Excel Expert

Session 1

Instructor: Don Bremer

Excel Templates

Instead of doing a lot of the same work over again – use a template to off load some of your work for you.

File->New-> Loan Amortization Schedule

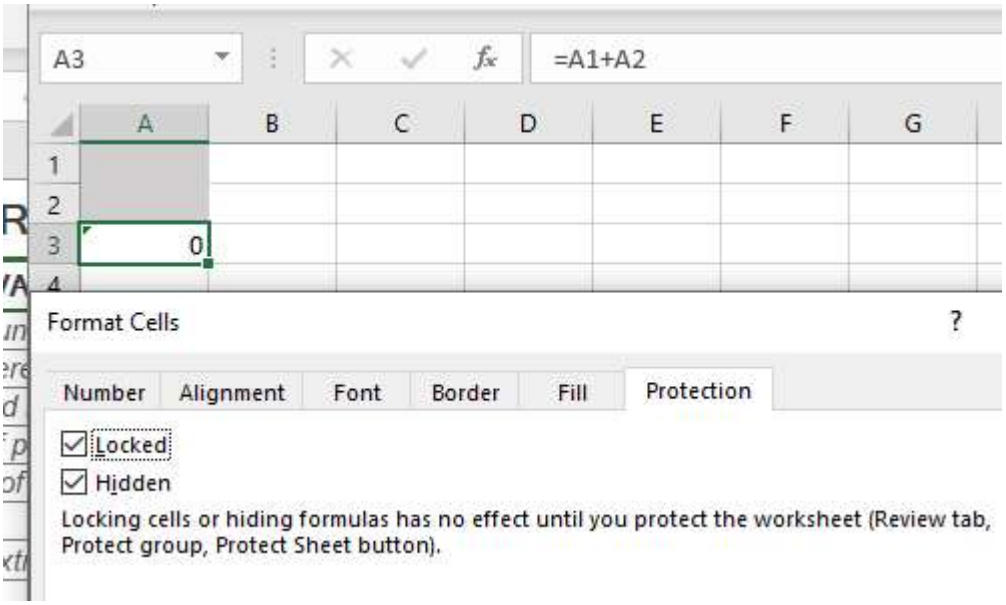
A lot of work has been done to make this useful – all we need to do is add our information.

Create our own template-

In A1 and A2 – make the background gray. In A3 – create the formula =A1+A2.

Right click on the cell A3 and go to Format Cells…

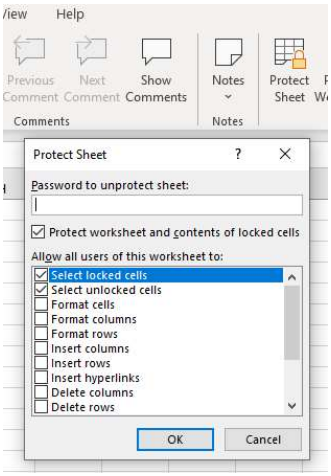
Under the Protection Tab – Check Locked and Check Hidden.

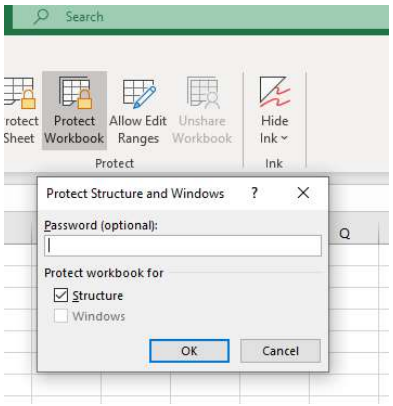


Select A1 and A2 and the Uncheck Locked.

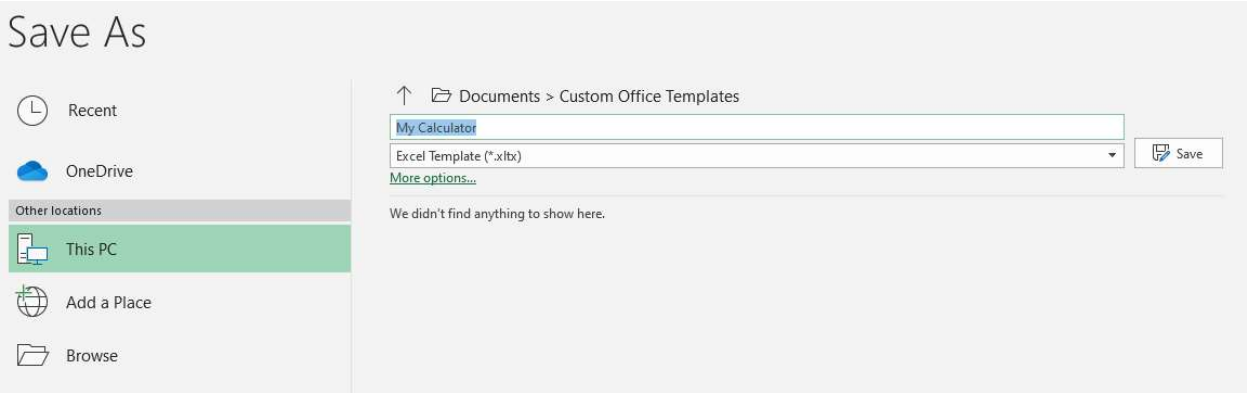
Before we save this, let's protect the worksheet or workbook:

Worksheet: or Workbook:

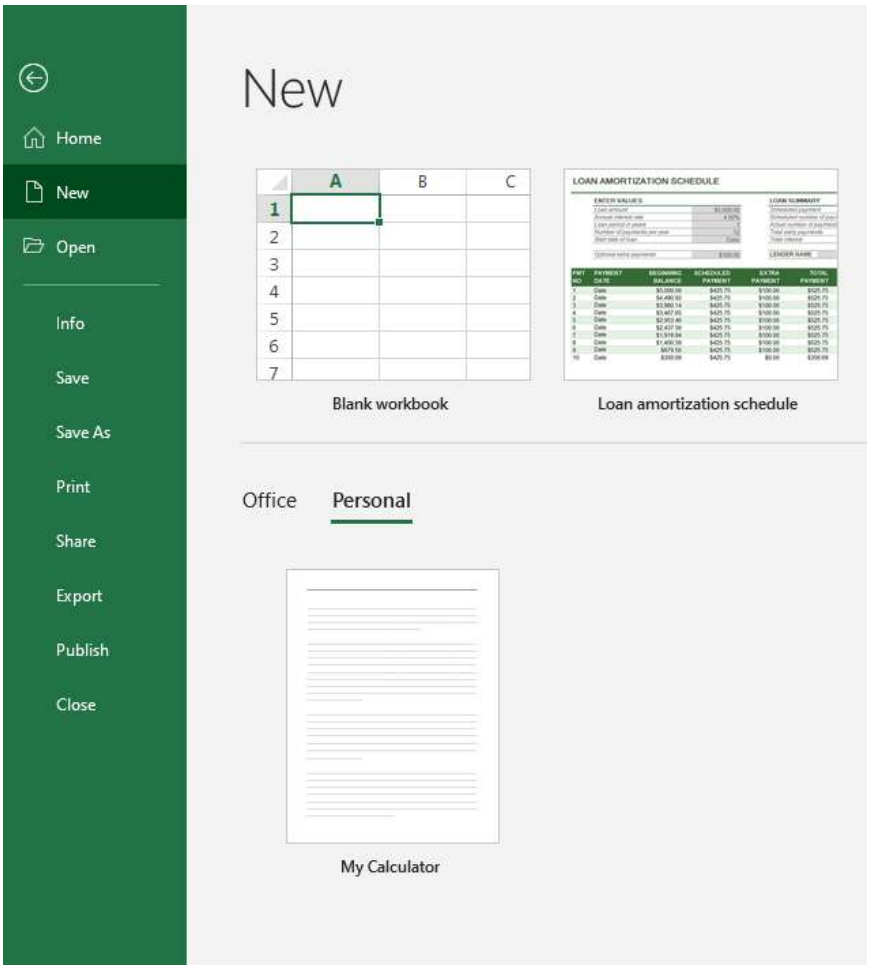




Save the Workbook as My Calculator



It will then be something you can use under File->New….



Macros

Macros are simply small programs written in Visual Basic that do specific tasks. These macros can be created or downloaded and incorporated into a spreadsheet.

In the View Tab of the Ribbon, Click on the Macros dropdown and Record New Macro

- Gets to the macro menu Macros… Alt+F8
  Shows the macros in the workbook.
- Record New Macro…
  Allows a person to create macros even if you don't know the language
- Visual Basic Editor Alt+F11
  This shows the language and commands used to create the macro

Macro Window

Hello World

The first program customarily written by a new programmer is "Hello World!". This program simply prints out Hello World. To do this using Excel macros,

- In the View Tab of the Ribbon, Click on the Macros dropdown and Record New Macro
  New Macro:
  - Name
  - Shortcut (ctrl and/or shift)
  - Store macro in…
  - Description (help others)
- Enter in these commands
  - Name – HelloWorld
  - Shortcut – ctrl+w
  - Store macro in… (This workbook)
  - Description (Print out Hello World – First program)
- Click OK
- New Form comes up with a stop button
- Type in Hello World! and click on the stop button.
- Test it – in various places

Hello World (part 2)

It goes back to the same spot, regardless of where you start. To make it relative, click on the red arrow right next to the stop button. Follow the steps above and try again. Did it work?

Looking at the code

The commands that are used to create the Hello World should look like the following:

```
Sub HelloWorld2()
'
' HelloWorld2 Macro
' Macro recorded 11/1/2003 by Don Bremer
'
' Keyboard Shortcut: Ctrl+w
'
     ActiveCell.FormulaR1C1 = "Hello World"
     ActiveCell.Offset(1, 0).Range("A1").Select
End Sub
```

The ‘ marks represent comments and are not read by the computer.

Structures of programming

There are actually 3 structures to any programming language. They are:

- Statement
- Loop
- Logical

To make a loop in a macro with a known number of times through, we can use the for command. We can change the commands of the macro to read:

```
For i = 1 To 5
     ActiveCell.FormulaR1C1 = "Hello World"
     ActiveCell.Offset(1, 0).Range("A1").Select
Next
```

This will write “Hello World” on the cell, move down one cell, and write “Hello World” again. Try this….

Logicals will ask a question. Depending on the answer, it will do a set of commands. Take for example the last “Hello World” example. Instead of having it say “Hello World” all 5 times, what if we have it say “Goodbye Cruel World!” on the last loop. We can do that using the conditional (or if) statement:

```
For i = 1 To 5
     If i = 5 Then
          ActiveCell.FormulaR1C1 = "Goodbye Cruel World!"
     Else
          ActiveCell.FormulaR1C1 = "Hello World"
     End If
          ActiveCell.Offset(1, 0).Range("A1").Select
Next
```

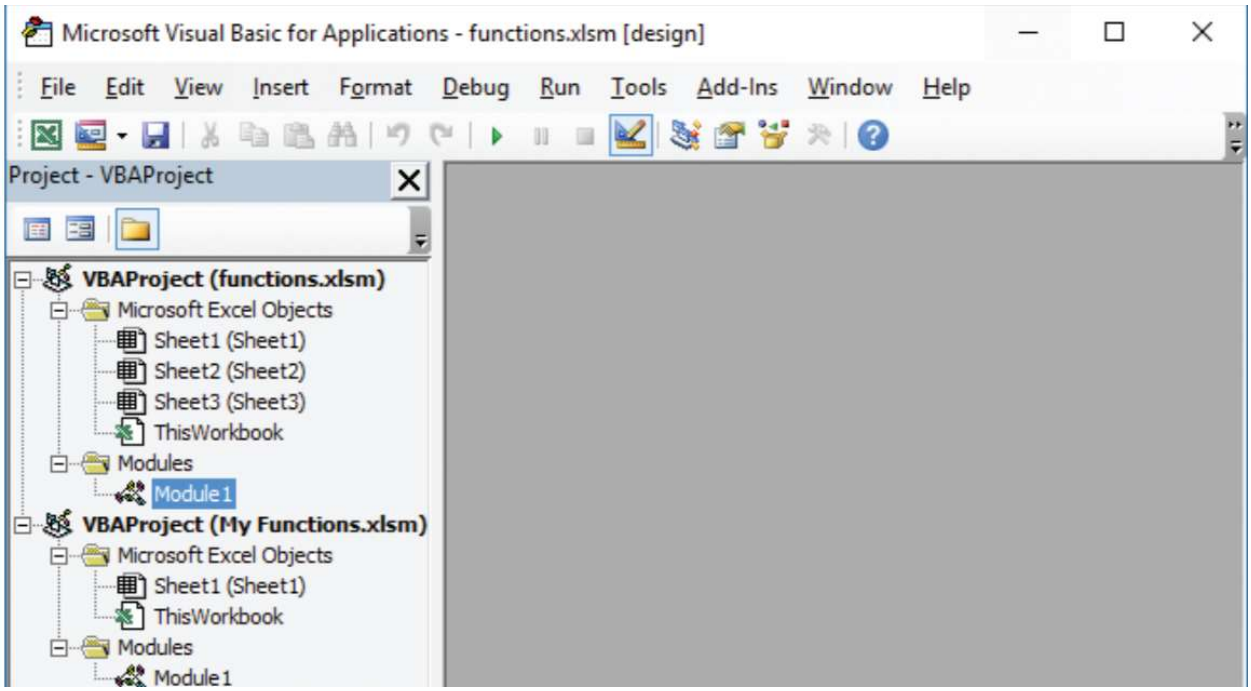
Copy macros between workbooks

If you receive a workbook that contains one or more macros that you find useful, you can continue to run those macros from within that workbook. However, you might find it useful or necessary to copy those macros to one of your own workbooks. For example, if you have a workbook that you keep open all the time, you might prefer to run the macros from that workbook rather than always having to keep the original workbook open. Similarly, many macros make use of an Excel object named ThisWorkbook, which refers to the workbook in which the macro is running. The only way to get such a macro to run successfully in another workbook is to copy it to that file.

You can make all your macros easily and conveniently available by storing them in a special file called the Personal Macro Workbook. However, before you can use this file, you must create it by recording a macro and using the Personal Macro Workbook to store the resulting code. After you have created the Personal Macro Workbook, it will appear in the Visual Basic Editor's Project Explorer pane, so you can follow the steps from the procedure for copying macros to copy macros to the Personal Macro Workbook.

In the Project Explorer pane, locate the workbook that contains the macros you want to copy, and then open that workbook's branches until you see the contents of the Modules folder.

Drag the module you want to copy to the VBAProject branch of your other workbook. Excel copies the module, creating the Modules branch in the other workbook if necessary.

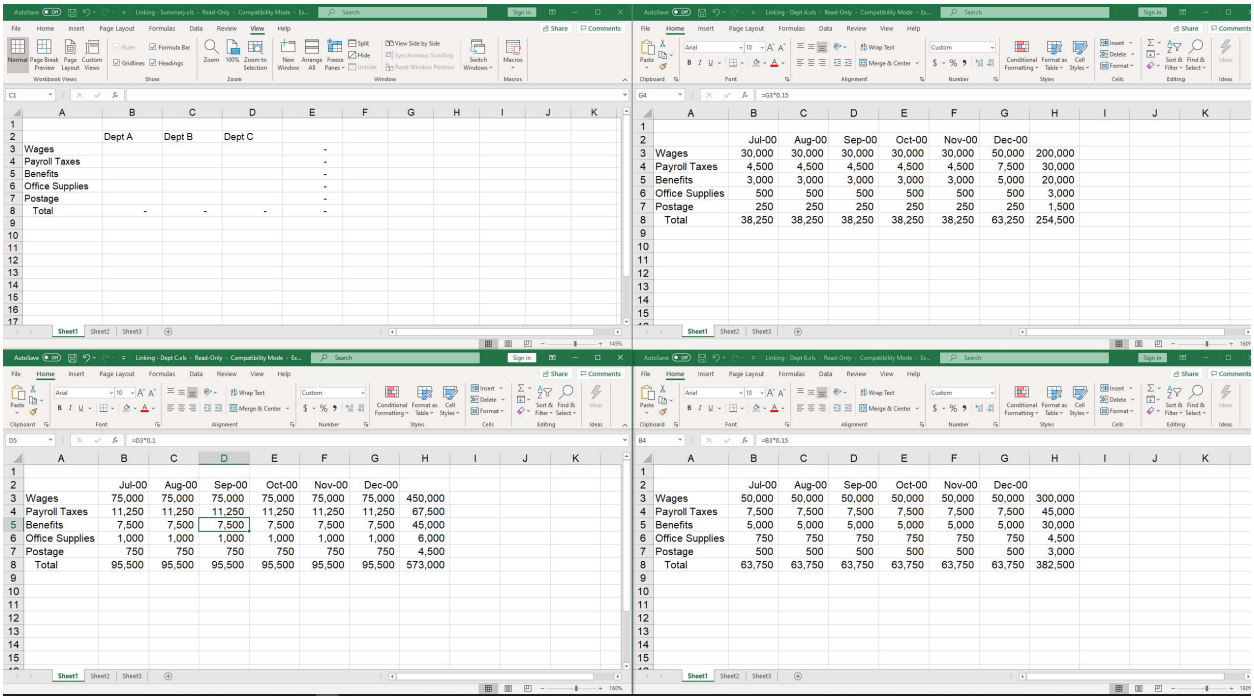


Link Workbooks

This is used when there is more than one workbook involved and the information needs to be linking into another spreadsheet. For example, if 3 different departments have budget information and you need to create a budget from all 3.

>OPEN>LINKING – SUMMARY, LINKING – DEPT A, LINKING – DEPT B, LINKING – DEPT C

- Open all of the workbooks that need to be linked (make sure only those 4 are open)
- Click on the “View” Tab.
- Select the “Arrange All” button in the Window Section.
- Select Tiled
- Click on OK



	A	B	C	D	E
1					
2		Dept A	Dept B	Dept C	
3	Wages				-
4	Payroll Taxes				-
5	Benefits				-
6	Office Supplies				-
7	Postage				-
8	Total	-	-	-	-

	A	B	C	D	E	F	G	H
1								
2		Jul-00	Aug-00	Sep-00	Oct-00	Nov-00	Dec-00	
3	Wages	30,000	30,000	30,000	30,000	30,000	50,000	200,000
4	Payroll Taxes	4,500	4,500	4,500	4,500	4,500	7,500	30,000
5	Benefits	3,000	3,000	3,000	3,000	3,000	5,000	20,000
6	Office Supplies	500	500	500	500	500	500	3,000
7	Postage	250	250	250	250	250	250	1,500
8	Total	38,250	38,250	38,250	38,250	38,250	63,250	254,500

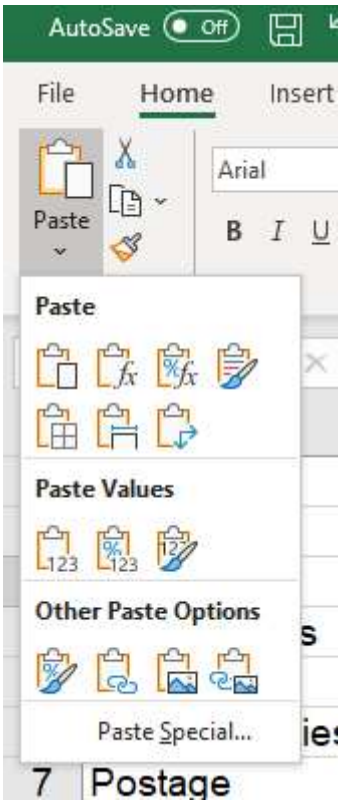
	A	B	C	D	E	F	G	H
1								
2		Jul-00	Aug-00	Sep-00	Oct-00	Nov-00	Dec-00	
3	Wages	75,000	75,000	75,000	75,000	75,000	75,000	450,000
4	Payroll Taxes	11,250	11,250	11,250	11,250	11,250	11,250	67,500
5	Benefits	7,500	7,500	7,500	7,500	7,500	7,500	45,000
6	Office Supplies	1,000	1,000	1,000	1,000	1,000	1,000	6,000
7	Postage	750	750	750	750	750	750	4,500
8	Total	95,500	95,500	95,500	95,500	95,500	95,500	573,000

	A	B	C	D	E	F	G	H
1								
2		Jul-00	Aug-00	Sep-00	Oct-00	Nov-00	Dec-00	
3	Wages	50,000	50,000	50,000	50,000	50,000	50,000	300,000
4	Payroll Taxes	7,500	7,500	7,500	7,500	7,500	7,500	45,000
5	Benefits	5,000	5,000	5,000	5,000	5,000	5,000	30,000
6	Office Supplies	750	750	750	750	750	750	4,500
7	Postage	500	500	500	500	500	500	3,000
8	Total	63,750	63,750	63,750	63,750	63,750	63,750	382,500

Creating the link
o Select the range of cells from Dept.A that represent the totals for Wages thru Postage.
o Copy the cells
o Click in the Summary Worksheet and click in B3

o Click on the “Paste” Dropdown on the Home Tab and select “Paste Link”
o Repeat for Dept’s B and C

	Dept A	Dept B	Dept C	
Wages	200,000	300,000	450,000	950,000
Payroll Taxes	30,000	45,000	67,500	142,500
Benefits	20,000	30,000	45,000	95,000
Office Supplies	3,000	4,500	6,000	13,500
Postage	1,500	3,000	4,500	9,000
Total	254,500	382,500	573,000	1,210,000



Reference table data by using structured references

When you need to reference part of a table in a formula, you could use a cell or range reference that points to the area within the table that you want to use in your calculation. That works, but it suffers from the same problem caused by using cell and range references in regular worksheet formulas: the references often make the formulas difficult to read and understand.

The solution for a regular worksheet formula is to replace cell and range references with defined names. For a table, you can use structured references. Excel offers a set of defined names—also called specifiers—for various table elements (such as the data, the headers, and the entire table), and it automatically creates names for the table fields. You can include these names in your table formulas to make your calculations much easier to read and maintain.

First, here are the predefined specifiers that Excel offers for tables:

- #All The entire table, including the column headers and total row
- #Data The table data (that is, the entire table, not including the column headers and total row)
- #Headers The table’s column headers
- #Totals The table’s total row
- @ The table row in which the formula appears