May, 2006

Advisor Answers

Accessing cells in use in Excel

VFP 9/8/7

Q: I'm automating modification of an existing Excel workbook. How can I find out which cell is the last one currently used so I can work on only the section of the sheet that's in use?

A: Usually, recording a macro is a good first step to solving an automation problem. In this case, though, such a macro is misleading. While Excel provides a way to go to the last cell (choose Edit | Goto from the menu, then click Special and choose Last Cell), the macro that results from recording this action has just one line:

```
Selection.SpecialCells(xlCellTypeLastCell).Select
```

Although you can look up the value of the constant xlCellTypeLastCell (it's 11), translating this VBA code to VFP automation code isn't a good way to find the range of cells in use because it works through the Selection object and Select method. Whenever possible (which is almost always), it's better to use Range objects in automation of both Excel and Word. While there's only one Selection, you can define as many ranges as you need. Also, setting the Selection requires the automation server to actually highlight the specified range.

The best approach to this problem is using the UsedRange property of the worksheet. UsedRange provides an object reference to the portion of the worksheet currently in use. You might have a line of code like this:

```
oRange = oExcel.ActiveSheet.UsedRange
```

Then, you can treat oRange like any other range. To find out which cell is at the bottom right-hand corner of the used range, use the Rows and Columns collections:

```
nLastRow = oRange.Rows.Count
nLastCol = oRange.Columns.Count
```

Of course, in this case, the number of rows and columns gives us the address of the last cell because we know that the used range always starts in row 1, column 1.

You can get the cell address of the range using the Address property:

```
cRangeAddress = oRange.Address
```

This property returns a value in the form R1C1, such as "$A$1:$D$57".

Finally, it's worth pointing out that Excel determines the used range by finding the last column containing any data and the last row containing any data. The used range ends where those two intersect, even if that cell itself is empty.

–Tamar

Specifying Print Area in Excel

VFP 9/8/7

Q: How can I specify the print area for an Excel worksheet via Automation? How do I indicate which rows and columns should appear on every page as headings?

A: Because worksheets have so much available work space, Excel lets you specify what portion of the sheet is printed when you choose Print interactively or call the PrintOut method. By default, it prints the used range, so if that's what you want, you don't have to do anything special.

If you do want to print a different range, set the PrintArea property of the PageSetup object for the worksheet in question. You need the address of the range to print. For example:

```
oExcel.ActiveSheet.PageSetup.PrintArea = "C7:G52"
```

Of course, you can use a variable containing a range; just reference its Address property.

```
oExcel.ActiveSheet.PageSetup.PrintArea = oRange.Address
```

Excel also allows you to mark one or more rows and columns as headings (or titles) that appear on each page of the print-out. These are controlled by the PrintTitleRows and PrintTitleColumns properties of the PageSetup object. As with PrintArea, you set them by specifying an address. The last row or column you specify is considered as the end of the title rows or columns, so for example, the following command makes row 1 to 3 titles:

```
oExcel.ActiveSheet.PageSetup.PrintTitleRows = "$3:$3"
```

In addition to putting some rows and columns on every page, you can also specify a header and footer for each page, containing information that's not part of the worksheet's data. Each worksheet has a header and footer, each divided into three sections: left, center and right. So the PageSetup object has properties like LeftHeader and CenterFooter. You specify text for each of these. For example:

```
oExcel.ActiveSheet.PageSetup.LeftFooter = ;
  "Prepared by Tamar E. Granor"
```

You can also put a picture in any of those places, using properties RightHeaderPicture and LeftFooterPicture; each points to an object. To specify the picture to use, set the Filename property of the object, like this:

```
oExcel.ActiveSheet.PageSetup.CenterHeaderPicture.Filename;
 = "Logo.jpg"
```

Once you specify a picture, you can use additional properties of the picture object to set height and width, as well as cropping.

When you've set up the print area, titles, headers and footers you want, you can print your worksheet by calling the PrintOut method or preview it by calling PrintPreview.

–Tamar