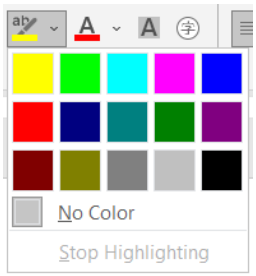
# 14 Editing – highlighting

Coloured highlighting (and in this I include the use of font colours) can provide another set of tools to aid the editor: you (or one of a number of macros) can mark bits of text in different colours with the colours having different significance. So, you need macros to allow you to easily add highlights of whatever colour, and then you need to be able to jump around the text, looking at the bits that are in different colours. Finally, you need the ability to get rid of the highlighting, either in a particular area of text, or selectively by colour – removing all the green highlighting, say, while leaving all the rest of the highlighting intact.

The following information, kindly provided by Robert Astle, will be helpful if you want to use different colours from the ones I have used in the following macros.

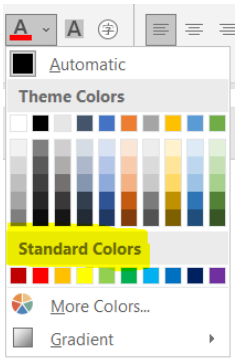
## *Highlight colour codes*




| | | | | |
|---|---|---|---|---|
| wdYellow | wdBrightGreen | wdTurquoise | wdPink | wdBlue |
| wdRed | wdDarkBlue | wdTeal | wdGreen | wdViolet |
| wdDarkRed | wdDarkYellow | wdGray50 | wdGray25 | wdBlack |

## *Standard font colours*

Font colours within macros seems to be a bit of a (multicoloured!) black art. We tried to offer the "Standard Colors" from the ribbon, but it's not as easy as that! So below we give a few suggested colours, and you can pick and choose.




| Sub ColourPlus() – Original version | Sub ColourPlus() – with "wdColor" prefixed to each of the Standard Colour names |
|---|---|
| Dim myCol(20)<br>myCol(0) = **wdColorAutomatic**<br>myCol(1) = **wdColorRed**<br>myCol(2) = **wdColorBlue**<br>myCol(3) = **wdColorGreen**<br>myCol(4) = **wdColorOrange**<br>myCol(5) = **wdColorLime** | Dim myCol(20)<br>myCol(0) = **wdColorAutomatic**<br>myCol(1) = **wdColorDarkRed**<br>myCol(2) = **wdColorRed**<br>myCol(3) = **wdColorOrange**<br>myCol(4) = **wdColorYellow**<br>myCol(5) = **wdColorLightGreen** |

| | |
|---|---|
| myCol(6) = **wdColorPlum**<br>myCol(7) = **wdColorSkyBlue**<br>myCol(8) = **wdColorPink**<br>myCol(9) = **wdColorBrightGreen**<br>myCol(10) = **wdColorGray25**<br>myCol(11) = **wdColorGray50**<br>myColTotal = 11 | myCol(6) = **wdColorGreen**<br>myCol(7) = **wdColorLightBlue**<br>myCol(8) = **wdColorBlue**<br>myCol(9) = **wdColorDarkBlue**<br>myColTotal = 9 |

## Add and remove highlights (basic)

These should be self-explanatory – mark some text and apply or remove highlights. (These first macros are very basic, so you could easily ignore them because you can do it all with the more powerful and flexible macros: 'Highlights – Pick any Colour!' That said, I do find 'Red Text On/Off' very useful.)

The first of the macros below adds highlight in the colour that is currently showing under the highlight icon on the toolbar; the second shows that you can choose to have a macro that highlights in one specific colour; the other three are for removing highlighting – that which is selected, the whole file and just the current line.

(The extra line in the three HiLightOFF macros – currently commented out – can be added if you want the same macro to both remove any highlighting *and* return any coloured text to normal, black text.)

```
Sub HiLightON()
```

```
Sub HiLightTurquoise()
```

```
Sub HiLightOFF()
```

```
Sub HiLightOffALL()
```

```
Sub HiLightOffCurrentLine()
```

Finally a macro that can, perhaps, be called from a DoMacro command in FRedit. It will remove all highlighting, text coloration and underlining.

```
Sub ClearHighlightAndColour()
```

## Red text on/off

Red text is probably the most commonly used text font colour (as it's the default colour that Word uses). This macro toggles the selected text between black (automatic) and red.

```
Sub ColourToggle()
```

## Highlights – pick any colour!

If you want to use a number of different colours of highlighting then these macros might help.

Basically, you have two macros – Plus and Minus. You can attach them to an icon on the toolbar or give them a keyboard shortcut, as you like.

At its simplest level, if you select some text and run Plus, it highlights the text in yellow, while Minus removes any highlighting.

However, if you click Plus a second time, the highlight colour you have just applied changes from green to yellow; click it again, and it goes to turquoise etc, etc. In fact, clicking Plus and Minus will take you forwards and backwards through the list of possible colours.

However, the 'list of colours', doesn't include all the colours that Word provides since some of them are virtually unusable for highlighting because they're so dark that you can hardly read the text. There's a list in the macro of the colours it uses, and you can fiddle around with it if you want to, adding or removing colours from the list.

If no text is selected when you run *HighlightPlus*, it selects – and therefore highlights – the current line , but if no text is selected when you run *HighlightMinus*, it ***removes*** the highlight from the current line.

(In what I just said about highlighting the current line, you can set the macro to highlight the current *paragraph* instead, by setting `doParagraph = True` at the beginning of the macro.)

Also, I've given *HighlightMinus* the extra facility that if the whole of the text is selected then it assumes you want to remove all the highlighting from the text (obviously!) but it also assumes that you want to remove any (a) font colouration, (b) all underlining, (c) any strike-through. (This is to prepare files ready to send back to my clients.)

If you don't want any of (a)–(c) then you can change the options to `False`:

```
colourOffToo = True
uLineOffToo = True
strikeThroughToo = True
```

*Colour choice*: The list of colours is set in the first few lines. You can have any colours you like, in any order you wish, though obviously you want the same list of colours for *HighlightMinus* as for *HighlightPlus*. To see how to change them, see under 'Text Fonts'.

*Keystrokes*: Thinking of Plus and Minus, I use Ctrl-Alt-NumPlus and Ctrl-Atl-NumMinus – the plus and minus on the numeric keypad being easy to get to, and if you're selecting a colour with Plus, and go too far, you can go back up through the list with Minus.

## Text font colour – pick any colour!

There are two equivalent macros that do exactly the same for text font colour, although the range of colours available is somewhat wider than for highlighting. However, I don't use font colouring much, so I just use four colours:

```
Dim myCol(20)
myCol(0) = wdColorAutomatic
myCol(1) = wdColorRed
myCol(2) = wdColorBlue
myCol(3) = wdColorPink
myCol(4) = wdColorBrightGreen
myColTotal = 4
```

*Keystrokes*: Again, I use plus and minus on the numeric keypad, but with Shift-Alt instead of Ctrl-Alt, so that's not too difficult remember.

Here are the four macros:

**`Sub HighlightPlus()`**

**`Sub HighlightMinus()`**

**`Sub ColourPlus()`**

**`Sub ColourMinus()`**

## Remove all highlight/colouration of current colour

This macro will remove all highlighting and text colouration, of one specific colour only. It senses what is at the cursor and removes that colour.

N.B. This macro removes the colouration of ALL text in the selected colour, so if any styles have a text colour as part of their definition, they will be changed to black.

```
Sub UnHighlightAndColour()
```

## Remove all highlighting

If you have text in text boxes [textboxes], selecting all the text with Ctrl-A and removing highlights doesn't work on the text boxes. This macros removes absolutely *all* the highlighting, including that in text boxes.

```
Sub HighlightAllOff()
```

## Remove all highlighting except...

The idea here is that you've used lots of different colours of highlighting in a text and you want to remove them all *except* one particular colour, say yellow, because that carries some special significance, maybe for the typesetters. This macro does that. Indeed, you can also use it to maintain *two* specific colours (named in the first lines of the macro. (Make the second one `keepColour2 = 0` if you only want to keep the one colour.)

This macro comes in two halves. That's because removing highlighting often generates loads of track changes: 'Formatted: Not Highlight'. So these track changes are then accepted.

```
Sub UnHighlightExcept()
```

## Remove all highlighting and colouration but not on program text

I use highlighting and colouration while editing, but I need to remove it before the author sees the edited text. However, for one job, the author had used coloured fonts and highlighting to draw attention to various things in the computer program listings within the text.

This macro removes all the font colouration and highlighting, but ***not*** if the text font is Courier New (the font used for the listings).

```
Sub SelectiveUnColourUnHighlight()
```

## Fixed spaces (and tabs) visible

(These two macros may well be superseded by the following one.)
It's sometimes useful to be able to see at a glance whether spaces are ordinary or fixed. This macro switches this highlighting on and off each time you run it.

The highlight colour is set in the first line – adjust to taste.

(The *HighlightSame* macro now perhaps makes this redundant. You select the space (or tab) and use *HighlightPlus* to add the highlight colour you want and then run *HighlightSame* which makes all the hard spaces (or tabs) the same highlight colour.)

```
Sub ShowFixedSpaces()
```

You can, of course, do the same thing in order to switch highlighted tabs on and off:

```
Sub ShowTabs()
```

# Highlight same text

This macro allows you to quickly highlight (or unhighlight) all occurrences of a word or phrase or symbol/punctuation mark. It can either use your preselected colours of highlighting, or, if you use *HighlightPlus* and/or *HighlightMinus*, it will use the colour you've just used.

The preselected colours are set at the beginning of the macro:

```
nonTextColour = wdGray25
textColour = wdBrightGreen
' colour for thin spaces
otherColour = wdYellow
```

– Click in a word and run the macro, and that word will be highlighted in `textColour` throughout the text. (But clicking in, say, 'the' will *not* highlight 'other' or 'theatre'.)

Run the macro again, and all occurrences of 'the' will be unhighlighted.

– Click to the left of, say, an endash and run the macro, and all en dashes will be highlighted in `nonTextColour`.

Run the macro again, and all en dashes will be unhighlighted.

– You can use it to highlight a phrase, or a part of a word: select the required text and highlight it with *HighlightPlus* and/or *HighlightMinus*, and then running *HighlightSame* will highlight all occurrences of that text in that same colour. So this last feature can also be used if you want to highlight a word, but in a colour other than `textColour`, i.e. select the word, highlight it in your chosen colour, then run *HighlightSame*.

I use it most often to make the thin spaces in a technical document easily visible:

$y = ax^2 + bx + c$

I can see immediately that a couple of the spaces are ordinary spaces, not thin ones.

(The macro should work with any non-alpha character, such as non-breaking hyphens and spaces, quotation marks, tabs etc. However, I did have to put in a fiddle to make it work with tabs and non-breaking hyphens, so if anything else you try doesn't work, please let me know.)

```
Sub HighlightSame()
```

# Font colour same text

This macro allows you to quickly font colour (or un-font-colour) all occurrences of a word or phrase. It can either use (a) the existing font colour (b) your preselected colour, or (c) if you have just used *ColourPlus* and/or *ColourMinus*, it will use that colour.

The preselected colour it uses is set at the beginning of the macro:

```
textColour = wdColorBlue
```

– Click in a word and run the macro, and that word will be highlighted in `textColour` throughout the text. (But clicking in, say, 'the' will *not* highlight 'other' or 'theatre'.)

Run the macro again, and all occurrences of 'the' will be unhighlighted.

– You can use it to colour a phrase, or a part of a word: select the required text and colour it with *ColourPlus* and/or *ColourMinus*, and then running *FontColourSame* will highlight all occurrences of that text in that same colour.
So this last feature can also be used if you want to highlight a word, but in a colour other than `textColour`, i.e. select the word, highlight it in your chosen colour, then run *FontColourSame*.

```
Sub FontColourSame()
```

## Remove highlight from this word

Removes highlight from all occurrences of the word at the cursor.

```
Sub HighlightOffWord()
```

## List all highlight colours used

For one application, I wanted to know which different colours of highlighting I'd used. This macro lists all the colours that *are* used in the current file:

BrightGreen
DarkYellow
Gray25
Gray50
Green
Pink
Red
Turquoise
Yellow

```
Sub HighlightLister()
```

## List all highlights used

If you want to use a highlight in a file and you want it to be a unique colour then I thought it would be helpful to list which colours have been used in the file so far, and more importantly which it does not have. This macro does so. As you see, it also shows you the colours:

BrightGreen
Gray25
Gray50
Pink
Red
Turquoise
Yellow

Blue
DarkBlue
DarkRed
DarkYellow
Green
Teal
Violet

So it shows you all the available colours (bottom) as well as those you have used (top).

```
Sub HighlightListerDeLuxe()
```

# Not highlight but background colour

Sometimes you get a file that *looks* as if it has used a highlight, but when you try to remove the highlight, it won't go. This might be because it's not actually highlighting but a change of background colour. There is a way of removing this (W2007/2010) by using Home–Theme Colors–No Color. However, weirdly, this only works provided that the text you select is within a single paragraph. If you select an area that goes over a paragraph end, the background colour remains untouched!

I've had a play with macros, and here's a (deceptively simple) macro that seems to do it:

```
Sub BackgroundColourOff()
```

If you select text within one paragraph, it removes the background from just that text. If you select an area of text that includes a paragraph end, the macro has to mess about selecting each of the paragraphs in turn, deselecting the paragraph end, and only then can it remove the background colour. Weird, but it was the only way I could get it to work after trying a range of different techniques.

However, I noticed that this was written in 2011, so I have done an updated version, *BackgroundColourAllOff*.

If you try this and find that it doesn't work (Word has a variety of ways of applying backgounds!) then please send me a sample file with your type of background, so that I can test it. Thanks.

[Later: I can confirm that this **doesn't** work on some files. If so, you'll have to use *BackgroundColourOff*, which may be slow on big files. Please use the 2020 version.]

```
Sub BackgroundColourAllOff()
```

# Highlight punctuation issues

This macro started when someone asked for a macro to highlight where some text in italic (e.g. the title of a book) is followed by a comma but the comma is also italic, when it should have been roman. Someone else pointed out that three book titles in a row probably had italic commas, but they too should have been roman. Then someone else added that a similar problem to the commas occurred with parentheses, colons, semicolons etc. And I added that it would be useful in a list that had bold headwords with colons and roman text to be able to highlight where the colons were bold (and the client wanted roman) or vice versa.

So this macro aims (says he, hopefully!) to highlight all of the above. However, at the beginning of the macro is a list of the various features so that you can switch them off (change 'True' into 'False') or change the colour used for each feature to your own favourite colour.

```
Sub PunctuationFormatChecker()
```

# Highlight punctuation issues (2)

Similar to the one above, this checks a wider range of issues.

I tested it out on a virtually completed PDF-on-paper proofread, and it found a couple of rogue colons, plus quite a number of italic commas I'd missed. It really appealed to my obsessiveness at finding mistakes, because it found some really obscure mistakes, some in the tiny writing. (I pity the poor person who has to implement my changes – one table has 45 very similar lines, each with the second comma correctly roman, but the first was incorrectly italic.)

The macro checks bold/italic colons following headwords:

**Headword – or words**: A word that needs to be defined.
**Headword – or words:** A word that needs to be defined.
*Headword – or words:* A word that needs to be defined.
*Headword – or words*: A word that needs to be defined.

You can specify independently whether colons should/shouldn't be bold/italic.

It also looks at italic commas, using various tests. Then if it's reasonably sure it's a rogue italic comma it highlights it in one colour, but if it's less certain, it highlights it in a lighter colour.

If anyone can think of any other punctuation formatting nuances you'd like it to check – or indeed any other kind of difficult-to-spot-by-eye (easy-to-miss-by-eye) errors, do please let me know.

**Similar type of macro**: *ParagraphEndChecker* alerts you to any paragraphs that don't have a terminal punctuation mark – so easy to miss.

`Sub HighlightOddPunctuationFormat()`

## Highlight roman punctuation following italic text

Sometimes, you might want the punctuation that follows some italic text to be italic also, e.g. "Pay *attention!*", he said. (I'm not here to argue typography; I just provide the tools!) So this macro will highlight roman punctuation that occurs after italic text, i.e. "Pay *attention*!", he said.

`Sub RomanPunctuationHighlight()`

## Highlight/colour all the words/phrases in a list

If you have a number of different words or phrases that you want highlighted throughout a text. Simply create the list and highlight and/or font colour each item in whatever colour(s) you fancy. This could have a range of applications, including drawing attention to weasel words, jargon, management speak etc.

some people
it is said
out of the box
at this point in time
currently
going forward

When you run the macro, it finds every occurrence of each of the words/phrases and adds the relevant highlighting and/or font colour.

To operate it, create the list as above in a separate file. Save the file with any name you like, as long as "list" (in any case) occurs somewhere within the filename (e.g. 'MyLiStGood'.)

You can run the macro with or without matching the case and with/without specifying whole words only. At the beginning of the macro is:

```
doWholeWordsOnly = True
doMatchCase = True
```

Set these as True or False, according to the application. (Remember that Word's 'Whole word only' only works when you F&R a single word, not a phrase.)

If you put a '#' on one line, e.g.

some people
it is said
#
out of the box
at this point in time
currently
going forward

it will do the first two, but then it will stop when it gets to the hash, and ignore the rest of the list.

**Summary of instructions:**
– Open a new file, type in a list of words/phrases, colouring and/or highlighting each of them as you like.
Save this Word file as, say “MyWordList”.

– When you want to run the macro, all you do is make sure that “MyWordList” is open on screen, open the file you want to highlight/colour and run the macro.

So to operate the macro, you need:
– Two open files: your word list + the text you want to highlight, but make sure that you put the cursor in the text you want to highlight, not in the word list. Then run the macro.

```
Sub HighlightWordList()
```

# Highlight all the edits in a text

You can use CHangeNext and ChangePrevious to jump about between the edits in a text, but if you just want to see clearly where there are some edits in the text, then this macro highlights all the edits.

```
Sub RevisionHighlight()
```

# Adding borders to text

*(Video: youtu.be/2PG7n5MCMCo)*

Someone asked if it was possible to add borders to specific words or phrases by using find and replace. They wanted to be able to do something like this:

e.g.Video about macros narrated by Paul Beverley

I tried but I couldn’t find any way to do this directly using find and replace. What I worked out instead is that you can specify the text to be ‘bordered’ by applying a specific combination of attributes that won’t appear anywhere else in the document, and then use a macro to convert those specific attributes to borders (of various shapes, styles and colours). I chose underline plus various colours of highlight to represent the different border styles.

Then you can use FRedit to apply the underline + highlight. So to achieve the bordering above, I’d use:

macros|^&
Paul Beverley|^&

and of course, since FRedit can run macros, you could do it all at one fell swoop:

macros|^&
Paul Beverley|^&
DoMacro|BordersAddToText

The code in the macro converts different highlight colours to various different border styles:

> From Word 2007 onwards, the functionality of *InstantFind* is probably redundant because of the more advanced find facilities now available. However, the *FindFwd*, *FindBack*, *FindFwdCase* and *FindBackCase* may still be very useful. Also, if you crave being able to jump advanced F&R dialog, try the *FindAdvanced* macro. When working with text, you want to be able to move around it, looking at various bits, checking them and changing them, so in this section there are tools to allow you to jump instantly to another heading of this.

I've deliberately chosen a range of different line style, thicknesses and colours, but you can change them, if you like.

Within the macro, you'll see a some lines of computer code where the links are established. I think they are reasonably self-explanatory. So the link turquoise –> 'blue 150pt single border'is established with:

```
Case wdTurquoise:
  With Options
    .DefaultBorderLineStyle = wdLineStyleSingle
    .DefaultBorderLineWidth = wdLineWidth150pt
    .DefaultBorderColor = wdColorBlue
  End With
```

And for light gray –> 'red 225pt dotted border' there is:

```
Case wdGray25:
  With Options
    .DefaultBorderLineStyle = wdLineStyleDot
    .DefaultBorderLineWidth = wdLineWidth225pt
    .DefaultBorderColor = wdColorRed
  End With
```

You can play with these, and the others, to your heart's content.

**Sub BordersAddToText()**

## Removing borders from text

This macro provides a way to remove borders from bits of the text. If you select an area, it will remove the borders only from that bit of text. If no text is selected, it will assume that you want to remove the borders from the whole file – though it will ask for confirmation before doing so.

**Sub BordersAllOff()**

## Splitting and joining paragraphs

Suppose you have some dialogue:

> "I don't think that was a concern back then." Katja said.
> Wim said.
> "I was not expecting that." They all stared...

and you want:

> "I don't think that was a concern back then." Katja said.
> Wim said. "I was not expecting that."
> They all stared...

Simple! Click in 'said' (the one I've underlined for identification) and run the macro. Click in 'that' (as underlined) and run the macro.

In other words, if you click in the last word on the line, the macro pulls the line below it up, to make it part of the same paragraph (and adds a space, of course). Then if you click in a word in the the middle of a line, the macro splits the paragraph (and deletes the stray space).

**Sub ParaSplitJoin()**