**ActiveDocument:**

**ActiveApplication:**

An extremely high rate of paging active memory to disk is occurring. Your application may be memory-bound.

Gets the active document.

**les évènements, Handlers ; eventdoc and opendoc**

Microsoft Internal Use Only.

This API supports the product infrastructure and is not intended to be used directly from your code.

**comment gérer un évènement?**

Layers word Doc

**ecrire un algorithm, pseudo-code pour modifier les parties d"un texte sur word**

string template, format

**les conbo-box( multi liste déroulante select)**

**uni-code**

**Compiler error:**

https://docs.microsoft.com/en-us/dotnet/csharp/language-reference/compiler-messages/cs0246

**Create new doc:**

this.Application.Documents.Add(@"C:\Test\SampleTemplate.dotx");

**Open existing document:**

this.Application.Documents.Open(@"C:\Test\NewDocument.docx");

**Close the active doc:**

Word.\_Document document = this.Application.ActiveDocument;

document.Close(Word.WdSaveOptions.wdDoNotSaveChanges);

or:(specifying its name)

Word.\_Document doc = Application.Documents["NewDocument.docx"] as Word.\_Document;

doc.Close(Word.WdSaveOptions.wdDoNotSaveChanges);

**save doc:**

this.Application.ActiveDocument.Save();

**Activewindow:**

Returns a Window object that represents the active window.

**Add footer:**

foreach (Word.Section wordSection in this.Application.ActiveDocument.Sections)

{

Word.Range footerRange = wordSection.Footers[Word.WdHeaderFooterIndex.wdHeaderFooterPrimary].Range;

footerRange.Font.ColorIndex = Word.WdColorIndex.wdDarkRed;

footerRange.Font.Size = 20;

footerRange.Text = "Confidential";

}

**Add header:**

foreach (Word.Section section in this.Application.ActiveDocument.Sections)

{

Word.Range headerRange = section.Headers[Word.WdHeaderFooterIndex.wdHeaderFooterPrimary].Range;

headerRange.Fields.Add(headerRange, Word.WdFieldType.wdFieldPage);

headerRange.ParagraphFormat.Alignment = Word.WdParagraphAlignment.wdAlignParagraphRight;

}

**Insert text in a range:**

Word.Range rng = this.Application.ActiveDocument.Range(0, 0);

rng.Text = "New Text";

rng.Select();

**Replace text in a range:**

Word.Range rng = this.Application.ActiveDocument.Range(0, 12);

rng.Text = "New Text";

rng.Select();

**Insert text using typetext:**

private void SelectionInsertText()

{

Word.Selection currentSelection = Application.Selection;

// Store the user's current Overtype selection

bool userOvertype = Application.Options.Overtype;

// Make sure Overtype is turned off.

if (Application.Options.Overtype)

{

Application.Options.Overtype = false;

}

// Test to see if selection is an insertion point.

if (currentSelection.Type == Word.WdSelectionType.wdSelectionIP)

{

currentSelection.TypeText("Inserting at insertion point. ");

currentSelection.TypeParagraph();

}

else

if (currentSelection.Type == Word.WdSelectionType.wdSelectionNormal)

{

// Move to start of selection.

if (Application.Options.ReplaceSelection)

{

object direction = Word.WdCollapseDirection.wdCollapseStart;

currentSelection.Collapse(ref direction);

}

currentSelection.TypeText("Inserting before a text block. ");

currentSelection.TypeParagraph();

}

else

{

// Do nothing.

}

// Restore the user's Overtype selection

Application.Options.Overtype = userOvertype;

}

**N.B:**

A partial definition of the ThisDocument class, which represents the programming model of the document and provides access to the object model of Word.

For more information, see Document host item and Word object model overview.

The remainder of the ThisDocument class is defined in a hidden code file that you should not modify.

Add a paragraph:

//Replace the ThisDocument\_Startup event handler with the following code. When the document is opened, this code adds a second paragraph of text to the document.

private void ThisDocument\_Startup(object sender, System.EventArgs e)

{

this.Paragraphs[1].Range.InsertParagraphAfter();

this.Paragraphs[2].Range.Text = "This text was added by using code.";

}

The ThisDocument\_Startup and ThisDocument\_Shutdown event handlers. These event handlers are called when the document is opened and closed.

Use these event handlers to initialize your customization when the document is opened, and to clean up resources used by your customization when the document

is closed. For more information, see Events in Office projects.