VBA: Tutorial Week 1

- Recording a macro
- Documenting a program
- Getting user input via a MsgBox
- Display output using an InputBox
- Storing information with common types of variables in VBA
- Common VBA operators

Official resource for MS-Office products: https://support.office.com

Activities In Tutorial

- TA demos:
  - Used for more complex features (typically multiple steps are required).
  - The tutorial instructor will show on the projector/instructor computer each step for running the feature in Excel.
  - Unless otherwise specified the tutorial material will take the form of a TA demonstrating the use of features in Excel.
  - Slides titled "Lecture Review" are covered for the second time and dealing with less complex material.
    - For this reason they will only be covered briefly in tutorial.
- Student exercises:
  - Used instead of TA demos for simpler features.
  - You will have already been given a summary of how to invoke the feature and the purpose of the exercise is to give you a chance to try it out and get help if needed.

Tutorial notes by James Tam

How To Create VBA Programs

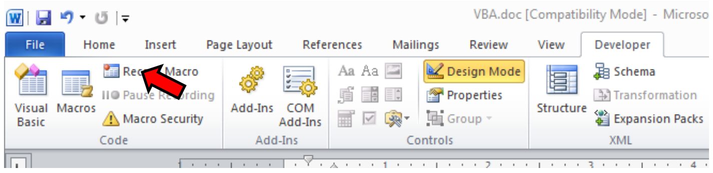
This topic: Shifted from the lecture material.

1. **Method 1**: Record the macro automatically: keystrokes and mouse selections will be stored as part of the program
2. **Method 2**: Manually enter the program (type it in yourself into the VBA editor)

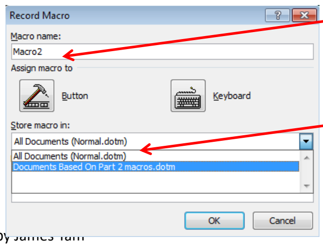
Tutorial notes by James Tam

Method 1: Recording A Macro

- Developer ribbon
  - Click on "`Record Macro`"



  - Recording details



Macro name: Macro2

Store macro in: All Documents (Normal.dotm)

All Documents (Normal.dotm)

Documents Based On Part 2 macros.dotm

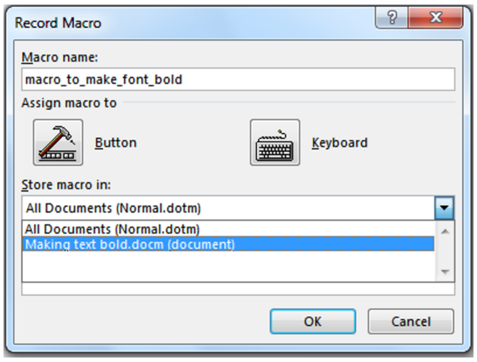
What to name the macro (next slide for more info)

Document to store the macro (select the current Word document)

Tutorial notes by James Tam

Recording A Macro (2)

- Give the macro a self explanatory name and press 'OK' (recording begins).

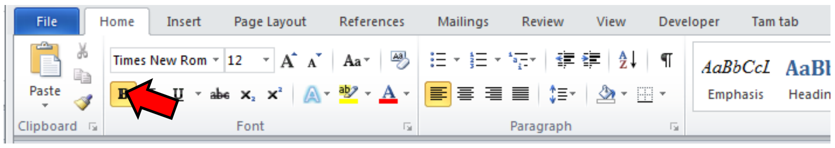


- **Important reminder: record the macro in the current document and not "All documents" (Important!)**

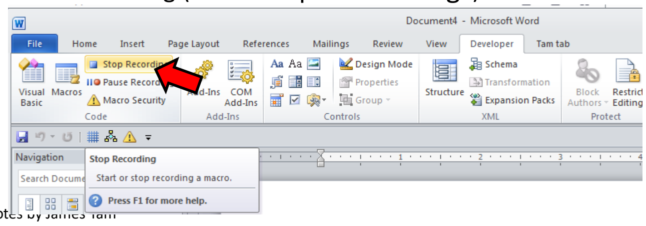
Tutorial notes by James Tam

Recording A Macro (3)

- While recording you can run whatever Word features you want to add to the recording of the macro
  - In this case you would select bold font



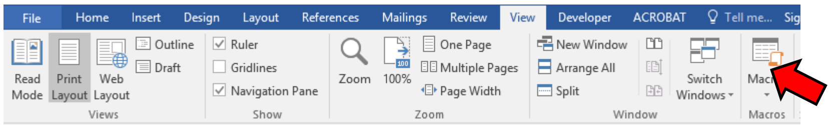
  - For this simple example all commands have been entered so you can stop the recording (click "Stop recording")



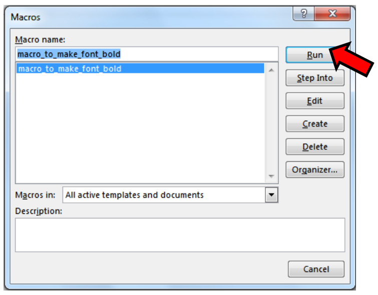
Tutorial notes by James Tam

Running A Recorded Macro

- Under the 'View' ribbon select 'Macros'



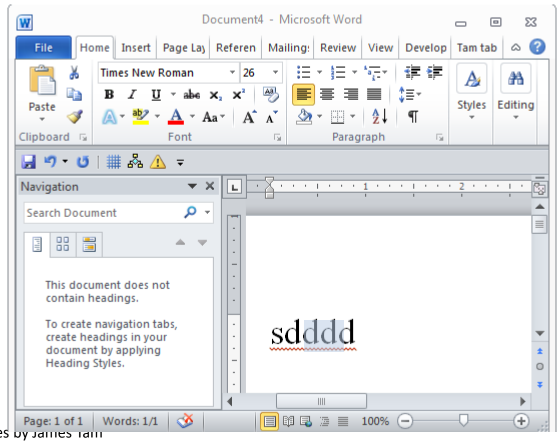
- Select the macro (Under "Macro name") and then click 'Run'



Tutorial notes by James Tam

Running A Recorded Macro (2)

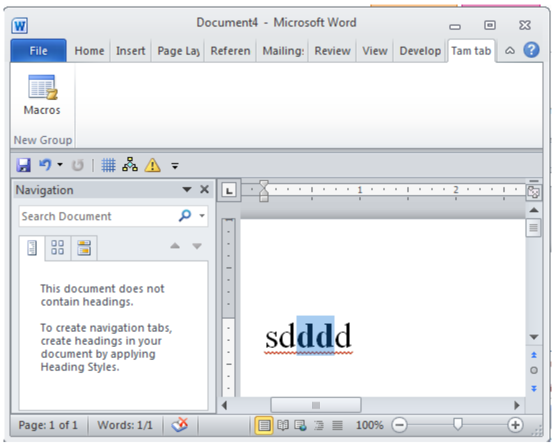
- In this case nothing happened?
  - This macro changes selected/highlighted text to bold
  - You need to select some text before running the macro



Tutorial notes by James Tam

Running A Recorded Macro (3)

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



This document does not contain headings.

To create navigation tabs, create headings in your document by applying Heading Styles.

sdddd

Tutorial notes by James Tam

Other Examples Of Useful (?) Recorded Macros

- Printing: selecting a printer and setting print options can be tedious if your print driver application does not save your previous selections.
  - Even if previous selections are saved recording a macro may be useful if you have multiple printing profiles e.g. I print low quality black and white double sided documents stapled on the top left for staff and high quality color single sided with multiple staples with glossy paper for clients.
- Entering hard to spell words (especially if they are new)
  - Example: “Gotterdammerung” is hard enough to spell but this word is supposed to have the “umlaut” for the ‘o’ (actually ö) and ‘a’ (actually ä).
  - JT: shifting to German keyboard in Word can be done via: `<Ctrl>-<shift>-<;>` (this key combination must be entered prior to typing *each* alternate character).

Tutorial notes by James Tam

Method 1: Recording Macros (Comments)

- It's a good and simple way for anyone to automate a tedious series of commands in Word.
- This approach is quite limited.
  - E.g. how would you record a macro to automatically open, edit and print all the documents in a folder? (You can't).
  - The main goal in introducing the recording approach is to make you aware of this capability for your future use. (Automating tedious tasks).
- For the assignment this approach will not work.
- Also on the exam you must manually write macros (on paper) or trace a macro (figure out what happens when it runs).
  - "Recording" is not feasible for the exam

How To Create VBA Programs For This Course

1. **Method 1**: Record the macro automatically: keystrokes and mouse selections will be stored as part of the program
2. **Method 2**: Manually enter the program (type it in yourself into the VBA editor)
   - This is how you are to complete your assignment and is how many VBA programs are created.
   - Consequently the remainder of the course notes will focus on Method 2

Microsoft Introduction/Overview Of VBA

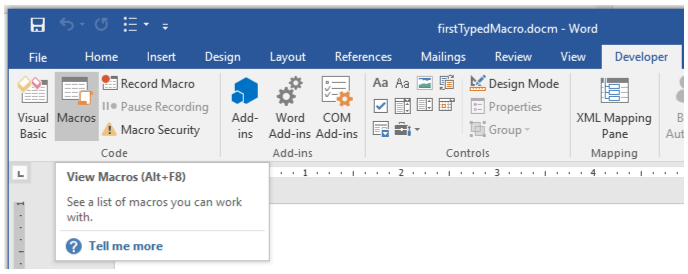
- https://docs.microsoft.com/en-us/office/vba/library-reference/concepts/getting-started-with-vba-in-office

Creating Macros

- The focus on this course will be creating macros by typing them into the VBA editor (not recording them)

Step 1: Finding The Macro Feature

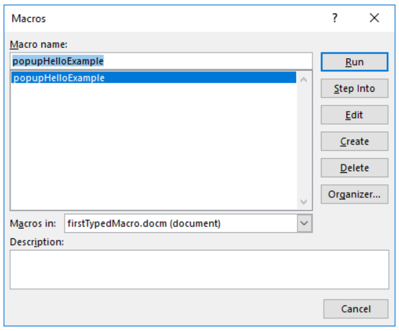
- To enter a new macro/view previously created macros it's the same as viewing a recorded macro: "Developer->Macro"



Tutorial notes by James Tam

Step 2: Naming And Saving The Macro

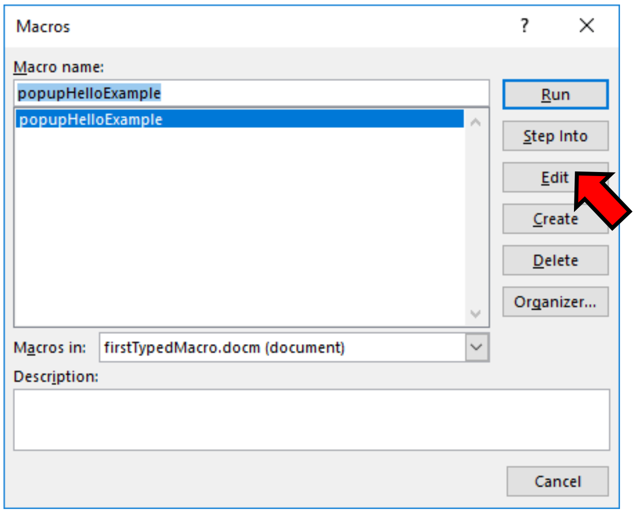
- Enter the name "Macro name" using a good self-descriptive name (small examples that don't carry out a useful function such as this example are more challenging to name).



- Important! Make sure you **save the macro in the current document** and not all documents ("`Normal.dotm`")

Tutorial notes by James Tam

Step 3: Select The Edit Option



popupHelloExample

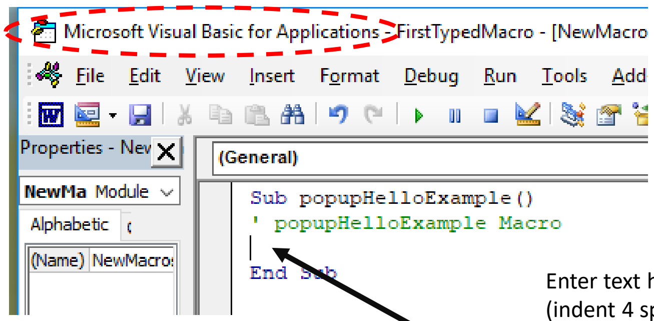
popupHelloExample

Macros in: firstTypedMacro.docm (document)

Tutorial notes by James Tam

Step 4: Enter the Macro In The VBA Editor

- The **VBA editor** is not the same as the Word editor



```
Sub popupHelloExample()
' popupHelloExample Macro

End Sub
```

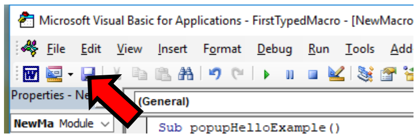
Enter text here
(indent 4 spaces)

```
MsgBox("Hello")
```

Tutorial notes by James Tam

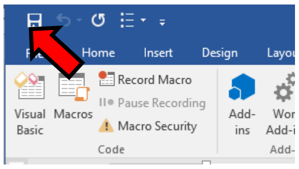
Step 5: Saving The Macro

- Typically if you either save via the VBA editor…


Microsoft Visual Basic for Applications - FirstTypedMacro - [NewMacro
File Edit View Insert Format Debug Run Tools Add
Properties - Ne
(General)
NewMa Module

```
Sub popupHelloExample()
```

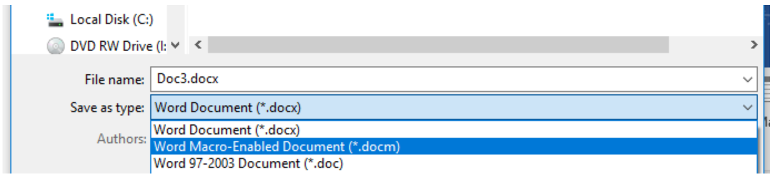
- …or the Word editor then the macro will be saved.



Tutorial notes by James Tam

Step 5: Saving The Macro

- To be safe you can save using both editors but in any case make sure you save the Word document as a “Word Macro-Enabled Document” (*.docm)



File name: Doc3.docx

Save as type: Word Document (*.docx)

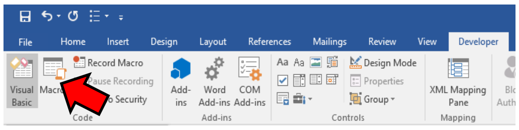
Authors:

Word Document (*.docx)
Word Macro-Enabled Document (*.docm)
Word 97-2003 Document (*.doc)

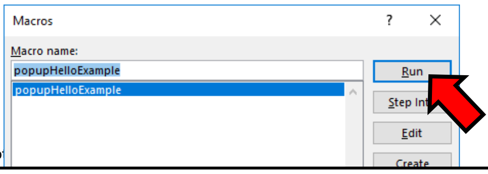
Tutorial notes by James Tam

Running The Macro

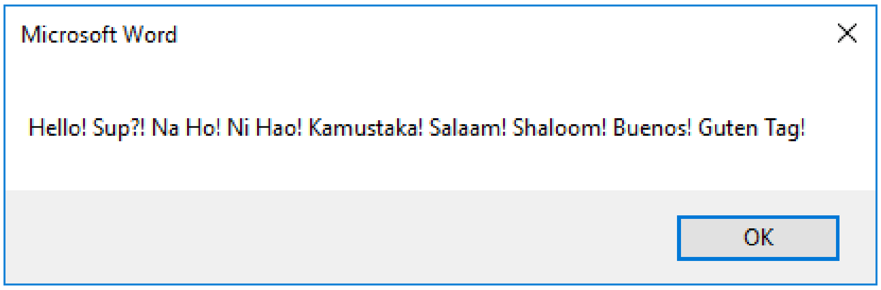
- Regardless of how a macro has been created:
  - Automatically recorded
  - Manually entered
- A macro will be run using the same series of steps.
- Developer->Macros



- The single macro should be highlighted, then click ‘run’



Result Of Running The Example Macro



Microsoft Word

Hello! Sup?! Na Ho! Ni Hao! Kamustaka! Salaam! Shaloom! Buenos! Guten Tag!

JT: if you need to see the solution to the exercise look at the document: `1firstTypedMacro.docm`

Student Exercise #1

- Writing and running a macro that displays your name
- Name to give your macro: ‘`firstMyName`’
- Name of macro-enabled Word document ‘`exercise1_first_typed_macro_student_exercise.docm`’

Tutorial notes by James Tam

Program Documentation

- Doesn’t contain executable instructions.
- Created for the reader of the program (other programmers, in this class it’s for your assignment marker)
- **Format**:

```
' <Documentation>
```

- **Example**:

```
' Author: James Tam
```

- No error: Everything after the quote until the end of the line will not be translated into machine language/binary
- That means documentation doesn’t have to be a valid and executable instruction

Tutorial notes by James Tam

Program Documentation: Assignments (A3)

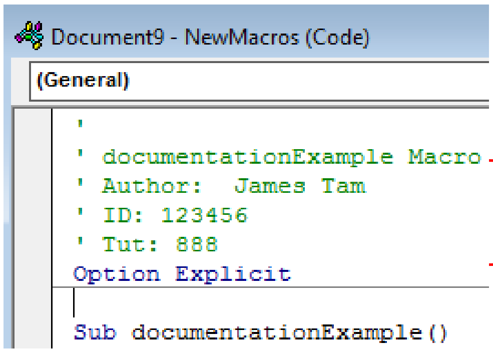
These requirements also apply to any other full assignments that involve program writing (if applicable).

- Your VBA assignment submission must include identification about you and information the features of your program
  - Full name
  - Student identification number
  - Tutorial number
  - List the program features (from the assignment description) and clearly indicate if the feature was completed or not completed.
  - Program version

Tutorial notes by James Tam

Location Of Program Documentation

- Contact information should be located before your program
- Before the 'sub' keyword
  - Before 'Option explicit' if included (more details on this later for now just take it as a given that including this is a good idea).


Document9 - NewMacros (Code)

(General)

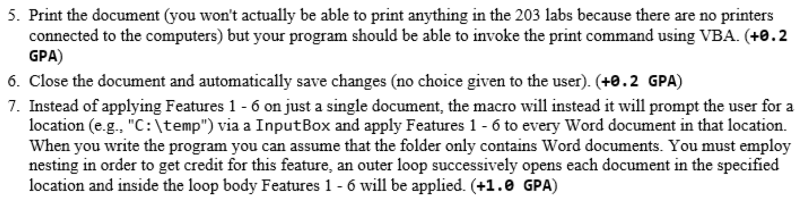
```
'
' documentationExample Macro
' Author:  James Tam
' ID: 123456
' Tut: 888
Option Explicit

Sub documentationExample()
```

Documentation: marked in red

Tutorial notes by James Tam

Program Documentation: A3 Documenting Program Features

- Program features (this will be worth many marks)
- Example assignment description



5. Print the document (you won't actually be able to print anything in the 203 labs because there are no printers connected to the computers) but your program should be able to invoke the print command using VBA. (**+0.2 GPA**)
6. Close the document and automatically save changes (no choice given to the user). (**+0.2 GPA**)
7. Instead of applying Features 1 - 6 on just a single document, the macro will instead it will prompt the user for a location (e.g., "`C:\temp`") via a `InputBox` and apply Features 1 - 6 to every Word document in that location. When you write the program you can assume that the folder only contains Word documents. You must employ nesting in order to get credit for this feature, an outer loop successively opens each document in the specified location and inside the loop body Features 1 - 6 will be applied. (**+1.0 GPA**)

- Program documentation
  - ' Author: James Tam ID: 123456
  - ' Version: Nov 2, 2015
  - ' Tutorial: 99
  - ' PROGRAM FEATURES
  - ' #5: Printing: completed
  - ' #6: Close and save: not completed
  - etc

Tutorial notes by James Tam

Program Versioning: What Students Need To Document In Order To Be Awarded Credit

A3.docm

```
' Version: Sept 20, 2012
' Program features completed:
' (5) Print
' (6) Save & close
```

Shows:
- Versioning system employed
- Versioning tied to a list of which features completed for that version.

Tutorial notes by James Tam

Variables

- A location in memory used to store information temporarily.
- At most a variable can store a single piece of information.

```
num = 1
```

```
num = 17
```

Tutorial notes by James Tam

Creating/Declaring Variables

- Creating and naming memory locations
  - `Dim anInt As Long`
  - `Dim aReal As Double`
  - `Dim aString As String`

- Assigning values to those locations
  - `anInt = 2000`
  - `aReal = 3.14`
  - `aString = "Peter Griffin"`

Tutorial notes by James Tam

Displaying The Contents Of Variables

- Format:
  - `MsgBox (<Name of the variable>)`
- Example (assumes a variable called 'age' has already been created and a value has been assigned to it)
  - `MsgBox (age)`
- Question for students: What's the output of the MsgBox

```
Dim name As String
name = "Smith"
MsgBox (name)
```

Tutorial notes by James Tam

Explanation Of Output

- When a message box can display a constant string the string must be enclosed in double quotes
  - E.g. `MsgBox("hello")`
- When a message box displays the contents of a variable then the name of the variable is not enclosed in quotes
  - E.g. `MsgBox (age)`

Tutorial notes by James Tam

Mixed Output: Strings And Variables

- Each field must be separated with and connected with an ampersand ‘&’
- **Format**:
  - `MsgBox ("<string>" & variable & "<string>"….)`
- **Example**:
  - `MsgBox ("Age=" & age)`

Tutorial notes by James Tam

Mixed Output Example

- **`2variablesMixedOutput.docm`**

```
Sub variablesMixedOutput()
    Dim age As Long
    Dim name As String
    age = 37
    name = "Homer J. Simpson"
    MsgBox ("Name: " & name & ", " & "Age: " & age)
End Sub
```

Tutorial notes by James Tam

Getting Input

- Getting information from the user as the program runs via an `InputBox`
- **Format**:
  - `InputBox ("<`*`Prompting message`*`>")`
- **Example**:
  - `InputBox ("Tell me your name")`

Tutorial notes by James Tam

Input Example

- **`3inputExample.docm`**

```
Sub inputExample()
    Dim age As Long
    Dim name As String
    age = InputBox("Tell me your age: ")
    name = InputBox("What is your name: ")
    MsgBox ("Name: " & name & ", " & "Age: " & age)
End Sub
```

Notice how the values displayed in the MsgBox will vary in this example according to user input

Tutorial notes by James Tam

Student Exercise #2

- Write a new VBA program in a new Word document
- Declare a String variable called ‘name’
- Write a program that will ask the user for their name using an `InputBox` and store the input in the variable called ‘`name`’
- The program will then display the value entered by the user using a MsgBox

Tutorial notes by James Tam

Solution #2

- `Dim name As String`
- `Name = InputBox("Name: ")`
- `MsgBox(Name)`
- `Complete solution: exercise2_get_display_name`

Tutorial notes by James Tam

Microsoft Introduction/Overview Of VBA

- https://docs.microsoft.com/en-us/office/vba/library-reference/concepts/getting-started-with-vba-in-office

Tutorial notes by James Tam

Basic VBA **Operators**

Operation	Symbol used in VBA	Example
(Same as Excel)		
Addition	**+**	2 **+** 2
Subtraction	**-**	3 **–** 2
Multiplication	*****	10 ***** 10
Division	**/**	81 **/** 9
Exponent	**^**	2 **^** 3
(Different operator)		
Concatenation	**&**	“hi” & “there”

Tutorial notes by James Tam

Common VBA Operators

4operators.docm

```
Sub operators
  ' Part I
  doubleNum1 = 2 + 2
  MsgBox (doubleNum1)
  doubleNum1 = 7 * 13
  MsgBox (doubleNum1)
  doubleNum1 = 15 / 2
  MsgBox (doubleNum1)
  doubleNum1 = 2 ^ 4
  MsgBox (doubleNum1)
  aString = "the" & "cat" + " in" & "- hat"
  MsgBox (aString)
```

Tutorial notes by James Tam

Common VBA Operators (2)

```
   ' Part II
   doubleNum1 = 2
   doubleNum2 = 3
   doubleNum3 = doubleNum1 ^ 3
   doubleNum1 = doubleNum1 * doubleNum2
   doubleNum2 = doubleNum1 + doubleNum3
   MsgBox ("doubleNum1=" & doubleNum1 & "-" &
     "doubleNum2=" & doubleNum2 & "-" & "doubleNum3=" &
     doubleNum3)
End Sub
```

Tutorial notes by James Tam

Student Exercise #2

- Write a new VBA program in a new Word document
- Declare a String variable called 'name'
- Write a program that will ask the user for their name using an `InputBox` and store the input in the variable called '`name`'
- The program will then display the value entered by the user using a MsgBox

Tutorial notes by James Tam