# **ResourceManager Class**

## **Definition**

Namespace:

[System.Resources](https://docs.microsoft.com/en-us/dotnet/api/system.resources?view=netframework-4.8)

Assemblies:

System.Resources.ResourceManager.dll, mscorlib.dll, netstandard.dll

Represents a resource manager that provides convenient access to culture-specific resources at run time.

C#Copy

[System.Runtime.InteropServices.ComVisible(true)]

[System.Serializable]public class ResourceManager

Inheritance

[Object](https://docs.microsoft.com/en-us/dotnet/api/system.object?view=netframework-4.8)

ResourceManager

Derived

[System.ComponentModel.ComponentResourceManager](https://docs.microsoft.com/en-us/dotnet/api/system.componentmodel.componentresourcemanager?view=netframework-4.8)

Attributes

[ComVisibleAttribute](https://docs.microsoft.com/en-us/dotnet/api/system.runtime.interopservices.comvisibleattribute?view=netframework-4.8" \o "System.Runtime.InteropServices.ComVisibleAttribute)  [SerializableAttribute](https://docs.microsoft.com/en-us/dotnet/api/system.serializableattribute?view=netframework-4.8" \o "System.SerializableAttribute)

## **Examples**

The following example demonstrates how to use an explicit culture and the implicit current UI culture to obtain string resources from a main assembly and a satellite assembly. For more information, see the "Directory Locations for Satellite Assemblies Not Installed in the Global Assembly Cache" section of the [Creating Satellite Assemblies](https://docs.microsoft.com/en-us/dotnet/framework/resources/creating-satellite-assemblies-for-desktop-apps?view=netframework-4.8) topic.

To run this example:

In the app directory, create a file named rmc.txt that contains the following resource strings:

Copy

day=Friday

year=2006

holiday="Cinco de Mayo"

Use the [Resource File Generator](https://docs.microsoft.com/en-us/dotnet/framework/tools/resgen-exe-resource-file-generator?view=netframework-4.8) to generate the rmc.resources resource file from the rmc.txt input file as follows:

Copy

resgen rmc.txt

Create a subdirectory of the app directory and name it "es-MX". This is the culture name of the satellite assembly that you will create in the next three steps.

Create a file named rmc.es-MX.txt in the es-MX directory that contains the following resource strings:

Copy

day=Viernes

year=2006

holiday="Cinco de Mayo"

Use the [Resource File Generator](https://docs.microsoft.com/en-us/dotnet/framework/tools/resgen-exe-resource-file-generator?view=netframework-4.8) to generate the rmc.es-MX.resources resource file from the rmc.es-MX.txt input file as follows:

Copy

resgen rmc.es-MX.txt

Assume that the filename for this example is rmc.vb or rmc.cs. Copy the following source code into a file. Then compile it and embed the main assembly resource file, rmc.resources, in the executable assembly. If you are using the Visual Basic compiler, the syntax is:

Copy

vbc rmc.vb /resource:rmc.resources

The corresponding syntax for the C# compiler is:

Copy

csc /resource:rmc.resources rmc.cs

Use the [Assembly Linker](https://docs.microsoft.com/en-us/dotnet/framework/tools/al-exe-assembly-linker?view=netframework-4.8) to create a satellite assembly. If the base name of the app is rmc, the satellite assembly name must be rmc.resources.dll. The satellite assembly should be created in the es-MX directory. If es-MX is the current directory, use this command:

Copy

al /embed:rmc.es-MX.resources /c:es-MX /out:rmc.resources.dll

Run rmc.exe to obtain and display the embedded resource strings.

C#Copy

using System;using System.Resources;using System.Reflection;using System.Threading;using System.Globalization;

class Example

{

public static void Main()

{

string day;

string year;

string holiday;

string celebrate = "{0} will occur on {1} in {2}.\n";

// Create a resource manager.

ResourceManager rm = new ResourceManager("rmc",

typeof(Example).Assembly);

Console.WriteLine("Obtain resources using the current UI culture.");

// Get the resource strings for the day, year, and holiday

// using the current UI culture.

day = rm.GetString("day");

year = rm.GetString("year");

holiday = rm.GetString("holiday");

Console.WriteLine(celebrate, holiday, day, year);

// Obtain the es-MX culture.

CultureInfo ci = new CultureInfo("es-MX");

Console.WriteLine("Obtain resources using the es-MX culture.");

// Get the resource strings for the day, year, and holiday

// using the specified culture.

day = rm.GetString("day", ci);

year = rm.GetString("year", ci);

holiday = rm.GetString("holiday", ci);// ---------------------------------------------------------------// Alternatively, comment the preceding 3 code statements and // uncomment the following 4 code statements:// ----------------------------------------------------------------// Set the current UI culture to "es-MX" (Spanish-Mexico).// Thread.CurrentThread.CurrentUICulture = ci;

// Get the resource strings for the day, year, and holiday // using the current UI culture. Use those strings to // display a message. // day = rm.GetString("day");// year = rm.GetString("year");// holiday = rm.GetString("holiday");// ---------------------------------------------------------------

// Regardless of the alternative that you choose, display a message // using the retrieved resource strings.

Console.WriteLine(celebrate, holiday, day, year);

}

}/\*

This example displays the following output:

Obtain resources using the current UI culture.

"5th of May" will occur on Friday in 2006.

Obtain resources using the es-MX culture.

"Cinco de Mayo" will occur on Viernes in 2006.

\*/

## **Remarks**

**Important**

Calling methods from this class with untrusted data is a security risk. Call the methods from this class only with trusted data. For more information, see **[Data Validation](https://www.owasp.org/index.php/Data_Validation)**.

The [ResourceManager](https://docs.microsoft.com/en-us/dotnet/api/system.resources.resourcemanager?view=netframework-4.8) class retrieves resources from a binary .resources file that is embedded in an assembly or from a standalone .resources file. If an app has been localized and localized resources have been deployed in [satellite assemblies](https://docs.microsoft.com/en-us/dotnet/framework/resources/creating-satellite-assemblies-for-desktop-apps?view=netframework-4.8), it looks up culture-specific resources, provides resource fallback when a localized resource does not exist, and supports resource serialization.

For more information about creating and managing resources in desktop apps and Windows 8.x Store apps, see the following sections:

[Desktop Apps](https://docs.microsoft.com/en-us/dotnet/api/system.resources.resourcemanager?view=netframework-4.8" \l "desktop)

[Creating Resources](https://docs.microsoft.com/en-us/dotnet/api/system.resources.resourcemanager?view=netframework-4.8" \l "creating_resources)

[Instantiating a ResourceManager Object](https://docs.microsoft.com/en-us/dotnet/api/system.resources.resourcemanager?view=netframework-4.8" \l "instantiating)

[ResourceManager and Culture-Specific Resources](https://docs.microsoft.com/en-us/dotnet/api/system.resources.resourcemanager?view=netframework-4.8" \l "CultureSpecific)

[Retrieving Resources](https://docs.microsoft.com/en-us/dotnet/api/system.resources.resourcemanager?view=netframework-4.8" \l "retrieving)

[Handling MissingManifestResourceException and MissingSatelliteAssemblyException Exceptions](https://docs.microsoft.com/en-us/dotnet/api/system.resources.resourcemanager?view=netframework-4.8" \l "exception)

[Resource Versioning](https://docs.microsoft.com/en-us/dotnet/api/system.resources.resourcemanager?view=netframework-4.8" \l "versioning)

[<satelliteassemblies> Configuration File Node](https://docs.microsoft.com/en-us/dotnet/api/system.resources.resourcemanager?view=netframework-4.8" \l "config)

[Windows Store Apps](https://docs.microsoft.com/en-us/dotnet/api/system.resources.resourcemanager?view=netframework-4.8" \l "ws)

### **Desktop Apps**

For desktop apps, the [ResourceManager](https://docs.microsoft.com/en-us/dotnet/api/system.resources.resourcemanager?view=netframework-4.8) class retrieves resources from binary resource (.resources) files. Typically, a language compiler or the [Assembly Linker (AL.exe)](https://docs.microsoft.com/en-us/dotnet/framework/tools/al-exe-assembly-linker?view=netframework-4.8) embeds these resource files in an assembly. You can also use a [ResourceManager](https://docs.microsoft.com/en-us/dotnet/api/system.resources.resourcemanager?view=netframework-4.8) object to retrieve resources directly from a .resources file that is not embedded in an assembly, by calling the [CreateFileBasedResourceManager](https://docs.microsoft.com/en-us/dotnet/api/system.resources.resourcemanager.createfilebasedresourcemanager?view=netframework-4.8) method.

**Caution**

Using standalone .resources files in an ASP.NET app will break XCOPY deployment, because the resources remain locked until they are explicitly released by the **[ReleaseAllResources](https://docs.microsoft.com/en-us/dotnet/api/system.resources.resourcemanager.releaseallresources?view=netframework-4.8)** method. If you want to deploy resources with your ASP.NET apps, you should compile your .resources files into satellite assemblies.

In a resource-based app, one .resources file contains the resources of the default culture whose resources are used if no culture-specific resources can be found. For example, if an app's default culture is English (en), the English language resources are used whenever localized resources cannot be found for a specific culture, such as English (United States) (en-US) or French (France) (fr-FR). Typically, the resources for the default culture are embedded in the main app assembly, and resources for other localized cultures are embedded in satellite assemblies. Satellite assemblies contain only resources. They have the same root file name as the main assembly and an extension of .resources.dll. For apps whose assemblies are not registered in the global assembly cache, satellite assemblies are stored in an app subdirectory whose name corresponds to the assembly's culture.

#### **Creating Resources**

When you develop a resource-based app, you store resource information in text files (files that have a .txt or .restext extension) or XML files (files that have a .resx extension). You then compile the text or XML files with the [Resource File Generator (Resgen.exe)](https://docs.microsoft.com/en-us/dotnet/framework/tools/resgen-exe-resource-file-generator?view=netframework-4.8) to create a binary .resources file. You can then embed the resulting .resources file in an executable or library by using a compiler option such as /resources for the C# and Visual Basic compilers, or you can embed it in a satellite assembly by using the . If you include a .resx file in your Visual Studio project, Visual Studio handles the compilation and embedding of default and localized resources automatically as part of the build process.

Ideally, you should create resources for every language your app supports, or at least for a meaningful subset of each language. The binary .resources file names follow the naming convention basename.cultureName.resources, where basename is the name of the app or the name of a class, depending on the level of detail you want. The [CultureInfo.Name](https://docs.microsoft.com/en-us/dotnet/api/system.globalization.cultureinfo.name?view=netframework-4.8) property is used to determine cultureName. A resource for the app's default culture should be named basename.resources.

For example, suppose that an assembly has several resources in a resource file that has the base name MyResources. These resource files should have names such as MyResources.ja-JP.resources for the Japan (Japanese) culture, MyResources.de.resources for the German culture, MyResources.zh-CHS.resources for the simplified Chinese culture, and MyResources.fr-BE.resources for the French (Belgium) culture. The default resource file should be named MyResources.resources. The culture-specific resource files are commonly packaged in satellite assemblies for each culture. The default resource file should be embedded in the app's main assembly.

Note that [Assembly Linker](https://docs.microsoft.com/en-us/dotnet/framework/tools/al-exe-assembly-linker?view=netframework-4.8) allows resources to be marked as private, but you should always mark them as public so they can be accessed by other assemblies. (Because a satellite assembly contains no code, resources that are marked as private are unavailable to your app through any mechanism.)

For more information about creating, packaging, and deploying resources, see the articles [Creating Resource Files](https://docs.microsoft.com/en-us/dotnet/framework/resources/creating-resource-files-for-desktop-apps?view=netframework-4.8), [Creating Satellite Assemblies](https://docs.microsoft.com/en-us/dotnet/framework/resources/creating-satellite-assemblies-for-desktop-apps?view=netframework-4.8), and [Packaging and Deploying Resources](https://docs.microsoft.com/en-us/dotnet/framework/resources/packaging-and-deploying-resources-in-desktop-apps?view=netframework-4.8).

#### **Instantiating a ResourceManager Object**

You instantiate a [ResourceManager](https://docs.microsoft.com/en-us/dotnet/api/system.resources.resourcemanager?view=netframework-4.8) object that retrieves resources from an embedded .resources file by calling one of its class constructor overloads. This tightly couples a [ResourceManager](https://docs.microsoft.com/en-us/dotnet/api/system.resources.resourcemanager?view=netframework-4.8) object with a particular .resources file and with any associated localized .resources files in satellite assemblies.

The two most commonly called constructors are:

[ResourceManager(String, Assembly)](https://docs.microsoft.com/en-us/dotnet/api/system.resources.resourcemanager.-ctor?view=netframework-4.8" \l "System_Resources_ResourceManager__ctor_System_String_System_Reflection_Assembly_) looks up resources based on two pieces of information that you supply: the base name of the .resources file, and the assembly in which the default .resources file resides. The base name includes the namespace and root name of the .resources file, without its culture or extension. Note that .resources files that are compiled from the command line typically do not include a namespace name, whereas .resources files that are created in the Visual Studio environment do. For example, if a resource file is named MyCompany.StringResources.resources and the [ResourceManager](https://docs.microsoft.com/en-us/dotnet/api/system.resources.resourcemanager?view=netframework-4.8) constructor is called from a static method named Example.Main, the following code instantiates a [ResourceManager](https://docs.microsoft.com/en-us/dotnet/api/system.resources.resourcemanager?view=netframework-4.8) object that can retrieve resources from the .resources file:

ResourceManager rm = new ResourceManager("MyCompany.StringResources",

typeof(Example).Assembly);

[ResourceManager(Type)](https://docs.microsoft.com/en-us/dotnet/api/system.resources.resourcemanager.-ctor?view=netframework-4.8" \l "System_Resources_ResourceManager__ctor_System_Type_) looks up resources in satellite assemblies based on information from a type object. The type's fully qualified name corresponds to the base name of the .resources file without its file name extension. In desktop apps that are created by using the Visual Studio Resource Designer, Visual Studio creates a wrapper class whose fully qualified name is the same as the root name of the .resources file. For example, if a resource file is named MyCompany.StringResources.resources and there is a wrapper class named MyCompany.StringResources, the following code instantiates a [ResourceManager](https://docs.microsoft.com/en-us/dotnet/api/system.resources.resourcemanager?view=netframework-4.8) object that can retrieve resources from the .resources file:

ResourceManager rm = new ResourceManager(typeof(MyCompany.StringResources));

If the appropriate resources cannot be found, the constructor call creates a valid [ResourceManager](https://docs.microsoft.com/en-us/dotnet/api/system.resources.resourcemanager?view=netframework-4.8) object. However, the attempt to retrieve a resource throws a [MissingManifestResourceException](https://docs.microsoft.com/en-us/dotnet/api/system.resources.missingmanifestresourceexception?view=netframework-4.8) exception. For information about dealing with the exception, see the [Handling MissingManifestResourceException and MissingSatelliteAssembly Exceptions](https://docs.microsoft.com/en-us/dotnet/api/system.resources.resourcemanager?view=netframework-4.8" \l "exception) section later in this article.

The following example shows how to instantiate a [ResourceManager](https://docs.microsoft.com/en-us/dotnet/api/system.resources.resourcemanager?view=netframework-4.8) object. It contains the source code for an executable named ShowTime.exe. It also includes the following text file named Strings.txt that contains a single string resource, TimeHeader:

Copy

TimeHeader=The current time is

You can use a batch file to generate the resource file and embed it into the executable. Here's the batch file to generate an executable by using the C# compiler:

resgen strings.txt

csc ShowTime.cs /resource:strings.resources

For the Visual Basic compiler, you can use the following batch file:

Copy

resgen strings.txt

vbc ShowTime.vb /resource:strings.resources

using System;using System.Resources;

public class Example

{

public static void Main()

{

ResourceManager rm = new ResourceManager("Strings",

typeof(Example).Assembly);

string timeString = rm.GetString("TimeHeader");

Console.WriteLine("{0} {1:T}", timeString, DateTime.Now);

}

}// The example displays output like the following:// The current time is 2:03:14 PM

#### **ResourceManager and Culture-Specific Resources**

A localized app requires resources to be deployed, as discussed in the article [Packaging and Deploying Resources](https://docs.microsoft.com/en-us/dotnet/framework/resources/packaging-and-deploying-resources-in-desktop-apps?view=netframework-4.8). If the assemblies are properly configured, the resource manager determines which resources to retrieve based on the current thread's [Thread.CurrentUICulture](https://docs.microsoft.com/en-us/dotnet/api/system.threading.thread.currentuiculture?view=netframework-4.8) property. (That property also returns the current thread's UI culture.) For example, if an app is compiled with default English language resources in the main assembly and with French and Russian language resources in two satellite assemblies, and the [Thread.CurrentUICulture](https://docs.microsoft.com/en-us/dotnet/api/system.threading.thread.currentuiculture?view=netframework-4.8) property is set to fr-FR, the resource manager retrieves the French resources.

You can set the [CurrentUICulture](https://docs.microsoft.com/en-us/dotnet/api/system.globalization.cultureinfo.currentuiculture?view=netframework-4.8) property explicitly or implicitly. The way you set it determines how the [ResourceManager](https://docs.microsoft.com/en-us/dotnet/api/system.resources.resourcemanager?view=netframework-4.8) object retrieves resources based on culture:

If you explicitly set the [Thread.CurrentUICulture](https://docs.microsoft.com/en-us/dotnet/api/system.threading.thread.currentuiculture?view=netframework-4.8) property to a specific culture, the resource manager always retrieves the resources for that culture, regardless of the user's browser or operating system language. Consider an app that is compiled with default English language resources and three satellite assemblies that contain resources for English (United States), French (France), and Russian (Russia). If the [CurrentUICulture](https://docs.microsoft.com/en-us/dotnet/api/system.globalization.cultureinfo.currentuiculture?view=netframework-4.8) property is set to fr-FR, the [ResourceManager](https://docs.microsoft.com/en-us/dotnet/api/system.resources.resourcemanager?view=netframework-4.8) object always retrieves the French (France) resources, even if the user's operating system language is not French. Make sure that this is the desired behavior before you set the property explicitly.

In ASP.NET apps, you must set the [Thread.CurrentUICulture](https://docs.microsoft.com/en-us/dotnet/api/system.threading.thread.currentuiculture?view=netframework-4.8) property explicitly, because it is unlikely that the setting on the server will match incoming client requests. An ASP.NET app can set the [Thread.CurrentUICulture](https://docs.microsoft.com/en-us/dotnet/api/system.threading.thread.currentuiculture?view=netframework-4.8) property explicitly to the user's browser accept language.

Explicitly setting the [Thread.CurrentUICulture](https://docs.microsoft.com/en-us/dotnet/api/system.threading.thread.currentuiculture?view=netframework-4.8) property defines the current UI culture for that thread. It does not affect the current UI culture of any other threads in an app.

You can set the UI culture of all threads in an app domain by assigning a [CultureInfo](https://docs.microsoft.com/en-us/dotnet/api/system.globalization.cultureinfo?view=netframework-4.8) object that represents that culture to the static [CultureInfo.DefaultThreadCurrentUICulture](https://docs.microsoft.com/en-us/dotnet/api/system.globalization.cultureinfo.defaultthreadcurrentuiculture?view=netframework-4.8) property.

If you do not explicitly set the current UI culture and you do not define a default culture for the current app domain, the [CultureInfo.CurrentUICulture](https://docs.microsoft.com/en-us/dotnet/api/system.globalization.cultureinfo.currentuiculture?view=netframework-4.8) property is set implicitly by the Windows GetUserDefaultUILanguage function. This function is provided by the Multilingual User Interface (MUI), which enables the user to set the default language. If the UI language is not set by the user, it defaults to the system-installed language, which is the language of operating system resources.

The following simple "Hello world" example sets the current UI culture explicitly. It contains resources for three cultures: English (United States) or en-US, French (France) or fr-FR, and Russian (Russia) or ru-RU. The en-US resources are contained in a text file named Greetings.txt:

Copy

HelloString=Hello world!

The fr-FR resources are contained in a text file named Greetings.fr-FR.txt:

Copy

HelloString=Salut tout le monde!

The ru-RU resources are contained in a text file named Greetings.ru-RU.txt:

Copy

HelloString=Всем привет!

Here's the source code for the example (Example.vb for the Visual Basic version or Example.cs for the C# version):

C#Copy

using System;using System.Globalization;using System.Resources;using System.Threading;

public class Example

{

public static void Main()

{

// Create array of supported cultures

string[] cultures = {"en-CA", "en-US", "fr-FR", "ru-RU" };

Random rnd = new Random();

int cultureNdx = rnd.Next(0, cultures.Length);

CultureInfo originalCulture = Thread.CurrentThread.CurrentCulture;

ResourceManager rm = new ResourceManager("Greetings", typeof(Example).Assembly);

try {

CultureInfo newCulture = new CultureInfo(cultures[cultureNdx]);

Thread.CurrentThread.CurrentCulture = newCulture;

Thread.CurrentThread.CurrentUICulture = newCulture;

string greeting = String.Format("The current culture is {0}.\n{1}",

Thread.CurrentThread.CurrentUICulture.Name,

rm.GetString("HelloString"));

Console.WriteLine(greeting);

}

catch (CultureNotFoundException e) {

Console.WriteLine("Unable to instantiate culture {0}", e.InvalidCultureName);

}

finally {

Thread.CurrentThread.CurrentCulture = originalCulture;

Thread.CurrentThread.CurrentUICulture = originalCulture;

}

}

}// The example displays output like the following:// The current culture is ru-RU.// Всем привет!

To compile this example, create a batch (.bat) file that contains the following commands and run it from the command prompt. If you're using C#, specify csc instead of vbc and Example.cs instead of Example.vb.

Copy

resgen Greetings.txt

vbc Example.vb /resource:Greetings.resources

resgen Greetings.fr-FR.txt

Md fr-FR

al /embed:Greetings.fr-FR.resources /culture:fr-FR /out:fr-FR\Example.resources.dll

resgen Greetings.ru-RU.txt

Md ru-RU

al /embed:Greetings.ru-RU.resources /culture:ru-RU /out:ru-RU\Example.resources.dll

#### **Retrieving Resources**

You call the [GetObject(String)](https://docs.microsoft.com/en-us/dotnet/api/system.resources.resourcemanager.getobject?view=netframework-4.8" \l "System_Resources_ResourceManager_GetObject_System_String_) and [GetString(String)](https://docs.microsoft.com/en-us/dotnet/api/system.resources.resourcemanager.getstring?view=netframework-4.8" \l "System_Resources_ResourceManager_GetString_System_String_) methods to access a specific resource. You can also call the [GetStream(String)](https://docs.microsoft.com/en-us/dotnet/api/system.resources.resourcemanager.getstream?view=netframework-4.8" \l "System_Resources_ResourceManager_GetStream_System_String_) method to retrieve non-string resources as a byte array. By default, in an app that has localized resources, these methods return the resource for the culture determined by the current UI culture of the thread that made the call. See the previous section, [ResourceManager and Culture-Specific Resources](https://docs.microsoft.com/en-us/dotnet/api/system.resources.resourcemanager?view=netframework-4.8" \l "CultureSpecific), for more information about how the current UI culture of a thread is defined. If the resource manager cannot find the resource for the current thread's UI culture, it uses a fallback process to retrieve the specified resource. If the resource manager cannot find any localized resources, it uses the resources of the default culture. For more information about resource fallback rules, see the "Resource Fallback Process" section of the article [Packaging and Deploying Resources](https://docs.microsoft.com/en-us/dotnet/framework/resources/packaging-and-deploying-resources-in-desktop-apps?view=netframework-4.8).

**Note**

If the .resources file specified in the **[ResourceManager](https://docs.microsoft.com/en-us/dotnet/api/system.resources.resourcemanager?view=netframework-4.8)** class constructor cannot be found, the attempt to retrieve a resource throws a **[MissingManifestResourceException](https://docs.microsoft.com/en-us/dotnet/api/system.resources.missingmanifestresourceexception?view=netframework-4.8)** or **[MissingSatelliteAssemblyException](https://docs.microsoft.com/en-us/dotnet/api/system.resources.missingsatelliteassemblyexception?view=netframework-4.8)** exception. For information about dealing with the exception, see the **[Handling MissingManifestResourceException and MissingSatelliteAssemblyException Exceptions](https://docs.microsoft.com/en-us/dotnet/api/system.resources.resourcemanager?view=netframework-4.8" \l "exception)** section later in this topic.

The following example uses the [GetString](https://docs.microsoft.com/en-us/dotnet/api/system.resources.resourcemanager.getstring?view=netframework-4.8) method to retrieve culture-specific resources. It consists of resources compiled from .txt files for the English (en), French (France) (fr-FR), and Russian (Russia) (ru-RU) cultures. The example changes the current culture and current UI culture to English (United States), French (France), Russian (Russia), and Swedish (Sweden). It then calls the [GetString](https://docs.microsoft.com/en-us/dotnet/api/system.resources.resourcemanager.getstring?view=netframework-4.8) method to retrieve the localized string, which it displays along with the current day and month. Notice that the output displays the appropriate localized string except when the current UI culture is Swedish (Sweden). Because Swedish language resources are unavailable, the app instead uses the resources of the default culture, which is English.

The example requires the text-based resource files listed in following table. Each has a single string resource named DateStart.

| **Culture** | **File name** | **Resource name** | **Resource value** |
| --- | --- | --- | --- |
| en-US | DateStrings.txt | DateStart | Today is |
| fr-FR | DateStrings.fr-FR.txt | DateStart | Aujourd'hui, c'est le |
| ru-RU | DateStrings.ru-RU.txt | DateStart | Сегодня |

Here's the source code for the example (ShowDate.vb for the Visual Basic version or ShowDate.cs for the C# version of the code).

C#Copy

using System;using System.Globalization;using System.Resources;using System.Threading;

[assembly:NeutralResourcesLanguage("en")]

public class Example

{

public static void Main()

{

string[] cultureNames = { "en-US", "fr-FR", "ru-RU", "sv-SE" };

ResourceManager rm = new ResourceManager("DateStrings",

typeof(Example).Assembly);

foreach (var cultureName in cultureNames) {

CultureInfo culture = CultureInfo.CreateSpecificCulture(cultureName);

Thread.CurrentThread.CurrentCulture = culture;

Thread.CurrentThread.CurrentUICulture = culture;

Console.WriteLine("Current UI Culture: {0}",

CultureInfo.CurrentUICulture.Name);

string dateString = rm.GetString("DateStart");

Console.WriteLine("{0} {1:M}.\n", dateString, DateTime.Now);

}

}

}// The example displays output similar to the following:// Current UI Culture: en-US// Today is February 03.// // Current UI Culture: fr-FR// Aujourd'hui, c'est le 3 février// // Current UI Culture: ru-RU// Сегодня февраля 03.// // Current UI Culture: sv-SE// Today is den 3 februari.

To compile this example, create a batch file that contains the following commands and run it from the command prompt. If you're using C#, specify csc instead of vbc and showdate.cs instead of showdate.vb.

Copy

resgen DateStrings.txt

vbc showdate.vb /resource:DateStrings.resources

md fr-FR

resgen DateStrings.fr-FR.txt

al /out:fr-FR\Showdate.resources.dll /culture:fr-FR /embed:DateStrings.fr-FR.resources

md ru-RU

resgen DateStrings.ru-RU.txt

al /out:ru-RU\Showdate.resources.dll /culture:ru-RU /embed:DateStrings.ru-RU.resources

There are two ways to retrieve the resources of a specific culture other than the current UI culture:

You can call the [GetString(String, CultureInfo)](https://docs.microsoft.com/en-us/dotnet/api/system.resources.resourcemanager.getstring?view=netframework-4.8" \l "System_Resources_ResourceManager_GetString_System_String_System_Globalization_CultureInfo_), [GetObject(String, CultureInfo)](https://docs.microsoft.com/en-us/dotnet/api/system.resources.resourcemanager.getobject?view=netframework-4.8" \l "System_Resources_ResourceManager_GetObject_System_String_System_Globalization_CultureInfo_), or [GetStream(String, CultureInfo)](https://docs.microsoft.com/en-us/dotnet/api/system.resources.resourcemanager.getstream?view=netframework-4.8" \l "System_Resources_ResourceManager_GetStream_System_String_System_Globalization_CultureInfo_) method to retrieve a resource for a specific culture. If a localized resource cannot be found, the resource manager uses the resource fallback process to locate an appropriate resource.

You can call the [GetResourceSet](https://docs.microsoft.com/en-us/dotnet/api/system.resources.resourcemanager.getresourceset?view=netframework-4.8) method to obtain a [ResourceSet](https://docs.microsoft.com/en-us/dotnet/api/system.resources.resourceset?view=netframework-4.8) object that represents the resources for a particular culture. In the method call, you can determine whether the resource manager probes for parent cultures if it is unable to find localized resources, or whether it simply falls back to the resources of the default culture. You can then use the [ResourceSet](https://docs.microsoft.com/en-us/dotnet/api/system.resources.resourceset?view=netframework-4.8) methods to access the resources (localized for that culture) by name, or to enumerate the resources in the set.

#### **Handling MissingManifestResourceException and MissingSatelliteAssemblyException Exceptions**

If you try to retrieve a specific resource, but the resource manager cannot find that resource and either no default culture has been defined or the resources of the default culture cannot be located, the resource manager throws a [MissingManifestResourceException](https://docs.microsoft.com/en-us/dotnet/api/system.resources.missingmanifestresourceexception?view=netframework-4.8) exception if it expects to find the resources in the main assembly or a [MissingSatelliteAssemblyException](https://docs.microsoft.com/en-us/dotnet/api/system.resources.missingsatelliteassemblyexception?view=netframework-4.8) if it expects to find the resources in a satellite assembly. Note that the exception is thrown when you call a resource retrieval method such as [GetString](https://docs.microsoft.com/en-us/dotnet/api/system.resources.resourcemanager.getstring?view=netframework-4.8) or [GetObject](https://docs.microsoft.com/en-us/dotnet/api/system.resources.resourcemanager.getobject?view=netframework-4.8), and not when you instantiate a [ResourceManager](https://docs.microsoft.com/en-us/dotnet/api/system.resources.resourcemanager?view=netframework-4.8) object.

The exception is typically thrown under the following conditions:

The appropriate resource file or satellite assembly does not exist. If the resource manager expects the app's default resources to be embedded in the main app assembly, they are absent. If the [NeutralResourcesLanguageAttribute](https://docs.microsoft.com/en-us/dotnet/api/system.resources.neutralresourceslanguageattribute?view=netframework-4.8) attribute indicates that the app's default resources reside in a satellite assembly, that assembly cannot be found. When you compile your app, make sure that resources are embedded in the main assembly or that the necessary satellite assembly is generated and is named appropriately. Its name should take the form appName.resources.dll, and it should be located in a directory named after the culture whose resources it contains.

Your app doesn't have a default or neutral culture defined. Add the [NeutralResourcesLanguageAttribute](https://docs.microsoft.com/en-us/dotnet/api/system.resources.neutralresourceslanguageattribute?view=netframework-4.8) attribute to a source code file or to the project information file (AssemblyInfo.vb for a Visual Basic app or AssemblyInfo.cs for a C# app) file.

The baseName parameter in the [ResourceManager(String, Assembly)](https://docs.microsoft.com/en-us/dotnet/api/system.resources.resourcemanager.-ctor?view=netframework-4.8" \l "System_Resources_ResourceManager__ctor_System_String_System_Reflection_Assembly_) constructor does not specify the name of a .resources file. The name should include the resource file's fully qualified namespace but not its file name extension. Typically, resource files that are created in Visual Studio include namespace names, but resource files that are created and compiled at the command prompt do not. You can determine the names of embedded .resources files by compiling and running the following utility. This is a console app that accepts the name of a main assembly or satellite assembly as a command-line parameter. It displays the strings that should be provided as the baseName parameter so that the resource manager can correctly identify the resource.

C#Copy

using System;using System.IO;using System.Reflection;using System.Resources;

public class Example

{

public static void Main()

{

if (Environment.GetCommandLineArgs().Length == 1) {

Console.WriteLine("No filename.");

return;

}

string filename = Environment.GetCommandLineArgs()[1].Trim();

// Check whether the file exists.

if (! File.Exists(filename)) {

Console.WriteLine("{0} does not exist.", filename);

return;

}

// Try to load the assembly.

Assembly assem = Assembly.LoadFrom(filename);

Console.WriteLine("File: {0}", filename);

// Enumerate the resource files.

string[] resNames = assem.GetManifestResourceNames();

if (resNames.Length == 0)

Console.WriteLine(" No resources found.");

foreach (var resName in resNames)

Console.WriteLine(" Resource: {0}", resName.Replace(".resources", ""));

Console.WriteLine();

}

}

If you are changing the current culture of your application explicitly, you should also remember that the resource manager retrieves a resource set based on the value of the [CultureInfo.CurrentUICulture](https://docs.microsoft.com/en-us/dotnet/api/system.globalization.cultureinfo.currentuiculture?view=netframework-4.8) property, and not the [CultureInfo.CurrentCulture](https://docs.microsoft.com/en-us/dotnet/api/system.globalization.cultureinfo.currentculture?view=netframework-4.8) property. Typically, if you change one value, you should also change the other.

#### **Resource Versioning**

Because the main assembly that contains an app's default resources is separate from the app's satellite assemblies, you can release a new version of your main assembly without redeploying the satellite assemblies. You use the [SatelliteContractVersionAttribute](https://docs.microsoft.com/en-us/dotnet/api/system.resources.satellitecontractversionattribute?view=netframework-4.8) attribute to use existing satellite assemblies and instruct the resource manager not to redeploy them with a new version of your main assembly,

For more information about versioning support for satellite assemblies, see the article [Retrieving Resources](https://docs.microsoft.com/en-us/dotnet/framework/resources/retrieving-resources-in-desktop-apps?view=netframework-4.8).

#### **<satelliteassemblies> Configuration File Node**

For executables that are deployed and run from a website (HREF .exe files), the [ResourceManager](https://docs.microsoft.com/en-us/dotnet/api/system.resources.resourcemanager?view=netframework-4.8) object may probe for satellite assemblies over the web, which can hurt your app's performance. To eliminate the performance problem, you can limit this probing to the satellite assemblies that you have deployed with your app. To do this, you create a <satelliteassemblies> node in your app's configuration file to specify that you have deployed a specific set of cultures for your app, and that the [ResourceManager](https://docs.microsoft.com/en-us/dotnet/api/system.resources.resourcemanager?view=netframework-4.8) object should not try to probe for any culture that is not listed in that node.

**Note**

The preferred alternative to creating a <satelliteassemblies> node is to use the **[ClickOnce Deployment Manifest](https://docs.microsoft.com/en-us/visualstudio/deployment/clickonce-deployment-manifest)** feature.

In your app's configuration file, create a section similar to the following:

Copy

<?xml version ="1.0"?>

<configuration>

    <satelliteassemblies>

        <assembly name="MainAssemblyName, Version=versionNumber, Culture=neutral, PublicKeyToken=null|yourPublicKeyToken">

            <culture>cultureName1</culture>

            <culture>cultureName2</culture>

            <culture>cultureName3</culture>

        </assembly>

    </satelliteassemblies>

</configuration>

Edit this configuration information as follows:

Specify one or more <assembly> nodes for each main assembly that you deploy, where each node specifies a fully qualified assembly name. Specify the name of your main assembly in place of MainAssemblyName, and specify the Version, PublicKeyToken, and Culture attribute values that correspond to your main assembly.

For the Version attribute, specify the version number of your assembly. For example, the first release of your assembly might be version number 1.0.0.0.

For the PublicKeyToken attribute, specify the keyword null if you have not signed your assembly with a strong name, or specify your public key token if you have signed your assembly.

For the Culture attribute, specify the keyword neutral to designate the main assembly and cause the [ResourceManager](https://docs.microsoft.com/en-us/dotnet/api/system.resources.resourcemanager?view=netframework-4.8) class to probe only for the cultures listed in the <culture> nodes.

For more information about fully qualified assembly names, see the article [Assembly Names](https://docs.microsoft.com/en-us/dotnet/standard/assembly/names?view=netframework-4.8). For more information about strong-named assemblies, see the article [Create and use strong-named assemblies](https://docs.microsoft.com/en-us/dotnet/standard/assembly/create-use-strong-named?view=netframework-4.8).

Specify one or more <culture> nodes with a specific culture name, such as "fr-FR", or a neutral culture name, such as "fr".

If resources are needed for any assembly not listed under the <satelliteassemblies> node, the [ResourceManager](https://docs.microsoft.com/en-us/dotnet/api/system.resources.resourcemanager?view=netframework-4.8) class probes for cultures using standard probing rules.

### **Windows 8.x Store Apps**

**Important**

Although the **[ResourceManager](https://docs.microsoft.com/en-us/dotnet/api/system.resources.resourcemanager?view=netframework-4.8)** class is supported in Windows 8.x Store apps, we do not recommend its use. Use this class only when you develop Portable Class Library projects that can be used with Windows 8.x Store apps. To retrieve resources from Windows 8.x Store apps, use the **[Windows.ApplicationModel.Resources.ResourceLoader](https://go.microsoft.com/fwlink/p/?LinkId=238182)** class instead.

For Windows 8.x Store apps, the [ResourceManager](https://docs.microsoft.com/en-us/dotnet/api/system.resources.resourcemanager?view=netframework-4.8) class retrieves resources from package resource index (PRI) files. A single PRI file (the application package PRI file) contains the resources for both the default culture and any localized cultures. You use the MakePRI utility to create a PRI file from one or more resource files that are in XML resource (.resw) format. For resources that are included in a Visual Studio project, Visual Studio handles the process of creating and packaging the PRI file automatically. You can then use the .NET Framework [ResourceManager](https://docs.microsoft.com/en-us/dotnet/api/system.resources.resourcemanager?view=netframework-4.8) class to access the app's or library's resources.

You can instantiate a [ResourceManager](https://docs.microsoft.com/en-us/dotnet/api/system.resources.resourcemanager?view=netframework-4.8) object for a Windows 8.x Store app in the same way that you do for a desktop app.

You can then access the resources for a particular culture by passing the name of the resource to be retrieved to the [GetString(String)](https://docs.microsoft.com/en-us/dotnet/api/system.resources.resourcemanager.getstring?view=netframework-4.8" \l "System_Resources_ResourceManager_GetString_System_String_) method. By default, this method returns the resource for the culture determined by the current UI culture of the thread that made the call. You can also retrieve the resources for a specific culture by passing the name of the resource and a [CultureInfo](https://docs.microsoft.com/en-us/dotnet/api/system.globalization.cultureinfo?view=netframework-4.8) object that represents the culture whose resource is to be retrieved to the [GetString(String, CultureInfo)](https://docs.microsoft.com/en-us/dotnet/api/system.resources.resourcemanager.getstring?view=netframework-4.8" \l "System_Resources_ResourceManager_GetString_System_String_System_Globalization_CultureInfo_) method. If the resource for the current UI culture or the specified culture cannot be found, the resource manager uses a UI language fallback list to locate a suitable resource.

## **Constructors**

|  |  |
| --- | --- |
| [ResourceManager()](https://docs.microsoft.com/en-us/dotnet/api/system.resources.resourcemanager.-ctor?view=netframework-4.8" \l "System_Resources_ResourceManager__ctor) | Initializes a new instance of the [ResourceManager](https://docs.microsoft.com/en-us/dotnet/api/system.resources.resourcemanager?view=netframework-4.8) class with default values. |
| [ResourceManager(String, Assembly)](https://docs.microsoft.com/en-us/dotnet/api/system.resources.resourcemanager.-ctor?view=netframework-4.8" \l "System_Resources_ResourceManager__ctor_System_String_System_Reflection_Assembly_) | Initializes a new instance of the [ResourceManager](https://docs.microsoft.com/en-us/dotnet/api/system.resources.resourcemanager?view=netframework-4.8) class that looks up resources contained in files with the specified root name in the given assembly. |
| [ResourceManager(String, Assembly, Type)](https://docs.microsoft.com/en-us/dotnet/api/system.resources.resourcemanager.-ctor?view=netframework-4.8" \l "System_Resources_ResourceManager__ctor_System_String_System_Reflection_Assembly_System_Type_) | Initializes a new instance of the [ResourceManager](https://docs.microsoft.com/en-us/dotnet/api/system.resources.resourcemanager?view=netframework-4.8) class that uses a specified [ResourceSet](https://docs.microsoft.com/en-us/dotnet/api/system.resources.resourceset?view=netframework-4.8) class to look up resources contained in files with the specified root name in the given assembly. |
| [ResourceManager(Type)](https://docs.microsoft.com/en-us/dotnet/api/system.resources.resourcemanager.-ctor?view=netframework-4.8" \l "System_Resources_ResourceManager__ctor_System_Type_) | Initializes a new instance of the [ResourceManager](https://docs.microsoft.com/en-us/dotnet/api/system.resources.resourcemanager?view=netframework-4.8) class that looks up resources in satellite assemblies based on information from the specified type object. |

## **Fields**

|  |  |
| --- | --- |
| [BaseNameField](https://docs.microsoft.com/en-us/dotnet/api/system.resources.resourcemanager.basenamefield?view=netframework-4.8) | Specifies the root name of the resource files that the [ResourceManager](https://docs.microsoft.com/en-us/dotnet/api/system.resources.resourcemanager?view=netframework-4.8) searches for resources. |
| [HeaderVersionNumber](https://docs.microsoft.com/en-us/dotnet/api/system.resources.resourcemanager.headerversionnumber?view=netframework-4.8) | Specifies the version of resource file headers that the current implementation of [ResourceManager](https://docs.microsoft.com/en-us/dotnet/api/system.resources.resourcemanager?view=netframework-4.8) can interpret and produce. |
| [MagicNumber](https://docs.microsoft.com/en-us/dotnet/api/system.resources.resourcemanager.magicnumber?view=netframework-4.8) | Holds the number used to identify resource files. |
| [MainAssembly](https://docs.microsoft.com/en-us/dotnet/api/system.resources.resourcemanager.mainassembly?view=netframework-4.8) | Specifies the main assembly that contains the resources. |
| [ResourceSets](https://docs.microsoft.com/en-us/dotnet/api/system.resources.resourcemanager.resourcesets?view=netframework-4.8) | Contains a [Hashtable](https://docs.microsoft.com/en-us/dotnet/api/system.collections.hashtable?view=netframework-4.8) that returns a mapping from cultures to [ResourceSet](https://docs.microsoft.com/en-us/dotnet/api/system.resources.resourceset?view=netframework-4.8) objects. |

## **Properties**

|  |  |
| --- | --- |
| [BaseName](https://docs.microsoft.com/en-us/dotnet/api/system.resources.resourcemanager.basename?view=netframework-4.8" \l "System_Resources_ResourceManager_BaseName) | Gets the root name of the resource files that the [ResourceManager](https://docs.microsoft.com/en-us/dotnet/api/system.resources.resourcemanager?view=netframework-4.8) searches for resources. |
| [FallbackLocation](https://docs.microsoft.com/en-us/dotnet/api/system.resources.resourcemanager.fallbacklocation?view=netframework-4.8" \l "System_Resources_ResourceManager_FallbackLocation) | Gets or sets the location from which to retrieve default fallback resources. |
| [IgnoreCase](https://docs.microsoft.com/en-us/dotnet/api/system.resources.resourcemanager.ignorecase?view=netframework-4.8" \l "System_Resources_ResourceManager_IgnoreCase) | Gets or sets a value that indicates whether the resource manager allows case-insensitive resource lookups in the [GetString(String)](https://docs.microsoft.com/en-us/dotnet/api/system.resources.resourcemanager.getstring?view=netframework-4.8" \l "System_Resources_ResourceManager_GetString_System_String_) and [GetObject(String)](https://docs.microsoft.com/en-us/dotnet/api/system.resources.resourcemanager.getobject?view=netframework-4.8" \l "System_Resources_ResourceManager_GetObject_System_String_) methods. |
| [ResourceSetType](https://docs.microsoft.com/en-us/dotnet/api/system.resources.resourcemanager.resourcesettype?view=netframework-4.8" \l "System_Resources_ResourceManager_ResourceSetType) | Gets the type of the resource set object that the resource manager uses to construct a [ResourceSet](https://docs.microsoft.com/en-us/dotnet/api/system.resources.resourceset?view=netframework-4.8) object. |

## **Methods**

|  |  |
| --- | --- |
| [CreateFileBasedResourceManager(String, String, Type)](https://docs.microsoft.com/en-us/dotnet/api/system.resources.resourcemanager.createfilebasedresourcemanager?view=netframework-4.8" \l "System_Resources_ResourceManager_CreateFileBasedResourceManager_System_String_System_String_System_Type_) | Returns a [ResourceManager](https://docs.microsoft.com/en-us/dotnet/api/system.resources.resourcemanager?view=netframework-4.8) object that searches a specific directory instead of an assembly manifest for resources. |
| [Equals(Object)](https://docs.microsoft.com/en-us/dotnet/api/system.object.equals?view=netframework-4.8" \l "System_Object_Equals_System_Object_) | Determines whether the specified object is equal to the current object.  (Inherited from [Object](https://docs.microsoft.com/en-us/dotnet/api/system.object?view=netframework-4.8)) |
| [GetHashCode()](https://docs.microsoft.com/en-us/dotnet/api/system.object.gethashcode?view=netframework-4.8" \l "System_Object_GetHashCode) | Serves as the default hash function.  (Inherited from [Object](https://docs.microsoft.com/en-us/dotnet/api/system.object?view=netframework-4.8)) |
| [GetNeutralResourcesLanguage(Assembly)](https://docs.microsoft.com/en-us/dotnet/api/system.resources.resourcemanager.getneutralresourceslanguage?view=netframework-4.8" \l "System_Resources_ResourceManager_GetNeutralResourcesLanguage_System_Reflection_Assembly_) | Returns culture-specific information for the main assembly's default resources by retrieving the value of the [NeutralResourcesLanguageAttribute](https://docs.microsoft.com/en-us/dotnet/api/system.resources.neutralresourceslanguageattribute?view=netframework-4.8) attribute on a specified assembly. |
| [GetObject(String)](https://docs.microsoft.com/en-us/dotnet/api/system.resources.resourcemanager.getobject?view=netframework-4.8" \l "System_Resources_ResourceManager_GetObject_System_String_) | Returns the value of the specified non-string resource. |
| [GetObject(String, CultureInfo)](https://docs.microsoft.com/en-us/dotnet/api/system.resources.resourcemanager.getobject?view=netframework-4.8" \l "System_Resources_ResourceManager_GetObject_System_String_System_Globalization_CultureInfo_) | Gets the value of the specified non-string resource localized for the specified culture. |
| [GetResourceFileName(CultureInfo)](https://docs.microsoft.com/en-us/dotnet/api/system.resources.resourcemanager.getresourcefilename?view=netframework-4.8" \l "System_Resources_ResourceManager_GetResourceFileName_System_Globalization_CultureInfo_) | Generates the name of the resource file for the given [CultureInfo](https://docs.microsoft.com/en-us/dotnet/api/system.globalization.cultureinfo?view=netframework-4.8) object. |
| [GetResourceSet(CultureInfo, Boolean, Boolean)](https://docs.microsoft.com/en-us/dotnet/api/system.resources.resourcemanager.getresourceset?view=netframework-4.8" \l "System_Resources_ResourceManager_GetResourceSet_System_Globalization_CultureInfo_System_Boolean_System_Boolean_) | Retrieves the resource set for a particular culture. |
| [GetSatelliteContractVersion(Assembly)](https://docs.microsoft.com/en-us/dotnet/api/system.resources.resourcemanager.getsatellitecontractversion?view=netframework-4.8" \l "System_Resources_ResourceManager_GetSatelliteContractVersion_System_Reflection_Assembly_) | Returns the version specified by the [SatelliteContractVersionAttribute](https://docs.microsoft.com/en-us/dotnet/api/system.resources.satellitecontractversionattribute?view=netframework-4.8) attribute in the given assembly. |
| [GetStream(String)](https://docs.microsoft.com/en-us/dotnet/api/system.resources.resourcemanager.getstream?view=netframework-4.8" \l "System_Resources_ResourceManager_GetStream_System_String_) | Returns an unmanaged memory stream object from the specified resource. |
| [GetStream(String, CultureInfo)](https://docs.microsoft.com/en-us/dotnet/api/system.resources.resourcemanager.getstream?view=netframework-4.8" \l "System_Resources_ResourceManager_GetStream_System_String_System_Globalization_CultureInfo_) | Returns an unmanaged memory stream object from the specified resource, using the specified culture. |
| [GetString(String)](https://docs.microsoft.com/en-us/dotnet/api/system.resources.resourcemanager.getstring?view=netframework-4.8" \l "System_Resources_ResourceManager_GetString_System_String_) | Returns the value of the specified string resource. |
| [GetString(String, CultureInfo)](https://docs.microsoft.com/en-us/dotnet/api/system.resources.resourcemanager.getstring?view=netframework-4.8" \l "System_Resources_ResourceManager_GetString_System_String_System_Globalization_CultureInfo_) | Returns the value of the string resource localized for the specified culture. |
| [GetType()](https://docs.microsoft.com/en-us/dotnet/api/system.object.gettype?view=netframework-4.8" \l "System_Object_GetType) | Gets the [Type](https://docs.microsoft.com/en-us/dotnet/api/system.type?view=netframework-4.8) of the current instance.  (Inherited from [Object](https://docs.microsoft.com/en-us/dotnet/api/system.object?view=netframework-4.8)) |
| [InternalGetResourceSet(CultureInfo, Boolean, Boolean)](https://docs.microsoft.com/en-us/dotnet/api/system.resources.resourcemanager.internalgetresourceset?view=netframework-4.8" \l "System_Resources_ResourceManager_InternalGetResourceSet_System_Globalization_CultureInfo_System_Boolean_System_Boolean_) | Provides the implementation for finding a resource set. |
| [MemberwiseClone()](https://docs.microsoft.com/en-us/dotnet/api/system.object.memberwiseclone?view=netframework-4.8" \l "System_Object_MemberwiseClone) | Creates a shallow copy of the current [Object](https://docs.microsoft.com/en-us/dotnet/api/system.object?view=netframework-4.8).  (Inherited from [Object](https://docs.microsoft.com/en-us/dotnet/api/system.object?view=netframework-4.8)) |
| [ReleaseAllResources()](https://docs.microsoft.com/en-us/dotnet/api/system.resources.resourcemanager.releaseallresources?view=netframework-4.8" \l "System_Resources_ResourceManager_ReleaseAllResources) | Tells the resource manager to call the [Close()](https://docs.microsoft.com/en-us/dotnet/api/system.resources.resourceset.close?view=netframework-4.8" \l "System_Resources_ResourceSet_Close) method on all [ResourceSet](https://docs.microsoft.com/en-us/dotnet/api/system.resources.resourceset?view=netframework-4.8) objects and release all resources. |
| [ToString()](https://docs.microsoft.com/en-us/dotnet/api/system.object.tostring?view=netframework-4.8" \l "System_Object_ToString) | Returns a string that represents the current object.  (Inherited from [Object](https://docs.microsoft.com/en-us/dotnet/api/system.object?view=netframework-4.8)) |

<https://docs.microsoft.com/en-us/dotnet/api/system.resources.resourcemanager?view=netframework-4.8>