

# Range

## Description

A [Range](#) object represents a range in a workbook.

### C#

```
[DefaultMember("Item")]  
public sealed class Range
```

### vb.net

```
<DefaultMember("Item")> _  
Public NotInheritable Class Range
```

## Remarks

A range is a collection of areas. An area is a rectangular collection of cells. The areas in a range may be non-adjacent, and a range can include areas in different worksheets.

To create a [Range](#) (without a name), call one of the following methods:

- [Workbook.CreateRange\(rangeFormula\)](#)
- [Worksheet.CreateRange\(rangeFormula\)](#)

If a range is named, it will be accessible when the workbook is opened in Microsoft Excel. To create a named range, call one of the following methods:

- [Workbook.CreateNamedRange\(rangeFormula, rangeName\)](#)
- [Worksheet.CreateNamedRange\(firstRow, firstColumn, numRows, numColumns, rangeName\)](#)
- [Worksheet.CreateNamedRange\(rangeFormula, rangeName\)](#)

## Examples

### C#

```
ExcelApplication xla = new ExcelApplication();  
Workbook wb = xla.Create();  
wb.Worksheets.CreateWorksheet("Sheet2");  
Range rng = wb.CreateRange("=Sheet1!A1:A3, Sheet2!A2:C5");
```

## vb.net

```
Dim xla As New ExcelApplication()  
Dim wb As Workbook = xla.Create()  
wb.Worksheets.CreateWorksheet("Sheet2")  
Dim rng As Range = wb.CreateRange("=Sheet1!A1:A3, Sheet2!A2:C5")
```

## Properties

Name	Description
AreaCount	Returns the number of <a href="#">Area</a> objects contained in this <a href="#">Range</a> .
Areas	Returns an array of the rectangular areas contained in the <a href="#">Range</a> .
BorderAround	Returns a <a href="#">Border</a> object that represents a border around the range.
FirstCellStyle	Returns the style for the first cell in the range. Changes to this style will apply <u>only</u> to the first cell. To assign this style to the entire <a href="#">Range</a> , set <a href="#">Range.SetStyle()</a> to this <a href="#">Style</a> reference.

## Indexers

Name	Description
Item(Int32)	Returns the <a href="#">Area</a> object at the specified 0-based index.

## Methods

Name	Description
ApplyStyle(Style)	Applies a style to the area. When a style is applied - rather than set (see <a href="#">Range.SetStyle</a> ) - only the differences between the new style and style properties previously assigned to the range (through the <a href="#">ExcelWriter</a> API) will take effect. For example, if the range has a background color and the new style applied does not contain a background color, the area's color will not be affected.
ClearContent()	Clears the content of all cells in the range. Calling this method is equivalent to setting a range's style to "Normal" and its cell values to null.
JoinRange(Range)	Adds another range to this range.
RemoveConditionalFormat()	This method removes any <a href="#">ConditionalFormat</a> objects from the <a href="#">Range</a> .
SetConditionalFormat(ConditionalFormat)	This method copies the specified <a href="#">ConditionaFormat</a> object and associates it with this <a href="#">Range</a> . If any <a href="#">ConditionalFormat</a> objects already exist within the <a href="#">Range</a> , they will be removed and replaced by the specified one.
SetDataValidation(DataValidation)	Assigns a data validation rule to all cells in the <a href="#">Range</a> . If the <a href="#">DataVali dation</a> object uses a local reference or area in a formula (i.e. "=A5:B7"), then the range will adjust those local references accordingly based on the range's location. The original <a href="#">DataValida tion</a> object will not be affected.

SetStyle(Style)	Sets the style for every cell in this range. When a style is set - rather than applied (see <a href="#">Range.ApplyStyle</a> ) - all previously assigned style properties, including font and number formatting, will be overwritten for all cells in the range.
ToString()	Returns a string representation of the <a href="#">Range</a> object.