https://www.tutorialsteacher.com/csharp/csharp-func-delegate

C# - Func Delegate

C# includes built-in generic delegate types Func and Action, so that you don't need to define custom delegates manually in most cases.

Func is a generic delegate included in the System namespace. It has zero or more *input* parameters and one *out* parameter. The last parameter is considered as an out parameter.

The Func delegate that takes one input parameter and one out parameter is defined in the System namespace, as shown below:

Signature: Func

namespace System

{

public delegate TResult Func<in T, out TResult>(T arg);

}

The last parameter in the angle brackets <> is considered the return type, and the remaining parameters are considered input parameter types, as shown in the following figure.

[Diagram

Description automatically generated](https://www.tutorialsteacher.com/Content/images/csharp/func-delegate.png)Func delegate

A Func delegate with two input parameters and one out parameters will be represented as shown below.

[Diagram

Description automatically generated with medium confidence](https://www.tutorialsteacher.com/Content/images/csharp/func-delegate2.png)Func delegate

The following Func delegate takes two input parameters of int type and returns a value of int type:

Func<int, int, int> sum;

You can assign any method to the above func delegate that takes two *int* parameters and returns an *int* value.

Example: Func

class Program

{

static int Sum(int x, int y)

{

return x + y;

}

static void Main(string[] args)

{

Func<int,int, int> add = Sum;

int result = add(10, 10);

Console.WriteLine(result);

}

}

[Try it](https://www.tutorialsteacher.com/codeeditor?cid=cs-2yWptk)

Output:

20

A Func delegate type can include 0 to 16 input parameters of different types. However, it must include an out parameter for the result. For example, the following Func delegate doesn't have any input parameter, and it includes only an out parameter.

Example: Func with Zero Input Parameter

Func<int> getRandomNumber;

C# Func with an Anonymous Method

You can assign an anonymous method to the Func delegate by using the delegate keyword.

Example: Func with Anonymous Method

Func<int> getRandomNumber = delegate()

{

Random rnd = new Random();

return rnd.Next(1, 100);

};

Func with Lambda Expression

A Func delegate can also be used with a lambda expression, as shown below:

Example: Func with lambda expression

Func<int> getRandomNumber = () => new Random().Next(1, 100);

//Or

Func<int, int, int> Sum = (x, y) => x + y;

[Try it](https://www.tutorialsteacher.com/codeeditor?cid=cs-CPmcNQ)

 Points to Remember :

1. Func is built-in delegate type.
2. Func delegate type must return a value.
3. Func delegate type can have zero to 16 input parameters.
4. Func delegate does not allow ref and out parameters.
5. Func delegate type can be used with an [anonymous method](https://www.tutorialsteacher.com/csharp/csharp-anonymous-method) or [lambda expression](https://www.tutorialsteacher.com/linq/linq-lambda-expression).