System.Func<-T1,-T2,+TResult> Delegate

```
[ILAsm]
.class public sealed System.Func`3<-T1,-T2,+TResult> extends
System.MulticastDelegate

[C#]
public delegate TResult Func<in T1,in T2,out TResult>(T1 arg1, T2 arg2);
```

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 two parameters and returns a value of the type specified by the *TResult* parameter.

Parameters

Parameter	Description
arg1	The first parameter of the method that this delegate encapsulates.
arg2	The second 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 two parameters, each of which is passed to it by value, and that it must return a value.

[*Note:* To reference a method that has two parameters and returns `void`, use the generic `System.Action`2<T1,T2>` delegate instead.

]

When you use the `System.Func`3<T1,T2,TResult>` delegate, you do not have to explicitly define a delegate that encapsulates a method with two parameters.

You can use the `System.Func`3<T1,T2,TResult>` delegate with anonymous methods in C#. (For an introduction to anonymous methods, see the C# standard.)