How to: Programmatically Populate Word Tables with Document Properties

The following example creates a Microsoft Office Word table at the top of the document and populates it with the properties of the host document.

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.

Populating Tables in a Document-Level Customization

To create a table and populate it with document properties

1. Set the range to the top of the document.

   C#

   ```
   object start = 0, end = 0;
   Word.Range rng = this.Range(ref start, ref end);
   ```

2. Insert a title for the table and include paragraph marks.

   C#

   ```
   rng.InsertBefore("Document Statistics");
   rng.Font.Name = "Verdana";
   rng.Font.Size = 16;
   rng.InsertParagraphAfter();
   rng.InsertParagraphAfter();
   rng.SetRange(rng.End, rng.End);
   ```

3. Add the table to the document at the range.

   C#

   ```
   rng.Tables.Add(this.Paragraphs[2].Range, 3, 2, ref missing, ref missing);
   ```

4. Format the table and apply a style.

C#

```
            Word.Table tbl = this.Tables[1];
            tbl.Range.Font.Size = 12;
            tbl.Columns.DistributeWidth();

            object styleName = "Table Professional";
            tbl.set_Style(ref styleName);
```

5. Insert the document properties into cells.

C#

```
            tbl.Cell(1, 1).Range.Text = "Document Property";
            tbl.Cell(1, 2).Range.Text = "Value";

            tbl.Cell(2, 1).Range.Text = "Subject";
            tbl.Cell(2, 2).Range.Text = ((Office.DocumentProperties)
(this.BuiltInDocumentProperties))
                [Word.WdBuiltInProperty.wdPropertySubject].Value.ToString();

            tbl.Cell(3, 1).Range.Text = "Author";
            tbl.Cell(3, 2).Range.Text = ((Office.DocumentProperties)
(this.BuiltInDocumentProperties))
                [Word.WdBuiltInProperty.wdPropertyAuthor].Value.ToString();
```

The following example shows the complete procedure. To use this code, run it from the `ThisDocument` class in your project.

C#

```
        private void CreateDocumentPropertyTable()
        {
            object start = 0, end = 0;
            Word.Range rng = this.Range(ref start, ref end);

            // Insert a title for the table and paragraph marks.
            rng.InsertBefore("Document Statistics");
            rng.Font.Name = "Verdana";
            rng.Font.Size = 16;
            rng.InsertParagraphAfter();
            rng.InsertParagraphAfter();
            rng.SetRange(rng.End, rng.End);

            // Add the table.
            rng.Tables.Add(this.Paragraphs[2].Range, 3, 2, ref missing, ref missing);

            // Format the table and apply a style.
            Word.Table tbl = this.Tables[1];
            tbl.Range.Font.Size = 12;
            tbl.Columns.DistributeWidth();
            object styleName = "Table Professional";
            tbl.set_Style(ref styleName);

            // Insert document properties into cells.
            tbl.Cell(1, 1).Range.Text = "Document Property";
            tbl.Cell(1, 2).Range.Text = "Value";

            tbl.Cell(2, 1).Range.Text = "Subject";
            tbl.Cell(2, 2).Range.Text = ((Office.DocumentProperties)
(this.BuiltInDocumentProperties))
                [Word.WdBuiltInProperty.wdPropertySubject].Value.ToString();

            tbl.Cell(3, 1).Range.Text = "Author";
            tbl.Cell(3, 2).Range.Text = ((Office.DocumentProperties)
(this.BuiltInDocumentProperties))
                [Word.WdBuiltInProperty.wdPropertyAuthor].Value.ToString();
        }
```

Populating Tables in a VSTO Add-in

To create a table and populate it with document properties

1. Set the range to the top of the document.

C#

```
            object start = 0, end = 0;
            Word.Document document = this.Application.ActiveDocument;
            Word.Range rng = document.Range(ref start, ref end);
```

2. Insert a title for the table and include paragraph marks.

C#

```
            rng.InsertBefore("Document Statistics");
            rng.Font.Name = "Verdana";
            rng.Font.Size = 16;
            rng.InsertParagraphAfter();
            rng.InsertParagraphAfter();
            rng.SetRange(rng.End, rng.End);
```

3. Add the table to the document at the range.

C#

```
            rng.Tables.Add(document.Paragraphs[2].Range, 3, 2, ref missing, ref missing);
```

4. Format the table and apply a style.

C#

```
            Word.Table tbl = document.Tables[1];
            tbl.Range.Font.Size = 12;
            tbl.Columns.DistributeWidth();

            object styleName = "Table Professional";
            tbl.set_Style(ref styleName);
```

5. Insert the document properties into cells.

C#

```
            tbl.Cell(1, 1).Range.Text = "Document Property";
            tbl.Cell(1, 2).Range.Text = "Value";

            tbl.Cell(2, 1).Range.Text = "Subject";
            tbl.Cell(2, 2).Range.Text = ((Office.DocumentProperties)
(document.BuiltInDocumentProperties))
                [Word.WdBuiltInProperty.wdPropertySubject].Value.ToString();

            tbl.Cell(3, 1).Range.Text = "Author";
            tbl.Cell(3, 2).Range.Text = ((Office.DocumentProperties)
(document.BuiltInDocumentProperties))
                [Word.WdBuiltInProperty.wdPropertyAuthor].Value.ToString();
```

The following example shows the complete procedure. To use this code, run it from the `ThisAddIn` class in your project.

C#

```
        private void CreateDocumentPropertyTable()
        {
            object start = 0, end = 0;
            Word.Document document = this.Application.ActiveDocument;
            Word.Range rng = document.Range(ref start, ref end);

            // Insert a title for the table and paragraph marks.
            rng.InsertBefore("Document Statistics");
            rng.Font.Name = "Verdana";
            rng.Font.Size = 16;
            rng.InsertParagraphAfter();
            rng.InsertParagraphAfter();
            rng.SetRange(rng.End, rng.End);

            // Add the table.
            rng.Tables.Add(document.Paragraphs[2].Range, 3, 2, ref missing, ref missing);

            // Format the table and apply a style.
            Word.Table tbl = document.Tables[1];
            tbl.Range.Font.Size = 12;
            tbl.Columns.DistributeWidth();

            object styleName = "Table Professional";
            tbl.set_Style(ref styleName);

            // Insert document properties into cells.
            tbl.Cell(1, 1).Range.Text = "Document Property";
            tbl.Cell(1, 2).Range.Text = "Value";

            tbl.Cell(2, 1).Range.Text = "Subject";
            tbl.Cell(2, 2).Range.Text = ((Office.DocumentProperties)
(document.BuiltInDocumentProperties))
                [Word.WdBuiltInProperty.wdPropertySubject].Value.ToString();

            tbl.Cell(3, 1).Range.Text = "Author";
            tbl.Cell(3, 2).Range.Text = ((Office.DocumentProperties)
(document.BuiltInDocumentProperties))
                [Word.WdBuiltInProperty.wdPropertyAuthor].Value.ToString();
        }
```

See Also

How to: Programmatically Create Word Tables
How to: Programmatically Add Text and Formatting to Cells in Word Tables
How to: Programmatically Add Rows and Columns to Word Tables
Optional Parameters in Office Solutions