Addressing Cells

There are three ways to address cells, areas, and ranges using ExcelApplication:

Excel's A1 Reference Style

A1-style references are easy to read, intuitive, and match the default reference style seen in Microsoft Excel.

Example

```
ExcelApplication xla = new ExcelApplication();
Workbook wb = xla.Create();
Worksheet ws = wb.Worksheets[0];
ws.Cells["A3"].Value = "Jan";
ws.Cells["B3"].Value = "Feb";
ws.Cells["C3"].Value = "Mar";
Area a = ws.CreateArea("=A1:G10");
```

A1 cell references are relative by default. Therefore, if you enter "=B1" in cell A1 and drag A1 down, the formula in cell Ax will be "=Bx". To make an A1 row or column reference absolute, prepend a dollar sign to the row or column. For example, if you enter "=B$1" in cell A1 and drag A1 down, the formula in cell Ax will be "=B$1".

The reference style used in the generated workbook (when it is opened in Excel) is determined by the value of the property Workbook.UseRCFormulaNotation. If it is false (the default value), Excel will use A1 references. If UseRCFormulaNotation is set to true, Excel will use R1C1-style references.

Excel's R1C1 Reference Style

In the R1C1 style, the location of a cell is specified with an "R" followed by a row number and a "C" followed by a column number. R1C1 references can be either absolute or relative. To create an absolute reference, specify the row and column numbers without brackets. For example, "=R1C2" equates to row 1, column 2. To create a relative reference for a cell, specify the row and column numbers in brackets; and enter the difference between that cell's row and column numbers and those of the cell it references. Use negative numbers to reference cells above or to the left of a cell. Use positive numbers to reference cells below or to the right of a cell. For example, entering "=R[-2]C[-1]" in cell R3C2 causes it to refence the value of cell R1C1.

Example

Relative R1C1 references enable you to reuse a single formula in multiple rows or columns. The following example copies one formula to three different columns to calculate the sum of those columns.

```
ExcelApplication xla = new ExcelApplication();
Workbook wb = xla.Create();
wb.UseRCFormulaNotation = true;
Worksheet ws = wb.Worksheets[0];
String formulaString = "=SUM(R[-12]C:R[-1]C)";
Cell columnTotal;
for (int i = 0;i < 3; i++)
{
     columnTotal = ws.Cells[13, i + 1];
     columnTotal.Formula = formulaString;
}
```

The reference style used in the generated workbook (when it is opened in Excel) is determined by the value of the property

Workbook.UseRCFormulaNotation. If it is false (the default value), Excel will use A1 references. If UseRCFormulaNotation is set to true, Excel will use R1C1-style references. **Note:** Workbooks created with ExcelApplication default to `False`.

There are several valid ways of addressing a cell or an area using RC notation:

Single Cell, Absolute	RxCy
Single Cell, Row Relative	R[x]Cy
Single Cell, Column Relative	RxC[y]
Single Cell, Both Relative	R[x]C[y]
Single Cell, Same Relative Row, Absolute Column	RCy
Single Cell, Same Relative Row, Relative Column	RC[y]
Single Cell, Absolute Row, Same Relative Column	RxC
Single Cell, Relative Row, Same Relative Column	R[x]C
Entire Row, Absolute	Rx
Entire Row, Relative	R[x]
Entire Row, Same, Relative	R
Entire Column, Absolute	Cy
Entire Column, Relative	C[y]
Entire Column, Same, Relative	C

Row Number and Column Number

Addressing cells by number allows you to iterate over cells in your ExcelWriter code. Row and Column number references are also processed faster than Excel-style references.

The following specifies cell A1 by 0-based row and column numbers: The first parameter of the Cell property is the row index, the second is the column index.

```
sheet1.Cells[2, 0].Value = "SoftArtisans OfficeWriter";
```

Using row and column numbers makes it easy to iterate over cells:for(int iRow = 3; iRow <= 23; iRow++)

```
for(int iCol = 0; iCol <= 2; iCol++)
     sheet1.Cells[iRow, iCol].Value = iRow + iCol;
```