?????????? ????????? ????????? ????????? ?? ?????? ????? ??? wordpress ? ????? ? ? ?????? ? React

Ниже — **полная архитектура переводов в WordPress**:

1️⃣ как это работает **в целом**
2️⃣ как это работает **в PHP**
3️⃣ как это работает **в React / Gutenberg**
4️⃣ как соединяются **PHP + JS переводы**
5️⃣ как это обычно делают профессиональные плагины.

---

1. ????? ??????????? ????????? ? WordPress

Система переводов в WordPress состоит из 4 уровней:

```
код → POT → PO → MO / JSON
```

1?? ???

В коде используются функции:

PHP:

```
__()
_e()
_n()
_x()
```

JS:

```
wp.i18n.__()
wp.i18n._n()
wp.i18n._x()
```

2?? POT (template)

.pot — шаблон переводов.

Пример:

```
nv-seo.pot
```

Он содержит **все строки из кода**.

3?? PO

Файл конкретного языка:

```
ru_RU.po
de_DE.po
fr_FR.po
```

Пример:

```
msgid "Save settings"
msgstr "Сохранить настройки"
```

4?? MO / JSON

WordPress использует:

тип	для
MO	PHP
JSON	JavaScript

---

2. ??????????? ????????? ??? PHP

????????? ???????

```
plugin
 ├─ languages
 │   ├─ nv-seo.pot
 │   ├─ nv-seo-ru_RU.po
 │   ├─ nv-seo-ru_RU.mo
 │
 ├─ nv-seo.php
```

??????????? textdomain

```
add_action('plugins_loaded', function () {

    load_plugin_textdomain(
        'nv-seo',
        false,
        dirname(plugin_basename(__FILE__)) . '/languages'
    );

});
```

????????????

```
__('Settings saved', 'nv-seo')
```

или

```
echo __('Save', 'nv-seo');
```

---

3. ??????????? ????????? ??? React / JS

В React используется пакет:

```
@wordpress/i18n
```

(глобально это `wp.i18n`)

????????????

```
import { __ } from '@wordpress/i18n';

__('Save settings', 'nv-seo')
```

?????? React

```
import { __ } from '@wordpress/i18n';
import { Button } from '@wordpress/components';

export default function SaveButton() {

    return (
        <Button variant="primary">
            {__('Save settings', 'nv-seo')}
        </Button>
    );

}
```

4. ??? WordPress ????????? JS

WordPress **не использует MO для JS**.

Он использует:

```
JSON
```

пример:

```
languages/nv-seo-ru_RU-3a4f2.json
```

5. ?????? PHP ? JS

Очень важная функция:

```
wp_set_script_translations()
```

Она говорит WordPress:

> "подключи JSON переводы для этого JS файла"

??????

```
wp_enqueue_script(
    'nv-seo-admin',
    plugins_url('build/index.js', __FILE__),
    ['wp-i18n', 'wp-element', 'wp-components'],
    '1.0',
    true
);

wp_set_script_translations(
    'nv-seo-admin',
    'nv-seo',
    plugin_dir_path(__FILE__) . 'languages'
);
```

---

6. ??? ??????????? JSON

WordPress CLI генерирует его из PO.

```
wp i18n make-json languages
```

Результат:

```
nv-seo-ru_RU-xxxx.json
```

---

7. ?????? ????????? ???????????? ???????

```
nv-seo
|
├─ src
|   └─ index.js
|
├─ build
```

```
│   └─ index.js
│
├─ languages
│   ├─ nv-seo.pot
│   ├─ nv-seo-ru_RU.po
│   ├─ nv-seo-ru_RU.mo
│   ├─ nv-seo-ru_RU-xxxx.json
│
├─ nv-seo.php
└─ package.json
```

---

8. ????? ???????????

1?? ????? ???

PHP:

```
__('Save settings', 'nv-seo')
```

JS:

```
__('Save settings', 'nv-seo')
```

2?? ?????????? POT

```
wp i18n make-pot . languages/nv-seo.pot
```

3?? ??????? ????????

```
ru_RU.po
```

через:

- Poedit
- Loco Translate

4?? ???????????

PO → MO

5?? ?????????? JSON

```
wp i18n make-json languages
```

9. ??? ?????? ??????? ???????

Например:

- Yoast SEO
- Rank Math
- WooCommerce

используют:

```
languages/
build/
src/
```

и **WordPress i18n CLI**.

10. ?????? ????????

?????? ?????????? textdomain

```
__('Save', 'nv-seo')
```

?? ??????????? ???????????? ??????

 плохо

```
__(text, 'nv-seo')
```

??????????? sprintf

```
sprintf(
    __('Found %d errors', 'nv-seo'),
    count
)
```

?????????? ????????

```
_x('Post', 'noun', 'nv-seo')
```

11. ????? ?????? ???????????? "Code splitting"

Если ваш интерфейс React большой, лучше:

```
code splitting
```

тогда JSON будет загружаться **по частям**.

Code splitting — это разбиение большого JavaScript-бандла на несколько файлов, которые загружаются **только тогда, когда они реально нужны**.
В WordPress это особенно полезно для **React-интерфейсов в админке**, чтобы не грузить весь JS сразу.

В сборке через **@wordpress/scripts** code splitting уже поддерживается, потому что внутри используется **Webpack**.

1. ???????? ????

Без code splitting:

```
index.js
   ↓
build/index.js (400 KB)
```

С code splitting:

```
index.js
   ↓
build/index.js
build/settings.js
build/schema.js
build/faq.js
```

Подгружается только нужный модуль.

2. ??? ??? ???????? ? React

Используется **dynamic import()**.

??????? ??????

```
import SettingsPage from './settings';
```

всё попадает в один файл.

---

dynamic import

```
const SettingsPage = import('./settings');
```

Webpack создаст отдельный chunk.

Но в React правильнее использовать **React.lazy**.

---

3. ?????? code splitting

?????????

```
src
 ├─ index.js
 ├─ pages
 │   ├─ dashboard.js
 │   ├─ schema.js
 │   └─ settings.js
```

---

index.js

```
import { lazy, Suspense } from '@wordpress/element';
import { Spinner } from '@wordpress/components';

const Dashboard = lazy(() => import('./pages/dashboard'));
const Settings = lazy(() => import('./pages/settings'));

export default function App() {

    return (
        <Suspense fallback={<Spinner />}>
            <Dashboard />
        </Suspense>
```

```
    );

}
```

4. ??? ??????? ??????

После

```
npm run build
```

в папке **build** будет:

```
build
 ├─ index.js
 ├─ dashboard.js
 ├─ settings.js
```

5. ??? WordPress ????????? chunks

Важный момент.

WordPress должен знать путь:

```
wp_enqueue_script(
    'nv-seo-admin',
    plugins_url('build/index.js', __FILE__),
    ['wp-element','wp-components','wp-i18n'],
    '1.0',
    true
);
```

Webpack автоматически будет подгружать:

```
build/dashboard.js
build/settings.js
```

через

```
import()
```

6. ??? ??? ???????? ? ????????

Без splitting:

```
index.js 400KB
```

Со splitting:

```
index.js 80KB
settings.js 100KB
schema.js 120KB
faq.js 100KB
```

---

7. ??? ??? ???????? ???????

Например в SEO-плагине можно разделить:

```
src
 ├─ index.js
 ├─ pages
 │   ├─ dashboard.js
 │   ├─ schema.js
 │   ├─ sitemap.js
 │   └─ redirects.js
```

Пользователь открывает **Schema** → грузится только:

```
schema.js
```

---

8. ?????? ? ?????????

Если используется **React Router:

```
const Settings = lazy(() => import('./pages/settings'));
const Schema = lazy(() => import('./pages/schema'));
```

---

9. ????? ???????? ???????????

Можно разбить даже **библиотеки**.

Например:

```
schema-builder.js
```

грузить только при редактировании Schema.

---

10. ?????? ?????? ??? WordPress

Чтобы chunks работали, **build папка должна содержать все файлы**:

```
build
 ├─ index.js
 ├─ index.asset.php
 ├─ 234.js
 ├─ 567.js
```

WordPress автоматически подхватывает зависимости из:

```
index.asset.php
```

---

11. ??????????? ????? asset

Правильный способ:

```
$asset = include plugin_dir_path(__FILE__) . 'build/index.asset.php';

wp_enqueue_script(
    'nv-seo-admin',
    plugins_url('build/index.js', __FILE__),
    $asset['dependencies'],
    $asset['version'],
    true
);
```

---

12. ???? ?? "Code splitting"

Code splitting позволяет:

✔ ускорить загрузку
✔ уменьшить первый бандл
✔ грузить код **по требованию**

Это стандартная практика для:

- Yoast SEO
- WooCommerce
- Rank Math

---

12. ???????? ???????? ????????

Они думают:

> JSON перевод нужно писать вручную.

Нет.

Он **генерируется автоматически**.

---

13. ????? ??????????? ???????????? ??? ?????? SEO-???????

```
nv-seo
 ├─ src
 │   └─ admin
 │       └─ index.js
 │
 ├─ build
 │   └─ admin.js
 │
 ├─ languages
 │   ├─ nv-seo.pot
 │   ├─ nv-seo-ru_RU.po
 │   ├─ nv-seo-ru_RU.mo
 │   ├─ nv-seo-ru_RU.json
 │
 └─ nv-seo.php
```

Revision #3
Created 2026-03-05 05:39:19 UTC by Nikolay
Updated 2026-03-05 06:01:20 UTC by Nikolay