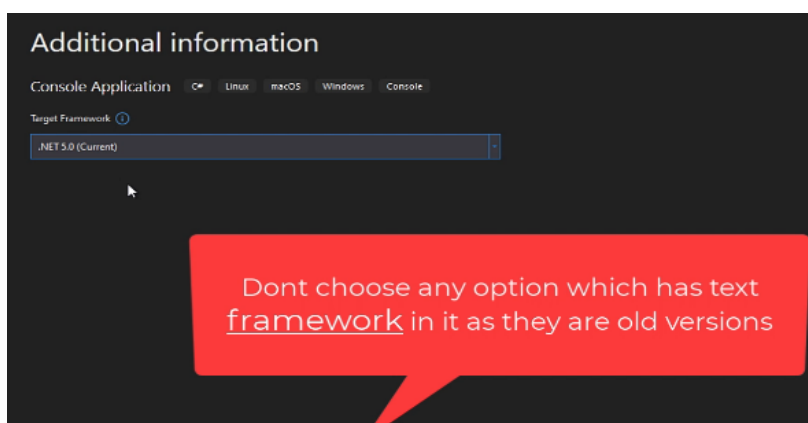
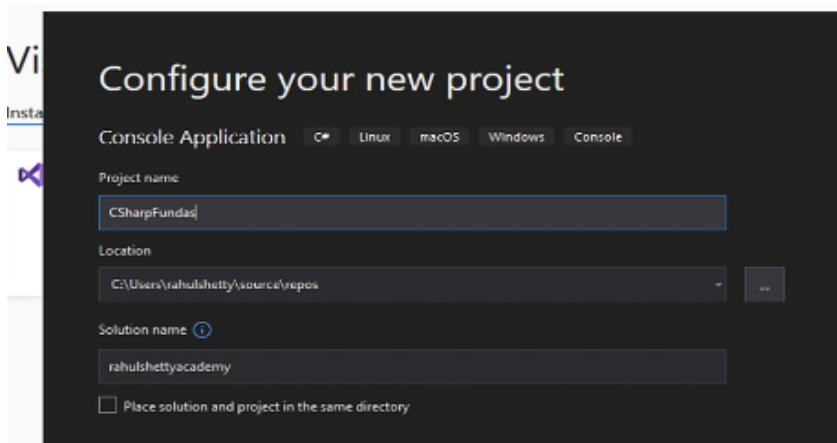
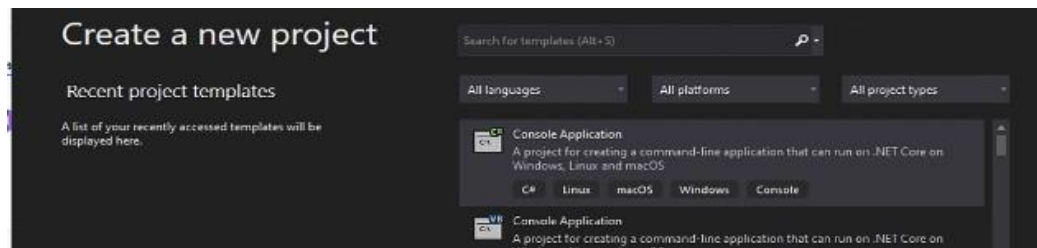
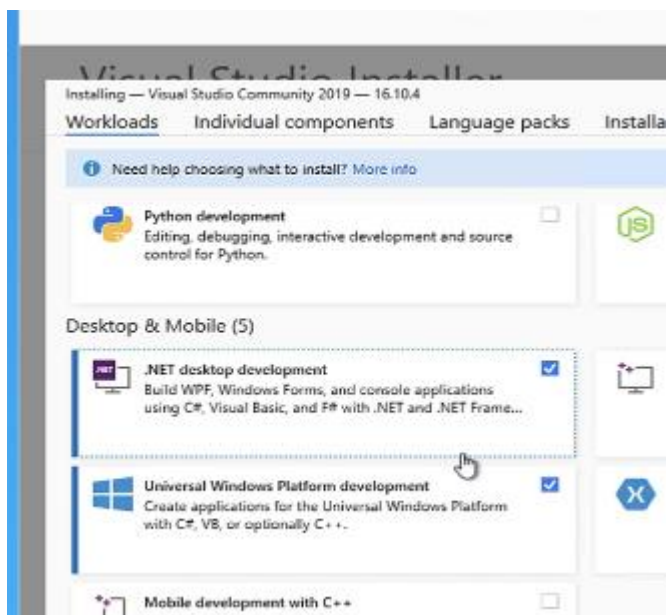


Environment setup:



In the project structure, bin stores the executable files (.dll), obj stores the compiled files, .sln stores the solution meta data details, .proj stores meta data about the project and .cs stores actual code.

```
using System;

namespace CSharpFundas
{
    class Program
    {
        static void Main(string[] args)
        {
            Console.WriteLine("Hello World!");
        }
    }
}
```

Or can write without using keyword.

```
System.Console.WriteLine("Hello World!");
```

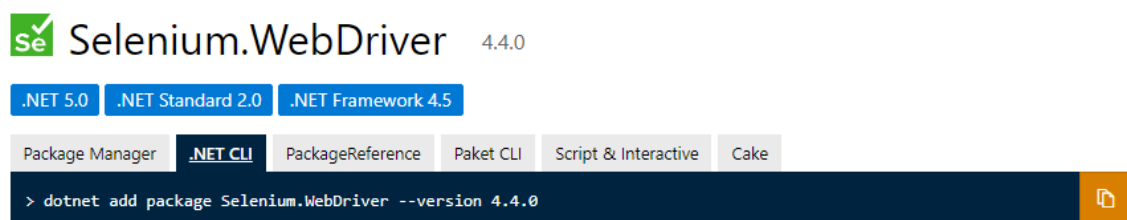
Debugging class.(Debug.WriteLine), to print on debug window.

Package manager for dotnet: <https://www.nuget.org>

Like maven in java, npm in javascript

Right click on project and select manage nuget packages.

Or in project folder, run below command.



Received from link: <https://www.nuget.org/packages/Selenium.WebDriver>

Right click – edit project file – see the PackageReference added.

Obj folder – compiled code for all files.

Bin folder – executable file(.dll/.exe) combined with all the compiled files above in obj folder.

Concatenation and evaluation string.

```
5 namespace CSharpFundas
6 {
7     class Program
8     {
9         static void Main(string[] args)
10        {
11            Console.WriteLine("Hello World!");
12            int a = 4;
13            Console.WriteLine("number is " + a);
14
15            String name = "Rahul";
16
17            Console.WriteLine("Name is " + name);
18
19            Console.WriteLine($"Name is {name}");
20        }
21    }
22 }
```

Static and dynamic datatypes.

int, float are static datatypes.

var is dynamic data type, which will set a type of data using run time assignment.

```
var age = 23;
```

readonly struct System.Int32
Represents a 32-bit signed integer.

Can't reassign a different type later, only done at first time.

dynamic. Same as var, but supports reassign also.

```
dynamic height = 13.2;
height = "hello";
```

Method.

```

class Program
{
    public void getData()
    {
        Console.WriteLine("I am inside the method");
    }
}

```

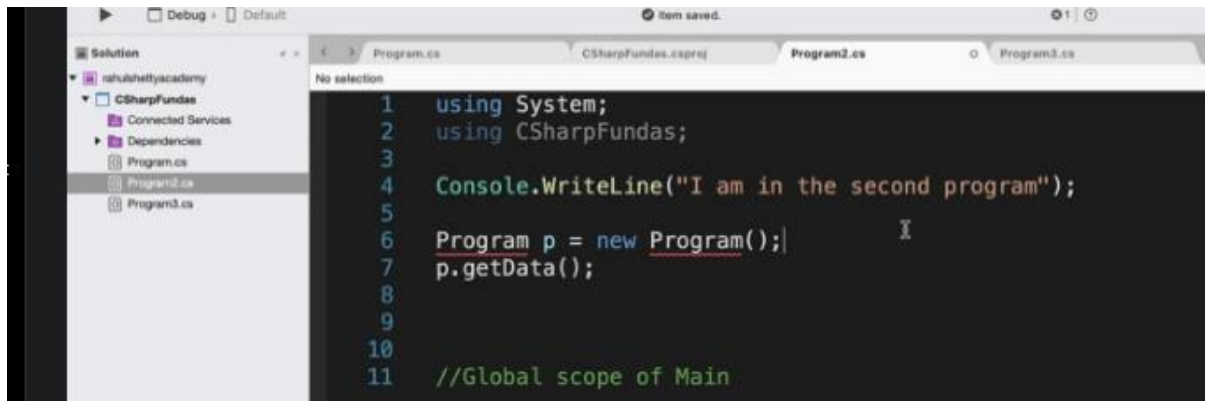
To use this, create object and use from that.

```

static void Main(string[] args)
{
    Program p = new Program();
    p.getData();
}

```

Using from another class.



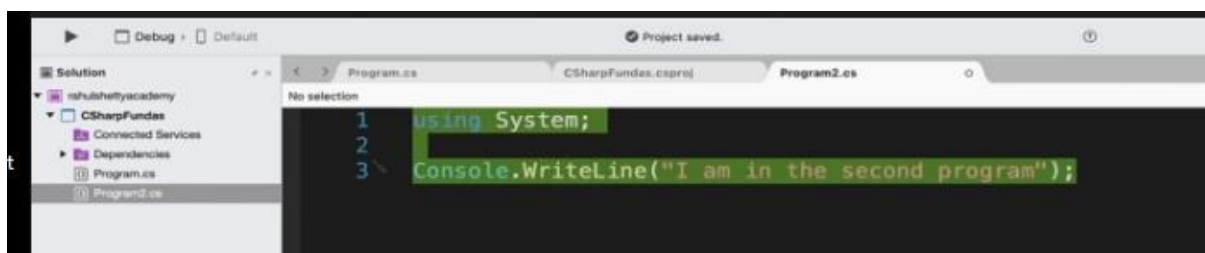
```

1  using System;
2  using CSharpFundas;
3
4  Console.WriteLine("I am in the second program");
5
6  Program p = new Program();
7  p.getData();
8
9
10
11 //Global scope of Main

```

We can write a new class file code without main block also, c# generates it at runtime.

Below code will come under Global scope of Main.



```

1  using System;
2
3  Console.WriteLine("I am in the second program");

```

Run selective file in the project with compiler setting.:

There can be only one Main entry in a project.

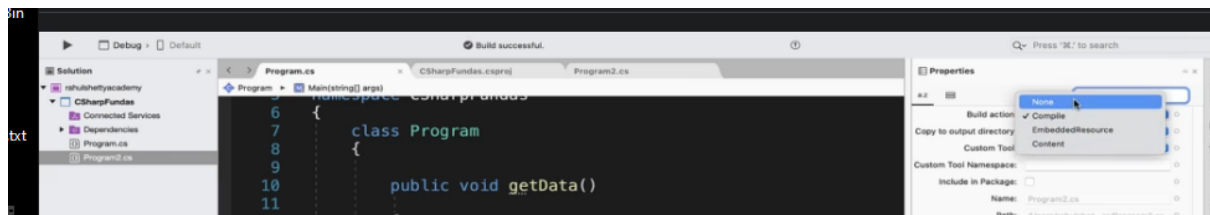
When multiple main are there in different cs files, global scope one will be taken.

If no global scope, then it will give error asking which to pick.

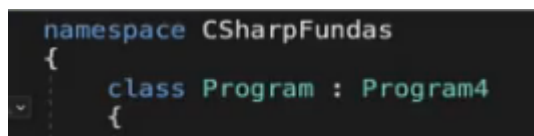
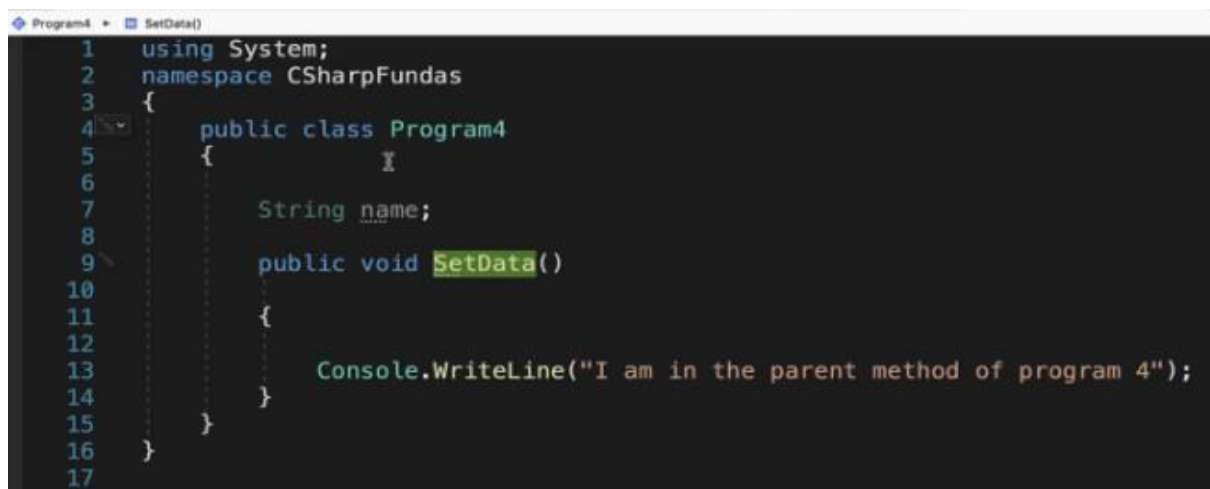
Right click on cs file – build action – compile

Right click on cs file – build action – None ... will skip the class from compiling when the project is build.

Or from properties of cs file.



Inheritance:



Constructor:

```

4
5 namespace CSharpFundas
6 {
7     class Program : Program4
8     {
9
10         String name;
11
12         //method default constructor
13
14         public Program(String name)
15         {
16             this.name = name;
17         }
18     }
19 }

```

Inside the Main method of class.

```

static void Main(string[] args)
{
    Program p = new Program("Rahul");
}

```

Can have multiple constructors with different arguments.

```

namespace CSharpFundas
{
    class Program : Program4
    {
        String name;
        String lastName;

        //method default constructor

        public Program(String name)
        {
            this.name = name;
        }

        public Program(String firstName, String lastName)
        {
            this.lastName = lastName;
        }
    }
}

```

```

static void Main(string[] args)
{
    Program p = new Program("Rahul");
    Program p1 = new Program("Rahul", "Shetty");
}

```

Arrays:

```
No selection
1  using System;
2
3  //
4  String[] a = { "hello", "bye", "Rahul", "Shetty" };
5  int[] b = { 1, 2, 3, 4, 5 };
6
7  String[] a1 = new String[4];
8  a1[0] = "hello";
9  a1[1] = "bye";
10
11 Console.WriteLine(a[1]);
12
13 for(int i =0;i<a.Length; i++)
14 {
15     Console.WriteLine(a[i]);
16     if(a[i] == "Rahul")
17     {
18         Console.WriteLine("Match found");
19         break;
20     }
21 }
22
23
24
25
26
```

ArrayList and enhanced for loops.

Size can be set at runtime.

```
1
2  using System;
3  using System.Collections;
4
5  ArrayList a = new ArrayList();
6  a.Add("hello");
7  a.Add("bye");
8  a.Add("Rahul");
9  a.Add("Apple");
10
11 Console.WriteLine(a[1]);
12
13
14 foreach(String item in a)
15 {
16     Console.WriteLine(item);
17 }
18
19
20 Console.WriteLine(a.Contains("Rahul"));
21 Console.WriteLine("After Sorting");
22 a.Sort();
23
24 foreach (String item in a)
25 {
26     Console.WriteLine(item);
27 }
28
29
```

