<https://www.tutorialsteacher.com/csharp/csharp-action-delegate>

# C# - Action Delegate

Action is a delegate type defined in the System namespace. An Action type delegate is the same as [Func delegate](https://www.tutorialsteacher.com/csharp/csharp-func-delegate) except that the Action delegate doesn't return a value. In other words, an Action delegate can be used with a method that has a void return type.

For example, the following delegate prints an int value.

Example: C# Delegate

public delegate void Print(int val);

static void ConsolePrint(int i)

{

Console.WriteLine(i);

}

static void Main(string[] args)

{

Print prnt = ConsolePrint;

prnt(10);

}

Output:

10

You can use an Action delegate instead of defining the above Print delegate, for example:

Example: Action delegate

static void ConsolePrint(int i)

{

Console.WriteLine(i);

}

static void Main(string[] args)

{

Action<int> printActionDel = ConsolePrint;

printActionDel(10);

}

[Try it](https://www.tutorialsteacher.com/codeeditor?cid=cs-Th00fz)

You can initialize an Action delegate using the new keyword or by directly assigning a method:

Action<int> printActionDel = ConsolePrint;

//Or

Action<int> printActionDel = new Action<int>(ConsolePrint);

An Action delegate can take up to 16 input parameters of different types.

An Anonymous method can also be assigned to an Action delegate, for example:

Example: Anonymous method with Action delegate

static void Main(string[] args)

{

Action<int> printActionDel = delegate(int i)

{

Console.WriteLine(i);

};

printActionDel(10);

}

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

Output:

10

A Lambda expression also can be used with an Action delegate:

Example: Lambda expression with Action delegate

static void Main(string[] args)

{

Action<int> printActionDel = i => Console.WriteLine(i);

printActionDel(10);

}

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

Thus, you can use any method that doesn't return a value with Action delegate types.

## Advantages of Action and Func Delegates

1. Easy and quick to define delegates.
2. Makes code short.
3. Compatible type throughout the application.

 Points to Remember :

1. Action delegate is same as func delegate except that it does not return anything. Return type must be void.
2. Action delegate can have 0 to 16 input parameters.
3. Action delegate can be used with [anonymous methods](https://www.tutorialsteacher.com/csharp/csharp-anonymous-method) or [lambda expressions](https://www.tutorialsteacher.com/linq/linq-lambda-expression).