



## **FileLocator Pro Core Searching SDK**

© 2003-2013 Mythicsoft Ltd. All Rights Reserved.

# FileLocator Pro Core Searching SDK

© 2003-2013 Mythicsoft Ltd. All Rights Reserved.

All rights reserved. No parts of this work may be reproduced in any form or by any means - graphic, electronic, or mechanical, including photocopying, recording, taping, or information storage and retrieval systems - without the written permission of the publisher.

Products that are referred to in this document may be either trademarks and/or registered trademarks of the respective owners. The publisher and the author make no claim to these trademarks.

While every precaution has been taken in the preparation of this document, the publisher and the author assume no responsibility for errors or omissions, or for damages resulting from the use of information contained in this document or from the use of programs and source code that may accompany it. In no event shall the publisher and the author be liable for any loss of profit or any other commercial damage caused or alleged to have been caused directly or indirectly by this document.

November 2013

## **Publisher**

*Mythicsoft Ltd*

# Table of Contents

Foreword	0
<b>Part I FileLocator Pro Core Searching SDK</b>	<b>10</b>
1 Introduction.....	10
2 Configuration Files.....	11
3 Class Library.....	12
<b>AttributeState Enumeration</b> .....	<b>13</b>
<b>DateOffset Class</b> .....	<b>13</b>
DateOffset Members.....	14
Properties .....	14
Active Property .....	14
DateResolution Property.....	14
Value Property .....	15
<b>DateRange Class</b> .....	<b>15</b>
DateRange Members.....	16
Methods .....	16
Clear Method .....	16
Properties .....	16
After Property .....	17
Before Property .....	17
Interfaces .....	17
IDateRange2 .....	17
IDateRange2 Members.....	17
Properties .....	18
AfterDateTime Property.....	18
BeforeDateTime Property.....	18
<b>DateResolutionType Enumeration</b> .....	<b>18</b>
<b>DateTimeValue Class</b> .....	<b>19</b>
DateTimeValue Members.....	19
Properties .....	19
Active .....	19
Date .....	20
Time .....	20
Methods .....	20
IsAbsoluteDateTime.....	21
SetAbsoluteDateTime.....	21
ToDisplay .....	21
<b>DateValue Class</b> .....	<b>22</b>
DateValue Members.....	22
Properties .....	22
Absolute .....	22
Offset .....	22
RelativeDateType.....	23
<b>EmailConfiguration Class</b> .....	<b>23</b>
EmailConfiguration Members.....	23
EmailConfiguration Properties .....	24
Active .....	24
SearchAttachments.....	24

StripHTMLMarkup.....	24
<b>ExportCriteria Class .....</b>	<b>25</b>
ExportCriteria Members.....	25
Properties .....	26
ExportContents Property.....	26
ExportFormat Property.....	26
ExportSurroundingLine Property.....	27
<b>ExportFormatType Enumeration .....</b>	<b>27</b>
<b>ExpressionType Enumeration .....</b>	<b>28</b>
<b>ExtensionPlugin Class .....</b>	<b>29</b>
ExtensionPlugin Members.....	29
Properties .....	30
Active Property .....	30
DisplayName Property.....	31
ExtensionType Property.....	31
FileTypes Property.....	31
SafeMode Property.....	32
UniqueName Property.....	32
UseFilter Property.....	32
<b>ExtensionPluginList Class .....</b>	<b>33</b>
ExtensionPluginList Members.....	34
Properties .....	34
Item Property .....	34
Count Property .....	35
Methods .....	35
Find Method .....	35
<b>ExtensionPluginType Enumeration .....</b>	<b>36</b>
<b>FileAttributes Class .....</b>	<b>36</b>
FileAttributes Members.....	37
Properties .....	38
Archive Property.....	39
Compressed Property.....	39
Encrypted Property.....	39
Folder Property .....	40
Hidden Property .....	40
Offline Property .....	40
ReadOnly Property.....	41
Sparse Property .....	41
SystemFile Property.....	41
Methods .....	42
Clear Method .....	42
<b>RelativeDateType Enumeration .....</b>	<b>42</b>
<b>RelativeTimeType Enumeration .....</b>	<b>43</b>
<b>RegularExpressionType Enumeration .....</b>	<b>43</b>
<b>ScriptingCriteria Class .....</b>	<b>44</b>
ScriptingCriteria Members.....	45
Properties .....	45
Active Property .....	45
CustomParameter Property.....	46
FileName Property.....	46
ScriptEngineProgId Property.....	46
<b>SearchConfiguration Class .....</b>	<b>46</b>
SearchConfiguration Members.....	47
Properties .....	48

EOLMac Property.....	48
EOLUnix Property.....	49
ExtensionPluginList Property.....	49
SearchOnePhase Property.....	49
SearchThreadCount Property.....	50
SevenBitChars Property.....	50
SurroundingLinesAfter Property.....	50
SurroundingLinesBefore Property.....	51
Interfaces .....	51
ISearchConfiguration2.....	51
ISearchConfiguration2 Members.....	51
Properties .....	52
IFilterTypes .....	52
LinesPerFile .....	52
ISearchConfiguration3.....	52
ISearchConfiguration3 Members.....	52
Properties .....	53
ExcludeBinaryFiles.....	53
IncludeFileNameInContent.....	54
SearchEmail .....	54
UseCache .....	54
UTF8DefaultTypes.....	54
Methods .....	55
ResyncFilters .....	55
<b>SearchCriteria Class .....</b>	<b>55</b>
SearchCriteria Members.....	56
Methods .....	57
Load Method .....	57
LoadAll Method .....	58
Save Method .....	59
Properties .....	59
ContainingText Property.....	60
ContainingTextScript Property.....	60
ContentsExprSpanFile Property.....	61
ContentsExprType Property.....	61
ExcludeFilename Property.....	61
FileAttributes Property.....	62
FileName Property.....	62
FileNameExprType Property.....	62
FileNameScript Property.....	63
FileSize Property.....	63
LookIn Property .....	63
LookInExprType Property.....	63
MatchContentsCase Property.....	64
MatchFilenameCase Property.....	64
ModifiedDate Property.....	64
RegExprType Property.....	65
SearchSubDirectory Property.....	65
<b>SearchEngine Class .....</b>	<b>65</b>
SearchEngine Members.....	66
Events .....	67
OnFileFound Event.....	67
OnProgress Event.....	68
OnSearchFinish Event.....	69

OnSearchStart Event.....	69
Methods .....	70
Cancel Method .....	70
Start Method .....	70
Properties .....	71
SearchConfiguration Property.....	71
SearchCriteria Property .....	72
Interfaces .....	72
ISearchEngine2 .....	72
ISearchEngine2 Members.....	72
Methods .....	73
New SearchResultItemList.....	73
<b>SearchResultItem Class .....</b>	<b>73</b>
SearchResultItem Members.....	74
Properties .....	75
FileName Property.....	75
IsFolder Property.....	75
ModifiedDate Property.....	76
Path Property .....	76
Size Property .....	76
TextLineList Property.....	76
Interfaces .....	77
ISearchResultItem2.....	77
ISearchResultItem2 Members.....	77
Methods .....	77
CopyFileTo .....	77
ExtractText .....	78
ISearchResultItem3.....	78
ISearchResultItem3 Members.....	78
Properties .....	78
Id .....	79
<b>SearchResultItemList Class .....</b>	<b>79</b>
SearchResultItemList Members.....	80
Properties .....	81
Count Property .....	81
Item Property .....	81
ExportCriteria Property.....	82
TotalFileSize Property.....	82
Methods .....	82
ExportToFile Method.....	82
Export Method .....	83
Interfaces .....	84
ISearchResultItemList2.....	84
ISearchResultItemList2 Members.....	84
Methods .....	84
AddItem .....	85
CreateItemFromXML.....	85
LoadFromFile .....	85
ISearchResultItemList3.....	86
ISearchResultItemList3 Members.....	86
Methods .....	86
CreateItemFromId.....	86
CreateItemFromPath.....	87
<b>SizeRange Class .....</b>	<b>87</b>

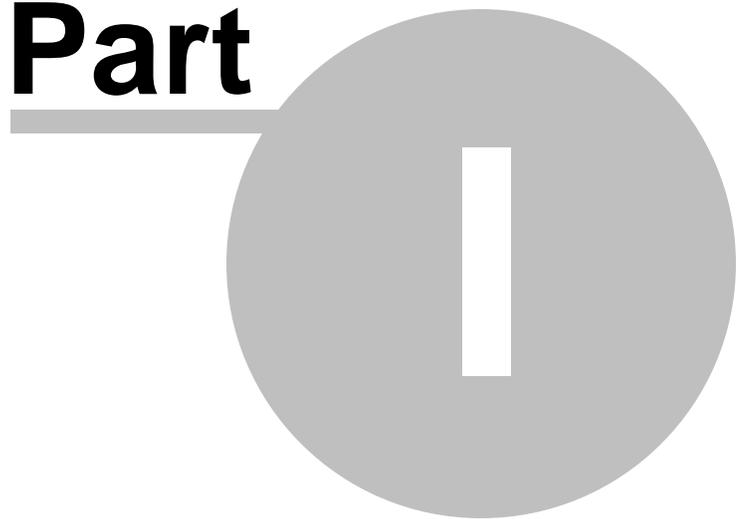
SizeRange Members.....	88
Methods .....	88
Clear Method .....	88
Properties .....	89
GreaterThan Property.....	89
LessThan Property.....	89
<b>TextLine Class .....</b>	<b>89</b>
TextLine Members.....	91
Properties .....	91
LineNumber Property.....	91
LinesAfter Property.....	92
LinesBefore Property.....	92
Text Property .....	92
TextMatchHighlightList Property.....	93
TextMatchList Property.....	93
<b>TextLineList Class .....</b>	<b>93</b>
TextLineList Members.....	94
Properties .....	95
Item Property .....	95
Count Property .....	95
<b>TextMatch Class .....</b>	<b>96</b>
TextMatch Members.....	97
Properties .....	97
Length Property .....	97
Start Property .....	98
<b>TextMatchList Class .....</b>	<b>98</b>
TextMatchList Members.....	99
Properties .....	99
Count Property .....	99
Item Property .....	100
<b>TimeOffset Class .....</b>	<b>100</b>
TimeOffset Members.....	100
Properties.....	101
Active Property.....	101
TimeResolution Property.....	101
Value Property .....	101
<b>TimeValue Class .....</b>	<b>102</b>
TimeValue Members.....	102
Properties.....	102
Absolute .....	102
Active .....	103
Offset .....	103
RelativeTimeType.....	103
<b>TimeResolutionType Enumeration .....</b>	<b>103</b>
<b>4 Extension Interfaces.....</b>	<b>104</b>
<b>Extension Configuration File .....</b>	<b>104</b>
<b>IExtCompositeInterpreter .....</b>	<b>105</b>
Close .....	106
ExtractFile.....	106
GetNextFileInfo.....	107
Open .....	108
SetFilePos.....	109
<b>IExtCompositeInterpreter2 .....</b>	<b>109</b>
SetContainerFileName.....	109

<b>IExtInitialize</b> .....	<b>110</b>
Initialize .....	110
Uninitialize.....	111
<b>IExtTextConverter</b> .....	<b>111</b>
ConvertFileToText.....	111

<b>Index</b>	<b>113</b>
--------------	------------

# FileLocator Pro Core Searching SDK

**Part**



# 1 FileLocator Pro Core Searching SDK

The FileLocator Pro Core Searching library is the foundation for file searching in FileLocator Pro. This documentation describes the class library, available for embedding the core search engine functionality into a Windows based application, and the extension interfaces, used for supplying custom file type processing.

See the Introduction for a simple example.

Please contact Mythicsoft technical support with any questions regarding this documentation:

On the web  
Technical Support ([www.mythicsoft.com](http://www.mythicsoft.com))

Or by email  
[techsupport@mythicsoft.com](mailto:techsupport@mythicsoft.com)

## 1.1 Introduction

The FileLocator Pro Core Search library is a COM based class library to simplify the embedding of the FileLocator Pro search engine functionality into any Windows based application.

The core classes are implemented in FLProCore.dll, which should have been registered during FileLocator Pro installation. To use the classes you will need to reference the DLL to import the type library (e.g. in Visual Studio simply add FLProCore to your References, or with C++ use #import).

The core class is the FLProCoreLib.SearchEngine class.

The search engine can be run in either an asynchronous or synchronous mode. In synchronous mode the calling thread is blocked until the search is completed. In asynchronous mode a new thread is created to run the search and progress information is posted back to the caller through the use of events, while the calling thread is freed to perform other tasks.

The following Visual Basic example shows the search engine running in synchronous, i.e. blocking, mode.

Note: The examples in this documentation often assume that the FLProCoreLib namespace has been imported into the global namespace, e.g.

VB : Imports FLProCoreLib

C# : using FLProCoreLib;

```
engineSearch = New SearchEngineClass

engineSearch.SearchCriteria.FileName           = "*.txt"
engineSearch.SearchCriteria.ContainingText    = "search"
engineSearch.SearchCriteria.LookIn            = "c:\search folder"

Dim listResult As SearchResultItemList = engineSearch.Start( False )
```

```
' Output the files, with their text lines and hits to the console

Dim buildText As New System.Text.StringBuilder

For nItem As Integer = 0 To (listResult.Count - 1)
    ' Each item in the list includes information about the found
    ' item (e.g. its name and path) and also the list of text lines
    ' found if this was a content search

    Dim item As SearchResultItem = listResult(nItem)

    buildText.Append(item.Path)
    buildText.Append(item.FileName)
    buildText.Append(vbCrLf)

    Dim listText As TextLineList = item.TextLineList

    For nText As Integer = 0 To (listText.Count - 1)
        ' Each text line includes the line number and text found and
also
        ' a list showing where the expression matches occurred.

        Dim line As TextLine = listText(nText)

        Dim listMatch As FLProCoreLib.TextMatchList = line.TextMatchList
        For nHit As Integer = 0 To (listMatch.Count - 1)
            buildText.AppendFormat(" {0}:{1} ", listMatch(nHit).
Start, listMatch(nHit).Length)
                Next
            buildText.Append(vbCrLf)
        Next
    Next

    System.Console.Write( buildText.ToString() )
```

The example shows the core steps for running a search:

- Creating a SearchEngine class
- Populating the criteria for the search
- Running the search (in blocking mode for this example)
- Reading the list of found SearchResultItems
- Reading the list of TextLines for each result item
- Reading the list of TextMatches on each line
- Outputting the results to the console.

## 1.2 Configuration Files

FileLocatorPro Core Searching library stores all search engine configuration information in XML files instead of using the Windows Registry. The configuration files are compatible with the retail version of FileLocator Pro.

All non-element node data is base64 encoded.

The configuration is broken up into three files:

### **config.xml**

Stores all search engine configuration information. The default location of the config.xml file is the

installation folder of the search engine but can be changed through the master configuration.

#### **regkey.xml**

Stores all registration and trial installation information for license compliance purposes. The location of the regkey.xml file is the installation folder of the search engine.

#### **master.xml**

Stores the location of the config.xml file and the log files folder.

## 1.3 Class Library

The FileLocator Pro Core Searching class library is the foundation for file searching in FileLocator Pro. The library classes control the core elements of file searching and are centered around the SearchEngine class, which is the only directly creatable class in the library.

The class library contains the following classes:

DateRange	Date range used by SearchCriteria.
ExportCriteria	Criteria used for specifying which data should be exported during an Export.
ExtensionPlugIn	Registered plug-in extension.
ExtensionPlugInList	List of registered plug-in extensions.
FileAttributes	Attribute criteria used for file searching.
ScriptingCriteria	Criteria used for extending the search engine matching algorithms with ActiveScripting scripts.
SearchConfiguration	Configuration information used to control an instance of SearchEngine.
SearchEngine	Core searching component.
SearchResultItem	Item found during a search.
SearchResultItemList	List of SearchResultItem items.
SizeRange	Size range used by SearchCriteria.
TextLine	Line of found or surrounding text.
TextLineList	List of TextLine items.
TextMatch	Positive match of the containing text expression on a given TextLine.
TextMatchList	List of matches on a given TextLine.

### 1.3.1 AttributeState Enumeration

Provides the fields that represent the search state of attributes.

[Visual Basic]

```
Public Enum AttributeState As Integer
```

#### Remarks

AttributeState enumeration is used by the FileAttributes class to indicate the search criteria for a given attribute.

#### Members

Either	Attribute can be either On or Off.
Off	Attribute must be Off.
On	Attribute must be On.

#### Example

[Visual Basic]

```
engineSearch = New SearchEngineClass

engineSearch.SearchCriteria.FileName           = "*.txt"
engineSearch.SearchCriteria.ContainingText    = "search"
engineSearch.SearchCriteria.LookIn           = "c:\search folder"

' To specify that no ReadOnly files should be found turn the attribute off.

engineSearch.SearchCriteria.FileAttributes.Clear()
engineSearch.SearchCriteria.FileAttributes.ReadOnly = AttributeState.Off

Dim listResult As SearchResultItemList = engineSearch.Start( False )
```

### 1.3.2 DateOffset Class

For a list of all members of this type, see DateOffset Members

#### Description

Represents an offset from a particular DateValue.

### 1.3.2.1 DateOffset Members

#### Public Properties

Active	Flag indicating whether or not the offset should be applied.
DateResolution	Units that the offset is specified in.
Value	Value for the offset, units are specified by DateResolution.

### 1.3.2.2 Properties

#### Public Properties

Active	Flag indicating whether or not the offset should be applied.
DateResolution	Units that the offset is specified in.
Value	Value for the offset, units are specified by DateResolution.

#### 1.3.2.2.1 Active Property

Flag indicating whether or not the offset should be applied.

[Visual Basic]

```
Public Property Active As Boolean
```

#### Property Value

True if the offset should be applied to the DateValue, otherwise False.

#### Remarks

#### 1.3.2.2.2 DateResolution Property

Units that the offset is specified in.

[Visual Basic]

```
Public Property DateResolution As DateResolutionType
```

#### Property Value

Units that the offset is specified in.

**Remarks**

## 1.3.2.2.3 Value Property

Value for the offset, units are specified by DateResolution Property.

[Visual Basic]

```
Public Property Value As Integer
```

**Property Value**

Value for the offset.

**Remarks****1.3.3 DateRange Class**

For a list of all members of this type, see DateRange Members.

**Description**

Represents a date range used by SearchCriteria.

**Thread Safety**

This class is STA based and therefore safe for multithreaded access as long as the creating process continues to process its message loop.

**Remarks**

To stop searching by date the date range needs to be cleared using the Clear method.

**Example**

[Visual Basic]

```
engineSearch = New SearchEngineClass

' Look for all files modified in the last week

engineSearch.SearchCriteria.ModifiedDate.After = DateTime.Now.AddDays(-7)
engineSearch.SearchCriteria.LookIn           = "c:\search folder"

Dim listResult As SearchResultItemList = engineSearch.Start( False )
```

### 1.3.3.1 DateRange Members

#### Public Properties

After	Beginning of the date range.
Before	End of the date range.

#### Public Methods

Clear	Clears the date range and marks it as not to be used during search.
-------	---

These interfaces provide additional functionality to the base DateRange Class:

IDateRange2

### 1.3.3.2 Methods

#### Public Methods

Clear	Clears the date range and marks it as not to be used during search.
-------	---

#### 1.3.3.2.1 Clear Method

Clears the date range and marks it as not to be used during search.

[Visual Basic]

```
Public Sub Clear()
```

#### Remarks

### 1.3.3.3 Properties

#### Public Properties

After	Beginning of the date range.
Before	End of the date range.

#### 1.3.3.3.1 After Property

Beginning of the date range.

[Visual Basic]

```
Public Property After As DateTime
```

#### Property Value

Beginning of the date range.

#### Remarks

#### 1.3.3.3.2 Before Property

End of the date range.

[Visual Basic]

```
Public Property Before As DateTime
```

#### Property Value

End of the date range.

#### Remarks

### 1.3.3.4 Interfaces

These interfaces provide additional functionality to the base DateRange Class:

IDateRange2

#### 1.3.3.4.1 IDateRange2

For a list of all members of this type, see IDateRange2 Members.

#### Description

Adds the ability to specify relative date/times in a date range.

#### 1.3.3.4.1.1 IDateRange2 Members

#### Public Properties

AfterDateTime	Beginning of the date range.
---------------	------------------------------

BeforeDateTime	End of the date range.
----------------	------------------------

## 1.3.3.4.1.2 Properties

**Public Properties**

AfterDateTime	Beginning of the date range.
BeforeDateTime	End of the date range.

Beginning of the date range.

[Visual Basic]

```
Public ReadOnly Property AfterDateTime As DateTimeValue
```

**Property Value**

Beginning of the date range.

**Remarks**

End of the date range.

[Visual Basic]

```
Public ReadOnly Property BeforeDateTime As DateTimeValue
```

**Property Value**

End of the date range.

**Remarks****1.3.4 DateResolutionType Enumeration**

Possible units for date parts.

[Visual Basic]

```
Public Enum RelativeDateType As Integer
```

**Remarks****Members**

Days	
Weeks	
Months	
Years	

### 1.3.5 DateTimeValue Class

For a list of all members of this type, see DateTimeValue Members.

#### Description

Represents a date/time value that can be relative or fixed.

#### 1.3.5.1 DateTimeValue Members

##### Public Properties

Active	Flag indicating whether or not the DateTime value is active.
Date	Retrieves the Date part of the DateTime.
Time	Retrieves the Time part of the DateTime

##### Public Methods

IsAbsoluteDateTime	Returns whether or not this is a fixed (ie. absolute) DateTime value.
SetAbsoluteDateTime	Sets the DateTime to be a fixed/absolute date time.
ToDisplay	User friendly representation of the DateTime value.

#### 1.3.5.2 Properties

##### Public Properties

Active	Flag indicating whether or not the DateTime value is active.
Date	Retrieves the Date part of the DateTime.
Time	Retrieves the Time part of the DateTime

##### 1.3.5.2.1 Active

Flag indicating whether or not the DateTime value is active.

[Visual Basic]

```
Public Property Active As Boolean
```

### Property Value

True if the value is active (ie. it should be included in the search), otherwise False.

### Remarks

#### 1.3.5.2.2 Date

Retrieves the Date part of the DateTime.

```
[Visual Basic]
```

```
Public ReadOnly Property Date As DateValue
```

### Property Value

The Date part of the DateTime.

### Remarks

#### 1.3.5.2.3 Time

Retrieves the Time part of the DateTime

```
[Visual Basic]
```

```
Public ReadOnly Property Time As TimeValue
```

### Property Value

The Time part of the DateTime value.

### Remarks

#### 1.3.5.3 Methods

##### Public Methods

IsAbsoluteDateTime	Returns whether or not this is a fixed (ie. absolute) DateTime value.
SetAbsoluteDateTime	Sets the DateTime to be a fixed/absolute date time.
ToDisplay	User friendly representation of the DateTime value.

#### 1.3.5.3.1 IsAbsoluteDateTime

Returns whether or not this is a fixed (ie. absolute) DateTime value.

[Visual Basic]

```
Public Function IsAbsoluteDateTime() as Boolean
```

#### Return Value

Returns whether or not this is a fixed (ie. absolute) DateTime value; or False if this is a relative DateTimeValue.

#### Remarks

#### 1.3.5.3.2 SetAbsoluteDateTime

Sets the DateTime to be a fixed/absolute date time.

[Visual Basic]

```
Public Sub SetAbsoluteDateTime(ByVal dtAbsoluteDateTime As DateTime)
```

#### Parameters

*dtAbsoluteDateTime*

Fixed DateTime to set this DateTimeValue to be.

#### Remarks

#### 1.3.5.3.3 ToDisplay

User friendly representation of the DateTime value.

[Visual Basic]

```
Public Function ToDisplay() as String
```

#### Return Value

User friendly representation of the DateTime value.

#### Remarks

### 1.3.6 DateValue Class

For a list of all members of this type, see DateValue Members.

#### Description

Represents a date value used in DateTimeValue Class.

#### 1.3.6.1 DateValue Members

##### Public Properties

Absolute	Specific date time value.
Offset	Offset for the date value.
RelativeDate	Relative date type for this date value.

#### 1.3.6.2 Properties

##### Public Properties

Absolute	Specific date time value.
Offset	Offset for the date value.
RelativeDate	Relative date type for this date value.

##### 1.3.6.2.1 Absolute

Specific date time value.

[Visual Basic]

```
Public Property Absolute As DateTime
```

##### Property Value

A specific/fixed (as opposed to relative) date time value.

##### Remarks

##### 1.3.6.2.2 Offset

Offset for the date value.

[Visual Basic]

```
Public ReadOnly Property Offset As DateTimeOffset
```

**Property Value**

Adjustment to the date value, e.g. +1 day.

**Remarks**

## 1.3.6.2.3 RelativeDateType

Relative date type for this date value.

[Visual Basic]

```
Public Property RelativeDateType As RelativeDateType
```

**Property Value**

Relative date type for this date value, e.g. Today.

**Remarks****1.3.7 EmailConfiguration Class**

For a list of all members of this type, see EmailConfiguration Members

**Description**

Represents the configuration information used to control PST/MSG searching.

**Thread Safety**

This class is STA based and therefore safe for multithreaded access as long as the creating process continues to process its message loop.

**Remarks**

This class is not created directly but referenced from an instance of SearchConfiguration Class.

**1.3.7.1 EmailConfiguration Members****Public Properties**

Active	Flag indicating whether or not PST/MSG searching is switched ON
SearchAttachments	Flag indicating whether or not attachments are also searched
StripHTMLMarkup	Flag indicating if HTML markup should be stripped from the body of HTML based messages before searching

### 1.3.7.2 EmailConfiguration Properties

#### Public Properties

Active	Flag indicating whether or not PST/MSG searching is switched ON
SearchAttachments	Flag indicating whether or not attachments are also searched
StripHTMLMarkup	Flag indicating if HTML markup should be stripped from the body of HTML based messages before searching

#### 1.3.7.2.1 Active

Flag indicating whether or not PST/MSG searching is switched ON

[Visual Basic]

```
Public Property Active As Boolean
```

#### Property Value

True if the PST/MSG files are searched, otherwise False.

#### Remarks

#### 1.3.7.2.2 SearchAttachments

Flag indicating whether or not attachments are also searched

[Visual Basic]

```
Public Property SearchAttachments As Boolean
```

#### Property Value

True if email attachments are searched, otherwise False.

#### Remarks

#### 1.3.7.2.3 StripHTMLMarkup

Flag indicating if HTML markup should be stripped from the body of HTML based messages before searching

[Visual Basic]

```
Public Property StripHTMLMarkup As Boolean
```

### Property Value

True if HTML based messages should be stripped of HTML markup, otherwise False.

### Remarks

## 1.3.8 ExportCriteria Class

For a list of all members of this type, see [ExportCriteria Members](#).

### Description

Represents criteria used for specifying which data should be exported during an Export.

### Thread Safety

This class is STA based and therefore safe for multithreaded access as long as the creating process continues to process its message loop.

### Remarks

### Example

[Visual Basic]

```
engineSearch = New SearchEngineClass

engineSearch.SearchCriteria.FileName          = "*.txt"
engineSearch.SearchCriteria.ContainingText   = "search"
engineSearch.SearchCriteria.LookIn          = "c:\search folder"

Dim listResult As SearchResultItemList = engineSearch.Start( False )

' Export the results to a file

listResult.ExportCriteria.ExportContents      = True
listResult.ExportCriteria.ExportSurroundingLines = False
listResult.ExportCriteria.ExportFormat       = ExportFormatType.HTML

listResult.ExportToFile( "C:\Results.html" )
```

### 1.3.8.1 ExportCriteria Members

[ExportCriteria overview](#)

### Public Properties

ExportContents	Flag indicating whether or not the found contents should be exported.
ExportFormat	Format for the exported data.
ExportSurroundingLines	Flag indicating whether or not the surrounding lines of found content should be exported.

### 1.3.8.2 Properties

#### Public Properties

ExportContents	Flag indicating whether or not the found contents should be exported.
ExportFormat	Format for the exported data.
ExportSurroundingLines	Flag indicating whether or not the surrounding lines of found content should be exported.

#### 1.3.8.2.1 ExportContents Property

Flag indicating whether or not the found contents should be exported.

[Visual Basic]

```
Public Property ExportContents As Boolean
```

#### Property Value

True if the found text should be included in the export, otherwise False.

#### Remarks

#### 1.3.8.2.2 ExportFormat Property

Format for the exported data.

[Visual Basic]

```
public Property ExportFormat As ExportFormatType
```

#### Property Value

Format for the exported data.

### Remarks

See ExportFormatType enumeration for a list of possible values.

#### 1.3.8.2.3 ExportSurroundingLine Property

Flag indicating whether or not the surrounding lines of found content should be exported.

[Visual Basic]

```
Public Property ExportSurroundingLines As Boolean
```

### Property Value

True if the text surrounding the found text should be included in the export, otherwise False.

### Remarks

## 1.3.9 ExportFormatType Enumeration

Provides the fields that represent the format type for exporting.

[Visual Basic]

```
Public Enum ExportFormatType As Integer
```

### Remarks

ExportFormatType enumeration is used by the ExportCriteria class.

### Members

CommaSeparated	Fields are comma separated (CSV)
HTML	Result is formatted in HTML
TabSeparated	Fields are tab separated
Text	Basic text output
XML	Fields are formatted in XML

### Example

[Visual Basic]

```
engineSearch = New SearchEngineClass
engineSearch.SearchCriteria.FileName = "*.txt"
engineSearch.SearchCriteria.ContainingText = "search"
```

```

engineSearch.SearchCriteria.LookIn           = "c:\search folder"

Dim listResult As SearchResultItemList = engineSearch.Start( False )

' Export the results to a file

listResult.ExportCriteria.ExportContents      = True
listResult.ExportCriteria.ExportSurroundingLines = False
listResult.ExportCriteria.ExportFormat       = ExportFormatType.HTML

listResult.ExportToFile( "C:\Results.html" )

```

### 1.3.10 ExpressionType Enumeration

Provides the fields that represent the expression type for string matching.

[Visual Basic]

```
Public Enum ExpressionType As Integer
```

#### Remarks

ExpressionType enumeration is used by the SearchCriteria class.

#### Members

Boolean	Boolean, e.g. AND, OR, NOT etc., expression.
DosExp	DOS style expression most often used for file matching, e.g. *.txt
Exact	Literal expression that must be matched exactly.
RegExp	Regular expression.

#### Example

[Visual Basic]

```

engineSearch = New SearchEngineClass

engineSearch.SearchCriteria.FileName          = "*.txt"
engineSearch.SearchCriteria.ContainingText   = "search"
engineSearch.SearchCriteria.LookIn          = "c:\search folder"

' switch the expression engine to Boolean

engineSearch.SearchCriteria.ContentsExprType = ExpressionType.Boolean

Dim listResult As SearchResultItemList = engineSearch.Start( False )

```

### 1.3.11 ExtensionPlugIn Class

For a list of all members of this type, see ExtensionPlugIn Members.

#### Description

Represents a registered plug-in extension.

#### Thread Safety

This class is STA based and therefore safe for multithreaded access as long as the creating process continues to process its message loop.

#### Remarks

Extensions are used to allow the search engine to more intelligently understand file types. The three main types of extensions are listed in the ExtensionPlugInType Enumeration.

The extension information is initially loaded from the plugin\_cfg sub-folder and modified with any configuration information stored in config.xml.

#### Example

[Visual Basic]

```
engineSearch = New SearchEngineClass

engineSearch.SearchCriteria.FileName           = "*.pdf"
engineSearch.SearchCriteria.ContainingText    = "search"
engineSearch.SearchCriteria.LookIn           = "c:\search folder"

' Switch on the pdf converter to extract the text from PDF files.

engineSearch.SearchConfiguration.ExtensionPlugInList.Find("http://www.
mythicsoft.com/pdfconverter").Active = True

Dim listResult As SearchResultItemList = engineSearch.Start( False )
```

#### 1.3.11.1 ExtensionPlugIn Members

ExtensionPlugIn overview

##### Public Properties

Active	Flag indicating whether or not the extension should be used.
DisplayName	Display name of the extension.
ExtensionType	Type of extension.

FileTypes	Comma separated list of file types the extension should process.
SafeMode	Flag indicating if the extension should be run in a separate process from the search process.
UniqueName	Globally unique name of the extension.
UseFilter	Flag indicating whether or not the search engine should use a registered IFilter instead of the extension (if one is available).

### 1.3.11.2 Properties

#### Public Properties

Active	Flag indicating whether or not the extension should be used.
DisplayName	Display name of the extension.
ExtensionType	Type of extension.
FileTypes	Comma separated list of file types the extension should process.
SafeMode	Flag indicating if the extension should be run in a separate process from the search process.
UniqueName	Globally unique name of the extension.
UseFilter	Flag indicating whether or not the search engine should use a registered IFilter instead of the extension (if one is available).

#### 1.3.11.2.1 Active Property

Flag indicating whether or not the extension should be used.

[Visual Basic]

```
Public Property Active As Boolean
```

#### Property Value

True if the extension should be used, otherwise False.

#### Remarks

#### 1.3.11.2.2 DisplayName Property

Display name of the extension.

[Visual Basic]

```
Public ReadOnly Property DisplayName As String
```

##### Property Value

Display name of the extension.

##### Remarks

This information is always loaded from the plugin\_cfg XML file and cannot be altered.

#### 1.3.11.2.3 ExtensionType Property

Type of extension.

[Visual Basic]

```
Public ReadOnly Property ExtensionType As ExtensionPlugInType
```

##### Property Value

Type of extension.

##### Remarks

This information is always loaded from the plugin\_cfg XML file and cannot be altered.

#### 1.3.11.2.4 FileTypes Property

Comma separated list of file types the extension should process.

[Visual Basic]

```
Public Property FileTypes As String
```

##### Property Value

Comma separated list of file types the extension should process.

##### Remarks

See `ExtensionPlugInType` enumeration for a list of valid values.

#### 1.3.11.2.5 SafeMode Property

Flag indicating if the extension should be run in a separate process from the search process.

[Visual Basic]

```
Public Property SafeMode As Boolean
```

#### Property Value

True if the extension should be run in a separate process from the search process, otherwise False if the extension should be run in-process.

#### Remarks

An extension is a COM object that is normally loaded in-process. If the extension is flagged to run in `SafeMode` the engine will try to load the extension with the `CLSCTX_LOCAL_SERVER` flag specified instead of `CLSCTX_INPROC_SERVER`, which will cause the extension to run in its own process space.

Running an extension in `SafeMode` will prevent a badly behaved extension from crashing the search engine's process space but since the extension will be running in a separate process method calls to the extension will be slower and therefore searching with the extension will be slower.

#### 1.3.11.2.6 UniqueName Property

Globally unique name of the extension.

[Visual Basic]

```
Public ReadOnly Property UniqueName As String
```

#### Property Value

Globally unique name of the extension.

#### Remarks

This information is always loaded from the `plugin_cfg` XML file and cannot be altered.

#### 1.3.11.2.7 UseIFilter Property

Flag indicating whether or not the search engine should use a registered `IFilter` instead of the extension (if one is available).

[Visual Basic]

```
Public Property UseIFilter As Boolean
```

### Property Value

True if the search engine should use a registered IFilter instead of the extension (if one is available), otherwise False.

### Remarks

Microsoft Indexing Service uses filters, much like FileLocator Pro extensions, to interpret given file types into easily searchable text. Using this flag the search engine can be configured to use the Indexing Service Filter instead of the configured extension. If a filter is not found, or cannot be loaded, the search engine will attempt to use the configured extension instead.

## 1.3.12 ExtensionPluginList Class

For a list of all members of this type, see ExtensionPluginList Members.

### Description

Represents the list of registered plug-in extensions.

### Thread Safety

This class is STA based and therefore safe for multithreaded access as long as the creating process continues to process its message loop.

### Remarks

The extension information is initially loaded from the plugin\_cfg sub-folder and modified with any configuration information stored in config.xml.

The contents of the list can be accessed through either a numeric index, using the Item property, or through the Find method.

This class is not created directly but referenced from an instance of SearchConfiguration.

### Example

[Visual Basic]

```
engineSearch = New SearchEngineClass

engineSearch.SearchCriteria.FileName           = "*.pdf"
engineSearch.SearchCriteria.ContainingText    = "search"
engineSearch.SearchCriteria.LookIn           = "c:\search folder"

' Switch on the pdf converter to extract the text from PDF files.

engineSearch.SearchConfiguration.ExtensionPluginList.Find("http://www.
mythicsoft.com/pdfconverter").Active = True

Dim listResult As SearchResultItemList = engineSearch.Start( False )
```

### 1.3.12.1 ExtensionPlugInList Members

ExtensionPlugInList overview

#### Public Properties

Count	Total number of extensions in the list.
Item	Integer indexed (zero based index) list of ExtensionPlugIn items.

#### Public Methods

Find	Finds an extension based on the extensions unique name.
------	---

### 1.3.12.2 Properties

#### Public Properties

Count	Total number of extensions in the list.
Item	Integer indexed (zero based index) list of ExtensionPlugIn items.

#### 1.3.12.2.1 Item Property

Integer indexed (zero based index) list of ExtensionPlugIn items.

[Visual Basic]

```
Public ReadOnly Item(ByVal nIndex As Integer) As ExtensionPlugIn
```

#### Parameters

*nIndex*

The zero-based index of the extension

#### Property Value

A `ExtensionPlugIn` item that contains the extension information.

#### Remarks

#### 1.3.12.2.2 Count Property

Total number of extensions in the list.

[Visual Basic]

```
Public ReadOnly Property Count As Integer
```

#### Property Value

Total number of extensions in the list.

#### Remarks

#### 1.3.12.3 Methods

##### Public Methods

Find	Finds an extension based on the extensions unique name.
------	---

#### 1.3.12.3.1 Find Method

Finds an extension based on the extensions unique name.

[Visual Basic]

```
Public Function Find(ByVal strUniqueName As String) As ExtensionPlugIn
```

#### Parameters

*strUniqueName*

Unique name of the extension, as defined in the `plugin_cfg` XML file.

#### Return Value

An instance of `ExtensionPlugIn` if found. Otherwise the method will return `Nothing`.

#### Remarks

`Find` is case-sensitive so the unique name must be specified exactly as it is configured in the `plugin_cfg` XML file.

**Example**

[Visual Basic]

```

engineSearch = New SearchEngineClass

engineSearch.SearchCriteria.FileName           = "*.pdf"
engineSearch.SearchCriteria.ContainingText    = "search"
engineSearch.SearchCriteria.LookIn           = "c:\search folder"

' Switch on the pdf converter to extract the text from PDF files.

engineSearch.SearchConfiguration.ExtensionPlugInList.Find("http://www.
mythicsoft.com/pdfconverter").Active = True

Dim listResult As SearchResultItemList = engineSearch.Start( False )

```

**1.3.13 ExtensionPlugInType Enumeration**

Provides the fields that represent the type of a registered extension.

[Visual Basic]

```
Public Enum ExtensionPlugInType As Integer
```

**Remarks**

ExtensionPlugInType enumeration is used by the ExtensionPlugIn class.

**Members**

CompositeFile	Extension is a composite file, i.e. a file that contains one or more files within it, e.g. ZIP file, TAR file.
TextConverter	Extension converts a file to a temporary text file for easy searching.
TextInterpreter	Extension provides a line by line conversion of a file in text format.
Unknown	Extension is not recognized.

**1.3.14 FileAttributes Class**

For a list of all members of this type, see FileAttributes Members.

**Description**

Represents attribute criteria used for file searching.

**Thread Safety**

This class is STA based and therefore safe for multithreaded access as long as the creating process continues to process its message loop.

### Remarks

This class controls search criteria regarding the attributes of files. The value of each attribute criteria is defined by the AttributeState Enumeration.

If no attribute searching is required then the Clear method should be used to clear all attributes (the default state).

### Example

[Visual Basic]

```
engineSearch = New SearchEngineClass

engineSearch.SearchCriteria.FileName           = "*.txt"
engineSearch.SearchCriteria.ContainingText    = "search"
engineSearch.SearchCriteria.LookIn           = "c:\search folder"

' To specify that no ReadOnly files should be found turn the attribute off.

engineSearch.SearchCriteria.FileAttributes.Clear()
engineSearch.SearchCriteria.FileAttributes.ReadOnly = AttributeState.Off

Dim listResult As SearchResultItemList = engineSearch.Start( False )
```

#### 1.3.14.1 FileAttributes Members

FileAttributes overview

#### Public Properties

Archive	Value indicating whether or not files with the archive bit set should be searched.
Compressed	Value indicating whether or not files which are compressed should be searched.
Encrypted	Value indicating whether or not files which are encrypted should be searched.
Folder	Value indicating whether or not only folders should be returned.
Hidden	Value indicating whether or not files which are hidden should be searched.
Offline	Value indicating whether or not files which are offline should be searched.
ReadOnly	Value indicating whether or not files which are

	read only should be searched.
Sparse	Value indicating whether or not files which are sparse files should be searched.
SystemFile	Value indicating whether or not files which are system files should be searched.

### Public Methods

Clear	Clears all attributes so that no attribute matching will be used during search.
-------	---

## 1.3.14.2 Properties

### Public Properties

Archive	Value indicating whether or not files with the archive bit set should be searched.
Compressed	Value indicating whether or not files which are compressed should be searched.
Encrypted	Value indicating whether or not files which are encrypted should be searched.
Folder	Value indicating whether or not only folders should be returned.
Hidden	Value indicating whether or not files which are hidden should be searched.
Offline	Value indicating whether or not files which are offline should be searched.
ReadOnly	Value indicating whether or not files which are read only should be searched.
Sparse	Value indicating whether or not files which are sparse files should be searched.
SystemFile	Value indicating whether or not files which are system files should be searched.

#### 1.3.14.2.1 Archive Property

Value indicating whether or not files with the archive bit set should be searched.

[Visual Basic]

```
Public Property Archive As AttributeState
```

##### Property Value

AttributeState.On if the property should be set, AttributeState.Off if the property should not be set, or AttributeState.Either if the property should not be part of the search criteria.

##### Remarks

If this property is not required for searching it should be set to AttributeState.Either (the default value).

#### 1.3.14.2.2 Compressed Property

Value indicating whether or not files which are compressed should be searched.

[Visual Basic]

```
Public Property Compressed As AttributeState
```

##### Property Value

AttributeState.On if the property should be set, AttributeState.Off if the property should not be set, or AttributeState.Either if the property should not be part of the search criteria.

##### Remarks

If this property is not required for searching it should be set to AttributeState.Either (the default value).

#### 1.3.14.2.3 Encrypted Property

Value indicating whether or not files which are encrypted should be searched.

[Visual Basic]

```
Public Property Encrypted As AttributeState
```

##### Property Value

AttributeState.On if the property should be set, AttributeState.Off if the property should not be set, or AttributeState.Either if the property should not be part of the search criteria.

##### Remarks

If this property is not required for searching it should be set to AttributeState.Either (the default value).

#### 1.3.14.2.4 Folder Property

Value indicating whether or not only folders should be returned.

[Visual Basic]

```
Public Property Folder As AttributeState
```

#### Property Value

AttributeState.On if the property should be set, AttributeState.Off if the property should not be set, or AttributeState.Either if the property should not be part of the search criteria.

#### Remarks

If this property is not required for searching it should be set to AttributeState.Either (the default value).

#### 1.3.14.2.5 Hidden Property

Value indicating whether or not files which are hidden should be searched.

[Visual Basic]

```
Public Property Hidden As AttributeState
```

#### Property Value

AttributeState.On if the property should be set, AttributeState.Off if the property should not be set, or AttributeState.Either if the property should not be part of the search criteria.

#### Remarks

If this property is not required for searching it should be set to AttributeState.Either (the default value).

#### 1.3.14.2.6 Offline Property

Value indicating whether or not files which are offline should be searched.

[Visual Basic]

```
Public Property Offline As AttributeState
```

#### Property Value

AttributeState.On if the property should be set, AttributeState.Off if the property should not be set, or AttributeState.Either if the property should not be part of the search criteria.

#### Remarks

If this property is not required for searching it should be set to AttributeState.Either (the default value).

#### 1.3.14.2.7 ReadOnly Property

Value indicating whether or not files which are read only should be searched.

[Visual Basic]

```
Public Property ReadOnly As AttributeState
```

##### Property Value

AttributeState.On if the property should be set, AttributeState.Off if the property should not be set, or AttributeState.Either if the property should not be part of the search criteria.

##### Remarks

If this property is not required for searching it should be set to AttributeState.Either (the default value).

#### 1.3.14.2.8 Sparse Property

Value indicating whether or not files which are sparse files should be searched.

[Visual Basic]

```
Public Property Sparse As AttributeState
```

##### Property Value

AttributeState.On if the property should be set, AttributeState.Off if the property should not be set, or AttributeState.Either if the property should not be part of the search criteria.

##### Remarks

If this property is not required for searching it should be set to AttributeState.Either (the default value).

#### 1.3.14.2.9 SystemFile Property

Value indicating whether or not files which are system files should be searched.

[Visual Basic]

```
Public Property SystemFile As AttributeState
```

##### Property Value

AttributeState.On if the property should be set, AttributeState.Off if the property should not be set, or AttributeState.Either if the property should not be part of the search criteria.

##### Remarks

If this property is not required for searching it should be set to AttributeState.Either (the default value).

### 1.3.14.3 Methods

#### Public Methods

Clear	Clears all attributes so that no attribute matching will be used during search.
-------	---

#### 1.3.14.3.1 Clear Method

Clears all attributes so that no attribute matching will be used during search.

[Visual Basic]

```
Public Sub Clear()
```

#### Remarks

## 1.3.15 RelativeDateType Enumeration

Possible values for relative date specifications

[Visual Basic]

```
Public Enum RelativeDateType As Integer
```

#### Remarks

#### Members

AbsoluteDate	Date type is not relative, ie it's fixed.
Today	
StartOfWeek	
StartOfMonth	
StartOfYear	
EndOfWeek	
EndOfMonth	
EndOfYear	

### 1.3.16 RelativeTimeType Enumeration

Possible values for relative time specifications

[Visual Basic]

```
Public Enum RelativeTimeType As Integer
```

#### Remarks

#### Members

AbsoluteTime	Time type is not relative, ie it's fixed.
Now	
StartOfMinute	
StartOfHour	
StartOfDay	
EndOfMinute	
EndOfHour	
EndOfDay	

### 1.3.17 RegularExpressionType Enumeration

Provides the fields that represent the regular expression type for string matching.

[Visual Basic]

```
Public Enum RegularExpressionType As Integer
```

#### Remarks

RegularExpressionType enumeration is used by the SearchCriteria class.

#### Members

Boost	Perl compatible syntax regexp engine provided by the Boost library.
Classic	Basic regular expression syntax based on the regexp library by Henry Spencer. Usually faster than the Boost library engine.

#### Example

[Visual Basic]

```

engineSearch = New SearchEngineClass

engineSearch.SearchCriteria.FileName           = "*.txt"
engineSearch.SearchCriteria.ContainingText    = "search"
engineSearch.SearchCriteria.LookIn           = "c:\search folder"

' switch the expression engine to classic regular expression.

engineSearch.SearchCriteria.ContentsExprType = ExpressionType.RegExp
engineSearch.SearchCriteria.RegExprType      = RegularExpressionType.
Classic

Dim listResult As SearchResultItemList = engineSearch.Start( False )

```

### 1.3.18 ScriptingCriteria Class

For a list of all members of this type, see [ScriptingCriteria Members](#).

#### Description

Represents criteria used for extending the search engine matching algorithms with ActiveScripting scripts.

#### Thread Safety

This class is STA based and therefore safe for multithreaded access as long as the creating process continues to process its message loop.

#### Remarks

Scripting provides a simple method for extending the search engine matching algorithm. Scripts can be written using any Active Scripting engine installed on the machine running the search.

Scripts are only called if the other search criteria is satisfied, i.e. file name or containing text expressions must be successfully matched first before their respective scripts are called.

#### Example

[Visual Basic]

```

engineSearch = New SearchEngineClass

engineSearch.SearchCriteria.FileName           = "*.txt"
engineSearch.SearchCriteria.ContainingText    = "search"
engineSearch.SearchCriteria.LookIn           = "c:\search folder"

engineSearch.SearchCriteria.FileNameScript.Active           = True
engineSearch.SearchCriteria.FileNameScript.ScriptEngineProgId = "JScript"
engineSearch.SearchCriteria.FileNameScript.FileName         = "c:\
\scripts\not_regexp.js"
engineSearch.SearchCriteria.FileNameScript.CustomParameter  = "negative"

```

```
Dim listResult As SearchResultItemList = engineSearch.Start( False )
```

### 1.3.18.1 ScriptingCriteria Members

ScriptingCriteria overview

#### Public Properties

Active	Flag indicating whether or not script should be used.
CustomParameter	Custom parameter passed directly to the script.
FileName	Full path name of the script file.
ScriptEngineProgId	ProgId of the ActiveScripting engine.

### 1.3.18.2 Properties

#### Public Properties

Active	Flag indicating whether or not script should be used.
CustomParameter	Custom parameter passed directly to the script.
FileName	Full path name of the script file.
ScriptEngineProgId	ProgId of the ActiveScripting engine.

#### 1.3.18.2.1 Active Property

Flag indicating whether or not the script should be used.

[Visual Basic]

```
Public Property Active As Boolean
```

#### Property Value

True if the script should be used, otherwise False.

#### Remarks

#### 1.3.18.2.2 CustomParameter Property

Custom parameter passed directly to the script.

[Visual Basic]

```
Public Property CustomParameter As String
```

##### **Property Value**

Custom parameter passed directly to the script.

##### **Remarks**

The custom parameter can be retrieved in the script using the SearchParms object.

#### 1.3.18.2.3 FileName Property

Full path name of the script file.

[Visual Basic]

```
Public Property FileName As String
```

##### **Property Value**

Full path name of the script file.

##### **Remarks**

#### 1.3.18.2.4 ScriptEngineProgId Property

ProgId of the Active Scripting engine.

[Visual Basic]

```
Public Property ScriptEngineProgId As String
```

##### **Property Value**

ProgId of the Active Scripting engine.

##### **Remarks**

Two Active Scripting engines normally available on computers are 'JScript' and 'VBScript'.

### **1.3.19 SearchConfiguration Class**

For a list of all members of this type, see SearchConfiguration Members.

**Description**

Represents the configuration information used to control an instance of SearchEngine.

**Thread Safety**

This class is STA based and therefore safe for multithreaded access as long as the creating process continues to process its message loop.

**Remarks**

This class is not created directly but referenced from an instance of SearchEngine. The base configuration is loaded from the config.xml file, for more information see Configuration Files.

**Example**

[Visual Basic]

```
engineSearch = New SearchEngineClass

engineSearch.SearchCriteria.FileName           = "*.txt"
engineSearch.SearchCriteria.ContainingText    = "search"
engineSearch.SearchCriteria.LookIn            = "c:\search folder"

engineSearch.SearchConfiguration.SearchThreadCount = 1

Dim listResult As SearchResultItemList = engineSearch.Start( False )
```

**1.3.19.1 SearchConfiguration Members**

SearchConfiguration overview

**Public Properties**

EOLMac	Flag indicating if Mac style End Of Line characters, i.e. Carriage Return (0x0d), are used.
EOLUnix	Flag indicating if Unix style End Of Line characters, i.e. Line Feed (0x0a), are used.
ExtensionPlugInList	List of registered plug-in extensions.
SearchOnePhase	Flag indicating if the search should be carried out in a single phase rather than the default two phase approach.
SearchThreadCount	Numbers of threads that can be used to search the contents of files.
SevenBitChars	Flag indicating if only the first seven bits of single byte characters should be used when searching.

SurroundingLinesAfter	Number of lines following a found line of text that should be returned with the text line.
SurroundingLinesBefore	Number of lines preceding a found line of text that should be returned with the text line.

These interfaces provide additional functionality to the SearchConfiguration class.

ISearchConfiguration2

ISearchConfiguration3

### 1.3.19.2 Properties

#### Public Properties

EOLMac	Flag indicating if Mac style End Of Line characters, i.e. Carriage Return (0x0d), are used.
EOLUnix	Flag indicating if Unix style End Of Line characters, i.e. Line Feed (0x0a), are used.
ExtensionPlugInList	List of registered plug-in extensions.
SearchOnePhase	Flag indicating if the search should be carried out in a single phase rather than the default two phase approach.
SearchThreadCount	Numbers of threads that can be used to search the contents of files.
SevenBitChars	Flag indicating if only the first seven bits of single byte characters should be used when searching.
SurroundingLinesAfter	Number of lines following a found line of text that should be returned with the text line.
SurroundingLinesBefore	Number of lines preceding a found line of text that should be returned with the text line.

#### 1.3.19.2.1 EOLMac Property

Flag indicating if Mac style End Of Line characters, i.e. Carriage Return (0x0d), are used.

[Visual Basic]

```
Public Property EOLMac As Boolean
```

#### Property Value

True if Mac style End Of Line characters, i.e. Carriage Return (0x0d), are valid; otherwise False.

#### Remarks

#### 1.3.19.2.2 EOLUnix Property

Flag indicating if Unix style End Of Line characters, i.e. Line Feed (0x0a), are used.

[Visual Basic]

```
Public Property EOLUnix As Boolean
```

#### Property Value

True if Unix style End Of Line characters, i.e. Line Feed (0x0a), are valid; otherwise False.

#### Remarks

#### 1.3.19.2.3 ExtensionPlugInList Property

List of registered plug-in extensions.

[Visual Basic]

```
Public ReadOnly Property ExtensionPlugInList As ExtensionPlugInList
```

#### Property Value

List of registered plug-in extensions.

#### Remarks

#### 1.3.19.2.4 SearchOnePhase Property

Flag indicating if the search should be carried out in a single phase rather than the default two phase approach.

[Visual Basic]

```
Public Property SearchOnePhase As Boolean
```

#### Property Value

True if the search should be carried out in a single phase rather than the default two phase approach, otherwise False indicates that the default two phase approach is used.

#### Remarks

By default the search engine works in a two phase process. The first phase grabs a list of files that match the file name criteria. The second phase searches the contents of the files found in the first phase. If many files are found in the first phase the search engine may use a significant amount of memory resources maintaining the list of files. Therefore a one phase approach may be useful to search the contents of the file as soon as the file name is matched, avoiding the maintenance of an internal file list. The only drawback to searching in one phase is that the progress returned in OnProgress contains running totals of the total file size rather than an accurate number.

#### 1.3.19.2.5 SearchThreadCount Property

Numbers of threads that can be used to search the contents of files.

[Visual Basic]

```
Public Property SearchThreadCount As Integer
```

#### Property Value

Numbers of threads that can be used to search the contents of files.

#### Remarks

The search engine can be configured to use more than one thread for searching the contents of files. This is only advisable if the machine the search engine is running on contains more than one processor core and/or has hard drives configured for fast multiple read access (e.g. RAID arrays). If the machine does not have much memory and/or slow file access then increasing the thread count may actually reduce search performance.

#### 1.3.19.2.6 SevenBitChars Property

Flag indicating if only the first seven bits of single byte characters should be used when searching.

[Visual Basic]

```
Public Property SevenBitChars As Boolean
```

#### Property Value

True if only the first seven bits of single byte characters should be used when searching.

#### Remarks

Useful when searching old file formats which sometimes used the 8th bit for formatting purposes.

#### 1.3.19.2.7 SurroundingLinesAfter Property

Number of lines following a found line of text that should be returned with the text line.

[Visual Basic]

```
Public Property SurroundingLinesAfter As Integer
```

**Property Value**

Number of lines following a found line of text that should be returned with the text line.

**Remarks**

## 1.3.19.2.8 SurroundingLinesBefore Property

Number of lines preceding a found line of text that should be returned with the text line.

[Visual Basic]

```
Public Property SurroundingLinesBefore As Integer
```

**Property Value**

Number of lines preceding a found line of text that should be returned with the text line.

**Remarks****1.3.19.3 Interfaces**

These interfaces provide additional functionality to the SearchConfiguration class.

ISearchConfiguration2

ISearchConfiguration3

## 1.3.19.3.1 ISearchConfiguration2

For a list of all members of this type, see ISearchConfiguration2 Members.

**Description**

This interface provides additional functionality for the SearchConfiguration Class.

## 1.3.19.3.1.1 ISearchConfiguration2 Members

**Public Properties**

IFilterTypes	
LinesPerFile	

## 1.3.19.3.1.2 Properties

**Public Properties**

IFilterTypes	
LinesPerFile	

Comma separated list of file types that should be tested for the existence of IFilters.

[Visual Basic]

```
Public Property IFilterTypes As String
```

**Property Value**

Comma separated list of file types that should be attempted to be opened with IFilters.

**Remarks**

If an IFilter does not exist the file will be searched using the default search method.

Maximum number of lines that can be returned for each file.

[Visual Basic]

```
Public Property LinesPerFile As Integer
```

**Property Value**

Maximum number of lines that can be returned for each file.

**Remarks**

## 1.3.19.3.2 ISearchConfiguration3

For a list of all members of this type, see ISearchConfiguration3 Members.

**Description**

This interface provides additional functionality for the SearchConfiguration Class.

## 1.3.19.3.2.1 ISearchConfiguration3 Members

**Public Properties**

ExcludeBinaryFiles	Flag indicating if binary files should be excluded during a content search.
--------------------	---

IncludeFileNameInContent	Flag indicating if the file name should be included in the content when searching contents.
SearchEmail	Configuration for email (PST/MSG) searching
UseCache	Flag indicating if any converted text should be stored/retrieved from the cache.
UTF8DefaultTypes	Comma separated list of file types that should default to UTF-8 file type if not specifically identified.

### Public Methods

ResyncIFilters	Synchronizes the list of IFilters with those that are installed on the system.
----------------	--

#### 1.3.19.3.2.2 Properties

### Public Properties

ExcludeBinaryFiles	Flag indicating if binary files should be excluded during a content search.
IncludeFileNameInContent	Flag indicating if the file name should be included in the content when searching contents.
SearchEmail	Configuration for email (PST/MSG) searching
UseCache	Flag indicating if any converted text should be stored/retrieved from the cache.
UTF8DefaultTypes	Comma separated list of file types that should default to UTF-8 file type if not specifically identified.

Flag indicating if binary files should be excluded during a content search.

[Visual Basic]

```
Public Property ExcludeBinaryFiles As Boolean
```

### Property Value

True if binary files should be excluded during a content search; otherwise False.

**Remarks**

Does not affect non-content searches.

Flag indicating if the file name should be included in the content when searching contents.

[Visual Basic]

```
Public Property IncludeFileNameInContent As Boolean
```

**Property Value**

True if the file name should be included in the content; otherwise False.

**Remarks**

Configuration for email (PST/MSG) searching.

[Visual Basic]

```
Public ReadOnly Property SearchEmail As EmailConfiguration
```

**Property Value**

The configuration data email searching.

**Remarks**

Flag indicating if any converted text should be stored/retrieved from the cache.

[Visual Basic]

```
Public Property UseCache As Boolean
```

**Property Value**

True if the cache can be used to store/retrieve text for file types that require conversion to text (e.g. PDF, DOC etc.); otherwise False.

**Remarks**

Comma separated list of file types that should default to UTF-8 file type if not specifically identified.

[Visual Basic]

```
Public Property UTF8DefaultTypes As String
```

**Property Value**

Comma separated list of file types that should default to UTF-8 file type if not specifically identified.

**Remarks**

## 1.3.19.3.2.3 Methods

**Public Methods**

ResyncIFilters	Synchronizes the list of IFilters with those that are installed on the system.
----------------	--

Synchronizes the list of IFilters with those that are installed on the system.

[Visual Basic]

```
Public Function ResyncIFilters(ByVal bFullResync As Boolean) as String
```

**Parameters**

*bFullResync*

Flag to indicate if a full re-sync is required regardless of whether one has already happened.

**Return Value**

Comma separated list of registered IFilters on the system.

**Remarks****1.3.20 SearchCriteria Class**

For a list of all members of this type, see SearchCriteria Members.

**Description**

Represents the search criteria used by an instance of SearchEngine.

**Thread Safety**

This class is STA based and therefore safe for multithreaded access as long as the creating process continues to process its message loop.

**Remarks**

This class is not created directly but referenced from an instance of SearchEngine. The criteria can be loaded or saved in the SRF format compatible with the retail version of FileLocator Pro.

### Example

[Visual Basic]

```
engineSearch = New SearchEngineClass

engineSearch.SearchCriteria.FileName           = "*.txt"
engineSearch.SearchCriteria.ContainingText    = "search"
engineSearch.SearchCriteria.LookIn           = "c:\search folder"

engineSearch.SearchCriteria.ContentsExprType = ExpressionType.RegExpClassic

Dim listResult As SearchResultItemList = engineSearch.Start( False )
```

#### 1.3.20.1 SearchCriteria Members

SearchCriteria overview

##### Public Properties

ContainingText	Text expression to search for in files.
ContainingTextScript	ActiveScript to be used for additional content matching.
ContentsExprSpanFile	Flag indicating whether or not the ContainingText expressions should be matched across the whole file and not just the line.
ContentsExprType	The ContainingText expression type.
ExcludeFilename	Flag indicating whether or not the FileName criteria specifies an exclusion list.
FileAttributes	Attribute criteria for the file search.
FileName	Expression for selecting which files to search.
FileNameExprType	The FileName expression type.
FileNameScript	ActiveScript to be used for additional file name matching.
FileSize	Size criteria for the file search.
LookIn	Folder(s) expression for selecting which folders to search in.
LookInExprType	The LookIn expression type.
MatchContentsCase	Flag indicating whether or not the content

	matching should be case sensitive.
MatchFilenameCase	Flag indicating whether or not the file name matching should be case sensitive.
ModifiedDate	Date criteria for the file search.
RegExprType	Type of regular expression engine to use if ExpressionType.RegExp is specified for any criteria expression type.
SearchSubDirectory	Flag indicating whether or not sub folders should also be searched.

### Public Methods

Load	Loads the object with previously saved criteria in a valid SRF format.
LoadAll	Loads the object and the search engine configuration with previously saved criteria in a valid SRF format.
Save	Saves the current criteria into a file in SRF format.

## 1.3.20.2 Methods

### Public Methods

Load	Loads the object with previously saved criteria in a valid SRF format.
LoadAll	Loads the object and the search engine configuration with previously saved criteria in a valid SRF format.
Save	Saves the current criteria into a file in SRF format.

#### 1.3.20.2.1 Load Method

Loads the object with previously saved criteria in a valid SRF format.

[Visual Basic]

```
Public Sub Load(ByVal strFileName As String)
```

**Parameters***strFileName*

File name to load the criteria from.

**Remarks**

Loads the object with previously saved criteria in a valid SRF format.

**Example**

[Visual Basic]

```
engineSearch = New SearchEngineClass

engineSearch.SearchCriteria.Load( "c:\SavedCriteria.srf" )
engineSearch.SearchCriteria.ContainingText = "search"

Dim listResult As SearchResultItemList = engineSearch.Start( False )
```

## 1.3.20.2.2 LoadAll Method

Loads the object and the search engine configuration with previously saved criteria in a valid SRF format.

[Visual Basic]

```
Public Sub LoadAll(ByVal strFileName As String)
```

**Parameters***strFileName*

File name to load the criteria and configuration from.

**Remarks**

A SRF file not only contains the criteria for a search but also the engine configuration at the time of the search. While the Load method only loads the criteria information into the SearchCriteria object the LoadAll method also loads the SearchConfiguration object for the search engine with the configuration information stored in the SRF file.

**Example**

[Visual Basic]

```
engineSearch = New SearchEngineClass

engineSearch.SearchCriteria.LoadAll( "c:\SavedCriteria.srf" )
engineSearch.SearchCriteria.ContainingText = "search"

Dim listResult As SearchResultItemList = engineSearch.Start( False )
```

## 1.3.20.2.3 Save Method

Saves the current criteria into a file in SRF format.

[Visual Basic]

```
Public Sub Save(ByVal strFileName As String)
```

### Parameters

*strFileName*

File name to write the criteria to.

### Remarks

Saves the current criteria into a file in SRF format.

### Example

[Visual Basic]

```
engineSearch = New SearchEngineClass

engineSearch.SearchCriteria.Load( "c:\SavedCriteria.srf" )
engineSearch.SearchCriteria.ContainingText = "search"

engineSearch.SearchCriteria.Save( "c:\SearchHistory\LastSearch.srf" )

Dim listResult As SearchResultItemList = engineSearch.Start( False )
```

## 1.3.20.3 Properties

### Public Properties

ContainingText	Text expression to search for in files.
ContainingTextScript	ActiveScript to be used for additional content matching.
ContentsExprSpanFile	Flag indicating whether or not the ContainingText expressions should be matched across the whole file and not just the line.
ContentsExprType	The ContainingText expression type.
ExcludeFilename	Flag indicating whether or not the FileName criteria specifies an exclusion list.
FileAttributes	Attribute criteria for the file search.
FileName	Expression for selecting which files to search.
FileNameExprType	The FileName expression type.
FileNameScript	ActiveScript to be used for additional file name

	matching.
FileSize	Size criteria for the file search.
LookIn	Folder(s) expression for selecting which folders to search in.
LookInExprType	The LookIn expression type.
MatchContentsCase	Flag indicating whether or not the content matching should be case sensitive.
MatchFilenameCase	Flag indicating whether or not the file name matching should be case sensitive.
ModifiedDate	Date criteria for the file search.
RegExprType	Type of regular expression engine to use if ExpressionType.RegExp is specified for any criteria expression type.
SearchSubDirectory	Flag indicating whether or not sub folders should also be searched.

#### 1.3.20.3.1 ContainingText Property

Text expression to search for in files.

[Visual Basic]

```
Public Property ContainingText As String
```

#### Property Value

Text expression to search for in files.

#### Remarks

Use the ContentsExprType property to specify the type of the expression.

#### 1.3.20.3.2 ContainingTextScript Property

ActiveScript to be used for additional content matching.

[Visual Basic]

```
Public ReadOnly Property ContainingTextScript As ScriptingCriteria
```

#### Property Value

ActiveScript to be used for additional content matching.

#### Remarks

#### 1.3.20.3.3 ContentsExprSpanFile Property

Flag indicating whether or not the ContainingText expressions should be matched across the whole file and not just the line.

[Visual Basic]

```
Public Property ContentsExprSpanFile As Boolean
```

#### Property Value

True if the expression should be matched across the whole file, otherwise False if the expression should be matched across just the line.

#### Remarks

This property is only valid if the containing text expression type is Boolean (i.e. ContentsExprType = ExpressionType.Boolean). If any other expression type is chosen this flag is ignored.

Usually the search engine matches an expression on a line by line basis, i.e. each line must individually satisfy the expression to be considered a match. By using this property a Boolean expression can be matched over a whole file, allowing each line to partially satisfy the expression until the end of the file when the whole expression is evaluated using the partial matches.

#### 1.3.20.3.4 ContentsExprType Property

The ContainingText expression type.

[Visual Basic]

```
Public Property ContentsExprType As ExpressionType
```

#### Property Value

The ContainingText expression type.

#### Remarks

See ExpressionType enumeration for a list of valid values.

If the chosen expression type is Boolean then the ContentsExprSpanFile flag can be set to True to match the expression across the whole file instead of just on a line by line basis.

#### 1.3.20.3.5 ExcludeFilename Property

Flag indicating whether or not the FileName criteria specifies an exclusion list.

[Visual Basic]

```
Public Property ExcludeFilename As Boolean
```

**Property Value**

True if the FileName expression specifies an exclusion list, otherwise False.

**Remarks**

## 1.3.20.3.6 FileAttributes Property

Attribute criteria for the file search.

[Visual Basic]

```
Public ReadOnly Property FileAttributes As FileAttributes
```

**Property Value**

Attribute criteria for the file search.

**Remarks**

## 1.3.20.3.7 FileName Property

Expression for selecting which files to search.

[Visual Basic]

```
Public Property FileName As String
```

**Property Value**

Expression for selecting which files to search.

**Remarks**

## 1.3.20.3.8 FileNameExprType Property

The FileName expression type.

[Visual Basic]

```
Public Property FileNameExprType As ExpressionType
```

**Property Value**

The FileName expression type.

**Remarks**

See ExpressionType enumeration for a list of valid values.

#### 1.3.20.3.9 FileNameScript Property

ActiveScript to be used for additional file name matching.

[Visual Basic]

```
Public ReadOnly Property FileNameScript As ScriptingCriteria
```

#### Property Value

ActiveScript to be used for additional file name matching.

#### Remarks

#### 1.3.20.3.10 FileSize Property

Size criteria for the file search.

[Visual Basic]

```
Public ReadOnly Property FileSize As SizeRange
```

#### Property Value

Size criteria for the file search.

#### Remarks

#### 1.3.20.3.11 LookIn Property

Folder(s) expression for selecting which folders to search in.

[Visual Basic]

```
Public Property LookIn As String
```

#### Property Value

Folder(s) expression for selecting which folders to search in.

#### Remarks

#### 1.3.20.3.12 LookInExprType Property

The LookIn expression type.

[Visual Basic]

```
Public Property LookInExprType As ExpressionType
```

### Property Value

The LookIn expression type.

### Remarks

See ExpressionType enumeration for a list of valid values.

#### 1.3.20.3.13 MatchContentsCase Property

Flag indicating whether or not the content matching should be case sensitive.

[Visual Basic]

```
Public Property MatchContentsCase As Boolean
```

### Property Value

True if the content matching should be case sensitive, otherwise False.

### Remarks

#### 1.3.20.3.14 MatchFilenameCase Property

Flag indicating whether or not the file name matching should be case sensitive.

[Visual Basic]

```
Public Property MatchFilenameCase As Boolean
```

### Property Value

True if the file name matching should be case sensitive, otherwise False.

### Remarks

#### 1.3.20.3.15 ModifiedDate Property

Date criteria for the file search.

[Visual Basic]

```
Public ReadOnly Property ModifiedDate As DateRange
```

### Property Value

Date criteria for the file search.

### Remarks

#### 1.3.20.3.16 RegExprType Property

Type of regular expression engine to use if ExpressionType.RegExp is specified for any criteria expression type.

[Visual Basic]

```
Public Property RegExprType As RegularExpressionType
```

### Property Value

Type of regular expression engine to use if ExpressionType.RegExp is specified for any criteria expression type.

### Remarks

See RegularExpressionType enumeration for a list of valid values.

#### 1.3.20.3.17 SearchSubDirectory Property

Flag indicating whether or not sub folders should also be searched.

[Visual Basic]

```
Public Property SearchSubDirectory As Boolean
```

### Property Value

True if the search engine should search sub folders, otherwise False to search only the folders explicitly specified in the LookIn property.

### Remarks

## 1.3.21 SearchEngine Class

For a list of all members of this type, see SearchEngine Members.

### Description

Represents the core searching component.

### Thread Safety

This class is STA based and therefore safe for multithreaded access as long as the creating process continues to process its message loop.

### Remarks

SearchEngineClass is the only createable class in the library all other classes are created directly or indirectly through this class.

Before starting a search the caller should make sure that both the SearchCriteria and SearchConfiguration properties are appropriately initialized.

### Example

[Visual Basic]

```
engineSearch = New SearchEngineClass

engineSearch.SearchCriteria.FileName           = "*.txt"
engineSearch.SearchCriteria.ContainingText    = "search"
engineSearch.SearchCriteria.LookIn            = "c:\search folder"

engineSearch.SearchCriteria.ContentsExprType = ExpressionType.RegExpClassic

engineSearch.SearchConfiguration.SearchThreadCount = 1

Dim listResult As SearchResultItemList = engineSearch.Start( False )
```

#### 1.3.21.1 SearchEngine Members

SearchEngine overview

#### Public Properties

SearchConfiguration	Search engine configuration that controls the way the search engine will process files.
SearchCriteria	Criteria for running the search the next time the Start method is executed.

#### Public Methods

Cancel	Stops the current search.
Start	Start the search with the current criteria and configuration.

#### Public Events

OnFileFound	Occurs when one or more files are found.
OnProgress	Occurs at regular intervals to report search progress.
OnSearchFinish	Occurs at end of search.
OnSearchStart	Occurs at start of search.

These interfaces provide additional functionality to the SearchEngine Class.

ISearchEngine2

### 1.3.21.2 Events

#### Public Events

OnFileFound	Occurs when one or more files are found.
OnProgress	Occurs at regular intervals to report search progress.
OnSearchFinish	Occurs at end of search.
OnSearchStart	Occurs at start of search.

#### 1.3.21.2.1 OnFileFound Event

Occurs when one or more files are found.

[Visual Basic]

```
Public Event OnFileFound(ByVal listFileFound As SearchResultItemList)
```

#### Event Data

The event handler receives an argument of type SearchResultItemList containing one or objects of type SearchResultItem.

#### Remarks

The OnFileFound event is not fired for each individual file found. Instead, for performance reasons, it is fired at given intervals (approx. 250ms) if files are found during the interval. The SearchResultItemList argument contains the list of files found and is a subset of the list returned when calling the Start method.

**Example**

[Visual Basic]

```

Private Sub SearchEngine_OnFileFound(ByVal listFound As SearchResultItemList)
Handles m_engineSearch.OnFileFound

    ' Received when one or more files has been found. A list of files found since
the
    ' last OnFileFound message are in the parameter listFound.

Dim buildText As New System.Text.StringBuilder

For nItem As Integer = 0 To (listFound.Count - 1)
    ' Each item in the list includes information about the found
    ' item (e.g. its name and path) and also the list of text lines
    ' found if this was a content search

Dim item As SearchResultItem = listFound(nItem)

buildText.Append(item.Path)
buildText.Append(item.FileName)
buildText.Append(vbCrLf)

Next

System.Console.Write( buildText.ToString() )
End Sub

```

## 1.3.21.2.2 OnProgress Event

Occurs at regular intervals to report search progress.

[Visual Basic]

```

Public Event OnProgress(ByVal nCurrentPhase As Integer, ByVal strLocation As String
,
    ByVal nContentTotal As Double, ByVal nFileTotal As Double)

```

**Event Data**

The event handler receives the following arguments representing the current search progress.

nCurrentPhase	Value is either 1 or 2. Representing the phase of the search. Phase 1 finds all files matching the filename criteria and Phase 2 searches the contents of those files.
strLocation	String representing the current location of the search. If there is only a single search thread then this will be the full path of the current search location otherwise it will be a list of files currently being searched.
nContentTotal	The total size, in bytes, of file content that has been searched. This will always be less than or

	equal to nFileTotal.
nFileTotal	The total size, in bytes, of files that have currently matched the filename criteria.

### Remarks

This event is fired at approximately 100ms intervals providing a snapshot of the current search progress.

### Example

[Visual Basic]

```

Private Const BYTES_PER_MB As Integer = 1024 * 1000

Private Sub SearchEngine_OnProgress(ByVal nPhase As Integer, ByVal
strLocation As String, _
ByVal nContentTotal As Double, ByVal nFileTotal As Double
) Handles m_engineSearch.OnProgress

    ' Received at regular intervals during search to show the current
    progress of the search.

    LabelProgress.Text = String.Format("Phase {0} ({1:n2} of {2:n2})", _
nPhase, nContentTotal /
BYTES_PER_MB, nFileTotal / BYTES_PER_MB)
    LabelLocation.Text = "Searching: " + strLocation
    LabelProgress.Refresh()

End Sub

```

#### 1.3.21.2.3 OnSearchFinish Event

Occurs at end of search.

[Visual Basic]

```
Public Event OnSearchFinish()
```

### Remarks

This event is fired at the end of the search regardless of whether the search has been successful or if it was cancelled.

#### 1.3.21.2.4 OnSearchStart Event

Occurs at start of search.

[Visual Basic]

```
Public Event OnSearchStart()
```

**Remarks**

This event is fired once at the beginning of the search when the search engine has been initialized and after the search has started.

**1.3.21.3 Methods****Public Methods**

Cancel	Stops the current search.
Start	Start the search with the current criteria and configuration.

**1.3.21.3.1 Cancel Method**

Stops any currently running search on the search engine.

[Visual Basic]

```
Public Sub Cancel()
```

**Remarks**

If no search is running no action is taken.

**1.3.21.3.2 Start Method**

Starts a new search with the current criteria and configuration.

[Visual Basic]

```
Public Function Start(ByVal bRunOnNewThread As Boolean) As SearchResultItemList
```

**Parameters***bRunOnNewThread*

Flag indicating whether or not the search should be run on a new worker thread (True) or should run on the same thread as the caller (False).

**Return Value**

A new SearchResultItemList where all found items will be placed.

## Remarks

The search can be run on the same thread as the calling thread or optionally on a new worker thread. If the search is run on the same thread as the caller the calling thread is blocked until the search is completed, no events will be fired during the search, and the search will not be able to be cancelled.

If the search is run on a new worker thread then the Start method will return immediately and search progress is reported back as events. Therefore the caller must maintain a reference to the search engine until the search has finished. All events are posted back onto the calling thread, which therefore requires that the calling thread does not block itself through sleeping, busy waiting, or any other such blocking operation, i.e. the calling thread must continue to process its message loop.

If the search engine is destructed before the search has finished the search will stop. References to returned objects are still valid.

If the search engine is already running a search the method will fail.

If any search criteria values are invalid the method will fail.

## Example

[Visual Basic]

```
engineSearch = New SearchEngineClass

engineSearch.SearchCriteria.FileName           = "*.txt"
engineSearch.SearchCriteria.ContainingText    = "search"
engineSearch.SearchCriteria.LookIn            = "c:\search folder"

engineSearch.SearchCriteria.ContentsExprType = ExpressionType.RegExpClassic

engineSearch.SearchConfiguration.SearchThreadCount = 1

Try
    Dim listResult As SearchResultItemList = engineSearch.Start( False )
Catch err As Exception
    MessageBox.Show(err.Message)
End Try
```

### 1.3.21.4 Properties

#### Public Properties

SearchConfiguration	Search engine configuration that controls the way the search engine will process files.
SearchCriteria	Criteria for running the search the next time the Start method is executed.

#### 1.3.21.4.1 SearchConfiguration Property

Search engine configuration that controls the way the search engine will process files.

[Visual Basic]

```
Public ReadOnly Property SearchConfiguration As SearchConfiguration
```

### Property Value

The configuration data for the search.

### Remarks

#### 1.3.21.4.2 SearchCriteria Property

Criteria for running the search the next time the Start method is executed.

[Visual Basic]

```
Public ReadOnly Property SearchCriteria As SearchCriteria
```

### Property Value

The criteria for the search.

### Remarks

## 1.3.21.5 Interfaces

These interfaces provide additional functionality to the SearchEngine Class.

ISearchEngine2

#### 1.3.21.5.1 ISearchEngine2

For a list of all members of this type, see ISearchEngine2 Members.

### Description

This interface provides additional functionality for the SearchEngine Class.

#### 1.3.21.5.1.1 ISearchEngine2 Members

### Public Methods

NewSearchResultItemList	Creates a new SearchResultItemList Class.
-------------------------	---

## Public Methods

NewSearchResultItemList	Creates a new SearchResultItemList Class.
-------------------------	---

Creates a new SearchResultItemList Class.

[Visual Basic]

```
Public Function NewSearchResultItemList() As SearchResultItemList
```

## Parameters

## Return Value

A new SearchResultItemList.

## Remarks

Use this method for situations where a list is required without first performing a search, e.g. for re-creating a list from an old search.

## 1.3.22 SearchResultItem Class

For a list of all members of this type, see SearchResultItem Members.

## Description

Represents an item found during a search.

## Thread Safety

This class is safe for multithreaded read access.

## Remarks

## Example

[Visual Basic]

```
engineSearch = New SearchEngineClass

engineSearch.SearchCriteria.FileName           = "*.txt"
engineSearch.SearchCriteria.ContainingText    = "search"
engineSearch.SearchCriteria.LookIn            = "c:\search folder"
```

```

Dim listResult As SearchResultItemList = engineSearch.Start( False )

' Output the files, with their text lines and hits to the console

Dim buildText As New System.Text.StringBuilder

For nItem As Integer = 0 To (listResult.Count - 1)
    ' Each item in the list includes information about the found
    ' item (e.g. its name and path) and also the list of text lines
    ' found if this was a content search

    Dim item As SearchResultItem = listResult(nItem)

    buildText.Append(item.Path)
    buildText.Append(item.FileName)
    buildText.Append(vbCrLf)

    Dim listText As TextLineList = item.TextLineList

    For nText As Integer = 0 To (listText.Count - 1)
        ' Each text line includes the line number and text found and
also
        ' a list showing where the expression matches occurred.

        Dim line As TextLine = listText(nText)

        Dim listMatch As FLProCoreLib.TextMatchList = line.TextMatchList
        For nHit As Integer = 0 To (listMatch.Count - 1)
            buildText.AppendFormat(" {0}:{1} ", listMatch(nHit).
Start, listMatch(nHit).Length)
        Next
        buildText.Append(vbCrLf)
    Next
Next

System.Console.Write( buildText.ToString() )

```

### 1.3.22.1 SearchResultItem Members

SearchResultItem overview

#### Public Properties

FileName	Name of the result item.
IsFolder	Flag indicating if the result item is a folder.
ModifiedDate	Last modified date of the result item.
Path	Path of the result item.
Size	File size of the result item.
TextLineList	List of lines found that match the given criteria.

These interfaces provide additional functionality to the SearchResultItem class.

ISearchResultItem2

ISearchResultItem3

### 1.3.22.2 Properties

#### Public Properties

FileName	Name of the result item.
IsFolder	Flag indicating if the result item is a folder.
ModifiedDate	Last modified date of the result item.
Path	Path of the result item.
Size	File size of the result item.
TextLineList	List of lines found that match the given criteria.

#### 1.3.22.2.1 FileName Property

Name of the result item.

[Visual Basic]

```
Public ReadOnly Property FileName As String
```

#### Property Value

Name of the result item.

#### Remarks

Usually a file name. The path of the result item is in the Path property.

#### 1.3.22.2.2 IsFolder Property

Flag indicating if the result item is a folder.

[Visual Basic]

```
Public ReadOnly Property IsFolder As Boolean
```

#### Property Value

True if the result item is a folder, otherwise False.

**Remarks**

## 1.3.22.2.3 ModifiedDate Property

Last modified date of the result item.

[Visual Basic]

```
Public ReadOnly Property ModifiedDate As DateTime
```

**Property Value**

Last modified date of the result item.

**Remarks**

## 1.3.22.2.4 Path Property

Path of the result item.

[Visual Basic]

```
Public ReadOnly Property Path As String
```

**Property Value**

Path of the result item.

**Remarks**

## 1.3.22.2.5 Size Property

File size of the result item.

[Visual Basic]

```
Public ReadOnly Property Size As Double
```

**Property Value**

File size of the result item in bytes.

**Remarks**

## 1.3.22.2.6 TextLineList Property

List of lines found that match the given criteria.

[Visual Basic]

---

```
Public ReadOnly Property TextLineList As TextLineList
```

### Property Value

List of lines found that match the given criteria.

### Remarks

If the search was not a content search then the property will be an empty list.

### 1.3.22.3 Interfaces

These interfaces provide additional functionality to the SearchResultItem class.

ISearchResultItem2  
ISearchResultItem3

#### 1.3.22.3.1 ISearchResultItem2

For a list of all members of this type, see ISearchResultItem2 Members.

### Description

This interface provides additional functionality for the SearchResultItem Class.

#### 1.3.22.3.1.1 ISearchResultItem2 Members

### Public Methods

CopyFileTo	Copies the file to a specific location
ExtractText	Extracts the text from the file

#### 1.3.22.3.1.2 Methods

### Public Methods

CopyFileTo	Copies the file to a specific location
ExtractText	Extracts the text from the file

Copies the file to a specific location

[Visual Basic]

```
Public Sub CopyFileTo(ByVal strCopyToFile As String)
```

### Parameters

*strCopyToFile*

Full path, including file name, of the destination file.

### Remarks

Copies the item (which must be a file) to a specific location.

Extracts the text from the file

[Visual Basic]

```
Public Function ExtractText() As String
```

### Remarks

The full text for the file.

#### 1.3.22.3.2 ISearchResultItem3

For a list of all members of this type, see [ISearchResultItem3 Members](#).

### Description

This interface provides additional functionality for the [SearchResultItem Class](#).

#### 1.3.22.3.2.1 ISearchResultItem3 Members

### Public Properties

Id	Id of the result item.
----	------------------------

#### 1.3.22.3.2.2 Properties

### Public Properties

Id	Id of the result item.
----	------------------------

Id of the result item.

[Visual Basic]

```
Public ReadOnly Property Id As String
```

### Property Value

Id of the result item.

### Remarks

This Id can be used to re-create the item using the CreateItemFromId method.

## 1.3.23 SearchResultItemList Class

For a list of all members of this type, see SearchResultItemList Members.

### Description

Represents a list of SearchResultItem items.

### Thread Safety

This class is safe for multithreaded read access.

### Remarks

### Example

[Visual Basic]

```
engineSearch = New SearchEngineClass

engineSearch.SearchCriteria.FileName      = "*.txt"
engineSearch.SearchCriteria.ContainingText = "search"
engineSearch.SearchCriteria.LookIn      = "c:\search folder"

Dim listResult As SearchResultItemList = engineSearch.Start( False )

' Output the files, with their text lines and hits to the console

Dim buildText As New System.Text.StringBuilder

For nItem As Integer = 0 To (listResult.Count - 1)
    ' Each item in the list includes information about the found
    ' item (e.g. its name and path) and also the list of text lines
    ' found if this was a content search

    Dim item As SearchResultItem = listResult(nItem)
```

```

        buildText.Append(item.Path)
        buildText.Append(item.FileName)
        buildText.Append(vbCrLf)

        Dim listText As TextLineList = item.TextLineList

        For nText As Integer = 0 To (listText.Count - 1)
            ' Each text line includes the line number and text found and
also
            ' a list showing where the expression matches occurred.

            Dim line As TextLine = listText(nText)

            Dim listMatch As FLProCoreLib.TextMatchList = line.TextMatchList
            For nHit As Integer = 0 To (listMatch.Count - 1)
                buildText.AppendFormat(" {0}:{1} ", listMatch(nHit).
Start, listMatch(nHit).Length)
            Next
            buildText.Append(vbCrLf)
        Next
    Next

    System.Console.Write( buildText.ToString() )

```

### 1.3.23.1 SearchResultItemList Members

SearchResultItemList overview

#### Public Properties

Count	Total number of search result items in the list.
ExportCriteria	Criteria used for exporting.
Item	Integer indexed (zero based index) list of SearchResultItem items.
TotalFileSize	Total size, in bytes, of the files in the list.

#### Public Methods

ExportToFile	Exports the list of items to a file using the current ExportCriteria.
Export	Exports the list of items to a string using the current ExportCriteria.

These interfaces provide additional functionality to the SearchResultItemList class.

ISearchResultItemList2  
ISearchResultItemList3

### 1.3.23.2 Properties

#### Public Properties

Count	Total number of search result items in the list.
ExportCriteria	Criteria used for exporting.
Item	Integer indexed (zero based index) list of SearchResultItem items.
TotalFileSize	Total size, in bytes, of the files in the list.

#### 1.3.23.2.1 Count Property

Total number of search result items in the list.

[Visual Basic]

```
Public ReadOnly Property Count As Integer
```

#### Property Value

Total number of search result items in the list.

#### Remarks

#### 1.3.23.2.2 Item Property

Integer indexed (zero based index) list of SearchResultItem items.

[Visual Basic]

```
Public ReadOnly Item(ByVal nIndex As Integer) As SearchResultItem
```

#### Parameters

*nIndex*

The zero-based index of the result item

#### Property Value

A SearchResultItem item that contains the result item information.

#### Remarks

## 1.3.23.2.3 ExportCriteria Property

Criteria used for exporting.

[Visual Basic]

```
Public ReadOnly Property ExportCriteria As ExportCriteria
```

#### Property Value

Criteria used for exporting.

#### Remarks

## 1.3.23.2.4 TotalFileSize Property

Total size, in bytes, of the files in the list.

[Visual Basic]

```
Public ReadOnly Property TotalFileSize As Double
```

#### Property Value

Total size, in bytes, of the files in the list.

#### Remarks

## 1.3.23.3 Methods

#### Public Methods

ExportToFile	Exports the list of items to a file using the current ExportCriteria.
Export	Exports the list of items to a string using the current ExportCriteria.

## 1.3.23.3.1 ExportToFile Method

Exports the list of items to a file using the current ExportCriteria.

[Visual Basic]

```
Public Sub ExportToFile(ByVal strFileName As String)
```

## Parameters

### *strFileName*

File name to write the export results to.

## Remarks

If the file already exists the method overwrites the file.

## Example

[Visual Basic]

```
engineSearch = New SearchEngineClass

engineSearch.SearchCriteria.FileName           = "*.txt"
engineSearch.SearchCriteria.ContainingText    = "search"
engineSearch.SearchCriteria.LookIn            = "c:\search folder"

Dim listResult As SearchResultItemList = engineSearch.Start( False )

' Export the results to a file

listResult.ExportCriteria.ExportContents      = True
listResult.ExportCriteria.ExportSurroundingLines = False
listResult.ExportCriteria.ExportFormat        = ExportFormatType.HTML

listResult.ExportToFile( "C:\Results.html" )
```

### 1.3.23.3.2 Export Method

Exports the list of items to a string using the current ExportCriteria.

[Visual Basic]

```
Public Function Export() As String
```

## Return Value

The exported data in the specified format.

## Remarks

## Example

[Visual Basic]

```
engineSearch = New SearchEngineClass

engineSearch.SearchCriteria.FileName           = "*.txt"
```

```

engineSearch.SearchCriteria.ContainingText = "search"
engineSearch.SearchCriteria.LookIn       = "c:\search folder"

Dim listResult As SearchResultItemList = engineSearch.Start( False )

' Export the results to a file

listResult.ExportCriteria.ExportContents      = True
listResult.ExportCriteria.ExportSurroundingLines = False
listResult.ExportCriteria.ExportFormat        = ExportFormatType.Text

System.Console.Write( listResult.Export() )

```

### 1.3.23.4 Interfaces

These interfaces provide additional functionality to the SearchResultItemList class.

ISearchResultItemList2  
ISearchResultItemList3

#### 1.3.23.4.1 ISearchResultItemList2

For a list of all members of this type, see ISearchResultItemList2 Members.

#### Description

This interface provides additional functionality for the SearchResultItemList Class.

##### 1.3.23.4.1.1 ISearchResultItemList2 Members

#### Public Methods

AddItem	Adds an existing item to the list.
CreateItemFromXML	Creates a new SearchResultItem based on provided Xml.
LoadFromFile	Loads the list based on Xml loaded from the given file.

#### Public Methods

AddItem	Adds an existing item to the list.
CreateItemFromXML	Creates a new SearchResultItem based on provided Xml.
LoadFromFile	Loads the list based on Xml loaded from the given file.

Adds an existing item to the list.

[Visual Basic]

```
Public Sub AddItem(ByVal itemAdd As SearchResultItem)
```

### Parameters

*itemAdd*

SearchResultItem to add to this list.

### Remarks

Creates a new SearchResultItem based on provided Xml.

[Visual Basic]

```
Public Function CreateItemFromXML(ByVal strXml As String, ByVal bAddToList As Boolean) As SearchResultItem
```

### Parameters

*strXml*

XML used to create a specific item.

*bAddToList*

Boolean to indicate if the created item should be added to the list.

### Return Value

A new SearchResultItem created from the Xml.

### Remarks

Loads the list based on Xml loaded from the given file.

[Visual Basic]

```
Public Sub LoadFromFile(ByVal strPath As String)
```

### Parameters

*strPath*

XML used to create a specific item.

## Return Value

## Remarks

### 1.3.23.4.2 ISearchResultItemList3

For a list of all members of this type, see ISearchResultItemList3 Members.

## Description

This interface provides additional functionality for the SearchResultItemList Class.

### 1.3.23.4.2.1 ISearchResultItemList3 Members

## Public Methods

CreateItemFromId	Creates a new SearchResultItem based on provided item Id.
CreateItemFromPath	Creates a new SearchResultItem based on the path to a given item.

### 1.3.23.4.2.2 Methods

## Public Methods

CreateItemFromId	Creates a new SearchResultItem based on provided item Id.
CreateItemFromPath	Creates a new SearchResultItem based on the path to a given item.

Creates a new SearchResultItem based on provided item Id.

[Visual Basic]

```
Public Function CreateItemFromId(ByVal strId As String) As SearchResultItem
```

## Parameters

*strId*

Id used to identify an item (based on the item's Id property).

**Return Value**

A new SearchResultItem created from the Xml.

**Remarks**

The Id for an item can be obtained by using the Id Property of an item.

Creates a new SearchResultItem based on the path to a given item.

[Visual Basic]

```
Public Function CreateItemFromPath(ByVal strPath As String) As SearchResultItem
```

**Parameters**

*strPath*

Path to an item.

**Return Value**

A new SearchResultItem created from the item's path.

**Remarks**

This method only works for standard file system items, i.e. it does NOT work for items contained within compressed files. To get items within compressed files perform a search setting the LookIn Property to be the path within the compressed file.

### 1.3.24 SizeRange Class

For a list of all members of this type, see SizeRange Members.

**Description**

Represents a size range used by SearchCriteria.

**Thread Safety**

This class is STA based and therefore safe for multithreaded access as long as the creating process continues to process its message loop.

**Remarks**

To stop searching by size the size range needs to be cleared using the Clear method.

**Example**

[Visual Basic]

```

engineSearch = New SearchEngineClass

' Look for all files less than 5k in size.

engineSearch.SearchCriteria.FileSize.LessThan    = (5 * 1024)
engineSearch.SearchCriteria.LookIn              = "c:\search folder"

Dim listResult As SearchResultItemList = engineSearch.Start( False )

```

**1.3.24.1 SizeRange Members**

SizeRange overview

**Public Properties**

GreaterThan	Beginning of the size range.
LessThan	End of the size range.

**Public Methods**

Clear	Clears the size range and marks it as not to be used during search.
-------	---

**1.3.24.2 Methods****Public Methods**

Clear	Clears the size range and marks it as not to be used during search.
-------	---

**1.3.24.2.1 Clear Method**

Clears the size range and marks it as not to be used during search.

[Visual Basic]

```
Public Sub Clear()
```

---

**Remarks****1.3.24.3 Properties****Public Properties**

GreaterThan	Beginning of the size range.
LessThan	End of the size range.

## 1.3.24.3.1 GreaterThan Property

Beginning of the size range.

[Visual Basic]

```
Public Property GreaterThan As Double
```

**Property Value**

Beginning of the size range.

**Remarks**

## 1.3.24.3.2 LessThan Property

End of the size range.

[Visual Basic]

```
Public Property LessThan As Double
```

**Property Value**

End of the size range.

**Remarks****1.3.25 TextLine Class**

For a list of all members of this type, see TextLineList Members.

**Description**

Represents a line of found or surrounding text.

### Thread Safety

This class is safe for multithreaded read access.

### Remarks

Text lines that are found will have one or more matches in the TextMatchList property but text lines that are surrounding lines will not have any matches.

### Example

[Visual Basic]

```
engineSearch = New SearchEngineClass

engineSearch.SearchCriteria.FileName           = "*.txt"
engineSearch.SearchCriteria.ContainingText    = "search"
engineSearch.SearchCriteria.LookIn           = "c:\search folder"

Dim listResult As SearchResultItemList = engineSearch.Start( False )

' Output the files, with their text lines and hits to the console

Dim buildText As New System.Text.StringBuilder

For nItem As Integer = 0 To (listResult.Count - 1)
    ' Each item in the list includes information about the found
    ' item (e.g. its name and path) and also the list of text lines
    ' found if this was a content search

    Dim item As SearchResultItem = listResult(nItem)

    buildText.Append(item.Path)
    buildText.Append(item.FileName)
    buildText.Append(vbCrLf)

    Dim listText As TextLineList = item.TextLineList

    For nText As Integer = 0 To (listText.Count - 1)
        ' Each text line includes the line number and text found and
also
        ' a list showing where the expression matches occurred.

        Dim line As TextLine = listText(nText)

        Dim listMatch As FLProCoreLib.TextMatchList = line.TextMatchList
        For nHit As Integer = 0 To (listMatch.Count - 1)
            buildText.AppendFormat(" {0}:{1} ", listMatch(nHit).
Start, listMatch(nHit).Length)
        Next
        buildText.Append(vbCrLf)
    Next
Next
Next
```

```
System.Console.WriteLine( buildText.ToString() )
```

### 1.3.25.1 TextLine Members

TextLine overview

#### Public Properties

LineNumber	Line number of the text line.
LinesAfter	Surrounding lines trailing the found line.
LinesBefore	Surrounding lines leading the found line.
Text	Text of the line.
TextMatchHighlightList	List of non-overlapping matches of the containing text criteria on the line.
TextMatchList	List of positive matches of the containing text criteria on the line.

### 1.3.25.2 Properties

#### Public Properties

LineNumber	Line number of the text line.
LinesAfter	Surrounding lines trailing the found line.
LinesBefore	Surrounding lines leading the found line.
Text	Text of the line.
TextMatchHighlightList	List of non-overlapping matches of the containing text criteria on the line.
TextMatchList	List of positive matches of the containing text criteria on the line.

#### 1.3.25.2.1 LineNumber Property

Line number of the text line.

[Visual Basic]

```
Public ReadOnly Property LineNumber As Integer
```

**Property Value**

Line number of the text line.

**Remarks**

## 1.3.25.2.2 LinesAfter Property

Surrounding lines trailing the found line.

[Visual Basic]

```
Public ReadOnly Property LinesAfter As TextLineList
```

**Property Value**

List of surrounding lines trailing the found line.

**Remarks**

Surrounding lines do not produce their own subsequent surrounding line list.

## 1.3.25.2.3 LinesBefore Property

Surrounding lines leading the found line.

[Visual Basic]

```
Public ReadOnly Property LinesBefore As TextLineList
```

**Property Value**

List of surrounding lines leading the found line.

**Remarks**

Surrounding lines do not return their own subsequent surrounding line list.

## 1.3.25.2.4 Text Property

Text of the line.

[Visual Basic]

```
Public ReadOnly Property Text As String
```

**Property Value**

Text of the line.

## Remarks

### 1.3.25.2.5 TextMatchHighlightList Property

List of non-overlapping matches of the containing text criteria on the line.

[Visual Basic]

```
Public ReadOnly Property TextMatchHighlightList As TextMatchList
```

## Property Value

List of non-overlapping matches of the containing text criteria on the line.

## Remarks

The list is generated from the TextMatchList property so that overlapping matches are concatenated into a single match. Surrounding lines return an empty match list.

### 1.3.25.2.6 TextMatchList Property

List of positive matches of the containing text criteria on the line.

[Visual Basic]

```
Public ReadOnly Property TextMatchList As TextMatchList
```

## Property Value

List of positive matches of the containing text criteria on the line.

## Remarks

Surrounding lines return an empty match list.

## 1.3.26 TextLineList Class

For a list of all members of this type, see TextLineList Members.

### Description

Represents a list of TextLine items.

### Thread Safety

This class is safe for multithreaded read access.

## Remarks

## Example

[Visual Basic]

```

engineSearch = New SearchEngineClass

engineSearch.SearchCriteria.FileName           = "*.txt"
engineSearch.SearchCriteria.ContainingText    = "search"
engineSearch.SearchCriteria.LookIn            = "c:\search folder"

Dim listResult As SearchResultItemList = engineSearch.Start( False )

' Output the files, with their text lines and hits to the console

Dim buildText As New System.Text.StringBuilder

For nItem As Integer = 0 To (listResult.Count - 1)
    ' Each item in the list includes information about the found
    ' item (e.g. its name and path) and also the list of text lines
    ' found if this was a content search

    Dim item As SearchResultItem = listResult(nItem)

    buildText.Append(item.Path)
    buildText.Append(item.FileName)
    buildText.Append(vbCrLf)

    Dim listText As TextLineList = item.TextLineList

    For nText As Integer = 0 To (listText.Count - 1)
        ' Each text line includes the line number and text found and
also
        ' a list showing where the expression matches occurred.

        Dim line As TextLine = listText(nText)

        Dim listMatch As FLProCoreLib.TextMatchList = line.TextMatchList
        For nHit As Integer = 0 To (listMatch.Count - 1)
            buildText.AppendFormat(" {0}:{1} ", listMatch(nHit).
Start, listMatch(nHit).Length)
        Next
        buildText.Append(vbCrLf)
    Next
Next

System.Console.Write( buildText.ToString() )

```

### 1.3.26.1 TextLineList Members

TextLineList overview

#### Public Properties

Count	Total number of lines in the list.
Item	Integer indexed (zero based index) list of TextLine items.

### 1.3.26.2 Properties

#### Public Properties

Count	Total number of lines in the list.
Item	Integer indexed (zero based index) list of TextLine items.

#### 1.3.26.2.1 Item Property

Integer indexed (zero based index) list of TextLine items.

[Visual Basic]

```
Public ReadOnly Item(ByVal nIndex As Integer) As TextLine
```

#### Parameters

*nIndex*

The zero-based index of the line

#### Property Value

A TextLine item that contains the line information.

#### Remarks

#### 1.3.26.2.2 Count Property

Total number of lines in the list.

[Visual Basic]

```
Public ReadOnly Property Count As Integer
```

#### Property Value

Total number of lines in the list.

#### Remarks

### 1.3.27 TextMatch Class

For a list of all members of this type, see TextMatch Members.

#### Description

Represents a positive match of the containing text expression on a given TextLine.

#### Thread Safety

This class is safe for multithreaded read access.

#### Remarks

#### Example

[Visual Basic]

```
engineSearch = New SearchEngineClass

engineSearch.SearchCriteria.FileName           = "*.txt"
engineSearch.SearchCriteria.ContainingText    = "search"
engineSearch.SearchCriteria.LookIn           = "c:\search folder"

Dim listResult As SearchResultItemList = engineSearch.Start( False )

' Output the files, with their text lines and hits to the console

Dim buildText As New System.Text.StringBuilder

For nItem As Integer = 0 To (listResult.Count - 1)
    ' Each item in the list includes information about the found
    ' item (e.g. its name and path) and also the list of text lines
    ' found if this was a content search

    Dim item As SearchResultItem = listResult(nItem)

    buildText.Append(item.Path)
    buildText.Append(item.FileName)
    buildText.Append(vbCrLf)

    Dim listText As TextLineList = item.TextLineList

    For nText As Integer = 0 To (listText.Count - 1)
        ' Each text line includes the line number and text found and
also
        ' a list showing where the expression matches occurred.

        Dim line As TextLine = listText(nText)
```

```

        Dim listMatch As FLProCoreLib.TextMatchList = line.TextMatchList
        For nHit As Integer = 0 To (listMatch.Count - 1)
            buildText.AppendFormat(" {0}:{1} ", listMatch(nHit).
Start, listMatch(nHit).Length)
        Next
        buildText.Append(vbCrLf)
    Next
Next
System.Console.Write( buildText.ToString() )

```

### 1.3.27.1 TextMatch Members

TextMatch overview

#### Public Properties

Length	The length of the match, in characters.
Start	Start point on the line of text where the match occurs.

### 1.3.27.2 Properties

#### Public Properties

Length	The length of the match, in characters.
Start	Start point on the line of text where the match occurs.

#### 1.3.27.2.1 Length Property

The length of the match, in characters.

[Visual Basic]

```
Public ReadOnly Property Length As Integer
```

#### Property Value

The length of the match, in characters.

#### Remarks

### 1.3.27.2.2 Start Property

Start point on the line of text where the match occurs.

[Visual Basic]

```
Public ReadOnly Property Start As Integer
```

#### Property Value

Start point on the line of text where the match occurs.

#### Remarks

## 1.3.28 TextMatchList Class

For a list of all members of this type, see [TextMatchList Members](#).

#### Description

Represents a list of matches on a given `TextLine`.

#### Thread Safety

This class is safe for multithreaded read access.

#### Remarks

#### Example

[Visual Basic]

```
engineSearch = New SearchEngineClass

engineSearch.SearchCriteria.FileName           = "*.txt"
engineSearch.SearchCriteria.ContainingText    = "search"
engineSearch.SearchCriteria.LookIn           = "c:\search folder"

Dim listResult As SearchResultItemList = engineSearch.Start( False )

' Output the files, with their text lines and hits to the console

Dim buildText As New System.Text.StringBuilder

For nItem As Integer = 0 To (listResult.Count - 1)
    ' Each item in the list includes information about the found
    ' item (e.g. its name and path) and also the list of text lines
    ' found if this was a content search

    Dim item As SearchResultItem = listResult(nItem)

    buildText.Append(item.Path)
    buildText.Append(item.FileName)
```

```

buildText.Append(vbCrLf)

Dim listText As TextLineList = item.TextLineList

For nText As Integer = 0 To (listText.Count - 1)
    ' Each text line includes the line number and text found and
also
    ' a list showing where the expression matches occurred.

    Dim line As TextLine = listText(nText)

    Dim listMatch As FLProCoreLib.TextMatchList = line.TextMatchList
    For nHit As Integer = 0 To (listMatch.Count - 1)
        buildText.AppendFormat(" {0}:{1} ", listMatch(nHit).
Start, listMatch(nHit).Length)
    Next
    buildText.Append(vbCrLf)
Next
Next

System.Console.Write( buildText.ToString() )

```

### 1.3.28.1 TextMatchList Members

TextMatchList overview

#### Public Properties

Count	Total number of matches in the list.
Item	Integer indexed (zero based index) list of TextMatch items.

### 1.3.28.2 Properties

#### Public Properties

Count	Total number of matches in the list.
Item	Integer indexed (zero based index) list of TextMatch items.

#### 1.3.28.2.1 Count Property

Total number of matches in the list.

[Visual Basic]

```
Public ReadOnly Property Count As Integer
```

### Property Value

Total number of matches in the list.

### Remarks

#### 1.3.28.2.2 Item Property

Integer indexed (zero based index) list of TextMatch items.

[Visual Basic]

```
Public ReadOnly Item(ByVal nIndex As Integer) As TextMatch
```

### Parameters

*nIndex*

The zero-based index of the match

### Property Value

A TextMatch item that contains the match information.

### Remarks

## 1.3.29 TimeOffset Class

For a list of all members of this type, see TimeOffset Members.

### Description

Represents an offset from a particular TimeValue Class.

#### 1.3.29.1 TimeOffset Members

##### Public Properties

Active	Flag indicating whether or not the offset should be applied.
TimeResolution	Units that the offset is specified in.
Value	Value for the offset, units are specified by TimeResolution.

### 1.3.29.2 Properties

#### Public Properties

Active	Flag indicating whether or not the offset should be applied.
TimeResolution	Units that the offset is specified in.
Value	Value for the offset, units are specified by TimeResolution.

#### 1.3.29.2.1 Active Property

Flag indicating whether or not the offset should be applied.

[Visual Basic]

```
Public Property Active As Boolean
```

#### Property Value

True if the offset should be applied to the TimeValue, otherwise False.

#### Remarks

#### 1.3.29.2.2 TimeResolution Property

Units that the offset is specified in.

[Visual Basic]

```
Public Property TimeResolution As TimeResolutionType
```

#### Property Value

Units that the offset is specified in.

#### Remarks

#### 1.3.29.2.3 Value Property

Value for the offset, units are specified by TimeResolution Property.

[Visual Basic]

```
Public Property Value As Integer
```

#### Property Value

Value for the offset.

#### Remarks

### 1.3.30 TimeValue Class

For a list of all members of this type, see TimeValue Members.

#### Description

Represents a time value used in DateTimeValue Class.

#### 1.3.30.1 TimeValue Members

##### Public Properties

Absolute	Specific time value.
Active	Flag indicating whether or not the time portion of the DateTimeValue should be used.
Offset	Offset for the time value.
RelativeTimeType	Relative time type for this time value.

#### 1.3.30.2 Properties

##### Public Properties

Absolute	Specific time value.
Active	Flag indicating whether or not the time portion of the DateTimeValue should be used.
Offset	Offset for the time value.
RelativeTimeType	Relative time type for this time value.

##### 1.3.30.2.1 Absolute

Specific time value.

[Visual Basic]

```
Public Property Absolute As DateTime
```

#### Property Value

A specific/fixed (as opposed to relative) time value.

#### Remarks

#### 1.3.30.2.2 Active

Flag indicating whether or not the time portion of the DateTimeValue should be used.

[Visual Basic]

```
Public Property Active As Boolean
```

#### Property Value

True if the time portion of the DateTimeValue should be used, otherwise False.

#### Remarks

#### 1.3.30.2.3 Offset

Offset for the time value.

[Visual Basic]

```
Public ReadOnly Property Offset As TimeSpan
```

#### Property Value

Adjustment to the time value, e.g. +1 min.

#### Remarks

#### 1.3.30.2.4 RelativeTimeType

Relative time type for this time value.

[Visual Basic]

```
Public Property RelativeTimeType As RelativeTimeType
```

#### Property Value

Relative time type for this time value, e.g. Now.

#### Remarks

### 1.3.31 TimeResolutionType Enumeration

Possible types of time units

[Visual Basic]

```
Public Enum TimeResolutionType As Integer
```

**Remarks****Members**

Second	
Minute	
Hour	

## 1.4 Extension Interfaces

By default FileLocator Pro searches all file types by reading the raw binary data of the file. However, for certain well known file types FileLocator Pro provides extensions to interpret the file types and provide more meaningful results. Since the extensions are simply COM objects the extension architecture is open to 3rd party developers to write their own extensions.

The extension interfaces:

IExtCompositeInterpreter	Supplies composite file information for files types such as ZIP, TAR etc.
IExtCompositeInterpreter2	Optional interface that if implemented provides the composite interpreter with the original file name of the composite file.
IExtDataInterpreter	(BETA interface not supported)
IExtInitialize	Optional interface that if implemented is called to Initialize and Uninitialize the component.
IExtTextConverter	Converts a given file into plain text for easy searching.

To inform the core search engine about the availability of an extension a configuration file is required.

### 1.4.1 Extension Configuration File

Extension definitions are supplied to the core search engine through XML files found in the plugin\_cfg sub-folder. The values specified are:

displayname	Display name for the extension.
uniquename	Globally unique name for the extension.
progid	COM Prog Id for the extension.

filetypes	Comma separated list of file types that the extension handles.
interpretertype	Type of the interpreter: compositefile, textconverter, or textinterpreter.
active	Flag indicating if the default setting of the extension is set to active.
safemode	Flag indicating if the default setting of the extension is to run in a separate process from the search process.
useisfilter	Flag indicating if the default setting of the extension is to test for the availability of a registered IFilter instead of the extension (if one is available).

**Example:**

```
<?xml version="1.0"?>
<InterpreterConfig xmlns="http://www.mythicsoft.com/FileLocator">
  <displayname>ZIP</displayname>
  <uniquename>http://www.mythicsoft.com/zipinterpreter</uniquename>
  <progid>Extensions.ZIPInterpreter</progid>
  <filetypes>zip</filetypes>
  <interpretertype>compositefile</interpretertype>
  <active>no</active>
  <safemode>no</safemode>
  <useisfilter>no</useisfilter>
</InterpreterConfig>
```

**1.4.2 IExtCompositeInterpreter**

Supplies composite file information for files types such as ZIP, TAR etc.

**Methods**

Close	Closes the composite file.
ExtractFile	Extracts the file, specified by its key, to the given path with the given name.
GetNextFileInfo	Iterates through the entries in the composite file.
Open	Opens the composite file.
SetFilePos	Sets which file the iterator points to.

**Remarks**

Composite interpreter rules:

1. Composite interpreters should return each name as the full pathname, within the composite to the file. When the file comes to be displayed file names will be extracted by looking for the last '\'.
2. Composite interpreters should be able to iterate through the entire list irregardless of the internal file hierarchy.
3. Composite interpreters should be able to access a file directly from the key provided during `GetNextFileInfo`.
4. Only files are ever returned from a composite file, directory entries should be ignored. Composite files within composite files should be returned as files and let core search engine decide that it is actually a container.

#### 1.4.2.1 Close

Closes the composite file.

[C++]

```
HRESULT Close();
```

#### Return Value

S\_OK

File was successfully closed.

E\_FAIL

An error occurred.

#### Remarks

#### 1.4.2.2 ExtractFile

Extracts the file, specified by its key, to the given path with the given name.

[C++]

```
HRESULT ExtractFile(  
    BSTR bstrKey,  
    BSTR bstrExtractToPath,  
    BSTR bstrExtractToName  
);
```

#### Parameters

*bstrKey*

[in] Unique key to identify the file to extract, as supplied by `GetNextFileInfo`.

*bstrExtractToPath*

[in] Path of the folder that the extracted file should be written to.

*bstrExtractToName*

[in] Name to give the extracted file.

### Return Value

S\_OK

File was successfully extracted.

E\_FAIL

An error occurred.

### Remarks

#### 1.4.2.3 GetNextFileInfo

Iterates through the entries in the composite file.

[C++]

```
HRESULT GetNextFileInfo(  
    BSTR * pbstrName,  
    BSTR * pbstrKey,  
    double * pfSize,  
    DATE * pdtModified  
);
```

### Parameters

*pbstrName*

[out] Full path of the file relative to the composite file.

*pbstrKey*

[out] Unique identifier used to extract the file from the composite file.

*pfSize*

[out] Size of the file in bytes.

*pdtModified*

[out] Last modified date of the file.

### Return Value

S\_OK

File was successfully returned.

CINT\_S\_NOMOREFILES

No more files left in list.

E\_FAIL

An error occurred.

## Remarks

Composite interpreter rules:

1. Composite interpreters should return each name as the full pathname, within the composite to the file. When the file comes to be displayed file names will be extracted by looking for the last '\'.
2. Composite interpreters should be able to iterate through the entire list irregardless of the internal file hierarchy.
3. Composite interpreters should be able to access a file directly from the key provided during `GetNextFileInfo`.
4. Only files are ever returned from a composite file, directory entries should be ignored. Composite files within composite files should be returned as files and let core search engine decide that it is actually a container.

When `GetNextFileInfo` has reached the end of the file list it should return `CINT_S_NOMOREFILES` which is defined as:

```
const HRESULT CINT_S_NOMOREFILES = MAKE_HRESULT( SEVERITY_SUCCESS,  
FACILITY_ITF, 0x0101);
```

### 1.4.2.4 Open

Opens the composite file.

[C++]

```
HRESULT Open(  
    BSTR bstrCompositeFilePath  
);
```

## Parameters

*bstrCompositeFilePath*

[in] Full path for the composite file.

## Return Value

S\_OK

File was successfully opened.

E\_FAIL

An error occurred.

## Remarks

### 1.4.2.5 SetFilePos

Sets which file the iterator points to.

```
[C++]  
  
HRESULT SetFilePos(  
    ULONG nFilePos  
);
```

#### Parameters

*nFilePos*  
[in] Index of the file to set the iterator to (zero-based).

#### Return Value

S\_OK  
Iterator was successfully positioned.  
E\_FAIL  
An error occurred.

#### Remarks

Currently only used to set file pos to beginning, i.e. always passes 0.

### 1.4.3 IExtCompositeInterpreter2

Optional interface that if implemented provides the composite interpreter with the original file name of the composite file.

#### Methods

SetContainerFileName	Sets the original file name of the composite file.
----------------------	--

#### Remarks

#### 1.4.3.1 SetContainerFileName

Sets the original file name of the composite file.

```
[C++]  
  
HRESULT SetContainerFileName(  
    BSTR bstrFileName  
);
```

**Parameters***bstrFileName*

[in] Original file name of the composite file.

**Return Value**

S\_OK

File name was successfully set.

E\_FAIL

An error occurred.

**Remarks**

Since the composite file may be located in a temporary location for opening (e.g. composite file within another composite file) the `IExtCompositeInterpreter::Open` method may not necessarily provide the original file name of the composite file. This interface is available if the composite interpreter needs to know the original file name.

Only available in version 4.0 or higher.

**1.4.4 IExtInitialize**

Optional interface that if implemented is called to Initialize and Uninitialize the extension.

**Methods**

Initialize	Initializes the extension.
Uninitialize	Uninitializes the extension.

**1.4.4.1 Initialize**

Initializes the extension.

[C++]

```
HRESULT Initialize(
    BSTR * pbstrErr
);
```

**Parameters***pbstrErr*

[out] Error information from initialization.

**Return Value**

S\_OK

Extension was successfully initialized.  
E\_FAIL  
An error occurred.

#### Remarks

#### 1.4.4.2 Uninitialize

Uninitializes the extension.

[C++]

```
HRESULT Uninitialize();
```

#### Return Value

S\_OK  
Extension was successfully uninitialized.  
E\_FAIL  
An error occurred.

#### Remarks

### 1.4.5 IExtTextConverter

Converts a given file into plain text for easy searching.

#### Methods

ConvertFileToText	Converts the given file into a text file.
-------------------	---

#### 1.4.5.1 ConvertFileToText

Converts the given file into a text file.

[C++]

```
HRESULT ConvertFileToText(  
    BSTR bstrPathName,  
    BSTR bstrConvertToName  
);
```

#### Parameters

*bstrPathName*  
[in] Full path for the file to be converted.

*bstrConvertToName*

[in] Full path for the temporary file that the converted text should be written to.

#### **Return Value**

S\_OK

File was successfully converted.

E\_FAIL

An error occurred.

#### **Remarks**

The temporary file to write the text to will have already been created to ensure that it is unique. Either overwrite the file or open the file and append to it.

# Index

## - A -

- Active property
  - ExtensionPlugIn class 30
  - ScriptingCriteria class 45
- After property
  - DateRange class 17
- Archive property
  - FileAttributes class 39

## - B -

- Before property
  - DateRange class 17

## - C -

- Cancel method
  - SearchEngine class 70
- Clear method
  - DateRange class 16
  - FileAttributes class 42
  - SizeRange class 88
- Close method
  - IExtCompositeInterpreter interface 106
- Compressed property
  - FileAttributes class 39
- ContainingText property
  - SearchCriteria class 60
- ContainingTextScript property
  - SearchCriteria class 60
- ContentsExprType property
  - SearchCriteria class 61
- ConvertFileToText method
  - IExtTextConverter interface 111
- Count property
  - ExtensionPlugInList class 35
  - SearchResultItemList class 81
  - TextLineList class 95
  - TextMatchList class 99
- CustomParameter property
  - ScriptingCriteria class 46

## - D -

- DateRange class
  - about DateRange class 15
  - all members 16
  - methods 16
  - properties 16
- DisplayName property
  - ExtensionPlugIn class 31

## - E -

- Encrypted property
  - FileAttributes class 39
- EOLMac property
  - SearchConfiguration class 48
- EOLUnix property
  - SearchConfiguration class 49
- ExcludeFilename property
  - SearchCriteria class 61
- Export method
  - SearchResultItemList class 83
- ExportContents property
  - ExportCriteria class 26
- ExportCriteria class
  - about ExportCriteria class 25
  - all members 25
  - properties 25, 26
- ExportCriteria property
  - SearchResultItemList class 82
- ExportFormat property
  - ExportCriteria class 26
- ExportFormatType enumeration 27
- ExportSurroundingLine property
  - ExportCriteria class 27
- ExportToFile method
  - SearchResultItemList class 82
- ExpressionType enumeration 28
- Extension configuration file 104
- Extension interfaces 104
- ExtensionPlugIn class
  - about ExtensionPlugIn class 29
  - all members 29
  - properties 29, 30
- ExtensionPlugInList class
  - about ExtensionPlugInList class 33

ExtensionPlugInList class  
 all members 34  
 methods 34, 35  
 properties 34

ExtensionPlugInList property  
 SearchConfiguration class 49

ExtensionPlugInType enumeration 36

ExtensionType property  
 ExtensionPlugIn class 31

External interfaces 104

ExtractFile method  
 IExtCompositeInterpreter interface 106

## - F -

FileAttributes class  
 about FileAttributes class 36  
 all members 37  
 methods 37, 42  
 properties 37, 38

FileAttributes property  
 SearchCriteria class 62

FileName property  
 ScriptingCriteria class 46  
 SearchCriteria class 62  
 SearchResultItem class 75

FileNameExprType property  
 SearchCriteria class 62

FileNameScript property  
 SearchCriteria class 63

FileSize property  
 SearchCriteria class 63

FileTypes property  
 ExtensionPlugIn class 31

Find method  
 ExtensionPlugInList class 35

Folder property  
 FileAttributes class 40

## - G -

GetNextFileInfo method  
 IExtCompositeInterpreter interface 107

GreaterThan Property  
 SizeRange class 89

## - H -

Hidden property  
 FileAttributes class 40

## - I -

IExtCompositeInterpreter interface  
 about IExtCompositeInterpreter interface 105

IExtCompositeInterpreter2 interface  
 about IExtCompositeInterpreter2 interface 109

IExtInitialize interface  
 about IExtInitialize interface 110

IExtTextConverter interface  
 about IExtTextConverter interface 111

Initialize method  
 IExtInitialize interface 110

IsFolder property  
 SearchResultItem class 75

Item property  
 ExtensionPlugInList class 34  
 SearchResultItemList class 81  
 TextLineList class 95  
 TextMatchList class 100

## - L -

Length property  
 TextMatch class 97

LessThan property  
 SizeRange class 89

LineNumber property  
 TextLine class 91

LinesAfter property  
 TextLine class 92

LinesBefore property  
 TextLine class 92

Load method  
 SearchCriteria class 57

LoadAll method  
 SearchCriteria class 58

LookIn property  
 SearchCriteria class 63

LookInExprType property  
 SearchCriteria class 63

**- M -**

MatchContentsCase property  
 SearchCriteria class 64

MatchFilenameCase property  
 SearchCriteria class 64

ModifiedDate property  
 SearchCriteria class 64  
 SearchResultItem class 76

**- O -**

Offline property  
 FileAttributes class 40

OnFileFound event  
 SearchEngine class 67

OnProgress event  
 SearchEngine class 68

OnSearchFinish event  
 SearchEngine class 69

OnSearchStart event  
 SearchEngine class 69

Open method  
 IExtCompositeInterpreter interface 108

**- P -**

Path property  
 SearchResultItem class 76

Plugin\_cfg XML configuration file 104

**- R -**

ReadOnly property  
 FileAttributes class 41

RegExpType property  
 SearchCriteria class 65

RegularExpressionType enumeration 43

**- S -**

SafeMode property  
 ExtensionPlugin class 32

Save method  
 SearchCriteria class 59

ScriptEngineProgId property  
 ScriptingCriteria class 46

ScriptingCriteria class  
 about ScriptingCriteria class 44  
 all members 45  
 properties 45

SearchConfiguration class  
 about SearchConfiguration class 46  
 all members 47  
 properties 47, 48

SearchConfiguration property  
 SearchEngine class 71

SearchCriteria class  
 about SearchCriteria class 55  
 all members 56  
 methods 56, 57  
 properties 56, 59

SearchCriteria property  
 SearchEngine class 72

SearchEngine class  
 about SearchEngine class 65  
 all members 66  
 events 66, 67  
 methods 66, 70  
 properties 66, 71

SearchOnePhase property  
 SearchConfiguration class 49

SearchResultItem class  
 about SearchResultItem class 73  
 all members 74  
 properties 74, 75

SearchResultItemList class  
 about SearchResultItemList class 79  
 all members 80  
 methods 80, 82  
 properties 80, 81

SearchSubDirectory property  
 SearchCriteria class 65

SearchThreadCount property  
 SearchConfiguration class 50

SetContainerFileName method  
 IExtCompositeInterpreter2 interface 109

SetFilePos method  
 IExtCompositeInterpreter interface 109

SevenBitChars property  
 SearchConfiguration class 50

Size property  
 SearchResultItem class 76

SizeRange class  
 about SizeRange class 87  
 all members 88  
 methods 88  
 properties 88, 89

Sparse property  
 FileAttributes class 41

Start method  
 SearchEngine class 70

Start property  
 TextMatch class 98

SurroundingLinesAfter property  
 SearchConfiguration class 50

SurroundingLinesBefore property  
 SearchConfiguration class 51

SystemFile property  
 FileAttributes class 41

## - T -

Text property  
 TextLine class 92

TextLine class  
 about TextLine class 89  
 all members 91  
 properties 91

TextLineList class  
 about TextLineList class 93  
 all members 94  
 properties 94, 95

TextLineList property  
 SearchResultItem class 76

TextMatch class  
 about TextMatch class 96  
 all members 97  
 properties 97

TextMatchHighlightList property  
 TextLine class 93

TextMatchList class  
 about TextMatchList class 98  
 all members 99  
 properties 99

TextMatchList property  
 TextLine class 93

TotalFileSize property  
 SearchResultItemList class 82

## - U -

Uninitialize method  
 IExtInitialize interface 111

UniqueName property  
 ExtensionPlugIn class 32

UseFilter property  
 ExtensionPlugIn class 32