Category B: Microsoft Office Automation

# What?

We can describe Microsoft Automation as the automating of different tasks in the Microsoft Office software packet, in this packet are different tools that allow the developer to create automated tasks.

Some examples are: Setting of fonts, font sizes, automatic centering, bold.

# How?

Microsoft Office delivers an API and different tools to the Developers, these API and tools allow the developer to use different functions in the Microsoft Office packet to extend it and to add extensions.

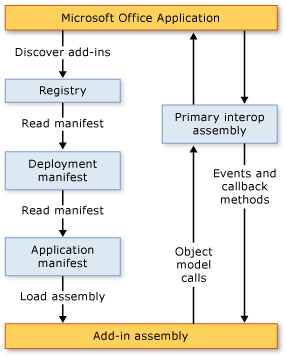
# API?

API stands for Application Programming Interface, it is a collection off different methods that the developer can use to communicate with a part of the application.

Examples are: a method to find the selected cells, a method to set the color, a method to find the used font, etc.

This allows the application that uses this to work on a high level off abstraction.

# Architecture + How it works



The architecture off a Microsoft Office add-in works because off that they load the application that has been made in Visual studio, this application can then be compiled and loaded in Microsoft Office.

1. When starting Microsoft Office the program will look into the register to see if there are any records that indicate on add-ins that were made with the Office Developer Tools.
2. When it found records it will load the Dynamic Library File “VSTOEE.dll”, this dll file then loads the “VSTOLoader.dll” These are the components for the Visual Studio 2010 Tools for Microsoft Office Runtime
3. The “VSTOLoader.dll” will load the .NET framework and a little piece of the Visual Studio Tools for the Microsoft Office Runtime, this so that the memory has been optimized and only the piece that is needed to run the add-in will be loaded.
4. The Visual Studio Tools will go look at the manifest to see if there are any updates for the add-in loader and it will install those.
5. There are different security checks to see if nothing can be compromised, examples here are exploits that could execute arbitrary code on a computer.
6. If the add-in has been trusted it will see if there are any application updates, this by using the application manifest file.
7. It will create the application domain that will initialize the add-in assembly.
8. The Visual Studio Tools for the Microsoft Office runtime loads the add-in assembly in the application domain.
9. The Visual Studio Tools for the Microsoft Office Runtime calls the **RequestComAddInAutomationService** method in the add-in if this was override in it.
10. The Visual Studio Tools for the Microsoft Office Runtimes calls the method **RequestService** up if this method was overridden.
11. Microsoft Office starts up.

# Tutorial

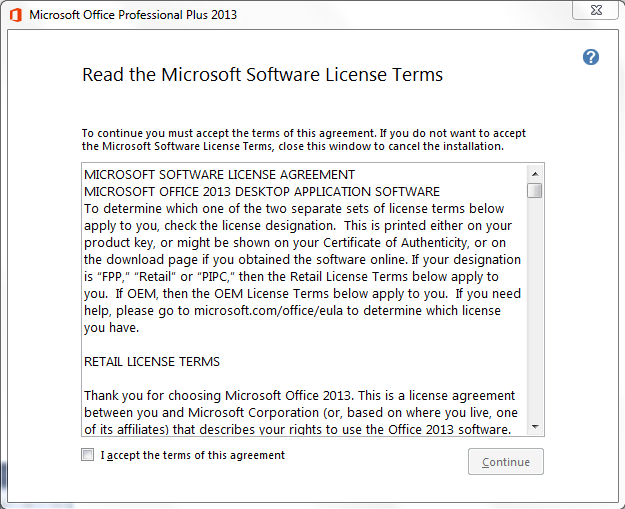
## Programs Used

* Microsoft Office 2013
* Visual Studio 2010 SP2

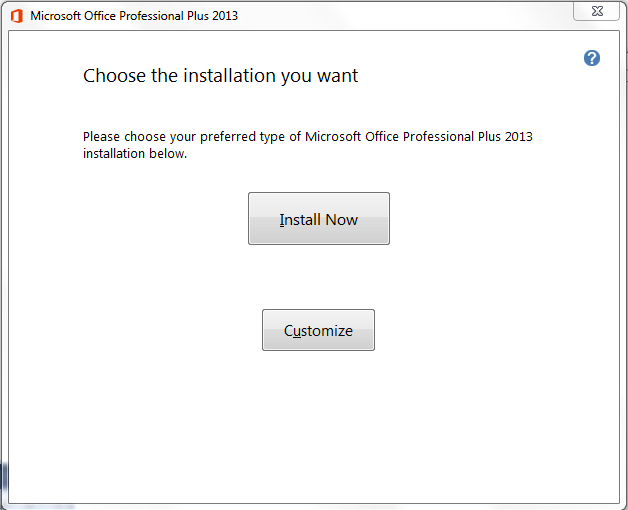
## Install Microsoft Office + Interop Assemblies.

### Open up the installation process

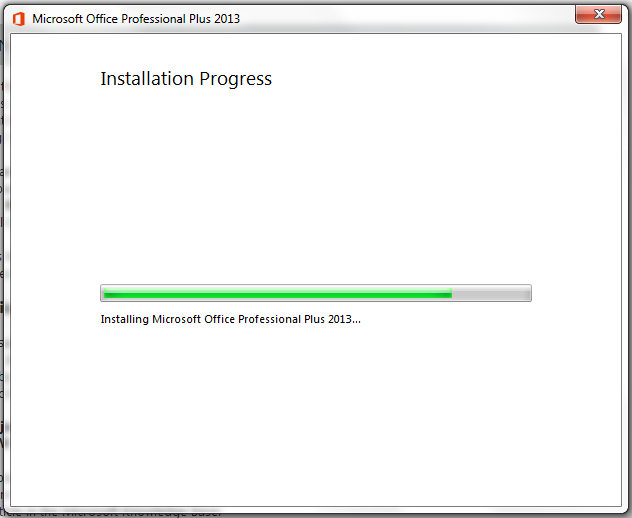
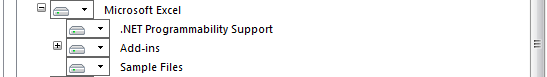
### Accept the license agreements and press continue



### Press customize and make sure the .NET computability support will be installed

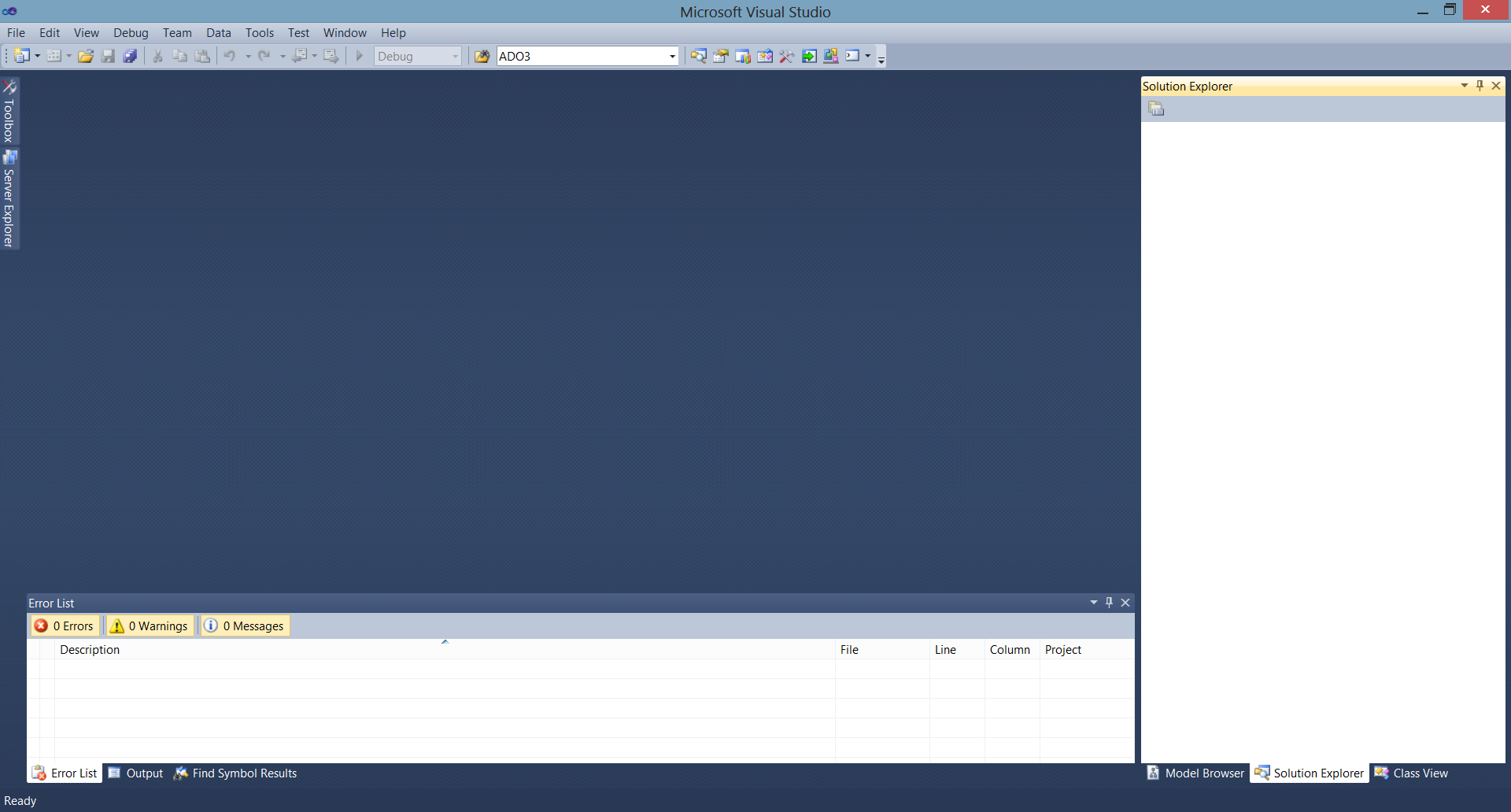


### Wait until Microsoft office is done installing

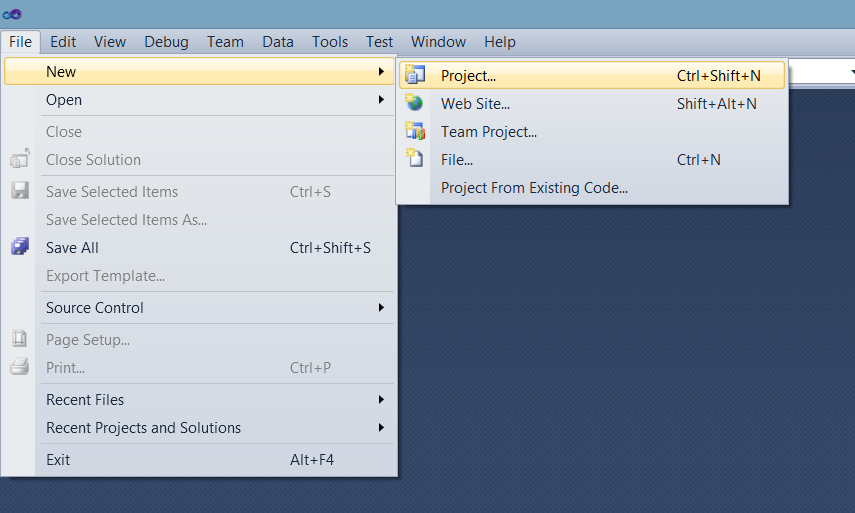


## Setting up Visual Studio for your first project.

### Open Visual studio



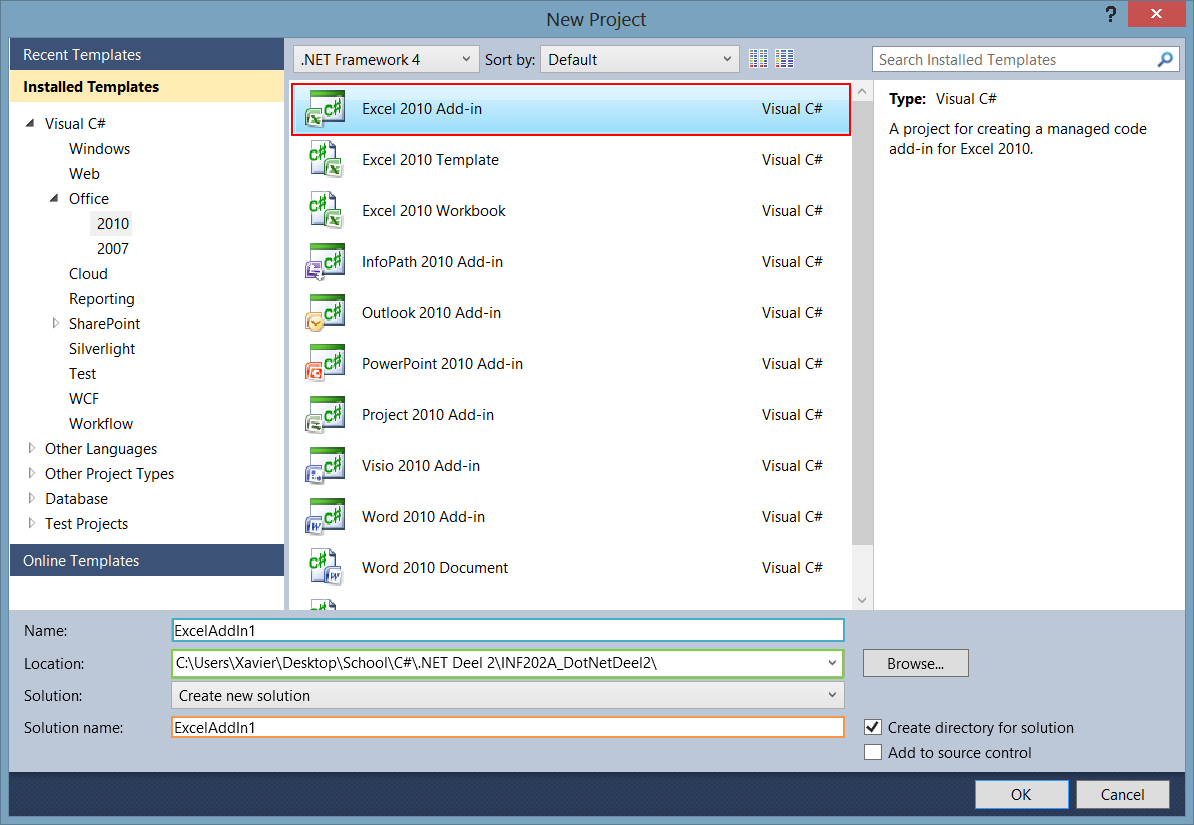
### Go to File 🡪 New 🡪 Project



### Make a new project

1. Select Windows Forms Application (Box a)
2. Fill in a name, this will be your project name, we use KDG for this example. (Box b)
3. Select where to save the project (Box c)
4. Fill in the solution name, this will hold your projects, we use Category B for this example

(Box d)



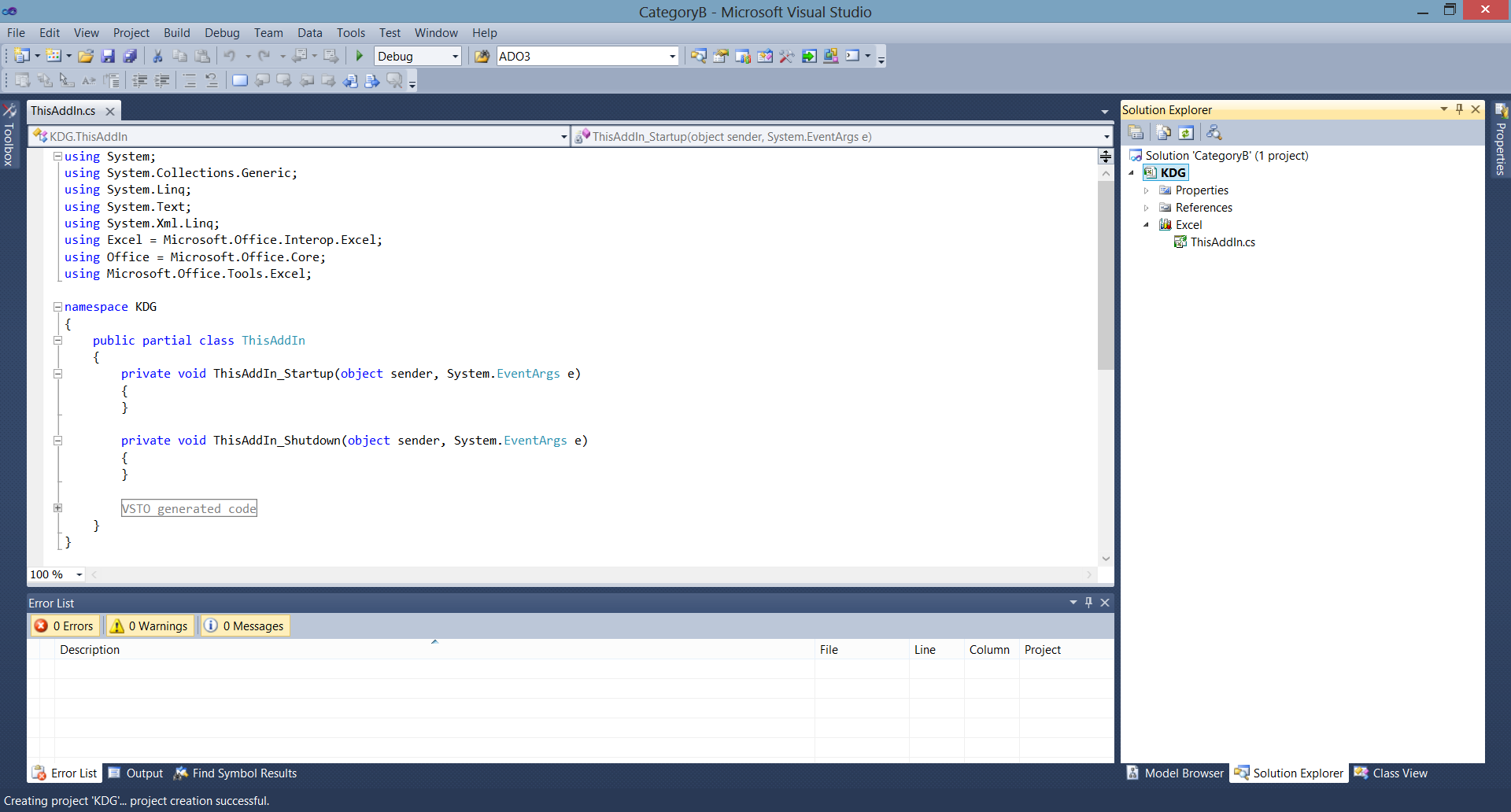
Box B

Box C

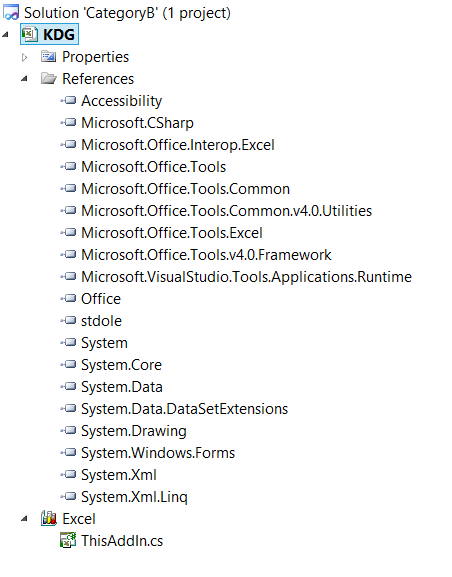
Box D

Box A

### Then you get this screen

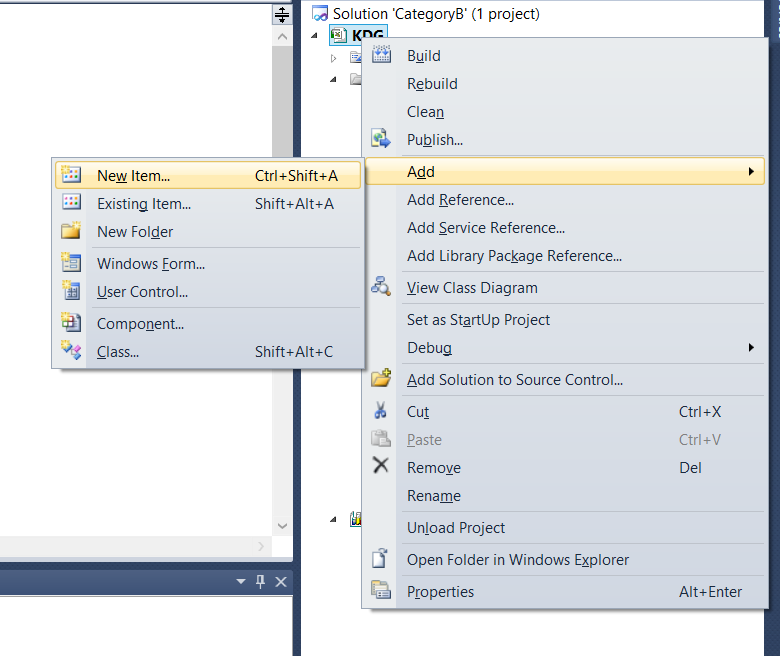


### After pressing Ok, you will see this in the solution explorer.



## Creating a custom Ribbon

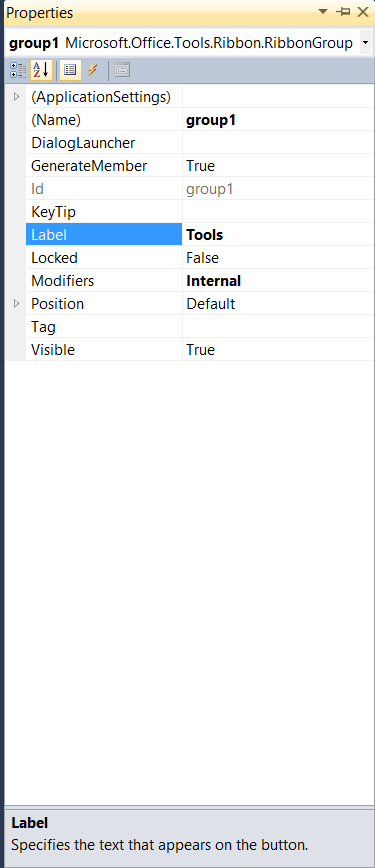
### Start by right clicking the project and going to add 🡪 new item



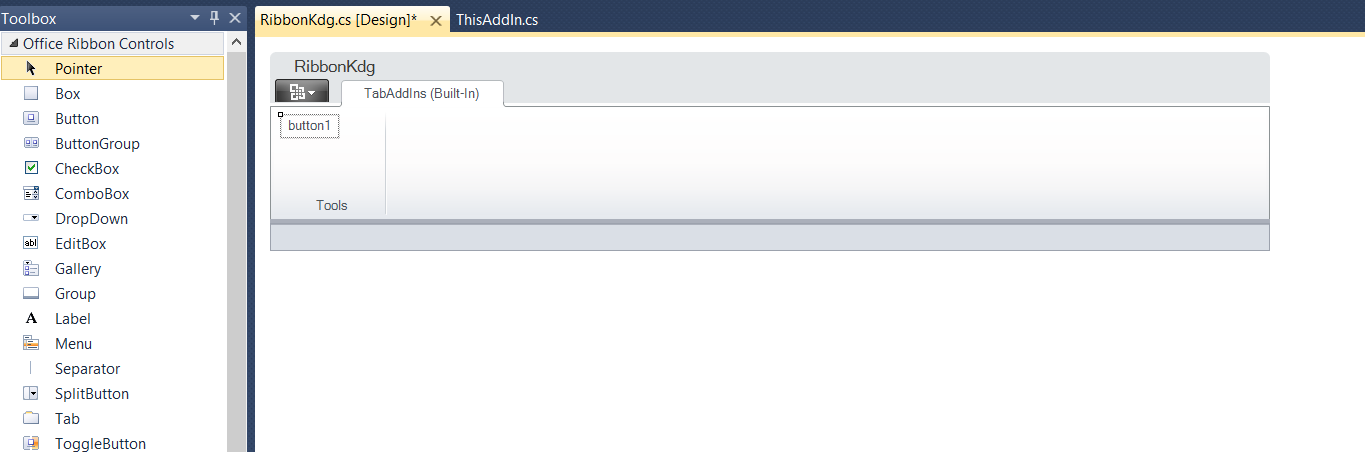
### Click on Ribbon (Visual Designer) and give it a name (RibbonKdg in this example)

# 

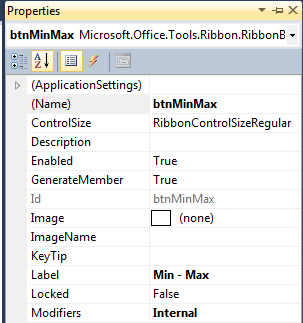
### Rename the ribbon group name to “Tools” By right clicking the group and pressing properties.



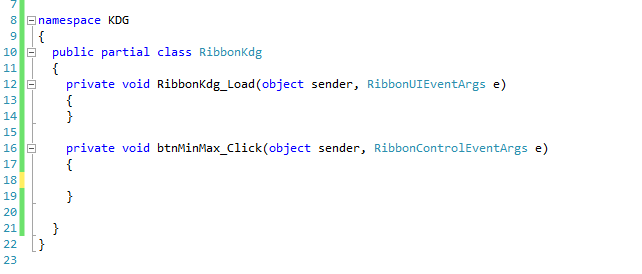
### Add a button by going to the toolbox and dragging a button on to it, name it Fill row. (This will fill the row from where you are till the end in a color)



### Right click the button 🡪 Properties and change the label + the name



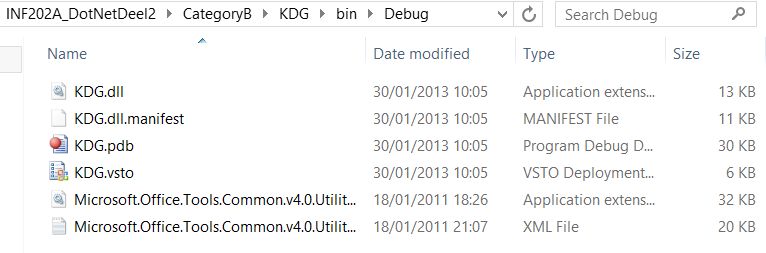
### Double click on the button and you will go to the code view.



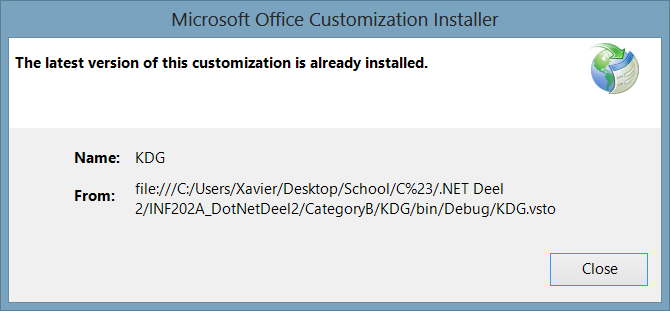
### Go to build 🡪 build solution



### Go to your source directory and then to bin/debug 🡪 Open the .vsto file to install it in excel.



### It will install the file into excel now and show this screen when it is completed.



### When you open excel now you will see this

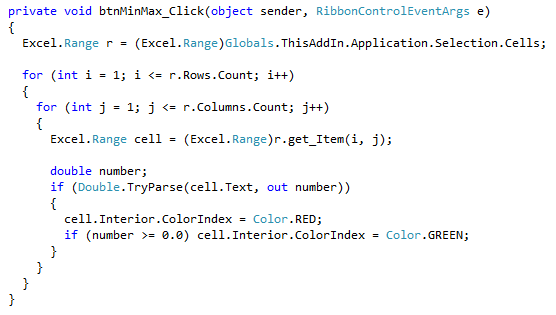


## Insert functionality into custom ribbon

### Make your own enum for colors with their own colorIndex (given below)

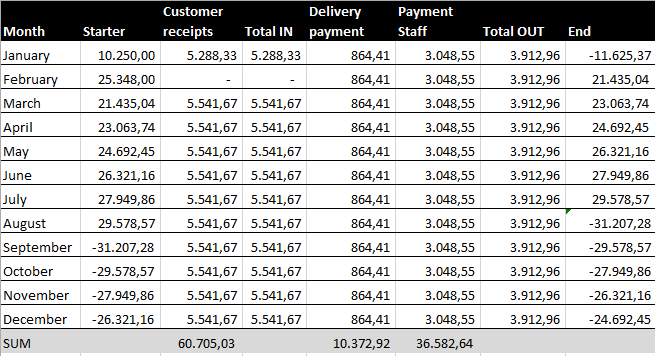
# C:\Users\Joey\Desktop\Tutorial microsoft office automation\tut13.jpgC:\Users\Joey\Desktop\Tutorial microsoft office automation\tut12.PNG

### Give your custom Ribbon button a functionality

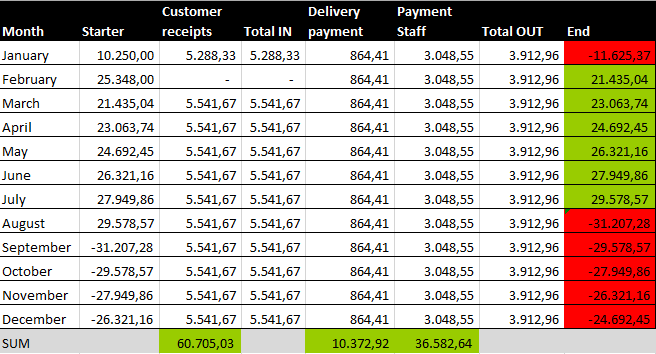


## Using the add-in

### Open an excel document (before)



### Use the Written function (after)



# Conclusion

Out of this whole research we can conclude that Office has some great Automation Tools that allow a developer to create his own add-ins for Microsoft Office.

The API has been extended very well and it is possible to change almost anything in Microsoft Office.