**简介**

任何应用至少有一种用户界面语言，很多应用有多种，ABP为一个应用提供了一个灵活的本地化系统。

**应用语言**

第一件事就是声明支持哪些语言，这在你的[模块](http://www.cnblogs.com/kid1412/p/5973274.html" \t "http://www.cnblogs.com/kid1412/p/_blank)的PreInitialize方法里完成，如下所示：

Configuration.Localization.Languages.Add(new LanguageInfo("en", "English", "famfamfam-flag-england", true));

Configuration.Localization.Languages.Add(new LanguageInfo("tr", "Türkçe", "famfamfam-flag-tr"));

**本地化源**

一个本地化源应当实现ILocalizationSource接口，然后它被注册到ABP本地化配置里。

**XML文件**

本地化文本可以存储在XML文件里，XML文件内容类似于如下所示：

<?xml version="1.0" encoding="utf-8" ?>

<localizationDictionary **culture="en"**>

<texts>

<text name="TaskSystem" value="Task System" />

<text name="TaskList" value="Task List" />

<text name="NewTask" value="New Task" />

<text name="Xtasks" value="{0} tasks" />

<text name="CompletedTasks" value="Completed tasks" />

<text name="EmailWelcomeMessage">Hi,

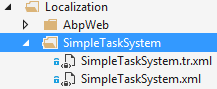
Welcome to Simple Task System! This is a sample

email content.</text>

</texts>

</localizationDictionary>

XML文件必须包含编码（utf-8），culture=”en“表明这个XML文件包含英文文本



SimpleTaskSystem是源名称，SimpleTaskSystem.xml定义当前语言，当一个文本被请求，ABP从当前语言的XML文件里获取文本（使用Thread.CurrentThread查找当前语言），如果文本不存在，它将从默认语言的XML文件里查找。

**注册XML本地化源**

XML文件可存储于文件系统里或嵌入到一个程序集里。

如果XML文件存储于文件系统，我们可以用如下方式注册一个XML本地化源：

Configuration.Localization.Sources.Add(

new DictionaryBasedLocalizationSource(

"SimpleTaskSystem",

new **XmlFileLocalizationDictionaryProvider**(

HttpContext.Current.Server.MapPath("~/Localization/SimpleTaskSystem")

)

)

);

这在一个模块的PreInitialize事件里完成（更多信息查看[模块系统](http://www.cnblogs.com/kid1412/p/5973274.html" \t "http://www.cnblogs.com/kid1412/p/_blank)），ABP从指定文件夹里找到所有XML文件并把它们注册为本地化源。

如果XML嵌入到一个程序集里，我们应当把所有本地化XML文件标记为嵌入的资源，然后我们可以用如下的方式注册这个本地化源：

Configuration.Localization.Sources.Add(

new DictionaryBasedLocalizationSource(

"SimpleTaskSystem",

new **XmlEmbeddedFileLocalizationDictionaryProvider**(

Assembly.GetExecutingAssembly(),

"MyCompany.MyProject.Localization.Sources"

)

)

);

XmlEmbeddedFileLocalizationDictionaryProvider获取一个程序集包含的xml文件

注意：当给嵌入的XML文件添加后缀时，不要使用.号，如“MySource.tr.xml"，应当用短横，如”MySource-tr.xml“，因为当查找资源时.号会引起命名空间的问题。

**JSON文件**

JSON文件可以用来存储一个本地化源的文本，下面是一个JSON本地化文件的示例：

{

**"culture": "en",**

"texts": {

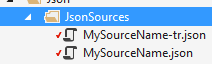
"TaskSystem": "Task system",

"Xtasks": "{0} tasks"

}

}

JSON文件应当编码（utf-8），culture：“en“表明这个JSON文件包含英文文本，我们可以为每个语言创建单独的JSON文件，如下所示：



MySourceName是源名称，MySourceName.json定义了默认语言，这类似于XML文件。

**注册JSON本地化源**

JSON文件可存储在文件系统里或是嵌入到一个程序集中。

如果存储在文件系统里，我们可以用以下方式注册它：

Configuration.Localization.Sources.Add(

new DictionaryBasedLocalizationSource(

"MySourceName",

new **JsonFileLocalizationDictionaryProvider**(

HttpContext.Current.Server.MapPath("~/Localization/MySourceName")

)

)

);

这在一个模块的PreInitialize事件里完成（更多信息查看[模块系统](http://www.cnblogs.com/kid1412/p/5973274.html" \t "http://www.cnblogs.com/kid1412/p/_blank)）

如果JSON嵌入到一个程序集里，我们应当把所有本地化JSON文件标记为嵌入的资源，然后我们可以用如下的方式注册这个本地化源：

 Configuration.Localization.Sources.Add(

new DictionaryBasedLocalizationSource(

"MySourceName",

new **JsonEmbeddedFileLocalizationDictionaryProvider**(

Assembly.GetExecutingAssembly(),

"MyCompany.MyProject.Localization.Sources"

)

)

);

**获取一个本地文本**

在创建一个源并把它注册到ABP本地化系统后，就可以很简单的本地化文本了。

**在服务端**

在服务端，我们可以[注入](http://www.cnblogs.com/kid1412/p/5980068.html" \t "http://www.cnblogs.com/kid1412/p/_blank)ILocalizationManager并使用它的GetString方法：

var s1 = \_localizationManager.GetString("SimpleTaskSystem", "NewTask");

为不一直重复获取源名称，你可以先获取源，然后从源里获取一个字符串：

var source = \_localizationManager.GetSource("SimpleTaskSystem");

var s1 = source.GetString("NewTask");

**在Mvc控制器里**

在Mvc控制器和视图里通常需要本地化文本，这里有一个快捷的方法，如下示例控制器所示：

public class HomeController : **SimpleTaskSystemControllerBase**

{

public ActionResult Index()

{

var helloWorldText = L("HelloWorld");

return View();

}

}

用L方法本地化一个字符串，当然必须先提供一个源名称，可以在如下所示的SimpleTaskSystemControllerBase里提供：

public abstract class SimpleTaskSystemControllerBase : **AbpController**

{

protected SimpleTaskSystemControllerBase()

{

**LocalizationSourceName = "SimpleTaskSystem";**

}

}

**在Mvc视图里**

同样，L方法也存在于视图中：

<div>

<form id="NewTaskForm" role="form">

<div class="form-group">

<label for="TaskDescription">**@L("TaskDescription")**</label>

<textarea id="TaskDescription" data-bind="value: task.description" class="form-control" rows="3" placeholder="**@L("EnterDescriptionHere")**" required></textarea>

</div>

<div class="form-group">

<label for="TaskAssignedPerson">**@L("AssignTo")**</label>

<select id="TaskAssignedPerson" data-bind="options: people, optionsText: 'name', optionsValue: 'id', value: task.assignedPersonId, optionsCaption: '**@L("SelectPerson")**'" class="form-control"></select>

</div>

<button data-bind="click: saveTask" type="submit" class="btn btn-primary">**@L("CreateTheTask")**</button>

</form>

</div>