C# Threads

Jim Fawcett
CSE681 – Software Modeling and Analysis
Fall 2005

Thread Class

- Every Win32 thread is passed a function to run when created.
  - When the thread returns from the function it terminates.
- In C#, Threads are managed by the System.Threading.Thread class.
  - C# threads are passed a static or instance function of some C# class using a standard delegate of type **ThreadStart**.

Starting C# Threads

-
  ```
  Thread thread =
     new Thread(new ThreadStart(ThreadFunc));
  ```

-
  ```
  thread.Start();
  ```
- ThreadFunc can be:
  - Static or instance member of the class instance that created the thread
  - Static or instance member of some other class, e.g.:

    ThreadStart(SomeClass.aStaticFunction);
    ThreadStart(someClassInstance.aNonStaticFunction);

Thread States

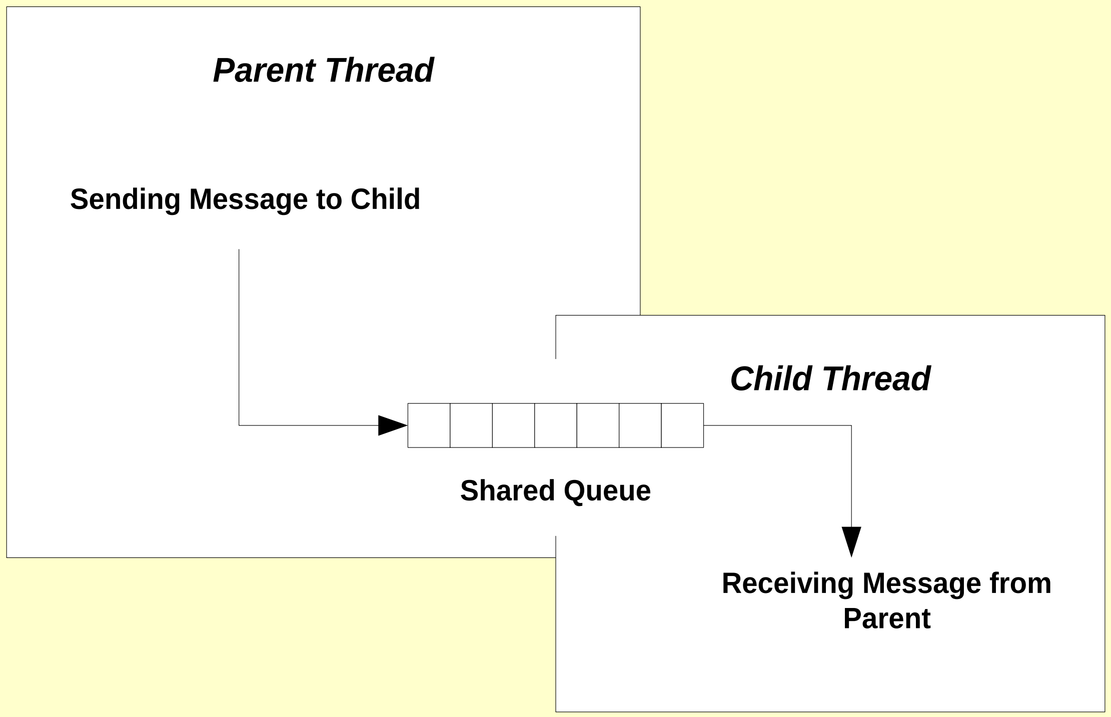
- A thread that has been started, but not yet terminated can be in one of the following states:
  - Running
  - Waiting to run
  - Suspended
  - Blocked

Thread Properties

- ***IsBackground*** – get, set
  - Process does not end until all Foreground threads have ended.
  - Background threads are terminated when application ends.
- **CurrentThread** – get, static
  - Returns thread reference to calling thread
- ***IsAlive*** – get
  - Has thread started but not terminated?
- ***Priority*** – get, set
  - Highest, AboveNormal, Normal, BelowNormal, Lowest
- ***ThreadState*** – get
  - Unstarted, Running, Suspended, Stopped, WaitSleepJoin, ..

Sharing Resources

- A child thread often needs to communciate with its parent thread. It does this via some shared resource, like a queue.



Synchronization

- When two or more threads share a common resource access needs to be serialized - a process called synchronization.
  - Consider the shared queue on the previous slide. Should the parent start to enqueue an element in an empty queue, but have its time-slice expire before finishing, the queues links are in an undefined state.
  - Now, if the child thread wakes up, and attempts to dequeue an element the result is undefined.

Synchronization with C# Lock

```
// send messages to child thread

     string msg = "";
     for(int i=0; i<50; ++i)
     {
       msg = "message #" + i.ToString();
       Console.Write("\n  Sending {0},",msg);

       // Enqueuing changes links so must lock

       lock(demo.threadQ) { demo.threadQ.Enqueue(msg); }

       // control writer speed - twice as fast as reader

       Thread.Sleep(50);
     }

     lock(demo.threadQ) { demo.threadQ.Enqueue("end"); }

     child.Join();
     Console.Write(
       "\n\n  child thread state = {0}\n\n",child.ThreadState.ToString()
     );
```

Demonstration Program

- QueuedMessages folder
  - Illustrates communication between parent and child threads using a queue.
  - Also illustrates use of C# lock operation.

Other Locking Mechanisms

- The .Net Threading Library also provides:
  - ***Monitor***
    - Locks an object, like C# lock, but provides more control.
  - ***Interlocked***
    - Provides atomic operations on 32 bit and 64 bit data types, e.g., ints, longs, pointers.
  - ***Mutex***
    - Guards a region of code.
    - Can synchronize across process boundaries.
  - ***AutoResetEvent and WaitOne***
    - Allows fine-grained control of the sequencing of thread operations.
  - ***ReaderWriterLock***
    - Locks only when writing, allowing free reads.

Locking Certain Collections

- ArrayList, Hashtable, Queue, Stack, and other collections provide Synchronized() function, supporting high performance locking.

  ```
  ArrayList unsync = new ArrayList();
  ArrayList sync = ArrayList.Synchronized(unsynch);
  ```

  Your code needs no lock constructs with sync.

Method Decoration

- Methods can be decorated with a MethodImpl attribute, synchronizing access much like a Win32 critical section.

```
[MethodImpl (MethodImplOptions.Synchronized)]
string myMethod(string input)
{
     …
}
```

Note that this synchronizes a region of code, while lock and Monitor synchronize objects.

WinForms and Worker Threads

- A UI thread is a thread that creates a window. A worker thread is a thread spawned by a UI thread to do work in the background while the UI thread services UI messages.

- A worker thread must never access UI functions directly. It accesses them through Form's Invoke, BeginInvoke, and EndInvoke functions, passing a delegate as an argument.

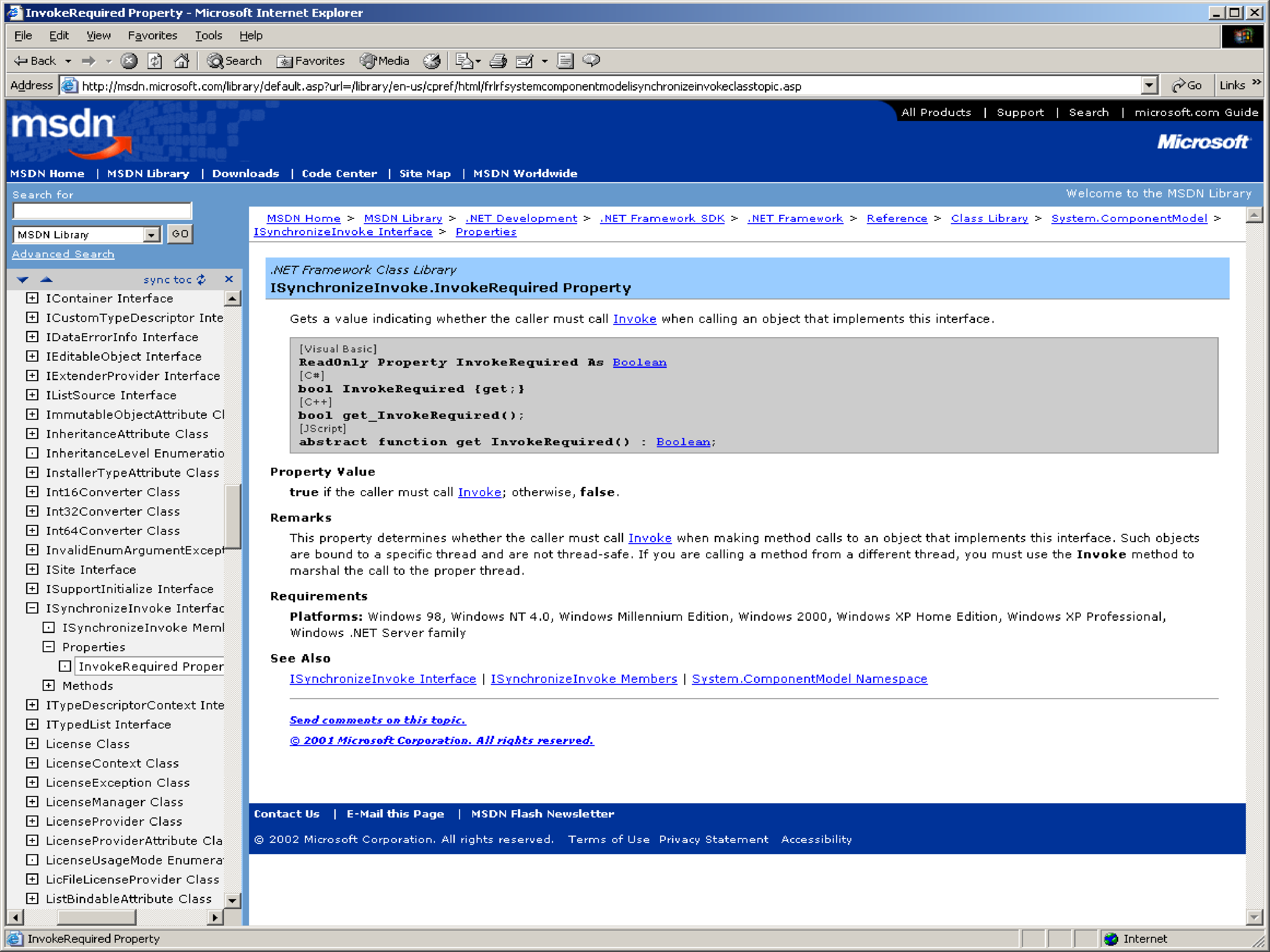
BeginInvoke Example

```
for (i = 1; i <= 25; i++)
{
  s = "Step number " + i.ToString() + " executed";
  Thread.Sleep(400);

  // Make asynchronous call to main form.

  // MainForm.AddString function runs in main thread
  // because we activated the delegate through form's
  // Invoke (synchronous) or BeginInvoke (asynchronous) functions.
  // To make synchronous call use Invoke.

  m_form.BeginInvoke(m_form.m_DelegateAddString, new Object[] {s});

  // check if thread is cancelled
  if ( m_EventStop.WaitOne(0, true) )
  {
    // clean-up operations may be placed here
    // ...

    // inform main thread that this thread stopped
    m_EventStopped.Set();

    return;
  }
}
```

Delegate arguments passed as an array of objects

Demonstration Programs

- ProcessDemo and ProcessDemoWin32
  - Illustrates creating a child process
- QueuedMessages
  - Illustrates communication between threads using queues and the C# lock operation.
- FormInvokeDemo folder
  - A more interesting demonstration of the above.
- WorkerThread folder
  - Simple Demonstration of UI and Worker thread communication using Form.Invoke(…)
- ThreadPoolDemo folder
  - Illustrates how to use the ThreadPool to run functions



InvokeRequired Property - Microsoft Internet Explorer

Address http://msdn.microsoft.com/library/default.asp?url=/library/en-us/cpref/html/frlrfsystemcomponentmodelisynchronizeinvokeclasstopic.asp

File Edit View Favorites Tools Help
Back Search Favorites Media
Go Links
All Products | Support | Search | microsoft.com Guide
MSDN Home | MSDN Library | Downloads | Code Center | Site Map | MSDN Worldwide
Search for
MSDN Library GO
Advanced Search
sync toc
Welcome to the MSDN Library

.NET Framework Class Library

ISynchronizeInvoke.InvokeRequired Property

Gets a value indicating whether the caller must call Invoke when calling an object that implements this interface.

```
[Visual Basic]
ReadOnly Property InvokeRequired As Boolean
[C#]
bool InvokeRequired {get;}
[C++]
bool get_InvokeRequired();
[JScript]
abstract function get InvokeRequired() : Boolean;
```

Property Value

true if the caller must call Invoke; otherwise, **false**.

Remarks

This property determines whether the caller must call Invoke when making method calls to an object that implements this interface. Such objects are bound to a specific thread and are not thread-safe. If you are calling a method from a different thread, you must use the **Invoke** method to marshal the call to the proper thread.

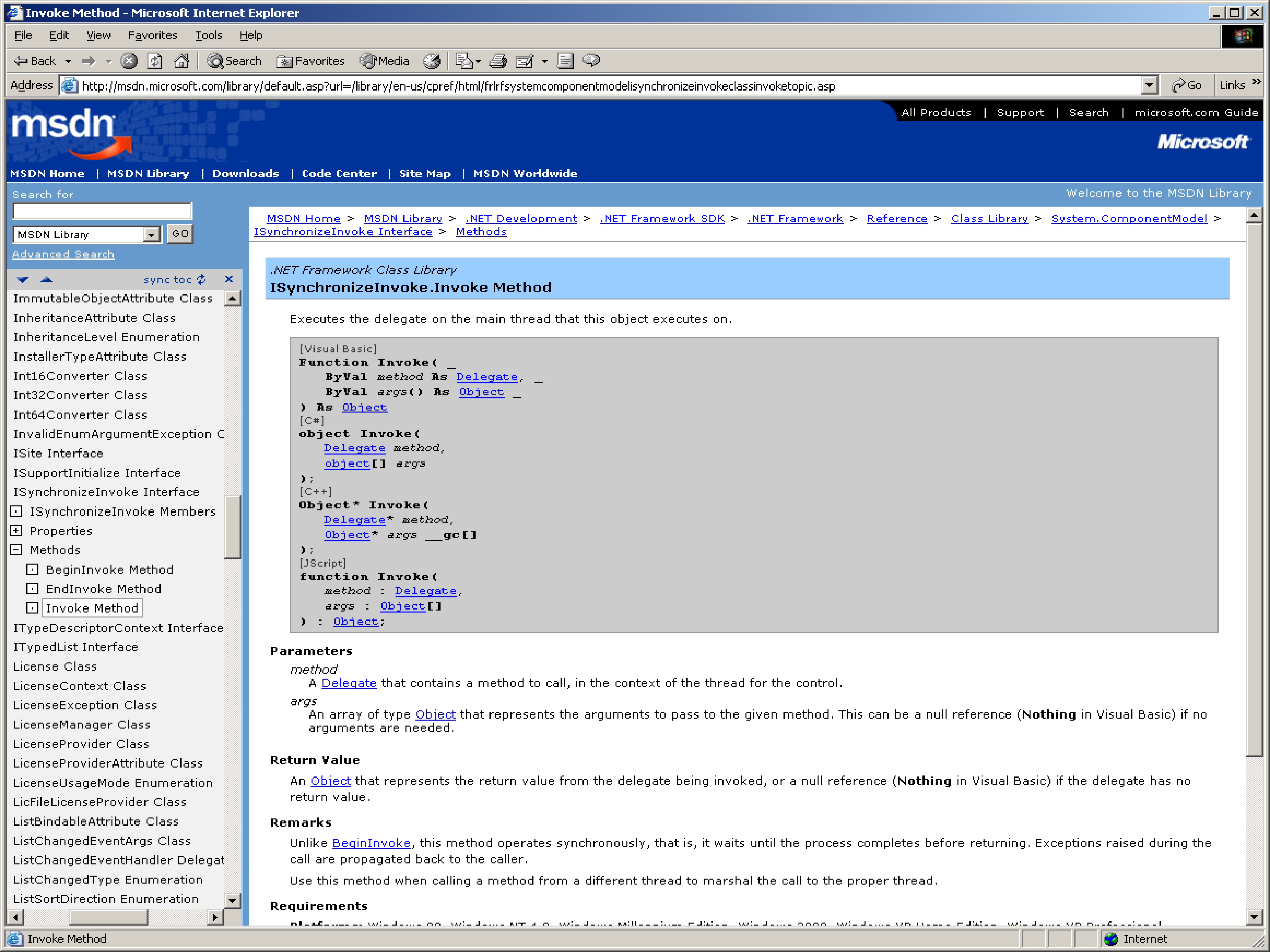
Requirements

Platforms: Windows 98, Windows NT 4.0, Windows Millennium Edition, Windows 2000, Windows XP Home Edition, Windows XP Professional, Windows .NET Server family

See Also

ISynchronizeInvoke Interface | ISynchronizeInvoke Members | System.ComponentModel Namespace

Send comments on this topic.


Invoke Method - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Search Favorites Media
Address http://msdn.microsoft.com/library/default.asp?url=/library/en-us/cpref/html/frlrfsystemcomponentmodelisynchronizeinvokeclassinvoketopic.asp
Go Links
msdn
All Products | Support | Search | microsoft.com Guide
Microsoft
MSDN Home | MSDN Library | Downloads | Code Center | Site Map | MSDN Worldwide
Search for
MSDN Library
GO
Advanced Search
sync toc
Welcome to the MSDN Library
ImmutableObjectAttribute Class
InheritanceAttribute Class
InheritanceLevel Enumeration
InstallerTypeAttribute Class
Int16Converter Class
Int32Converter Class
Int64Converter Class
InvalidEnumArgumentException C
ISite Interface
ISupportInitialize Interface
ISynchronizeInvoke Interface
ISynchronizeInvoke Members
Properties
Methods
BeginInvoke Method
EndInvoke Method
Invoke Method
ITypeDescriptorContext Interface
ITypedList Interface
License Class
LicenseContext Class
LicenseException Class
LicenseManager Class
LicenseProvider Class
LicenseProviderAttribute Class
LicenseUsageMode Enumeration
LicFileLicenseProvider Class
ListBindableAttribute Class
ListChangedEventArgs Class
ListChangedEventHandler Delegat
ListChangedType Enumeration
ListSortDirection Enumeration
Invoke Method
Internet

MSDN Home > MSDN Library > .NET Development > .NET Framework SDK > .NET Framework > Reference > Class Library > System.ComponentModel > ISynchronizeInvoke Interface > Methods

.NET Framework Class Library

ISynchronizeInvoke.Invoke Method

Executes the delegate on the main thread that this object executes on.

```
[Visual Basic]
Function Invoke( _
   ByVal method As Delegate, _
   ByVal args() As Object _
) As Object
[C#]
object Invoke(
   Delegate method,
   object[] args
);
[C++]
Object* Invoke(
   Delegate* method,
   Object* args __gc[]
);
[JScript]
function Invoke(
   method : Delegate,
   args : Object[]
) : Object;
```

Parameters

method
A Delegate that contains a method to call, in the context of the thread for the control.

args
An array of type Object that represents the arguments to pass to the given method. This can be a null reference (**Nothing** in Visual Basic) if no arguments are needed.

Return Value

An Object that represents the return value from the delegate being invoked, or a null reference (**Nothing** in Visual Basic) if the delegate has no return value.

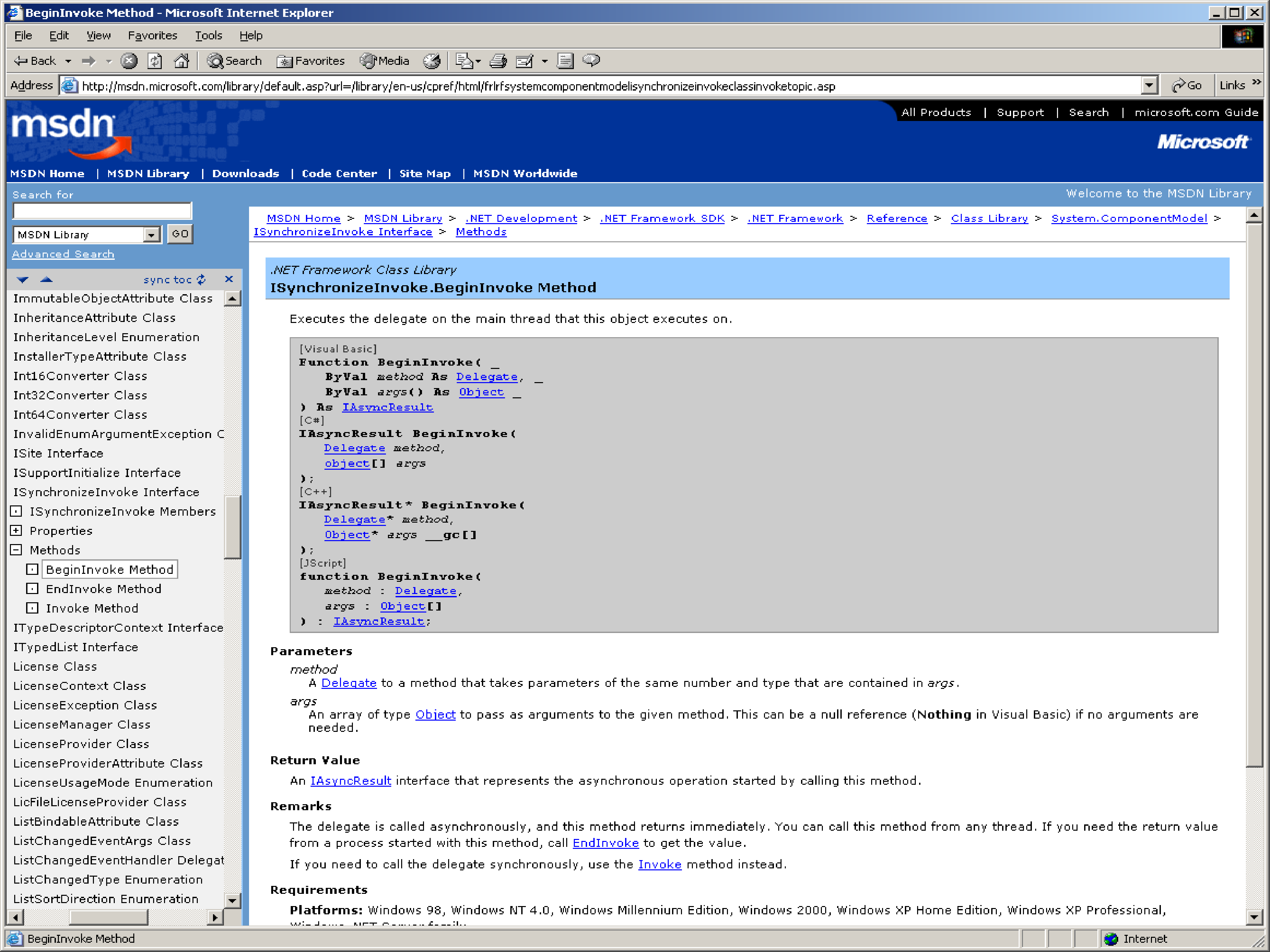
Remarks

Unlike BeginInvoke, this method operates synchronously, that is, it waits until the process completes before returning. Exceptions raised during the call are propagated back to the caller.

Use this method when calling a method from a different thread to marshal the call to the proper thread.

Requirements

Platforms: Windows 98, Windows NT 4.0, Windows Millennium Edition, Windows 2000, Windows XP Home Edition, Windows XP Professional



BeginInvoke Method - Microsoft Internet Explorer

Address: http://msdn.microsoft.com/library/default.asp?url=/library/en-us/cpref/html/frlrfsystemcomponentmodelisynchronizeinvokeclassinvoketopic.asp

MSDN Home | MSDN Library | Downloads | Code Center | Site Map | MSDN Worldwide

All Products | Support | Search | microsoft.com Guide

Welcome to the MSDN Library

MSDN Home > MSDN Library > .NET Development > .NET Framework SDK > .NET Framework > Reference > Class Library > System.ComponentModel > ISynchronizeInvoke Interface > Methods

.NET Framework Class Library

ISynchronizeInvoke.BeginInvoke Method

Executes the delegate on the main thread that this object executes on.

```
[Visual Basic]
Function BeginInvoke( _
   ByVal method As Delegate, _
   ByVal args() As Object _
) As IAsyncResult
[C#]
IAsyncResult BeginInvoke(
   Delegate method,
   object[] args
);
[C++]
IAsyncResult* BeginInvoke(
   Delegate* method,
   Object* args __gc[]
);
[JScript]
function BeginInvoke(
   method : Delegate,
   args : Object[]
) : IAsyncResult;
```

Parameters

method
A Delegate to a method that takes parameters of the same number and type that are contained in *args*.

args
An array of type Object to pass as arguments to the given method. This can be a null reference (**Nothing** in Visual Basic) if no arguments are needed.

Return Value

An IAsyncResult interface that represents the asynchronous operation started by calling this method.

Remarks

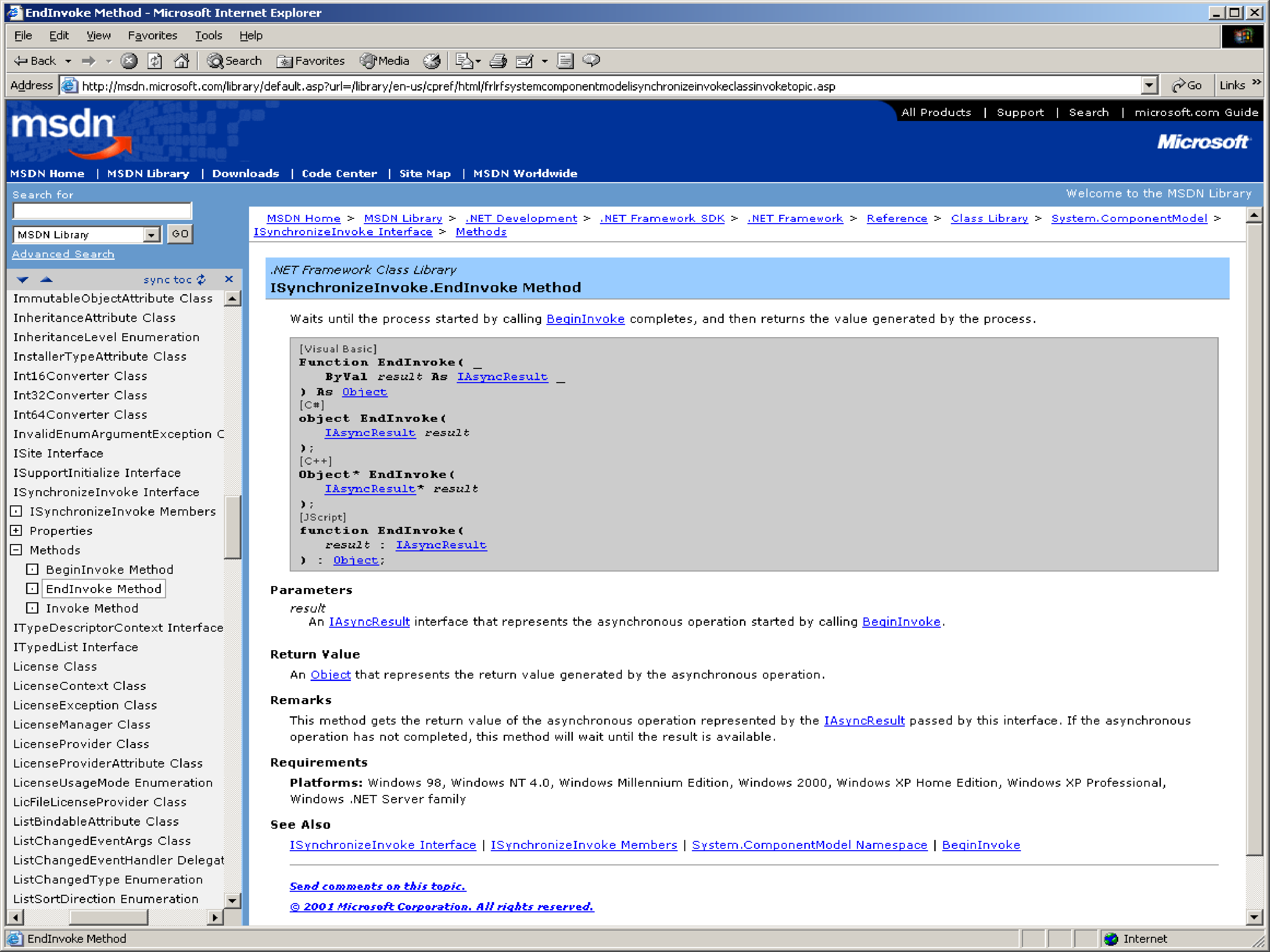
The delegate is called asynchronously, and this method returns immediately. You can call this method from any thread. If you need the return value from a process started with this method, call EndInvoke to get the value.

If you need to call the delegate synchronously, use the Invoke method instead.

Requirements

Platforms: Windows 98, Windows NT 4.0, Windows Millennium Edition, Windows 2000, Windows XP Home Edition, Windows XP Professional, Windows .NET Server family

BeginInvoke Method | Internet


EndInvoke Method - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Search Favorites Media
Address http://msdn.microsoft.com/library/default.asp?url=/library/en-us/cpref/html/frlrfsystemcomponentmodelisynchronizeinvokeclassinvoketopic.asp
Go Links
All Products | Support | Search | microsoft.com Guide
MSDN Home | MSDN Library | Downloads | Code Center | Site Map | MSDN Worldwide
Search for
MSDN Library GO
Advanced Search
Welcome to the MSDN Library
sync toc
ImmutableObjectAttribute Class
InheritanceAttribute Class
InheritanceLevel Enumeration
InstallerTypeAttribute Class
Int16Converter Class
Int32Converter Class
Int64Converter Class
InvalidEnumArgumentException C
ISite Interface
ISupportInitialize Interface
ISynchronizeInvoke Interface
ISynchronizeInvoke Members
Properties
Methods
BeginInvoke Method
EndInvoke Method
Invoke Method
ITypeDescriptorContext Interface
ITypedList Interface
License Class
LicenseContext Class
LicenseException Class
LicenseManager Class
LicenseProvider Class
LicenseProviderAttribute Class
LicenseUsageMode Enumeration
LicFileLicenseProvider Class
ListBindableAttribute Class
ListChangedEventArgs Class
ListChangedEventHandler Delegat
ListChangedType Enumeration
ListSortDirection Enumeration
EndInvoke Method
Internet

MSDN Home > MSDN Library > .NET Development > .NET Framework SDK > .NET Framework > Reference > Class Library > System.ComponentModel > ISynchronizeInvoke Interface > Methods

.NET Framework Class Library

ISynchronizeInvoke.EndInvoke Method

Waits until the process started by calling BeginInvoke completes, and then returns the value generated by the process.

```
[Visual Basic]
Function EndInvoke( _
   ByVal result As IAsyncResult _
) As Object
[C#]
object EndInvoke(
   IAsyncResult result
);
[C++]
Object* EndInvoke(
   IAsyncResult* result
);
[JScript]
function EndInvoke(
   result : IAsyncResult
) : Object;
```

Parameters

result
An IAsyncResult interface that represents the asynchronous operation started by calling BeginInvoke.

Return Value

An Object that represents the return value generated by the asynchronous operation.

Remarks

This method gets the return value of the asynchronous operation represented by the IAsyncResult passed by this interface. If the asynchronous operation has not completed, this method will wait until the result is available.

Requirements

Platforms: Windows 98, Windows NT 4.0, Windows Millennium Edition, Windows 2000, Windows XP Home Edition, Windows XP Professional, Windows .NET Server family

See Also

ISynchronizeInvoke Interface | ISynchronizeInvoke Members | System.ComponentModel Namespace | BeginInvoke

Send comments on this topic.

End of Presentation