How to: Programmatically Search for and Replace Text in Documents

The Find object is a member of both the Selection and the Range objects, and you can use either one to search for text in Microsoft Office Word documents. The replace command is an extension of the find command.

Use a Find object to loop through a Microsoft Office Word document and search for specific text, formatting, or style, and use the P:Microsoft.Office.Interop.Word.Find.Replacement property to replace any of the items found.

Applies to: The information in this topic applies to document-level projects and VSTO add-in projects for Word. For more information, see Features Available by Office Application and Project Type.

Using a Selection Object

When you use a Selection object to find text, any search criteria you specify are applied only against currently selected text. If the Selection is an insertion point, then the document is searched. When the item is found that matches the search criteria, it is automatically selected.

It is important to note that the Find criteria are cumulative, which means that criteria are added to previous search criteria. Clear formatting from previous searches by using the M:Microsoft.Office.Interop.Word.Find.ClearFormatting method prior to the search.

To find text using a Selection object

1. Assign a search string to a variable.

C#

```
object findText = "find me";
```

2. Clear formatting from previous searches.

C#

```
Application.Selection.Find.ClearFormatting();
```

3. Execute the search and display a message box with the results.

C#

```
            if (Application.Selection.Find.Execute(ref findText,
                ref missing, ref missing, ref missing, ref missing, ref missing, ref
missing,
                ref missing, ref missing, ref missing, ref missing, ref missing, ref
missing,
                ref missing, ref missing))
            {
                MessageBox.Show("Text found.");
            }
            else
            {
                MessageBox.Show("The text could not be located.");
            }
```

The following example shows the complete method.

C#

```
        private void SelectionFind()
        {
            object findText = "find me";

            Application.Selection.Find.ClearFormatting();

            if (Application.Selection.Find.Execute(ref findText,
                ref missing, ref missing, ref missing, ref missing, ref missing, ref missing,
                ref missing, ref missing, ref missing, ref missing, ref missing, ref missing,
                ref missing, ref missing))
            {
                MessageBox.Show("Text found.");
            }
            else
            {
                MessageBox.Show("The text could not be located.");
            }
        }
```

Using a Range Object

Using a Range object enables you to search for text without displaying anything in the user interface. The Find object returns **True** if text is found that matches the search criteria, and **False** if it does not. It also redefines the Range object to match the search criteria if the text is found.

To find text using a Range object

1. Define a Range object that consists of the second paragraph in the document.

   The following code example can be used in a document-level customization.

C#

```
            Word.Range rng = this.Paragraphs[2].Range;
```

The following code example can be used in a VSTO Add-in. This example uses the active document.

C#

```
            Word.Document document = this.Application.ActiveDocument;
            Word.Range rng = document.Paragraphs[2].Range;
```

2. Using the P:Microsoft.Office.Interop.Word.Range.Find property of the Range object, first clear any existing formatting options, and then search for the string **find me**.

C#

```
            rng.Find.ClearFormatting();

            if (rng.Find.Execute(ref findText,
                ref missing, ref missing, ref missing, ref missing, ref missing, ref
missing,
                ref missing, ref missing, ref missing, ref missing, ref missing, ref
missing,
                ref missing, ref missing))
            {
```

3. Display the results of the search in a message box, and select the Range to make it visible.

C#

```
                MessageBox.Show("Text found.");
            }
            else
            {
                MessageBox.Show("Text not found.");
            }

            rng.Select();
```

If the search fails, the second paragraph is selected; if it succeeds, the search criteria are displayed.

The following example shows the complete code for a document-level customization. To use this example, run the code from the `ThisDocument` class in your project.

C#

```
        private void RangeFind()
        {
            object findText = "find me";

            Word.Range rng = this.Paragraphs[2].Range;

            rng.Find.ClearFormatting();

            if (rng.Find.Execute(ref findText,
                ref missing, ref missing, ref missing, ref missing, ref missing, ref missing,
                ref missing, ref missing, ref missing, ref missing, ref missing, ref missing,
                ref missing, ref missing))
            {
                MessageBox.Show("Text found.");
            }
            else
            {
                MessageBox.Show("Text not found.");
            }

            rng.Select();
        }
```

The following example shows the complete code for a VSTO Add-in. To use this example, run the code from the `ThisAddIn` class in your project.

C#

```
        private void RangeFind()
        {
            object findText = "find me";

            Word.Document document = this.Application.ActiveDocument;
            Word.Range rng = document.Paragraphs[2].Range;

            rng.Find.ClearFormatting();

            if (rng.Find.Execute(ref findText,
                ref missing, ref missing, ref missing, ref missing, ref missing, ref missing,
                ref missing, ref missing, ref missing, ref missing, ref missing, ref missing,
                ref missing, ref missing))
            {
                MessageBox.Show("Text found.");
            }
            else
            {
                MessageBox.Show("Text not found.");
            }

            rng.Select();
        }
```

Searching For and Replacing Text in Documents

The following code searches the current selection and replaces all of the occurrences of the string **find me** with the string **Found**.

To search for and replace text in documents

1. Add the following example code to the `ThisDocument` or `ThisAddIn` class in your project.

C#

```
        private void SearchReplace()
        {
            Word.Find findObject = Application.Selection.Find;
            findObject.ClearFormatting();
            findObject.Text = "find me";
            findObject.Replacement.ClearFormatting();
            findObject.Replacement.Text = "Found";

            object replaceAll = Word.WdReplace.wdReplaceAll;
            findObject.Execute(ref missing, ref missing, ref missing, ref missing, ref
missing,
                ref missing, ref missing, ref missing, ref missing, ref missing,
                ref replaceAll, ref missing, ref missing, ref missing, ref missing);
        }
```

   The Find class has a M:Microsoft.Office.Interop.Word.Find.ClearFormatting method, and the T:Microsoft.Office.Interop.Word.Replacement class also has its own M:Microsoft.Office.Interop.Word.Replacement.ClearFormatting method. When you are performing find-and-replace operations, you must use the ClearFormatting method of both objects. If you use it only on the Find object, you might get unanticipated results in the replacement text.

2. Use the M:Microsoft.Office.Interop.Word.Find.Execute(System.Object@,System.Object@,System.Object@,System.Object@,System.Object@,System.Object@,System.Object@,System.Object@,System.Object@,System.Object@,System.Object@,System.Object@,System.Object@,System.Object@) method of the Find object to replace each found item. To specify which items to replace, use the *Replace* parameter. This parameter can be one of the following T:Microsoft.Office.Interop.Word.WdReplace values:

   - F:Microsoft.Office.Interop.Word.WdReplace.wdReplaceAll replaces all found items.
   - F:Microsoft.Office.Interop.Word.WdReplace.wdReplaceNone replaces none of the found items.
   - F:Microsoft.Office.Interop.Word.WdReplace.wdReplaceOne replaces the first found item.

See Also

How to: Programmatically Set Search Options in Word
How to: Programmatically Loop Through Found Items in Documents

How to: Programmatically Define and Select Ranges in Documents
How to: Programmatically Restore Selections After Searches
Optional Parameters in Office Solutions