React Native 国际化指南

目录

- Extract strings only used in the native platform
  - NPM 命令
- 提供自定义翻译（仅限原生平台使用的字符串）
- 获取翻译（用于网页和原生平台的字符串）
  - NPM 命令

编辑器的原生版本引用两种类型的字符串：

1. 在网页和原生平台中使用的字符串。
2. 仅用于原生平台的字符串。

对于第一种类型，这些字符串的翻译遵循 本指南
中为网页版本描述的相同流程；然而对于后者，需要您自行提供翻译。

Extract strings only used in the native platform

In order to identify these strings, you can use the script `extract-used-strings` located in `packages/react-native-editor/bin/extract-used-strings.js` to generate a JSON object that contains all the strings referenced including the platforms where they are used, as well as the files that reference it. Here you can see the format:

```
 {
□"gutenberg": {
□  "<string>": {
□    "string": String value.
□□"stringPlural": String value with its plural form. [optional]
□□"comments": Comments for translators. [default value is an empty string]
□□"reference": Array containing the paths of the source files that reference the string.
□□"platforms": Array containing the platforms where the string is being used, values are
```

```
"android" | "ios" | "web".
  },
  ...
},
"other-domain-plugin": {
  ...
},
    ...
}
```

This command also supports passing extra plugins, in case the React Native bundle of the editor is generated including other plugins.

It's important to note that the JSON object contains all used strings, so in order to identify the ones only used in the native platform, it’s required to provide your own script/process to extract them. This can easily be done by going through the strings and filtering out the ones that include the "web" platform.

NPM 命令

提取已使用的字符串:

```
npm run native i18n:extract-used-strings -- "$PWD/used-strings.json"
```

注意: 我们需要传递绝对路径，否则它会将 `packages/react-native-editor` 作为相对路径的根路径。

提取已使用的字符串（包括额外插件）:

```
npm run native i18n:extract-used-strings -- "$PWD/used-strings.json" "domain-plugin-1"
<PLUGIN-1_SOURCE_PATH> "domain-plugin-2" <PLUGIN-2_SOURCE_PATH> ...
```

提供自定义翻译（仅限原生平台使用的字符串）

获取原生平台使用的字符串列表后，这些字符串需要被翻译，但此过程不在原生版本的范畴内，因此您需要提供自己的翻译。

将翻译数据注入编辑器的方式是通过初始化时传递给编辑器的 `translations` 初始属性:

- Android 参考
- iOS 参考

通过上述 `translations` 初始属性将提供的翻译集成到移动客户端的具体机制在此不作描述，因为这取决于移动客户端的具体实现，且可通过不同方式实现。但重要的是，这些翻译必须通过上述初始属性提供，因为编辑器负责将它们与已包含的翻译合并。

注意：请记住，与编辑器中已有字符串匹配的那些字符串将被覆盖。

获取翻译（用于网页和原生平台的字符串）

翻译文件本质上是一个 JSON 对象，包含每个字符串翻译的键值对。这些内容从 translate.wordpress.org 获取，该网站托管了 WordPress 以及一系列插件（如 Gutenberg）的翻译。

这些文件可以缓存在一个文件夹下并进行优化。此外，会生成一个索引文件，作为导入和获取插件翻译的入口点。

获取的翻译包含所有可翻译的插件字符串，包括编辑器原生版本中未使用的字符串。然而，可以通过过滤掉已使用字符串 JSON 文件中未引用的字符串来减小文件大小。

默认情况下，在安装依赖项时，如果缓存不存在，可能会下载未优化的 Gutenberg 翻译，并位于 `i18n-cache` 文件夹中。

这些翻译文件中包含的字符串会在编辑器初始化时导入（参考），并与 `translations` 初始属性提供的额外翻译合并。

NPM 命令

获取未优化的翻译：

```
npm run native i18n:fetch-translations -- "gutenberg" <OUTPUT_PATH>
```

注意：我们需要传递绝对路径，否则它会将 *`packages/react-native-editor`* 作为相对路径的根路径。

获取优化后的翻译：

```
npm run native i18n:fetch-translations -- "gutenberg" <OUTPUT_PATH> <USED_STRINGS_FILE>
```

版本号 #2

由 壳壳虫 创建于 2026-03-27 01:08:43 UTC

由 壳壳虫 更新于 2026-03-27 01:08:43 UTC