Практическая работа №3

Методы расширения

Цель работы: изучить создание методов расширения в языке C#.

Теоретическая часть

Методы расширения (`extension methods`) позволяют добавлять новые методы в уже существующие типы без создания нового производного класса. Эта функциональность бывает особенно полезна, когда требуется добавить в некоторый тип новый метод, но сам тип (класс или структуру) изменить нельзя ввиду отсутствия доступа к исходному коду. Либо если невозможно использовать стандартный механизм наследования, например, если классы определены с модификатором `sealed`.

Для того, чтобы создать метод расширения, вначале надо создать **статический** класс, который и будет содержать этот метод. Затем нужно объявить **статический** метод. По сути метод расширения – это обычный статический метод, который в качестве первого параметра всегда принимает конструкцию: `this имя_типа название_параметра`

Применение методов расширения очень удобно, но при этом надо помнить, что метод расширения никогда не будет вызван, если он имеет ту же сигнатуру, что и метод, изначально определенный в типе. Также следует учитывать, что методы расширения действуют на уровне пространства имен.

Рассмотрим пример создания метода расширения для класса `String`. Требуется создать метод, который подсчитывает количество вхождений определенного символа в строку.

```
public static class StringExtension
{
    0 references
    public static int CharCount(this string str, char c)
    {
        int counter = 0;
        for (int i = 0; i < str.Length; i++)
        {
            if (str[i] == c)
            {
                counter++;
            }
        }
        return counter;
    }
}
```

Рисунок 1 – Метод CharCount

Следует обратить внимание, что:

- Класс `StringExtension` является статическим;
- Метод `CharCount` является статическим;
- В качестве первого аргумента метода объявлено `this string str`, то есть этот метод будет относиться к типу `string`.

Теперь можно применить этот метод к строке, причём первый аргумент указывать не нужно.

```
string testString = "abcdef";
testString.CharCount('f');
```

Рисунок 2 – Вызов метода

Этот метод также можно увидеть при работе со строками.

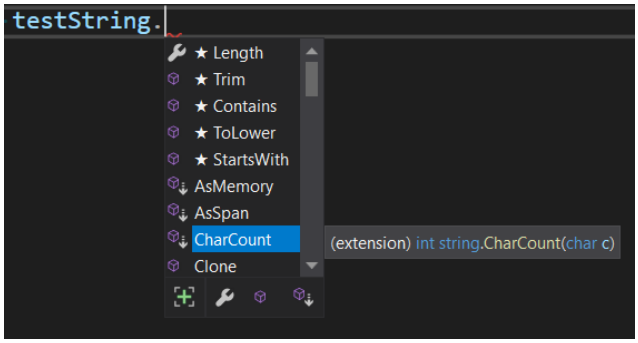

Рисунок 3 – Методы класса String

Практическая часть

1) Разработать структуру `StringInfo`, которая будет использоваться для хранения информации о строке, а именно: длина строки, количество цифр, количество букв.
2) Разработать метод расширения для класса `String`, который будет возвращать информацию о строке в виде структуры.
3) Разработать метод расширения для класса `Group` (ЛР№2 Переопределение методов и операторов). Назначение метода придумать самостоятельно.