

## 第15期-本地化与全球化的多语言实现

2021年2月8日 12:32

### 国际化涉及到全球化与本地化

全球化是设计支持不同区域性的应用程序的过程。全球化添加了对一组有关特定地理区域的已定义语言脚本的输入、显示和输出支持。

本地化是将已经针对可本地化性进行处理的全球化应用调整为特定的区域性设置的过程。

- 全球化 (G11N)：使应用支持不同语言和区域的过程。
- 本地化 (L10N)：针对给定语言和区域自定义应用的过程。
- 国际化 (I18N)：介绍了全球化和本地化。
- 区域性：它是一种语言和区域（可选）。
- 非特定区域性：具有指定语言但不具有区域的区域性。（例如，“en”，“zh”）
- 特定区域性：具有指定语言和区域的区域性。（例如，“en-US”，“zh-CN”）
- 父区域性：包含特定区域性的非特定区域性。（例如，“en”是“en-US”的父区域性）
- 区域设置：区域设置与区域性相同。

### I18N、L10N、G11N的区别？

I18N - - 是“Internationalization”的缩写，由于“Internationalization”单词较长，所以为了书写简便，通常缩写为“I18N”。中间的18代表在首字母“I”和尾字母“N”之间省略了18个字母。单词“Internationalization”翻译成中文是“国际化”的意思 - 是使产品或软件具有不同国际市场的普遍适应性，从而无需重新设计就可适应多种语言和文化习俗的过程。真正的国际化要在软件设计和文档开发过程中，使产品或软件的功能和代码设计能处理多种语言和文化习俗，具有良好的本地化能力。

G11N - - 是“Globalization”的缩写，由于“Globalization”单词较长，所以为了书写简便，通常缩写为“G11N”，中间的11代表在首字母“G”和尾字母“N”之间省略了11个字母。单词“Globalization”翻译成中文是“全球化”的意思 - 是使产品或软件进入全球市场而进行的有关的商务活动。包括正确的国际化设计，本地化集成，以及在全球市场进行的市场推广、销售和支持的全部过程。企业通过全球化实现其全球化发展战略，实现全球化业务，扩大市场规模，降低软件成本，提升综合竞争力，展现企业发展实力，增强用户信心，树立市场形象。

L10N - - 是“Localization”的缩写，由于“Localization”单词较长，所以为了书写简便，通常缩写为“L10N”，中间的10代表在首字母“L”和尾字母“N”之间省略了10个字母。单词“Localization”翻译成中文是“本地化”的意思，是将产品或软件针对特定国际语言和文化进行加工，使之符合特定区域市场的过程。真正的本地化要考虑目标区域市场的语言、文化、习俗、特征和标准。通常包括改变软件的书写系统（输入法）、键盘使用、字体、日期、时间和货币格式等。

### 语言代码规范

<https://www.iso.org/iso-639-language-codes.html>

## 多语言 .NET 代码演示

```
foreach (CultureInfo ci in CultureInfo.GetCultures(CultureTypes.AllCultures))
{
    Console.WriteLine("{0,-7} {1,-30}", ci.Name, ci.EnglishName);
}
```

```
Console.WriteLine(Thread.CurrentThread.CurrentCulture);
Console.WriteLine(DateTimeOffset.Now.ToString());
Console.WriteLine(88.ToString("C"));
```

```
Thread.CurrentThread.CurrentCulture = CultureInfo.CreateSpecificCulture("en-US");
Console.WriteLine(Thread.CurrentThread.CurrentCulture);
Console.WriteLine(DateTimeOffset.Now.ToString());
Console.WriteLine(88.ToString("C"));
```

## ASP.NET Core 全球化与本地化

<https://docs.microsoft.com/zh-cn/aspnet/core/fundamentals/localization>

第19期 全球化&本地化&多语言

第20期 全球化&本地化&多语言

```
services.AddLocalization(options => options.ResourcesPath = "Resources");
```

## 多语言切换

```
string[] supportedCultures = new[] { "zh-CN", "en-US" };
RequestLocalizationOptions localizationOptions = new RequestLocalizationOptions
{
    ApplyCurrentCultureToResponseHeaders = false
};
localizationOptions.SetDefaultCulture(supportedCultures.First()).AddSupportedCultures(supportedCultures).AddSupportedUICultures(supportedCultures);
app.UseRequestLocalization(localizationOptions);

app.UseTenantMiddleware();
```

## 数据注解验证消息多语言

```
mvcBuilder.AddDataAnnotationsLocalization(options =>
options.DataAnnotationLocalizerProvider = (type, factory) =>
factory.Create(type));
```