

# 1 System.AppDomain Class

```
2 [ILAsm]  
3 .class public sealed AppDomain extends System.MarshalByRefObject  
4 [C#]  
5 public sealed class AppDomain: MarshalByRefObject
```

## 6 Assembly Info:

- 7 • *Name:* mscorlib
- 8 • *Public Key:* [00 00 00 00 00 00 00 00 04 00 00 00 00 00 00 00]
- 9 • *Version:* 2.0.x.x
- 10 • *Attributes:*
  - 11 ○ CLSCompliantAttribute(true)

## 12 Summary

13 Represents an application domain, which is an isolated environment where applications  
14 execute.

## 15 Inherits From: System.MarshalByRefObject

16  
17 **Library:** RuntimeInfrastructure

18  
19 **Thread Safety:** All public static members of this type are safe for multithreaded operations.  
20 No instance members are guaranteed to be thread safe.

## 22 Description

23 Application domains, which are represented by `System.AppDomain` objects, provide  
24 isolation, unloading, and security boundaries for executing managed code.

25  
26 Multiple application domains can run in a single process; however, there is not a one-to-  
27 one correlation between application domains and threads. Several threads can belong to  
28 a single application domain, and while a given thread is not confined to a single  
29 application domain, at any given time, a thread executes in a single application domain.

30  
31 Application domains are created using the `CreateDomain` method. `System.AppDomain`  
32 instances are used to load and execute assemblies (`System.Reflection.Assembly`).  
33 When a `System.AppDomain` is no longer in use, it can be unloaded.

34  
35 The `System.AppDomain` class implements a set of events to enable applications to  
36 respond to the following conditions:

Condition	Event
An assembly was loaded.	<code>System.AppDomain.AssemblyLoad</code>

An application domain will be unloaded.	<code>System.AppDomain.DomainUnload</code>
An unhandled exception was thrown.	<code>System.AppDomain.UnhandledException</code>

1

2

# 1 AppDomain.AssemblyLoad Event

```
2 [ILAsm]  
3 .event public event AssemblyLoad
```

```
4 [C#]  
5 public event AssemblyLoadEventHandler AssemblyLoad
```

## 6 Summary

7 Raised when an assembly is loaded.

## 8 Description

9 *[Note:* This event is handled by a `System.AssemblyLoadEventHandler` delegate.  
10 Information about the event is passed to the delegate in a  
11 `System.AssemblyLoadEventArgs` instance.

12  
13 For additional information about events, see Partitions I and II of the CLI Specification.

14 ]  
15

16

# 1 AppDomain.DomainUnload Event

```
2 [ILAsm]  
3 .event public event DomainUnload  
4 [C#]  
5 public event EventHandler DomainUnload
```

## 6 Summary

7 Raised when a `System.AppDomain` is about to be unloaded.

## 8 Description

9 *[Note:* This event is handled by a `System.EventHandler` delegate. Information about  
10 the event is passed to the delegate in a `System.EventArgs` instance. The delegate for  
11 this event can perform any termination activities before the application domain is  
12 unloaded.

13 For additional information about events, see Partitions I and II of the CLI Specification.

14 ]  
15 ]  
16 ]

17

# 1 AppDomain.UnhandledException Event

2 [ILAsm]  
3 .event public event UnhandledException

4 [C#]  
5 public event UnhandledExceptionHandler UnhandledException

## 6 Summary

7 Raised when an exception is not caught by the default application domain.

## 8 Description

9 [Note: This event is handled by a `System.UnhandledExceptionHandler` delegate.  
10 Information about the event is passed to the delegate in a  
11 `System.UnhandledExceptionEventArgs` instance. The delegate provides default  
12 handling for uncaught exceptions. When this event is not handled, the system default  
13 handler reports the exception to the user and might terminate the application. For  
14 additional information, see `System.UnhandledExceptionEventArgs.IsTerminating.`  
15  
16  
17

18 This event is raised only for the application domain that is created by the system when  
19 an application is started. If an application creates additional application domains,  
20 specifying a delegate for this event in those applications domains has no effect.  
21

22 [Note: For additional information about events, see Partitions I and II of the CLI  
23 Specification.]  
24  
25

26

# 1 AppDomain.CreateDomain(System.String)

## 2 Method

```
3 [ILAsm]  
4 .method public hidebysig static class System.AppDomain CreateDomain(string  
5 friendlyName)  
  
6 [C#]  
7 public static AppDomain CreateDomain(string friendlyName)
```

### 8 Summary

9 Creates and returns a new application domain with the specified name.

### 10 Parameters

Parameter	Description
<i>friendlyName</i>	A System.String containing the friendly name of the domain.

11

### 12 Return Value

13 A System.AppDomain representing the newly created application domain.

### 14 Description

15 [Note: The *friendlyName* parameter is intended to identify the domain in a manner that  
16 is meaningful to humans. This string should be suitable for display in user interfaces.]  
17  
18

### 19 Exceptions

Exception	Condition
<b>System.ArgumentNullException</b>	<i>friendlyName</i> is null.

20

21

# 1 AppDomain.ToString() Method

```
2 [ILAsm]  
3 .method public hidebysig virtual string ToString()  
4 [C#]  
5 public override string ToString()
```

## 6 Summary

7 Returns a `System.String` representation of the current instance.

## 8 Return Value

9 A `System.String` containing information about the current `System.AppDomain` instance.

## 10 Description

11 The string returned by this method includes the friendly name of the application domain.

12

13 [*Note:* This method overrides `System.Object.ToString`.]

14

15

16

# 1 AppDomain.Unload(System.AppDomain)

## 2 Method

```
3 [ILAsm]  
4 .method public hidebysig static void Unload(class System.AppDomain domain)  
5 [C#]  
6 public static void Unload(AppDomain domain)
```

### 7 Summary

8 Unloads the specified application domain.

### 9 Parameters

Parameter	Description
<i>domain</i>	A System.AppDomain representing the application domain to be unloaded.

10

### 11 Description

12 If the thread that invoked System.AppDomain.Unload is running in *domain*, another  
13 thread is started to perform the unload operation. If *domain* cannot be unloaded, a  
14 System.CannotUnloadAppDomainException is thrown in that thread, not the original  
15 thread that invoked System.AppDomain.Unload. However, if the thread that invoked  
16 System.AppDomain.Unload is running outside *domain*, that is the thread that receives  
17 the exception.

18

19 The threads in *domain* are terminated using the System.Threading.Thread.Abort  
20 method, which throws the thread an instance of  
21 System.Threading.ThreadAbortException. [Note: Although the thread should  
22 terminate promptly, it can continue executing for an unbounded amount of time in its  
23 finally clause.]

24

25

### 26 Exceptions

Exception	Condition
System.ArgumentNullException	<i>domain</i> is null.
System.CannotUnloadAppDomainException	<i>domain</i> could not be unloaded.

27

28

# 1 AppDomain.FriendlyName Property

```
2 [ILAsm]  
3 .property string FriendlyName { public final hidebysig virtual specialname  
4 string get_FriendlyName() }  
  
5 [C#]  
6 public string FriendlyName { get; }
```

## 7 Summary

8 Gets the friendly name of the current instance.

## 9 Property Value

10 A `System.String` containing the friendly name of the current application domain.

## 11 Description

12 This property is read-only.

13  
14 The friendly name of a `System.AppDomain` instance created by an application is specified  
15 to the constructor. The friendly name of the default `System.AppDomain` is the name of  
16 the assembly file loaded in the application domain. The friendly name is formed by  
17 stripping the directory information from the assembly's file name. For example, if the  
18 loaded assembly has the name "`\MyAppDirectory\MyAssembly.exe`", the friendly name  
19 of the default application domain is "`MyAssembly.exe`".

20