Модификатор параметров out

Ключевое out инициирует передачу аргументов по ссылке. В результате этот формальный параметр становится псевдонимом для аргумента, который должен представлять собой переменную. Другими словами, любая операция в параметре осуществляется с аргументом. Оно схоже с ключевым словом ref за исключением того, что при использовании ref перед передачей переменную необходимо инициализировать. Оно также похоже на ключевое слово in за исключением того, что in не позволяет вызываемому методу изменять значение аргумента. Для применения параметра out определение метода и метод вызова должны явно использовать ключевое слово out. Пример:

```
int initializeInMethod;

OutArgExample(out initializeInMethod);

Console.WriteLine(initializeInMethod);    // value is now 44


void OutArgExample(out int number)

{

   number = 44;

}
```

Переменные, передаваемые в качестве аргументов out, не требуется инициализировать перед передачей в вызове метода. Но перед передачей управления из вызванного метода он должен присвоить значение.

Ключевые слова in, ref и out не считаются частью сигнатуры метода для разрешения перегрузки. Таким образом, методы не могут быть перегружены, если единственное различие состоит в том, что один метод принимает аргумент ref или in, а другой — out. Следующий код, например, компилироваться не будет.

```
class CS0663_Example

{

   // Compiler error CS0663: "Cannot define overloaded

   // methods that differ only on ref and out".

   public void SampleMethod(out int i) { }
```

```
    public void SampleMethod(ref int i) { }
}
```

Перегрузка допустима, если один метод принимает аргумент ref, in или out, а другой не использует ни один из этих модификаторов, как показано ниже.

```
class OutOverloadExample
{
    public void SampleMethod(int i) { }
    public void SampleMethod(out int i) => i = 5;
}
```

Компилятор выбирает наиболее подходящую перегрузку, сравнивая модификаторы параметров в месте вызова с модификаторами параметров в вызове метода.

Свойства не являются переменными и поэтому не могут быть переданы как параметры out.

Ключевые слова in, ref и out запрещено использовать для следующих типов методов.

- Асинхронные методы, которые определяются с помощью модификатора async.
- Методы итератора, которые включают оператор yield return или yield break.

Кроме того, методы расширения имеют следующие ограничения:

- Ключевое слово out нельзя использовать в первом аргументе метода расширения.
- Ключевое слово ref нельзя использовать в первом аргументе метода расширения, если аргумент не является структурой, или если универсальный тип не ограничен структурой.
- Ключевое слово in нельзя использовать, если только первый аргумент не является структурой. Ключевое слово in нельзя использовать ни для одного универсального типа, даже если оно ограничено структурой.

Объявление параметров out

Объявление метода с аргументами out — стандартное решение для возвращения нескольких значений. Начиная с версии C# 7.0, вы можете использовать кортежи значений для таких сценариев. В следующем примере используется out для возвращения трех переменных с помощью вызова одного метода. Третьему аргументу присвоено значение NULL. Это позволяет методам возвращать значения по желанию.

```
void Method(out int answer, out string message, out string stillNull)
{
   answer = 44;
   message = "I've been returned";
   stillNull = null;
}

int argNumber;
string argMessage, argDefault;
Method(out argNumber, out argMessage, out argDefault);
Console.WriteLine(argNumber);
Console.WriteLine(argMessage);
Console.WriteLine(argDefault == null);

// The example displays the following output:
//     44
//     I've been returned
//     True
```

Вызов метода с аргументом out

В C# 6 и более ранних версиях необходимо было объявлять переменную в отдельном операторе, прежде чем передавать ее как аргумент out. В

следующем примере переменная number объявляется перед передачей в метод Int32.TryParse, который пытается преобразовать строку в число.

```
string numberAsString = "1640";


int number;
if (Int32.TryParse(numberAsString, out number))
   Console.WriteLine($"Converted '{numberAsString}' to {number}");
else
   Console.WriteLine($"Unable to convert '{numberAsString}'");
// The example displays the following output:
//      Converted '1640' to 1640
```

Начиная с C# версии 7.0 переменную out можно объявлять в списке аргументов вызова метода, а не отдельно. Это делает код более кратким и удобным для восприятия, а также предотвращает непреднамеренное присвоение значения переменной перед вызовом метода. Следующий пример аналогичен предыдущему за тем исключением, что переменная number объявляется в вызове метода Int32.TryParse.

```
string numberAsString = "1640";


if (Int32.TryParse(numberAsString, out int number))
   Console.WriteLine($"Converted '{numberAsString}' to {number}");
else
   Console.WriteLine($"Unable to convert '{numberAsString}'");
// The example displays the following output:
//      Converted '1640' to 1640
```

В предыдущем примере переменная number строго типизирована как int. Вы также можете объявить неявно типизированную локальную переменную, как в приведенном ниже примере.

```
string numberAsString = "1640";
```

```
if (Int32.TryParse(numberAsString, out var number))
   Console.WriteLine($"Converted '{numberAsString}' to {number}");
else
   Console.WriteLine($"Unable to convert '{numberAsString}'");
// The example displays the following output:
//      Converted '1640' to 1640
```