

Um alle folgenden Beispiele nachvollziehen zu können sollten Sie sich zuerst das aktuellste Microsoft Open XML Format SDK herunterladen. Zum jetzigen Zeitpunkt ist dies das SDK in der Version 2.0.

[Hier gehts zum Download für den Open XML Format SDK 2.0](#)

Im Visual Studio ist dann nach dem Anlegen des Projektes noch ein Verweis auf die dll einzufügen.

## 1. Finden des Textteils

Text wird im OpenXML SDK in Form der `<DocumentFormat.OpenXml.Wordprocessing.Text>` Klasse (entspricht dem `<w:t>` Tag eingefügt. Da diese per Definition nicht alleine im Dokument stehen darf, muss diese von einer `<DocumentFormat.OpenXml.Wordprocessing.Run>` Klasse (entspricht dem `<w:r>` Tag) und einer `<DocumentFormat.OpenXml.Wordprocessing.Paragraph>` Klasse (entspricht dem `<w:p>` Tag) umgeben sein. Aufgrund dieser Tatsache müssen wir diese Hierarchie durch iterieren:

```
WordprocessingDocument doc =  
WordprocessingDocument.Open(pathToDocument,true);  
  
Body body = doc.MainDocumentPart.Document.Body;  
  
foreach (Paragraph par in body.Elements<Paragraph>())  
{  
    foreach (Run run in par.Elements<Run>())  
    {  
        foreach (Text text in run.Elements<Text>())  
        {  
            .....  
        }  
    }  
}
```

In der inneren foreach-Schleife können wir dann den Text mit dem gesuchten Textteil vergleichen:

```
if (text.Text == suchtext)  
{  
    .....  
}
```

Um diesen Textblock zu löschen (um ihn durch das Bild zu ersetzen) ruft man einfach die Methode

```
text.Remove();
```

auf. Diese löscht den Block `<w:t>suchtext</w:t>` aus dem XML Dokument heraus und lässt so nur noch den Paragraph und den Run Block übrig. In diese können wir nun "einfach" das Bild einsetzen....

## 2. Einfügen des Bildes in die docx Ordnerstruktur

Um nun ein Bild einfügen zu können muss dieses zunächst einmal in die Ordnerstruktur des docx Files eingebunden werden. Das bedeutet zum einen, dass das Bild als Datei in den Media Ordner des docx Files gelegt werden muss, zum anderen, dass eine eindeutige ID generiert werden muss, auf die wir uns später beim Einfügen beziehen können.

```
MainDocumentPart mainPart = doc.MainDocumentPart;
```

```
ImagePart imagePart = mainPart.AddImagePart(ImagePartType.Jpeg);
```

Die obere Codezeile geht davon aus, dass wir ein JPG Bild einfügen möchten es gibt weitere Typen für alle gängigen Bildtypen.

```
using (FileStream stream = new FileStream(m_ImagePath, FileMode.Open))  
{  
    imagePart.FeedData(stream);  
}
```

Die string Variable "m\_ImagePath" zeigt auf die Bilddatei die eingefügt werden soll also z.B. auf "C:\tmp\test.jpg".

Nach diesem Block ist das Bild in die Struktur integriert und wird beim Speichern im Media Ordner abgelegt.

## 3. Einfügen des Bildes in das Dokument

Um das Bild in das eigentliche Word Dokument einfügen zu können benötigen wir die vom SDK generierte eindeutige ID des Bildes.

```
string ipID = mainPart.GetIdOfPart(imagePart);
```

Anhand dieser ID kann der Wordparser später das Bild darstellen.

Der eigentliche Einfügevorgang ist etwas aufwändiger. Zunächst legen wir eine Fill-Klasse an. Diese entspricht im XML Dokument dem Tag <w:Fill>.

```
Fill fill = new Fill();
```

Diese nimmt später das Bild auf und legt dessen Attribute genau fest (wie z.B. Breite und Höhe etc.).

Das Bild fügen wir mit folgender Codezeile ein.

```
fill.SetAttribute(new OpenXmlAttribute("id",  
"http://schemas.openxmlformats.org/officeDocument/2006/relationships", ipID));  
fill.SetAttribute(new OpenXmlAttribute("type", "", "frame"));
```

Hier geben wir die zuvor abgefragte ID des Bildes an. Der zweite Parameter ist die sog. Namespace URI. Diese ist unbedingt erforderlich.

Jetzt müssen wir ein Rechteck anlegen. Dies und weitere Parametrisierungen machen wir mit folgenden Codezeilen.

```
Rectangle rect = new Rectangle();  
rect.SetAttribute(new OpenXmlAttribute("style", "", "width:12cm;height:9cm;"));  
rect.SetAttribute(new OpenXmlAttribute("stroked", "", "t"));
```

Es ist extrem wichtig nicht das Rectangle des Drawing Namespaces zu verwenden sondern das aus dem

### **DocumentFormat.OpenXml.Vml**

Namespace!!! Andernfalls wird das Dokument nicht geöffnet werden können!!

Für width und height können alle üblichen Maßeinheiten verwendet werden wie z.B. px oder pt.

Das Attribut stroked bewirkt das Zeichnen eines Rahmens (bei einem Wert von "t" == true) bzw. das Weglassen eines Rahmens um das Bild (bei einem Wert von "f" == false).

Jetzt müssen wir noch das Rechteck in das Fill Tag einbetten.

```
rect.PrependChild<Fill>(fill);
```

Das Fill Tag in ein <w:pict> Tag einbauen.

```
Picture pict = new Picture(new OpenXmlElement[] { rect });
```

Und das Picture Tag in das <w:r> Tag einbauen.

```
run.InsertAt<Picture>(pict, 0);
```

#### **4. Anmerkungen**

Diese Vorgehensweise ist natürlich nicht der alleinige Weg um an's Ziel zu kommen.  
Allerdings funktioniert er zuverlässig ;-)