Начиная с версии 2.0, в C# появилась возможность разделять определение класса,

структуры или интерфейса на две или более части с сохранением каждой из них в отдельном

файле. Это делается с помощью контекстного ключевого слова partial. Все

эти части объединяются вместе во время компиляции программы.

(Можно в разных файлах описать один и тот же класс при помощи ключевого слова partial)

Если модификатор partial используется для создания частичного типа, то он

принимает следующую общую форму:

partial тип имя\_типа { // ...

где *имя\_типа* обозначает имя класса, структуры или интерфейса, разделяемого на части.

Каждая часть получающегося частичного типа должна указываться вместе с модификатором

partial.

Рассмотрим пример разделения простого класса, содержащего координаты XY, на

три отдельных файла. Ниже приведено содержимое первого файла.

partial class XY

{

public XY(int a, int b)

{

X = a;

Y = b;

}

}

Далее следует содержимое второго файла.

partial class XY

{

public int X { get; set; }

}

И наконец, содержимое третьего файла.

partial class XY

{

public int Y { get; set; }

}

В приведенном ниже файле исходного текста программы демонстрируется применение

класса XY.

// Продемонстрировать определения частичного класса.

using System;

class Test

{

static void Main()

{

XY xy = new XY(1, 2);

Console.WriteLine(xy.X + +xy.Y);

}

}

Для того чтобы воспользоваться классом XY, необходимо включить в компиляцию

все его файлы. Так, если файлы класса XY называются xy1.cs, ху2.cs и ху3.cs,

а класс Test содержится в файле test.cs, то для его компиляции достаточно ввести

в командной строке следующее. И последнее замечание: в C# допускаются частичные обобщенные классы. Но параметрытипа в объявлении каждого такого класса должны совпадать с теми, что указываются в остальных его частях.

**Частичные методы**

Начиная с версии 3.0, в C# появилась возможность

использовать этот модификатор и для создания *частичного метода* в элементе данных

частичного типа. Частичный метод объявляется в одной его части, а реализуется в другой.

Следовательно, с помощью модификатора partial можно отделить объявление

метода от его реализации в частичном классе или структуре.

Главная особенность частичного метода заключается в том, что его реализация не

требуется! Если частичный метод не реализуется в другой части класса или структуры,

то все его вызовы молча игнорируются. Это дает возможность определить, но не востребовать

дополнительные, хотя и не обязательные функции класса. Если эти функции

не реализованы, то они просто игнорируются.

Ниже приведена расширенная версия предыдущей программы, в которой создается

частичный метод Show(). Этот метод вызывается другим методом, ShowXY(). Ради

удобства все части класса XY представлены в одном файле, но они могут быть распределены

по отдельным файлам, как было показано в предыдущем разделе.

(***glava20\_13***)

partial class XY

{

public XY(int a, int b)

{

X = a;

Y = b;

}

//partial method

partial void Show();

}

partial class XY

{

public int X { get; set; }

//realize partial method Show

partial void Show()

{

Console.WriteLine("{0}, {1}", X, Y);

}

}

partial class XY

{

public int Y { get; set; }

//call part method

public void ShowXY()

{

Show();

}

}

class Test

{

static void Main()

{

XY xy = new XY(1, 2);

xy.ShowXY();

}}

Обратите внимание на то, что метод Show() объявляется в одной части класса XY,

а реализуется в другой его части. В реализации этого метода выводятся значения координат

X и Y. Это означает, что когда метод Show() вызывается из метода ShowXY(),

то данный вызов действительно имеет конкретные последствия: вывод значений

координат X и Y. Но если закомментировать реализацию метода Show(), то его вызов

из метода ShowXY() ни к чему не приведет.

Частичным методам присущ ряд следующих ограничений. Они должны возвращать

значение типа void. У них не может быть модификаторов доступа и они не могут

быть виртуальными. В них нельзя также использовать параметры out.