



# Programming the .NET Framework 4.5

## Module 06 - Application Domains

# In This Chapter

- ✦ AppDomains as isolation boundaries
  - ✦ Creating and unloading AppDomains
  - ✦ Executing code in an AppDomain
  - ✦ AppDomain boundaries
  - ✦ Overview of .NET Remoting
  - ✦ Lab
-

# System Isolation Boundaries

Network

The diagram illustrates the hierarchy of system isolation boundaries. At the top is a blue box labeled 'Network'. Below it are two red boxes: 'Machine A' on the left and 'Machine B' on the right. Under 'Machine A' is a green box labeled 'Process 1'. Under 'Machine B' are two green boxes: 'Process 2' on the left and 'Process 3' on the right. Below 'Process 2' and 'Process 3' are three purple boxes, each labeled 'Application Domain X', with the third box labeled 'Application Domain Z'.

Machine A

Machine B

Process 1

Process 2

Process 3

Application  
Domain X

Application  
Domain X

Application  
Domain Z

# Application Domain Isolation

## Shared

- Virtual memory
- Operating system handles
- Unmanaged resources

## Isolated

- Managed objects
  - Assemblies and types
  - Configuration information
-

# Why AppDomains?

- ✦ Reliability (fault isolation)
  - ✦ Dynamic code unloading
  - ✦ Security
  - ✦ Configuration
  - ✦ Versioning
  - ✦ Monitoring
-

# Properties of an AppDomain

- ✦ Friendly name, id
  - ✦ Base directory
  - ✦ Application configuration
  - ✦ Security policy
  - ✦ Shadow copy configuration
-

# Things to Look Out for

- ✦ Static data is domain-private
  - ✦ Not every object can cross domain boundaries
-

# Creating an AppDomain

## ✦ AppDomain.CreateDomain

```
1 AppDomain domain =  
2     AppDomain.CreateDomain("MyFirstDomain");  
3 Console.WriteLine("Base directory: " +  
4     domain.BaseDirectory);  
5 Console.WriteLine("Id: " + domain.Id);  
6 Console.WriteLine("Configuration file: " +  
7     domain.SetupInformation.ConfigurationFile);
```



# Retrieving Assemblies

## ✦ AppDomain.GetAssemblies

```
1 Array.ForEach(  
2     domain.GetAssemblies(), Console.WriteLine);  
3 //Output:  
4 //mscorlib, Version=2.0.0.0, Culture=neutral,  
5 //PublicKeyToken=b77a5c561934e089
```

# Unloading an AppDomain

## ✦ AppDomain.Unload

```
1 AppDomain.Unload(domain);  
2 domain.GetAssemblies();  
3 //Output:  
4 //Unhandled Exception:  
5 //System.AppDomainUnloadedException:  
6 //Attempted to access an unloaded AppDomain.  
7 // at System.AppDomain.GetAssemblies()
```

# Executing Code in an AppDomain

- ✦ `AppDomainInitializer`
  - ✦ `ExecuteAssembly`
  - ✦ `CreateInstanceAndUnwrap`
  - ✦ `DoCallback`
-

# Executing Code in an AppDomain

## Demo



# Crossing AppDomain Boundaries

Process

AppDomain 1

AppDomain  
2

Shapes.dll

Rectangle

Triangle

Circle

Rectangle  
A

Rectangle  
B

# Marshal-by-Value

Process

AppDomain 1

AppDomain  
2

Shapes.dll

Shapes.dll

Rectangle

Triangle

Circle

Rectangle

Rectangle A

Rectangle B

**Copy** of  
Rectangle B

# Marshal-by-Reference

Process

AppDomain 1

AppDomain 2

Shapes.dll

Shapes.dll

Rectangle

Triangle

Circle

Rectangle

Rectangle A

Rectangle B

**Proxy to  
Rectangle B**



MBV vs. MBR

# Demo





# AppDomain Monitoring

- ✦ Monitor CPU and memory usage of individual AppDomains
- ✦ Enable through configuration, code, or hosting

```
1 AppDomain.MonitoringIsEnabled = true;  
2 AppDomain domain = AppDomain.CreateDomain(...);  
3 if (domain.MonitoringTotalProcessorTime >  
4     TimeSpan.FromSeconds(5)) . . .
```

```
<configuration>  
  <runtime>  
    <appDomainResourceMonitoring enabled="true" />  
  </runtime>  
</configuration>
```

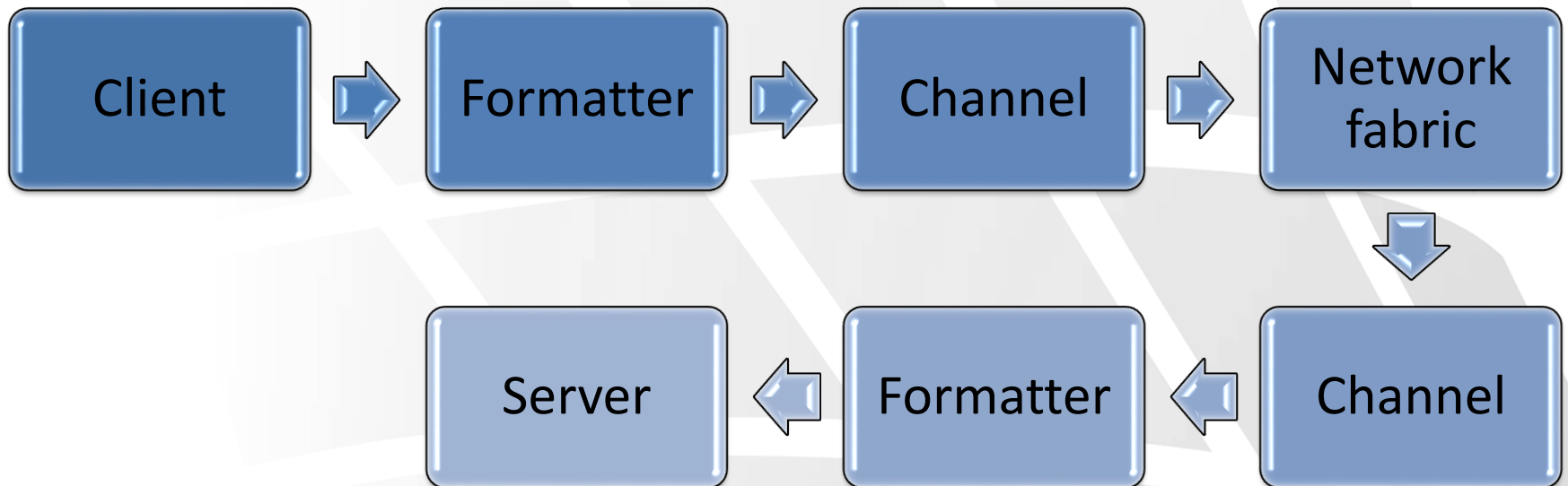
# AppDomain Monitoring

# Demo



# .NET Remoting Overview

- ⚡ Distributed computing infrastructure
- ⚡ The same framework used by MBR



Remote Exec

# Demo



# Plugin Framework

# Lab



# Summary

- ✦ AppDomains as isolation boundaries
  - ✦ Creating and unloading AppDomains
  - ✦ Executing code in an AppDomain
  - ✦ AppDomain boundaries
  - ✦ Overview of .NET Remoting
  - ✦ Lab
-

# Questions

