	Hex Value	Description
E_MRTK_ GENERAL_FAIL	0x80040400L	Indicates a general error such as an inability to initialize the CABatchTask object.
E_MRTK_NOT_ IN_BATCH_MODE	0x80040401L	Indicates that a batch record process was attempted when not in batch mode.
E_MRTK_ MATCHING_ LOAD_FAILED	0x80040402L	Indicates that the SERP Matching Engine could not be loaded.
E_MRTK_ ENGINE_NOT_ INITIALIZED	0x80040403L	Indicates that the SERP Matching Engine could not be initialized.
E_MRTK_ BROWSER_NOT_ INITIALIZED	0x80040404L	Indicates that the Postal Code Browser could not be initialized.
E_MRTK_ INTERFACE_ NOT_AVAILABLE	0x80040405L	Indicates that an interface cannot be created or used. This can occur if a user has a Small Business Edition of BCC Architect, but they attempt to use Architect features such as CASortTask.
E_MRTK_OUT_ OF_MEMORY	0x80040406L	Indicates that not enough memory was available to allocate an object or interface.
E_MRTK_NULL_ ARG	0x80040407L	Indicates that a NULL was used as an argument to a function that requires non-NULL Arguments. For example, calling the CASortTask.Send function with a NULL argument.
E_MRTK_ INVALID_ARG	0x80040408L	Indicates that a function argument was not in the correct range, etc. For instance, calling CASortTask.Retrieve after setting the number of records to retrieve to a negative number.
E_MRTK_ INTERFACE_ FAILED	0x80040409L	Indicates that an interface used by a given task failed. The CAPostalCodeTask.PrepareTask function could return this error if the SERP Engine interface failed to be created.
E_MRTK_ INCORRECT_ INPUT_FIELDMAP	0x8004040aL	Indicates that some fields required for SERP certification were not mapped. For example, AddressLine1 or PostalCode are not mapped. See the Field IDs table for details as to which fields are required.
E_MRTK_ INVALID_INPUT_ BLOCK	0x8004040bL	Indicates that the input block is formatted incorrectly. This can occur if an empty address block is passed to the CABatchTask.Update function.
E_MRTK_ INVALID_CALL_ SEQUENCE	0x8004040cL	Indicates that the sequence of function calls in a particular task was not correct. For instance, PrepareTask and ValidateProperties must be called before the CASOAReportTask.PrintSOA function.

E_MRTK_ DATAFILE_NOT_ FOUND	0x8004040dL	Indicates that a required data file could not be found.
E_MRTK_ DATAFILE_ FORMAT_ INCORRECT	0x8004040eL	Indicates that the format of the data files is incorrect.
E_MRTK_ DATAFILE_ EXPIRED	0x8004040fL	Indicates that the data files have expired. Currently, SERP data files expire every two (2) months.
E_MRTK_ DATAFILE_OLD	0x80040410L	Indicates that a new or current version of BCC Architect is trying to access expired data files.
E_MRTK_ DATAFILE_NEW	0x80040411L	Indicates that an old version of BCC Architect is trying to access current data files.
E_MRTK_ EXPIRED	0x80040413L	Indicates that a component of BCC Architect has expired. For instance, CABatchTask may have expired so a call to CABatchTask.PrepareTask will return this error.
E_MRTK_USER_ CANCELED	0x80040414L	Indicates that "Cancel" has hit during a process. For instance, CASort.Send will return this error if the user hits the "Cancel" button. This assumes that graphical user elements have been enabled.
E_MRTK_NOT_ IMPLEMENTED	0x80040415L	Indicates that a specific feature, function or interface has not been implemented.
E_MRTK_ PRINTER_ERROR	0x80040416L	Indicates a general printer error such as attempting to print to a printer that does not exist.
E_MRTK_ UPDATE_ RESTART_ NEEDED	0x80040417L	Indicates that a system reboot is required to complete the BCC Architect update.
E_MRTK_ UPDATE_FAILED	0x80040418L	Indicates that the auto-updater failed to update BCC Architect
E_MRTK_SORT_ NOT_ENOUGH_ PIECES	0x80040419L	Indicates that the minimum number of pieces required for a particular sort were not provided.
E_MRTK_FAILED_ TEMP_FILE	0x8004041aL	Indicates that one of several temporary file managers failed during their construction, initialization, etc.
E_MRTK_ INVALID_REG_ KEY	0x8004041bL	Indicates that the current registration key is invalid.
E_MRTK_ REQUEST_KEY_ INVALID	0x80040440L	BCC Architect Server Specific: Indicates that the Request key is invalid.
E_MRTK_IN_ FIELD_CNT_LOW	0x80040442L	BCC Architect Server Specific: Indicates that the input field count is too low for the specified Request Type.

E_MRTK_IN_FIELD_CNT_HIGHT	0x80040442L	BCC Architect Server Specific: Indicates that the input field count is too high for the specified Request Type
E_MRTK_OUT_FIELD_CNT_LOW	0x80040443L	BCC Architect Server Specific: Indicates that the output field count is too low for the specified Request Type
E_MRTK_OUT_FIELD_CNT_HIGH	0x80040444L	BCC Architect Server Specific: Indicates that the output field count is too high for the specified Request Type
E_MRTK_REQUEST_TOO_LARGE	0x80040445L	BCC Architect Server Specific: Indicates that the request is over 1024K.
E_MRTK_REQUEST_LEN_INVALID	0x80040446L	BCC Architect Server Specific: Indicates that the size specified within the Request does not match the actual input size.
E_MRTK_REQUEST_FORMAT_INVALID	0x80040447L	BCC Architect Server Specific: Indicates that the Request format is invalid.
E_MRTK_REGISTRATION_KEY_INVALID	0x80040448L	BCC Architect Server Specific: Indicates the registration key specified within the Request is invalid.
E_MRTK_COMM_ERROR	0x80040449L	BCC Architect Server Specific: Indicates that the client was unable to connect to the remote computer and that the local networking is ok.
E_MRTK_SERVER_INVALID	0x8004044aL	BCC Architect Server Specific: Indicates that the BCC Architect Server specified is incorrect. Make sure that the format is of the type “ServerNameOrIP:PortNumber”
E_MRTK_SYSTEM_COMM_ERROR	0x8004044bL	BCC Architect Server Specific: Indicates that the client was unable to load the Winsock communication libraries or that the local network is not set up correctly.
E_MRTK_CONNECTION_REFUSED	0x8004044cL	BCC Architect Server Specific: Indicates that MRTK was able to connect to the Server but the connection was refused. A common reason for this is that the specified port is incorrect.
E_MRTK_SERVER_ERROR	0x80040450L	BCC Architect Server Specific: Indicates general error.
E_MRTK_UPDATE_FAILED_DISKFULL	0x80040480L	Indicates that auto-updater failed because disk was full.
E_MRTK_SORT_CONSTRAINT_FAILURE	0x8004041fL	Indicates that the discontinued sort “Catalog Mail” was selected

.NET Classes Reference

Contents

The .NET POSTALCODEAssembly Class for Correcting Single Addresses

The BCC Architect POSTALCODEAssembly object verifies and corrects a single address. The POSTALCODEAssembly object is designed for use with address entry forms, Web pages, and any other environment where single address verification is needed.

After checking an address, you can retrieve the corrected address information. This includes the route number, station name and all the other individual elements of an address, if the address was corrected. Returning the individual elements is a powerful tool that can be used to test if certain elements are missing. For example, with just a few lines of code, you can tell if the user neglected to enter required unit information (e.g., suite or apartment number). If an address could not be verified, an error code and associated description can be retrieved, providing details about what caused the verification process to fail.

The POSTALCODEAssembly object can act as a client of the BCC Architect Server software. This allows the matching process to be done on a server rather than the computer running the POSTALCODEAssembly object. This is very useful in keeping the requirements low on the workstation. Putting the POSTALCODEAssembly in client mode is as simple as setting the BCC Architect Server property of the object. This property specifies where the BCC Architect Server is located and whether or not to use it. If you are using the POSTALCODEAssembly in a Web environment, we highly recommended that you use the BCC Architect Server.

Overview

The following general procedure should be used to implement the POSTALCODEAssembly class:

1. Add a reference to the .NET component Satori.MRTKCAAssembly.
2. While not required, you can reduce the amount of typing needed by adding using statements (C#) or Imports statements (Visual Basic) for the Satori.Architect.CA and Satori.Architect.CA.Interfaces namespaces.
3. Create a POSTALCODEAssembly object.
4. Call PrepareTask.
5. Set the properties for the input address. The minimum amount of data required is AddressLine1, Municipality Province, and PostalCode. For best results, we recommended that all of the basic address properties be used: Business, AddressLine1, AddressLine2, Municipality Province, and PostalCode. Alternatively, the entire address can be passed in at once using the AddressBlock property.
6. Set the formatting and configuration properties as desired. For example, you can control the casing of the output address with the Casing property.
7. Call CheckAddress.
8. Retrieve the updated properties for the address. Of particular interest is the ErrorCode property, which provides information about the results of the address check.
9. Call EndTask.

POSTALCODEAssembly Functions

The POSTALCODEAssembly functions are defined below. If you have added the Satori.Architect.CA reference to your .NET project, then you can view all of the available POSTALCODEAssembly members, and their definitions, in the Object Browser and IntelliSense.

BrowseAddress

Syntax

```
bool BrowseAddress();
```

Description

Launches the Postcode Browser. The Postcode Browser is a tool that allows the user to search through all the addresses contained in the address files.

Parameters

None.

Return values

true

Indicates user clicked Update in the Postcode Browser.

false

Indicates user clicked Cancel in the Postcode Browser.

Notes

- This function will launch the Postcode Browser and try to look up the address contained within the POSTALCODEAssembly object.
- If you want to launch the Postcode Browser without automatically looking up an address then call ClearAddress before calling this function.
- If the return value is true, then the POSTALCODEAssembly object will contain the address elements for the address that the user chose to keep within the Postcode Browser.
- By checking the ErrorCode property after calling the CheckAddress function, the Postcode Browser can be automatically launched for an address that does not correct.

See also

- CheckAddress
- ClearAddress
- ErrorCode

CheckAddress

Syntax

```
void CheckAddress();
```

Description

Correct and format an address.

Parameters

None.

Return values

None.

Notes

- This function will try to match the address contained in the POSTALCODEAssembly object against the Canada Post database.
- A call to CheckAddress will update all of the POSTALCODEAssembly object property values.
- After a call to CheckAddress, you can get the results of the process by checking the value of the ErrorCode property. In some situations, you may want to call BrowseAddress if the ErrorCode property indicates that the address could not be corrected.
- The address elements returned after calling CheckAddress will be formatted according to the values of the various formatting properties.

See also

- BrowseAddress
- ErrorCode

ClearAddress

Syntax

```
void ClearAddress();
```

Description

Resets all of the POSTALCODEAssembly object's property values.

Parameters

None.

Return values

None.

Notes

- This function can be used to quickly clear the address information from a POSTALCODEAssembly object, thus bringing the object into a known state (i.e., no address, error code, etc.) before setting the object's properties.

See also

CheckAddress

EndTask

Syntax

```
void EndTask();
```

Description

Cleans up and releases a POSTALCODEAssembly object.

Parameters

None.

Return values

None.

Notes

- EndTask releases the address-matching engine.
- We recommended that you call EndTask when you are done with the POSTALCODEAssembly object.
- You are not required to call EndTask after each call to CheckAddress.

See also

PrepareTask

Syntax

```
void PrepareTask();
```

Description

Initializes and prepares the POSTALCODEAssembly object.

Parameters

None.

Return values

None.

Notes

- Call PrepareTask only once, after you create the POSTALCODEAssembly object.
- This function must be called before calling any of the other functions or setting any of the properties of the POSTALCODEAssembly object. Failing to do so will cause subsequent function calls to fail. The one exception to this rule is the MailRoomServer property, which must be set prior to calling PrepareTask.

See also

- MailRoomServer

POSTALCODEAssembly Properties

The POSTALCODEAssembly properties are defined below. If you have added the Satori.Architect.CA reference to your .NET project, then you can view all of the available POSTALCODEAssembly members, and their definitions, in the Object Browser and IntelliSense.

AddressBlock

Syntax

```
string AddressBlock;
```

Description

A read-write property containing the address as two or more lines of text.

Notes

- You can call CheckAddress with the address data input as an address block. This process will take two or more lines of text, identify which lines contain address data, check the address and then reconstruct the address block using the formatted address lines, business name and last line (if the address was matched).

Input:

- The address block must be a stream of delimited text. The end of each line in the address block should be delimited with a carriage return/line feed, line feed, carriage return or tab. The delimiter used must be consistent throughout the address block.
- The following definitions are used to describe the lines in the block of text:
- Last Line – This line contains the municipality, province and Postal Code indicating where the piece of mail should be delivered.
- Address Line – This line is always required and specifies the actual delivery address. It should be above the Last Line.
- The algorithm for this process will first locate the address line by starting at the bottom-most line and then moving up until a recognizable Postal Code or municipality/province combination is located. This line will be designated the last line of the address.

Output:

- On output, the address block will be constructed according to the values of the various formatting properties. All spaces at the beginning and end of each line, and extra spaces between words will be removed. The output address block will be created with a carriage return/line feed as the delimiter.
- Any lines of data that are not part of the corrected address will be returned in the address block. They will be placed above the address information.

See also

- CheckAddress
- Casing

AddressLine1

Syntax

```
string AddressLine1;
```

Description

A read-write property containing the first address line.

Notes

- AddressLine1 is the top address line, above the municipality, province and Postal Code.
- If there are two address lines, AddressLine2 will contain the primary address line and AddressLine1 will contain the secondary address information.
- Both AddressLine1 and AddressLine2 are formatted according to the values of the various formatting properties.

See also

- CheckAddress
- AddressLine2
- Casing

AddressLine2

Syntax

```
string AddressLine2;
```

Description

A read-write property containing the second address line.

Notes

- AddressLine2 is the bottom address line, below AddressLine1 and above the municipality, province and Postal Code.
- If there are two address lines, AddressLine2 will contain the primary address line and AddressLine1 will contain the secondary address information.

Both AddressLine1 and AddressLine2 are formatted according to the values of the various formatting properties.

See also

- CheckAddress
- AddressLine1
- Casing

Business

Syntax

```
string BusinessName;
```

Description

A read-write property containing the business name.

Notes

The business name is optional.

See also

CheckAddress function

Casing

Casing

Syntax

```
int casing;
```

Description

A read-write property that determines the casing format applied to the address elements.

Notes

- You must call CheckAddress before the selected casing option is applied to the address elements in the POSTALCODEAssembly object.
- This property should be set before calling CheckAddress.
- The following table lists the available property values:

Value	Description
0	Upper case
2	Mixed case (default value)

See also

CheckAddress

CASSDate

Syntax

```
int CASSDate;
```

Description

A read-only property containing information, such as the issue date, about the last attempt to check an address.

Notes

If you are going to batch process it is strongly recommended that you save this data because it will allow a batch process (via BATCHAssembly, for instance) to skip this record if it was previously coded with the current issue.

- CheckAddress should be called before attempting to retrieve the value of this property.
- See also

CheckAddress

CertifyFlag

Syntax

```
int CertifyFlag;
```

Description

Indicates whether to skip records that have been previously corrected based on the CASSDate field

Notes

- Use the following values to indicate whether to skip previously corrected addresses:
- 0 – Check only records not corrected with this CD.
- 1 – Check every record.

See also

CheckAddress

CASSDate

CivicNumber

Syntax

```
string CivicNumber;
```

Description

A read-write property that contains the civic number for this address.

Notes

This value is included in the street address, before the street name.

See also

CheckAddress

AddressLine1

AddressLine2

CivicNumberSuffix

Syntax

```
string CivicNumberSuffix;
```

Description

A read-write property containing the civic number suffix of an address.

Notes

This value is included in the street address, before the street name.

Civic number suffixes are usually letters or fractions.

See also

CheckAddress

AddressLine1

AddressLine2

Country

Syntax

```
string Country;
```

Description

A read-only property containing the country for a foreign address.

Notes

This property has not been implemented.

See also

CheckAddress

DataFileLocation

Syntax

```
string DataFileLocation;
```

Description

A read-write property that contains the location of the address data file.

Notes

- The Datafile.dat file is required to perform address matching and its location must be specified by the user.
- You do not need to set this property unless you have moved the data file.
- You must set this location prior to calling PrepareTask.

See also

PrepareTask

ErrorCode

Syntax

```
int ErrorCode;
```

Description

A read-only property containing a code that indicates the results of an address-matching attempt.

Notes

- The error code indicates whether an address was matched to the Canada Post database. It also provides information about the changes, if any, that were made to the input address in order to correct it, or the reason an address was unable to be corrected.
- An error code less than 100 indicates that the address was corrected, while an error code greater than 100 indicates that the address was unable to be corrected.
- CheckAddress should be called before attempting to retrieve the value of this property.
- See the Error Codes table for a complete list of error code values and their descriptions.

See also

CheckAddress

ErrorCodeString

ErrorCodeString

Syntax

```
string ErrorCodeString;
```

Description

A read-only property containing a description of the results of an address-matching operation.

Notes

- CheckAddress should be called before attempting to retrieve the value of this property.
- The error description corresponds to the value of the ErrorCode property.

See also

CheckAddress function

ErrorCode

GDType

Syntax

```
string GDType;
```

Description

A read-write property that contains the general delivery type of a address.

Notes

- There are two types of general delivery addresses:
- STN – Station
- RPO – Retail Postal Outlet

See also

StationName

StationQualifier

StationType

MailRoomServer

Syntax

```
string MailRoomServer;
```

Description

A read-write property containing the location of the BCC Architect Server.

Notes

- Setting this property creates a TCP/IP connection to the BCC Architect Server, which can reside on the local network or virtually anywhere.
- This property should be set before calling PrepareTask.
- We recommend that the BCC Architect Server be used when correcting addresses from a Web site.
- The format is: Server Name (or IP Address):Port.
- Currently, going outside of the proxy server might not be supported.
- You cannot use the Postal Browser with a BCC Architect Server connection.

See also

PrepareTask

Municipality

Syntax

```
string Municipality;
```

Description

A read-write property containing the municipality.

Notes

CheckAddress should be called before attempting to retrieve the value of this property.

See also

CheckAddress

POBoxNumber

Syntax

```
string POBoxNumber;
```

Description

A read-write property that contains the PO Box number for an address.

Notes

This information may be contained in the address line properties.

See also

CheckAddress

AddressLine1 property

AddressLine2

POBoxType

Syntax

```
string POBoxType;
```

Description

A read-write property that contains the PO Box type for an address.

Notes

This information may be contained in the address line properties.

See also

CheckAddress

AddressLine1 property

AddressLine2

PostalCode

Syntax

```
string PostalCode;
```

Description

A read-write property that contains the Postal Code.

Notes

- This property is essential to a complete address.
- After a successful call to CheckAddress, this propertywill be formatted with a space separating the Forward Sortation Area (first three characters) and the Local Delivery Unit (last three characters).

See also

CheckAddress

Province

Syntax

```
string Province;
```

Description

A read-write property containing the province.

Notes

This property is essential to a complete address.

See also

CheckAddress

RouteNumber

Syntax

```
string RouteNumber;
```

Description

A read-write property that contains the rural route number for an address.

Notes

This information may be contained in the address line properties.

See also

CheckAddress

AddressLine1 property

AddressLine2

RouteType

Syntax

```
string RouteType;
```

Description

A read-write property that contains the type of a route service.

Notes

- This information may be contained in the address line properties.
- Contains one of the following:
- RR – Rural Route
- SS – Suburban Service

- MR – Mobile Route
- GD – General Delivery

See also

CheckAddress

AddressLine1 property

AddressLine2

StationName

Syntax

```
string StationName;
```

Description

A read-write property that contains the name of the area in which the station resides.

Notes

- Also called the Delivery Installation Area Name.

See also

StationQualifier

Syntax

```
string StationQualifier;
```

Description

A read-write property that contains the station qualifier.

Notes

If an area contains multiple delivery stations, this property will uniquely identify the specific delivery station.

- Also called the Delivery Installation Qualifier Name.

See also

StationName

StationType

Syntax

```
string StationType;
```

Description

- A read-write property that contains the station type.
- Also called the Delivery Installation Type.

Notes

- Contains one of the following:
- BDP – Bureau De Poste
- CC – Concession Commerciale
- CDO – Commercial Dealership Outlet
- CMC – Community Mail Centre
- CPC – Centre Postal Communautaire
- CSP – Comptoir Service Postal
- LDVD – Letter Carrier Depot
- PDF – Poste De Facteurs
- PO – Post Office
- RPO – Retail Postal Outlet
- STN – Station
- SUCC – Succursale

See also

StreetDirection

Syntax

```
string StreetDirection;
```

Description

A read-write property containing the street direction prefix, if any.

Notes

- CheckAddress should be called before attempting to retrieve the value of this property.
- An example of a street direction would be "NW" in "416 NW Lake St."
- This information may be contained in the address line properties.

See also

CheckAddress

AddressLine1 property

AddressLine2

StreetName

Syntax

```
string StreetName;
```

Description

A read-write property containing the street name.

Notes

- CheckAddress should be called before attempting to retrieve the value of this property.
- An example of a street name would be "Lake" in "416 NW Lake St."
- This information may be contained in the address line properties.

See also

CheckAddress

AddressLine1 property

AddressLine2

StreetType

Syntax

```
string StreetType;
```

Description

A read-write property containing the street type.

Notes

- CheckAddress should be called before attempting to retrieve the value of this property.
- An example of a street type would be "St." in "416 NW Lake St."
- This information may be contained in the address line properties.

See also

CheckAddress

AddressLine1 property

AddressLine2

UnitDesignator

Syntax

```
string UnitDesignator;
```

Description

A read-write property containing the unit designator.

Notes

- CheckAddress should be called before attempting to retrieve the value of this property.
- An example of a unit designator is "Suite" in “301 Maryland Ave Suite 1.”
- This information may be contained in the address line properties.

See also

CheckAddress

UnitNumber

Syntax

```
string UnitNumber;
```

Description

A read-write property containing the unit number.

Notes

- Call CheckAddress before attempting to retrieve the value of this property.
- An example of a unit number would be "10" in "10-123 Main St." or “1” in “301 Maryland Ave Suite 1.”
- This information may be contained in the address line properties.

See also

CheckAddress

The .NET BATCHAssembly Class for Correcting Batch Addresses

The BCC Architectt BATCHAssembly object performs address verification and correction on a batch of addresses. BATCHAssembly provides a flexible interface that allows you to control the amount of information returned for each address as well as the number of records processed at a time. BATCHAssembly can be configured to skip records that were processed successfully on a previous occasion with the current issue. By doing so, you can avoid a costly rewrite to a database, thereby increasing processing speed.

Overview

The following general procedure should be used to implement the BATCHAssembly class:

1. Add a reference to the .NET component Satori.Architect.CA.
2. While not required, you can reduce the amount of typing needed by adding using statements (C#) or Imports statements (Visual Basic) for the Satori.Architect.CA and Satori.Architect.CA.Interfaces namespaces.
3. Create a BATCHAssembly object.

4. Call PrepareTask.
5. Create a CAAddressFieldList object that defines the input fields for each record. Fields are added to the CAAddressFieldList object using the CAFields.Field enumeration. The minimum amount of data required is ADDRESS_LINE_1, MUNICIPALITY, PROVINCE and POSTAL_CODE. For best results, we recommend that all of the basic address fields be used: BUSINESS, ADDRESS_LINE_1, ADDRESS_LINE_2, MUNICIPALITY, PROVINCE and POSTAL_CODE.
6. Create a CAAddressFieldList object that defines the updated output fields for each processed record. Typically, this would include the basic address fields plus any of the data fields that result from the address matching process that are of use to you (e.g., STATION_NAME or LARGE_VOLUME_RECEIVER_NAME). Also of particular interest is the ERROR_CODE field, which provides information about the results of the address check.
7. Call SetProperty to set the desired properties. The individual properties are specified using the CAProperties.Batch enumeration. You must set SETTINGS_FIELD_LIST_IN and SETTINGS_FIELD_LIST_OUT using the CAAddressFieldList objects that were created in steps 5 and 6, respectively. You should also specify the number of records to be processed at a time with SETTINGS_INPUT_BLOCK_RECORD_COUNT.
8. Call ValidateProperties.
9. Create a CAAddressRecordBlock object.
10. Loop through the records in your database, and for each record to be processed:
11. Create a CAAddressRecord object. A CAAddressFieldList object is required when creating a CAAddressRecord object. Use the CAAddressFieldList object created in step 5.
12. Set the input field values using the Fields property of the CAAddressRecord object.
13. Add the CAAddressRecord object to the CAAddressRecordBlock object.
14. Repeat steps a through c until the CAAddressRecordBlock object contains SETTINGS_INPUT_BLOCK_RECORD_COUNT CAAddressRecord objects.
15. Call Update, passing in the CAAddressRecordBlock object. A new, updated CAAddressRecordBlock object will be returned.
16. Iterate through the CAAddressRecordBlock object to get each updated CAAddressRecord object. In turn, iterate through each CAAddressRecord object to get the updated fields for each record. The fields for each output record will match those specified by the SETTINGS_FIELD_LIST_OUT property set earlier.
17. Repeat steps 10 through 12 until all records have been processed. Call the Clear function of the CAAddressRecordBlock object to reset it before adding a new block of records to it.
18. Call PreviewBatchReports or PrintBatchReports to preview or print the Processing Summary Report. You can also save the report as a PDF file using the SaveReportsAsPDF function.
19. Call EndTask.

BATCHAssembly Functions

The BATCHAssembly functions are defined below. If you have added the Satori.Architect.CA reference to your .NET project, then you can view all of the available BATCHAssembly members, and their definitions, in the Object Browser and IntelliSense.

AbortTask

Syntax

```
void AbortTask();
```

Description

Aborts processing.

Parameters

None.

Return values

None.

Notes

Call this function to end processing prematurely.

See also

EndTask

Syntax

```
void EndTask();
```

Description

Cleans up and releases a BATCHAssembly object.

Parameters

None.

Return values

None.

Notes

EndTask cleans up and releases used resources and should be called when done with the BATCHAssembly object.

- You must call EndTask after processing the last batch of addresses.

See also

GetProperty

Syntax

```
string GetProperty(CAProperties.Batch batchPropertyId);
```

Description

Retrieves the current value of a BATCHAssembly property.

Parameters

batchPropertyId

Specifies the enum name of the property to get.

Return values

The value of the property specified by batchPropertyId.

Notes

All property values are returned as a string, regardless of the data type passed into SetProperty.

See also

See the BATCHAssembly Properties section for a complete list of properties.

GetPropertySummary

Syntax

```
string GetPropertySummary(CAProperties.Batch batchPropertyId);
```

Description

Returns a string containing a description of a property.

Parameters

batchPropertyId

Specifies the enum name of the property for which to get information.

Return values

A description of the property specified by batchPropertyId.

Notes

This function can be used to display the description of a property on screen to the user.

See also

See the BATCHAssembly Properties section for a complete list of properties.

PrepareTask

Syntax

```
void PrepareTask();
```

Description

Initializes and prepares the BATCHAssembly object.

Parameters

None.

Return values

None.

Notes

- PrepareTask should be called only once, after the BATCHAssembly object is created.
- This function must be called before calling any of the other functions or setting any of the properties of BATCHAssembly. Failing to do so will cause subsequent function calls to fail. The only exceptions to this rule are setting the SETTINGS_DATAFILE_LOCATION and SETTINGS_MAILROOM_SERVER_LIST properties, which must be defined prior to calling PrepareTask.

See also

PreviewBatchReports

Syntax

```
void PreviewBatchReports();
```

Description

Previews all correction reports.

Parameters

None.

Return values

None.

Notes

The Statement of Accuracy Report is the only report currently available.

See also

PreviewReport

PreviewReport

Syntax

```
void PreviewReport(CAReports.Batch mrtkReportId);
```

Description

Previews a correction report.

Parameters

mrtkReportId

The ID of the report to preview.

Return values

None.

Notes

- Unlike PreviewBatchReports, which previews all of the correction reports, this function only displays the specified report.
- The Statement of Accuracy Report is the only correction report currently available.

See also

PreviewBatchReports

See the BATCHAssembly Reports section for a complete list of reports.

PrintBatchReports

Syntax

```
void PrintBatchReports(string printerName, bool showPrintSetupDialog);
```

Description

Prints all correction reports.

Parameters

printerName

The name of the printer you wish to print to.

showPrintSetupDialog

Set to true to show the printer setup dialog box or false to hide it.

Return values

None.

Notes

- If printerName is set to an empty string then the default printer will be used to print the form.
- If showPrintSetupDialog is set to true then the value for printerName will be ignored.
- If the printer is a network printer, the printerName parameter must be set to the full name (e.g., \\Server\HP Laser Jet 5). To find the full printer name, you could print a test page from the print driver.
- The Statement of Accuracy Report is the only correction report currently available.

See also

PrintReport

PrintReport

Syntax

```
void PrintReport(CAReports.Batch mrtkReportId, string printerName, bool
showPrintSetupDialog);
```

Description

Prints a correction report.

Parameters

mrtkReportId

The ID of the report to print.

printerName

The name of the printer you wish to print to.

showPrintSetupDialog

Set to true to show the printer setup dialog box or false to hide it.

Return values

None.

Notes

- If printerName is set to an empty string then the default printer will be used to print the form.

If showPrintSetupDialog is set to true then the value for printerName will be ignored.

- If the printer is a network printer, the printerName parameter must be set to the full name (e.g., \\Server\HP Laser Jet 5). To find the full printer name, you could print a test page from the print driver.
- Unlike PrintBatchReports, which prints all of the correction reports, this function only prints the specified report.
- The Statement of Accuracy Report is the only correction report currently available.

812B811See also

PrintBatchReports

See the BATCHAssembly Reports section for a complete list of reports.

RetrieveReviewed

Syntax

```
CAAddressRecordBlock RetrieveReviewed();
```

Description

Retrieves the addresses kept by the user in the Review Errors window.

Parameters

None.

Return values

A CAAddressRecordBlock object containing the updated addresses.

Notes

- Only those addresses the user chooses to keep in the Review Errors window will be returned by this function.

See also

SaveReportAsPDF

Syntax

```
string SaveReportAsPDF(CAReports.Batch mrtkReportId, string fileName);
```

Description

Saves a report as a PDF file.

Parameters

mrtkReportId

The report to save, as specified by the CAReports.Batch enum.

filename

The name, including path, of the file to save.

Return values

Returns a string containing the path and name of the saved file.

Notes

- Unlike SaveReportsAsPDF, which saves all of the correction reports, this function only saves the specified report.
- The Statement of Accuracy Report is the only correction report currently available.

See also

SaveReportsAsPDF

See the BATCHAssembly Reports section for a complete list of reports.

SaveReportsAsPDF

Syntax

```
string SaveReportsAsPDF(string fileName);
```

Description

Saves all correction reports as a PDF file.

Parameters

filename

The name, including path, of the file to save.

Return values

Returns a string containing the path and name of the saved file.

Notes

The Statement of Accuracy Report is the only correction report currently available.

See also

SaveReportAsPDF

SetProperty

Syntax

```
void SetProperty(CAProperties.Batch batchPropertyId,
object val)
```

Description

Sets the value of a BATCHAssembly property.

Parameters

batchPropertyId

Specifies the enum name of the property to set.

val

Specifies the new property value.

Return values

None.

Notes

The parameter val accepts multiple data types. See the BATCHAssembly Properties section for the data type of each property.

See also

See the BATCHAssembly Properties section for a complete list of properties.

ShowBatchWizard

Syntax

```
void ShowBatchWizard();
```

Description

Displays the Batch Wizard.

Parameters

None.

Return values

None.

Notes

Call this function if you want to display the Batch Wizard. The Batch Wizard provides a graphical interface that leads a user through the various processing options.

See also

Update

Syntax

```
CAAddressRecordBlock Update(CAAddressRecordBlock addressBlock);
```

Description

Corrects the addresses contained in the address block.

Parameters

addressBlock

A CAAddressRecordBlock object containing the input addresses to be processed.

Return values

A CAAddressRecordBlock object containing the corrected addresses.

Notes

- You may want to experiment with the SETTINGS_INPUT_BLOCK_RECORD_COUNT property. In preliminary tests, we have found the optimal setting to be around 25-50. The number of fields that you want returned (see SETTINGS_FIELD_LIST_OUT property) greatly affects this number.
- For optimal performance, only ask for the output fields that you need. The reason for this is that the extra information requires additional lookups that slow processing.

See also

See the BATCHAssembly Properties section for the definition of:

SETTINGS_FIELD_LIST_IN

SETTINGS_FIELD_LIST_OUT

SETTINGS_INPUT_BLOCK_RECORD_COUNT

ValidateProperties

Syntax

```
void ValidateProperties();
```

Description

Verifies that the BATCHAssembly object is set up correctly and ready to run.

Parameters

None.

Return values

None.

Notes

- This function verifies that the basic requirements of a BATCHAssembly object have been met:
- The address matching engine is loaded and able to run.
- The input field list consists of the minimum set of fields, namely, ADDRESS_LINE_1, MUNICIPALITY, PROVINCE and POSTAL_CODE.

This function needs to be called before you call Update.

See also

SetProperty function

See the BATCHAssembly Properties section for a complete list of properties.

BATCHAssembly Properties

The BATCHAssembly properties are members of the CAProperties.Batch enumeration and are defined below. These enum names are used as arguments of the GetProperty and SetProperty functions. If you have added the Sator-i.Architect.CA reference to your .NET project, then you can view all of the available CAProperties.Batch enums in the Object Browser and IntelliSense.

BATCH_CERTIFY_FLAG

Data type

int

Description

Determines if previously corrected records should be skipped in order to increase processing speed.

Notes

- Setting this property to 0 speeds up processing of a batch of addresses by skipping those addresses that have already been corrected with the current issue.
- The CASSDATE field must be supplied as input in order for an address to be skipped.
- The following table lists the available property values:

Valu-e	Description
0	Check only those records not corrected with this issue (default value).

1	Check every record.
4	Rebuild the Statement of Accuracy report based on the CASSDATE field with only records that have been corrected with this CD.

BATCH_LIST_NAME

Data type

string

Description

Specifies the name of the list to be printed on the Statement of Accuracy report.

Notes

The default value for this property is an empty string.

BATCH_MAILERS_ADDRESS

Data type

string

Description

Specifies the address of the mailer to be printed on the Statement of Accuracy report.

Notes

The default value for this property is an empty string.

BATCH_MAILERS_COMPANY

Data type

string

Description

Specifies the company name for the mailer that will be printed on the Statement of Accuracy report.

Notes

The default value for this property is an empty string.

BATCH_MAILERS_CPC_NUMBER

Data type

string

Description

Specifies the CPC number of the mailer's Canada Post account to be printed on the Statement of Accuracy report.

Notes

The default value for this property is an empty string.

BATCH_MAILERS_MUNICIPALITY

Data type

string

Description

Specifies the municipality of the mailer to be printed on the Statement of Accuracy report.

Notes

The default value for this property is an empty string.

BATCH_MAILERS_NAME

Data type

string

Description

Specifies the name of the mailer to be printed on the Statement of Accuracy report.

Notes

The default value for this property is an empty string.

BATCH_MAILERS_POSTAL_CODE

Data type

string

Description

Specifies the Postal Code of the mailer to be printed on the Statement of Accuracy report.

Notes

The default value for this property is an empty string.

BATCH_MAILERS_PROVINCE

Data type

string

Description

Specifies the province of the mailer to be printed on the Statement of Accuracy report.

Notes

The default value for this property is an empty string.

FORMAT_CASING

Data type

int

Description

Determines the casing format applied to the address elements.

Notes

The following table lists the available property values:

Value	Description
0	Mixed case (default value)
2	Upper case

SETTINGS_DATAFILE_LOCATION

Data type

string

Description

Specifies the location of the address correction files.

Notes

- Setting this property will update the path to the address correction files in the mrtk.ini file.
- The default value for this property is an empty string.

SETTINGS_FIELD_LIST_IN

Data type

CAAddressFieldList

Description

A CAAddressFieldList object that defines which fields are supplied as input to the address-matching engine.

Notes

- This property defines the input fields that are contained in each CAAddressRecord to be processed.
- The input field list should contain, at a minimum, ADDRESS_LINE_1, MUNICIPALITY, PROVINCE and POSTAL_CODE.

SETTINGS_FIELD_LIST_OUT

Data type

CAAddressFieldList

Description

A CAAddressFieldList object that defines which fields are returned as output from the address-matching engine.

Notes

This property defines the output fields that are contained in each CAAddressRecord that has been processed.

SETTINGS_HIDE_PROGRESS_AFTER_BATCH

Data type

bool

Description

Determines if the progress dialog box remains visible after processing is complete.

Notes

- Set to true to hide the progress dialog box after processing is complete.
- The Statement of Accuracy report can be previewed or printed from the progress dialog box.
- The default value for this property is false.

SETTINGS_INI_FILE_NAME

Data type

string

Description

Determines the location and name of the BCC Architect CA .INI file.

Notes

- We recommend that you do not change this value.
- The default value for this property is “mrtkca.ini”.

SETTINGS_INPUT_BLOCK_RECORD_COUNT

Data type

int

Description

Specifies the number of records contained in an address block.

Notes

- This property specifies the number of CAAddressRecord objects contained in a CAAddressRecordBlock object.
- The size of the record block determines the number of records processed with each call to Update.

SETTINGS_MAILROOM_SERVER_LIST

Data type

string

Description

Specifies the location of the BCC Architect Server.

Notes

- Setting this property creates a TCP/IP connection to the BCC Architect Server, which can reside on the local network or virtually anywhere.
- This property should be set before calling PrepareTask.
- We recommend that you use the BCC Architect Server when processing addresses from a Web site.
- The format is: Server Name (or IP Address):Port.
- Currently, going outside of the proxy server might not be supported.
- The default value for this property is an empty string.

SETTINGS_RECORD_COUNT

Data type

int

Description

Specifies the total number of records to be processed.

Notes

The default value for this property is 0.

SETTINGS_SHOW_PROGRESS

Data type

bool

Description

Determines if the progress dialog box is displayed during processing.

Notes

- Set to true to display the progress dialog box during processing.
- The Statement of Accuracy report can be previewed or printed from the progress dialog box after processing is complete.
- The default value for this property is false.

BATCHAssembly Reports

The BATCHAssembly reports are members of the CAReports.Batch enumeration and are defined below. These enum names are used as arguments of the various functions to preview, print or save a report. If you have added the Satori.Architect.CA reference to your .NET project, then you can view all of the available CAReports.Batch enums in the Object Browser and IntelliSense.

Name	Description
STATEMENT_OF_ACCURACY	Statement of Accuracy Report; this report is required for postal discounts.

BATCHAssembly Fields

The BATCHAssembly fields are members of the CAFields.Batch enumeration and are defined below. These enum names are used as arguments of the various functions of the CAAddressFieldList and CAAddressFields objects. If you have added the Satori.Architect.CA reference to your .NET project, then you can view all of the available CAFields.Batch enums in the Object Browser and IntelliSense.

Name	Description
ADDRESS_LANGUAGE	
ADDRESS_LINE_1	
ADDRESS_LINE_2	
BUSINESS	
CASSDATE	
CIVIC_NUMBER	
CIVIC_NUMBER_SUFFIX	
COUNTRY	
ERROR_CODE	
ERROR_STRING	
FIRST_NAME	
GD_TYPE	
LARGE_VOLUME_RECEIVER_NAME	
LARGE_VOLUME_RECEIVER_BRANCH_NAME	

LAST_NAME	
MUNICIPALITY	
NONSTANDARD_EXTRA_INFORMATION	
PO_BOX_NUMBER	
PO_BOX_TYPE	
POSTAL_CODE	
PROVINCE	
RECORD_ID	
ROUTE_NUMBER	
ROUTE_TYPE	
SKIPPED_CERTIFY	
STANDARD_EXTRA_INFORMATION	
STATION_NAME	
STATION_QUALIFIER	
STATION_TYPE	
STREET_DIRECTION	
STREET_NAME	
STREET_TYPE	
UNIT_DESIGNATOR	
UNIT_NUMBER	
USER_DEFINED_1	
USER_DEFINED_2	
USER_DEFINED_3	
USER_DEFINED_4	
USER_DEFINED_5	
USER_DEFINED_6	
USER_DEFINED_7	
USER_DEFINED_8	
USER_DEFINED_9	

USER_DEFINED_10	
USER_DEFINED_11	
USER_DEFINED_12	
USER_DEFINED_13	
USER_DEFINED_14	
USER_DEFINED_15	

The .NET SORTAssembly Class for Presorting Mailings

The BCC Architect SORTAssembly object performs postal presorting on a list of addresses. SORTAssembly offers a flexible interface, allowing you to specify the amount of information returned for each address, whether mailing lists are returned in presorted order, the number of records sent and returned at a time, how user interface windows are displayed, etc. In addition, SORTAssembly can be configured to incorporate address correction during the sorting process.

Overview

The following general procedure should be used to implement the SORTAssembly class:

1. Add a reference to the .NET component Satori.Architect.CA.
2. While not required, you can reduce the amount of typing needed by adding using statements (C#) or Imports statements (Visual Basic) for the Satori.Architect.CA and Satori.Architect.CA Interfaces namespaces.
3. Create a SORTAssembly object.
4. Call PrepareTask.
5. Create a CAAddressFieldList object that defines the input fields for each record. Fields are added to the CAAddressFieldList object using the CAFields.Field enumeration. The minimum amount of address data required is POSTAL_CODE.
6. Create a CAAddressFieldList object that defines the updated output fields for each processed record. Of particular interest is SORT_ID, which indicates a record's position in the sorted list.
7. Call ShowSortWizard. The Sort Wizard allows you to define the settings for a mailing, which can then be saved as a template for future use. By doing this, you can run sorts without showing the wizard. This step is optional.
8. Call SetProperty to set the desired properties not set with the Sort Wizard. The individual properties are specified using the CAProperties.Sort enumeration. You must set SETTINGS_FIELD_LIST_IN and SETTINGS_FIELD_LIST_OUT using the CAAddressFieldList objects that were created in steps 5 and 6, respectively. You should also specify the number of records to be processed at a time with SETTINGS_INPUT_BLOCK_RECORD_COUNT.
9. Call ValidateProperties.
10. Create a CAAddressRecordBlock object.
11. Loop through the records in your database, and for each record to be processed:

12. Create a CAAddressRecord object. A CAAddressFieldList object is required when creating a CAAddressRecord object. Use the CAAddressFieldList object created in step 5.
13. Set the input field values using the Fields property of the CAAddressRecord object.
14. Add the CAAddressRecord object to the CAAddressRecordBlock object.
15. Repeat steps a through c until the CAAddressRecordBlock object contains SETTINGS_INPUT_BLOCK_RECORD_COUNT CAAddressRecord objects.
16. Call Send, passing in the CAAddressRecordBlock object.
17. Repeat steps 11 and 12 until all records have been sent. Call the Clear function of the CAAddressRecordBlock object to reset it before adding a new block of records to it.
18. Call DoSort.
19. Call Retrieve. A new, updated CAAddressRecordBlock object containing SETTINGS_RECORD_COUNT_PER_RECEIVE records will be returned.
20. Iterate through the CAAddressRecordBlock object to get each updated CAAddressRecord object. In turn, iterate through each CAAddressRecord object to get the updated fields for each record. The fields for each output record will match those specified by the SETTINGS_FIELD_LIST_OUT property set earlier.
21. Repeat steps 15 and 16 until all records have been retrieved.
22. Call PreviewSortReports or PrintSortReports to preview or print the presort reports. You can also save the reports as a PDF file using the SaveReportsAsPDF function.
23. Call EndTask.

SORTAssembly Functions

The SORTAssembly functions are defined below. If you have added the Satori.Architect.CA reference to your .NET project, then you can view all of the available SORTAssembly members, and their definitions, in the Object Browser and IntelliSense.

DoSort

Syntax

```
void DoSort();
```

Description

Performs the selected sort on all records sent to the sort engine.

Parameters

None.

Return values

None.

Notes

- You should call this function after your final call to Send.
- Addresses will be certified prior to sorting if the SETTINGS_BATCH_PROCESS_FIRST property is set to true with the SetProperty function.

See also

Send

EndTask

Syntax

```
void EndTask();
```

Description

Cleans up and releases a SORTAssembly object.

Parameters

None.

Return values

None.

Notes

EndTask cleans up and releases used resources and should be called when done with the SORTAssembly object.

See also

GetProperty

Syntax

```
string GetProperty(CAProperties.Batch batchPropertyId); string GetProperty(CAProp-
erties.Sort sortPropertyId);
```

Description

Retrieves the current value of a SORTAssembly property.

Parameters

batchPropertyId

Specifies the enum name of the Batch property to get.

sortPropertyId

Specifies the enum name of the Sort property to get.

Return values

The value of the property specified by batchPropertyId or sortPropertyId.

Notes

- All property values are returned as a string, regardless of the data type passed into SetProperty.
- This function is overloaded to accept both Batch and Sort properties.

See also

See the BATCHAssembly Properties section for a complete list of properties.

See the SORTAssembly Properties section for a complete list of properties.

GetPropertySummary

Syntax

```
string GetPropertySummary(CAProperties.Sort sortPropertyId); string GetPropertySummary
(CAProperties.Batch sortPropertyId);
```

Description

Returns a string containing a description of a property.

Parameters

sortPropertyId

Specifies the enum name of the property to get information for.

Return values

A description of the property specified by sortPropertyId.

Notes

This function can be used to display the description of a property on screen to the user.

See also

See the BATCHAssembly Properties section for a complete list of properties.

See the SORTAssembly Properties section for a complete list of properties.

PrepareTask

Syntax

```
void PrepareTask();
```

Description

Initializes and prepares the SORTAssembly object.

Parameters

None.

Return values

None.

Notes

- PrepareTask should be called only once, after the SORTAssembly object is created.

This function must be called before calling any of the other functions or setting any of the properties of SORTAssembly. Failing to do so will cause subsequent function calls to fail. The one exception to this rule is setting the SETTINGS_MAILROOM_SERVER_LIST property, which must be defined prior to calling PrepareTask.

See also

PreviewSortReports

Syntax

```
void PreviewSortReports();
```

Description

Previews all selected sort reports.

Parameters

None.

Return values

None.

Notes

- This function previews the reports selected within the Sort Wizard or whose properties have been set to true with SetProperty.
- Setting the REPORT_PRINT_ALL_REPORTS property equal to true will set the properties of all reports to true and hence a call to this function will preview all reports.
- The previewed reports will display together in a single window.

See also

PreviewReport

See the SORTAssembly Properties section for the definition of:

REPORT_PRINT_ALL_REPORTS

REPORT_PRINT_CONTAINER_LABELS

REPORT_PRINT_FACING_SLIPS

REPORT_PRINT_KEEPER_TAGS

REPORT_PRINT_PACKING_REPORT

REPORT_PRINT_SEPARATOR_CARDS

REPORT_PRINT_SUMMARY_REPORT

REPORT_PRINT_CONTAINER_LABELS

REPORT_PRINT_TIE_ON_TAGS

PreviewReport

Syntax

```
void PreviewReport(CAReports.Sort mrtkReportId);
```

Description

Previews a Sort report.

Parameters

mrtkReportId

The ID of the report to preview.

Return values

None.

Notes

- Unlike PreviewSortReports, which previews all of the selected sort reports, this function only displays the specified report.
- Each call to this function will create a new preview window for the specified report. Hence, multiple calls to PreviewReport will result in multiple preview windows being displayed.
- Calling this function will override the selections made within the Sort Wizard for the specified report.

To preview the reports specified in the Sort Wizard, call PreviewSortReports.

See also

PreviewSortReports

See the SORTAssembly Reports section for a complete list of reports.

PrintSortReports

Syntax

```
void PrintSortReports();
```

Description

Prints all selected sort reports.

Parameters

None.

Return values

None.

Notes

- This function prints the reports selected within the Sort Wizard or whose properties have been set to true with SetProperty.
- Setting the REPORT_PRINT_ALL_REPORTS property equal to true will set the properties of all reports to true and hence a call to this function will print all reports.
- If the reports have been set to be previewed or saved as a PDF file in the Sort Wizard, then they will be previewed or saved as a PDF instead of printed when calling this function.

See also

PrintReport

See the SORTAssembly Properties section for the definition of:

REPORT_PRINT_ALL_REPORTS

REPORT_PRINT_CONTAINER_LABELS

REPORT_PRINT_FACING_SLIPS

REPORT_PRINT_KEEPER_TAGS

REPORT_PRINT_PACKING_REPORT

REPORT_PRINT_SEPARATOR_CARDS

REPORT_PRINT_SUMMARY_REPORT

REPORT_PRINT_CONTAINER_LABELS

REPORT_PRINT_TIE_ON_TAGS

PrintReport

Syntax

```
void PrintReport(CAReports.Sort mrtkReportId, string printerName, bool
showPrintSetupDialog);
```

Description

Prints a Sort report.

Parameters

mrtkReportId

The ID of the report to print.

printerName

The name of the printer you wish to print to.

showPrintSetupDialog

Set to true to show the printer setup dialog box or false to hide it.

Return values

None.

Notes

- If printerName is set to an empty string then the default printer will be used to print the form.

If showPrintSetupDialog is set to true then the value for printerName will be ignored.

- If the printer is a network printer, the printerName parameter must be set to the full name (e.g., \\Server\HP Laser Jet 5). To find the full printer name, you could print a test page from the print driver.
- Unlike PrintSortReports, which prints all of the selected sort reports, this function only prints the specified report.
- Calling this function will override the selections made within the Sort Wizard for the specified report.
- To print the reports specified in the Sort Wizard, call PrintSortReports.

See also

PrintSortReports

See the SORTAssembly Reports section for a complete list of reports.

Retrieve

Syntax

```
CAAddressRecordBlock Retrieve();
```

Description

Retrieves the processed records from the sort engine.

Parameters

None.

Return values

A CAAddressRecordBlock object containing the updated addresses.

Notes

- The number of records in the returned record block is determined by the SETTINGS_RECORD_COUNT_PER_RECEIVE property.
- The fields in each CAAddressRecord object in the address block are determined by the SETTINGS_FIELD_LIST_OUT property.

See also

See the SORTAssembly Properties section for the definition of:

SETTINGS_FIELD_LIST_OUT

SETTINGS_ RECORD_COUNT_PER_RECEIVE

SaveReportsAsPDF

Syntax

```
string SaveReportsAsPDF(string fileName);
```

Description

Saves all selected Sort reports.

Parameters

filename

The name, including path, of the file to save.

Return values

Returns a string containing the path and name of the saved file.

Notes

- This function saves the reports selected within the Sort Wizard or whose properties have been set to true with SetProperty.
- Setting REPORT_PRINT_ALL_REPORTS equal to true will set the properties of all reports to true and hence a call to this function will save all reports.
- The reports will all be saved as a single file.

See also

SaveReportAsPDF

See the SORTAssembly Properties section for the definition of:

REPORT_PRINT_ALL_REPORTS

REPORT_PRINT_CONTAINER_LABELS

REPORT_PRINT_FACING_SLIPS

REPORT_PRINT_KEEPER_TAGS

REPORT_PRINT_PACKING_REPORT

REPORT_PRINT_SEPARATOR_CARDS

REPORT_PRINT_SUMMARY_REPORT

REPORT_PRINT_CONTAINER_LABELS

REPORT_PRINT_TIE_ON_TAGS

SaveReportAsPDF

Syntax

```
string SaveReportAsPDF(CAReports.Sort mrtkReportId,
string fileName);
```

Description

Saves a report as a PDF file.

Parameters

mrtkReportId

The ID of the report to save.

filename

The name, including path, of the file to save.

Return values

Returns a string containing the path and name of the saved file.

Notes

Unlike SaveReportsAsPDF , which saves all of the selected sort reports, this function only saves the specified report.

See also

SaveReportsAsPDF

See the SORTAssembly Reports section for a complete list of reports.

Send

Syntax

```
void Send(CAAddressRecordBlock addressBlock);
```

Description

Sends an address block to the sort engine.

Parameters

addressBlock

A CAAddressRecordBlock object containing the input addresses to be processed.

Return values

None.

Notes

- You may want to experiment with the number of records in an input block as specified by the SETTINGS_INPUT_BLOCK_RECORD_COUNT property. In preliminary tests, we have found the optimal setting to be around 25-50.
- The fields in each CAAddressRecord object in the address block are determined by the SETTINGS_FIELD_LIST_IN property.
- You should call DoSort after you have finished sending all of your records to the sort engine.

See also

DoSort

See the SORTAssembly Properties section for the definition of:

SETTINGS_FIELD_LIST_IN

SETTINGS_INPUT_BLOCK_RECORD_COUNT

SetProperty

Syntax

```
void SetProperty(CAProperties.Batch batchPropertyId,
object val) void SetProperty(CAProperties.Sort sortPropertyId,
object val)
```

Description

Sets the value of a SORTAssembly property.

Parameters

batchPropertyId

Specifies the enum name of the Batch property to set.

sortPropertyId

Specifies the enum name of the Sort property to set.

val

Specifies the new property value.

Return values

None.

Notes

This function is overloaded to accept both Batch and Sort properties.

See also

GetProperty

See the BATCHAssembly Properties section for a complete list of Batch properties.

See the SORTAssembly Properties section for a complete list of Sort properties.

ShowSortWizard

Syntax

```
void ShowSortWizard();
```

Description

Displays the Sort Wizard.

Parameters

None.

Return values

None.

Notes

- Call this function if you want to display the Sort Wizard. The Sort Wizard provides a graphical interface that leads a user through the steps necessary to do a mailing.
- This function should be called before calling Send.
- The wizard needs to be displayed only the first time a sort is performed or set up. Using the wizard, the user can specify their mail sort settings and save those settings as a template. This template can then be accessed with the property SETTINGS_TEMPLATE_NAME_TO_USE.
- You can only display this wizard when processing locally. Client/server implementations cannot display this wizard.
- The SETTINGS_SHOW_STATEMENT_OF_ACCURACY_CHECKBOX_IN_ WIZARD property must be set to true if you want to print or preview the correction report.
- The screen that allows the user to select the reports to print will only be visible if either SHOW_SORT_ PROGRESS is set to false or SETTINGS_HIDE_PROGRESS_AFTER_SORT is set to true before calling this function.
- Properties set through the wizard do not have to be set through SetProperty.

See also

SetProperty

See the SORTAssembly Properties section for the definition of:

SETTINGS_TEMPLATE_NAME_TO_USE

SETTINGS_SHOW_STATEMENT_OF_ACCURACY_CHECKBOX_IN_ WIZARD

SETTINGS_SHOW_SORT_PROGRESS

SETTINGS_HIDE_PROGRESS_AFTER_SORT

ValidateProperties

Syntax

```
void ValidateProperties();
```

Description

Verifies that the SORTAssembly object is set up correctly and ready to run.

Parameters

None.

Return values

None.

Notes

- This function verifies that the basic requirements of a SORTAssembly object have been met.
- If you have specified that the mailing list should be address corrected before sorting, then the requirements for

the BATCHAssembly will also have to be met.

- This function needs to be called after setting property values and before calling Send.

See also

SetProperty function

SORTAssembly Properties

The SORTAssembly properties are members of the CAProperties.Presort enumeration and are defined below. These enum names are used as arguments of the GetProperty and SetProperty functions. These two functions are overloaded to also accept properties from the CAProperties.Batch enumeration. If you have added the Satori.Architect.CA reference to your .NET project, then you can view all of the available CAProperties.Presort and CAProperties.Batch enums in the Object Browser and IntelliSense.

CONTAINER_LABEL_BOTTOM_MARGIN

Data type

int

Description

Specifies how much white space each label will have below the address.

Notes

The default value for this property is 0.

CONTAINER_LABEL_COLUMNS

Data type

int

Description

Specifies the number of columns for labels.

Notes

The default value for this property is 0.

CONTAINER_LABEL_CONTINUOUS

Data type

bool

Description

Specifies whether labels are to be printed on continuous paper.

Notes

- Set to true if labels are to be printed on a dot-matrix printer.
- The default value for this property is false.

CONTAINER_LABEL_HORIZONTAL_PAGE_OFFSET

Data type

int

Description

Specifies the horizontal offset for labels in millimeters.

Notes

The default value for this property is 0.

CONTAINER_LABEL_RIGHT_MARGIN

Data type

int

Description

Specifies how much white space each label will have to the right of the address.

Notes

The default value for this property is 0.

CONTAINER_LABEL_ROWS

Data type

int

Description

Specifies the number of rows per page of labels.

Notes

The default value for this property is 0.

CONTAINER_LABEL_VERTICAL_PAGE_OFFSET

Data type

int

Description

Specifies the vertical offset for labels in millimeters.

Notes

The default value for this property is 0.

FACING_SLIP_BOTTOM_MARGIN

Data type

int

Description

Specifies how much white space each facing slip will have below the address.

Notes

The default value for this property is 0.

FACING_SLIP_COLUMNS

Data type

int

Description

Specifies the number of columns for facing slips.

Notes

The default value for this property is 0.

FACING_SLIP_CONTINUOUS

Data type

bool

Description

Specifies whether facing slips are to be printed on continuous paper.

Notes

- Set to true if facing slips are to be printed on a dot-matrix printer.
- The default value for this property is false.

FACING_SLIP_HORIZONTAL_PAGE_OFFSET

Data type

int

Description

Specifies the horizontal offset for facing slips in millimeters.

Notes

The default value for this property is 0.

FACING_SLIP_RIGHT_MARGIN

Data type

int

Description

Specifies how much white space each facing slip will have to the right of the address.

Notes

The default value for this property is 0.

FACING_SLIP_ROWS

Data type

int

Description

Specifies the number of rows per page of facing slips.

Notes

The default value for this property is 0.

FACING_SLIP_VERTICAL_PAGE_OFFSET

Data type

int

Description

Specifies the vertical offset for facing slips in millimeters.

Notes

The default value for this property is 0.

KEEPER_TAG_BOTTOM_MARGIN

Data type

int

Description

Specifies how much white space each keeper tag will have below the address.

Notes

The default value for this property is 0.

KEEPER_TAG_COLUMNS

Data type

int

Description

Specifies the number of columns for keeper tags.

Notes

The default value for this property is 0.

KEEPER_TAG_CONTINUOUS

Data type

bool

Description

Specifies whether keeper tags are to be printed on continuous paper.

Notes

- Set to true if keeper tags are to be printed on a dot-matrix printer.
- The default value for this property is false.

KEEPER_TAG_HORIZONTAL_PAGE_OFFSET

Data type

int

Description

Specifies the horizontal offset for keeper tags in millimeters.

Notes

The default value for this property is 0.

KEEPER_TAG_RIGHT_MARGIN

Data type

int

Description

Specifies how much white space each keeper tag will have to the right of the address.

Notes

The default value for this property is 0.

KEEPER_TAG_ROWS

Data type

int

Description

Specifies the number of rows per page of keeper tags.

Notes

The default value for this property is 0.

KEEPER_TAG_VERTICAL_PAGE_OFFSET

Data type

int

Description

Specifies the vertical offset for keeper tags in millimeters.

Notes

The default value for this property is 0.

REPORT_FOLDER_NAME_ELECTRONIC_PLAN

Data type

string

Description

Specifies the directory to use when saving the electronic mailing plan files.

Notes

The default value for this property is an empty string.

REPORT_PRINT_ACCURACY_REPORT

Data type

bool

Description

Determines whether the Statement of Accuracy report is printed.

Notes

- The PrintSortReports, PreviewSortReports and SaveReportsAsPDF functions will produce all reports whose properties are set to true.
- The default value for this property is false.

REPORT_PRINT_ALL_REPORTS

Data type

bool

Description

Specifies whether all of the sort reports should be printed.

Notes

- Setting this property to true will set the value of all of the individual report properties (e.g., REPORT_PRINT_MAILING_SUMMARY, REPORT_PRINT_TIE_ON_TAGS) to true.
- The PrintSortReports, PreviewSortReports and SaveReportsAsPDF functions will produce all reports whose properties are set to true.
- The default value for this property is false.

REPORT_PRINT_CONTAINER_LABELS

Data type

bool

Description

Determines whether the container labels are printed.

Notes

- The PrintSortReports, PreviewSortReports and SaveReportsAsPDF functions will produce all reports whose properties are set to true.
- The default value for this property is false.

REPORT_PRINT_FACING_SLIPS

Data type

bool

Description

Determines whether the facing slips are printed.

Notes

- The PrintSortReports, PreviewSortReports and SaveReportsAsPDF functions will produce all reports whose properties are set to true.
- The default value for this property is false.

REPORT_PRINT_KEEPER_TAGS

Data type

bool

Description

Determines whether the keeper tags are printed.

Notes

- The PrintSortReports, PreviewSortReports and SaveReportsAsPDF functions will produce all reports whose properties are set to true.
- The default value for this property is false.

REPORT_PRINT_PACKING_REPORT

Data type

bool

Description

Determines whether the packing report is printed.

Notes

- The PrintSortReports, PreviewSortReports and SaveReportsAsPDF functions will produce all reports whose properties are set to true.
- The default value for this property is false.

REPORT_PRINT_SEPARATOR_CARDS

Data type

bool

Description

Determines whether the separator cards are printed.

Notes

- The PrintSortReports, PreviewSortReports and SaveReportsAsPDF functions will produce all reports whose properties are set to true.
- The default value for this property is false.

REPORT_PRINT_SUMMARY_REPORT

Data type

bool

Description

Determines whether the summary report is printed.

Notes

- The PrintSortReports, PreviewSortReports and SaveReportsAsPDF functions will produce all reports whose properties are set to true.
- The default value for this property is false.

REPORT_PRINT_TIE_ON_TAGS

Data type

bool

Description

Determines whether the tie on tags are printed.

Notes

- The PrintSortReports, PreviewSortReports and SaveReportsAsPDF functions will produce all reports whose properties are set to true.
- The default value for this property is false.

REPORT_PRINTER_ACCURACY_REPORT

Data type

string

Description

Specifies the printer to use for the Statement of Accuracy Report.

Notes

- The default value for this property is an empty string.

REPORT_PRINTER_CONTAINER_LABELS

Data type

string

Description

Specifies the printer to use for the container labels.

Notes

- The default value for this property is an empty string.

REPORT_PRINTER_FACING_SLIPS

Data type

string

Description

Specifies the printer to use for the facing slips.

Notes

The default value for this property is an empty string.

REPORT_PRINTER_KEEPER_TAGS

Data type

string

Description

Specifies the printer to use for the keeper tags.

Notes

The default value for this property is an empty string.

REPORT_PRINTER_PACKING_REPORT

Data type

string

Description

Specifies the printer to use for the packing report report.

Notes

The default value for this property is an empty string.

REPORT_PRINTER_SEPARATOR_CARDS

Data type

string

Description

Specifies the printer to use for the separator cards.

Notes

- The default value for this property is an empty string.

REPORT_PRINTER_SUMMARY_REPORT

Data type

string

Description

Specifies the printer to use for the summary report.

Notes

The default value for this property is an empty string.

REPORT_PRINTER_TIE_ON_TAGS

Data type

string

Description

Specifies the printer to use for the tie on tags.

Notes

- The default value for this property is an empty string.

REPORT_SAVE_ELECTRONIC_PLAN

Data type

bool

Description

Specifies whether to save the electronic mailing plan files.

Notes

The default value for this property is false.

SEPARATOR_CARD_BOTTOM_MARGIN

Data type

int

Description

Specifies how much white space each separator card will have below the address.

Notes

The default value for this property is 0.

SEPARATOR_CARD_COLUMNS

Data type

int

Description

Specifies the number of columns for separator cards.

Notes

The default value for this property is 0.

SEPARATOR_CARD_CONTINUOUS

Data type

bool

Description

Specifies whether separator cards are to be printed on continuous paper.

Notes

- Set to true if separator cards are to be printed on a dot-matrix printer.
- The default value for this property is false.

SEPARATOR_CARD_HORIZONTAL_PAGE_OFFSET

Data type

int

Description

Specifies the horizontal offset for separator cards in millimeters.

Notes

The default value for this property is 0.

SEPARATOR_CARD_RIGHT_MARGIN

Data type

int

Description

Specifies how much white space each separator card will have to the right of the address.

Notes

The default value for this property is 0.

SEPARATOR_CARD_ROWS

Data type

int

Description

Specifies the number of rows per page of separator cards.

Notes

The default value for this property is 0.

SEPARATOR_CARD_VERTICAL_PAGE_OFFSET

Data type

int

Description

Specifies the vertical offset for separator cards in millimeters.

Notes

The default value for this property is 0.

SETTINGS_BATCH_PROCESS_FIRST

Data type

bool

Description

Determines if address correction is performed prior to sorting the list.

Notes

- Set to true to perform address correction prior to sorting.
- The default value for this property is false.

SETTINGS_DATAFILE_LOCATION

Data type

string

Description

Specifies the location of the address correction files.

Notes

- Setting this property will update the file path in the mrtkca.ini file.
- The default value for this property is an empty string.

SETTINGS_FIELD_LIST_IN

Data type

CAAddressFieldList

Description

A CAAddressFieldList object that defines which fields are supplied as input to the sort engine.

Notes

- This property defines the input fields that are contained in each CAAddressRecord to be processed.
- The input field list should contain, at a minimum, POSTAL_CODE.

SETTINGS_FIELD_LIST_OUT

Data type

CAAddressFieldList

Description

A CAAddressFieldList object that defines which fields are returned as output from the sort engine.

Notes

This property defines the output fields that are contained in each CAAddressRecord that has been processed.

SETTINGS_HIDE_PROGRESS_AFTER_SORT

Data type

bool

Description

Determines if the progress dialog box remains visible after processing is complete.

Notes

- Set to true to hide the progress dialog box after processing is complete.
- The sort reports can be previewed or printed from the progress dialog box.
- The default value for this property is false.

SETTINGS_INI_FILE_NAME

Data type

string

Description

Specifies the full name and path of the ini file to use.

Notes

- The default value for this property is “MRTKCA.INI.”
- Unless specified otherwise, the path is the Windows folder.

SETTINGS_INPUT_BLOCK_RECORD_COUNT

Data type

int

Description

Specifies the number of records contained in an address block.

Notes

- This property specifies the number of CAAddressRecord objects contained in a CAAddressRecordBlock object.
- The size of the record block determines the number of records sent with each call to Send.

SETTINGS_MAILROOM_SERVER_LIST

Data type

string

Description

Specifies the location of the BCC Architect Server.

Notes

- Setting this property creates a TCP/IP connection to the BCC Architect Server, which can reside on the local network or virtually anywhere.
- This property should be set before calling PrepareTask.
- The format is: Server Name (or IP Address):Port.
- Currently, going outside of the proxy server might not be supported.
- The default value for this property is an empty string.

SETTINGS_RECORD_COUNT

Data type

int

Description

Specifies the total number of records to be processed.

Notes

The default value for this property is 0.

SETTINGS_SHOW_PRINT_DIALOG

Data type

bool

Description

Determines whether the standard print dialog box is displayed before printing.

Notes

The default value for this property is false.

SETTINGS_SHOW_SORT_PROGRESS

Data type

bool

Description

Determines if the progress dialog box is displayed during processing.

Notes

- Set to true to display the progress dialog box during processing.
- The sort reports can be previewed or printed from the progress dialog box after processing is complete.
- The default value for this property is true.

SETTINGS_SHOW_STATEMENT_OF_ACCURACY_CHECKBOX_IN_WIZARD

Data type

bool

Description

Determines if the Statement of Accuracy checkbox is visible in the Sort Wizard and the progress dialog box.

Notes

- Set to true to show the Statement of Accuracy check box; false to hide it.
- You may want to set this property to true when SETTINGS_BATCH_PROCESS_FIRST is true.
- The default value for this property is false.

SETTINGS_SORT_WIZARD_CAPTION

Data type

string

Description

Specifies the caption that appears in the title bar of the Sort Wizard.

Notes

The default value for this property is “Sort Wizard.”

SETTINGS_TEMPLATE_NAME_TO_USE

Data type

string

Description

Specifies the name of the template to use for the sort.

Notes

The default value for this property is an empty string.

SORT_CONTAINER_TYPE

Data type

string

Description

Indicates the type of container to be used to send this mailing.

Notes

SORT_CUSTOMER_ADDRESS

Data type

string

Description

The address of the person sending the mailing.

Notes

- This will be printed on the postal summary report.
- The default value for this property is an empty string.

SORT_CUSTOMER_COMPANY_NAME

Data type

string

Description

The name of the company sending the mailing.

Notes

- This will be printed on the postal summary report.
- The default value for this property is an empty string.

SORT_CUSTOMER_ID

Data type

string

Description

The Canada Post identification number of the person sending the mailing.

Notes

- This will be printed on the postal summary report.
- The default value for this property is an empty string.

SORT_CUSTOMER_NAME

Data type

string

Description

The name of the person sending the mailing.

Notes

- This will be printed on the postal summary report.
- The default value for this property is an empty string.

SORT_CUSTOMER_POSTAL_CODE

Data type

string

Description

The Postal Code of the person sending the mailing.

Notes

- This will be printed on the postal summary report.
- The default value for this property is an empty string.

SORT_CUSTOMER_TELEPHONE

Data type

string

Description

The phone number of the person sending the mailing.

Notes

- This will be printed on the postal summary report.
- The default value for this property is an empty string.

SORT_DELIVERY_OFFICE

Data type

string

Description

The delivery office at which you will deposit this mailing.

Notes

- This will be printed on the postal summary report.
- The default value for this property is an empty string.

SORT_MAILING_DATE

Data type

string

Description

Specifies the date of the mailing.

Notes

The default value for this property is the current date.

SORT_MAILING_TYPE

Data type

string

Description

Specifies the type of mailing that you are sending.

Notes

The default value for this property is an empty string.

SORT_PIECE_LENGTH

Data type

int

Description

Specifies the length of a mail piece in millimeters.

Notes

- The default value for this property is 0.

SORT_PIECE_THICKNESS

Data type

int

Description

Specifies the thickness of a mail piece in millimeters.

Notes

- The default value for this property is 0.

SORT_PIECE_WEIGHT

Data type

int

Description

Specifies the weight of a mail piece in grams.

Notes

- The default value for this property is 0.

SORT_PIECE_WIDTH

Data type

int

Description

Specifies the width of a mail piece in millimeters.

Notes

- The default value for this property is 0.

SORT_PUB_MAIL_DEPOSIT_POSTAL_CODE

Data type

string

Description

Specifies the Postal Code of the office of deposit for a Publications mailing.

Notes

- The default value for this property is an empty string.

SORT_STATEMENT_OF_MAILING_ID

Data type

string

Description

The identification string for the Statement of Mailing report.

Notes

The default value for this property is an empty string.

SORT_TEMPLATE_LIST

Data type

string

Description

Returns a semicolon-delimited string of all currently defined presort templates.

Notes

- This property returns the template list for the current ini file as specified by the SETTINGS_INI_FILE_NAME property.
- The default value for this property is an empty string.

TIE_ON_TAGS_BOTTOM_MARGIN

Data type

int

Description

Specifies how much white space each tie on tag will have below the address.

Notes

The default value for this property is 0.

TIE_ON_TAGS_COLUMNS

Data type

int

Description

Specifies the number of columns for tie on tags.

Notes

The default value for this property is 0.

TIE_ON_TAGS_CONTINUOUS

Data type

bool

Description

Specifies whether tie on tags are to be printed on continuous paper.

Notes

- Set to true if tie on tags are to be printed on a dot-matrix printer.
- The default value for this property is false.

TIE_ON_TAGS_HORIZONTAL_PAGE_OFFSET

Data type

int

Description

Specifies the horizontal offset for tie on tags in millimeters.

Notes

The default value for this property is 0.

TIE_ON_TAGS_RIGHT_MARGIN

Data type

int

Description

Specifies how much white space each tie on tag will have to the right of the address.

Notes

The default value for this property is 0.

TIE_ON_TAGS__ROWS

Data type

int

Description

Specifies the number of rows per page of tie on tags.

Notes

The default value for this property is 0.

TIE_ON_TAGS_VERTICAL_PAGE_OFFSET

Data type

int

Description

Specifies the vertical offset for tie on tags in millimeters.

Notes

The default value for this property is 0.

SORTAssembly Reports

The SORTAssembly reports are members of the CAReports.Sort enumeration and are defined below. These enum names are used as arguments of the various functions to preview, print or save a report. If you have added the Satori.Architect.CA reference to your .NET project, then you can view all of the available CAReports.Sort enums in the Object Browser and IntelliSense.

Name	Description
CONTAINER _LABELS	Container/bag labels; this report is required for postal discounts.
ELECTRONIC_PLAN	Not a report, but files that allow for electronic mailing submission.
FACING_SLIPS	Labels that must be affixed to every bundle that is not being delivered to a Level 1 Delivery Facility.
KEEPER_TAGS	Small labels that must be inserted in the label holder of each bag of mail.
MAILING_SUMMARY	Details the specifics of your mailing, including the postage for various bundles. Required for discounts.
PACKING	The packing report.
SEPARATOR_CARDS	These cards separate individual groupings of mail within a hardsided container.
STATEMENT_OF_ ACCURACY	Statement of Accuracy report that details correction results; this report is required for automation sorts.
TIE_ON_TAGS	Labels that must be tied to each bag of mail. These labels detail the contents of the bag.

SORTAssembly Fields

The SORTAssembly fields are members of the CAFields.Sort enumeration and are defined below. These enum names are used as arguments of the various functions of the CAAddressFieldList and CAAddressFields objects. If you have added the Satori.Architect.CA reference to your .NET project, then you can view all of the available CAFields.Presort enums in the Object Browser and IntelliSense.

Name	Description
ADDRESS_LINE_1	
ADDRESS_LINE_2	
BUNDLE_NUMBER	
BUSINESS	
CONTAINER_NUMBER	
COUNTRY	
DELIVERY_MODE	
FIRST_NAME	
LAST_NAME	
MUNICIPALITY	
PIECE_POSTAGE	
POSTAL_CODE	
PRESORT_ID	
PROVINCE	
PUB_MAIL_DISTANCE_CODE	
RECORD_ID	
USER_DEFINED_1	
USER_DEFINED_2	
USER_DEFINED_3	
USER_DEFINED_4	
USER_DEFINED_5	
USER_DEFINED_6	
USER_DEFINED_7	
USER_DEFINED_8	
USER_DEFINED_9	
USER_DEFINED_10	
USER_DEFINED_11	
USER_DEFINED_12	

USER_DEFINED_13	
USER_DEFINED_14	
USER_DEFINED_15	

Web Services Reference

Contents

The Web Services POSTALCODEService Interface for Correcting Single Addresses

The POSTALCODEService is a Web Service interface to the PostalCodeTask library. BCC Software provides this interface as an alternative to the COM and .NET interfaces. This service takes a single address, corrects it using a SERP Certified address correction process and returns it to you.

The POSTALCODEService uses a single function: CheckAddress. This function takes an address, your registration and Add-on key credentials and a properties object. It returns the processed address. With just a few lines of code, you can process an address to make it more complete and more likely to arrive at its destination.

Overview

Use the following procedure to process an address with the POSTALCODEService:

1. Add a Web reference to the POSTALCODEService. The address of the service is https://ws.satorisoftware.com/Architect/CA/POSTALCODE/POSTALCODEService.asmx.
2. Create a POSTALCODEService object.
3. Create a POSTALCODEService.Credentials object and assign the ProductKey and AddOnKeys.
4. Create a POSTALCODEService.POSTALCODEServiceProperties object and set the properties as you wish. We recommend that you set all properties. See the Properties section for more information.
5. Create a POSTALCODEService.POSTALCODEAddress object and build your address with this object.
6. Call CheckAddress on these three objects. Make sure you store the return value using a POSTALCODEAddressOutput object.

POSTALCODEService Functions

Below are the functions, properties and fields available in POSTALCODEService. If you have added a Web reference to the POSTALCODEService in Visual Studio, you can view all of these functions in the Object Browser.

Note that in the function below, we use the namespace MyPOSTALCODEServiceReference. This may be different on your system, depending on the development environment. The function example below assumes that the development environment is C#; other environments may use different syntax.

CheckAddress

Syntax

```
MyPOSTALCODEServiceReference. CheckAddress(MyPOSTALCODEServiceReference.Credentials,
MyPOSTALCODEServiceReference.POSTALCODEServiceProperties, MyPOSTALCODESer-
viceReference.POSTALCODEAddress);
```

Description

Processes a single address and attempts to correct it.

Parameters

Credentials

A MyPOSTALCODEServiceReference.Credentials object that contains two objects: ProductKey, a string that contains your BCC Architect license key, and AddOnKeys, an array of strings that contain license keys for your Add-ons.

POSTALCODEServiceProperties

A MyPOSTALCODEServiceReference.POSTALCODEServiceProperties object that contains the settings for all properties. We recommend that you set all property values. See below for the list of available properties.

POSTALCODEAddress

A MyPOSTALCODEServiceReference.POSTALCODEAddress object that contains the address you want to process. See below for the available fields in this object.

Return values

A MyPOSTALCODEServiceReference. POSTALCODEAddressOutput object that contains your processed address. See below for the valid fields in this object.

Notes

The web service is configured to accept a maximum of eight characters. Therefore, when you pass a US ZIP+4 postal code to the POSTALCODEService.CheckAddress method, the web service returns a truncated value. It uses a ">" as the ninth character to indicate that the field length has been exceeded.

See also

POSTALCODEService Properties

The properties listed below are values within the POSTALCODEServiceProperties structure. To set properties for your POSTALCODEService process, instantiate a POSTALCODEServiceProperties object, set the member properties listed below and pass this structure to the CheckAddress() function.

We recommend that you set all property values. Any property that you do not set will contain the default value, which is usually false, 0 or the first enumeration value. However, this may not always be the case for all programming environments. So, to make sure that you receive the best results for your application, you should set every property.

Note that in all the properties below, we use the namespace MyPOSTALCODEServiceReference. This may be different on your system, depending on the development environment.

Casing

Data type

string

Description

A read-write property that determines the casing format applied to the address elements.

Notes

- Use one of the following:
- MyPOSTALCODEServiceReference.Casing.eUpper – The address will be returned in all UPPER CASE. Default value.
- MyPOSTALCODEServiceReference.Casing.eMixed – The address will be returned in Mixed Case.
- This property should be set before calling CheckAddress.

POSTALCODEService Fields

The items below are members of the CheckAddress structure.

For an effective POSTALCODEService process, we recommend that you use at least the following fields as input: AddressLine1, Municipality, Province and PostalCode.

Note that in all the fields below, we use the namespace MyPOSTALCODEServiceReference. This may be different on your system, depending on the development environment.

AddressLanguage

Data Type

string

Description

The language in which the address is written. E for English or F for French.

Notes

-

See also

AddressLine1

Data Type

string

Description

The top address line, above the municipality, province and Postal Code.

Notes

- If there are two address lines, AddressLine2 will contain the primary address line and AddressLine1 will contain the secondary address information.
- Both AddressLine1 and AddressLine2 are formatted according to the values of the various formatting properties.

See also

- CheckAddress
- AddressLine2Casing

AddressLine2

Data Type

string

Description

The bottom address line, below AddressLine1 and above the municipality, province and Postal Code.

Notes

- If there are two address lines, AddressLine2 will contain the primary address line and AddressLine1 will contain the secondary address information.
- Both AddressLine1 and AddressLine2 are formatted according to the values of the various formatting properties.

See also

- CheckAddress
- AddressLine1
- Casing

BusinessName

Data Type

string

Description

The business name.

Notes

- The business name is optional.

See also

- CheckAddress function
- Casing

CASSDate

Data Type

int

Description

Information, such as the issue date, about the last attempt to check an address.

Notes

- CheckAddress should be called before attempting to retrieve the value of this property.

See also

- CheckAddress

CivicNumber

Data Type

string

Description

The civic number for this address.

Notes

- This value is included in the street address, before the street name.
- CheckAddress should be called before attempting to retrieve the value of this property.

See also

- CheckAddress
- AddressLine1
- AddressLine2

CivicNumberSuffix

Data Type

string

Description

The civic number suffix of an address.

Notes

- This value is included in the street address, before the street name.
- Civic number suffixes are usually letters or fractions.
- CheckAddress should be called before attempting to retrieve the value of this property.

See also

- CheckAddress
- AddressLine1
- AddressLine2

ErrorCode

Data Type

int

Description

The code that indicates the results of an address-matching attempt.

Notes

- The error code indicates whether an address was matched to the Canada Post database. It also provides information about the changes, if any, that were made to the input address in order to correct it, or the reason an address was unable to be corrected.
- An error code less than 100 indicates that the address was corrected, while an error code greater than 100

indicates that the address was unable to be corrected.

- CheckAddress should be called before attempting to retrieve the value of this property.
- See the Error Codes table for a complete list of error code values and their descriptions.

See also

- CheckAddress
- ErrorCodeString

ErrorCodeString

Data Type

string

Description

The description of the results of an address-matching operation.

Notes

- CheckAddress should be called before attempting to retrieve the value of this property.
- The error description corresponds to the value of the ErrorCode property.

See also

- CheckAddress function
- ErrorCode

GDType

Data Type

string

Description

The general delivery type of an address.

Notes

- There are two types of general delivery addresses:
- STN – Station
- RPO – Retail Postal Outlet
- CheckAddress should be called before attempting to retrieve the value of this property.

See also

- StationName
- StationQualifier
- StationType

LastLine

Data Type

string

Description

The last line of the address block.

Notes

- You can retrieve this field instead of Municipality, Province and PostalCode
- CheckAddress should be called before attempting to retrieve the value of this property.

See also

LVRBranchName

Data Type

string

Description

The branch name for the Large Volume Receiver.

Notes

- CheckAddress should be called before attempting to retrieve the value of this property.

See also

LVRName

Data Type

string

Description

The name of the Large Volume Receiver.

Notes

- CheckAddress should be called before attempting to retrieve the value of this property.

See also

Municipality

Data Type

string

Description

The municipality.

Notes

- CheckAddress should be called before attempting to retrieve the value of this property.

See also

- CheckAddress

NonStandardExtra

Data Type

string

Description

Nonstandard extra information.

Notes

-

See also

POBoxNumber

Data Type

string

Description

The PO Box number for an address.

Notes

- This information may be contained in the address line properties.
- CheckAddress should be called before attempting to retrieve the value of this property.

See also

- CheckAddress
- AddressLine1

- AddressLine2

PostalCode

Data Type

string

Description

The Postal Code.

Notes

- This property is essential to a complete address.
- After a successful call to CheckAddress, this propertywill be formatted with a space separating the Forward Sortation Area (first three characters) and the Local Delivery Unit (last three characters).

See also

- CheckAddress

Province

Data Type

string

Description

The province.

Notes

- This property is essential to a complete address.

See also

- CheckAddress

RecordType

Data Type

string

Description

Indicates the type of address that this is.

Notes

- CheckAddress should be called before attempting to retrieve the value of this property.
- Contains one of the following characters:

- '1' - street address (street)
- '2' - street served by route address (ssbr)
- '3' - po box address (box)
- '4' - route delivery address (route)
- '5' - general delivery address (gd)

See also

RouteNumber

Data Type

string

Description

The rural route number for an address.

Notes

- This information may be contained in the address line properties.
- CheckAddress should be called before attempting to retrieve the value of this property.

See also

- CheckAddress
- AddressLine1
- AddressLine2

RouteType

Data Type

string

Description

The type of a route service.

Notes

- This information may be contained in the address line properties.
- Contains one of the following:
  - RR – Rural Route
  - SS – Suburban Service

    - MR – Mobile Route
    - GD – General Delivery
- CheckAddress should be called before attempting to retrieve the value of this property.

See also

- CheckAddress
- AddressLine1
- AddressLine2

StandardExtra

Data Type

string

Description

Standard extra information.

Notes

- CheckAddress should be called before attempting to retrieve the value of this property.

See also

StationName

Data Type

string

Description

The name of the area in which the station resides.

Notes

- Also called the Delivery Installation Area Name.
- CheckAddress should be called before attempting to retrieve the value of this property.

See also

StationQualifier

Data Type

string

Description

The station qualifier.

Notes

- If an area contains multiple delivery stations, this property will uniquely identify the specific delivery station.
- Also called the Delivery Installation Qualifier Name.
- CheckAddress should be called before attempting to retrieve the value of this property.

See also

- StationName

StationType

Data Type

string

Description

The station type. Also called the Delivery Installation Type.

Notes

- Contains one of the following:
    - BDP – Bureau De Poste
    - CC – Concession Commerciale
    - CDO – Commercial Dealership Outlet
    - CMC – Community Mail Centre
    - CPC – Centre Postal Communautaire
    - CSP – Comptoir Service Postal
    - LDVD – Letter Carrier Depot
    - PDF – Poste De Facteurs
    - PO – Post Office
    - RPO – Retail Postal Outlet
    - STN – Station
    - SUCC – Succursale
- CheckAddress should be called before attempting to retrieve the value of this property.

See also

StreetDirection

Data Type

string

Description

The street direction prefix, if any.

Notes

- CheckAddress should be called before attempting to retrieve the value of this property.
- An example of a street direction would be "NW" in "416 NW Lake St."
- This information may be contained in the address line properties.

See also

- CheckAddress
- AddressLine1
- AddressLine2

StreetName

Data Type

string

Description

The street name.

Notes

- CheckAddress should be called before attempting to retrieve the value of this property.
- An example of a street name would be "Lake" in "416 NW Lake St."
- This information may be contained in the address line properties.

See also

- CheckAddress
- AddressLine1
- AddressLine2

StreetType

Data Type

string

Description

The street type.

Notes

- CheckAddress should be called before attempting to retrieve the value of this property.
- An example of a street type would be "St." in "416 NW Lake St."
- This information may be contained in the address line properties.

See also

- CheckAddress
- AddressLine1
- AddressLine2

UnitDesignator

Data Type

string

Description

The unit designator.

Notes

- CheckAddress should be called before attempting to retrieve the value of this property.
- An example of a unit designator is "Suite" in “301 Maryland Ave Suite 1.”
- This information may be contained in the address line properties.

See also

- CheckAddress

UnitNumber

Data Type

string

Description

The unit number.

Notes

- Call CheckAddress before attempting to retrieve the value of this property.
- An example of a unit number would be "10" in "10-123 Main St." or “1” in “301 Maryland Ave Suite 1.”

- This information may be contained in the address line properties.

See also

- CheckAddress

POSTALCODEService Field Names Summary Table

Field Name	Data Direction	Data Type	Description
AddressLanguage	out	string	The language in which the address is written. E for English or F for French.
AddressLine1	in / out	string	The top address line, above the municipality, province and Postal Code.
AddressLine2	in / out	string	The bottom address line, below AddressLine1 and above the municipality, province and Postal Code.
BusinessName	in / out	string	The business name.
CASSDate	out	int	Information, such as the issue date, about the last attempt to check an address.
CivicNumber	out	string	The civic number for this address.
CivicNumberSuffix	out	string	The civic number suffix of an address.
ErrorCode	out	int	The code that indicates the results of an address-matching attempt.
ErrorCodeString	out	string	The description of the results of an address-matching operation.
GDType	out	string	The general delivery type of a address.
LastLine	out	string	The last line of the address block.
LVRBranchName	out	string	The branch name for the Large Volume Receiver.
LVRName	out	string	The name of the Large Volume Receiver.
Municipality	in / out	string	The municipality.
NonStandardExtra	out	string	Nonstandard extra information
POBoxNumber	out	string	The PO Box number for an address.
PostalCode	in / out	string	The Postal Code.
Province	in / out	string	The province.
RecordType	out	string	
RouteNumber	out	string	The rural route number for an address.
RouteType	out	string	The type of a route service.
StandardExtra	out	string	Standard extra information

StationName	out	string	The name of the area in which the station resides.
StationQualifier	out	string	The station qualifier.
StationType	out	string	The station type. Also called the Delivery Installation Type.
StreetDirection	out	string	The street direction prefix, if any.
StreetName	out	string	The street name.
StreetType	out	string	The street type.
UnitDesignator	out	string	The unit designator.
UnitNumber	out	string	The unit number.

Results Codes

Address Correction Results and Error Codes

The error codes below indicate either the success or failure of an attempt to certify an address (see ErrorCode property). Codes 0 – 99 and 500 – 599 are considered successful.

Value/Range	Description
Coded	
0	Coded with no changes
1-99	Warnings
62	Street name changed
63	Street type changed
64	Street direction changed
65	Street number changed
66	Unit keyword changed
67	Unit number changed
68	Station name changed
69	Station type changed
70	Station qualifier changed
71	PO Box keyword changed
72	PO Box number changed
73	Route number changed
74	Route keyword changed
75	GD keyword changed
76	Municipality changed
77	Province changed
78	Postal code changed
79	Extra information changed
80	Address format changed
81	Address changed

82	Matched to LVR Postal Code
83	Matched to General Delivery Postal Code
Errors	
100-499	Not Certified
151	Address and postal code conflict
152	Street and postal code conflict
153	Delivery mode changed
154	Foreign address
155	No match
Coded	
500-599	Certify Errors

Additional Resources

The following resources are available to help you with your software.

Knowledge Base

Did you know? BCC Software offers tips, tricks, and best practices for using our products. Knowledge Base and Support articles can help empower both experts and new users. To learn more, visit the BCC Knowledge Base.

How to Contact Support

- BCC Software Community online: https://community.satorisoftware.com/community/s/contactsupport
- Email: support@bccsoftware.com