System.Action<-T1,-T2> Delegate

```
[ILAsm]
.class public sealed System.Action`2<-T1,-T2> extends
System.MulticastDelegate

[C#]
public delegate void Action<in T1,in T2>(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 does not return a value.

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

Description

You can use the `System.Action`2<T1,T2>` delegate to pass a method 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 that are both passed to it by value, and it must not return a value. Typically, such a method is used to perform an operation.

[*Note:* To reference a method that has two parameters and returns a value, use the generic `System.Func`3<T1,T2,TResult>` delegate instead.

]

When you use the `System.Action`2<T1,T2>` delegate, you do not have to explicitly

define a delegate that encapsulates a method with two parameters.

You can also use the `System.Action`2<T1,T2>` delegate with anonymous methods in C#. (For an introduction to anonymous methods, see the C# standard.)