PARTIAL CLASS

While programming in C# (or OOP), we can split the definition of a class over two or more source files. The source files contains a section of the definition of class, and all parts are combined when the application is compiled. For splitting a class definition, we need to use the partial keyword

**Important points:**

* When you want to chop the functionality of the class, method, interface, or structure into multiple files, then you should use *partial*keyword and all the files are mandatory to available at compile time for creating final file.
* The *partial*modifier can only present instantly before the keywords like struct, class, and interface.
* Every part of the partial class definition should be in the same assembly and [namespace](https://www.geeksforgeeks.org/c-namespaces/), but you can use different source file name.
* Every part of the partial class definition should have the same accessibility like private, protected, etc.
* If any part of the partial class is declared as an abstract, sealed, or base, then the whole class is declared of the same type.
* The user is also allowed to use nested partial types.
* Dissimilar part may have dissimilar base types, but the final type must inherit all the base types.
* With the help of partial class multiple developers can work simultaneously on the same class in different files.
* With the help of partial class concept you can split the UI of design code and the business logic code to read and understand the code.
* When you were working with automatically generated code, the code can be added to the class without having to recreate the source file like in Visual studio.
* You can also maintain your application in an efficient manner by compressing large classes into small ones.

We have a project named as HeightWeightInfo which shows height and weight.

We have a file named as File1.cs with a partial class named as Record. It has two integer variables h & w and a method/constructor named as Record which is assigning the values of h & w.

amespace HeightWeightInfo

{

class File1

{

}

public partial class Record

{

private int h;

private int w;

public Record(int h, int w)

{

this.h = h;

this.w = w;

}

}

}

namespace HeightWeightInfo

{

class File2

{

}

public partial class Record

{

public void PrintRecord()

{

Console.WriteLine("Height:"+ h);

Console.WriteLine("Weight:"+ w);

}

}

}

Here now we can see the main method of the project:

namespace HeightWeightInfo

{

class Program

{

static void Main(string[] args)

{

Record myRecord = new Record(10, 15);

myRecord.PrintRecord();

Console.ReadLine();

}

}

}

Places where partial class can be used:

1. While working on a larger projects with more than one developer, it helps the developers to work on the same class simultaneously.
2. Codes can be added or modified to the class without re-creating source files which are automatically generated by the IDE (i.e. Visual Studio).

**Introuction to Partial Methods**

A partial class may contain a partial method. One part of the class contains the signature of the method. An optional implementation may be defined in the same part or another part. If the implementation is not supplied, then the method and all calls are removed at compile time.

**Example 2:**

Let's take an example as a partial class Car defined in file1.cs which has three methods InitializeCar(), BuildRim() and BuildWheels(). Among those methods, InitializeCar is defined as partial.

public partial class Car

{

partial void InitializeCar();

public void BuildRim() { }

public void BuildWheels() { }

}

And we have another file named as file2.cs which has two methods BuildEngine and InitializeCar. The method InitializeCar is partial method which is also defined in file1.cs.

public partial class Car

{

public void BuildEngine() { }

partial void InitializeCar()

{

string str = "Car";

}

}

A partial method declaration consists of two parts:

1. The definition as in file1.cs.
2. The implementation as in file2.cs.

They may be in separate parts of the partial class, or in the same part.

**Things to remember about Partial Method**

* partial keyword.
* return type void .
* implicitly private.
* and cannot be virtual.