Index

目录

- 国际化安全
- 检查垃圾信息和恶意字符串
- 转义国际化字符串
- 对 URL 使用占位符
- 自行编译 .mo 二进制文件

国际化安全

在讨论国际化时，安全问题常被忽视，但有几个重要事项需要牢记。

检查垃圾信息和恶意字符串

当译者向您提交本地化内容时，务必检查确认其翻译中未包含垃圾信息或其他恶意词汇。您可以使用 Google 翻译 将译文回译至您的母语，以便轻松比对原文与翻译后的字符串。

转义国际化字符串

不能假设译者仅会在本地化中添加无害文本；若其有意为之，可能会添加恶意 JavaScript 或其他代码。为防止此类情况，必须像对待其他不可信输入一样处理国际化字符串。

若需输出这些字符串，则应进行转义处理。

不安全示例

```
_e( 'The REST API content endpoints were added in WordPress 4.7.', 'your-text-domain' );
```

安全示例

```
esc_html_e( 'The REST API content endpoints were added in WordPress 4.7.', 'your-text-domain'
);
```

另一种方案是选择依赖翻译验证机制，而非在代码中添加转义。验证机制的典型范例是 WordPress 多语言团队在 translate.wordpress.org 中使用的 编辑者角色。该机制确保任何由不可信贡献者提交的翻译，在获准前均经过可信编辑者的验证。

对 URL 使用占位符

请勿在国际化字符串中包含 URL，因为恶意译者可能将其更改为指向其他网址。应使用 `printf()` 或 `sprintf()` 的占位符替代。

不安全示例

```
_e(
  'Please <a href="https://login.wordpress.org/register"> register for a WordPress.org
account</a>.',
  'your-text-domain'
);
```

安全示例

```
printf(
  esc_html__( 'Please %1$s register for a WordPress.org account %2$s.', 'your-text-domain' ),
  '<a href="https://login.wordpress.org/register">',
  '</a>'
);
```

自行编译 .mo 二进制文件

译者通常会同时提交编译后的 .mo 文件与纯文本 .po 文件，但您应弃用其提供的 .mo 文件并自行编译。因为无法确认该文件是否由对应的 .po 文件编译而来，抑或是基于其他文件编译。若基于不同文件编译，则可能在不被察觉的情况下包含垃圾信息或恶意字符串。

使用 PoEdit 生成二进制文件会覆盖 .po 文件中的头部信息，因此更推荐通过命令行编译：

```
msgfmt -cv -o /path/to/output.mo /path/to/input.po
```

□□□ #2
□ □□□ □□□ 2026-03-27 01:08:39 UTC
□ □□□ □□□ 2026-03-27 01:08:39 UTC