

# Using Static Classes and Methods

Not all .NET Framework classes can be created by using `New-Object`. For example, if you try to create a **System.Environment** or a **System.Math** object with `New-Object`, you will get the following error messages:

PowerShellCopy

`New-Object System.Environment`

PowerShellCopy

`New-Object System.Math`

OutputCopy

183.82.125.202:4499 - Remote Desktop Connection

```
PowerShell 7 (x64)
PS C:\Users\devops34\Desktop> $xl = New-Object -ComObject Excel.Application -Strict
PS C:\Users\devops34\Desktop> New-Object System.Environment
New-Object: A constructor was not found. Cannot find an appropriate constructor for type System.Environment.
PS C:\Users\devops34\Desktop> [System.Environment]

IsPublic IsSerial Name                                     BaseType
-----
True     False     Environment                                     System.Object

PS C:\Users\devops34\Desktop> New-Object System.Environment
New-Object: A constructor was not found. Cannot find an appropriate constructor for type System.Environment.
PS C:\Users\devops34\Desktop> New-Object System.Math
New-Object: A constructor was not found. Cannot find an appropriate constructor for type System.Math.
PS C:\Users\devops34\Desktop> _
```

These errors occur because there is no way to create a new object from these classes. These classes are reference libraries of methods and properties that do not change state. You don't need to create them, you simply use them. Classes and methods such as these are called **static classes** because they are not created, destroyed, or changed. To make this clear we will provide examples that use static classes.

## Getting Environment Data with System.Environment

Usually, the first step in working with an object in Windows PowerShell is to use `Get-Member` to find out what members it contains. With static classes, the process is a little different because the actual class is not an object.

### Referring to the Static System.Environment Class

You can refer to a static class by surrounding the class name with square brackets. For example, you can refer to **System.Environment** by typing the name within brackets. Doing so displays some generic type information:

PowerShellCopy

`[System.Environment]`

As we mentioned previously, Windows PowerShell automatically prepends **'System.'** to type names when you use `New-Object`. The same thing happens when using a bracketed type name, so you can specify `[System.Environment]` as `[Environment]`.

The **System.Environment** class contains general information about the working environment for the current process, which is `powershell.exe` when working within Windows PowerShell.

If you try to view details of this class by typing `[System.Environment] | Get-Member`, the object type is reported as being **System.RuntimeType**, not **System.Environment**:

PowerShellCopy

`[System.Environment] | Get-Member`

Output :

```
183.82.125.202-4499 - Remote Desktop Connection
PowerShell 7 (x64)
New-Object: A constructor was not found. Cannot find an appropriate constructor for type System.Math.
PS C:\Users\devops34\Desktop> [System.Environment]

IsPublic IsSerial Name                                     BaseType
-----
True     False     Environment                                             System.Object

PS C:\Users\devops34\Desktop> [System.Environment] | Get-Member

TypeName: System.RuntimeType

Name                                     MemberType Definition
-----
AsType                                  Method     type AsType()
Clone                                  Method     System.Object Clone(), System.Object ICloneable.Clone()
Equals                                  Method     bool Equals(System.Object obj), bool Equals(type o)
FindInterfaces                          Method     type[] FindInterfaces(System.Reflection.TypeFilter filter, System.Object filterCriteria)
FindMembers                             Method     System.Reflection.MemberInfo[] FindMembers(System.Reflection.MemberTypes memberType, System.Reflection.BindingFlags b...
GetArrayRank                           Method     int GetArrayRank()
GetConstructor                          Method     System.Reflection.ConstructorInfo GetConstructor(type[] types), System.Reflection.ConstructorInfo GetConstructor(Syst...
GetConstructors                         Method     System.Reflection.ConstructorInfo[] GetConstructors(System.Reflection.BindingFlags bindingAttr, System.Reflection.Co...
GetCustomAttributes                     Method     System.Object[] GetCustomAttributes(bool inherit, System.Object[] GetCustomAttributes(type attributeType, bool inher...
GetCustomAttributesData                 Method     System.Collections.Generic.IList[System.Reflection.CustomAttributeData] GetCustomAttributesData()
GetDeclaredEvent                        Method     System.Reflection.EventInfo GetDeclaredEvent(string name)
GetDeclaredField                        Method     System.Reflection.FieldInfo GetDeclaredField(string name)
GetDeclaredMethod                       Method     System.Reflection.MethodInfo GetDeclaredMethod(string name)
GetDeclaredMethods                      Method     System.Collections.Generic.IEnumerable[System.Reflection.MethodInfo] GetDeclaredMethods(string name)
GetDeclaredNestedType                   Method     System.Reflection.TypeInfo GetDeclaredNestedType(string name)
GetDeclaredProperty                     Method     System.Reflection.PropertyInfo GetDeclaredProperty(string name)
GetDefaultMembers                       Method     System.Reflection.MemberInfo[] GetDefaultMembers()
GetElementType                          Method     type GetElementType()
GetEnumName                             Method     string GetEnumName(System.Object value)
GetEnumNames                            Method     string[] GetEnumNames()
GetEnumUnderlyingType                   Method     type GetEnumUnderlyingType()
GetEnumValues                           Method     array GetEnumValues()
GetEvent                                Method     System.Reflection.EventInfo GetEvent(string name, System.Reflection.BindingFlags bindingAttr, System.Reflection.Even...
GetEvents                               Method     System.Reflection.EventInfo[] GetEvents(System.Reflection.BindingFlags bindingAttr, System.Reflection.EventInfo[] Ge...
GetField                                Method     System.Reflection.FieldInfo GetField(string name, System.Reflection.BindingFlags bindingAttr, System.Reflection.Fiel...
GetFields                               Method     System.Reflection.FieldInfo[] GetFields(System.Reflection.BindingFlags bindingAttr, System.Reflection.FieldInfo[] Ge...
GetGenericArguments                     Method     type[] GetGenericArguments()
GetGenericParameterConstraints           Method     type[] GetGenericParameterConstraints()
```

PowerShellCopy

`[System.Environment] | Get-Member`

To view static members with `Get-Member`, specify the **Static** parameter:

PowerShellCopy

`[System.Environment] | Get-Member -Static`

OutputCopy

```
183.82.125.202:4499 - Remote Desktop Connection
PowerShell 7 (x64)
UnderlyingSystemType      Property      type UnderlyingSystemType {get;}

PS C:\Users\devops34\Desktop> [System.Environment] | Get-Member -Static

TypeName: System.Environment

Name      MemberType Definition
-----
Equals    Method      static bool Equals(System.Object objA, System.Object objB)
Exit      Method      static void Exit(int exitCode)
ExpandEnvironmentVariables Method      static string ExpandEnvironmentVariables(string name)
FailFast  Method      static void FailFast(string message), static void FailFast(string message, System.Exception exception), static void FailFast(str
GetCommandLineArgs Method      static string[] GetCommandLineArgs()
GetEnvironmentVariable Method      static string GetEnvironmentVariable(string variable), static string GetEnvironmentVariable(string variable, System.EnvironmentV
GetEnvironmentVariables Method      static System.Collections.IDictionary GetEnvironmentVariables(System.EnvironmentVariableTarget target), static System.Collectio
GetFolderPath Method      static string GetFolderPath(System.Environment+SpecialFolder folder), static string GetFolderPath(System.Environment+SpecialFold
GetLogicalDrives Method      static string[] GetLogicalDrives()
ReferenceEquals Method      static bool ReferenceEquals(System.Object objA, System.Object objB)
SetEnvironmentVariable Method      static void SetEnvironmentVariable(string variable, string value), static void SetEnvironmentVariable(string variable, string va
CommandLine Property      static string CommandLine {get;}
CurrentDirectory Property      static string CurrentDirectory {get;set;}
CurrentManagedThreadId Property      static int CurrentManagedThreadId {get;}
ExitCode  Property      static int ExitCode {get;set;}
HasShutdownStarted Property      static bool HasShutdownStarted {get;}
Is64BitOperatingSystem Property      static bool Is64BitOperatingSystem {get;}
Is64BitProcess Property      static bool Is64BitProcess {get;}
MachineName Property      static string MachineName {get;}
NewLine   Property      static string NewLine {get;}
OSVersion Property      static System.OperatingSystem.OSVersion {get;}
ProcessId Property      static int ProcessId {get;}
ProcessorCount Property      static int ProcessorCount {get;}
ProcessPath Property      static string ProcessPath {get;}
StackTrace Property      static string StackTrace {get;}
SystemDirectory Property      static string SystemDirectory {get;}
SystemPageSize Property      static int SystemPageSize {get;}
TickCount Property      static int TickCount {get;}
TickCount64 Property      static long TickCount64 {get;}
UserDomainName Property      static string UserDomainName {get;}
UserInteractive Property      static bool UserInteractive {get;}
UserName  Property      static string UserName {get;}
```

## Displaying Static Properties of System.Environment

The properties of System.Environment are also static, and must be specified in a different way than normal properties. We use :: to indicate to Windows PowerShell that we want to work with a static method or property. To see the command that was used to launch Windows PowerShell, we check the **CommandLine** property by typing:

```
PowerShellCopy
[System.Environment]::CommandLine
```

To check the operating system version, display the OSVersion property by typing:

```
PowerShellCopy
[System.Environment]::OSVersion
```

We can check whether the computer is in the process of shutting down by displaying the **HasShutdownStarted** property:

```
PowerShellCopy
[System.Environment]::HasShutdownStarted
OutputCopy:
```

```
183.82.125.202:4499 - Remote Desktop Connection
PowerShell 7 (x64)

PS C:\Users\devops34\Desktop> [System.Environment]::CommandLine
"C:\Program Files\PowerShell\7\pwsh.dll" -NoExit -RemoveWorkingDirectoryTrailingCharacter -WorkingDirectory C:\Users\devops34\Desktop! -Command "$host.UI.RawUI.WindowTitle = 'PowerShell 7 (x64)'"
PS C:\Users\devops34\Desktop> [System.Environment]::OSVersion

Platform ServicePack Version      VersionString
-----
Win32NT      10.0.17763.0 Microsoft Windows NT 10.0.17763.0

PS C:\Users\devops34\Desktop> [System.Environment]::HasShutdownStarted
False
PS C:\Users\devops34\Desktop> s_
```

