

C# 4.0: Method parameter default values as array

Asked 9 years, 4 months ago Active 9 years, 4 months ago Viewed 5k times

▲ Is it possible in C# 4.0 method default values parameters as array (for ex. `string[] sArray`)? if yes, how to implement it?

6

I tried call function like below:



```
MethodA(string[] legends=new string[]{"a","b"},float[] values=new float[]{1,2}, string  
alt="sd");
```



1

It's not work

.net-4.0

c#-4.0

edited Jun 29 '10 at 6:45

asked Jun 29 '10 at 6:30



loviji

5,171

17

53

85

2 Answers



10

As others have said, default values can't be arrays. However, one way of avoiding this is to make the default value null and then initialize the array if necessary:



```
public void Foo(int[] x = null)  
{  
    x = x ?? new int[] { 5, 10 };  
}
```



Or if you're not going to mutate the array or expose it to the caller:

```
private static readonly int[] FooDefault = new int[] { 5, 10 };
public void Foo(int[] x = null)
{
    x = x ?? FooDefault;
}
```

Note that this assumes `null` isn't a value that you'd want to use for any other reason. It's not a globally applicable idea, but it works well in some cases where you can't express the default value as a compile-time constant. You can use the same thing for things like `Encoding.UTF8` as a default encoding.

If you want a value type parameter, you can just make that a nullable one. For example, suppose you wanted to default a parameter to the number of processors (which clearly isn't a compile-time constant) you could do:

```
public void RunInParallel(int? cores = null)
{
    int realCores = cores ?? Environment.ProcessorCount;
}
```

answered Jun 29 '10 at 6:39



[Jon Skeet](#)

1142k 721 8243
8622

Default values must be compile-time constant, which means arrays cannot be used.

3

The standard (pg 312) says this:

The expression in a default-argument must be one of the following:

- a constant-expression
- an expression of the form `new S()` where `S` is a value type
- an expression of the form `default(S)` where `S` is a value type


edited Jun 29 '10 at 6:39


answered Jun 29 '10 at 6:34



[Dean Harding](#)

62.2k 8 125 170

Oh wow that's terrible. It's the same issue as attributes except I'm pretty sure you can actually use arrays for them. Why can't you declare an initialised array? They are constant – [Matt Mitchell](#) Jun 29 '10 at 6:36 

-
- 1 @Graphain: In what way is any array other than an empty one constant? Non-empty arrays are always mutable. The compiler would have to create a copy of the array on each call, in case something (e.g. the called method) had changed the contents. – [Jon Skeet](#) Jun 29 '10 at 6:42 

@Jon Skeet - Well one would assume that if defaults require a constant the compiler would be smart enough to treat an array as read only. The only way that is changing is if someone tinkers with reflection right, which would be the same issue for true constants (e.g. string etc.). – [Matt Mitchell](#) Jun 29 '10 at 6:51

-
- 2 @Graphain: There's no such thing as a read-only array. Would the method not be able to pass it to anything else which took an array argument either? You're basically talking about a different type here. – [Jon Skeet](#) Jun 29 '10 at 7:03

@Jon Skeet, my mistake. Still it's not like it would let you use a read-only IEnumerable type is it? – [Matt Mitchell](#) Jun 30 '10 at 1:12
