**Partial methods in C# - Part 63**

**Suggested Videos**  
[Part 61 - Partial classes in C#](http://csharp-video-tutorials.blogspot.com/2012/11/partial-classes-in-c-part-61.html)  
[Part 62 - Creating partial classes in c#](http://csharp-video-tutorials.blogspot.com/2012/11/part-62-creating-partial-classes-in-c.html)  
  
**A partial class or a struct can contain partial methods. A partial method is created using the partial keyword.** Let us understand partial methods with an example. Create a console application. Add a class file, with name **PartialClassFileOne.cs**, to the project. copy and paste the following code.  
  
Notice, that, the **SampleMethod**() definition has the **partial keyword**, and does not have a **body(implementation) only the signature**. The implementation for a partial method is optional. If we don't provide the implementation, the compiler removes the signature and all calls to the method.  
  
**The implementation can be provided in the same physical file, or in another physical file**, that contains the partial class. In this example, the partial SampleMethod() is invoked in the PublicMethod().  
partial class SampleClass  
{  
    // Declaration of the partial method.  
    partial void SampleMethod();  
  
    // A public method calling the partial method  
    public void PublicMethod()  
    {  
        Console.WriteLine("Public Method Invoked");  
        SampleMethod();  
    }  
}

**Copy and paste the following code in the Main() method of the console application.**When we run the application now, notice that, we don't get a compiler error, in spite of not having an implementation for the partial **SampleMethod**(). Since, the implementation for the partial method is missing, the compiler will remove the signature and all calls to the method.  
SampleClass SC = new SampleClass();  
SC.PublicMethod();  
  
**Now, add a class file**, with name **PartialClassFileTwo.cs**. Copy and paste the following code. The implementation for the partial method is provided here.  
partial class SampleClass  
{  
    // Partial method implemented  
    partial void SampleMethod()  
    {  
        Console.WriteLine("Partial SampleMethod Invoked");  
    }  
}

**Now, run the console application and notice the output.** The partial method and the public method messages are printed on the console.   
  
**A partial method declaration consists of two parts.**  
**1.** The definition (only the method signature ending with a semi-colon, without method body)  
**2.** The implementation.   
**These may be in separate parts of a partial class, or in the same part.**  
  
**Partial methods are private by default**, and it is a compile time error to include any access modifiers, including private. The following code will raise an error stating - A partial method cannot have access modifiers or the virtual, abstract, override, new, sealed, or extern modifiers.  
partial class SampleClass  
{  
    private partial void SampleMethod();  
}  
  
**It is a compile time error, to include declaration and implementation at the same** time for a partial method. Code below produces a compile time error - No defining declaration found for implementing declaration of partial method 'PartialMethodsDemo.SampleClass.SampleMethod()'  
partial class SampleClass  
{  
    partial void SampleMethod()  
    {  
        Console.WriteLine("SampleMethod Implemented");  
    }  
}  
  
**A partial method return type must be void.** Including any other return type is a compile time error - Partial methods must have a void return type  
partial class SampleClass  
{  
    partial int SampleMethod();  
}  
  
**A partial method must be declared within a partial class or partial struct.** A non partial class or struct cannot include partial methods.  
  
**Signature of the partial method declaration**, must match with the signature of the implementation.

**A partial method can be implemented only once**. Trying to implement a partial method more than once, raises a compile time error - A partial method may not have multiple implementing declarations.