

***Easily
Beginning C#
Application
Development***



Beginning C# Application Development

Part I - Development Environment

Setup

When starting implementation, the very first things to have is a development environment. We will be developing in C# .NET and while it is possible to write applications with just the .NET framework without an IDE, it makes no sense for development. We will be using the visual studio IDE for our development and I'll walk you through the basic setup, if you do not have one installed. You may skip this part if you already have the setup in place.

To download visual studio 2012, point your browser to the following address:

<http://www.microsoft.com/visualstudio/eng/products/visual-studio-premium-2012>



This is the premium version of visual studio. Of course you can download the express version but if you are serious about development, the express versions will only be a set back as you will lack the following important tools of development:

- Project templates – these are life saving starting points for any particular project you want to build
- Performance and code analysis features
- Additional XML features
- Additional code-editing features – particularly useful for refactoring

- UML Features





You can find the rest of the story at <http://msdn.microsoft.com/en-us/library/ms349441.aspx>

Click on download

Visual Studio Premium 2012 offers an integrated ALM solution to bring together stakeholders, users, and software development functions to deliver compelling applications as a unified team. After installation, you can try Premium 2012 for up to 30 days. You can extend your trial period to 90 days by registering to obtain a free trial product key.

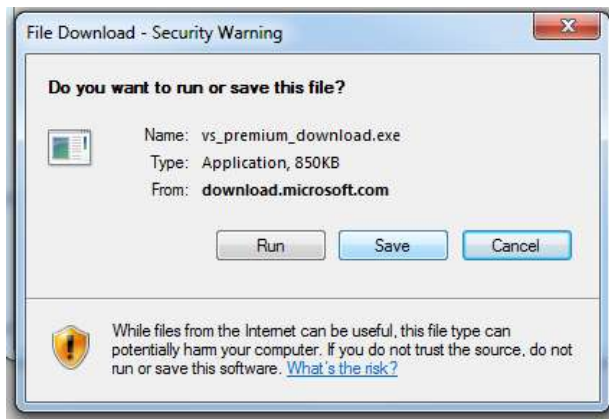
Download language
English ▼

Installation options

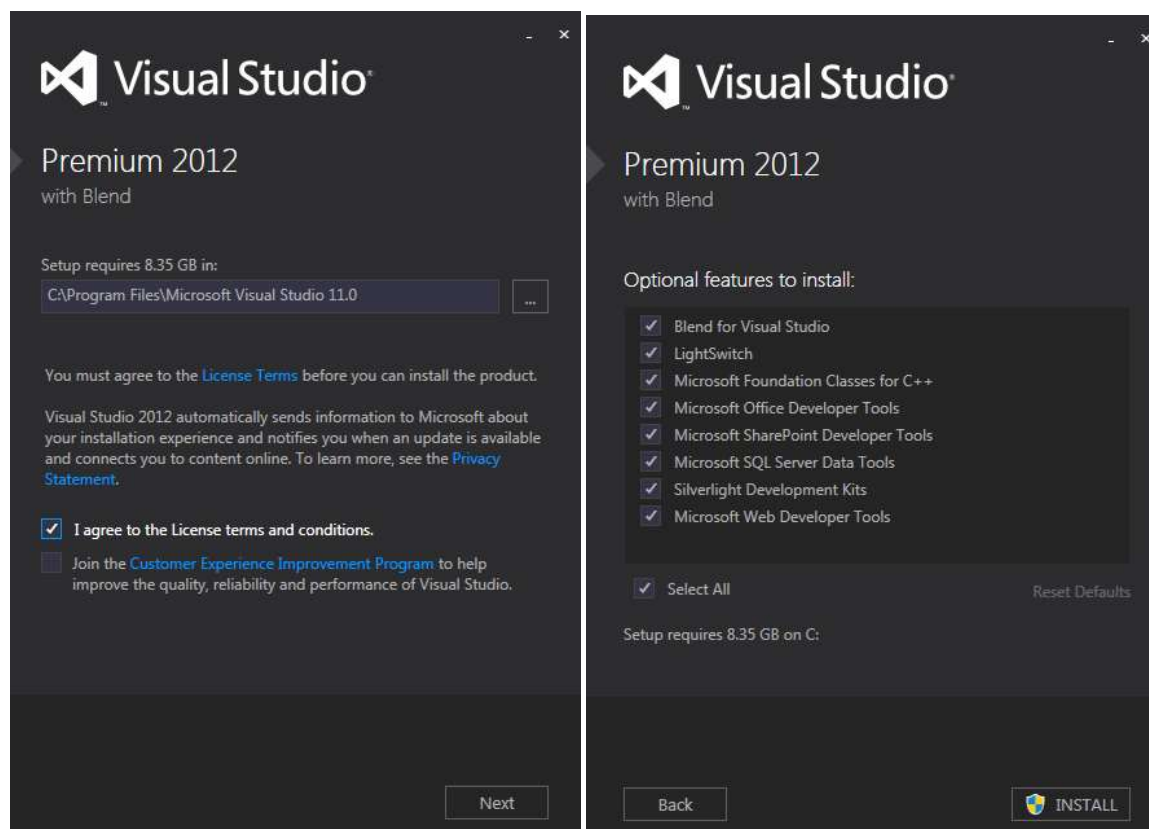
-  Visual Studio Premium 2012 90-day trial - English
[Install now](#)
-  Visual Studio Premium 2012 90-day trial - English
[Download now](#)
-  Visual Studio Premium 2012 90-day trial - English
[DVD5 ISO image \(VS2012_PREM_enu.iso\)](#)
-  After installation, you can try Premium 2012 for up to 30 days. You can extend your trial period to 90 days by registering to obtain a free trial product key.
[Register now](#)

Click on install now.

You'll get a prompt which allows you to save the file. I'll Choose Run. I want to save mine on the Desktop.



The startup interface will appear. check "I agree to the license terms and conditions" and click "Next"



Then click "INSTALL" and the installation process will begin.



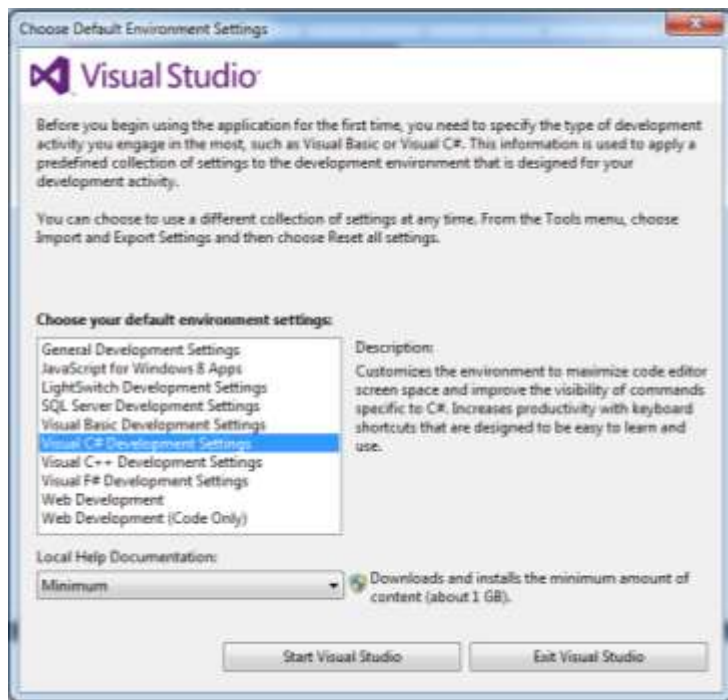
The installation will take a while depending on your internet connection. You may be asked to reboot your system before the installation is fully completed. When that comes up, click “Restart Now”



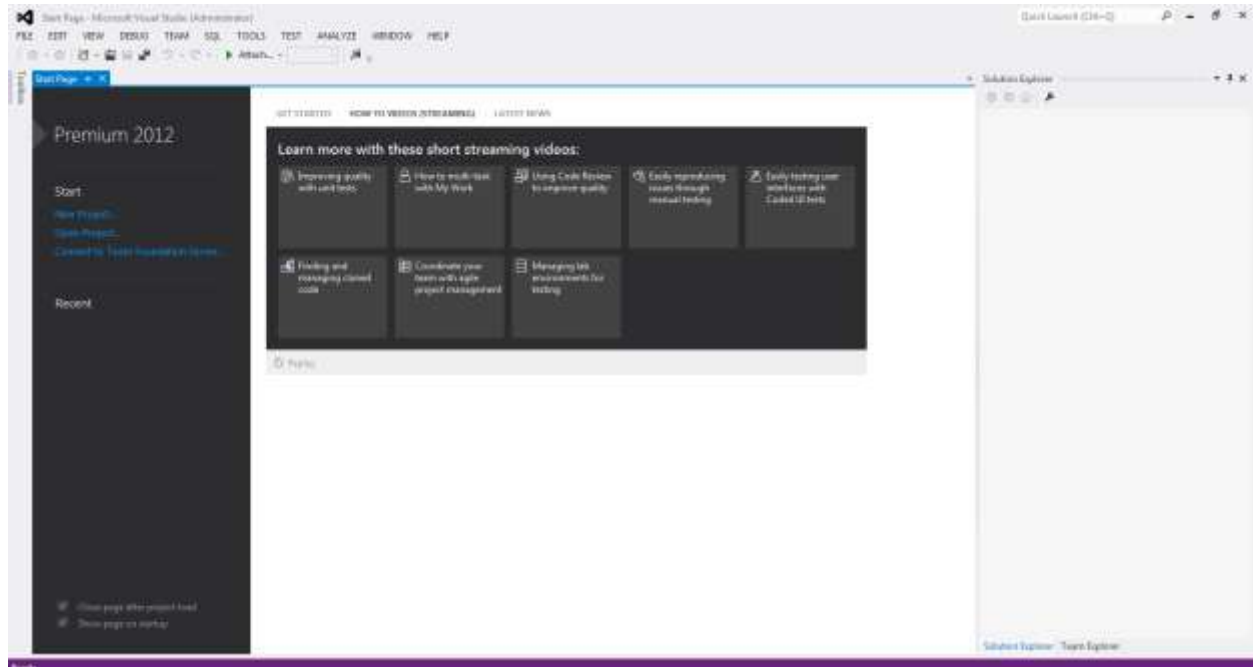
And all you need to do once the installation is finished is to click “LAUNCH” to fire up the visual studio IDE.



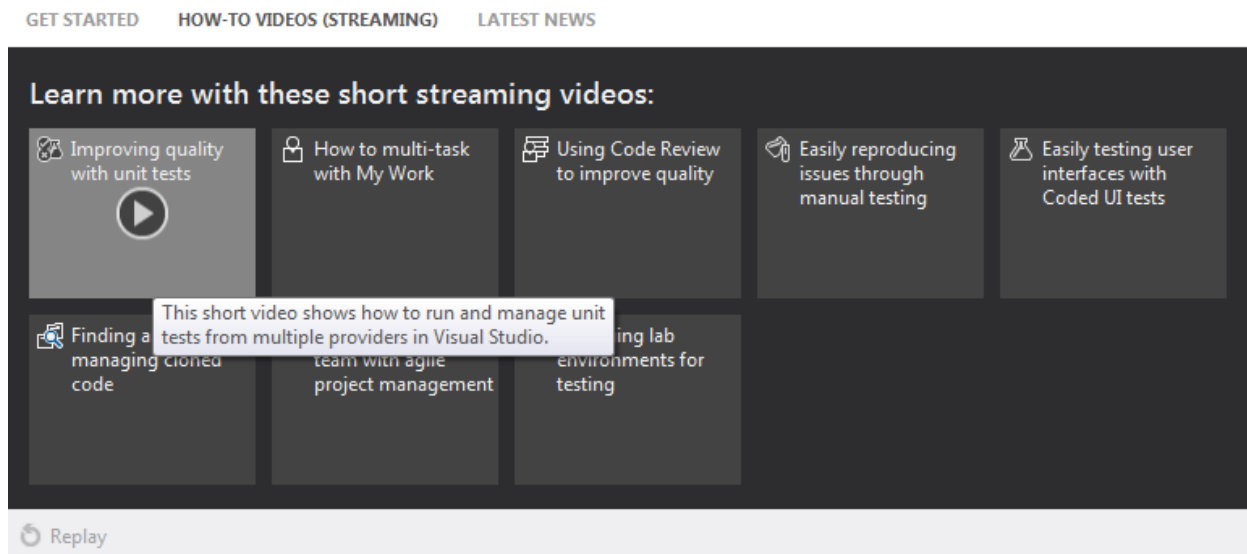
Then we get this window allowing us to order the product. Ill leave the ordering process for now and just jump right in by clicking “Cancel”. Then I get this:



We will be developing in C# so choose the Visual C# Development environment Setting and start visual studio. The process will take a while after which the IDE will appear in all its glory.



Right away you have access to free video tutorials to get you started right before your eyes.



At this point we are ready to start developing. In general, the structural path in development with visual studio will revolve around creating Solution, Projects and Items.

Another very important component of application development is source control. Microsoft provides Team Foundation server. This allows a team of developers work together on the same project or solution effectively. We won't be looking at team foundation server at the moment.

Beginning C# Application Development

Part II - Sample Problem | One Way To Do It

Introducing the problem

Now that you have installed visual studio installed its time to begin our main journey. We will be considering a very simple problem and then develop an application around it. I will introduce a lot of concepts along the way using the same problem. In real life you will have to take some time to learn the problem domain (this is essentially a description of what the application being developed is meant to do) peculiar to each organization and this could take a while to understand. It is important you fully understand the problem domain of any application you are working on before writing any code.

Imagine you just got a new job as a developer and you are lucky (very lucky and unusual case) to be a part of the first developers of a simple financial application. Here is what the application is meant to do:

Enable any one predict their financial future so they can make adjustments where possible financially to avoid going into debt.

Let's say the application will be called **Personal Financial Wizard (PFW)**

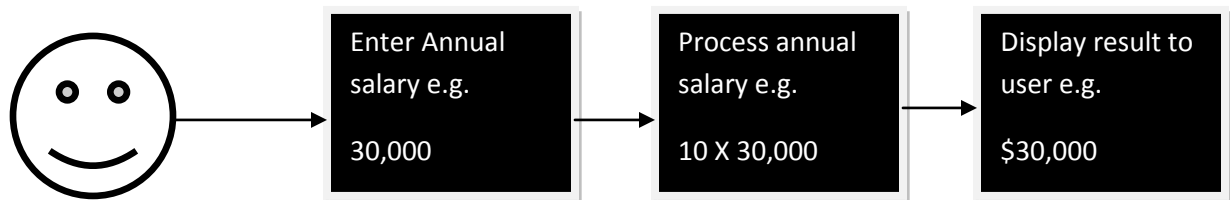
Not a bad name I guess. Now you are asked to write a program, it should be very simple like:

A user will enter their annual or bi-weekly or monthly salary. The application should calculate how much money the user would have earned after 10 years period.

Assuming I earn \$30,000 annually. The application should be able to tell me that in 10 years, I should have made \$300,000. This means if I have been hoping to buy a car worth \$500,000 in ten years time, then I need to rethink again. May be I need to get a new job now.

This may seem like a very simple application. But using this problem domain, I will illustrate principles of application development components.

Let me start of by sketching using boxes to further drive home the understanding of the problem we are about to solve. Latter on we will be using UML and all that but for now, just small notes like you would have taken in a meeting. In real application development, there will be meetings with managers where they discuss the problem and what type of solution they want. You need to pay close attention to everything and marking out points where you need clarification. This is the beginning of development.



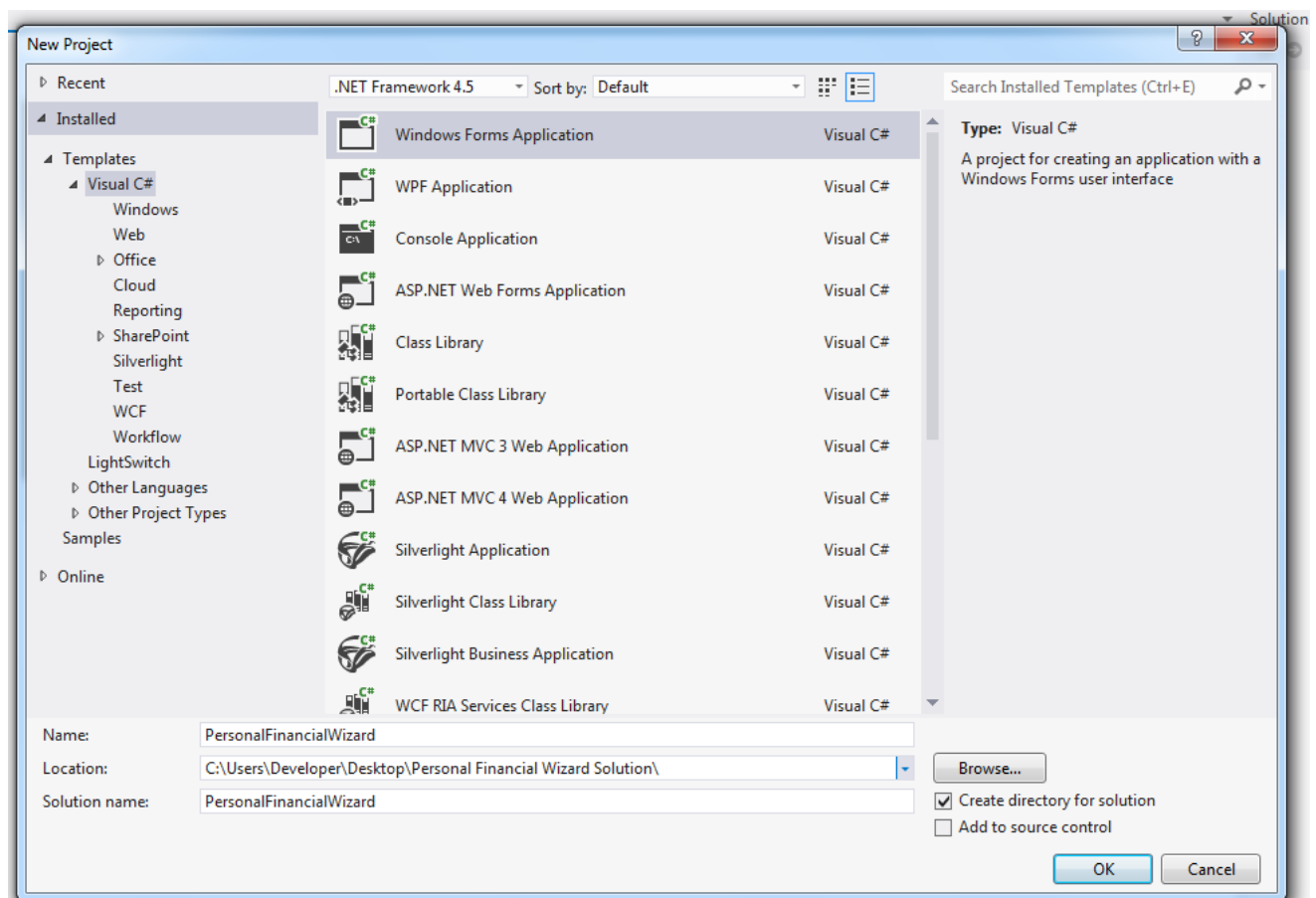
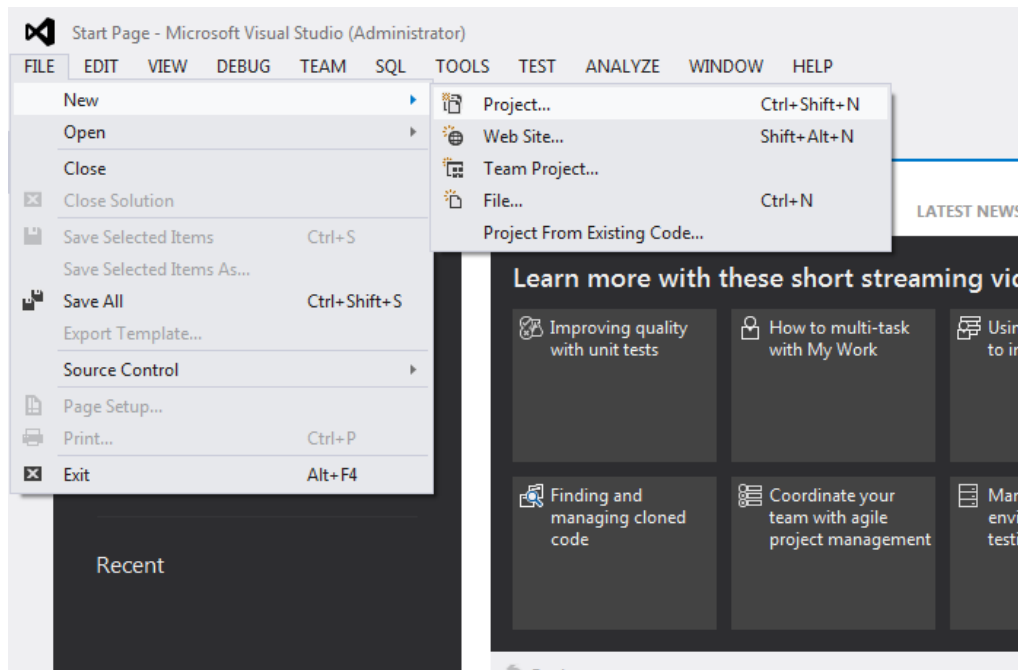
Then a new specification comes out.

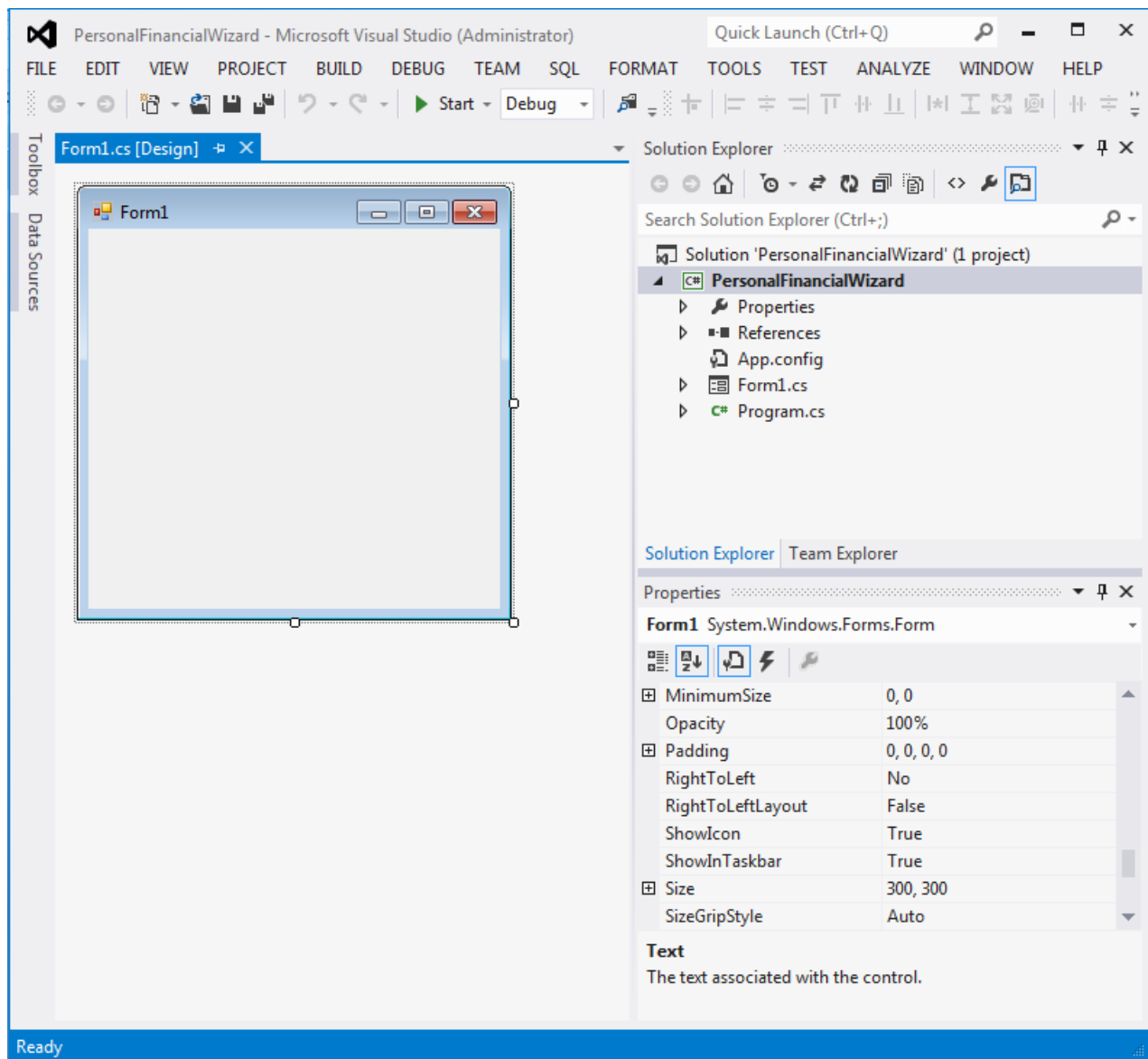
It is required that the final product be a windows forms application. This means that a user will interact with buttons and forms to interact with the system.

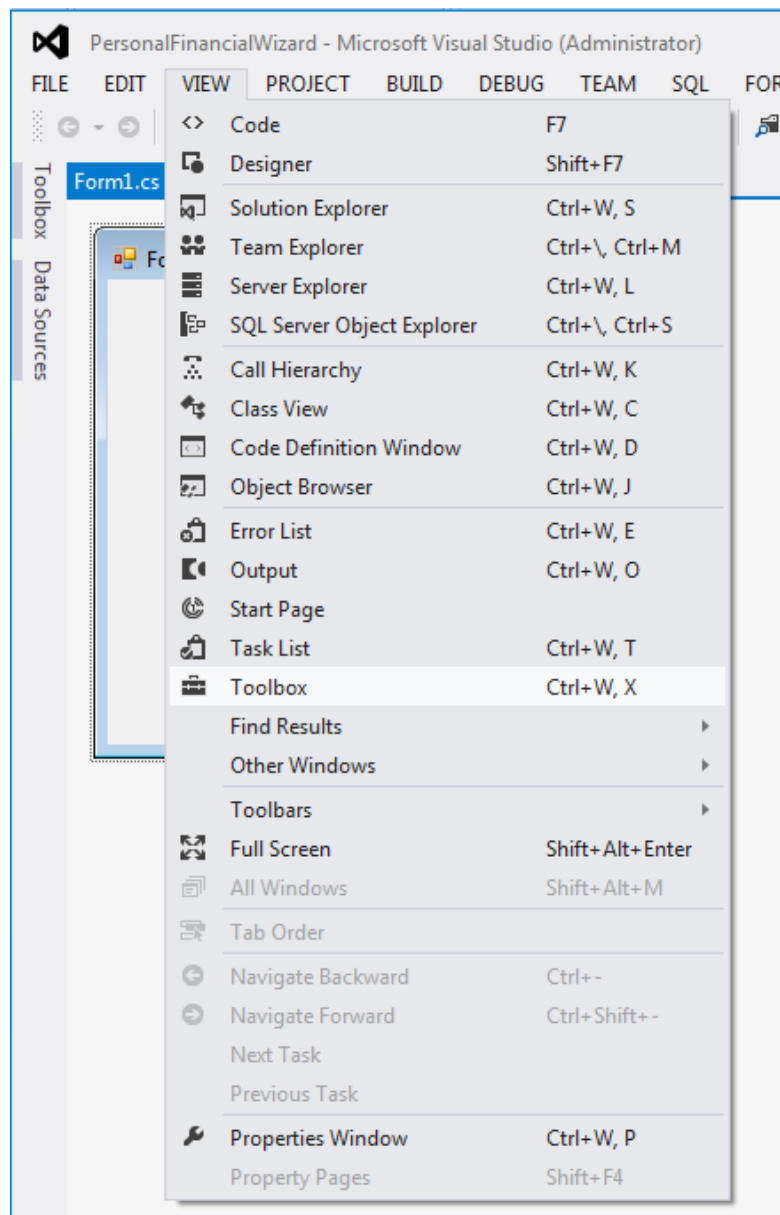
Personal Financial Wizard Implementation: One Way to do it

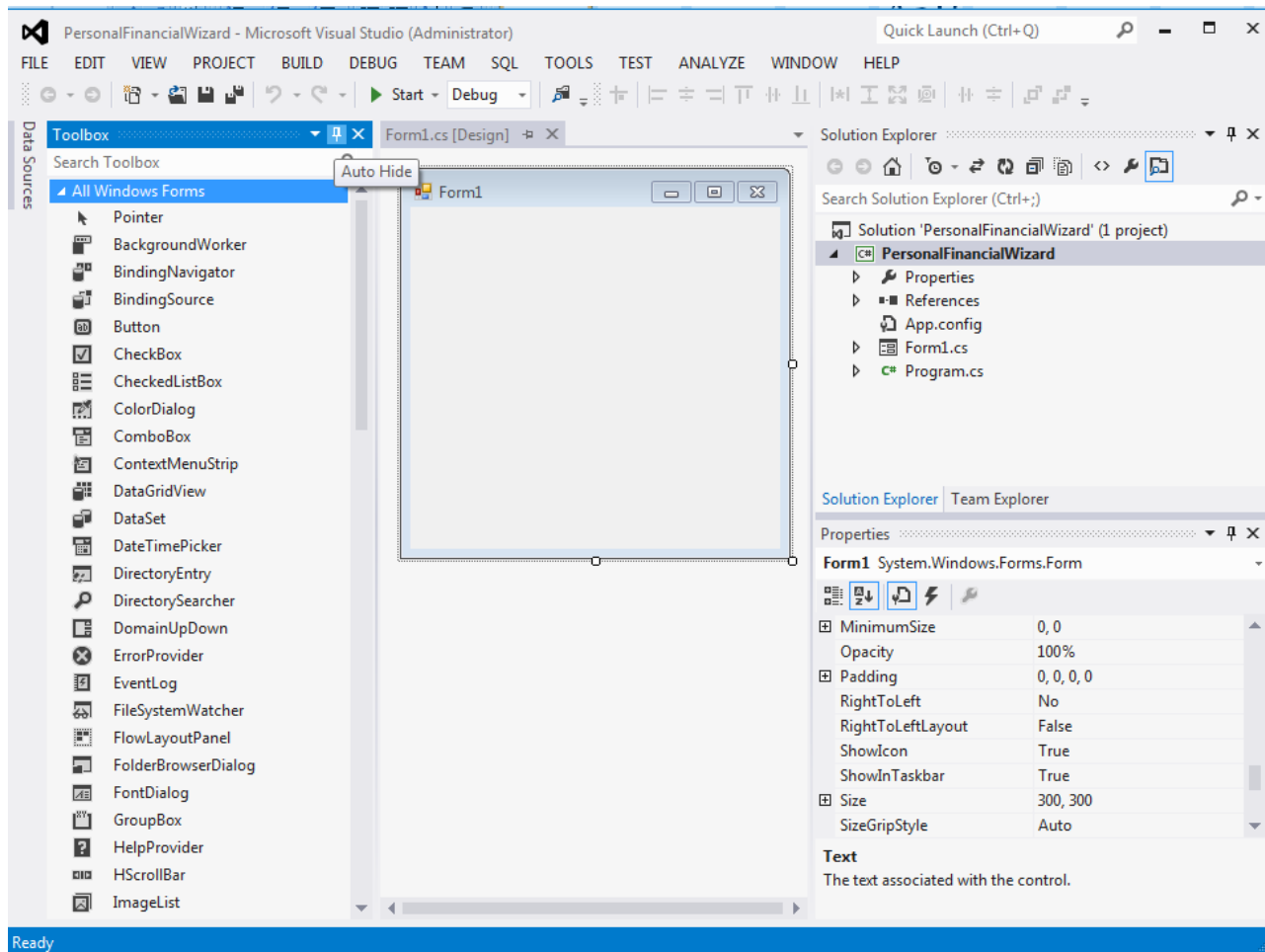
For now I will show you how a programmer can complete this task. In the next chapter ill take you far beyond this level of approach.

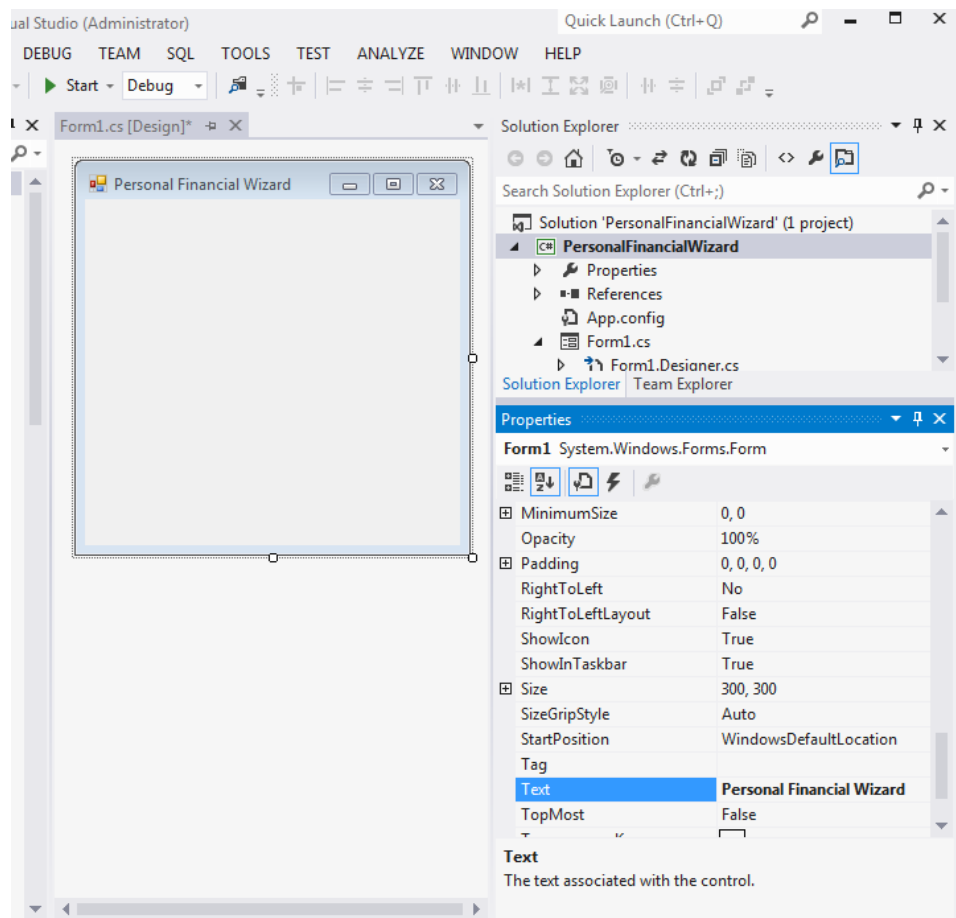
We will be developing a windows form application so we will create a new project (which will automatically create a new solution in visual studio).

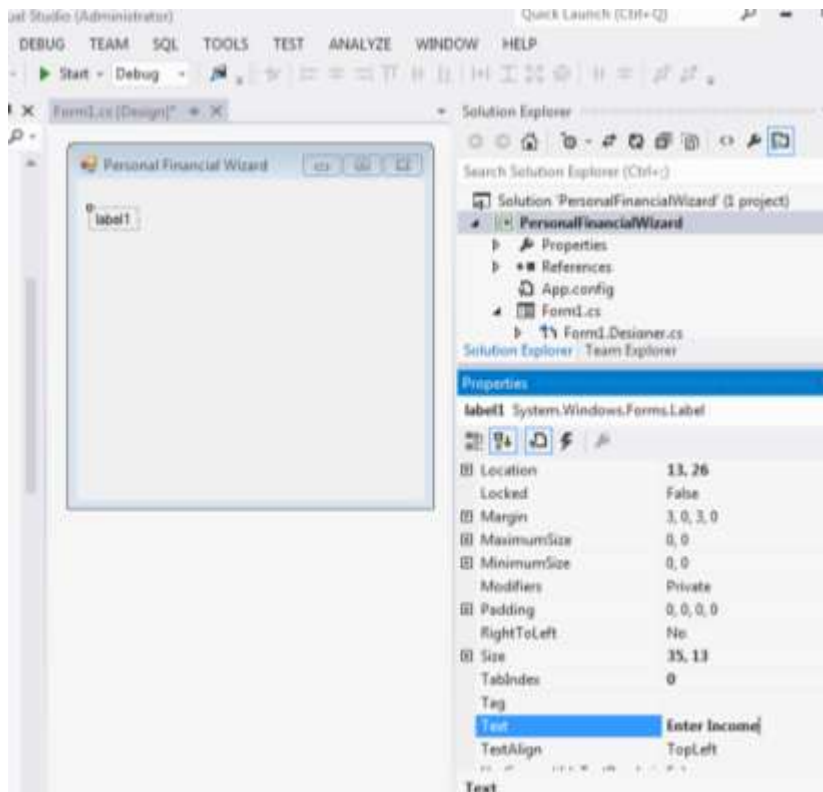
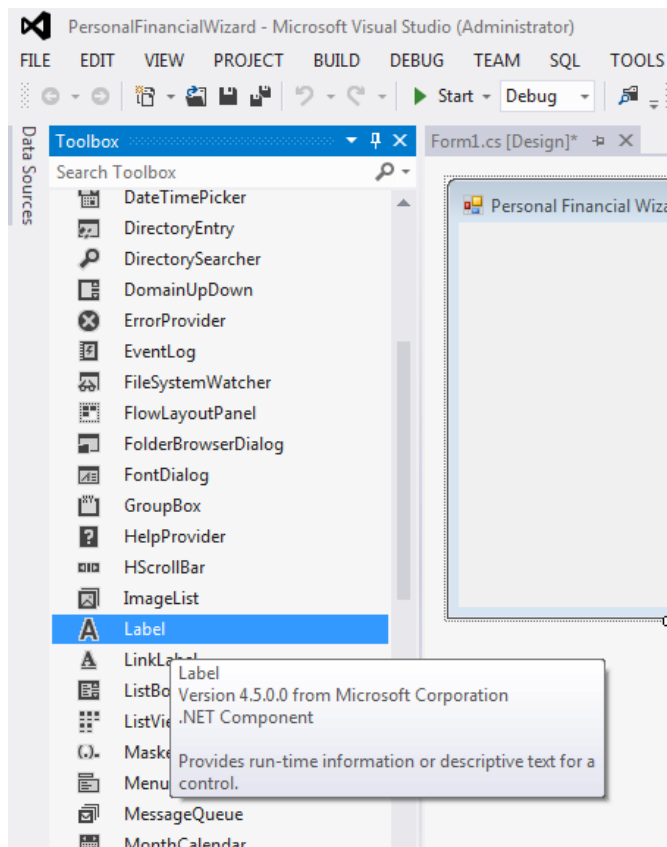


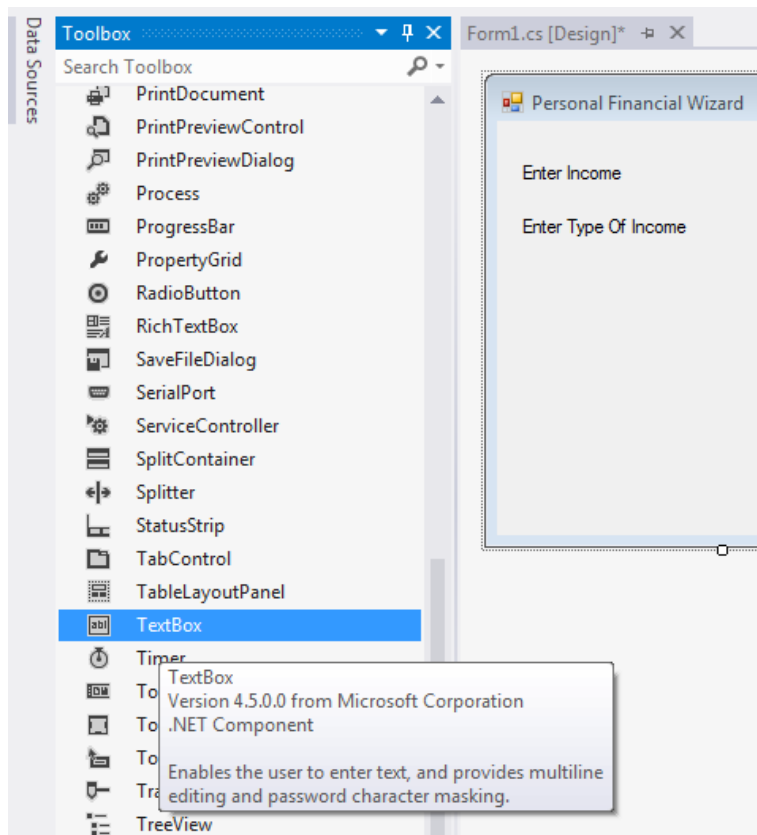
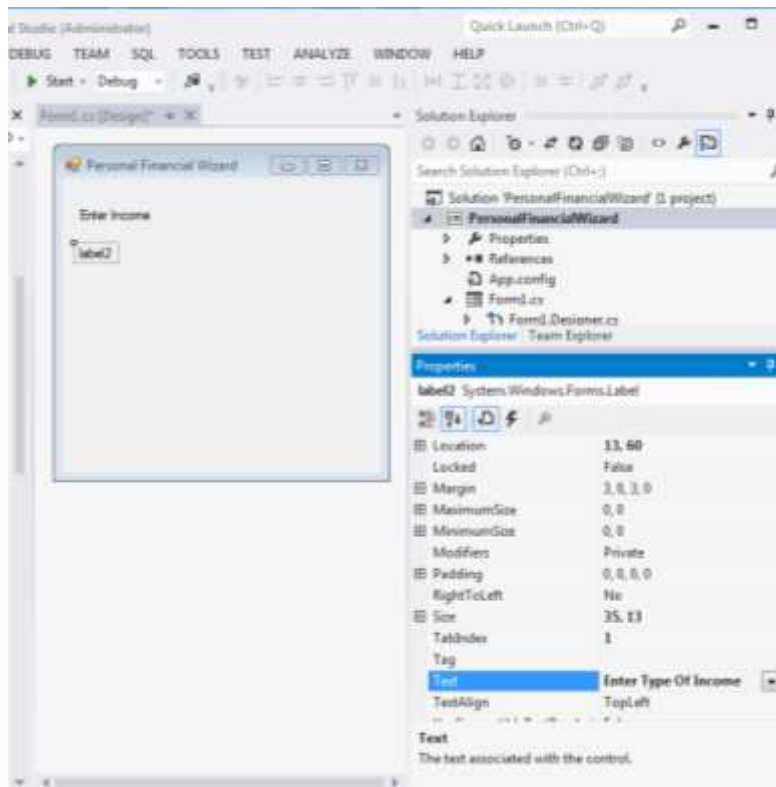


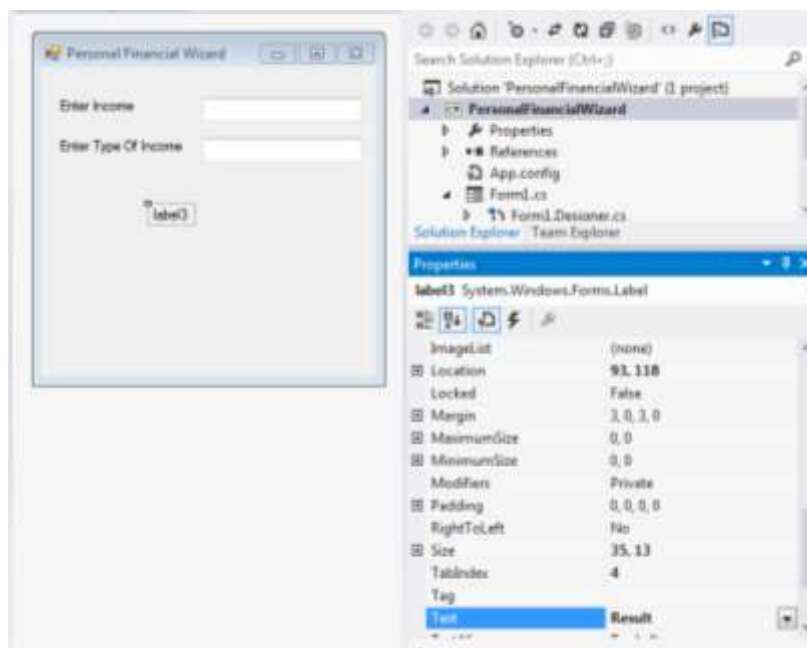
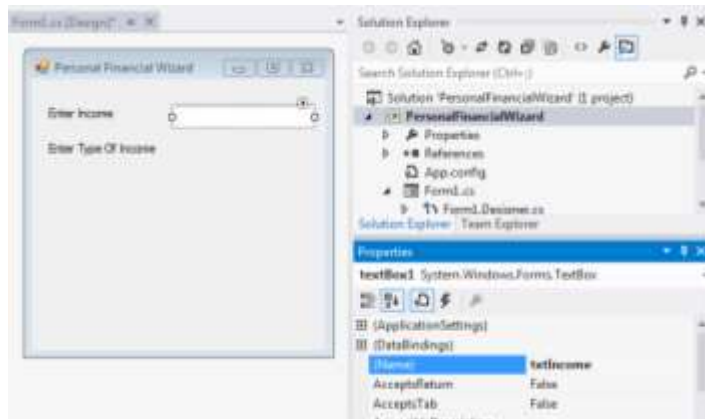


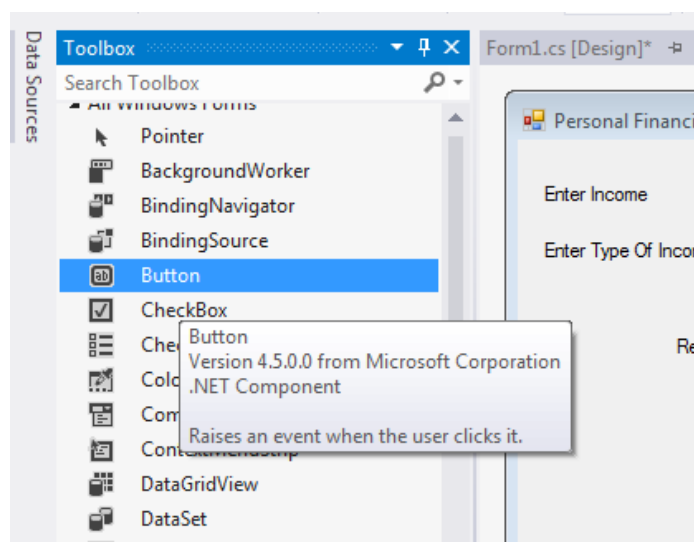
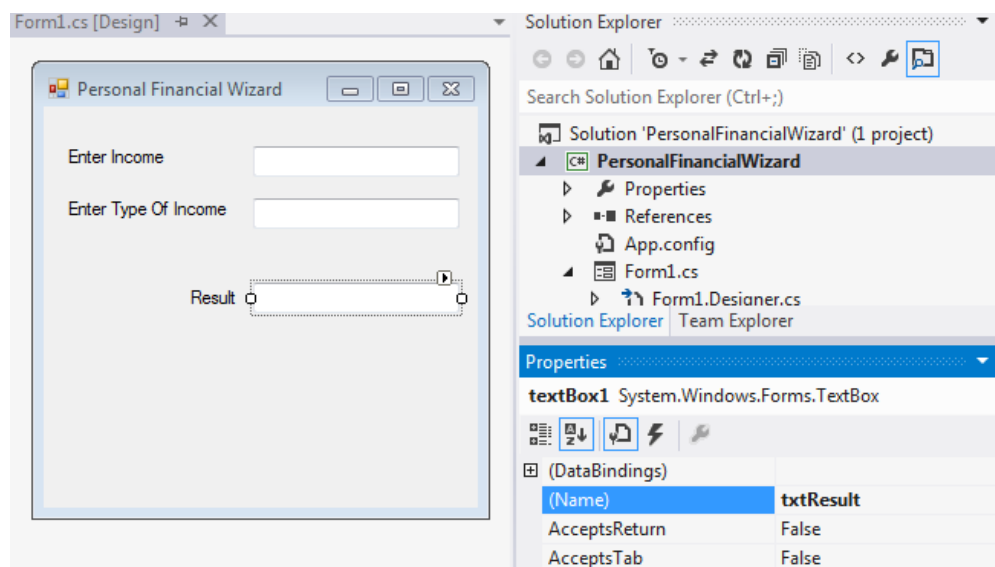


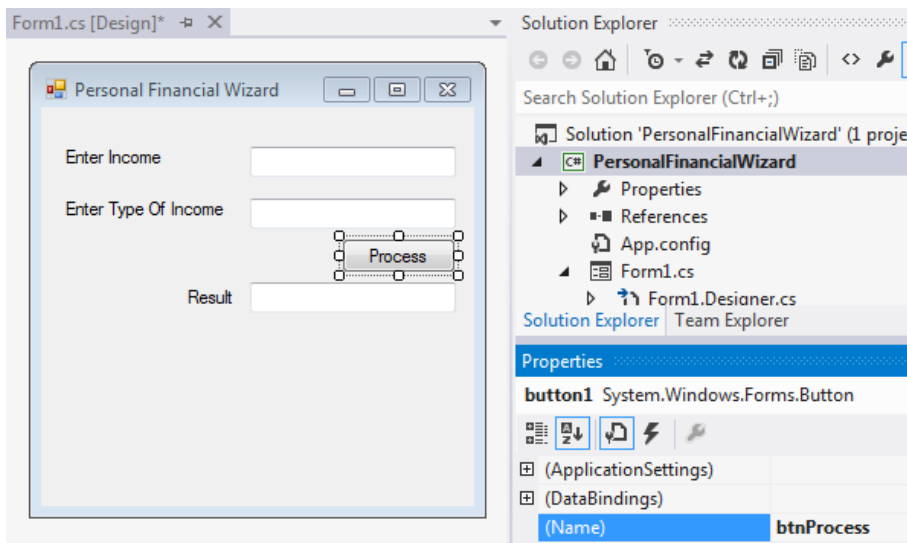
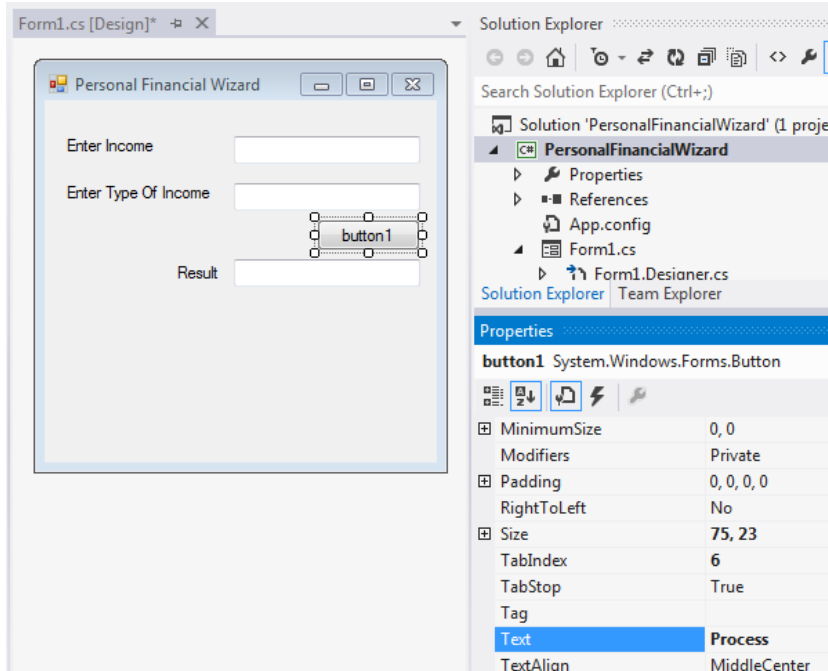








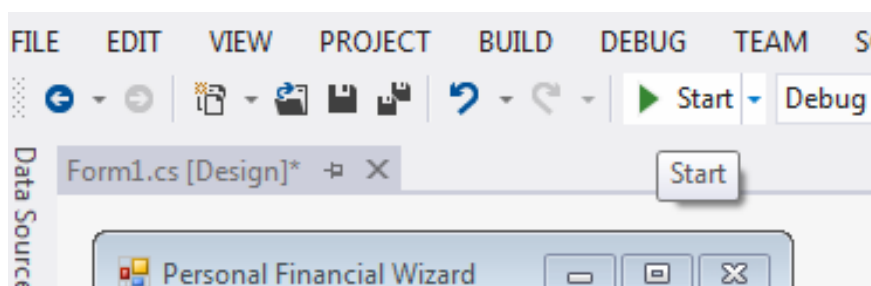


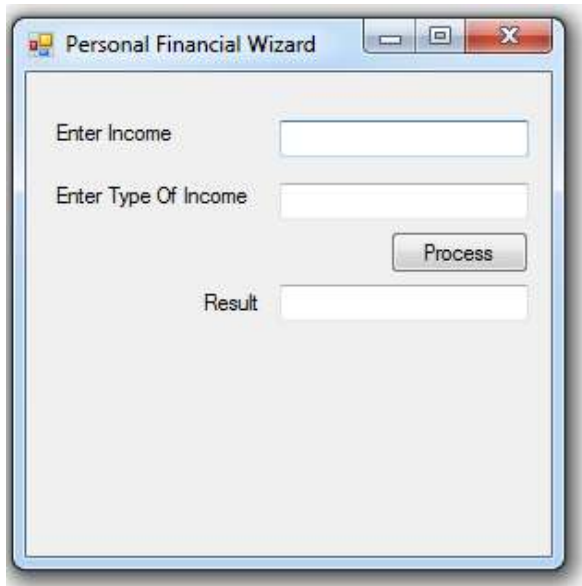


Here is the summary of what we have done

| Control | Property | Values | Comment |
|---------|----------|----------------------|---|
| Label | Text | Enter Income | Just a label printed on the form to indicate where user will enter income |
| Label | Text | Enter Type Of Income | Just a label printed on the form to indicate where user will enter the type of income e.g either annually monthly or biweekly |
| Label | Text | Result | Just a Label printed on the form to indicate where the result will appear |
| TextBox | Name | txtIncome | This specifies the reference to the box from which we will obtain the income the user has entered |
| TextBox | Name | txtType | This specifies the reference to the box from which we will obtain the type of income the user entered |
| TextBox | Name | txtResult | This specifies the reference to the box into which we will put the result of our computation |
| Button | Name | btnProcess | The user will press this button when the entry has been made. This name property specifies the reference to the button whose press event will notify us that the user is ready to process the entry |
| | Text | Process | This is the name printed on the button that gives the user the idea of what action will be carried out when the button is pressed |

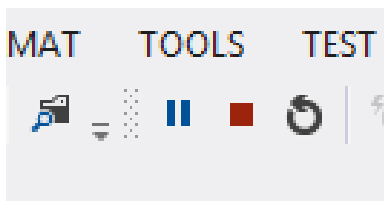
Let's try to run this by hitting the start button and see what we have



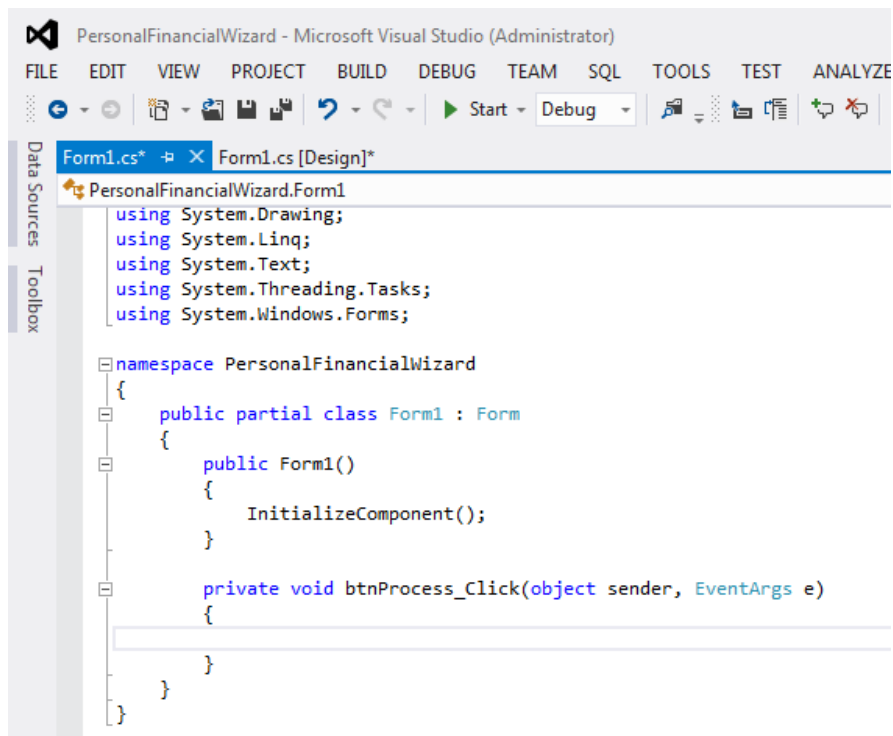
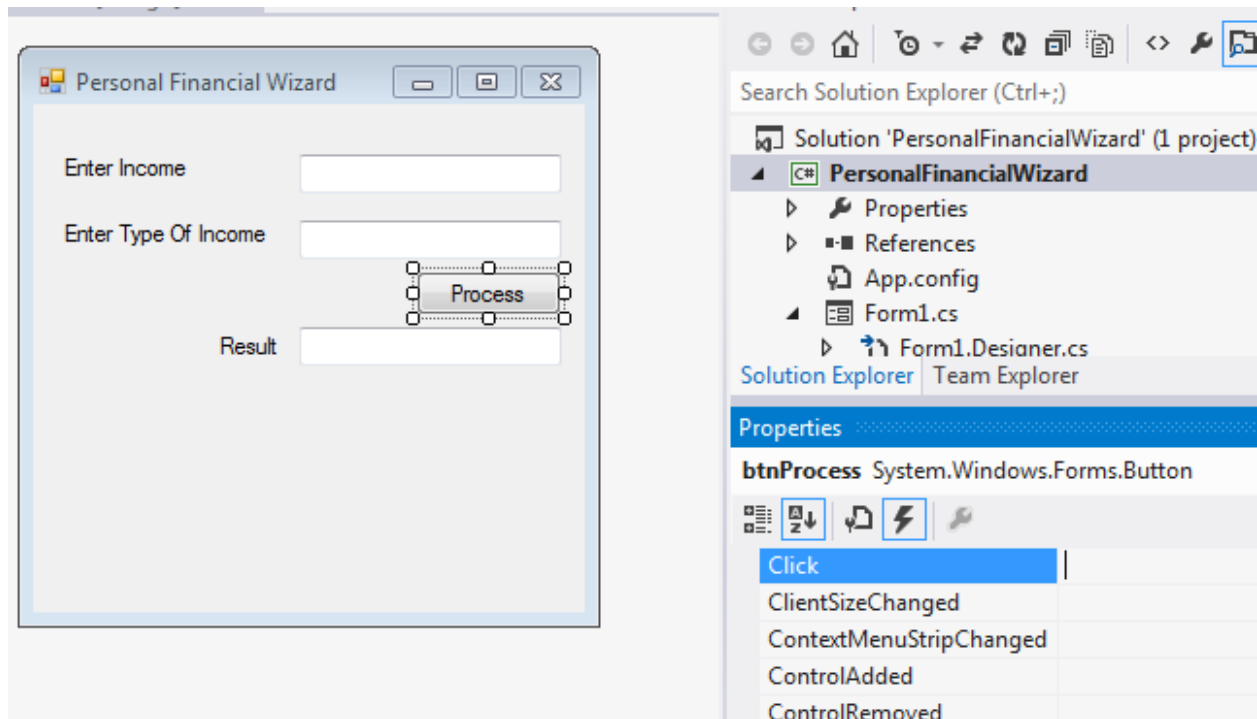


The program is simple but not useful in this state as we have not written any code to make it function. Also you'll notice that the form is not as descriptive and user friendly as we would have wanted it , but for the moment, let's keep it simple. I'll return to these points later. However if you have never done windows forms before, you would have had a lot to pick up thus far.

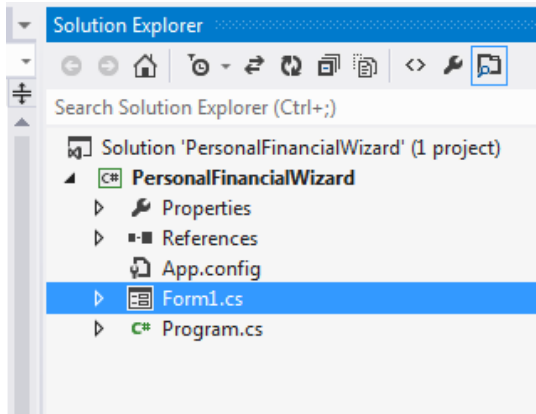
Close the form and switch back to the designer. You can also close the form by clicking on the "Stop Debugging" button in visual studio. This is generally standard with most visual studio projects



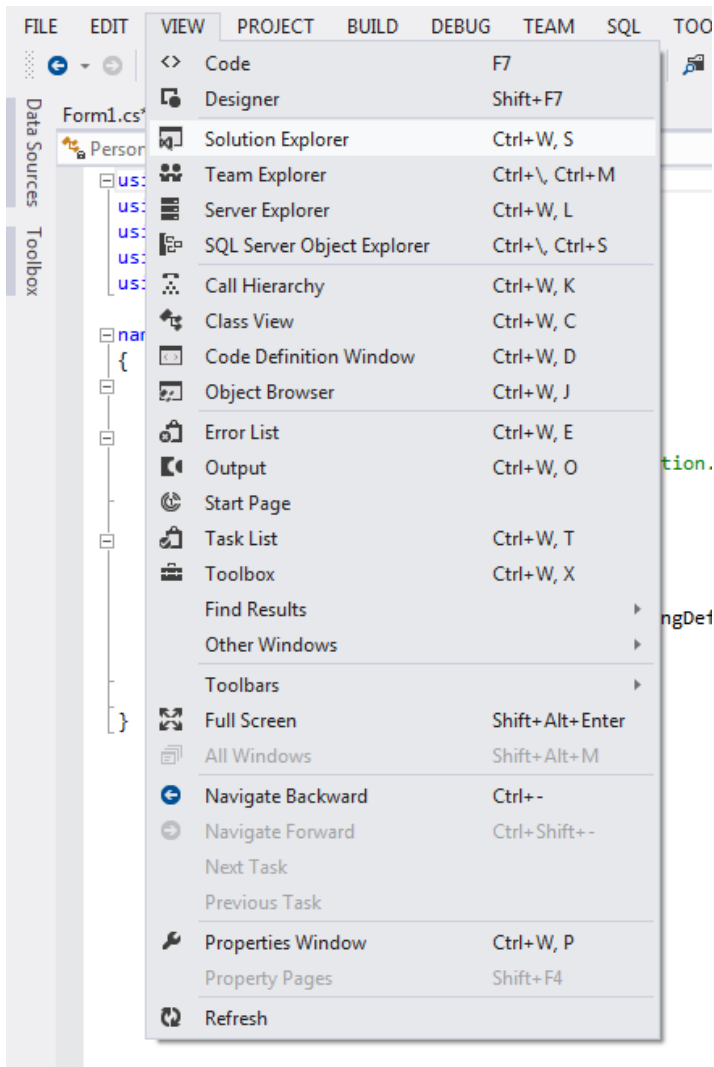
Then double click on the button and you will be taken straight to the region into which we will write the code to process the user entry as soon as they press the process button. This is actually called the button click event handler. You can also get to the event handler by clicking on the events button (with thunder bolt icon) in the property explorer, and then double clicking on the "Click" option.



You can switch back to the designer window at any time by going to the solution explorer and double clicking on the win form Icon of the form we are working on



Sometimes you may not find the property explorer or the solution explorer. You can go to View menu to bring them back up.



Now we are going to add code to process the user entry. We will be working with three variables which are txtIncome (which will hold the income amount), txtType (which will hold the type of income) and txtResult (into which we will save out result). What we want to do is to take the amount and multiply it by 10 (for ten years) to obtain the prediction for ten years. Add the following code to the btnProcess_Click event handler so it looks like this:

```
private void btnProcess_Click(object sender, EventArgs e)
{
    double income = Convert.ToDouble(txtIncome.Text);
    string type = txtType.Text;
    if (type == "annual")
    {
        txtResult.Text = (income * 10).ToString();
    }
    else
    {
        if (type == "monthly")
        {
            txtResult.Text = (income * 10 * 12).ToString();
        }
        else
        {
            if (type == "bi-weekly")
            {
                txtResult.Text = (income * 10 * 12 * 2).ToString();
            }
        }
    }
}
```

I will only add some comments to the same code for readability. Do read through to get a hang of what is going on.

```
private void btnProcess_Click(object sender, EventArgs e)
{
    ///comment
    ///we will convert the entry which is a string into a kind of number
    double income = Convert.ToDouble(txtIncome.Text);

    ///comment
    ///we will get the type of income from user entry and save into variable
    ///called type
    string type = txtType.Text;

    ///comment
    ///now we will check if user entered "annual"
    if (type == "annual")
    {
        ///comment
        ///multiply by 10 to obtain prediction for 10 years
        txtResult.Text = (income * 10).ToString();//save and send the result to
        ///the screen
    }else
    {
        ///comment
        ///otherwise we should check if user entered "monthly"
        if (type == "monthly")
        {
            ///comment
            ///multiply by 10 to obtain prediction for 10 years
            ///since there are 12 months in a year, also multiply by 12
            txtResult.Text = (income * 10 * 12).ToString();//save and send the
            ///result to the screen
        }
    }
}
```



```

else
    ///comment
    ///otherwise we should check if user entered "bi-weekly"
    if (type == "bi-weekly")
    {
        ///comment
        ///multiply by 10 to obtain prediction for 10 years
        ///since there are 12 months in a year, also multiply by 12
        ///since there are 2 bi-weekly in a month, we should multiply by
        ///2 as well
        txtResult.Text = (income * 10 * 12 * 2).ToString();//save and
        ///send the result to the screen
    }
}
}

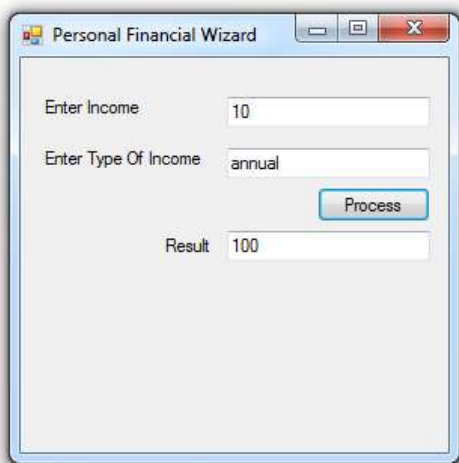
```

Now let us test if our program is working. Run the program and make the following entry CAREFULLY.

Enter 10 in the income box

Enter annual in the Type of income box

Then press "Process"



If you did not get the same result as the above then you must have done something wrong. Go over the process again till you get the same result.

Let see in summary what we accomplished

- We designed how the user interface will look like
- Then we wrote some code all in our button click event handler
- Then we tested to see if our program works

It works any way right?

The code we have written is manageable only if any of the following is true

- You will never add more functionality to the program
- Only you will use the program
- You will use the program for only one day like only today after which you will delete it
- You are just practicing the basics or windows form programming

Well if you intend to be a good developer, you cannot afford to do things this way. You cannot afford to think this way as well. In the next chapter, we will take a look at this problem and see how we can improve over this in the direction of development. You will understand that when you develop, the expectations from the final product include reliability, efficiency, security and maintainability, extensibility

Beginning C# Application Development Part III - Sample Problem | Better Way To Do It

*REQUEST THE REST OF
THE CHAPTERS AT
SAM@SAMQCODE.COM*