Pass by Reference

In this lesson we'll discuss how to use pass by reference to pass parameters to methods in C#

WE'LL COVER THE FOLLOWING

Introduction

From the documentation :

Passing by *reference* enables function

- members
- methods
- properties
- indexers
- operators
- constructors

to change the value of the *parameters* and have that change persist in the calling environment. To pass a parameter by reference, use the `ref` keyword.

Syntax

Here's the syntax used to pass a value by *reference*:

```
//calling method DoubleNumber and passing the parameter number to it by reference
DoubleNumber(ref number); // calling code
```

```
DoubleNumber(ref number); // calling code

//passing the parameter number by reference in the definition of the method
DoubleNumber(ref int number)
{
  //body of code
}
```

Pass by Reference Syntax

Remarks

- *Parameters* passed with `ref` can be changed or left unchanged.
- For *reference* types, only the **addresses** of the *parameters* are passed to the function.
- In pass by *reference* type once the value gets changed in *function* then the *original* variable's value gets changed.

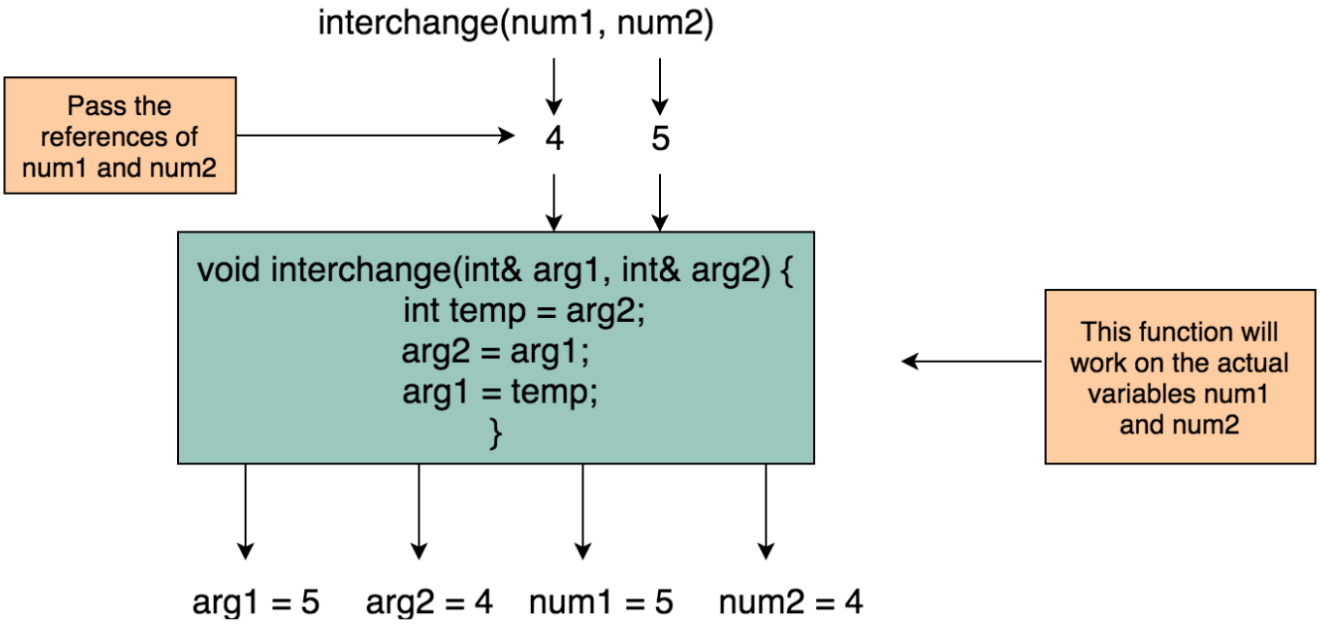
Example

Let's take a look at an example implementing the *pass by reference* way of passing *parameters* to *methods*.

```
using System;

class PassByRefExample
{
    static int num1 = 4;
    static int num2 = 5;

    static void Main()
    {
        System.Console.WriteLine("Before num1 is {0} , num2 is {1}", num1, num2);
        System.Console.WriteLine("Calling interchange function");
        interchange(ref num1,ref num2);
        System.Console.WriteLine("Now num1 is {0} , num2 is {1}", num1, num2);
    }
    private static void interchange(ref int arg1, ref int arg2) // passing parameters by refe
    {
        int temp = arg2;  //creating a variable temp and setting equal to arg2
        arg2 = arg1;    // setting the value of arg2 equal to arg1
        arg1 = temp;  //setting the value of arg1 equal to temp which is equal to arg2
    }
}
```



Explanation

In the code above

- We made the method `interchange(ref int arg1, ref int arg2)`
- It takes two values by *reference* and **swaps** their values
- The *method* `interchange` is then called in the `Main` with **4** and **5** passed as the values of `num1` and `num2`
- Since these values are passed by *reference* as seen in line **12**, they automatically get updated hence the type of `interchnage` method is `void`.

In the next lesson we'll take a look at *method overloading*.