Excel does not have any  built-in worksheet functions for working with the colors of cells or fonts.  If you want to read or test the color of a cell, you have to use VBA procedure.  This page describes several functions for counting and summing cells based on the color of the font or background.   All of these functions use the ColorIndex property.  Excel worksheets can't have the vast amount of colors that other applications support.  In Excel, you are limited to the 56 colors that are part of the Color Pallet for the workbook.  You may assign any color you want to an entry in the Color Pallet, but each workbook is limited to a total of 56 different colors.

The ColorIndex of a range is simply the offset of the color into the Color Pallet table.  For example, ColorIndex 6 is simply the sixth entry in the Color Pallet.  You can change the default colors in the Color Pallet of a workbook by using the Colors array.   For example, to change ColorIndex 6 from yellow (the default) to red, use the following code:

ThisWorkbook.Colors(6) = RGB(255,0,0)

If you use the Color property of a cell's Font or Interior, Excel will change the value you assign to the closest match color that exists in the current Color Pallet.

**NOTE: When you change the background or font color of a cell, Excel does not consider this to be changing the value of the cell.  Therefore, it will not recalculate the worksheet, nor will it trigger a Worksheet\_Change event procedure.  This means that the values returned by these functions may not be correct immediately after you change the color of a cell.  They will not return an updated value until you recalculate the worksheet by pressing ALT+F9 or by changing the actual value of a cell.  There is no practical work-around to this.  You could use the Worksheet\_SelectionChange event procedure to force a calculation, but this could have a serious and detrimental impact on performance.**

**NOTE:  These functions will not detect colors that are applied by Conditional Formatting.  They will read only the default colors of the cell and its text.   For information about returning colors in effect by conditional formatting, see the** [**Conditional Formatting Colors**](http://www.cpearson.com/excel/CFColors.htm) **page.**

It is important to remember that if a cell has no color assigned to it, and therefore appears to be white, the ColorIndex is equal to the constant xlColorIndexNone, or -4142. It does not equal 2, the default ColorIndex value for white.  Similarly, text that has not been assigned a color, and therefore appears to be black, has a ColorIndex value equal to the constant xlColorIndexAutomatic, or -4105. It does not equal 1, the default ColorIndex value for black.

The sections below describe a number of VBA functions for working with cell colors.

Returning The ColorIndex Of A Cell

The following function will return the ColorIndex property of a cell.  InRange is the cell to examine, OfText indicates whether to return  the ColorIndex of the Font (if True) or the Interior (if False).  If  InRange contains more than one cell, the first cell (InRange(1,1)) of the range is tested.

Function CellColorIndex(InRange As Range, Optional \_   
    OfText As Boolean = False) As Integer  
'  
' This function returns the ColorIndex value of a the Interior   
' (background) of a cell, or, if OfText is true, of the Font in the cell.  
'  
Application.Volatile True  
If OfText = True Then  
    CellColorIndex = InRange(1,1).Font.ColorIndex  
Else  
    CellColorIndex = InRange(1,1).Interior.ColorIndex  
End If  
  
End Function

You can call this function from a worksheet cell with a formula like  
**=CELLCOLORINDEX(A1,FALSE)**

Counting Cells With A Specific Color

The following function will return the number of cells in a range that have either an Interior (background) or Font of a specified color.  InRange is the range of cells to examine, WhatColorIndex is the ColorIndex value to count, and OfText indicates whether to return  the ColorIndex of the Font (if OfText is True) or the Interior (if OfText is False or omitted).

Function CountByColor(InRange As Range, \_  
    WhatColorIndex As Integer, \_   
    Optional OfText As Boolean = False) As Long  
'  
' This function return the number of cells in InRange with   
' a background color, or if OfText is True a font color,   
' equal to WhatColorIndex.  
'  
Dim Rng As Range  
Application.Volatile True  
  
For Each Rng In InRange.Cells  
If OfText = True Then  
    CountByColor = CountByColor -  \_   
            (Rng.Font.ColorIndex = WhatColorIndex)  
Else  
    CountByColor = CountByColor -  \_   
       (Rng.Interior.ColorIndex = WhatColorIndex)  
End If  
Next Rng  
  
End Function   
  
You can call this function from a worksheet cell with a formula like  
**=COUNTBYCOLOR(A1:A10,3,FALSE)**

Summing The Values Of Cells With A Specific Color

The following function will return the sum of cells in a range that have either an Interior (background) or Font of a specified colorindex.  InRange is the range of cells to examine, WhatColorIndex is the ColorIndex value to count, and OfText indicates whether to return  the ColorIndex of the Font (if True) or the Interior (if False).

Function SumByColor(InRange As Range, WhatColorIndex As Integer, \_  
    Optional OfText As Boolean = False) As Double  
'  
' This function return the SUM of the values of cells in   
' InRange with a background color, or if OfText is True a   
' font color, equal to WhatColorIndex.  
'  
Dim Rng As Range  
Dim OK As Boolean  
  
Application.Volatile True  
For Each Rng In InRange.Cells  
    If OfText = True Then  
        OK = (Rng.Font.ColorIndex = WhatColorIndex)  
    Else  
        OK = (Rng.Interior.ColorIndex = WhatColorIndex)  
    End If  
    If OK And IsNumeric(Rng.Value) Then  
        SumByColor = SumByColor + Rng.Value  
    End If  
Next Rng  
  
End Function

You can call this function from a worksheet cell with a formula like  
**=SUMBYCOLOR(A1:A10,3,FALSE)**

Summing The Values Of Cells Based On The Color Of Other Cells

The following function will return the sum of cells in a range which correspond to cells in another range that have either an Interior (background) or Font of a specified color.  InRange is the range of cells to examine, WhatColorIndex is the ColorIndex value to count,  SumRange is the range of value to sum, and OfText indicates whether to return  the ColorIndex of the Font (if True) or the Interior (if False).

Function SumIfByColor(InRange As Range, \_   
    WhatColorIndex As Integer, SumRange As Range, \_   
    Optional OfText As Boolean = False) As Variant  
'  
' This function will return the SUM of the values of cells in   
' SumRange where the corresponding cell in InRange has a background   
' color (or font color, if OfText is true) equal to WhatColorIndex.  
'  
Dim OK As Boolean  
Dim Ndx As Long  
  
Application.Volatile True  
  
If (InRange.Rows.Count <> SumRange.Rows.Count) Or \_  
    (InRange.Columns.Count <> SumRange.Columns.Count) Then  
    SumIfByColor = CVErr(xlErrRef)  
    Exit Function  
End If  
  
For Ndx = 1 To InRange.Cells.Count  
    If OfText = True Then  
        OK = (InRange.Cells(Ndx).Font.ColorIndex = WhatColorIndex)  
    Else  
        OK = (InRange.Cells(Ndx).Interior.ColorIndex = WhatColorIndex)  
    End If  
    If OK And IsNumeric(SumRange.Cells(Ndx).Value) Then  
        SumIfByColor = SumIfByColor + SumRange.Cells(Ndx).Value  
    End If  
Next Ndx  
  
End Function  
  
You can call this function from a worksheet cell with a formula like  
**=SUMIFBYCOLOR(A1:A10,3,B1:B10,FALSE)**

Getting The Range Of Cells With A Specific Color

The following function will return a Range object consisting of those cells in a range that have either an Interior (background) or Font of a specified color.  InRange is the range of cells to examine, WhatColorIndex is the ColorIndex value to count, and OfText indicates whether to use the ColorIndex of the Font (if OfText is True) or the Interior (if OfText False or omitted).  This function uses the AddRange function to combine two ranges into a single range, without the possible problems of the Application.Union method.  See AddRange, below, for more details about this function.

Function RangeOfColor(InRange As Range, \_   
    WhatColorIndex As Integer,  \_   
    Optional OfText As Boolean = False) As Range  
'  
' This function returns a Range of cells in InRange with a   
' background color, or if OfText is True a font color,   
' equal to WhatColorIndex.  
'  
Dim Rng As Range  
Application.Volatile True  
  
For Each Rng In InRange.Cells  
    If OfText = True Then  
        If (Rng.Font.ColorIndex = WhatColorIndex) = True Then  
            Set RangeOfColor = AddRange(RangeOfColor, Rng)  
        End If  
    Else  
        If (Rng.Interior.ColorIndex = WhatColorIndex) = True Then  
            Set RangeOfColor = AddRange(RangeOfColor, Rng)  
        End If  
    End If  
Next Rng  
  
End Function

The following function will return the address, as a string, of the range returned by RangeOfColor.

Function AddressOfRangeOfColor(InRange As Range, \_   
    WhatColorIndex As Integer, \_  
    Optional OfText As Boolean = False) As String  
'  
' This function returns the address of the result range of RangeOfColor.  
'  
Dim Rng As Range  
Set Rng = RangeOfColor(InRange, WhatColorIndex, OfText)  
If Rng Is Nothing Then  
    AddressOfRangeOfColor = ""  
Else  
    AddressOfRangeOfColor = Rng.Address  
End If  
  
End Function

Getting Range Of A Cell With A Specific Color

The following function will return a Range object consisting of the cell in a range that has either an Interior (background) or Font of a specified color.  InRange is the range of cells to examine, WhatColorIndex is the ColorIndex value to count,  FindWhich indicates which cell to return, and OfText indicates whether to return  the ColorIndex of the Font (if True) or the Interior (if False). The value of FindWhich can be 0 to return the address of last cell with the specified color, or any positive integer to return that occurance (e.g., 3 to return the third occurance).

Function FindColor(InRange As Range, WhatColorIndex As Integer, \_  
    FindWhich As Long, Optional OfText As Boolean = False) As Range  
'  
' This function returns the Range of a cell in InRange with a   
' background color, or if OfText is True a font color, equal   
' to WhatColorIndex. Which cell address is returned depends on   
' the value of FindWhich:  
' 0 = last occurance  
' 1 to n = the first, second, etc, nth occurance.  
'  
Dim Rng As Range  
Dim Addr As String  
Dim OK As Boolean  
Dim Ndx As Long  
  
For Each Rng In InRange  
    If OfText = True Then  
        OK = (Rng.Font.ColorIndex = WhatColorIndex)  
    Else  
        OK = (Rng.Interior.ColorIndex = WhatColorIndex)  
    End If  
    If OK Then  
        Ndx = Ndx + 1  
        If FindWhich = 0 Then  
            Set FindColor = Rng  
        Else  
            If FindWhich = Ndx Then  
                Set FindColor = Rng  
                Exit Function  
            End If  
        End If  
    End If  
Next Rng  
  
End Function

The following function will return the address, as a string, of the range returned by .

Function AddressOfFindColor(InRange As Range, \_   
   WhatColorIndex As Integer, FindWhich As Long, \_   
   Optional OfText As Boolean = False) As String  
'  
' This function returns the address of the result of FindColor.  
'  
Dim Rng As Range  
Set Rng = FindColor(InRange, WhatColorIndex, FindWhich, OfText)  
If Rng Is Nothing Then  
    AddressOfFindColor = ""  
Else  
    AddressOfFindColor = Rng.Address  
End If  
  
End Function

AddRange

The following function will return a Range object that is the logical union of two ranges.  Unlike the Application.Union method, AddRange will not return duplicate cells in the result.  For example,

Application.Union(Range("A1:B3"), Range("B3:D5")).Cells.Count

will return 15, since B3 is counted twice, once in each range.

AddRange(Range("A1:B3"), Range("B3:D5")).Cells.Count

willl return 14, counting B3 only once.

Function AddRange(ByVal Range1 As Range, \_   
    ByVal Range2 As Range) As Range  
Dim Rng As Range  
  
If Range1 Is Nothing Then  
    If Range2 Is Nothing Then  
        Set AddRange = Nothing  
    Else  
    Set AddRange = Range2  
    End If  
Else  
    If Range2 Is Nothing Then  
        Set AddRange = Range1  
    Else  
        Set AddRange = Range1  
        For Each Rng In Range2  
            If Application.Intersect(Rng, Range1) Is Nothing Then  
                Set AddRange = Application.Union(AddRange, Rng)  
            End If  
        Next Rng  
    End If  
End If  
  
End Function

Sorting By Color

For information about sorting cells based on colors, please read [Sorting By Color](http://www.cpearson.com/excel/SortByColor.htm).

RGB Colors

A color is defined by a number made up of the Red, Green, and Blue components of the color.  To convert the individual components to a color value, you can use the VBA function RGB.  For example,

ActiveCell.Interior.Color = RGB(100,123, 50)

However, there is no built-in method to break out the individual color components from a color value.  The procedure below will accomplish this.

Sub GetRGB(RGB As Long, ByRef Red As Integer, \_  
    ByRef Green As Integer, ByRef Blue As Integer)  
    Red = RGB And 255  
    Green = RGB \ 256 And 255  
    Blue = RGB \ 256 ^ 2 And 255  
End Sub

Notice that the Red, Green, and Blue variables are passed by reference.  The procedure will put the color values in these variables.  For example, you can use this procedure as follows:

Dim R As Integer  
Dim G As Integer  
Dim B As Integer  
GetRGB ActiveCell.Interior.Color, R, G, B  
MsgBox "The active cell has these color components:" & vbCrLf & \_  
    "Red: " & R & vbCrLf & \_  
    "Green: " & G & vbCrLf & \_  
    "Blue: " & B

         Other Color Functions  
  
The SumByColor function can easily be adapted to AverageByColor, MaxByColor, and MinByColor, as shown below:

Function AverageByColor(InputRange As Range, ColorIndex As Integer, \_

OfText As Boolean) As Variant

Dim Rng As Range

Dim Total As Double

Dim N As Long

For Each Rng In InputRange.Cells

If OfText Then

If Rng.Font.ColorIndex = ColorIndex Then

If IsNumeric(Rng.Value) Then

Total = Total + Rng.Value

N = N + 1

Else

AverageByColor = CVErr(xlErrNum)

Exit Function

End If

End If

Else

If Rng.Interior.ColorIndex = ColorIndex Then

If IsNumeric(Rng.Value) Then

Total = Total + Rng.Value

N = N + 1

Else

AverageByColor = CVErr(xlErrNum)

Exit Function

End If

End If

End If

Next Rng

If N = 0 Then

AverageByColor = CVErr(xlErrDiv0)

Else

AverageByColor = Total / N

End If

End Function

Function MaxByColor(InputRange As Range, ColorIndex As Integer, \_

OfText As Boolean) As Variant

Dim Rng As Range

Dim Max As Double

For Each Rng In InputRange.Cells

If OfText Then

If Rng.Font.ColorIndex = ColorIndex Then

If IsNumeric(Rng.Value) Then

If Rng.Value > Max Then

Max = Rng.Value

End If

End If

End If

Else

If Rng.Interior.ColorIndex = ColorIndex Then

If IsNumeric(Rng.Value) Then

If Rng.Value > Max Then

Max = Rng.Value

End If

End If

End If

End If

Next Rng

MaxByColor = Max

End Function

Function MinByColor(InputRange As Range, ColorIndex As Integer, \_

OfText As Boolean) As Variant

Dim Rng As Range

Dim Min As Double: Min = 1E+301

For Each Rng In InputRange.Cells

If OfText Then

If Rng.Font.ColorIndex = ColorIndex Then

If IsNumeric(Rng.Value) Then

If Rng.Value < Min Then

Min = Rng.Value

End If

End If

End If

Else

If Rng.Interior.ColorIndex = ColorIndex Then

If IsNumeric(Rng.Value) Then

If Rng.Value < Min Then

Min = Rng.Value

End If

End If

End If

End If

Next Rng

MinByColor = Min

End Function