Boldon James
A QINETIQ company

Classifier API

UM642705

November 18

Customer Documentation

Contents

1 INTRODUCTION

Boldon James Classifier API enables limited Classifier functionality to be utilized programmatically. The API functionality includes reading, editing, comparing and applying labels to files as well as querying available selectors and values in a Classifier configuration.

Note: There is currently no support for Streams.

Note: ***SISL*** refers to the internal encoded format of a Classifier classification (label).

1.1 Classifier Policy Control

The Classifier API operates using the current Classifier label configuration. Refer to the Classifier Administration Guide for further information.

2 INSTALLATION

Run the supplied installation file (ClassifierAPI_x86.msi or ClassifierAPI_x64.msi).

3 API METHODS

3.1 Object Instantiation

An instance of a Classifier API COM object can be instantiated using the standard COM mechanisms and its ProgID (“BoldonJames.Classifier.API”). Some examples are given below:

3.1.1 Visual C++

- Import the Classifier API typelib into a project with the following code:

```
#import "ClassifierAPI.tlb"
```

- Create an instance of the Classifier API object using the following sample code:

```
CoInitialize(NULL);
ClassifierAPI::IClassifierAPI* pClassifierAPI = NULL;
CLSID;
HRESULT hr = CLSIDFromProgID(_T("BoldonJames.Classifier.API"), &clsid);
if (SUCCEEDED(hr))
{
	hr = CoCreateInstance(clsid, NULL, CLSCTX_INPROC_SERVER,
	__uuidof(ClassifierAPI::IClassifierAPI), (void**)&pClassifierAPI);
	if (SUCCEEDED(hr))
	{
		// Perform relevant operations using the object
		...

		// Finished with the object
		pClassifierAPI->Release();
	}
}
```

3.1.2 C#

- Add a reference to “ClassifierAPI.dll” to the project
- Create an instance of the Classifier API object using the following sample code:

```
Type t = Type.GetTypeFromProgID("BoldonJames.Classifier.API");
BoldonJames.Classifier.IClassifierAPI classifierApi = Activator.CreateInstance(t)
as BoldonJames.Classifier.IClassifierAPI;
if (classifierApi != null)
{
    // Perform relevant operations using the object
```

3.2 Initialise

eReturnCode Initialise(string culture, string licenceFileContent)

eReturnCode Initialise(string culture, string licenceFileContent, bool useServiceMode)

Initialise is responsible for loading the Classifier configuration. Having instantiated a Classifier COM object, a call to **Initialise** must be made before calling any other function on the object.

The call can fail if a valid Classifier configuration cannot be found or if the licence contents are invalid.

Note: The API looks for the configuration in the same way as the Classifier Client or, if useServiceMode is set to true, like a service product. Refer to the Classifier Administration Guide for more information.

Parameters:

[in] culture

Type: string

Description: Name of the culture to use (e.g. “EN-US”, “FR”). If this parameter is null or empty, the API will use the current thread culture. The parameter is only required with multiple-language Classifier configurations.

[in] licenceFileContent

Type: string

Description: A string of the contents from a valid Classifier licence file (impCPI.lic).

[in] useServiceMode

Type: bool

Description: Specify if you want the API to run in Service Mode. Default is false.

Return Value:

Type: eReturnCode

Ok if the call succeeded.

If the call failed, the following may be returned:

- **BadCulture**.
- **Unlicenced**.

- **InvalidConfig**.

3.3 GetClassification

eReturnCode GetClassification(string fileName, out string SISL)

eReturnCode GetClassification(IntPtr handle, string filename, out string SISL)

GetClassification is used to obtain the Classifier classification (as a string) from the given file.

Parameters:

[in] handle

Type: IntPtr

Description: File handle that will be used to obtain the classification. If the file type is not supported, fileName is used to get the marking from ADS.

[in] fileName

Type: string

Description: Name of the file (including path) from which to obtain a Classifier classification.

[out] SISL

Type: string

Description: Classifier classification in its SISL format. This string can then be used as a parameter in other methods to get/set different elements of the classification.

Return Value:

Type: eReturnCode

Ok if the call succeeded. This includes if the file is unlabelled.

If the call failed, the following may be returned:

- **Uninitialised**
- **Unlicenced**
- **InvalidFile**
- **InvalidHandle**
- **CannotDecodeLabel**
- **UnsupportedFileType**

3.4 GetLabelSelectorNames

eReturnCode GetLabelSelectorNames(string SISL, out string[] names)

GetLabelSelectorNames is used to obtain a string array of all selector names in a given classification.

Parameters:

[in] SISL

Type: string

Description: Classifier classification in its own internal label format. This string can be obtained from a call to GetClassification

[out] names

Type: string[]

Description: String array of the names of all selectors within the SISL. If the SISL is invalid or empty, this parameter will be returned as an empty string array.

Return Value:

Type: eReturnCode

Ok if the call succeeded. This includes when the label has no selectors applied.

If the call failed, the following may be returned:

- **Uninitialised**
- **Unlicenced**
- **CannotDecodeLabel.**

3.5 GetLabelValues

eReturnCode GetLabelValues(string SISL, out string[] values)

GetLabelValues is used to obtain a string array of all selector values in a given classification.

Parameters:

[in] SISL

Type: string

Description: Classifier classification in its own internal label format. This string can be obtained from a call to GetClassification

[out] values

Type: string[]

Description: String array of all values within the SISL. Returns an empty string if there are no values or the SISL is invalid.

Return Value:

Type: eReturnCode

Ok if the call succeeded. This includes when there are no values applied to the SISL (an empty label).

If the call failed, the following may be returned:

- **Uninitialised**
- **Unlicenced**
- **CannotDecodeLabel**

3.6 GetSelectorValue

eReturnCode GetSelectorValue(string SISL, string selectorName, out string selectorValue)

GetSelectorValue is used to obtain the value for a given selector from a label in SISL format.

Parameters:

[in] SISL

Type: string

Description: Label classification in SISL format. This string can be obtained from a call to **GetClassification**

[in] selectorName

Type: string

Description: Name of a selector in the Classifier configuration. Alternate names not applicable.

[out] selectorValue

Type: string

Description: The language-specific name of the value in the specified Classifier classification for the specified selector.

Return Value:

Type: eReturnCode

Ok if the call succeeded.

If the call failed, the following may be returned:

- **Uninitialised**
- **Unlicenced**
- **InvalidSelector**
- **WrongSelectorType**
- **CannotDecodeLabel**

3.7 GetMultiSelectorValues

eReturnCode GetMultiSelectorValues(string SISL, string selectorName, out string[] selectorValues)

GetMultiSelectorValues is used to obtain the value for a given multi-selector from a Classifier classification.

Parameters:

[in] SISL

Type: string

Description: Classifier classification in its SISL label format. This string can be obtained from a call to **GetClassification**

[in] selectorName

Type: string

Description: Name of a selector in the Classifier configuration. Alternate names not applicable.

[out] selectorValues

Type: string[]

Description: The language-specific, if applicable, name(s) of the value in the specified Classifier classification for the specified selector.

Return Value:

Type: eReturnCode

Ok if the call succeeded.

If the call failed, the following may be returned:

- **Uninitialised**
- **Unlicenced**
- **SelectorNotFound**
- **InvalidSelector**
- **WrongSelectorType**
- **CannotDecodeLabel**

3.8 SetClassification

eReturnCode SetClassification(string fileName, string SISL, bool refreshRequired)

eReturnCode SetClassification(IntPtr handle, string filename, string SISL, bool refreshRequired)

SetClassification is used to set the Classifier classification for the given file.

Parameters:

[in] handle

Type: IntPtr

Description: File handle that will be used to set the classification. If the file is not supported, fileName will be used instead.

[in] fileName

Type: string

Description: Name of the file (including path) for which the Classifier classification is to be set.

[in] SISL

Type: string

Description: The classification in SISL format. This string can be derived using other methods on the API.

[in] refreshRequired

Type: bool

Description: Sets a property on the file. If **True**, Office Classifier will update any visual markings (e.g. header/footer) the next time that the document is opened. This ensures that the visual markings on the document match the updated classification.

Return Value:

Type: eReturnCode

Ok if the call succeeded.

If the call failed, the following may be returned:

- **Uninitialised**
- **Unlicenced**
- **InvalidFile**
- **InvalidHandle**
- **UnsupportedFileType**
- **CannotDecodeLabel**
- **CantWriteToFile**
- **CantWriteViaHandle**

3.9 SetSelectorValue

eReturnCode SetSelectorValue(string currentSISL, string selectorName, string selectorValue, out string newSISL)

SetSelectorValue is used to set the value for a given selector in a Classifier classification. In order to clear the value for a selector, a null/empty string should be specified for the value.

Parameters:

[in] currentSISL

Type: string

Description: Label classification in SISL format. This string can be obtained from a call to **GetClassification**.

[in] selectorName

Type: string

Description: Name of a selector in the Classifier configuration.

[in] selectorValue

Type: string

Description: The name of the value in the specified Classifier classification for the specified selector that should be set in the Classifier classification.

This parameter must match the name value in the Culture the API is operating in.

Note: For ‘Date Picker’ selectors, the value should be in the form “yyyyMMdd”

[out] newSISL

Type: string

Description: The updated Classifier classification in its own internal label format after setting the specified selector value.

Return Value:

Type: eReturnCode

Ok if the call succeeded.

If the call failed, the following may be returned:

- **Uninitialised**
- **Unlicenced**
- **InvalidSelector**
- **InvalidSelectorValue**
- **CannotDecodeLabel**
- **WrongSelectorType**
- **SelectorNotFound**

3.10 SetMultiSelectorValues

eReturnCode SetMultiSelectorValues(string currentSISL, string selectorName, string[] selectorValues, out string newSISL)

SetMultiSelectorValues is used to set the values for a given selector in a Classifier classification. In order to clear the value for a selector, a null/empty string should be specified as the value.

Parameters:

[in] currentSISL

Type: string

Description: Classifier classification in SISL format. This string can be obtained from a call to **GetClassification**

[in] selectorName

Type: string

Description: Name of a selector in the Classifier configuration.

[in] selectorValues

Type: string[]

Description: Name(s) of the values in the specified classification that should be set for the specified selector.

The selector value names should match their names in the culture the API was initialised with.

[out] newSISL

Type: string

Description: The updated Classifier classification in SISL format after setting the specified selector value.

Return Value:

Type: eReturnCode

Ok if the call succeeded.

If the call failed, the following may be returned:

- **Uninitialised**
- **Unlicenced**
- **InvalidSelector**
- **InvalidSelectorValue**
- **WrongSelectorType**
- **SelectorNotFound**
- **CannotDecodeLabel**

3.11 GetSelectorObjects

eReturnCode GetSelectorObjects(out IAPISelector[] selectors)

GetSelectorObjects will return an array of all selector objects in the label configuration.

Parameters:

[out] selectors

Type: IAPISelector[]

Description: Array of selectors in the label configuration.

Return Value:

Type: eReturnCode

Ok if the call succeeded.

If the call failed, the following may be returned:

- **Uninitialised**
- **Unlicenced**
- **SelectorNotFound**

3.12 GetSelectorValueObjects

eReturnCode GetSelectorValueObjects(string SelectorID, out IAPISelectorValue[] values)

GetSelectorValueObjects will return an array of all selector values available for a specific selector.

The object's values will match the culture that the API is using.

Parameters:

[in] SelectorID

Type: string

Description: ID of the selector to return the values of. The selector ID can be determined by calling **GetID** on a IAPISelector object.

[out] values

Type: IAPISelectorValue[]

Description: An array of selector values associated with a specific selector.

Return Value:

Type: eReturnCode

Ok if the call succeeded.

If the call failed, the following may be returned:

- **Uninitialised**
- **Unlicenced**
- **InvalidSelector**
- **SelectorNotFound**

3.13 Compare

eReturnCode Compare(string sisl1, string sisl2)

Compare compares two labels in SISL format to determine if they match, if one label is greater or if they are not comparable.

Parameters:

[in] sisl1

Type: string

Description: first SISL for comparison.

[in] sisl2

Type: string

Description: second SISL for comparison.

Return Value:

Type: eReturnCode

LabelsAreEqual if the labels are the same.

FirstLabelDominates if sisl1 is greater than sisl2.

SecondLabelDominates if sisl1 is less than sisl2.

LabelsAreDifferent if the labels are not the same and neither is greater.

If the call failed, the following may be returned:

- **CannotDecodeLabel**
- **Unlicenced**
- **Uninitialised**

3.14 GetHighWaterMark

eReturnCode GetHighWaterMark(string[] labels, out string hwmk)

GetHighWaterMark takes a collection of labels in SISL format and outputs a SISL representing the high-water mark.

The high-water mark label contains the highest present selector value for each selector in use in a collection of labels.

Parameters:

[in] labels

Type: string[]

Description: Array of strings in SISL format that are used to obtain the high-water mark.

[out] hwmk

Type: string

Description: SISL string that represents the high-water mark of the array of labels.

Return Value:

Type: eReturnCode

Ok if the call succeeds.

CannotCalculateHighWatermark if no valid high-water mark can be determined.

If the call failed, the following may be returned:

- **InvalidCollection**
- **CannotDecodeLabel**
- **Unlicenced**
- **Uninitialised**

3.15 RefreshConfiguration

eReturnCode RefreshConfiguration()

RefreshConfiguration re-gets the label configuration for the API to use.

Return Value:

Type: eReturnCode

Ok if the call succeeds.

If the call failed, the following may be returned:

- **Unlicenced**
- **Uninitialised**

4 SELECTOR OBJECT METHODS

The Selector object, or **IAPISelector**, contains the properties of a selector in a label configuration. The following section details the function call to access these properties.

4.1 GetName

string GetName()

GetName returns the name of the selector.

Return Value:

Type: string

Returns the name of the selector in the current culture.

4.2 GetID

string GetID()

GetID returns the ID of the selector.

Return Value:

Type: string

Returns the selector ID

4.3 GetSelectorType

eSelectorType GetSelectorType()

GetSelectorType returns an enum representing the selector type.

Return Value:

Type: eSelectorType

eSelectorTypes contain the following values:

- Invalid
- SingleSelect
- MultiSelect
- FreeText
- DateSelect
- DateOffset

4.4 IsValid

bool IsValid()

IsValid checks whether the selector is valid. A selector may be invalid if it is instantiated with an invalid Selector ID.

Return Value:

Type: bool

true if the selector is valid.

false if the selector is invalid.

5 SELECTOR VALUE OBJECT METHODS

The Selector Value object, or **IAPISelectorValue**, contains the properties of a selector value within a selector. The following section details the function call to access these properties.

5.1 GetName

string GetName()

GetName returns the name of the selector value.

Return Value:

Type: string

Returns the name of the selector value in the current culture.

5.2 GetID

string GetID()

GetID returns the ID of the selector value.

Return Value:

Type: string

Returns the selector value GUID

5.3 GetColour

string GetColour()

GetColour returns the selector value colour as a hex code string.

Return Value:

Type: string

Returns a string that represents the hex code for the selector value's colour.

5.4 GetPortion

string GetPortion()

GetPortion returns the portion mark string of the selector value.

Return Value:

Type: string

Returns the portion mark string if it exists. Otherwise, returns an empty string

5.5 GetAltName

Note: There are 3 separate functions, GetAltName1(), GetAltName2(), GetAltName3()

string GetAltName1()
string GetAltName2()
string GetAltName3()

The **GetAltName** functions return the selector values of the respective alternate names.

Return Value:

Type: string

Returns an alternate name of the selector value if it exits. Otherwise, returns an empty string.

5.6 GetHierarchyValue

bool GetHierarchyValue(out int value);

GetHierarchyValue is to determine whether or not the selector is hierarchal and if it is, where this value sits in the selector hierarchy.

Parameters:

[out] value

Type: int

Description: The integer value of the selector. The higher the value, the more sensitive the value is deemed. The value will be -1 if the selector is non-hierarchical.

Return Value:

Type: Boolean

true if the selector this value belongs to is hierarchical.

false if the selector this value belongs to is not hierarchical.

6 RETURN CODES

Most Classifier API functions will return a return code. The return code is an enum (**eReturnCode**) and specifies if the call succeeded and the reason for failure if it does not. Below is a list of all the return codes in the Classifier API:

Ok	Call was successful.
InvalidLicence	Licence provided to API is invalid.
Uninitialised	Function call failed because the API has not been initialised.
Unlicenced	Licence provided to the API is invalid or has expired.
InvalidConfig	API is not using a valid Classifier Configuration.
CannotDecodeLabel	The SISL passed in to the function cannot be read. Either it is not a valid SISL, or it belongs to a configuration not in use.
SelectorNotFound	The selector the API is looking for does not exist within a given SISL.
InvalidSelector	The selector is invalid i.e. null or empty.
InvalidSelectorValue	The selector value does not exist or is otherwise invalid.
InvalidFile	File does not exist or is otherwise invalid or corrupt.
InvalidHandle	The file handle is invalid.
CantWriteToFile	The file cannot be written to. It is either read-only or is currently being accessed by another process.
WrongSelectorType	Occurs when calling a multi-select function with a single-select selector specified as a parameter and vice versa.
BadCulture	Culture specified not recognised as a valid culture.
UnsupportedFileType	API does not support the reading or writing of this file type.

	Currently, the unsupported types are all Visio file types and can be enabled for read/write via ADS via the Classifier configuration.
CantWriteViaHandle	Unable to write to this file when passing the handle as a parameter. Try calling SetClassification() without a handle parameter.
InvalidCollection	Array passed as parameter is null, empty or otherwise invalid.
CannotCalculateHighWaterMark	Valid high-water mark label could not be determined from the collection of labels.
LabelsAreEqual	Labels are the same.
FirstLabelDominates	Label 1 > Label 2
SecondLabelDominates	Label 1 < Label 2
LabelsAreDifferent	Labels are not the same but neither is greater than the other.
Unknown	Unexpected exception thrown. Refer to BJTrace logging to help determine the cause.