Internationalization

- What is internationalization?
- Why is internationalization important?
- The basics
- Setting up your plugin and theme to i18n
  - Internationalizing JavaScript
    - Internationalizing JavaScript before WP 5.0
- Internationalization Guidelines

What is internationalization?

Internationalization is the process of developing your application in a way it can easily be translated into other languages. Internationalization is often abbreviated as `i18n` (because there are 18 letters between the letters i and n).

Why is internationalization important?

WordPress is used all over the world, by people who speak many different languages. The strings in WordPress need to be coded in a special way so that they can be easily translated into other languages. As a developer, you may not be able to provide localization for all your users; however, a translator can successfully localize your code without needing to modify the source code itself.

While making your code translatable is called Internationalization, the act of translating it and adapting the strings to a specific location is called Localization. Read more about Localization in WordPress.

The basics

In order to make a string translatable, you have to wrap the original strings in a call to one of the WordPress i18n functions.

For example, the PHP function _e() echoes a translatable string:

```
_e('Edit post');
```

You will find code like this all over WordPress core files. However, if you are internationalizing a theme or a plugin, there is another argument that all i18n functions take called Text Domain.

Text Domains set the domain your plugin or theme translations belong. This assures there is no conflict between strings in plugins, themes, and the WordPress core.

With a text-domain, the most basic call to a i18n function that will output a string would be like:

```
_e('Edit movie', 'my-plugin');
```

Setting up your plugin and theme to i18n

Setting up i18n is slightly different for Plugins and Themes, therefore this information is detailed in each respective Handbook:

- How to internationalize your theme
- How to internationalize your plugin

Internationalizing JavaScript

Since WordPress 5.0 it’s possible to internationalize JavaScript files using the same set of i18n functions used in PHP.

In order to be able to use i18n functions in your JavaScript code, you have to declare `wp-i18n` as a dependency on your script when registering or enqueueing it. For example:

```
wp_register_script(
     'my-script',
     plugins_url( 'js/my-script.js', FILE ),
     array( 'wp-i18n' ),
     '0.0.1'
 );
```

Now that you have added the dependency to your script, you can use i18n functions in it, however you still have to tell WordPress to load the translations.

You do this by calling `wp_set_script_translations()`. This function takes three arguments: the registered/enqueued script handle, the text domain, and optionally a path to the directory containing translation files. The latter is only needed if your plugin or theme is not hosted on WordPress.org, which provides these translation files automatically.

```
wp_set_script_translations('my-script', 'my-plugin');
```

For better performance, always make sure to enqueue your scripts and load their translations only when they are really used.

In your JavaScript code you will use i18n functions pretty much the same way you do in your PHP code:

```
const { __, _x, _n, sprintf } = wp.i18n;

// simple string
__( 'Hello World', 'my-plugin' );

// string with context
_x( 'My Gutenberg block name', 'block name', 'my-plugin' );
```

The available i18n for you to use in your JS code are (See internationalization functions for more details):

- __()
- _x()
- _n()
- _nx()
- sprintf()

Notice that the wp-i18n package also includes the `sprintf` function. This is very useful to internationalize strings that have variables in it.

Now refer to the Internationalization Guidelines to learn how to use all these functions and the best practices on writing your strings.

If you are not hosting your plugin or theme on WordPress.org, you will need to create your translation files yourself. Check this post out to learn how to do this.

Internationalizing JavaScript before WP 5.0

Another way to internationalize your JavaScript files is to use the wp_localize_script() function.

With this function you can register translatable strings and have them available in your JavaScript to be used.

Please refer to the `wp_localize_script` () reference to learn how to use it.

Internationalization Guidelines

Now that you are ready to internationalize your application, read through the Internationalization Guidelines and learn what each i18n function is for, how to use them, and the best practices when writing your strings.

□□□ #2
□ □□□ □□□ 2026-10-31 18:07:11 UTC
□ □□□ □□□ 2026-10-31 18:07:12 UTC