

# Turbo charge your applications with AppFabric caching



**DEVELOPMENT**MENTOR

DEVELOPING PEOPLE WHO DEVELOP SOFTWARE

# Objectives

- **Understand the reason we need caching**
- **Know the basic Windows Server AppFabric caching constructs**
- **How to use Windows Server AppFabric caching**

# Why do we need caching

- **Retrieving the same information often is wasteful**
  - But often required in a stateless environment like ASP.NET or WCF
  - Lots of extra identical database calls
- **Caching the result prevents this**
  - Better **performance**
  - Better **scalability**
- **Caching is always a **tradeoff****
  - IO versus memory
  - Performance versus complexity



# Data Classification

- **Reference data**
  - Used in a **read-only** fashion
  - Example: Articles on sale
  - Changes are infrequent
  - Good for caching
- **Activity data**
  - Used in a read-write fashion by a **single client**
  - Example: Shopping basket
  - Good for caching
- **Resource data**
  - Used in a read-write fashion by many clients
    - Example: Article quantity in stock
  - Harder to cache
  - Use the concurrency patterns



# Caching in ASP.NET

- **System.Web.Caching.Cache**
  - Been around since .NET 1.1
  - `HttpContext.Current.Cache[cacheKey]`
- **ASP.NET Output Caching**
  - Caches the HTTP response
  - `<%@ OutputCache Duration="60" VaryByParam="None"%>`
- **ASP.NET Session state**
  - `Session[sessionKey]`
  - Can be out of process
- **Caching occurs per server**
  - Inefficient with server farms
  - Caches can get out of sync between servers



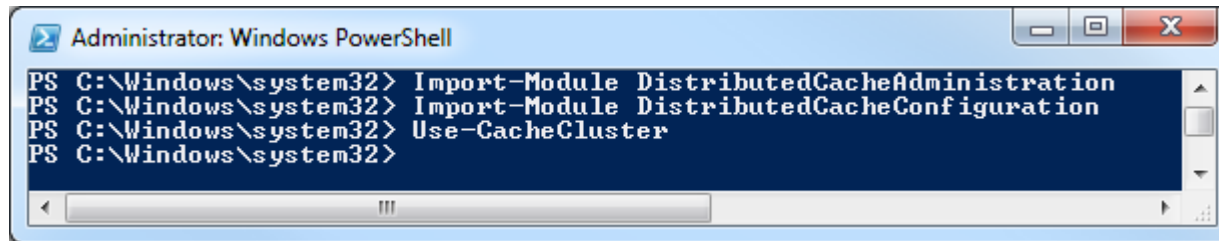
# AppFabric Caching

- **Distributed caching**
  - Stores state on 1 or more machines
  - Managed as 1 cache cluster
- **Supports ASP.NET caching**
  - Session state provider
- **High availability**
  - Store data on multiple servers
- **Local Cache**
  - Faster local access
  - Syncs with the cluster
- **Security model**
  - Only known users can access the cache cluster



# PowerShell

- **All management is done through PowerShell**
  - Needs to be an **elevated** PowerShell Window
- **Two modules**
  - DistributedCacheAdministration
  - DistributedCacheConfiguration
- **Open the cluster to work with it**
  - Use-CacheCluster
- **From the Start Menu**
  - Caching Administration Windows PowerShell



```
Administrator: Windows PowerShell
PS C:\Windows\system32> Import-Module DistributedCacheAdministration
PS C:\Windows\system32> Import-Module DistributedCacheConfiguration
PS C:\Windows\system32> Use-CacheCluster
PS C:\Windows\system32>
```

# PowerShell Commands

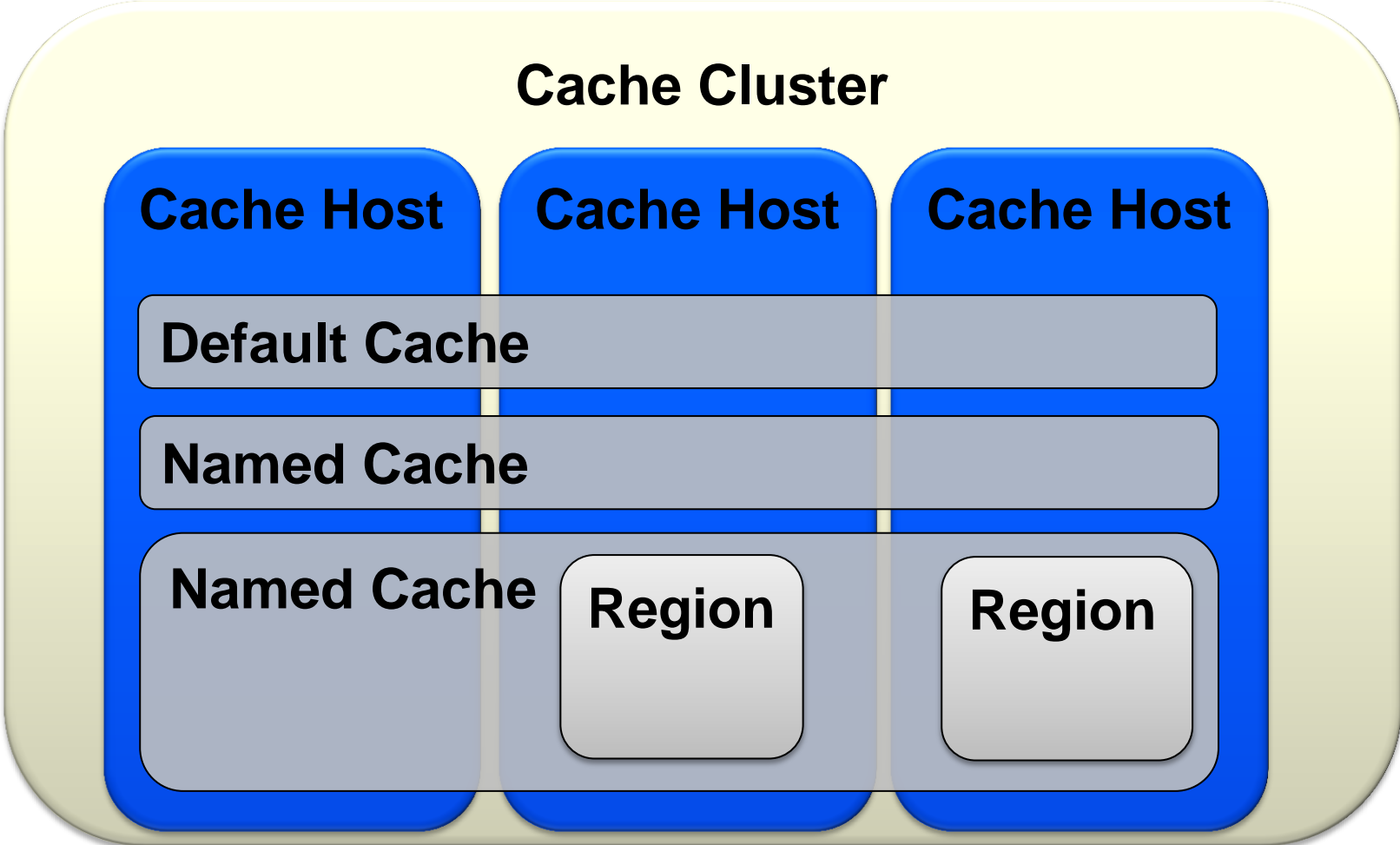
```
Administrator: powershell.exe -noexit -command "Import-Module DistributedCacheAdministratio...
PS C:\Windows\system32> Get-Command -module DistributedCacheAdministration

CommandType      Name                                     Definition
-----
Cmdlet            Clear-CacheLogging                     Clear-CacheLogging [-Verbose...
Cmdlet            Export-CacheClusterConfig              Export-CacheClusterConfig [-...
Cmdlet            Get-Cache                               Get-Cache [[-HostName] <Stri...
Cmdlet            Get-CacheAllowedClientAccounts         Get-CacheAllowedClientAccoun...
Cmdlet            Get-CacheClusterHealth                 Get-CacheClusterHealth [-Ver...
Cmdlet            Get-CacheConfig                         Get-CacheConfig [-CacheName]...
Cmdlet            Get-CacheHost                           Get-CacheHost [-Verbose] [-D...
Cmdlet            Get-CacheHostConfig                    Get-CacheHostConfig [-HostNa...
Cmdlet            Get-CacheRegion                         Get-CacheRegion [[-CacheName...
Cmdlet            Get-CacheStatistics                    Get-CacheStatistics [-CacheN...
Cmdlet            Grant-CacheAllowedClientAccount        Grant-CacheAllowedClientAcco...
Cmdlet            Import-CacheClusterConfig              Import-CacheClusterConfig [-...
Cmdlet            Invoke-CacheGC                          Invoke-CacheGC [-Verbose] [-...
Cmdlet            New-Cache                               New-Cache [-CacheName] <Stri...
Cmdlet            Remove-Cache                            Remove-Cache [-CacheName] <S...
Cmdlet            Restart-CacheCluster                    Restart-CacheCluster [-HostI...
Cmdlet            Restart-CacheHost                       Restart-CacheHost [-HostName...
Cmdlet            Revoke-CacheAllowedClientAcc...        Revoke-CacheAllowedClientAcc...
Cmdlet            Set-CacheClusterSecurity               Set-CacheClusterSecurity [-S...
Cmdlet            Set-CacheConfig                         Set-CacheConfig [-CacheName]...
Cmdlet            Set-CacheHostConfig                    Set-CacheHostConfig [-HostNa...
Cmdlet            Set-CacheLogging                       Set-CacheLogging [-LogLevel]...
Cmdlet            Start-CacheCluster                      Start-CacheCluster [-HostTim...
Cmdlet            Start-CacheHost                         Start-CacheHost [-HostName] ...
Cmdlet            Stop-CacheCluster                       Stop-CacheCluster [-HostTime...
Cmdlet            Stop-CacheHost                          Stop-CacheHost [-HostName] <...
Cmdlet            Test-CacheConfigAvailability            Test-CacheConfigAvailability...
Cmdlet            Use-CacheCluster                        Use-CacheCluster [[-Provider...

PS C:\Windows\system32>
```



# Logical overview



# Cache Cluster

- **A collection of 1 or more cache hosts**
  - Usually more than 1 for reliability
- **Cache managed through the cache cluster**
  - Uses PowerShell commandlets
- **Uses a central configuration**
  - XML file on a network share
  - SQL Server
  - Custom



# Cache Host

- **Runs as a Windows service**
  - Dedicated machines are recommended
  - Requires .NET 4.0
- **Configuring the first host creates the Cluster**
  - Subsequent hosts join the cluster

# Configuring the Cache Cluster

The screenshot shows the 'Configure Caching Service' step of the Windows Server AppFabric Configuration Wizard. The left sidebar contains a navigation pane with the following items: 'Before You Begin', 'Hosting Services', 'Caching Service' (highlighted in blue), 'Cache Node', and 'Application'. The main content area has a title bar with a folder icon and the text 'Configure Caching Service'. Below the title bar, there is a description: 'This page lets you set the system-level configuration for the Caching Service.' A checkbox labeled 'Set Caching Service configuration' is checked. Below this, a green checkmark icon indicates 'This machine is a member of an AppFabric Caching cluster.' The 'Caching Service account' is set to 'Maurice-PC\Maurice' with a 'Change...' button. The 'Caching Service configuration provider' is set to 'XML' with a 'Configure...' button. A link for 'How to install additional providers' is provided. The 'File share (UNC server share required: \\server\share):' is set to '\\MAURICE-PC' with a 'Browse...' button. An information icon and text state: 'The High Availability feature of Windows Server AppFabric caching features requires all nodes in the cache cluster to be running Windows Server Enterprise Edition or higher. Please confirm that all High Availability cache nodes are running on a supported operating system.' A link for 'More information about High Availability' is provided. At the bottom, there are two radio buttons: 'New cluster' (selected) and 'Join cluster'. Below them, the 'Cluster size' is set to 'Small (1-5 Machines)'. A link for 'More about cluster sizes' is provided. At the bottom of the wizard, there are four buttons: 'Help', '< Previous', 'Next >', and 'Cancel'.

Windows Server AppFabric Configuration Wizard

## Configure Caching Service

Before You Begin

Hosting Services

**Caching Service**

Cache Node

Application

This page lets you set the system-level configuration for the Caching Service.

Set Caching Service configuration

This machine is a member of an AppFabric Caching cluster.

Caching Service account:  
Maurice-PC\Maurice Change...

Caching Service configuration provider:  
XML Configure...

[How to install additional providers](#)

File share (UNC server share required: \\server\share):  
\\MAURICE-PC Browse...

**i** The High Availability feature of Windows Server AppFabric caching features requires all nodes in the cache cluster to be running Windows Server Enterprise Edition or higher. Please confirm that all High Availability cache nodes are running on a supported operating system.  
[More information about High Availability](#)

New cluster  Join cluster

Cluster size: Small (1-5 Machines)

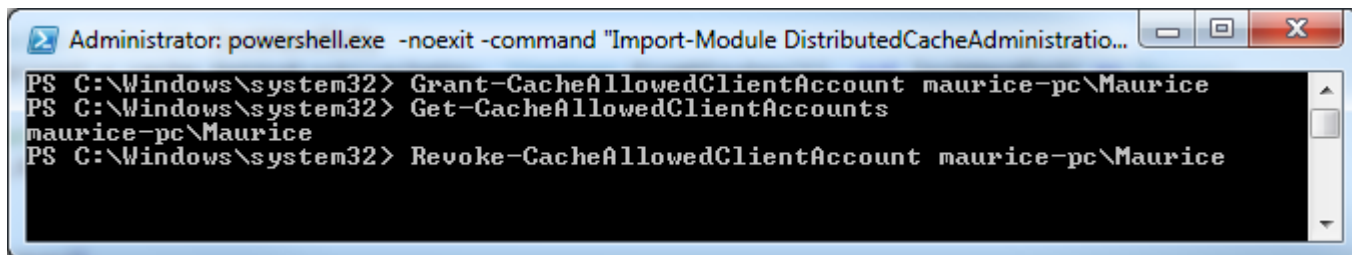
[More about cluster sizes](#)

Help < Previous Next > Cancel



# Security

- **The cache cluster is secured by default**
  - Can be **disabled** if needed
- **Security is only at a cluster level**
  - Can't protect named caches or regions
- **Users managed through PowerShell script**
  - Grant-CacheAllowedClientAccount
  - Get-CacheAllowedClientAccounts
  - Revoke-CacheAllowedClientAccount

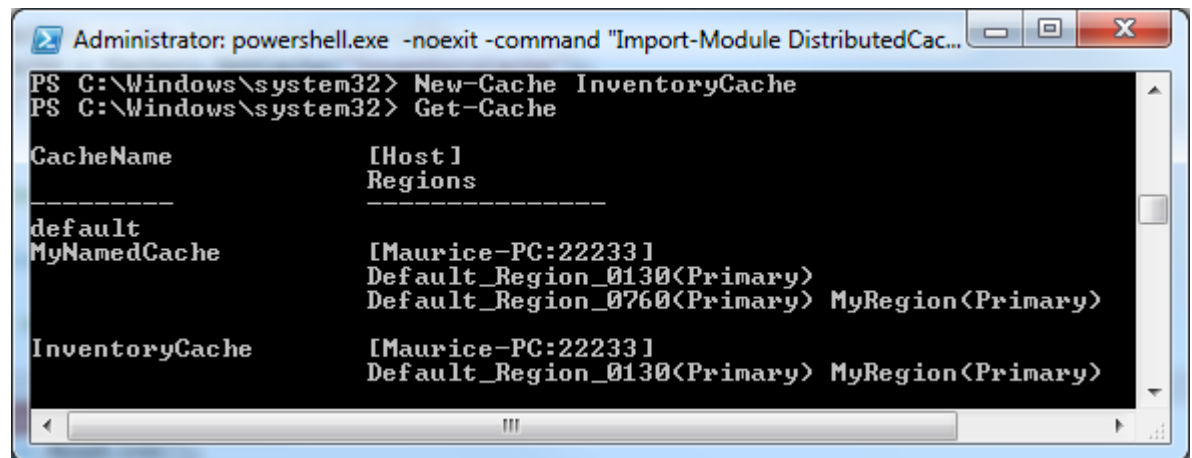


```
Administrator: powershell.exe -noexit -command "Import-Module DistributedCacheAdministratio...  
PS C:\Windows\system32> Grant-CacheAllowedClientAccount maurice-pc\Maurice  
PS C:\Windows\system32> Get-CacheAllowedClientAccounts  
maurice-pc\Maurice  
PS C:\Windows\system32> Revoke-CacheAllowedClientAccount maurice-pc\Maurice
```



# Caches

- **Default Cache**
  - Special cache with the name “default”
  - Always there
- **Named Cache**
  - Need to be created by an administrator
  - Maximum of 128 named caches
- **Default storage behavior configured per cache**
  - Expiration
  - Eviction
  - High reliability



```
Administrator: powershell.exe -noexit -command "Import-Module DistributedCac...
PS C:\Windows\system32> New-Cache InventoryCache
PS C:\Windows\system32> Get-Cache

CacheName           [Host]
-----
Regions
-----
default
MyNamedCache        [Maurice-PC:22233 ]
                    Default_Region_0130<Primary>
                    Default_Region_0760<Primary> MyRegion<Primary>

InventoryCache      [Maurice-PC:22233 ]
                    Default_Region_0130<Primary> MyRegion<Primary>
```



# Regions

- **Always tied to a single machine**
  - **Limits** the AppFabric in scalability
- **Allows you to search for cached items**
  - Items can have one or more tags as metadata
- **Create regions in code as you need them**



# Cache Clients

- **Uses a cache aside pattern**
  - Using the cache is an **explicit** action
- **Uses WCF to communicate with the cache hosts**
  - Through the NetTcpBinding
- **Configured programmatically or declaratively**
  - **Declaratively** is recommended
- **Requires .NET 3.5 SP1 or later**



# Programmatically configuring a DataCacheFactory

- **Use the DataCacheFactoryConfiguration**
- **Add one or more DataCacheServerEndpoint's**
  - Best to add **more than one**
- **Used to connect to the cache cluster**
  - Not every host in the cluster is required
  - Other hosts will be loaded through the cache cluster

```
var config = new DataCacheFactoryConfiguration();

var servers = new List<DataCacheServerEndpoint>();
servers.Add(new DataCacheServerEndpoint("CacheServer1", 22233));
servers.Add(new DataCacheServerEndpoint("CacheServer2", 22233));
config.Servers = servers;

var factory = new DataCacheFactory(config);
var cache = factory.GetCache("InventoryCache");
```



# Declaratively configuring a DataCacheFactory

```
<configuration>
  <configSections>
    <!-- required to read the <dataCacheClient> element -->
    <section name="dataCacheClient"
      type="Microsoft.ApplicationServer.Caching.DataCacheClientSection,
      Microsoft.ApplicationServer.Caching.Core, Version=1.0.0.0,
      Culture=neutral, PublicKeyToken=31bf3856ad364e35"
      allowLocation="true"
      allowDefinition="Everywhere"/>
  </configSections>

  <dataCacheClient>
    <hosts>
      <host name="CacheServer1" cachePort="22233"/>
      <host name="CacheServer2" cachePort="22233"/>
    </hosts>
  </dataCacheClient>
</configuration>
```



# ASP.NET Session Caching

```
<system.web>
  <sessionState mode="Custom"
    customProvider="AppFabricCacheSessionStoreProvider">
    <providers>
      <add
        name="AppFabricCacheSessionStoreProvider"
        type="Microsoft.ApplicationServer.Caching.DataCacheSessionStoreProvider,
          Microsoft.ApplicationServer.Caching.Client,
          Version=1.0.0.0, Culture=neutral,
          PublicKeyToken=31bf3856ad364e35"
        cacheName="MyCacheName"
        sharedId="MyApplicationName"
      />
    </providers>
  </sessionState>
</system.web>
```



# Opening a DataCache

- **The DataCache is the [main object](#) to work with**
  - Add/Get/Remove objects
  - Create/Remove regions
  - Add/Remove notification callbacks
  - Lock/Unlock objects
- **Add reference to:**
  - Microsoft.ApplicationServer.Caching.Core
  - Microsoft.ApplicationServer.Caching.Client
- **Created through the [DataCacheFactory](#)**
  - Best used as a [singleton](#)
  - Use GetDefaultCache() or GetCache() to load DataCache

```
var factory = new DataCacheFactory();  
var cache = factory.GetDefaultCache();
```

# Client Cache API

- **Adding data to the cache**
  - Add() or Put()
  - Add() throws an **exception** if the key already exists
- **Loading data from the cache**
  - Get()
  - GetCacheItem()
- **Updating data in the cache**
  - Put()
- **Searching for items**
  - GetObjectsByTag()
    - Only with items in regions
  - GetObjectsInRegion()
    - Also works with default region names
- **Item keys are scoped to a cache or region**



# The Cache Aside Pattern

```
var cacheKey = "MyKey";
var person = cache.Get(cacheKey) as Person;
if (person == null)
{
    person = new Person()
    {
        FirstName = "Maurice",
        LastName = "de Beijer"
    };
    cache.Add(cacheKey, person);
}
// Use person object
Console.WriteLine("{0}, {1}", person.LastName, person.FirstName);
```



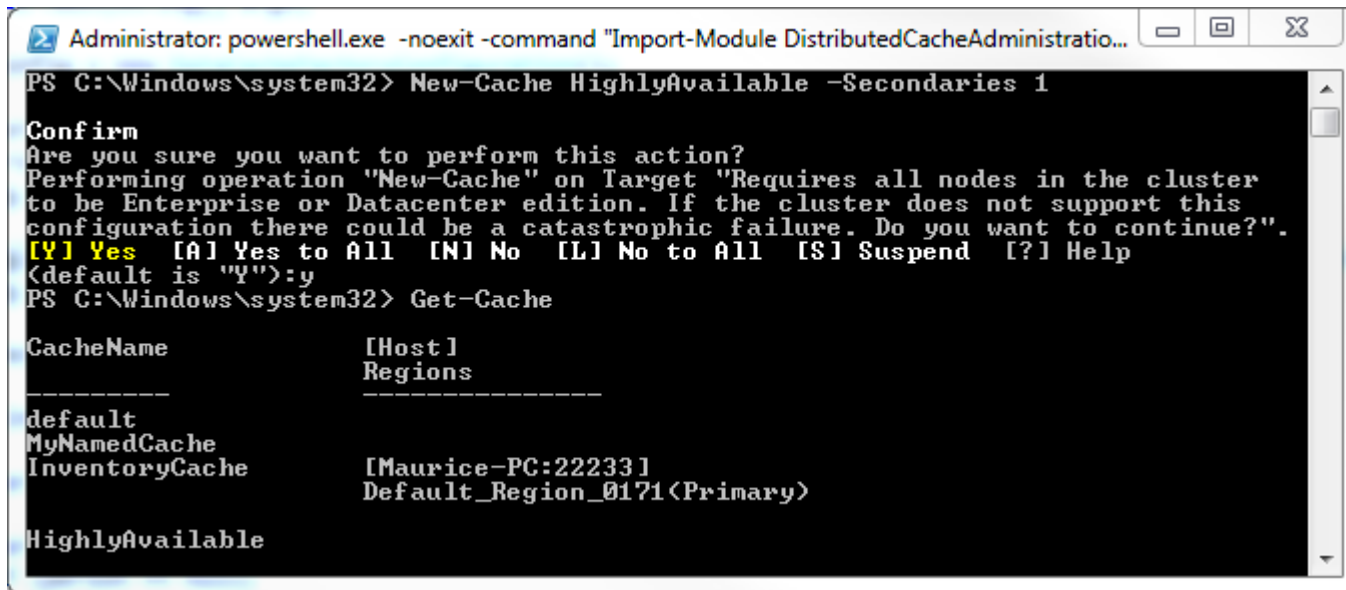
# Local Cache

- **Improves performance**
  - Keeps a local object reference
  - No **deserialization** overhead
  - No **network** transport
- **Can get **out of sync** with the distributed cache**
  - Don't set the time to live to high (in seconds)
  - Can invalidate using notification

```
<dataCacheClient>  
  <localCache isEnabled="true"  
    sync="TimeoutBased"  
    objectCount="100000"  
    ttlValue="300" />  
</dataCacheClient>
```

# High Availability

- **All data is stored on two cache hosts**
  - Use at least three cache hosts for reliability
  - With a **single host** in the cluster it is considered **down**.
- **Use New-Cache with –Secondaries**
  - Value must be 1 for high availability
- **Requires Windows Server Enterprise Edition**



```
Administrator: powershell.exe -noexit -command "Import-Module DistributedCacheAdministratio...
PS C:\Windows\system32> New-Cache HighlyAvailable -Secondaries 1

Confirm
Are you sure you want to perform this action?
Performing operation "New-Cache" on Target "Requires all nodes in the cluster
to be Enterprise or Datacenter edition. If the cluster does not support this
configuration there could be a catastrophic failure. Do you want to continue?".
[Y] Yes [A] Yes to All [N] No [L] No to All [S] Suspend [?] Help
(default is "Y"):y
PS C:\Windows\system32> Get-Cache

CacheName           [Host]
-----
Regions
-----
default
MyNamedCache
InventoryCache      [Maurice-PC:22233]
                    Default_Region_0171 <Primary>

HighlyAvailable
```

# Summary

- **Understand the reason we need caching**
  - Improve an application scalability
- **Basic Windows Server AppFabric caching constructs**
  - Cache Cluster
  - Cache Hosts
  - Named Caches
- **How to use Windows Server AppFabric caching**
  - DataCacheFactory
  - DataCache
  - High Availability
  - Local Cache

