

# System.Func<-T1,-T2,-T3,-T4,-T5,+TResult> Delegate

```
[ILAsm]
.class public sealed System.Func`6<-T1,-T2,-T3,-T4,-T5,+TResult> extends
System.MulticastDelegate

[C#]
public delegate TResult Func<in T1,in T2,in T3,in T4,in T5,out TResult>(T1
arg1, T2 arg2, T3 arg3, T4 arg4, T5 arg5);
```

## Assembly Info:

- *Name:* mscorlib
- *Public Key:* [00 00 00 00 00 00 00 00 04 00 00 00 00 00 00 00]
- *Version:* 4.0.0.0
- *Attributes:*
  - CLSCompliantAttribute(true)

## Summary

Encapsulates a method that has five parameters and returns a value of the type specified by the *TResult* parameter.

## Parameters

Parameter	Description
<i>arg1</i>	The first parameter of the method that this delegate encapsulates.
<i>arg2</i>	The second parameter of the method that this delegate encapsulates.
<i>arg3</i>	The third parameter of the method that this delegate encapsulates.
<i>arg4</i>	The fourth parameter of the method that this delegate encapsulates.
<i>arg5</i>	The fifth parameter of the method that this delegate encapsulates.

## Inherits From: System.MulticastDelegate

**Library:** BCL

## Returns

The return value of the method that this delegate encapsulates.

## Description

You can use this delegate to represent a method that can be passed as a parameter without explicitly declaring a custom delegate. The encapsulated method must correspond to the method signature that is defined by this delegate. This means that the encapsulated method must have five parameters, each of which is passed to it by value, and that it must return a value.

[*Note:* To reference a method that has five parameters and returns `void`, use the generic `System.Action`5<T1,T2,T3,T4,T5>` delegate instead.

]

You can also use the `System.Func`6<T1,T2,T3,T4,T5,TResult>` delegate with anonymous methods.