

XDoc.PDF Developer Guide – Annotation Module

Table of Contents

XDoc.PDF Developer Guide – Annotation Module	1
Add Annotation.....	3
Strikethrough the selected text	3
Underline the selected text	4
Highlight the selected text.....	6
Insert text at cursor.....	7
Add note to replace text	8
Draw paths and shapes.....	9
Attach file.....	13
Sticky note.....	14
Textbox.....	15
Text callout.....	16
Text annotation.....	17
Link annotation	18
Retrieve Annotation Information	19
Retrieve all annotations information from a PDFDocument.....	19
Retrieve all annotations information from a PDF page	20
Delete text.....	21
Underline text	22
Highlight text.....	23
Insert text.....	24
Update text	25
Shapes	26
Attachment file	30
Sticky notes	31

Textbox.....	32
Text callout.....	33
Text	34
Retrieve popup window's properties	35
Delete Annotation.....	36
Delete all annotation in a page.....	36
Annotation Flags	37
Retrieve annotation flags.....	37
Set annotation flags	38
Stamp Annotation.....	39
Prepare a stamp source file (PDF file).....	39
Create a new stamp annotation template from an exist PDF file	41
Create a new stamp annotation template by using PDF Context.....	41
Create a stamp annotation template with dynamic fields	42
Add a new stamp annotation template.....	43
Add a stamp annotation to page	44
Get all embedded stamp template names in the SDK.....	44
Get preview (Btmap) of an embedded stamp template.....	45
Add a stamp annotation by using embedded stamp template	46
Annotation Data File (.fdf, .xfdf) Import & Export	48
Export annotations from a PDF document to an FDF file	48
Export annotations from a specified page to an FDF file.....	48
Import an FDF file to a PDF document.....	48
Others	49
Move all annotations in a document without update other properties	49
Update subject, content, artist and modified datetime values of an annotation.....	50
Update popup windows of an annotation.....	51

Add Annotation

Strikethrough the selected text

C#
<pre>String inputFilePath = Program.RootPath + "\\\" + "2.pdf"; String outputFilePath = Program.RootPath + "\\\" + "Annot_1.pdf"; // open a PDF file PDFDocument doc = new PDFDocument(inputFilePath); // get the 2nd page PDFPage page = (PDFPage)doc.GetPage(1); // create the annotation PDFAnnotDeleteLine annot = new PDFAnnotDeleteLine(); annot.StartPoint = new PointF(100F, 200F); annot.EndPoint = new PointF(300F, 400F); // add annotation to the page PDFAnnotHandler.AddAnnotation(page, annot); // save to a new file doc.Save(outputFilePath);</pre>
VB
<pre>Dim inputFilePath As String = Program.RootPath + "\\\" + "2.pdf" Dim outputFilePath As String = Program.RootPath + "\\\" + "Annot_1.pdf" ' open a PDF file Dim doc As PDFDocument = New PDFDocument(inputFilePath) ' get the 2nd page Dim page As PDFPage = doc.GetPage(1) ' create the annotation Dim annot As PDFAnnotDeleteline = New PDFAnnotDeleteline() With annot .StartPoint = New PointF(100.0F, 200.0F) .EndPoint = New PointF(300.0F, 400.0F) End With ' add annotation to the page PDFAnnotHandler.AddAnnotation(page, annot)</pre>

```
' save to a new file
doc.Save(outputFilePath)
```

Underline the selected text

C#

```
String inputFilePath = Program.RootPath + "\\\" + "2.pdf";
String outputFilePath = Program.RootPath + "\\\" + "Annot_2.pdf";

// open a PDF file
PDFDocument doc = new PDFDocument(inputFilePath);
// get the 2nd page
PDFPage page = (PDFPage)doc.GetPage(1);

// create the annotation
PDFAnnotUnderLine annot = new PDFAnnotUnderLine();

annot.StartPoint = new PointF(100F, 200F);
annot.EndPoint = new PointF(300F, 400F);

// add annotation to the page
PDFAnnotHandler.AddAnnotation(page, annot);

// save to a new file
doc.Save(outputFilePath);
```

VB

```
Dim inputFilePath As String = Program.RootPath + "\\\" + "2.pdf"
Dim outputFilePath As String = Program.RootPath + "\\\" + "Annot_2.pdf"

' open a PDF file
Dim doc As PDFDocument = New PDFDocument(inputFilePath)
' get the 2nd page
Dim page As PDFPage = doc.GetPage(1)

' create the annotation
Dim annot As PDFAnnotUnderLine = New PDFAnnotUnderLine()

With annot
    .StartPoint = New PointF(100.0F, 200.0F)
    .EndPoint = New PointF(300.0F, 400.0F)
End With

' add annotation to the page
PDFAnnotHandler.AddAnnotation(page, annot)

' save to a new file
doc.Save(outputFilePath)
```


Highlight the selected text

C#

```
String inputFilePath = Program.RootPath + "\\\" + \"2.pdf\";
String outputFilePath = Program.RootPath + "\\\" + \"Annot_3.pdf\";

// open a PDF file
PDFDocument doc = new PDFDocument(inputFilePath);
// get the 2nd page
PDFPage page = (PDFPage)doc.GetPage(1);

// create the annotation
PDFAnnotHighlight annot = new PDFAnnotHighlight();

annot.StartPoint = new PointF(100F, 200F);
annot.EndPoint = new PointF(300F, 400F);

// add annotation to the page
page.AddPDFAnnot(annot);

// save to a new file
doc.Save(outputFilePath);
```

VB

```
Dim inputFilePath As String = Program.RootPath + "\\\" + \"2.pdf\"
Dim outputFilePath As String = Program.RootPath + "\\\" + \"Annot_3.pdf\"

' open a PDF file
Dim doc As PDFDocument = New PDFDocument(inputFilePath)
' get the 2nd page
Dim page As PDFPage = doc.GetPage(1)

' create the annotation
Dim annot As PDFAnnotHighlight = New PDFAnnotHighlight()

annot.StartPoint = New PointF(100.0F, 200.0F)
annot.EndPoint = New PointF(300.0F, 400.0F)

' add annotation to the page
page.AddPDFAnnot(annot)

' save to a new file
doc.Save(outputFilePath)
```

Insert text at cursor

C#

```
String inputFilePath = Program.RootPath + "\\\" + "2.pdf";
String outputFilePath = Program.RootPath + "\\\" + "Annot_4.pdf";

// open a PDF file
PDFDocument doc = new PDFDocument(inputFilePath);
// get the 2nd page
PDFPage page = (PDFPage)doc.GetPage(1);

// create the annotation
PDFAnnotTextInsert annot = new PDFAnnotTextInsert();

annot.Position = new PointF(300F, 500F);

// add annotation to the page
PDFAnnotHandler.AddAnnotation(page, annot);

// save to a new file
doc.Save(outputFilePath);
```

VB

```
Dim inputFilePath As String = Program.RootPath + "\\\" + "2.pdf"
Dim outputFilePath As String = Program.RootPath + "\\\" + "Annot_3.pdf"

' open a PDF file
Dim doc As PDFDocument = New PDFDocument(inputFilePath)
' get the 2nd page
Dim page As PDFPage = doc.GetPage(1)

' create the annotation
Dim annot As PDFAnnotHighlight = New PDFAnnotHighlight()

annot.StartPoint = New PointF(100.0F, 200.0F)
annot.EndPoint = New PointF(300.0F, 400.0F)

' add annotation to the page
page.AddPDFAnnot(annot)

' save to a new file
doc.Save(outputFilePath)
```

Add note to replace text

C#

```
String inputFilePath = Program.RootPath + "\\\" + \"2.pdf\";
String outputFilePath = Program.RootPath + "\\\" + \"Annot_5.pdf\";

// open a PDF file
PDFDocument doc = new PDFDocument(inputFilePath);
// get the 2nd page
PDFPage page = (PDFPage)doc.GetPage(1);

// create the annotation
PDFAnnotTextReplace annot = new PDFAnnotTextReplace();

annot.StartPoint = new PointF(100F, 200F);
annot.EndPoint = new PointF(300F, 400F);

// add annotation to the page
PDFAnnotHandler.AddAnnotation(page, annot);

// save to a new file
doc.Save(outputFilePath);
```

VB

```
Dim inputFilePath As String = Program.RootPath + "\\\" + \"2.pdf\"
Dim outputFilePath As String = Program.RootPath + "\\\" + \"Annot_5.pdf\"

' open a PDF file
Dim doc As PDFDocument = New PDFDocument(inputFilePath)
' get the 2nd page
Dim page As PDFPage = doc.GetPage(1)

' create the annotation
Dim annot As PDFAnnotTextReplace = New PDFAnnotTextReplace()

With annot
    .StartPoint = New PointF(100.0F, 200.0F)
    .EndPoint = New PointF(300.0F, 400.0F)
End With

' add annotation to the page
PDFAnnotHandler.AddAnnotation(page, annot)

' save to a new file
doc.Save(outputFilePath)
```


Draw paths and shapes

C#

```
String inputFilePath = Program.RootPath + "\\\" + \"2.pdf\";
String outputFilePath = Program.RootPath + "\\\" + \"Annot_6.pdf\";

// open a PDF file
PDFDocument doc = new PDFDocument(inputFilePath);
// get the 1st page
PDFPage page = (PDFPage)doc.GetPage(0);

{
    // create the annotation
    PDFAnnotLine annot = new PDFAnnotLine();

    annot.Start = new PointF(100F, 100F);
    annot.End = new PointF(200F, 200F);

    // add annotation to the page
    PDFAnnotHandler.AddAnnotation(page, annot);
}

{
    // create the annotation
    PDFAnnotArrow annot = new PDFAnnotArrow();

    annot.Start = new PointF(300F, 100F);
    annot.End = new PointF(400F, 150F);

    // add annotation to the page
    PDFAnnotHandler.AddAnnotation(page, annot);
}

{
    // create the annotation
    PDFAnnotEllipse annot = new PDFAnnotEllipse();

    annot.Location = new PointF(300, 300);
    annot.Width = 50F;
    annot.Height = 50F;

    // add annotation to the page
    PDFAnnotHandler.AddAnnotation(page, annot);
}

{
    // create the annotation
    PDFAnnotRectangle annot = new PDFAnnotRectangle();

    annot.Location = new PointF(400, 400);
    annot.Width = 50F;
    annot.Height = 50F;

    // add annotation to the page
```

```

PDFAnnotHandler.AddAnnotation(page, annot);
}
{
    // create the annotation
    PDFAnnotPolygon annot = new PDFAnnotPolygon();

    annot.Points = new PointF[5] {
        new PointF(50F, 450F),
        new PointF(100F, 450F),
        new PointF(120F, 480F),
        new PointF(100F, 500F),
        new PointF(50F, 500F)
    };

    // add annotation to the page
    PDFAnnotHandler.AddAnnotation(page, annot);
}
{
    // create the annotation
    PDFAnnotPolyline annot = new PDFAnnotPolyline();

    annot.Points = new PointF[] {
        new PointF(250F, 450F),
        new PointF(300F, 450F),
        new PointF(320F, 480F),
        new PointF(300F, 500F),
        new PointF(250F, 500F)
    };

    // add annotation to the page
    PDFAnnotHandler.AddAnnotation(page, annot);
}

// save to a new file
doc.Save(outputFilePath);

```

VB

```

Dim inputFilePath As String = Program.RootPath + "\\\" + "2.pdf"
Dim outputFilePath As String = Program.RootPath + "\\\" + "Annot_6.pdf"

' open a PDF file
Dim doc As PDFDocument = New PDFDocument(inputFilePath)
' get the 1st page
Dim page As PDFPage = doc.GetPage(0)

' create the annotation
Dim annot0 As PDFAnnotLine = New PDFAnnotLine()

annot0.Start = New PointF(100.0F, 100.0F)
annot0.End = New PointF(200.0F, 200.0F)

' add annotation to the page
PDFAnnotHandler.AddAnnotation(page, annot0)

```

```

' create the annotation
Dim annot1 As PDFAnnotArrow = New PDFAnnotArrow()

annot1.Start = New PointF(300.0F, 100.0F)
annot1.End = New PointF(400.0F, 150.0F)

' add annotation to the page
PDFAnnotHandler.AddAnnotation(page, annot1)

' create the annotation
Dim annot2 As PDFAnnotEllipse = New PDFAnnotEllipse()

With annot2
    .Location = New PointF(300, 300)
    .Width = 50.0F
    .Height = 50.0F
End With

' add annotation to the page
PDFAnnotHandler.AddAnnotation(page, annot2)

' create the annotation
Dim annot3 As PDFAnnotRectangle = New PDFAnnotRectangle()

With annot3
    .Location = New PointF(400, 400)
    .Width = 50.0F
    .Height = 50.0F
End With

' add annotation to the page
PDFAnnotHandler.AddAnnotation(page, annot3)

' create the annotation
Dim annot4 As PDFAnnotPolygon = New PDFAnnotPolygon()

Dim elem0(4) As PointF
annot4.Points = elem0
With annot4
    .Points(0) = New PointF(50.0F, 450.0F)
    .Points(1) = New PointF(100.0F, 450.0F)
    .Points(2) = New PointF(120.0F, 480.0F)
    .Points(3) = New PointF(100.0F, 500.0F)
    .Points(4) = New PointF(50.0F, 500.0F)
End With

' add annotation to the page
PDFAnnotHandler.AddAnnotation(page, annot4)

' create the annotation
Dim annot5 As PDFAnnotPolyline = New PDFAnnotPolyline()

```

```
Dim elem1(4) As PointF
annot5.Points = elem1
With annot5
    .Points(0) = New PointF(250.0F, 450.0F)
    .Points(1) = New PointF(300.0F, 450.0F)
    .Points(2) = New PointF(320.0F, 480.0F)
    .Points(3) = New PointF(300.0F, 500.0F)
    .Points(4) = New PointF(250.0F, 500.0F)
End With

' add annotation to the page
PDFAnnotHandler.AddAnnotation(page, annot5)

' save to a new file
doc.Save(outputFilePath)
```

Attach file

C#

```
String inputFilePath = Program.RootPath + "\\\" + \"2.pdf\";
String outputFilePath = Program.RootPath + "\\\" + \"Annot_7.pdf\";

// open a PDF file
PDFDocument doc = new PDFDocument(inputFilePath);
// get the 1st page
PDFPage page = (PDFPage)doc.GetPage(0);

{
    // create the annotation
    PDFAnnotFileAttach annot = new PDFAnnotFileAttach();

    annot.Position = new PointF(100F, 100F);
    annot.FilePath = @\"C:\AttachmentFile.txt\";

    // add annotation to the page
    PDFAnnotHandler.AddAnnotation(page, annot);
}

// save to a new file
doc.Save(outputFilePath);
```

VB

```
Dim inputFilePath As String = Program.RootPath + "\\\" + \"2.pdf\"
Dim outputFilePath As String = Program.RootPath + "\\\" + \"Annot_7.pdf\"

' open a PDF file
Dim doc As PDFDocument = New PDFDocument(inputFilePath)
' get the 1st page
Dim page As PDFPage = doc.GetPage(0)

' create the annotation
Dim annot As PDFAnnotFileAttach = New PDFAnnotFileAttach()

annot.Position = New PointF(100.0F, 100.0F)
annot.FilePath = \"C:\AttachmentFile.txt\"

' add annotation to the page
PDFAnnotHandler.AddAnnotation(page, annot)

' save to a new file
doc.Save(outputFilePath)
```

Sticky note

C#

```
String inputFilePath = Program.RootPath + "\\\" + \"2.pdf\";
String outputFilePath = Program.RootPath + "\\\" + \"Annot_8.pdf\";

// open a PDF file
PDFDocument doc = new PDFDocument(inputFilePath);
// get the 1st page
PDFPage page = (PDFPage)doc.GetPage(0);

// create the annotation
PDFAnnotStickyNote annot = new PDFAnnotStickyNote();

annot.Position = new PointF(100F, 100F);

// add annotation to the page
PDFAnnotHandler.AddAnnotation(page, annot);

// save to a new file
doc.Save(outputFilePath);
```

VB

```
Dim inputFilePath As String = Program.RootPath + "\\\" + \"2.pdf\"
Dim outputFilePath As String = Program.RootPath + "\\\" + \"Annot_8.pdf\"

' open a PDF file
Dim doc As PDFDocument = New PDFDocument(inputFilePath)
' get the 1st page
Dim page As PDFPage = doc.GetPage(0)

' create the annotation
Dim annot As PDFAnnotStickyNote = New PDFAnnotStickyNote()

annot.Position = New PointF(100.0F, 100.0F)

' add annotation to the page
PDFAnnotHandler.AddAnnotation(page, annot)

' save to a new file
doc.Save(outputFilePath)
```

Textbox

C#
<pre>String inputFilePath = Program.RootPath + "\\\" + \"2.pdf\"; String outputFilePath = Program.RootPath + "\\\" + \"Annot_9.pdf\"; // open a PDF file PDFDocument doc = new PDFDocument(inputFilePath); // get the 1st page PDFPage page = (PDFPage)doc.GetPage(0); // create the annotation PDFAnnotTextBox annot = new PDFAnnotTextBox(); annot.Boundary = new RectangleF(100F, 100F, 400F, 300F); // add annotation to the page PDFAnnotHandler.AddAnnotation(page, annot); // save to a new file doc.Save(outputFilePath);</pre>
VB
<pre>Dim inputFilePath As String = Program.RootPath + "\\\" + \"2.pdf\" Dim outputFilePath As String = Program.RootPath + "\\\" + \"Annot_9.pdf\" ' open a PDF file Dim doc As PDFDocument = New PDFDocument(inputFilePath) ' get the 1st page Dim page As PDFPage = doc.GetPage(0) ' create the annotation Dim annot As PDFAnnotTextBox = New PDFAnnotTextBox() annot.Boundary = New RectangleF(100.0F, 100.0F, 400.0F, 300.0F) ' add annotation to the page PDFAnnotHandler.AddAnnotation(page, annot) ' save to a new file doc.Save(outputFilePath)</pre>

Text callout

C#

```
String inputFilePath = Program.RootPath + "\\\" + \"2.pdf\";
String outputFilePath = Program.RootPath + "\\\" + \"Annot_10.pdf\";

// open a PDF file
PDFDocument doc = new PDFDocument(inputFilePath);
// get the 1st page
PDFPage page = (PDFPage)doc.GetPage(0);

// create the annotation
PDFAnnotTextCallout annot = new PDFAnnotTextCallout();

annot.StartPoint = new PointF(100F, 100F);
annot.Boundary = new RectangleF(400F, 500F, 300F, 80F);

annot.Content = @\"This is a text callout annotation\";

// add annotation to the page
PDFAnnotHandler.AddAnnotation(page, annot);

// save to a new file
doc.Save(outputFilePath);
```

VB

```
Dim inputFilePath As String = Program.RootPath + "\\\" + \"2.pdf\"
Dim outputFilePath As String = Program.RootPath + "\\\" + \"Annot_10.pdf\"

' open a PDF file
Dim doc As PDFDocument = New PDFDocument(inputFilePath)
' get the 1st page
Dim page As PDFPage = doc.GetPage(0)

' create the annotation
Dim annot As PDFAnnotTextCallout = New PDFAnnotTextCallout()

annot.StartPoint = New PointF(100.0F, 100.0F)
annot.Boundary = New RectangleF(400.0F, 500.0F, 300.0F, 80.0F)

annot.Content = \"This is a text callout annotation\"

' add annotation to the page
PDFAnnotHandler.AddAnnotation(page, annot)

' save to a new file
doc.Save(outputFilePath)
```


Text annotation

C#

```
String inputFilePath = Program.RootPath + "\\\" + \"2.pdf\";
String outputFilePath = Program.RootPath + "\\\" + \"Annot_11.pdf\";

// open a PDF file
PDFDocument doc = new PDFDocument(inputFilePath);
// get the 1st page
PDFPage page = (PDFPage)doc.GetPage(0);

// create the annotation
PDFAnnotText annot = new PDFAnnotText();

annot.Boundary = new RectangleF(400F, 500F, 300F, 80F);

annot.Content = @\"This is a text annotation\";

// add annotation to the page
PDFAnnotHandler.AddAnnotation(page, annot);

// save to a new file
doc.Save(outputFilePath);
```

VB

```
Dim inputFilePath As String = Program.RootPath + "\\\" + \"2.pdf\"
Dim outputFilePath As String = Program.RootPath + "\\\" + \"Annot_11.pdf\"

' open a PDF file
Dim doc As PDFDocument = New PDFDocument(inputFilePath)
' get the 1st page
Dim page As PDFPage = doc.GetPage(0)

' create the annotation
Dim annot As PDFAnnotText = New PDFAnnotText()

annot.Boundary = New RectangleF(400.0F, 500.0F, 300.0F, 80.0F)

annot.Content = \"This is a text annotation\"

' add annotation to the page
PDFAnnotHandler.AddAnnotation(page, annot)

' save to a new file
doc.Save(outputFilePath)
```

Link annotation

C#

```
String inputFilePath = Program.RootPath + "\\\" + \"1.pdf\";
String outputFilePath = Program.RootPath + "\\\" + \"Output.pdf\";

// open a PDF file
PDFDocument doc = new PDFDocument(inputFilePath);
// get the 1st page
PDFPage page = (PDFPage)doc.GetPage(0);

// set a rectangle area start from Point (0, 0) with width 400, height 200 pixels (in 96 dpi)
RectangleF linkAnnotBoundary = new RectangleF(0, 0, 400, 200);
// set URL value
String uri = @\"http://www.google.com\";

// create the annotation
PDFAnnotLink annot = new PDFAnnotLink();
annot.SetBoundary(linkAnnotBoundary);
annot.Uri = uri;

// add annotation to the page
page.AddPDFAnnot(annot);

// save to a new file
doc.Save(outputFilePath);
```

VB

```
Dim inputFilePath As String = Program.RootPath + "\\\" + \"1.pdf\"
Dim outputFilePath As String = Program.RootPath + "\\\" + \"Output.pdf\"

' open a PDF file
Dim doc As PDFDocument = New PDFDocument(inputFilePath)
' get the 1st page
Dim page As PDFPage = doc.GetPage(0)

' set a rectangle area start from Point (0, 0) with width 400, height 200 pixels (in 96 dpi)
Dim linkAnnotBoundary As RectangleF = New RectangleF(0, 0, 400, 200)
' set URL value
Dim uri As String = \"http://www.google.com\"

' create the annotation
Dim annot As PDFAnnotLink = New PDFAnnotLink()
annot.SetBoundary(linkAnnotBoundary)
annot.Uri = uri

' add annotation to the page
page.AddPDFAnnot(annot)

' save to a new file
doc.Save(outputFilePath)
```

Retrieve Annotation Information

Retrieve all annotations information from a PDFDocument

C#

```
String inputFilePath = Program.RootPath + "\\\" + "1_Annots.pdf";

PDFDocument doc = new PDFDocument(inputFilePath);
List<IPDFAnnot> annots = PDFAnnotHandler.GetAllAnnotations(doc);

Console.WriteLine("Number of Annotations: " + annots.Count);
if (annots.Count > 0)
{
    foreach (IPDFAnnot annot in annots)
    {
        Console.WriteLine("Annotation");
        Console.WriteLine(" Page:  " + annot.PageIndex);
        Console.WriteLine(" Aritst:  " + annot.Artist);
        Console.WriteLine(" Subject: " + annot.Subject);
        Console.WriteLine(" Content: " + annot.Content);
    }
}
```

VB

```
Dim inputFilePath As String = Program.RootPath + "\\\" + "1_Annots.pdf"

Dim doc As PDFDocument = New PDFDocument(inputFilePath)
Dim annots = PDFAnnotHandler.GetAllAnnotations(doc)

Console.WriteLine("Number of Annotations: " + annots.Count)
If annots.Count > 0 Then
    For Each annot As IPDFAnnot In annots
        Console.WriteLine("Annotation")
        Console.WriteLine(" Page:  " + annot.PageIndex)
        Console.WriteLine(" Aritst:  " + annot.Artist)
        Console.WriteLine(" Subject: " + annot.Subject)
        Console.WriteLine(" Content: " + annot.Content)
    Next
End If
```

Retrieve all annotations information from a PDF page

C#
<pre>String inputFilePath = Program.RootPath + "\\\" + "1_Annots.pdf"; PDFDocument doc = new PDFDocument(inputFilePath); PDFPage page = (PDFPage)doc.GetPage(0); List<IPDFAnnot> annots = PDFAnnotHandler.GetAllAnnotations(page);</pre>
VB
<pre>Dim inputFilePath As String = Program.RootPath + "\\\" + "1_Annots.pdf" Dim doc As PDFDocument = New PDFDocument(inputFilePath) Dim page As PDFPage = doc.GetPage(0) Dim annots = PDFAnnotHandler.GetAllAnnotations(page)</pre>

Delete text

C#

```
String inputFilePath = Program.RootPath + "\\\" + "Annot_1.pdf";

PDFDocument doc = new PDFDocument(inputFilePath);
List<IPDFAnnot> annots = PDFAnnotHandler.GetAllAnnotations(doc);
foreach (IPDFAnnot annot in annots)
{
    if (annot is PDFAnnotDeleteLine)
    {
        PDFAnnotDeleteLine obj = (PDFAnnotDeleteLine)annot;
        Console.WriteLine("Color: " + obj.Color.ToString());
        Console.WriteLine("Line Boundaries: ");
        foreach (RectangleF line in obj.GetLineBoundaries())
        {
            Console.WriteLine(" " + line.ToString());
        }
    }
}
```

VB

```
Dim inputFilePath As String = Program.RootPath + "\\\" + "Annot_1.pdf"

Dim doc As PDFDocument = New PDFDocument(inputFilePath)
Dim annots = PDFAnnotHandler.GetAllAnnotations(doc)
For Each annot As IPDFAnnot In annots
    If TypeOf annot Is PDFAnnotDeleteLine Then
        Dim obj As PDFAnnotDeleteLine = annot
        Console.WriteLine("Color: " + obj.Color.ToString())
        Console.WriteLine("Line Boundaries: ")
        For Each line As RectangleF In obj.GetLineBoundaries()
            Console.WriteLine(" " + line.ToString())
        Next
    End If
Next
```

Underline text

C#

```
String inputFilePath = Program.RootPath + "\\\" + "Annot_2.pdf";

PDFDocument doc = new PDFDocument(inputFilePath);
List<IPDFAnnot> annots = PDFAnnotHandler.GetAllAnnotations(doc);
foreach (IPDFAnnot annot in annots)
{
    if (annot is PDFAnnotUnderLine)
    {
        PDFAnnotUnderLine obj = (PDFAnnotUnderLine)annot;
        Console.WriteLine("Color: " + obj.Color.ToString());
        Console.WriteLine("Line Boundaries: ");
        foreach (RectangleF line in obj.GetLineBoundaries())
        {
            Console.WriteLine(" " + line.ToString());
        }
    }
}
```

VB

```
Dim inputFilePath As String = Program.RootPath + "\\\" + "Annot_2.pdf"

Dim doc As PDFDocument = New PDFDocument(inputFilePath)
Dim annots = PDFAnnotHandler.GetAllAnnotations(doc)
For Each annot As IPDFAnnot In annots
    If TypeOf annot Is PDFAnnotUnderLine Then
        Dim obj As PDFAnnotUnderLine = annot
        Console.WriteLine("Color: " + obj.Color.ToString())
        Console.WriteLine("Line Boundaries: ")
        For Each line As RectangleF In obj.GetLineBoundaries()
            Console.WriteLine(" " + line.ToString())
        Next
    End If
Next
```

Highlight text

C#

```
String inputFilePath = Program.RootPath + "\\\" + "Annot_3.pdf";

PDFDocument doc = new PDFDocument(inputFilePath);
List<IPDFAnnot> annots = PDFAnnotHandler.GetAllAnnotations(doc);
foreach (IPDFAnnot annot in annots)
{
    if (annot is PDFAnnotHighlight)
    {
        PDFAnnotHighlight obj = (PDFAnnotHighlight)annot;
        Console.WriteLine("Color: " + obj.Color.ToString());
        Console.WriteLine("Line Boundaries: ");
        foreach (RectangleF line in obj.GetLineBoundaries())
        {
            Console.WriteLine(" " + line.ToString());
        }
    }
}
```

VB

```
Dim inputFilePath As String = Program.RootPath + "\\\" + "Annot_3.pdf"

Dim doc As PDFDocument = New PDFDocument(inputFilePath)
Dim annots = PDFAnnotHandler.GetAllAnnotations(doc)
For Each annot As IPDFAnnot In annots
    If TypeOf annot Is PDFAnnotHighlight Then
        Dim obj As PDFAnnotHighlight = annot
        Console.WriteLine("Color: " + obj.Color.ToString())
        Console.WriteLine("Line Boundaries: ")
        For Each line As RectangleF In obj.GetLineBoundaries()
            Console.WriteLine(" " + line.ToString())
        Next
    End If
Next
```

Insert text

C#

```
String inputFilePath = Program.RootPath + "\\\" + "Annot_4.pdf";

PDFDocument doc = new PDFDocument(inputFilePath);
List<IPDFAnnot> annots = PDFAnnotHandler.GetAllAnnotations(doc);
foreach (IPDFAnnot annot in annots)
{
    if (annot is PDFAnnotTextInsert)
    {
        PDFAnnotTextInsert obj = (PDFAnnotTextInsert)annot;
        Console.WriteLine("Color: " + obj.Color.ToString());
        Console.WriteLine("Insert Position: " + obj.Position.ToString());
    }
}
```

VB

```
Dim inputFilePath As String = Program.RootPath + "\\\" + "Annot_4.pdf"

Dim doc As PDFDocument = New PDFDocument(inputFilePath)
Dim annots = PDFAnnotHandler.GetAllAnnotations(doc)
For Each annot As IPDFAnnot In annots
    If TypeOf annot Is PDFAnnotTextInsert Then
        Dim obj As PDFAnnotTextInsert = annot
        Console.WriteLine("Color: " + obj.Color.ToString())
        Console.WriteLine("Insert Position: " + obj.Position.ToString())
    End If
Next
```


Update text

C#

```
String inputFilePath = Program.RootPath + "\\\" + "Annot_5.pdf";

PDFDocument doc = new PDFDocument(inputFilePath);
List<IPDFAnnot> annots = PDFAnnotHandler.GetAllAnnotations(doc);
foreach (IPDFAnnot annot in annots)
{
    if (annot is PDFAnnotTextReplace)
    {
        PDFAnnotTextReplace obj = (PDFAnnotTextReplace)annot;
        Console.WriteLine("Color: " + obj.Color.ToString());
        Console.WriteLine("Line Boundaries: ");
        foreach (RectangleF line in obj.GetLineBoundaries())
        {
            Console.WriteLine(" " + line.ToString());
        }
    }
}
```

VB

```
Dim inputFilePath As String = Program.RootPath + "\\\" + "Annot_5.pdf"

Dim doc As PDFDocument = New PDFDocument(inputFilePath)
Dim annots = PDFAnnotHandler.GetAllAnnotations(doc)
For Each annot As IPDFAnnot In annots
    If TypeOf annot Is PDFAnnotTextReplace Then
        Dim obj As PDFAnnotTextReplace = annot
        Console.WriteLine("Color: " + obj.Color.ToString())
        Console.WriteLine("Line Boundaries: ")
        For Each line As RectangleF In obj.GetLineBoundaries()
            Console.WriteLine(" " + line.ToString())
        Next
    End If
Next
```

Shapes

```
C#
String inputFilePath = Program.RootPath + "\\\" + "Annot_6.pdf";

PDFDocument doc = new PDFDocument(inputFilePath);
List<IPDFAnnot> annots = PDFAnnotHandler.GetAllAnnotations(doc);
foreach (IPDFAnnot annot in annots)
{
    if (annot is PDFAnnotLine)
    {
        PDFAnnotLine obj = (PDFAnnotLine)annot;
        Console.WriteLine("Line Annotation: ");
        Console.WriteLine("Start Point: " + obj.Start.ToString());
        Console.WriteLine("End Point: " + obj.End.ToString());
        Console.WriteLine("Line Color: " + obj.LineColor.ToString());
        Console.WriteLine("Line Width: " + obj.LineWidth);
    }
    else if (annot is PDFAnnotArrow)
    {
        PDFAnnotArrow obj = (PDFAnnotArrow)annot;
        Console.WriteLine("Arrow Annotation: ");
        Console.WriteLine("Start Point: " + obj.Start.ToString());
        Console.WriteLine("End Point: " + obj.End.ToString());
        Console.WriteLine("Line Color: " + obj.LineColor.ToString());
        Console.WriteLine("Line Width: " + obj.LineWidth);
    }
    else if (annot is PDFAnnotEllipse)
    {
        PDFAnnotEllipse obj = (PDFAnnotEllipse)annot;
        Console.WriteLine("Ellipse Annotation: ");
        Console.WriteLine("Location: " + obj.Location.ToString());
        Console.WriteLine("Width: " + obj.Width);
        Console.WriteLine("Height: " + obj.Height);
        Console.WriteLine("Line Color: " + obj.LineColor.ToString());
        Console.WriteLine("Line Width: " + obj.LineWidth);
        Console.WriteLine("Fill Color: " + obj.FillColor.ToString());
    }
    else if (annot is PDFAnnotRectangle)
    {
        PDFAnnotRectangle obj = (PDFAnnotRectangle)annot;
        Console.WriteLine("Rectangle Annotation: ");
        Console.WriteLine("Location: " + obj.Location.ToString());
        Console.WriteLine("Width: " + obj.Width);
        Console.WriteLine("Height: " + obj.Height);
        Console.WriteLine("Line Color: " + obj.LineColor.ToString());
        Console.WriteLine("Line Width: " + obj.LineWidth);
        Console.WriteLine("Fill Color: " + obj.FillColor.ToString());
    }
    else if (annot is PDFAnnotPolygon)
    {

```

```

    PDFAnnotPolygon obj = (PDFAnnotPolygon)annot;
    Console.WriteLine("Polygon Annotation: ");
    Console.WriteLine("End Points: ");
    foreach (PointF pt in obj.Points)
    {
        Console.WriteLine(" " + pt.ToString());
    }
    Console.WriteLine("Line Color: " + obj.LineColor.ToString());
    Console.WriteLine("Line Width: " + obj.LineWidth);
    Console.WriteLine("Fill Color: " + obj.FillColor.ToString());
}
else if (annot is PDFAnnotPolyline)
{
    PDFAnnotPolyline obj = (PDFAnnotPolyline)annot;
    Console.WriteLine("Polyline Annotation: ");
    Console.WriteLine("End Points: ");
    foreach (PointF pt in obj.Points)
    {
        Console.WriteLine(" " + pt.ToString());
    }
    Console.WriteLine("Line Color: " + obj.LineColor.ToString());
    Console.WriteLine("Line Width: " + obj.LineWidth);
    Console.WriteLine("Fill Color: " + obj.FillColor.ToString());
}
else if (annot is PDFAnnotPen)
{
    PDFAnnotPen obj = (PDFAnnotPen)annot;
    Console.WriteLine("Pen Annotation: ");
    Console.WriteLine("Points: ");
    foreach (PointF pt in obj.Points)
    {
        Console.WriteLine(" " + pt.ToString());
    }
    Console.WriteLine("Line Color: " + obj.LineColor.ToString());
    Console.WriteLine("Line Width: " + obj.LineWidth);
    Console.WriteLine("Path: " + obj.Path.ToString());
}
}
}

```

VB

```
Dim inputFilePath As String = Program.RootPath + "\\\" + "Annot_6.pdf"
```

```
Dim doc As PDFDocument = New PDFDocument(inputFilePath)
```

```
Dim annots = PDFAnnotHandler.GetAllAnnotations(doc)
```

```
For Each annot As IPDFAnnot In annots
```

```
    If TypeOf annot Is PDFAnnotLine Then
```

```
        Dim obj As PDFAnnotLine = annot
```

```
        Console.WriteLine("Line Annotation: ")
```

```
        Console.WriteLine("Start Point: " + obj.Start.ToString())
```

```
        Console.WriteLine("End Point: " + obj.End.ToString())
```

```
        Console.WriteLine("Line Color: " + obj.LineColor.ToString())
```

```
        Console.WriteLine("Line Width: {0}", obj.LineWidth)
```

```
    ElseIf TypeOf annot Is PDFAnnotArrow Then
```

```

Dim obj As PDFAnnotArrow = annot
Console.WriteLine("Arrow Annotation: ")
Console.WriteLine("Start Point: " + obj.Start.ToString())
Console.WriteLine("End Point: " + obj.End.ToString())
Console.WriteLine("Line Color: " + obj.LineColor.ToString())
Console.WriteLine("Line Width: {0}", obj.LineWidth)
ElseIf TypeOf annot Is PDFAnnotEllipse Then
Dim obj As PDFAnnotEllipse = annot
Console.WriteLine("Ellipse Annotation: ")
Console.WriteLine("Location: " + obj.Location.ToString())
Console.WriteLine("Width: {0}", obj.Width)
Console.WriteLine("Height: {0}", obj.Height)
Console.WriteLine("Line Color: " + obj.LineColor.ToString())
Console.WriteLine("Line Width: {0}", obj.LineWidth)
Console.WriteLine("Fill Color: " + obj.FillColor.ToString())
ElseIf TypeOf annot Is PDFAnnotRectangle Then
Dim obj As PDFAnnotRectangle = annot
Console.WriteLine("Rectangle Annotation: ")
Console.WriteLine("Location: " + obj.Location.ToString())
Console.WriteLine("Width: {0}", obj.Width)
Console.WriteLine("Height: {0}", obj.Height)
Console.WriteLine("Line Color: " + obj.LineColor.ToString())
Console.WriteLine("Line Width: {0}", obj.LineWidth)
Console.WriteLine("Fill Color: " + obj.FillColor.ToString())
ElseIf TypeOf annot Is PDFAnnotPolygon Then
Dim obj As PDFAnnotPolygon = annot
Console.WriteLine("Polygon Annotation: ")
Console.WriteLine("End Points: ")
For Each pt As PointF In obj.Points
Console.WriteLine(" " + pt.ToString())
Next
Console.WriteLine("Line Color: " + obj.LineColor.ToString())
Console.WriteLine("Line Width: {0}", obj.LineWidth)
Console.WriteLine("Fill Color: " + obj.FillColor.ToString())
ElseIf TypeOf annot Is PDFAnnotPolyline Then
Dim obj As PDFAnnotPolyline = annot
Console.WriteLine("Polyline Annotation: ")
Console.WriteLine("End Points: ")
For Each pt As PointF In obj.Points
Console.WriteLine(" " + pt.ToString())
Next
Console.WriteLine("Line Color: " + obj.LineColor.ToString())
Console.WriteLine("Line Width: {0}", obj.LineWidth)
Console.WriteLine("Fill Color: " + obj.FillColor.ToString())
ElseIf TypeOf annot Is PDFAnnotPen Then
Dim obj As PDFAnnotPen = annot
Console.WriteLine("Pen Annotation: ")
Console.WriteLine("Points: ")
For Each pt As PointF In obj.Points
Console.WriteLine(" " + pt.ToString())
Next
Console.WriteLine("Line Color: " + obj.LineColor.ToString())
Console.WriteLine("Line Width: {0}", obj.LineWidth)

```

```
    Console.WriteLine("Path: " + obj.Path.ToString())  
End If  
Next
```

Attachment file

C#

```
String inputFilePath = Program.RootPath + "\\\" + \"Annot_7.pdf\";

PDFDocument doc = new PDFDocument(inputFilePath);
List<IPDFAnnot> annots = PDFAnnotHandler.GetAllAnnotations(doc);
foreach (IPDFAnnot annot in annots)
{
    if (annot is PDFAnnotFileAttach)
    {
        PDFAnnotFileAttach obj = (PDFAnnotFileAttach)annot;
        Console.WriteLine(\"Color: \" + obj.Color.ToString());
    }
}
```

VB

```
Dim inputFilePath As String = Program.RootPath + "\\\" + \"Annot_7.pdf\"

Dim doc As PDFDocument = New PDFDocument(inputFilePath)
Dim annots = PDFAnnotHandler.GetAllAnnotations(doc)
For Each annot As IPDFAnnot In annots
    If TypeOf annot Is PDFAnnotFileAttach Then
        Dim obj As PDFAnnotFileAttach = annot
        Console.WriteLine(\"Color: \" + obj.Color.ToString())
    End If
Next
```

Sticky notes

C#

```
String inputFilePath = Program.RootPath + "\\\" + \"Annot_8.pdf\";

PDFDocument doc = new PDFDocument(inputFilePath);
List<IPDFAnnot> annots = PDFAnnotHandler.GetAllAnnotations(doc);
foreach (IPDFAnnot annot in annots)
{
    if (annot is PDFAnnotStickyNote)
    {
        PDFAnnotStickyNote obj = (PDFAnnotStickyNote)annot;
        Console.WriteLine(\"Color: \" + obj.Color.ToString());
    }
}
```

VB

```
Dim inputFilePath As String = Program.RootPath + "\\\" + \"Annot_8.pdf\"

Dim doc As PDFDocument = New PDFDocument(inputFilePath)
Dim annots = PDFAnnotHandler.GetAllAnnotations(doc)
For Each annot As IPDFAnnot In annots
    If TypeOf annot Is PDFAnnotStickyNote Then
        Dim obj As PDFAnnotStickyNote = annot
        Console.WriteLine(\"Color: \" + obj.Color.ToString())
    End If
Next
```

Textbox

C#

```
String inputFilePath = Program.RootPath + "\\\" + "Annot_9.pdf";

PDFDocument doc = new PDFDocument(inputFilePath);
List<IPDFAnnot> annots = PDFAnnotHandler.GetAllAnnotations(doc);
foreach (IPDFAnnot annot in annots)
{
    if (annot is PDFAnnotTextBox)
    {
        PDFAnnotTextBox obj = (PDFAnnotTextBox)annot;
        Console.WriteLine("Textbox Boundary: " + obj.Boundary.ToString());
        Console.WriteLine("Textbox Border Color: " + obj.LineColor.ToString());
        Console.WriteLine("Textbox Border Width: " + obj.LineWidth);
        Console.WriteLine("Textbox Fill Color: " + obj.FillColor.ToString());
    }
}
```

VB

```
Dim inputFilePath As String = Program.RootPath + "\\\" + "Annot_9.pdf"

Dim doc As PDFDocument = New PDFDocument(inputFilePath)
Dim annots = PDFAnnotHandler.GetAllAnnotations(doc)
For Each annot As IPDFAnnot In annots
    If TypeOf annot Is PDFAnnotTextBox Then
        Dim obj As PDFAnnotTextBox = annot
        Console.WriteLine("Textbox Boundary: " + obj.Boundary.ToString())
        Console.WriteLine("Textbox Border Color: " + obj.LineColor.ToString())
        Console.WriteLine("Textbox Border Width: {0}", obj.LineWidth)
        Console.WriteLine("Textbox Fill Color: " + obj.FillColor.ToString())
    End If
Next
```


Text callout

C#

```
String inputFilePath = Program.RootPath + "\\\" + "Annot_10.pdf";

PDFDocument doc = new PDFDocument(inputFilePath);
List<IPDFAnnot> annots = PDFAnnotHandler.GetAllAnnotations(doc);
foreach (IPDFAnnot annot in annots)
{
    if (annot is PDFAnnotTextCallout)
    {
        PDFAnnotTextCallout obj = (PDFAnnotTextCallout)annot;
        Console.WriteLine("Textbox Boundary: " + obj.GetTextBoxBoundary().ToString());
        Console.WriteLine("Textbox Border Color: " + obj.LineColor.ToString());
        Console.WriteLine("Textbox Border Width: " + obj.LineWidth);
        Console.WriteLine("Textbox Fill Color: " + obj.FillColor.ToString());
        Console.WriteLine("Callout Line: ");
        foreach (PointF pt in obj.GetEndPoints())
        {
            Console.WriteLine(" " + pt.ToString());
        }
    }
}
```

VB

```
Dim inputFilePath As String = Program.RootPath + "\\\" + "Annot_10.pdf"

Dim doc As PDFDocument = New PDFDocument(inputFilePath)
Dim annots = PDFAnnotHandler.GetAllAnnotations(doc)
For Each annot As IPDFAnnot In annots
    If TypeOf annot Is PDFAnnotTextCallout Then
        Dim obj As PDFAnnotTextCallout = annot
        Console.WriteLine("Textbox Boundary: " + obj.GetTextBoxBoundary().ToString())
        Console.WriteLine("Textbox Border Color: " + obj.LineColor.ToString())
        Console.WriteLine("Textbox Border Width: {0}", obj.LineWidth)
        Console.WriteLine("Textbox Fill Color: " + obj.FillColor.ToString())
        Console.WriteLine("Callout Line: ")
        For Each pt As PointF In obj.GetEndPoints()
            Console.WriteLine(" " + pt.ToString())
        Next
    End If
Next
```

Text

C#

```
String inputFilePath = Program.RootPath + "\\\" + "Annot_11.pdf";

PDFDocument doc = new PDFDocument(inputFilePath);
List<IPDFAnnot> annots = PDFAnnotHandler.GetAllAnnotations(doc);
foreach (IPDFAnnot annot in annots)
{
    if (annot is PDFAnnotText)
    {
        PDFAnnotText obj = (PDFAnnotText)annot;
        Console.WriteLine("Text Color: " + obj.TextColor.ToString());
        Console.WriteLine("Text Boundary: " + obj.Boundary.ToString());
    }
}
```

VB

```
Dim inputFilePath As String = Program.RootPath + "\\\" + "Annot_11.pdf"

Dim doc As PDFDocument = New PDFDocument(inputFilePath)
Dim annots = PDFAnnotHandler.GetAllAnnotations(doc)
For Each annot As IPDFAnnot In annots
    If TypeOf annot Is PDFAnnotText Then
        Dim obj As PDFAnnotText = annot
        Console.WriteLine("Text Color: " + obj.TextColor.ToString())
        Console.WriteLine("Text Boundary: " + obj.Boundary.ToString())
    End If
Next
```

Retrieve popup window's properties

C#

```
String inputFilePath = Program.RootPath + "\\\" + "Annot_1.pdf";

PDFDocument doc = new PDFDocument(inputFilePath);
List<IPDFAnnot> annots = PDFAnnotHandler.GetAllAnnotations(doc);
foreach (IPDFAnnot annot in annots)
{
    if (annot is IPDFPopupAnnot)
    {
        Console.WriteLine("Annotation has popup window");

        IPDFPopupAnnot obj = (IPDFPopupAnnot)annot;
        Console.WriteLine("Is open in the begin: " + obj.Popup.IsOpen);
        Console.WriteLine("Popup window boundary: " + obj.Popup.Boundary.ToString());
    }
    else
    {
        Console.WriteLine("Annotation has no popup window");
    }
}
```

VB

Delete Annotation

Delete all annotation in a page

C#
<pre>String inputFilePath = Program.RootPath + "\\\" + "1_Annots.pdf"; PDFDocument doc = new PDFDocument(inputFilePath); // get all annotations in the 1st page PDFPage page = (PDFPage)doc.GetPage(0); List<IPDFAnnot> annots = PDFAnnotHandler.GetAllAnnotations(page); // remove all annotations in the 1st page PDFAnnotHandler.DeleteAnnotation(doc, annots);</pre>
VB
<pre>Dim inputFilePath As String = Program.RootPath + "\\\" + "1_Annots.pdf" Dim doc As PDFDocument = New PDFDocument(inputFilePath) ' get all annotations in the 1st page Dim page As PDFPage = doc.GetPage(0) Dim annots = PDFAnnotHandler.GetAllAnnotations(page) ' remove all annotations in the 1st page PDFAnnotHandler.DeleteAnnotation(doc, annots)</pre>

Annotation Flags

Retrieve annotation flags

C#

```
String inputFilePath = Program.RootPath + "\\\" + "Annot_1.pdf";

PDFDocument doc = new PDFDocument(inputFilePath);
List<IPDFAnnot> annots = PDFAnnotHandler.GetAllAnnotations(doc);
foreach (IPDFAnnot annot in annots)
{
    Console.WriteLine("Annotation Flags");
    Console.WriteLine(" Invisible:  " + annot.IsInvisible);
    Console.WriteLine(" Hidden:    " + annot.IsHidden);
    Console.WriteLine(" Print:     " + annot.IsPrint);
    Console.WriteLine(" No Zoom:   " + annot.NoZoom);
    Console.WriteLine(" No Rotate: " + annot.NoRotate);
    Console.WriteLine(" No View:   " + annot.NoView);
    Console.WriteLine(" Read Only: " + annot.IsReadOnly);
    Console.WriteLine(" Locked:    " + annot.IsLocked);
    Console.WriteLine(" Toggle No View: " + annot.IsToggleNoView);
    Console.WriteLine(" Locked Contents: " + annot.IsLockedContents);
}
```

VB

Set annotation flags

C#

```
String inputFilePath = Program.RootPath + "\\\" + \"2.pdf\";
String outputFilePath = Program.RootPath + "\\\" + \"Annot_1.pdf\";

// create the annotation
PDFAnnotDeleteLine annot = new PDFAnnotDeleteLine();
annot.StartPoint = new PointF(100F, 200F);
annot.EndPoint = new PointF(300F, 400F);

// set annotation flags
annot.IsInvisible = false;
annot.IsPrint = true;
annot.NoRotate = true;
annot.NoZoom = true;

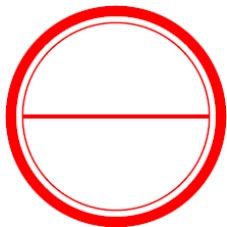
// add annotation
PDFAnnotHandler.AddAnnotation(inputFilePath, 1, annot, outputFilePath);
```

VB

Stamp Annotation

Prepare a stamp source file (PDF file)

C#
<pre>String outputPath = Program.RootPath + "\\\" + "StampTemplateBg1.pdf"; // initial a region (1 inch in both width and height) in the context for designing a template background // Region Boundary: [0, 0, 96, 96] (in 96 dpi) PDFContext ctx = new PDFContext(); ctx.SetWidth(new RLength(1, Units.IN)); ctx.SetHeight(new RLength(1, Units.IN)); // draw the outer circle ctx.DrawEllipse(new RPen(Color.Red, 6F), 3, 3, 90, 90); // draw the inner circle ctx.DrawEllipse(new RPen(Color.Red, 1F), 8, 8, 80, 80); // draw the seperater line ctx.DrawLine(new RPen(Color.Red, 1F), 8, 48, 88, 48); // output to a file ctx.SaveToFile(outputFilePath);</pre>
VB



C#

```
String outputPath = Program.RootPath + "\\\" + "StampTemplateBg2.pdf";
```

```
// initial a region in the context for designing a template background
```

```
// Region Boundary: [0, 0, 192, 96] (in 96 dpi)
```

```
PDFContext ctx = new PDFContext();
```

```
ctx.SetWidth(new RLength(2, Units.IN));
```

```
ctx.SetHeight(new RLength(1, Units.IN));
```

```
// draw an outer rectangle
```

```
ctx.DrawRectangle(new RPen(Color.Red, 6F), 3, 3, 186, 90);
```

```
// draw an inner rectangle
```

```
ctx.DrawRectangle(new RPen(Color.Red, 1F), 8, 8, 176, 80);
```

```
// output to a file
```

```
ctx.SaveToFile(outputFilePath);
```

VB



Create a new stamp annotation template from an exist PDF file

C#
<pre>String sourceFilePath = Program.RootPath + "\\\" + "StampTemplateBg1.pdf"; int pageIndex = 0; // create template by the specified page PDFStampTemplate template = PDFStampTemplate.Create(sourceFilePath, pageIndex); // to add optional dynamic fields for the template ...</pre>
VB

Create a new stamp annotation template by using PDF Context

C#
<pre>PDFContext ctx = new PDFContext(); ctx.SetWidth(new RLength(1F, Units.IN)); ctx.SetHeight(new RLength(1F, Units.IN)); // draw shapes to context ctx.DrawEllipse(new RPen(Color.Red, 6F), 3, 3, 90, 90); ctx.DrawEllipse(new RPen(Color.Red, 1F), 8, 8, 80, 80); ctx.DrawLine(new RPen(Color.Red, 1F), 8, 48, 88, 48); // create template by context PDFStampTemplate template = PDFStampTemplate.Create(ctx); // to add optional dynamic fields for the template ...</pre>
VB

Create a stamp annotation template with dynamic fields

C#

```
PDFContext ctx = new PDFContext();
ctx.SetWidth(new RELength(1F, Units.IN));
ctx.SetHeight(new RELength(1F, Units.IN));

// draw shapes to context
ctx.DrawEllipse(new RPen(Color.Red, 6F), 3, 3, 90, 90);
ctx.DrawEllipse(new RPen(Color.Red, 1F), 8, 8, 80, 80);
ctx.DrawLine(new RPen(Color.Red, 1F), 8, 48, 88, 48);

// create template by context
PDFStampTemplate template = PDFStampTemplate.Create(ctx);

// define a dynamic text field for User Info - Name
PDFStampTemplateField field0 = new PDFStampTemplateField();
// dynamic text field region in the context
field0.Boundary = new RectangleF(0, 18, 100, 20);
field0.FieldType = PDFStampTemplateFieldType.Name;
field0.TextColor = Color.FromArgb(255, 0, 0);
field0.TextFont = new Font("Times New Roman", 10, FontStyle.Regular);
// add the field
template.AddField(field0);

// define a dynamic text field for User Info - Company
PDFStampTemplateField field1 = new PDFStampTemplateField();
field1.Boundary = new RectangleF(0, 62, 100, 20);
field1.FieldType = PDFStampTemplateFieldType.Company;
field1.TextColor = Color.FromArgb(255, 0, 0);
field1.TextFont = new Font("Times New Roman", 8, FontStyle.Regular);
template.AddField(field1);

// define a dynamic text field for User Info - Current Date Time
PDFStampTemplateField field2 = new PDFStampTemplateField();
field2.Boundary = new RectangleF(0, 40, 100, 20);
field2.FieldType = PDFStampTemplateFieldType.CurrertDateTime;
field2.TextColor = Color.FromArgb(255, 0, 0);
field2.TextFont = new Font("Times New Roman", 6, FontStyle.Regular);
template.AddField(field2);
```

VB

Add a new stamp annotation template

C#

```
// ...create a stamp annotation template ...
PDFStampTemplate template = PDFStampTemplate.Create("", 0);

// template ID must be unique in the SDK
String templateID = "Template 1";
// add new template to the SDK
PDFStampTemplateMgr.AddTemplate(templateID, template, false);

if (PDFStampTemplateMgr.IsTemplateExist(templateID))
{
    Console.WriteLine("Template Exist");
    // get preview of a template with a given size
    Bitmap preview = PDFStampTemplateMgr.GetTemplatePreview(templateID, new Size(300, 300));
}
else
{
    Console.WriteLine("Template Dost Not Exist");
}
```

VB

Add a stamp annotation to page

C#
<pre>String inputFilePath = Program.RootPath + "\\\" + "1.pdf"; String outputFilePath = Program.RootPath + "\\\" + "1_Annots.pdf"; // Create a stamp annotation PDFAnnotStamp annot = new PDFAnnotStamp(); // select stamp template by the template ID String templateID = @"Template 1"; annot.StampTemplateID = templateID; // set annotation position and size annot.Boundary = new RectangleF(300, 300, 50, 50); // set user info annot.UserInfo.Company = "XYZ Ltd."; annot.UserInfo.FamilyName = "Kitty"; annot.UserInfo.GivenName = "Hello"; // other annotation properties annot.Opacity = 1.0; annot.PageIndex = 0; // add annotation PDFAnnotHandler.AddAnnotation(inputFilePath, annot, outputFilePath);</pre>
VB

Get all embedded stamp template names in the SDK

C#
<pre>String[] templateIDs = PDFStampTemplateMgr.GetAllEmbeddedStampNames(); Console.WriteLine("Embedded Stamp Template Names:"); foreach (String templateID in templateIDs) { Console.WriteLine("- " + templateID); }</pre>
VB

Remark

Do not use these stamp template name for custom stamp template, otherwise, the new stamp template created by user would be override by the embedded template with the same name

Get preview (Btmap) of an embedded stamp template

Standard Business Stamp Template – Completed

C#
<pre>String outputPath = Program.RootPath + "\\\" + "Annot_Sample37a.png"; // get an embedded stamp template name (Standard Business Stamp - Completed) // valid enumerations // STDStamp - for standard stamps // SBStamp - for standard business stamps // DStamp - for dynamic stamps // SHStamp - for sign-here stamps String templateID = PDFStampDef.GetStampName(SBStamp.Completed); // get preview image of the template with its actual size. Bitmap bitmap = PDFStampTemplateMgr.GetEmbeddedStampTemplatePreview(templateID, SizeF.Empty); // output the image bitmap.Save(outputFilePath);</pre>
VB

Dynamic Stamp Template – Approved

C#
<pre>String outputPath = Program.RootPath + "\\\" + "Annot_Sample37b.png"; // get an embedded stamp template name (Dynamic Stamp - Approved) String templateID = PDFStampDef.GetStampName(DStamp.Approved); // get preview image of the template with its actual size. Bitmap bitmap = PDFStampTemplateMgr.GetEmbeddedStampTemplatePreview(templateID, SizeF.Empty); // output the image bitmap.Save(outputFilePath);</pre>
VB

Add a stamp annotation by using embedded stamp template

Standard Business Stamp Template – Not For Public Release

C#
<pre>String inputFilePath = Program.RootPath + "\\\" + "1.pdf"; String outputFilePath = Program.RootPath + "\\\" + "Annot_Sample38a.pdf"; // get an embedded stamp template name (Standard Business Stamp - Not For Public Release) String templateID = PDFStampDef.GetStampName(SBStamp.NotForPublicRelease); // get actual size of the stamp template SizeF stampActualSize = PDFStampTemplateMgr.GetEmbeddedStampTemplateActualSize(templateID); // location of the stamp annotation PointF location = new PointF(50F, 200F); PDFAnnotStampUserInfo userInfo = new PDFAnnotStampUserInfo(); userInfo.GivenName = "Hello"; userInfo.FamilyName = "Kitty"; userInfo.Company = "Raster Edge"; // create the stamp annotation PDFAnnotStamp annot = new PDFAnnotStamp(); // assign the stamp template ID annot.StampTemplateID = templateID; // set the location and size of the annotation annot.Boundary = new RectangleF(location, stampActualSize); // set user info annot.UserInfo = userInfo; // add stamp annotation to the file PDFAnnotHandler.AddAnnotation(inputFilePath, annot, outputFilePath);</pre>
VB

Dynamic Stamp Template – Reviewed

C#

```
String inputFilePath = Program.RootPath + "\\\" + "1.pdf";
String outputFilePath = Program.RootPath + "\\\" + "Annot_Sample38b.pdf";

// get an embedded stamp template name (Dynamic Stamp - Reviewed)
String templateID = PDFStampDef.GetStampName(DStamp.Reviewed);
// get actual size of the stamp template
SizeF stampActualSize = PDFStampTemplateMgr.GetEmbeddedStampTemplateActualSize(templateID);
// location of the stamp annotation
PointF location = new PointF(50F, 200F);

PDFAnnotStampUserInfo userInfo = new PDFAnnotStampUserInfo();
userInfo.GivenName = "Hello";
userInfo.FamilyName = "Kitty";
userInfo.Company = "Raster Edge";

// create the stamp annotation
PDFAnnotStamp annot = new PDFAnnotStamp();
// assign the stamp template ID
annot.StampTemplateID = templateID;
// set the location and size of the annotation
annot.Boundary = new RectangleF(location, stampActualSize);
// set user info
annot.UserInfo = userInfo;

// add stamp annotation to the file
PDFAnnotHandler.AddAnnotation(inputFilePath, annot, outputFilePath);
```

VB

Annotation Data File (.fdf, .xfdf) Import & Export

Export annotations from a PDF document to an FDF file

C#
<pre>String inputFilePath = Program.RootPath + "\\\" + "1.pdf"; String outputFilePath = Program.RootPath + "\\\" + "1.fdf"; PDFDocument.ExportFDFDocument(inputFilePath, outputFilePath);</pre>
VB

Export annotations from a specified page to an FDF file

C#
<pre>String inputFilePath = Program.RootPath + "\\\" + "1.pdf"; String outputFilePath = Program.RootPath + "\\\" + "1.fdf"; // export all annotations in the 1st page int pageIndex = 0; PDFDocument.ExportFDFDocument(inputFilePath, pageIndex, outputFilePath);</pre>
VB

Import an FDF file to a PDF document

C#
<pre>String inputFilePath = Program.RootPath + "\\\" + "2.pdf"; String srcFDFFilePath = Program.RootPath + "\\\" + "1.fdf"; String outputFilePath = Program.RootPath + "\\\" + "Annot_Sample43.pdf"; ImportFDFOptions ops = new ImportFDFOptions(); PDFDocument.ImportFDFDocument(inputFilePath, srcFDFFilePath, outputFilePath, ops);</pre>
VB

Others

Move all annotations in a document without update other properties

C#
<pre>String inputFilePath = Program.RootPath + "\\\" + "11.pdf"; String outputFilePath = Program.RootPath + "\\\" + "Annot_Sample39.pdf"; // open the document PDFDocument doc = new PDFDocument(inputFilePath); // get all annotations in the document List<IPDFAnnot> annots = PDFAnnotHandler.GetAllAnnotations(doc); // move all annotations foreach (IPDFAnnot annot in annots) { // move the annotation to 100 pixels right and 100 pixels down. PDFAnnotHandler.MoveTo(annot, doc, 100F, 100F); } // save the new document doc.Save(outputFilePath);</pre>
VB

Remarks:

Some types of annotations **CANNOT BE** moved, such as:

- PDFAnnotUnderLine
- PDFAnnotDeleteLine
- PDFAnnotTextInsert
- PDFAnnotTextReplace

Update subject, content, artist and modified datetime values of an annotation

C#
<pre>String inputFilePath = Program.RootPath + "\\\" + "11.pdf"; String outputFilePath = Program.RootPath + "\\\" + "Annot_Sample40.pdf"; PDFDocument doc = new PDFDocument(inputFilePath); List<IPDFAnnot> annots = PDFAnnotHandler.GetAllAnnotations(doc); foreach (IPDFAnnot annot in annots) { // new subject value String subject = "Subject"; // null means this property would not be changed String content = null; // new artist value String artist = "DT"; // new modified datetime DateTime modifiedDatetime = DateTime.Now; // update annotation by the new values PDFAnnotHandler.Update(annot, doc, 4, subject, content, artist, modifiedDatetime); } doc.Save(outputFilePath);</pre>
VB

Update popup windows of an annotation

C#
<pre>String inputFilePath = Program.RootPath + "\\\" + "11.pdf"; String outputFilePath = Program.RootPath + "\\\" + "Annot_Sample41.pdf"; PDFDocument doc = new PDFDocument(inputFilePath); List<IPDFAnnot> annots = PDFAnnotHandler.GetAllAnnotations(doc); int y = 0; foreach (IPDFAnnot annot in annots) { // define popup window's boundary RectangleF popupWindowsBoundary = new RectangleF(600, y, 200, 80); // set open popup windows flag to true bool isOpenPopupWindow = true; PDFAnnotHandler.UpdatePopup(annot, doc, -1, popupWindowsBoundary, isOpenPopupWindow); y += 90; } doc.Save(outputFilePath);</pre>
VB

Remarks:

Types of annotation **WITHOUT** popup window:

- PDFAnnotFileAttach
- PDFAnnotMovie
- PDFAnnotSound
- PDFAnnotText
- PDFAnnotTextBox
- PDFAnnotTextCallout
- PDFAnnotLink