Пользовательские типы записей, пользовательские таксономии и метаданные

7

В этой главе:

- Понимание и создание пользовательских типов записей
- Отображение и использование контента записей пользовательского типа
- Создание и использование пользовательских таксономий
- Понимание и использование метаданных

Самая важная часть любого веб-сайта на WordPress — контент. В WordPress по умолчанию определены различные типы контента и таксономий, но нередко вам требуется определить собственные типы контента для построения именно того сайта, который вам нужен.

За последние несколько лет WordPress представил несколько весьма продвинутых и простых в использовании инструментов для работы со всеми типами произвольного контента. Это помогло ему развиться в полноценную систему управления контентом, способную поддерживать абсолютно любые типы сайтов, вне зависимости от содержимого.

В этой части вы узнаете, как создавать пользовательские типы записей и контента в WordPress. Вы также изучите работу с пользовательскими таксономиями для группировки и классификации контента. Наконец, вы выясните, как присоединять произвольные фрагменты метаданных к контенту.

Понимание данных в WordPress

При работе с различными типами данных в WordPress важно понимать, что это за данные и как они могут быть индивидуализированы. В базовой копии WordPress есть пять предустановленных типов записей:

1. **Post (Запись).** Записи или статьи в блоге, обычно упорядоченные по дате.
2. **Page (Страница).** Иерархические статичные страницы с контентом.
3. **Attachment (Приложение).** Медиафайлы, загруженные в WordPress и прикрепленные к записям, такие как изображения и файлы.
4. **Revision (Редакция).** Редакции записей, используемые как резервные копии, которые могут быть восстановлены при необходимости.
5. **Nav Menus (Меню навигации).** Пункты меню, добавленные в меню навигации с использованием функции управления меню WordPress.

Для базового блога или небольшого сайта достаточно этих предустановленных типов записей. Однако если вы планируете построение более сложного сайта с системой управления контентом CMS, вам могут понадобиться пользовательские типы записей.

Что такое пользовательский тип записи?

Пользовательский тип записи в WordPress — это фрагмент контента, заданный пользователем. Эти типы записей позволяют определить любой тип контента в WordPress, чтобы не ограничиваться только предустановленными типами записей, перечисленными в предыдущем разделе. Это дает бесконечное число возможностей.

Среди возможных идей для индивидуальных типов записей:

- продукты;
- события;
- видео;
- благодарности;
- цитаты;
- журнал ошибок.

Не забывайте, что пользовательские типы записей могут быть абсолютно чем угодно, не только доступными публично фрагментами контента. Например, вы можете установить пользовательский тип записи как журнал ошибок, чтобы отслеживать ошибки в приложении. Здесь вы ограничены только собственным воображением.

Регистрация пользовательского типа записей

Для создания нового пользовательского типа записи вы будете использовать функцию `register_post_type()`:

```
<?php register_post_type( $post_type, $args ); ?>
```

Функция `register_post_type()` принимает два параметра:

1. `$post_type` — название типа записи. Может содержать только строчные буквы, пробелы запрещены, максимальная длина — 20 символов.
2. `$args` — массив аргументов, которые определяют тип записи и различные параметры в WordPress.

Теперь посмотрим на базовый пример регистрации пользовательского типа записи. Вы можете зарегистрировать тип записи в WordPress в двух разных местах: первое — это файл темы `functions.php`; второе — это пользовательский плагин. Вы можете добавить приведенный ниже код и в пользовательский плагин, но для данного примера внесем его в файл темы `functions.php`.

```
<?php
add_action( 'init', 'prowp_register_my_post_types' );
function prowp_register_my_post_types() {
    register_post_type( 'products',
        array(
            'labels' => array(
                'name' => 'Products'
            ),
            'public' => true,
        )
    );
}
?>
```

Теперь зайдите в консоль WordPress. Вы увидите, что в ней ниже вкладки Комментарии (Comments) появилась вкладка Products (Продукты), как показано на рис. 7.1. Это новый пользовательский тип записи, который вы зарегистрировали с помощью предыдущего кода.

Как видите, WordPress автоматически создает пользовательский интерфейс администратора для нового пользовательского типа записи. Новый пункт меню позволяет вам создавать новые записи типа «Продукты», а также редактировать существующие записи наравне с прочими в WordPress. Это базовый пример, но вы уже видите, насколько легко задавать произвольный контент в WordPress.

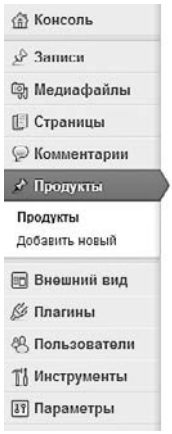

Рис. 7.1. Пользовательский тип записи «Продукты»

ЗАМЕЧАНИЕ

При регистрации пользовательских типов записей нужно всегда использовать зацепку `init`. Это первая зацепка, доступная после полной инициализации WordPress, которая будет проверять, что пользовательский тип записи достаточно рано регистрируется в процессе.

При регистрации пользовательского типа записи доступно множество аргументов. Для того чтобы ими пользоваться, важно их понимать.

public

Определяет, доступен тип записи публично в консоли администратора или в пользовательском интерфейсе сайта. По умолчанию выставлено значение `false`, скрывающее тип записи от просмотра. Предустановленные значения для `show_ui`, `exclude_from_search`, `publicly_queryable` и `show_in_nav_menus` зависят от этой настройки.

show_ui

Определяет, создается ли по умолчанию пользовательский интерфейс в консоли администратора WordPress для управления этим типом записи. По умолчанию значение определяется аргументом `public`.

publicly_queryable

Определяет, может ли контент записи этого типа быть публично запрошен через пользовательский интерфейс сайта. Если выставлено значение `false`, все запросы от пользователей будут возвращать ошибку 404, поскольку запрос запрещен. По умолчанию значение определяется аргументом `public`.

exclude_from_search

Позволяет исключить записи пользовательского типа из результатов поиска WordPress. По умолчанию значение определяется аргументом `public`.

show_in_nav_menus

Определяет, будет ли тип записи доступен для выбора при управлении меню WordPress. По умолчанию значение определяется аргументом `public`.

supports

Этот аргумент позволяет определить, какие метаполя появляются на экране при создании или редактировании записи нового типа (по умолчанию `title` и `editor`):

- `title` — задает название записи.
- `editor` — отображает редактор контента на экране редактирования записи с загрузчиком медиафайлов.
- `author` — выбирает поле для определения автора записи.
- `thumbnail` — задает метаполе миниатюры для записи.
- `excerpt` — отображает редактор цитат на экране редактирования типа записи.

- `comments` — определяет, будут ли активированы комментарии для записи этого типа.
- `trackbacks` — определяет, будут ли активированы трэкбеки и пингбеки.
- `custom-fields` — отображает метаполе зоны редактирования произвольных полей.
- `page-attributes` — отображает поле атрибутов для выбора порядка записи. Для этого аргументу `hierarchical` следует задать значение `true`.
- `revisions` — отображает поле редакций записи.
- `post-formats` — отображает метаполе форматов записи с зарегистрированными форматами записи.

labels

Определяет массив ярлыков, описывающих тип записи в консоли администратора. См. раздел «Определение ярлыков типа записи» далее в этой главе.

hierarchical

Аргумент `hierarchical` позволяет задать, является ли тип записи иерархическим, подобно страницам WordPress. `hierarchical` позволяет иметь древовидную структуру контента записей данного типа. По умолчанию аргументу присвоено значение `false`.

has_archive

Позволяет типу записи иметь архивную страницу. Архивная страница типа записи подобна странице записей WordPress, отображающей последние записи в блоге. Это позволяет отображать список записей данного типа в порядке, определенном в файле шаблона темы.

can_export

Определяет, будет ли контент записей данного типа доступен для экспорта с использованием встроенной возможности экспорта WordPress (Инструменты ▸ Экспорт). По умолчанию аргументу присвоено значение `true`.

taxonomies

Дает название массиву зарегистрированных таксономий для прикрепления к индивидуальному типу записей. Например, вы можете передать в этом массиве `category` и `post_tag`, чтобы прикрепить предустановленные таксономии рубрик и меток к вашему типу записи. По умолчанию к пользовательскому типу записи никакие таксономии не прикреплены.

menu_position

Позволяет выбрать позицию, в которой пользовательский тип записи будет отображаться в меню администратора. По умолчанию новые типы записей отображаются после вкладки Комментарии.

menu_icon

Устанавливает индивидуальную иконку в меню для вашего типа записи. По умолчанию используется иконка для записей.

show_in_menu

Определяет, будет ли отображаться меню администратора для вашего типа записи. Аргумент принимает три значения: `true`, `false` или строку. Строка может быть либо страницей верхнего уровня, как `tools.php`, либо `edit.php?post_type=page`. Вы также можете задать строке параметр `menu_slug`, чтобы добавить пользовательский тип записи как объект подменю в существующем пользовательском меню. По умолчанию определяется значением аргумента `show_ui`.

show_in_admin_bar

Определяет, будет ли отображаться пользовательский тип записи в панели администратора WordPress. По умолчанию определяется значением аргумента `show_in_menu`.

capability_type

Дает название строке или массиву характеристик для этого типа записи. По умолчанию приписывается значение `post`.

capabilities

Это массив индивидуальных характеристик, необходимых для редактирования, удаления, просмотра и публикации записей данного типа.

query_var

Этот аргумент устанавливает переменную запроса записей данного типа. По умолчанию его значение `true` и устанавливается для `$post_type`.

rewrite

Аргумент `rewrite` создает уникальные постоянные ссылки для этого типа записей. Это позволит вам сделать индивидуальным слаг типа записи в URL. Аргументу могут быть присвоены значения `true`, `false` или массив значений.

- `slug` — определяет индивидуальный слаг постоянной ссылки. Значение по умолчанию для `$post_type`.
- `with_front` — определяет, будет ли тип записи использовать основу постоянных ссылок. Например, если префиксом ваших постоянных ссылок является `/blog`, а значение `with_front` определено как `true`, постоянные ссылки записей этого типа будут начинаться с `/blog`.
- `pages` — определяет, будет ли постоянная ссылка обеспечивать разбиение на страницы. Значение по умолчанию `true`.
- `feeds` — определяет, будет ли постоянная ссылка на фид встроена в записи этого типа. Значение по умолчанию совпадает со значением `has_archive`.

По умолчанию значение аргумента определено как `true`, а `$post_type` используется в качестве слага.

Этот раздел охватывает множество аргументов типа записи. Ниже приведены наиболее распространенные примеры их использования.

```
<?php
add_action( 'init', 'prowp_register_my_post_types' );
function prowp_register_my_post_types() {
  $args = array(
        'public' => true,
        'has_archive' => true,
        'taxonomies' => array( 'category' ),
        'rewrite' => array( 'slug' => 'product' ),
        'supports' => array( 'title', 'editor', 'author', 'thumbnail', 'comments' )
    );
    register_post_type( 'products', $args );
}
?>
```

В этом примере вы сначала определяете, что тип записи будет публичным. Вы также указываете, что у типа записи есть архивная страница, приписывая аргументу `has_archive` значение `true`. Используя аргумент `taxonomies`, вы прикрепляете предустановленную таксономию Category к индивидуальному типу записи Products.

В данном примере вы хотите изменить слаг постоянной ссылки для вашего типа записи. Вместо http://example.com/products/zombie-bait, используя предустановленный слаг `products` из имени типа записи, вы задаете слаг типа записи для отдельного `product`. Будет сгенерирована постоянная ссылка типа http://example.com/product/zombie-bait. Это делается с использованием аргумента `rewrite` и определением индивидуального слага для вашего типа записи. В итоге вы устанавливаете аргумент `supports`. Этот код добавляет название, редактора, автора, миниатюру и метаполе комментариев к экрану создания и редактирования записи пользовательского типа.

Для получения более подробной информации по функции `register_post_type()` обратитесь к официальной странице Кодекса http://codex.wordpress.org/Function_Reference/register_post_type.

ЗАМЕЧАНИЕ

При регистрации нового пользовательского типа записи важно сбросить правила преобразования в WordPress. Вы можете сделать это, вызвав функцию `flush_rewrite_rules()` в модуле активации плагина или же вручную, через Параметры ▸ Постоянные ссылки и сохранение настроек постоянных ссылок. Это поможет избежать ошибок 404 по постоянным ссылкам записей нового типа.

Определение ярлыков типа записи

При создании пользовательского типа записи в WordPress для него показывается несколько строк в консоли администратора. Эти строки обычно представляют собой ссылку, кнопку или дополнительную информацию о типе записи. По умолчанию термин «запись» используется для неиерархического типа записей, а термин «страница» — для иерархического.

Например, если вы используете приведенный ранее в этой части базовый код регистрации пользовательского типа записи, то при добавлении нового товара в верхней части страницы увидите текст «Add New Post» («Добавить новую запись»). Причина в том, что товар — это запись типа Product. Это не совсем точно, потому что вы на самом деле добавляете не запись, а новый товар. Настройка аргумента `labels` (ярлыки) при регистрации пользовательского типа записи позволит вам точно определять, как этот тип отображается на экране и в меню.

Среди доступных ярлыков для пользовательского типа записи:

- `name` — общее название типа записи, обычно во множественном числе. Используется в консоли WordPress, а также другими плагинами и темами.
- `singular_name` — название в единственном числе. Используется в консоли WordPress, а также другими плагинами и темами.
- `add_new` — ярлык для добавления объекта подменю «Add New» («Добавить новую»). Текст по умолчанию: «Add New».
- `add_new_item` — заголовок на главной странице со списком записей для добавления новой записи. Текст по умолчанию: «Add New Post/Page» («Добавить новую запись/страницу»).
- `edit_item` — используется для редактирования отдельной записи. Текст по умолчанию: «Edit Post/Page» («Редактировать запись/страницу»).
- `new_item` — текст для создания новой записи. Текст по умолчанию: «New Post/Page» («Новая запись/страница»).
- `view_item` — текст для отображения отдельной записи. Текст по умолчанию: «View Post/Page» («Просмотр записи/страницы»).
- `search_items` — текст, отображаемый для поиска записей этого типа. Текст по умолчанию: «Search Posts/Pages» («Поиск записей/страниц»).

- `not_found` — текст, отображаемый, если ни одна запись не найдена. Текст по умолчанию: «No posts/pages found» («Записей/страниц не найдено»).
- `not_found_in_trash` — текст, отображаемый, если в корзине не найдено ни одной записи. Текст по умолчанию: «No posts/pages found in Trash» («Записей/страниц в корзине не найдено»).
- `parent_item_colon` — текст, отображаемый при показе родительской страницы. Используется только для записей иерархического типа, текст по умолчанию «Parent Page» («Родительская страница»).

Настройка каждого значения делает администрирование WordPress более приятным. В следующем коде производится изменение оригинального кода регистрации пользовательского типа записи и настраиваются ярлыки для типа записи Product:

```
<?php
add_action( 'init', 'prowp_register_my_post_types' );
function prowp_register_my_post_types() {
    $labels = array(
        'name' => 'Products',
        'singular_name' => 'Product',
        'add_new' => 'Add New Product',
        'add_new_item' => 'Add New Product',
        'edit_item' => 'Edit Product',
        'new_item' => 'New Product',
        'all_items' => 'All Products',
        'view_item' => 'View Product',
        'search_items' => 'Search Products',
        'not_found' =>  'No products found',
        'not_found_in_trash' => 'No products found in Trash',
        'parent_item_colon' => '',
        'menu_name' => 'Products'
    );
    $args = array(
        'labels'    =>    $labels,
        'public'    =>    true
    );
    register_post_type( 'products', $args );
}
?>
```

Работа с пользовательскими типами записи

Теперь, когда вы понимаете, как регистрировать пользовательский тип записи, давайте изучим, как использовать его при разработке сайта на WordPress. Обычно это задача темы — отображать записи на титульной странице сайта. Однако это не всегда так, поскольку некоторые из пользовательских типов записей не требуют публичного отображения — например, журнал ошибок. Все зависит от того, какова функция этого типа записи.

Чтобы отобразить данные записи пользовательского типа, можно использовать произвольный цикл `WP_Query`, приведенный как пример в главе 5. Помните, что

WP_Query принимает параметр post_type, который определяет, какой тип контента возвращать. В приведенном далее примере возвращаются все записи типа Product в WordPress:

```
$args =  array(
    'posts_per_page' => '-1',
    'post_type' => 'products',
);
$myProducts = new WP_Query( $args );
// Цикл
while ( $myProducts->have_posts() ) : $myProducts->the_post();
    ?><a href="<?php the_permalink(); ?>"><?php the_title(); ?></a><br /><?php
endwhile;
// Восстанавливаем значение глобальной переменной $post
wp_reset_postdata();
```

Обратите внимание, что параметр post_type определен как products, что является значением параметра $post_type при регистрации пользовательского типа записи Products.

Теперь изменим произвольный цикл, чтобы возвращать только товары рубрики Specials:

```
$args =  array(
    'posts_per_page' => '-1',
    'post_type' => 'products',
    'tax_query' => array(
        array(
            'taxonomy' => 'category',
            'field' => 'slug',
            'terms' => 'specials'
        )
    )
);
$myProducts = new WP_Query( $args );
```

Используя параметр tax_query в WP_Query, произвольный цикл вернет только записи типа Product, относящиеся к рубрике Specials.

Вы можете использовать все методы создания произвольных циклов с WP_Query, которые описаны в главе 5, чтобы отображать записи пользовательского типа. Насколько полезны пользовательские записи в WordPress, становится понятно при разработке более сложных сайтов.

Файлы шаблона записи пользовательского типа

Ранее в этой части при регистрации пользовательского типа записи вы изучили has_archive. Активация этого аргумента позволит вам создавать файл по шаблону архива, который будет отображать все записи пользовательского типа по умолчанию. Шаблон архива для записи пользовательского типа должен быть назван по стандарту archive-{post-type}.php. Например, шаблон архива для