### 如何启用

#### 启动模板

如果您从[启动模板](https://aspnetboilerplate.com/Templates)创建项目，则可以跳过此部分，因为模板默认启用了基于数据库的本地化。如果您在此功能之前创建了项目，请阅读此内容以便为您的应用程序启用它。

#### EnableDbLocalization

首先，我们启用它：

Configuration.Modules.Zero().LanguageManagement.EnableDbLocalization();

这应该在**顶级**模块的 [PreInitialize](https://aspnetboilerplate.com/Pages/Documents/Module-System" \l "preinitialize) 方法中完成（它是Web应用程序的Web模块。

### 管理语言

**IApplicationLanguageManager**接口用于管理语言。它具有GetLanguagesAsync，AddAsync，RemoveAsync，UpdateAsync等方法来管理主机和租户的语言。

#### ApplicationLanguage实体

ApplicationLanguage实体表示租户或主机的语言。

[Serializable]

[Table("AbpLanguages")]

public class ApplicationLanguage : FullAuditedEntity, IMayHaveTenant

{

//...

}

它的基本属性是：

* **TenantId**（可空）：如果此语言是特定于租户的，则包含相关租户的ID。如果这是一种宿主语言，则为null。
* **Name**：语言名称。这**必须是一个文化代码**从 [这个名单](https://msdn.microsoft.com/en-us/library/ee825488(v=cs.20).aspx)。
* **DisplayName**：语言的显示名称。这可以是任意名称，但通常是 [CultureInfo.DisplayName](https://msdn.microsoft.com/en-us/library/system.globalization.cultureinfo.displayname(v=vs.110).aspx)。
* **Icon**：语言的任意图标/标志。这可以用于在UI上显示语言的标志。

### 管理本地化文本

**IApplicationLanguageTextManager**用于管理本地化的文本。它具有获取/设置租户或主机的本地化文本所需的方法。

#### ApplicationLanguageText实体

ApplicationLanguageText实体用于在数据库中存储本地化值。

[Serializable]

[Table("AbpLanguageTexts")]

public class ApplicationLanguageText : AuditedEntity<long>, IMayHaveTenant

{

//...

}

它的基本属性是;

* **TenantId**（可空）：如果此本地化文本是特定于租户的，则包含相关租户的ID。如果这是主机本地化文本，则为null。
* **LanguageName**：**语言的**名称。这**必须是一个文化代码**从[这个名单](https://msdn.microsoft.com/en-us/library/ee825488(v=cs.20).aspx)。这与ApplicationLanguage.Name匹配，但不是强制外键使其独立于语言条目。IApplicationLanguageTextManager正确处理它。
* **Source**：本地化源名称。
* **Key**：本地化文本的键/名称。
* **Value**：本地化价值。