Assemblies

# Generic terms

## Applications for Global markets

When writing applications for international distribution, different cultures and regions should be kept in mind. Different cultures has diverging calendars and use different number and date formats. To make applications fit for global markets, you have to **localize** and **globalize**. .NET supports globalization and Localization of windows and web applications.

## Globalization

Globalization is the process of designing the application in such a way that it can be used by users from across the globe (multiple cultures). Classes in the **System.Globalization** facilitates globalization.

## Localization

Localization, on the other hand, is the process of customization to make our application behave as per the current culture and locale. Classes in the **System.Resources** facilitates localization.

## Cultures and CultureInfo class

Every culture in .Net is identified by primary and secondary tags. Primary tag refers to language and secondary tag refers to Locale it represents. For e.g. "en-US" represents English language of United States locale. "ar-SA" represents arabic language spoken in Saudi Arabia Locale.

### Types of Cultures

1. **Neutral Culture**: Neutral culture is a culture that is associated with a language but not with a country or region.
2. **Specific Culture**: Specific culture is a culture that is associated with both a language and a country or region. For example, fr is the name for the neutral French culture, while fr-FR is the name for the French culture in France.
3. **Invariant culture**: It is associated with the English language (for historical reasons) but not with any country/region. We specify the invariant culture by name by using an empty string ("") in the call to a CultureInfo instantiation method.

### Class CultureInfo

This class holds information about a specific culture

* Names for the culture
* Writing system
* Calendar used
* Sort order of strings
* Formatting dates and numbers

## Locale

Net identifies specific regions of the world as locale. .Net assigns unique Ids to each region of the world. Microsoft Locale ID Values.

<https://docs.microsoft.com/en-us/previous-versions/windows/embedded/ms912047(v=winembedded.10)?redirectedfrom=MSDN>

## Versioning

**MAJOR.MINOR.REVISION.BUILDNUMBER**

**A possible scheme.**

**1.0.1.0 - Start**

**1.0.1.1 - added the utils class and compiled this.**

**1.0.1.2 - changed all the collections to LINQ queries.**

**1.1.1.2 - INTRODUCED THE PATIENT MODULE IN THE APPLICATION.**

**1.2.1.2 - HOSPITAL WEB SERVICES INCLUDED IN THIS.**

**1.2.2.2 - ADDED THE PATIENTS WITH AN INSURANCE.**

**1.2.2.3 - Patientid COLUMN CHANGED TO UInt32.**

# Satellite Assemblies

A .Net assembly that contains only culture-specific resources are called satellite assemblies. These assemblies do not contain your C# code or any programming logic. It contains only localization resources which can be text, image, Icon, Audio or any other similar stuff being used in your application. For each culture you want to support in your application you create one separate satellite assembly which will contain localization resources of that specific culture.

For instance if I want my application to support 4 languages - Tamil(ta), Hindi(hi), Kannada(ka) and English(en-US), then we will be creating 4 assemblies for all deployements -

## ****Primary assembly****

Containing all the C# code and logic. Primary assembly should always be kept culture neutral. Default Language (fall back) resources are always present in the primary assembly. Your assemblyInfo.cs file should have following declaration to make it culture-neutral assembly:

**[assembly: AssemblyCulture("")]**

* One satellite assembly for "ta" culture.
* One satellite assembly for "hi" culture.
* One satellite assembly for "ka" culture.

# References:

1. Cultures: <https://zetcode.com/csharp/cultureinfo/>
2. Cultures: <https://csharp.net-tutorials.com/working-with-culture-and-regions/the-cultureinfo-class/>
3. Satellite Assembly: <https://www.codeproject.com/Articles/801343/All-you-wanted-to-know-about-Satellite-Assemblies>