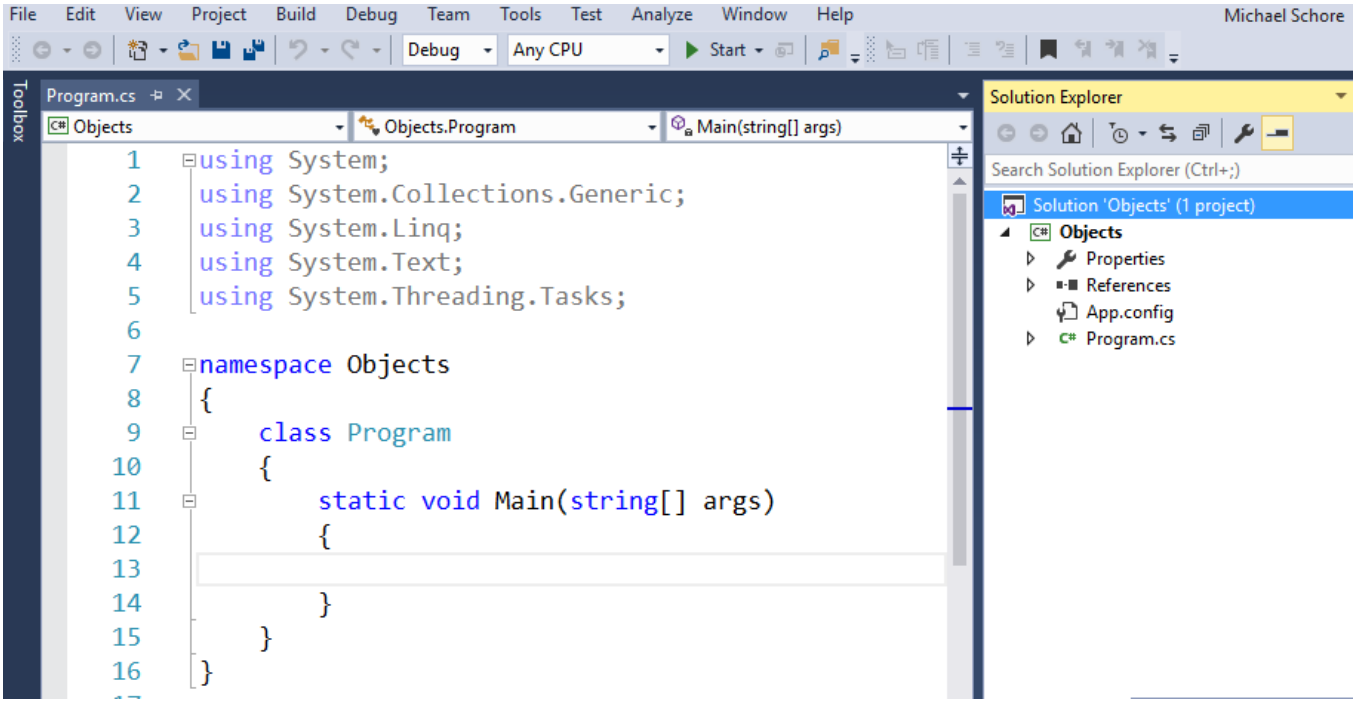
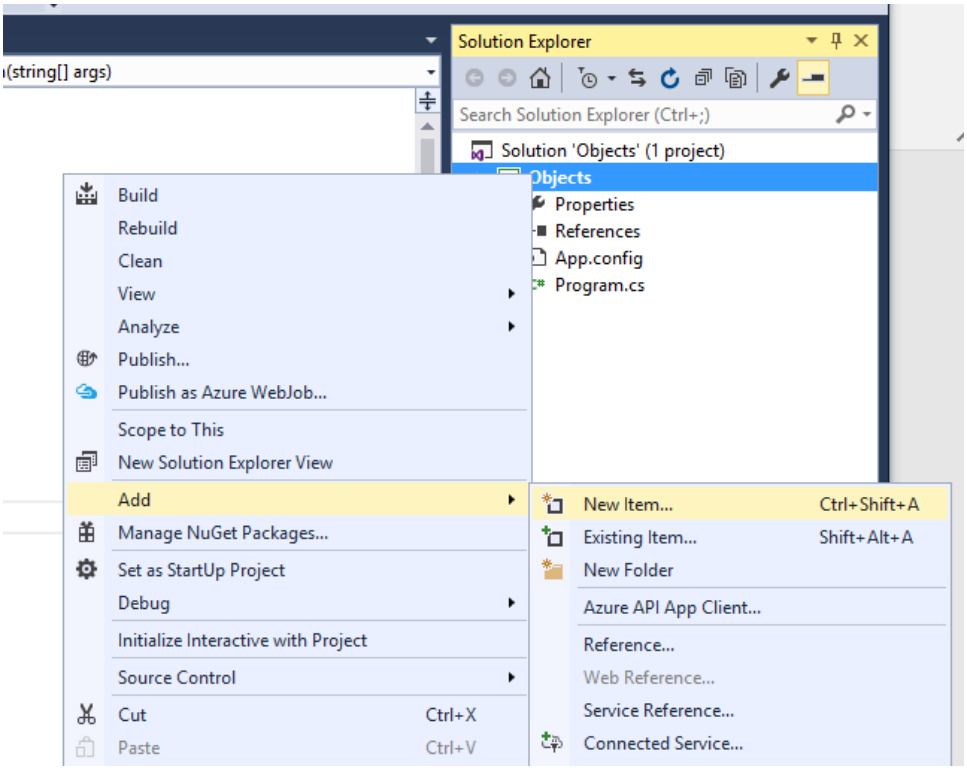
Lesson 3 – Working with OOP

This tutorial assumes that you have already created a console application in Visual Studio (VS). In the case below the project is named Objects. As always VS has created the shell of the project with a Main function that gets launched when the app is run.

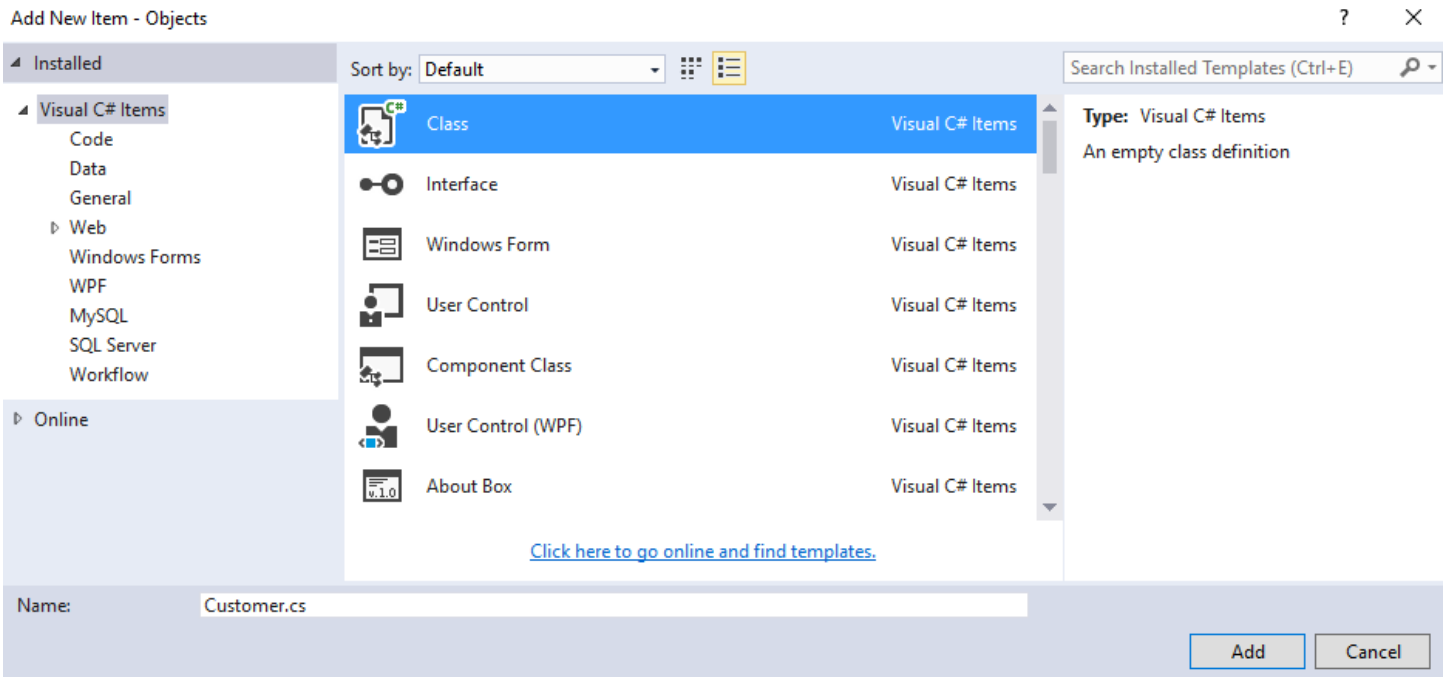
As a quick aside, you can have only one Main function within the project for console apps. VS needs some reference as to what needs to run when you run the app.



Let's turn our attention to objects now and create an additional file to house our Customer object in. Right-click the Objects name on the Solution Explorer and then roll over Add and then click New Item…



When the dialog opens, select the Class option and name the file Customer.cs.



By default, VS inserts a lot of using directives at the top of the file that will remain greyed unless we need a function from a particular library. For our simple class we won't need them but will leave in place until we are certain of that.

Whenever I create a new class I like to create the parts of the object in this order:

- Properties
- Constructors/Destructors
- Methods

Looking at the requirements of the Customer object we will need the following properties. You should always add comments when coding when it clarifies the way the code was written. See my zip code comment below.

```
// Properties
public int custID { get; set; }
public string fName { get; set; }
public string lName { get; set; }
public string address { get; set; }
public string city { get; set; }
public string state { get; set; }
public string zip { get; set; }        // use 5 or 5+4 digit zip
```

Once we get past the properties, we need to create a way to build or construct the object when needed. For that we use the constructor.

```
// Constructors

public Customer()
{
    custID = 0;          // 0 should not be used for valid customer
    fName = lName = address = city = zip = "";
}

public Customer(int id, string fn, string ln, string addr,
    string cy, string st, string zp)
{
    custID = id;
```

And finally, we add specific methods to our object. Remember, object properties are what they have while object methods are what they do. In the case of the Customer object I simply want you to output an address block of the Customer. The first line of that would look like:

```
    // Methods
    //
    // Output customer block to screen
    //
    public void PrintBlock()
    {
        Console.WriteLine("  Customer ID: " + custID.ToString());
```

And that is all there is to it. Since this object and its file are in the project we don't need a using reference in Program.cs. We simply reference the Customer object in a new statement and it knows what it is. See the image below.

```
7   namespace Objects
8   {
9       class Program
10      {
11          static void Main(string[] args)
12          {
13              Console.WriteLine();
14
15              // create new customer
16              Customer cust = new Objects.Customer(11011, "Michael", "Schore",
17                  "100 Main St.", "Raleigh", "NC", "27603");
18              // another way to create a new Customer
19              Customer nextCust = new Customer();
20
21              // ouput customer block
22              cust.PrintBlock();
23              Console.WriteLine();
24
```