System.Threading.AutoResetEvent Class

```
[ILAsm]
.class public sealed beforefieldinit System.Threading.AutoResetEvent
extends System.Threading.EventWaitHandle

[C#]
public sealed class AutoResetEvent: System.Threading.EventWaitHandle
```

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

Notifies a waiting thread that an event has occurred. This class cannot be inherited.

Inherits From: System.Threading.EventWaitHandle

Library: BCL

Description

`AutoResetEvent` allows threads to communicate with each other by signaling. Typically, this communication concerns a resource to which threads need exclusive access.

A thread waits for a signal by calling `System.Threading.WaitHandle.WaitOne` on the `AutoResetEvent`. If the `AutoResetEvent` is in the non-signaled state, the thread blocks, waiting for the thread that currently controls the resource to signal that the resource is available by calling `System.Threading.EventWaitHandle.Set`.

Calling `Set` signals `AutoResetEvent` to release a waiting thread. `AutoResetEvent` remains signaled until a single waiting thread is released, and then automatically returns to the non-signaled state. If no threads are waiting, the state remains signaled indefinitely.

If a thread calls `System.Threading.WaitHandle.WaitOne` while the `System.Threading.AutoResetEvent` is in the signaled state, the thread does not block. The `System.Threading.AutoResetEvent` releases the thread immediately and returns to the non-signaled state.

[*Note:* There is no guarantee that every call to the `System.Threading.EventWaitHandle.Set` method will release a thread. If two calls are too close together, so that the second call occurs before a thread has been released, only one thread is released. It is as if the second call did not happen. Also, if `System.Threading.EventWaitHandle.Set` is called when there are no threads waiting and the `System.Threading.AutoResetEvent` is already signaled, the call has no effect.

]

You can control the initial state of an `AutoResetEvent` by passing a Boolean value to the constructor, `true` if the initial state is signaled and `false` otherwise.

`AutoResetEvent` can also be used with the `staticSystem.Threading.WaitHandle.WaitAll` and `System.Threading.WaitHandle.WaitAny` methods.

AutoResetEvent(System.Boolean) Constructor

```
[ILAsm]
.method public hidebysig specialname rtspecialname instance void
.ctor(bool initialState) cil managed

[C#]
public AutoResetEvent (bool initialState)
```

Summary

Initializes a new instance of the `System.Threading.AutoResetEvent` class with a Boolean value indicating whether to set the initial state to signaled.

Parameters

Parameter	Description
initialState	`true` to set the initial state to signaled; `false` to set the initial state to non-signaled.