Qt 语言学家手册：基于文本 ID 的翻译

文本ID翻译机制是国际化和本地化的"产业实力"体系。应用程序中的每个文本都分配有一个唯一标识符（文本 ID），这些标识符直接在源代码中用于代替纯文本。这需要用户界面开发人员做更多的工作，但可以更轻松地管理大量翻译文本。

> **注意：** 在一个应用程序中，必须仅使用基于纯文本或仅使用基于文本 ID 的函数。如果将它们混合在一起，您最终会得到一组不完整的文本需要翻译。

使用文本 ID 进行国际化

当使用文本 ID 而不是纯文本时，国际化应用程序的一般方法是相同的，但细节略有不同：

1. 基于文本 ID 的翻译系统的函数和宏与纯文本系统不同。您可以使用函数代替 qsTr（），使用宏而不是QT_TR_NOOP（），使用宏而不是QT_TR_N_NOOP（））。qsTrId()QT_TRID_NOOP()QT_TRID_N_NOOP()
2. 使用文本 ID 作为用户界面字符串，而不是纯文本字符串。例如qsTrId("id-back-not-front")
3. 不能指定具有文本 ID 的上下文参数。如果存在具有不同含义的相同拼写的单词，则需要单独的文本 ID。例如，将区分后退步骤"后退"和对象后退"后退"。qsTrId("id-back-backstep")
4. 您在开发版本的用户界面中看到的"工程英语"文本用注释表示。如果未包含此信息，则文本 ID 将显示在用户界面中。当您有带参数的文本时，这一点尤其重要。注释需要在字符串中包含参数指标。例如//%//%//% "Number of files: %1"
5. 向翻译人员提供额外信息的注释在纯文本系统中是可选的。但是，对于基于文本ID的系统，这些额外的信息变得至关重要，因为没有它，您只有文本ID，翻译人员可能无法在没有进一步上下文的情况下从中进行合理的翻译。您可以使用长描述性文本 ID，不使用注释，但注释通常更易于理解。//:

下面的并排代码段显示了基于文本 ID 的翻译和基于纯文本的翻译的比较：

基于文本 ID	基于纯文本

`Text {`
`  id: backTxt;`
`  //: The back of the object, not the front`
`  //% "Back"`
`  //~ Context Not related to back-stepping`
`  text: qsTrId("id-back-not-front");`
`}`
	`Text {`
	`  id: backTxt;`
	`  //: The back of the object, not the front`
	`  //~ Context Not related to back-stepping`
	`  text: qsTr("Back","Not front")`
	`}`

使用文本 ID 进行本地化

使用文本 ID 进行本地化的过程与纯文本的过程大致相同。

```
lupdate <myapp>.pro
```

请注意，翻译文件中的源值将是文本 ID，而不是纯文本。这意味着您需要非常具有描述性的文本 ID 或良好的附加注释，或两者兼而有之，以确保译员进行正确的翻译。

上面基于文本 ID 的示例用户界面文本在 .ts 文件中产生以下内容：

```
<message id="id-back-not-front">
    <source>Back</source>
    <extracomment>The back of the object, not the front</extracomment>
    <translation type="unfinished"></translation>
    <extra-Context>Not related to back-stepping</extra-Context>
</message>
```

使用 时，需要指定已翻译文本的键基于文本 ID，而不是纯文本。如果指定了代码中的字符串，则未设置“id”属性，因此 它们将被 忽略。`lreleaseqsTr()lrelease`

此命令为您的应用程序生成所有已编译的翻译 .qm 文件：

```
lrelease -idbased <myapp>.pro
```

但是，如果给定文本没有可用的翻译（通常这种情况直到开发后期），则文本 ID 将显示在用户界面中，而不是正确的文本。为了使应用程序更易于测试，您可以使用“工程英语”源文本（来自注释）作为翻译文本，并用一些指示器标记它，以便您可以看到尚未翻译的文本。`lrelease//%`

例如，以下命令生成 .qm 文件，并在未翻译的文本前面放置一个 “！ ”：

```
lrelease -idbased -markuntranslated ! <myapp>.pro
```

Advanced Usage

For projects that target a large number of locales, you can remove the TRANSLATIONS info from the .pro file and, instead, manage the translations with a separate script. The script can call lrelease and lupdate for each of the desired targets.

The updates could be scripted something like this:

```
lupdate -recursive <project-dir> -ts <project-dir>/i18n/myapp-text_en_GB.ts
lupdate -recursive <project-dir> -ts <project-dir>/i18n/myapp-text_en_US.ts
...
```

The generation of the final .qm files could be scripted something like this:

```
lrelease -idbased <project-dir>/i18n/myapp-text_en_GB.ts
lrelease -idbased <project-dir>/i18n/myapp-text_en_US.ts
...
```

Qt DOCUMENTATION