

1 System.Func<-T,+TResult> Delegate

```
2 [ILAsm]  
3 .class public sealed System.Func`2<-T,+TResult> extends  
4 System.MulticastDelegate  
  
5 [C#]  
6 public delegate TResult Func<in T,out TResult>(T arg);
```

7 Assembly Info:

- 8 • *Name:* mscorlib
- 9 • *Public Key:* [00 00 00 00 00 00 00 00 04 00 00 00 00 00 00 00]
- 10 • *Version:* 4.0.0.0
- 11 • *Attributes:*
 - 12 ○ CLSCompliantAttribute(true)

13 Summary

14 Encapsulates a method that has one parameter and returns a value of the type specified
15 by the *TResult* parameter.

16 Parameters

Parameter	Description
<i>arg</i>	The parameter of the method that this delegate encapsulates.

17 Inherits From: System.MulticastDelegate

18 **Library:** BCL

22 Returns

23 The return value of the method that this delegate encapsulates.

26 Description

27 You can use this delegate to represent a method that can be passed as a parameter
28 without explicitly declaring a custom delegate. The encapsulated method must
29 correspond to the method signature that is defined by this delegate. This means that the
30 encapsulated method must have one parameter that is passed to it by value, and that it
31 must return a value.

32
33 [*Note:* To reference a method that has one parameter and returns `void`, use the generic
34 `System.Action`1<T>` delegate instead.

35

1]

2

3 When you use the `System.Func`2<T1, TResult>` delegate, you do not have to explicitly
4 define a delegate that encapsulates a method with a single parameter.

5

6 You can use the `System.Func`2<T1, TResult>` delegate with anonymous methods in C#.
7 (For an introduction to anonymous methods, see the C# standard.)

8