

WordPress: Using Non-Default URIs

WordPress Meetup Kyiv

Oleg Butuzov, 2018

Про що розмова?

**Змінюємо поведінку WordPress
впливаючи на URI**

Обробка HTTP запиту з WordPress

Клієнт робить запит.

```
GET /tag/wp/page/2 HTTP/1.1
```

```
Host: example.com
```

DNS каже клієнту кудя йому піти

```
$ dig a example.com  
> ;; ANSWER SECTION:  
> example.com.      6244      IN      A      93.184.216.34
```

webserver за місцем призначення
віддає запит на обробку хосту до якого прямує запит

```
<VirtualHost *:80>  
    DocumentRoot "/www/html/example"  
    ServerName example.com  
</VirtualHost>
```

webserver перевіряє location

І якщо локейшен не знайдено запускаються
"привиля" віртуальних шляхів

```
<IfModule mod_rewrite.c>
    RewriteRule ^index\.php$ - [L]
    RewriteCond %{REQUEST_FILENAME} !-f
    RewriteCond %{REQUEST_FILENAME} !-d
    RewriteRule . /index.php [L]
</IfModule>
```

WordPress & **rewrite_rules**

```
$rewrite_rules = [  
    '^wp-json/?$' => 'index.php?rest_route=/'  
    '^wp-json/(.*)?' => 'index.php?rest_route=/' . $matches[1]  
    '^index.php/wp-json/?$' => 'index.php?rest_route=/'  
    '^index.php/wp-json/(.*)?' => 'index.php?rest_route=/' . $matches[1]  
    'category/(.+?)/feed/(feed|rdf|rss|rss2|atom)/?$' => 'index.php?category_name=$matches[1]&feed=$matches[2]  
    'category/(.+?)/(feed|rdf|rss|rss2|atom)/?$' => 'index.php?category_name=$matches[1]&feed=$matches[2]  
    'category/(.+?)/embed/?$' => 'index.php?category_name=$matches[1]&embed=true'  
    'category/(.+?)/page/?([0-9]{1,})/?$' => 'index.php?category_name=$matches[1]&paged=$matches[2]  
    'category/(.+?)/?$' => 'index.php?category_name=$matches[1]  
    'tag/([^\s]+)/feed/(feed|rdf|rss|rss2|atom)/?$' => 'index.php?tag=$matches[1]&feed=$matches[2]  
    'tag/([^\s]+)/(feed|rdf|rss|rss2|atom)/?$' => 'index.php?tag=$matches[1]&feed=$matches[2]  
    'tag/([^\s]+)/embed/?$' => 'index.php?tag=$matches[1]&embed=true'  
    'tag/([^\s]+)/page/?([0-9]{1,})/?$' => 'index.php?tag=$matches[1]&paged=$matches[2]  
    'tag/([^\s]+)/?$' => 'index.php?tag=$matches[1]  
    // Тут було ще 71 правила розбору URI на складові, але я їх прибрав. Все не влізло.  
    'index.php?pagename=$matches[1]&feed=$matches[2]  
    '(.?+?)/(feed|rdf|rss|rss2|atom)/?$' => 'index.php?pagename=$matches[1]&feed=$matches[2]  
    '(.?+?)/page/?([0-9]{1,})/?$' => 'index.php?pagename=$matches[1]&paged=$matches[2]  
    '(.?+?)/comment-page-([0-9]{1,})/?$' => 'index.php?pagename=$matches[1]&cpage=$matches[2]  
    '(.?+?)(?:/([0-9]+))/?$' => 'index.php?pagename=$matches[1]&page=$matches[2]  
];
```


URI & RegExp

```
/tag/wp/page/2 => tag/([^\s/]+)/page/?([0-9]{1,})/?$ => index.php?tag=wp&paged=2
```

Query_String & WP_Query

```
'tag=wp&paged=2' => new WP_Query([ 'tag' => 'wp', 'paged' => 2]);
```

Завантаження та Відображення

Завантаження записів з допомогою WP_Query

Локація необхідного темплейту

Відображення записів у темплейті

майже

The End

Важливе питання #1

Що собою являють ці правила?

Важливе питання #1 — Що собою являють ці правила?

[`match_rule` => `match_query`]

Важливе питання #1 — Що собою являють ці правила?

Ключі – **Match Rule**

```
tag/([ ^ / ]+)/page/?( [ 0-9 ] { 1 , } ) / ? $
```

Це **регулярні вирази** що мають своєю метою співпасти з адресою запиту і (можливо) отримати його фрагменти для подальшого використання.

Важливе питання #1 — Що собою являють ці правила?

Значення - **Match Query**

```
index.php?tag=$matches[1]&paged=$matches[2]
```

Рядок-шаблон: на місце **\$matches[\d{1,}]** ми
підставимо ті значення, що ми змогли вийняти з URI за
допомогою RegExr Match Rule

Важливе питання #2 — Що собою являють ці правила?

Результат збігу - Query Vars

https://codex.wordpress.org/WordPress_Query_Vars

```
'tag=wp&paged=2' && [ 'tag' => 'wp', 'paged' => 2 ]
```

Спеціальні змінні які передаються класу WP_Query у вигляді рядку або масиву. Ми використовуємо їх що діставати різноманітні записи.

Важливе питання #2

Навіщо змінювати правила?

Важливе питання #2 — Навіщо змінювати правила?

Правила "надлишкові"

```
$rewrite_rules = [  
    '^wp-json/?$' => 'index.php?rest_route=/'  
    '^wp-json/(.*)?' => 'index.php?rest_route=/$matches[1]'  
    '^index.php/wp-json/?$' => 'index.php?rest_route=/'  
    '^index.php/wp-json/(.*)?' => 'index.php?rest_route=/$matches[1]'  
    'category/(.+?)/feed/(feed|rdf|rss|rss2|atom)/?$' => 'index.php?category_name=$matches[1]&feed=$matches[2]'  
    'category/(.+?)/(feed|rdf|rss|rss2|atom)/?$' => 'index.php?category_name=$matches[1]&feed=$matches[2]'  
    'category/(.+?)/embed/?$' => 'index.php?category_name=$matches[1]&embed=true'  
    'category/(.+?)/page/?([0-9]{1,})/?$' => 'index.php?category_name=$matches[1]&paged=$matches[2]'  
    'category/(.+?)/?$' => 'index.php?category_name=$matches[1]'  
    'tag/([^/]+)/feed/(feed|rdf|rss|rss2|atom)/?$' => 'index.php?tag=$matches[1]&feed=$matches[2]'  
    'tag/([^/]+)/feed/(feed|rdf|rss|rss2|atom)/?$' => 'index.php?tag=$matches[1]&feed=$matches[2]'  
    'tag/([^/]+)/embed/?$' => 'index.php?tag=$matches[1]&embed=true'  
    'tag/([^/]+)/page/?([0-9]{1,})/?$' => 'index.php?tag=$matches[1]&paged=$matches[2]'  
    'tag/([^/]+)/?$' => 'index.php?tag=$matches[1]'  
    // Тут було ще 70 правил розбору URI, але я їх прибрав. Все не влізло.  
    '(?.+?)/feed/(feed|rdf|rss|rss2|atom)/?$' => 'index.php?pagename=$matches[1]&feed=$matches[2]'  
    '(?.+?)/(feed|rdf|rss|rss2|atom)/?$' => 'index.php?pagename=$matches[1]&feed=$matches[2]'  
    '(?.+?)/page/?([0-9]{1,})/?$' => 'index.php?pagename=$matches[1]&paged=$matches[2]'  
    '(?.+?)/comment-page-([0-9]{1,})/?$' => 'index.php?pagename=$matches[1]&cpage=$matches[2]'  
    '(?.+?)(?:/([0-9]+))/?$' => 'index.php?pagename=$matches[1]&page=$matches[2]'  
];
```

Важливе питання #2 — Навіщо змінювати правила?

Це весело! І надихає на дослідження!

Важливе питання #2 — Навіщо змінювати привила?

Інколи, це питання безпеки

Важливе питання #2 — Навіщо змінювати привила?

Гнучкий контроль над WP_Query

Важливе питання #3

Як змінювати правила?

Важливе питання #3 — Як змінювати правила?

add_rewrite_rule

```
add_rewrite_rule( 'filter/(.*)\/?$', 'index.php?filter=$matches[1]', 'top' );
```

- =) Простота використання
- =(Складно дебажити

Важливе питання #3 — Як змінювати правила?

"Фільтри" – хліб та масло розробника WP.

```
add_action( 'post_rewrite_rules', function( array $rewrite_rules ) : array {  
    $rewrite_rules[ '^api/' ] = 'index.php?do_api_list=1';  
    return $rewrite_rules;  
} );
```

Використовуйте фільтри для `rewrite_rules`

- => Їх багато і вистачить надовго
- => Правила можна переписувати не тільки для index.php
- =(Порядок правил має велику вагу
- =(Розуміння роботи Regular Expressions
- =(Не забувайте навішувати коректні фільтри на одиниці контенту якщо вони додані до правил

Важливе питання #4

А як генерувати посилання?

Важливе питання #4 — А як генерувати посилання?

add_permastruct

```
add_action( 'init', function() {  
    add_permastruct( 'tvshow', '/videoteka/%tvshow%.html' );  
});
```

Але, тут багато "але"...

- =(Простота використання
- =(Націлене на taxonomu та post_type
- =(Лімітоване API
- =(Багато додаткової роботи

Важливе питання #4 — А як генерувати посилання?

add_rewrite_tag

```
add_rewrite_tag( '%random%', '(.)', 'random=' );
// Оголошуємо цей тег публічним - він з'явиться в wp-admin/options-permalinks.php
add_filter( 'available_permalink_structure_tags', function( array $tags ) : array {
    $tags['random'] = 'Заміна в посиланні на кльову якусь штуку';
    return $tags;
});
// Або 'post_link', або або інший фільтр для лінку
add_filter( 'post_type_link', function( string $permalink ) : string {
    if (false === strpos( $permalink, '%random%' ) ) {
        return $permalink;
    }
    return str_replace( '%random%', rand(1000, 4000), $permalink);
});
```

- Простота використання з нормальним API
- Допомогає `add_permalink` бути адекватним

Важливе питання #4 — А як генерувати посилання?

Використовуючи фільтри посилань

Наприклад `term_link` або `post_type_link`

```
// Певні категорії будуть мати інше розташування
add_filter( 'term_link', function( $term_link, $term, $taxonomy ) {
    if ( 'superduper' !== $term->taxonomy ) {
        return $term_link;
    }
    if ( '1' !== get_term_meta( $term->term_id, 'special' ) ) {
        return $term_link;
    }
    return home_url( get_option( 'alternative_base_superduper' ) . '/' . $term->slug );
}, 10, 3);
```

Важливе питання #4 — А як генерувати посилання?

Ручками

「_(ツ)_/」

Кейси – коротко про приклади

Case #1 — Трансліт місяців

`https://example.ua/blog/2018/kvyten/20`

Хочемо

vs

`https://example.ua/blog/2018/04/20`

Маємо

Код @ <https://github.com/butuzov/WP-Using-Non-Default-URIs>

Нам допоможуть

- Зміни до `rewrite_rules` - `post_rewrite_rules`, `date_rewrite_rules`
- Генерація нових посилань - `post_link`, `day_link`, `month_link`
- Query String підміняємо за допомогою ... `query_string`

Case #2 & #3 — Прибираємо "зайві" правила

Зручність роботи з `rewrite_rules` зворотно пропорційна кількості

Код @ <https://github.com/butuzov/WP-Using-Non-Default-URIs>

Фільтруємо

- Глобальні правила — `category_rewrite_rules`, `category_rewrite_rules`, `post_format_rewrite_rules`, `post_rewrite_rules`, `date_rewrite_rules`, `root_rewrite_rules`, `search_rewrite_rules`, `page_rewrite_rules`, `tag_rewrite_rules`
- Динамічні правила — `{ $post_type }_rewrite_rules`, `{ $taxonomy }_rewrite_rules`
- Усі правила — `rewrite_rules_array`

Case #4 — Змінюємо стандартну поведінку архіву

Визначаємо кількість записів для однієї сторінки архіву.

Код @ <https://github.com/butuzov/WP-Using-Non-Default-URIs>

Реалізація

- Використовуйте фільтр `parse_query` щоб змінити поведінку.
- Коли можна/треба - змінюйте `matched query`.
- Не забувайте змінювати правила після оновлень.

Case #5 — Рознесені Категорії

Дати кожній категорії власний URI

Код @ <https://github.com/butuzov/WP-Using-Non-Default-URIs>

Реалізація

- Збереження мета інформації в `term_meta`
- Генерація окремих правил для категорій під час `${taxonomy}_rewrite_rule`
- Генерація окремих окремих посилань для категорій

Case #6 — Скоро на екранах

Показуємо `future` записи

Код @ <https://github.com/butuzov/WP-Using-Non-Default-URIs>

Реалізація

- Показує `future` записи на окремій сторінці `upcoming`
- Додає правило з новим `query_var` - `upcoming`
- Під час парсингу \$wp_query з `upcoming` - ми змінємо поведінку WP

майже

The End

Дякую

Хвилинка реклами

Debug Bar Rewrite Rules

<https://wordpress.org/plugins/debug-bar-rewrite-rules/>

Плагін для роботи з rewrite rules


- Тест ваших правих правил проти URI
- Пошук у правилах
- Перегляд доступних та задіяних фільтрів

Resources

- Як WordPress працює з запитамі: https://codex.wordpress.org/Query_Overview
- Приклади: <https://github.com/butuzov/WP-Using-Non-Default-URIs>

Питання?

 made.ua  [butuzov@made.ua](https://t.me/butuzov@made.ua)

 [butuzov](https://twitter.com/butuzov)  [butuzov](https://github.com/butuzov)  [butuzov](https://in.linkedin.com/company/butuzov)  [butuzov](https://wp.butuzov.com)