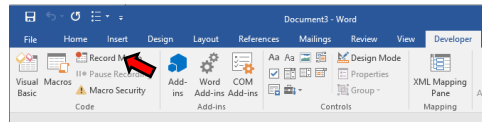


## Week8: Second Tutorial

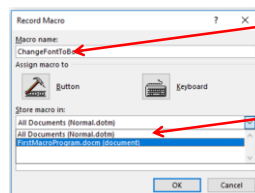
- (First tutorials: Monday, no classes due to Reading Days, Tuesday, open tutorial)
- Recording macros in Word
- Discussion of Assignment 3 requirements

## Recording Macros

- Developer ribbon
  - “Record Macro”



- Recording details

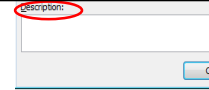


What to name the macro

Where to store the macro

Author: James Tam

## Naming The Macro: Conventions



- Part of your assignment marks will be awarded according to Macros should be given a good self-explanatory name: describes the purpose of the program e.g., 'formatting\_resume\_headings'
  - Additional information about the program can be provided in the 'description' field but for this class we will do this using 'program documentation' (described later).
  
- Requirements for naming the macro:
  - Must start with an alphabetic letter, after than any combination of letters and numbers may be used
    - OK: "assignment1", "a2939" Not OK: "1assignment", "\*assignment"
  - Maximum length of 80 characters
  - It cannot contain spaces, punctuation or special characters such as # or !
    - 'resume headings' (Not Allowed: space character)
    - 'macros!' (Not Allowed: special character)
  - Can contain underscores (separate long names)

Author: James Tam

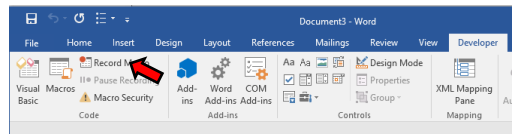
## The First Simple Macro

- With word processing there's sometimes a need to apply multiple formatting styles (bold, italics, underline) to highlighted text
- Manually applying the required formatting to each block of text can be tedious
  - Recall: Macros can be used to automate or shorten some tasks
- This first example macro program will be used to show:
  - How to create a VBA macro for MS-Word
  - How to automate a task using a macro

Author: James Tam

## Recording A Simple Macro

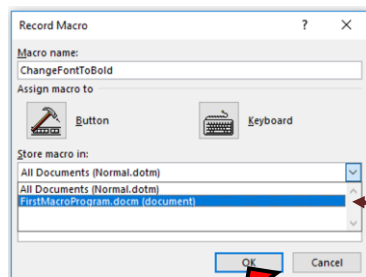
- (Of course a macro isn't needed to use this formatting effect but it's easiest to start with a simple example).
- Bold face highlighted text.
  - Select the developer tab and press record



Author: James Tam

## Recording A Macro (2)

- Give the macro a self explanatory name and press 'OK' in order to begin recording .

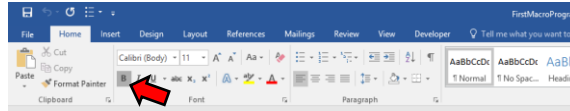


- Note: Make sure you record the macro in the **current document** and not **"All documents"** (Important!)

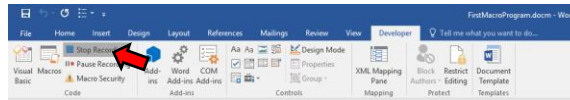
Author: James Tam

## Recording A Macro (3)

- Select whatever options you want to add to the recording of the macro
  - In this case you would select bold font



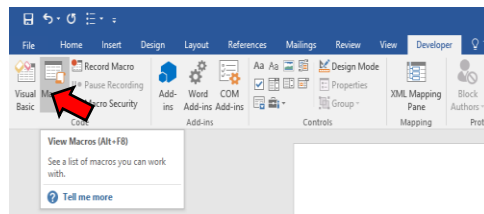
- All commands have been entered so you can stop the recording



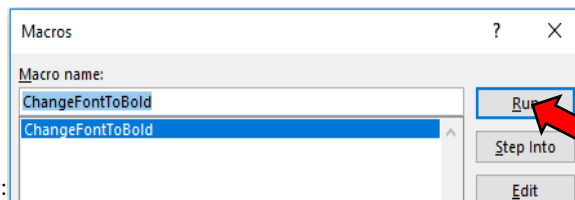
Author: James Tam

## Running A Recorded Macro

- Under the Developer ribbon select 'macros'



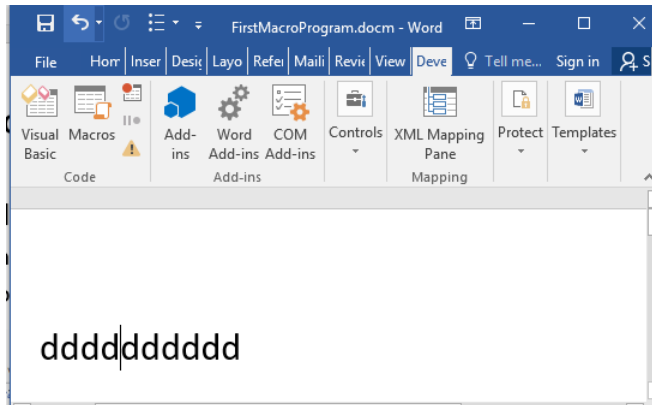
- Select the macro and then 'run' it



Author:

## Running A Recorded Macro (2)

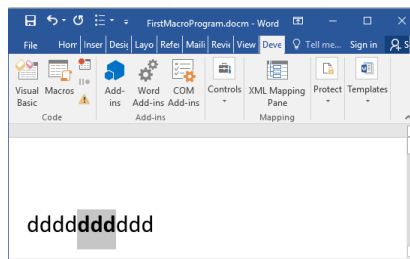
- In this case nothing happened?
  - This macro changes selected/highlighted text to bold
  - You need to select some text before running the macro



Author: James Tam

## Running A Recorded Macro (3)

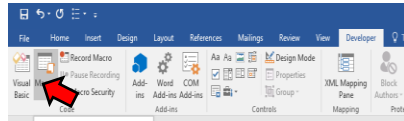
- After selecting the text and running the macro again, whatever text was highlighted now becomes bold.



Author: James Tam

## Recording Macros: Additional Comments

- Don't rely on creating all your macros by recording them.



- Drawbacks:
  - (Problem in terms of this class) to demonstrate your understanding of concepts you will be asked to manually write VBA code
    - You won't be adequately prepared if you rely on automatically recording your programs
  - (Problem with doing this in real work) The automatically generated program code automatically is larger and more complicated than is necessary
    - "Bloated" code
    - Look online under search terms such as *+bloated "vba code" +recorded* for examples of why automating recording VBA programs can be problematic.

Author: James Tam

## Auto Generated VBA Program: 24 Lines

```
Sub heading()
  With Selection.ParagraphFormat
    .LeftIndent = InchesToPoints(0)
    .RightIndent = InchesToPoints(0)
    .SpaceBefore = 0
    .SpaceBeforeAuto = False
    .SpaceAfter = 6
    .SpaceAfterAuto = False

    .OutlineLevel = wdOutlineLevelBodyText
    .CharacterUnitLeftIndent = 0
    .CharacterUnitRightIndent = 0
    .CharacterUnitFirstLineIndent = 0
    .LineUnitBefore = 0
    .LineUnitAfter = 0
    .MirrorIndents = False
    .TextboxTightWrap = wdTightNone
  End With
End Sub
```

Author: End Sub

## VBA Statements *Actually Needed*: 1 Line

```
Sub headingManual()  
    Selection.ParagraphFormat.SpaceAfter = 6  
End Sub
```

Author: James Tam

## Recording Macros: Additional Comments

- Benefits:
  - You can use the macro code that is automatically generated *in order to learn* how to do manually.
  - Sometimes this is very useful if you don't know the wording of a command or how to access a property.
  - Example (VBA program code for the previous example)
    - Record the commands for the macro
    - Then view the commands so you can learn how to do it manually

```
Sub AutoGeneratedFontChange()  
    Selection.Font.Bold = wdToggle  
End Sub
```

Author: James Tam

## Recording Macros

- Bottom line: use it for learning “how to do” things
- Don’t:
  - Just use the auto-generated code to study for the exam without creating any code of your code
  - Just hand in the auto-generated VBA code for your assignment
- While taking this course: Use the auto-generated code to figure out how to “type the program from scratch” yourself (I will show how to do this shortly)
- After this course is done: if ever you find your usage of Office tedious and repetitive (multiple clicks) then you can record all those steps into one macro!

Author: James Tam

## Recording Macros: Student Exercise

- You can use your own Word document so long as it has text that you can use.
- If you don’t have a document handy:
  - From this week’s tutorial link: Download the Word document called “document\_for\_student\_macro\_recording\_exercise1.docm”
- Within this document record a macro called “student\_macro1” that will complete the following tasks:
  - Toggle font to/from bold
  - Change font size to 24 point
  - Change the font type to ‘Consolas’

Author: James Tam



## Final Hint

- If you run the macro and 'nothing happens' keep in mind that this macro formats the currently selected text.
- In order for the changes to occur some of the text in the document must first be selected.

Author: James Tam

## Assignment 3

- Unlike the previous two assignments where you learned how to use pre-created programs (A1 = Excel spreadsheet, A2 = Access database) you will learn how to write/create a small VBA program for MS-Word.
- You will manually enter the program for A3 (cannot be automatically recorded)

Author: James Tam

## Feature #1

- Find and replace a style:
  - All instances of “Heading 1”

### Figures in this document

#### Figure

Figure 1: Japanese garden .....

Figure 2: Ages ago when I still used to

-

### Figures in this document

#### Figure

Figure 1: Japanese garden .....

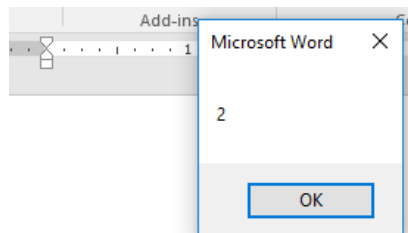
Figure 2: Ages ago when I still use

Figure 3: HING WO Shaolin's drag

Author: James Tam

## Feature #2

- Count and display the number of typographical errors
- Example



The kat in the hawt

Author: James Tam

## Feature 3a

3. Evaluate the quality of the document. (**Max of 1.7 GPA**). In order to get credit for this feature, Feature #2 must be correctly implemented first. Also the document status (REJECTED, NEUTRAL, SUPERIOR) must clearly appear at the top of the document or you will be awarded reduced credit or even no credit for this feature.

a. If there are any spelling mistakes in document, then the document is automatically rejected (text = "REJECTED" which is written to the top of the document) (**+0.3 GPA**).

- Typos in the document:
  - "REJECTED" added to the top of the document)

REJECTED

Mary had a little faawn

Author: James Tam

## Feature 3b(i)

- No typos
  - Zero instances of the search word entered by the user
  - E.g. the user entered "tesseract" into the document below

NEUTRAL

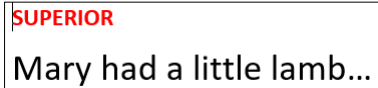
Mary had a little lamb...

- No typographical errors found but no instances of the word "tesseract" found in the document

Author: James Tam

## Feature 3b(ii)

- No typos
  - One or more instances of the search word entered by the user
  - E.g. the user entered “mary” sd(non case sensitive) into the document below

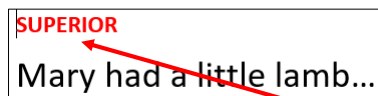


**SUPERIOR**  
Mary had a little lamb...

Author: James Tam

## Feature 3c

- Visually highlight the text that's written to the top of the document.



**SUPERIOR**  
Mary had a little lamb...

- **Bold font**
- **14 point size**
- **Colored red**

Author: James Tam

## Feature 4

- (Use the appropriate VBA library command to attempt to print the document)
- It may or may not result in a print dialog coming up but your marker will be able see if the proper call to the print command was included in your program

Author: James Tam

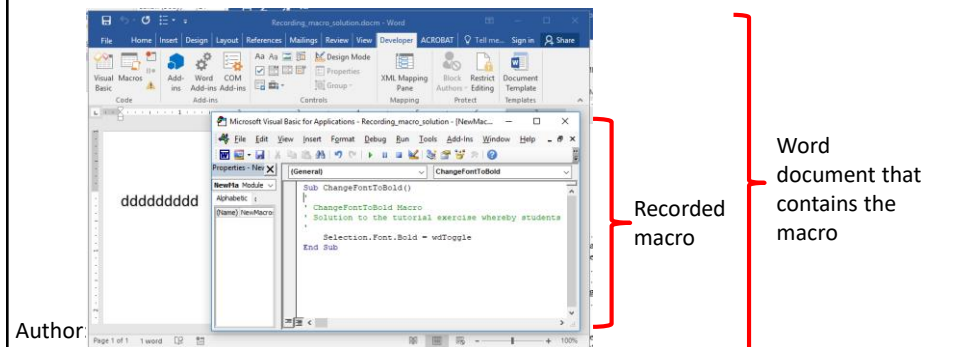
## Feature 5

- Automatically save and close the document (in order to have the previous features take effect a document needs to be opened)

Author: James Tam

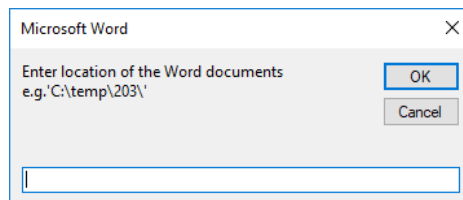
## Features #1 - #5

- To get credit #1 – 5 the VBA program will apply these features on the Word document in which the VBA program is contained within.
- Example: your first recorded macro was recorded in a Word document



## Feature #6

- Instead of just running the previous features on a single document, the program will invoke Features #1 – 5 on all the Word documents in a specified location.
- Prompt the user for a location.



Author: James Tam

## Feature #6

- The program will check the validity of the location (e.g. not an empty location/path or an invalid location).
- Given that the location is valid then Features #1 - #5 will then be applied to all the Word documents in this folder

Name	Date modified	Type	Size
limeric.docx	11/6/2017 5:15 PM	Microsoft Word D...	12 KB
resume.doc	11/6/2017 5:28 PM	Microsoft Word 9...	26 KB
Test document.docx	11/6/2017 5:27 PM	Microsoft Word D...	0 KB

Author: James Tam

## Outline Of Feature #6

- Prompt the user for the location
- Check the validity of the location and react accordingly
- As long as there is another Word document in the location
  - Open a document that has not yet been opened by the VBA program
  - Run Features #1 - 5 on the document
    - Style find and replace
    - Count the number of typographical errors,
    - Write the appropriate text at the document)
    - Print the document
    - Save and automatically close the document
  - Move onto the next document

Author: James Tam

## Caution!

- Many students find this assignment the most challenging of all the assignments.
- Start working on whatever features that you have learned in lecture (and tutorial) as soon as possible.
  - Don't wait until the last minute
  - Backup and submit your work to D2L on a regular basis (as shown in the lecture notes it's an assignment requirement to have a versioning system).
- Make sure you understand the feature (e.g. displaying a popup box to the user as the program runs) **before** attempting to add the feature to the assignment.
  - Trying to learn VBA at the same time you are trying to implement the features of the assignment will make your work nearly impossible.

Author: James Tam

## Reminder!

- No group work is allowed for this class.
- Each student must submit their own work.
- Students should not see the solution of the other students (conversely you should not make your solution available to others).
- (In the above scenario the student providing the solution and the student who views the solution could be investigated for academic misconduct).
- You do not want to be investigated as a possible case of academic misconduct.
- Additional information:
  - <http://pages.cpsc.ucalgary.ca/~tamj/2017/203F/assignments/misconduct.html>

Author: James Tam