

## Изучаем файлы шаблона Wordpress

Короткое вступление.

Всем моим читателям – несколько пояснений по тексту.

Я публиковал серию статей на своем сайте. Знаю, что не всегда удобно читать частями какой-то обзор, да еще и сидя перед монитором. Решил, что кто-то из вас с удовольствием скачает этот обзор в формате PDF – документа, и спокойно будет читать в удобное время у себя на компе или распечатает на работе и будет читать как все нормальные люди.

В этой статье вы найдете много интересных примеров и анализа кода PHP файлов шаблона Wordpress. В качестве примера «разбора» файлов я использовал один, довольно простой и стандартный шаблон O2, автор - <http://blog.eches.net>.

Шаблон можно скачать на моем сайте вот по этой ссылке - <http://www.wpfreethemes.ru/wp-content/files/o2-20.zip>

В этой статье я писал много отрывков кода PHP. Возможно, что он не совсем корректно здесь отображен. Я выделил все части кода серым фоном, чтобы вам было лучше ориентироваться на страницах статьи.

Надеюсь, что эта статья пригодится тем из вас, кто действительно хочет немного лучше знать свой сайт и свой шаблон Wordpress.

Удачи и успехов в выполнении всех ваших желаний!

### Статья первая

Всем доброго времени суток!

Вы спросите - а зачем их изучать? Взял понравившийся шаблон, забросил его на сервер, активировал и - супер! Все классно! Можно писать, добавлять картинки, и чувствовать себя супер-пупер великим веб-мастером.

А потом приходит момент, когда вам захочется что-то улучшить или изменить, например:

- Добавить кнопки социальных закладок. Такие красивые увидел на одном блоге! И себе захотелось.
- Не нравится шрифт в заголовке, хочется изменить, а как - ума не приложу. Где искать?
- Вместо текстового заголовка хочется поставить Логотип-картинку (или наоборот). Где искать код и как изменить?
- Как изменить цикл The Loop вывода публикаций. Например, на главной странице? Убрать какую-то рубрику и перенести ее, например, в сайдбар? Использовать в сайдбаре цикл The Loop или что-то другое?
- Как сделать в шаблоне вместо одного сайдбара - два? Что надо изменить, в каком месте?

Вы можете продолжить этот список вопросов. Я его написал на основе ваших писем, которые вы присылаете мне на почту. Вопросов много, и большинство этих вопросов крутится вокруг знания файлов шаблона, знания функций PHP, которые отвечают за выполнение тех или иных запросов к базе данных, в которой хранится вся публикуемая вами информация.

Вот я и решил начать публиковать для всех вас серию статей, которая будет посвящена детальному описанию файлов стандартного шаблона Wordpress, в котором нет ничего лишнего, никаких встроенных нестандартных функций PHP, отсутствуют всякие нестандартные запросы к базе данных.

Я подберу для изучения в качестве образца один из шаблонов Wordpress, дам возможность вам его скачать и на его примере буду рассказывать вам о каждом файле шаблона, описывать все находящиеся в нем функции PHP, рассказывать, для чего они служат, какие теги HTML и свойства CSS сопровождают тот или иной код файла. Я надеюсь, что мои публикации помогут тем из вас, кто хочет детально изучить файлы своего шаблона и научиться вносить в них свои изменения.

Найти идеальный шаблон практически невозможно. В каждом есть свои недостатки с точки зрения присутствия встроенного функционала, дизайна и макетирования.

Думаю, что материала будет много.

Надеюсь, если ничего не помешает, начать публикацию этой серии статей уже завтра. Так что заходите, читайте, изучайте.

Будут вопросы - там же и задавайте. Отвечу всем. Как всегда.

## Статья вторая

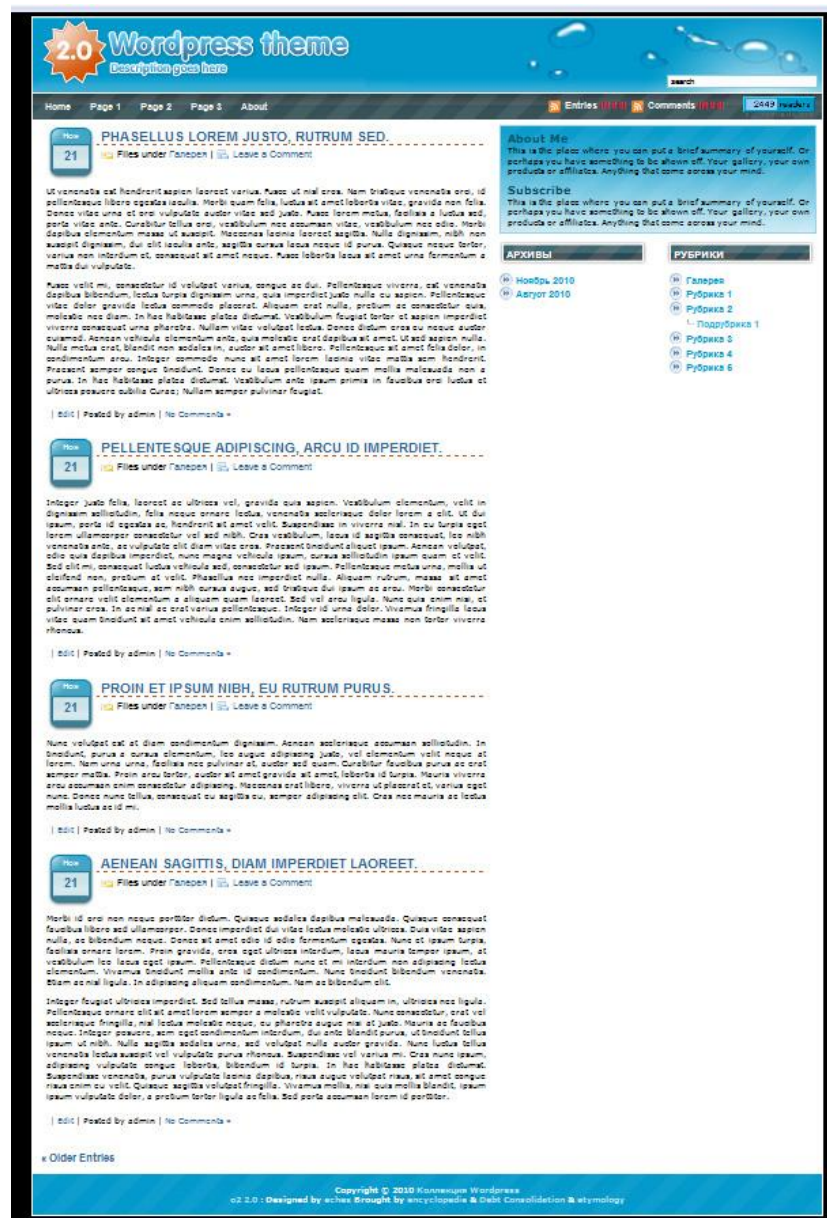
Сегодня я хочу вам рассказать о том, как строится наш сайт на Wordpress, из каких основных файлов состоит и почему.

Для того, чтобы вы понимали и видели, о каком коде идет речь, я предложил всем, кому интересна эта публикация, скачать **шаблон o2**, который я буду использовать как пример.

Если вы не скачали его в первой статье – можете [скачать его сейчас](#).

Чтобы понимать в целом, что такое наша страница сайта (начнем с главной), я нарисовал несколько рисунков и хочу свой рассказ ими сопроводить.

Шаблон o2 был написан автором еще под Wordpress версии 2.5., но отлично работает и на тройке. Итак, смотрим на наш сайт на этом шаблоне:



Стандартная тема: 3 колонки, основная с записями слева, 2 сайдбара справа, над ними – небольшая секция об авторе.

Узкий заголовок с картинкой – логотипом слева, справа – окно поиска на сайте.

Навигационное меню страниц (с поддержкой внутренних – дойдет очередь, расскажу), справа, в строке меню – подписка читателей на записи и комментарии, а также кнопка для подписки через сервис Feedburner.

В подвале – обычные ссылки на авторский сайт, спонсоров и копирайт.

Я писал, что кроме анализа кода файлов, я покажу вам как сделать перевод на русский внутри файлов, не пользуясь специальной программой локализации.

Если еще раз посмотреть на скриншот, вы увидите через призму самой страницы как-бы секции, блоки, на которые условно можно разделить нашу страницу сайта:

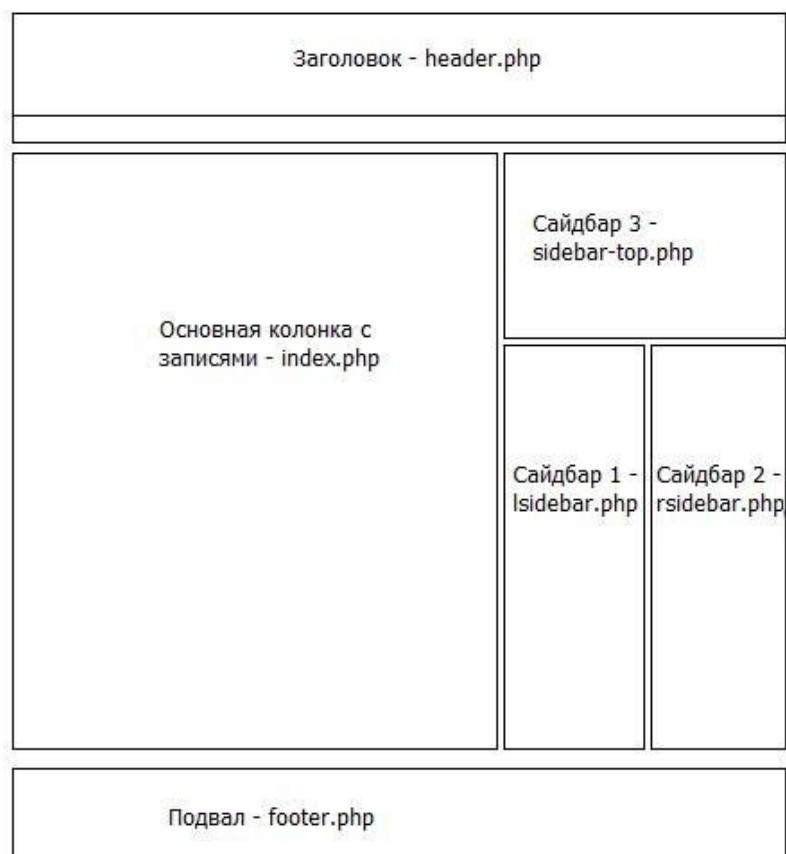
Заголовок

Основная колонка с текстом

2 сайдбара

Подвал

Теперь давайте эти секции нарисуем:



На рисунке я выделил еще одну секцию – верхний блок над сайдбарами, где в шаблоне выводится короткая “вьюха” (от слова превью – просмотр, анонс) об авторе.

Рядом с названием каждой секции я написал имя файла из нашего шаблона. Таким образом, я вам показал, какие основные файлы составляют нашу главную страницу. Основные – потому что без них мы не увидим всей страницы. О второстепенных файлах – чуть позже.

Пару слов о свойствах и стилях. Без них наш сайт не будет выглядеть так как выглядит сейчас.

На сегодняшний день есть как-бы два стандарта, или два варианта (кому как больше нравится говорить) верстки шаблонов сайта (не важно – WordPress или что-то другое):

- Блочная верстка
- Табличная верстка

Отличия между ними в том, что при написании кода HTML в табличной верстке, сайт условно дизайнером и программистом делится на таблицу, внутри которой – ячейки, колонки, строки. Каждой ячейке присваивается определенный порядковый номер, а также задаются определенные свойства стилей (начертание, цвет, размещение элементов и т.д.).

В блочной верстке немного по-другому: весь сайт, от самой первой строки и колонки (условно – левый верхний угол, первый пиксель на экране в левом верхнем углу – и до нижнего правого угла, пикселя) разделен на секции и блоки, которые на языке файла стилей CSS называются набором параметров форматирования (на языке программистов – дивы и классы).

Основной параметр – див, закрыт тегами и выглядит у вас на странице так:

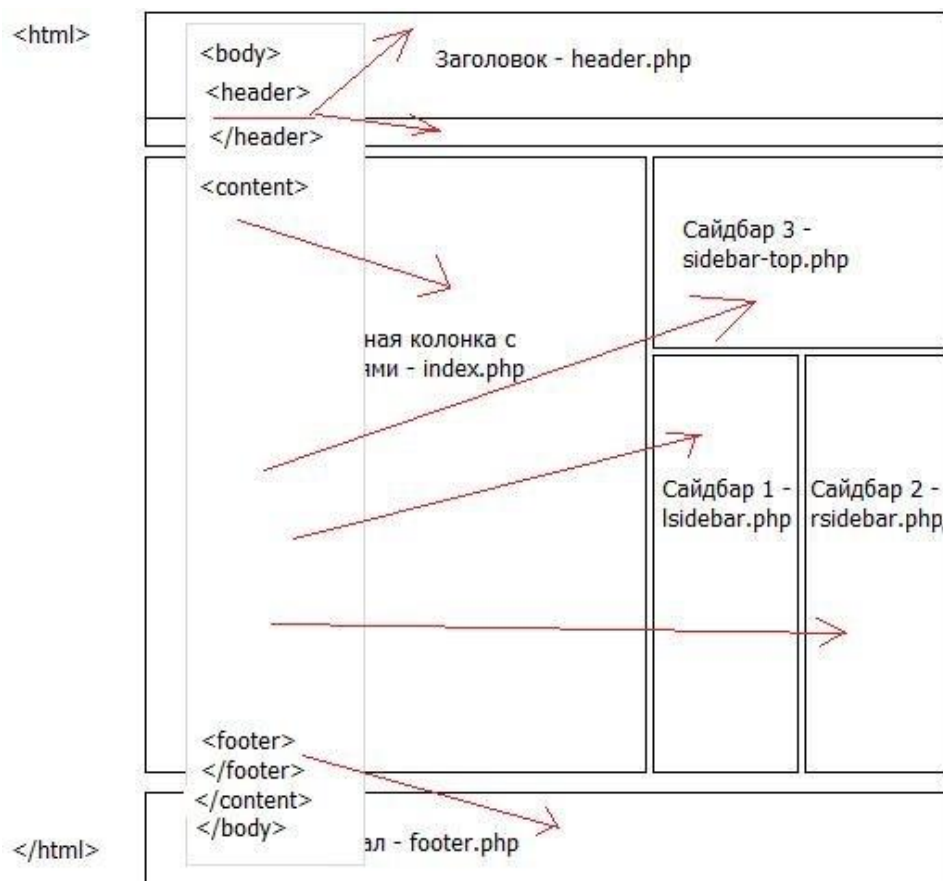
```
<div id="body">
```

Свойства, параметры

```
</div>
```

Выше – обычная стандартная конструкция “дива”. Подробнее вы можете почитать в самоучителях CSS, я не буду вас “морочить” всем этим, просто для, так сказать, вводной, хочу, чтобы вы понимали основные элементарные вещи, без знания которых не стоит даже открывать файлы и что-то там искать – не найдете. Но когда откроете и увидите то, о чем я вам сейчас рассказываю – вам станет легче .

Открытый и закрытый тег дива, между тегами – свойства и параметры этой секции или назовем его –блока, т.к. я рассказываю о блочной верстке. Вот так строится наш сайт. Основные дивы, с которых чаще всего начинается любой сайт – html, body, и так далее. Вот смотрите как это может выглядеть, на картинке ниже:



Пример пока не из нашего шаблона (я имею ввиду дивы), но дальше, когда будем разбирать файлы, будете вспоминать этот рисунок и вам станет намного понятнее.

Открывается тег (или див), затем идет код или свойство, закрывается тег (или див). Начало тега или дива обычно это слово, заключенное в скобки (<html>), закрывается тег или див – перед словом внутри скобок стоит слеш – </html>.

Все, что находится между этими тегами, и есть наша страница. Внутри большого дива – меньший, в который могут входить следующие, и так далее.

Можно сравнить с пазлами, или с набором прямоугольных элементов детской игры “паски”, сложенными один в один. В БОльший прямоугольник вкладываем меньшие, и так далее.

Посмотрите на скриншот выше – я примерно набросал вам вариант размещения некоторых условных для нашего шаблона тегов, последовательность написания которых показывает нам, как построен код страницы из тегов HTML.

Свойства построены немного по другому принципу. Например, есть див с именем header, и ему присвоены определенные параметры: ширина, высота, расположение, цвет и так далее. На странице сайта есть тег HTML под названием <header>, который имеет внутри себя еще какой-то код HTML, или функции PHP, а также свойства стилей CSS. Из этого набора и состоит наш заголовок. И так можно сказать о каждой части сайта, или файле.

Хочу еще раз повторить. Я не собираюсь вам пересказывать самоучитель по HTML или CSS. Если вам интересно изучить эти языки – пожалуйста, вот вам реально качественный ресурс – <http://htmlbook.ru/>. Здесь вы сможете найти все, кроме того, о чем я вам буду рассказывать дальше.

### Статья третья – изучаем файл header.php

Всем доброго времени суток!

Эта статья - продолжение моей серии публикаций на тему изучения файлов шаблона Wordpress. Сегодня речь пойдет о первом файле, с которого начинается наша главная страница сайта, а также все остальные страницы, которые выводят в окне браузера контент.

Напомню, что разбирать и анализировать мы будем файлы шаблона o2, который я буду показывать "изнутри" в качестве примера. Кто еще не скачал этот шаблон - [скачайте](#).

Итак. Если открыть файл в редакторе (советую пользоваться Notepad ++, у него работает подсветка кода, что очень удобно для чтения файла). Если нет - вот вам ссылка ([http://biblprog.org.ua/ru/notepad\\_plus/](http://biblprog.org.ua/ru/notepad_plus/)) на последнюю версию этой программы - 5.8.4.

Файл начинается с основного служебного кода, с которого начинается любой файл формата HTML - это так называемый элемент `<!DOCTYPE>` и предназначен он для указания типа текущего документа — DTD (document type definition, описание типа документа). Необходим этот элемент для того, чтобы браузер понимал, как следует интерпретировать текущую веб-страницу, поскольку HTML существует в нескольких версиях, кроме того, имеется XHTML (EXtensible HyperText Markup Language, расширенный язык разметки гипертекста), похожий на HTML, но различающийся с ним по синтаксису:

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
```

Следующая строка - это открытие тега `html`, в котором указан параметр задания пространства имен корневого элемента в HTML документе. Все это важно и нам надо просто об этом знать. Итак, тег HTML открыт и у нас началось самое интересное.



```

<title><?php bloginfo('name'); ?><?php wp_title(); ?></title>

<meta http-equiv="Content-Type" content="<?php bloginfo('html_type'); ?>;
charset=<?php bloginfo('charset'); ?>" />
<meta name="generator" content="WordPress <?php bloginfo('version'); ?>" />
<!-- leave this for stats please -->

<link rel="stylesheet" href="<?php bloginfo('stylesheet_url'); ?>"
type="text/css" media="screen" />
<link rel="alternate" type="application/rss+xml" title="RSS 2.0" href="<?php
bloginfo('rss2_url'); ?>" />
<link rel="alternate" type="text/xml" title="RSS .92" href="<?php
bloginfo('rss_url'); ?>" />
<link rel="alternate" type="application/atom+xml" title="Atom 0.3"
href="<?php bloginfo('atom_url'); ?>" />
<link rel="pingback" href="<?php bloginfo('pingback_url'); ?>" />

<?php wp_get_archives('type=monthly&format=link'); ?>
<?php //comments_popup_script(); // off by default ?>
<?php wp_head(); ?>
</head>

```

Тег title - функция PHP сообщает серверу, что необходимо вывести название сайта, которое вы задали на странице настроек: Параметры - > Общие. Здесь сразу хочу напомнить вам:

*Чем меньше запросов создает ваша страница к серверу, тем легче работает ваш сайт.*

Поэтому, если есть желание и настроение, давайте сделаем так. Сначала активируйте шаблон у себя на локальном сервере (правда, не у всех из вас он есть). Затем откройте главную страницу сайта с этим шаблоном и правой кнопкой мыши найдите в контекстном меню функцию - Исходный код страницы. Откройте его и вы увидите в начале страницы вот такой код:

```

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//E
<html xmlns="http://www.w3.org/1999/xhtml">
<head profile="http://gmpg.org/xfn/11">

<title>Коллекция Wordpress</title>

<meta http-equiv="Content-Type" content="text/html;
<meta name="generator" content="WordPress 3.0.1" />

```

Между тегами title стоит название сайта. Что нам мешает сделать руками то же самое? Ничего!

Давайте исправим файл header.php: удалим код функции PHP между тегами title и напишем также, как в примере выше (вы можете писать название своего сайта). Сделали. Сохранили. Перегрузили главную. Ничего не поменялось. Но! Сервер не подключается. Нет к нему запроса. Шаблон стал «чуть легче».

Следующие две строки файла после тега title:



```
<meta http-equiv="Content-Type" content="<?php bloginfo('html_type'); ?>; charset=<?php bloginfo('charset'); ?>" />
```

```
<meta name="generator" content="WordPress <?php bloginfo('version'); ?>" /> <!-- leave this for stats please -->
```

В первой строчке идет о кодировке нашего сайта. Она необходима всем браузерам правильно интерпретировать страницы сайта.

Вторая строка говорит всем о том, что на сайте стоит CMS WordPress и указывает ее версию. Я уже как-то писал, что этот мета-тег не обязателен и лишний раз показывает взломщику, на каком движке работает ваш сайт. Если вы посмотрите скриншот выше – видно, что сообщает исходный код. Стоит версия 3.0.1. WordPress.

Данный тег не обязателен и его можно удалить.

Следующий блок данных:

```
<link rel="stylesheet" href="<?php bloginfo('stylesheet_url'); ?>" type="text/css" media="screen" />
```

```
<link rel="alternate" type="application/rss+xml" title="RSS 2.0" href="<?php bloginfo('rss2_url'); ?>" />
```

```
<link rel="alternate" type="text/xml" title="RSS .92" href="<?php bloginfo('rss_url'); ?>" />
```

```
<link rel="alternate" type="application/atom+xml" title="Atom 0.3" href="<?php bloginfo('atom_url'); ?>" />
```

```
<link rel="pingback" href="<?php bloginfo('pingback_url'); ?>" />
```

Первая строка сообщает браузеру какой файл стилей использовать. По умолчанию используется стандартное имя – style.css и место файла в корне шаблона.

Следующие теги сообщают браузеру каким образом работают ссылки на ленту подписки, а также т.н. «пинги» с других сайтов.

Это служебные записи, они всегда есть во всех шаблонах WordPress.

Следующая строка – сообщение об обработке функции вывода архива.

Следующая строка сообщает, что всплывающее окно комментариев отключено по умолчанию. И последняя строка из приведенного выше блока – функция `wp_head`. Вот что о этой функции написано в Справочнике по функциям WordPress:

*Обязательно применяйте эту функцию перед закрытием тега HEAD. Она необходима для некоторых плагинов, которые вы используете в шаблоне (если нет этой функции – плагины могут не работать), а также для подключения дополнительных скриптов и стилей.*

Согласитесь, функция очень нужная. Пусть стоит.

Мы дошли до начала тега BODY. О нем и о остальном коде файла header.php речь пойдет в следующей статье.

#### Статья четвертая – файл header.php, продолжение.

Итак, я остановился перед тегом BODY в файле header.php шаблона, который мы изучаем – называется он o2.

Тег BODY – один из основных тегов HTML любого файла HTML или файла PHP для WordPress.

Как и у большинства файлов шаблона WordPress, этот тег начинается (открывается) в файле header.php а заканчивается чаще всего в файле footer.php. Внутри этого тега лежат все основные функции PHP нашего шаблона, а также все дивы файла стилей style.css.

Для изучения файлов PHP я советовал вам скачать и установить редактор кода Notepad ++ еще и потому (главная причина), что он подсвечивает не только код файла, он еще показывает нам где начинается и заканчивается тот или иной див. Смотрите картинку ниже. Вот кусок кода из файла, где после открытого тега BODY есть див HEADER. Если поставить курсор на открытии этого дива, но редактор покажет закрывающийся тег этого дива чуть ниже:



```
20 <body>
21 <div id="outer">
22 <div id="wrapper">
23
24 <div id="header"><div id="header_l"></div>
25 <a href="php echo get_settings('home'); ?" title="php bloginfo('name'); ?">
26 </a>
27
28 <div id="header_r">
29 <div id="header_r_ads">
30 </div>
31 <?php include (TEMPLATEPATH . "/searchform.php"); ?>
32 </div><!-- header_r -->
33
34 </div><!-- header -->
35
```

Нам сразу видно див HEADER, его начало и его конец, а также видно, какие еще классы свойств в него входят (в нашем случае дивы HEADER\_R, HEADER\_R\_ADS, а также видны все функции кода PHP шаблона WordPress).

Давайте их коротко рассмотрим.

После открытия тега BODY есть два открывшихся дива – OUT и WRAPPER. При наведении курсора дивы не подсвечиваются, значит, в файле header.php они не заканчиваются. В таких дивах обычно записываются свойства, общие для всей страницы, которая будет выведена в браузере.

Далее идет див HEADER, о котором я только что писал. В нем я вижу функцию вывода текстового заголовка:

```
<a href="<?php echo get_settings('home'); ?>" title="<?php bloginfo('name'); ?>"></a>
```

а ниже – подключение файла searchform.php:

```
<?php include (TEMPLATEPATH . "/searchform.php"); ?>
```

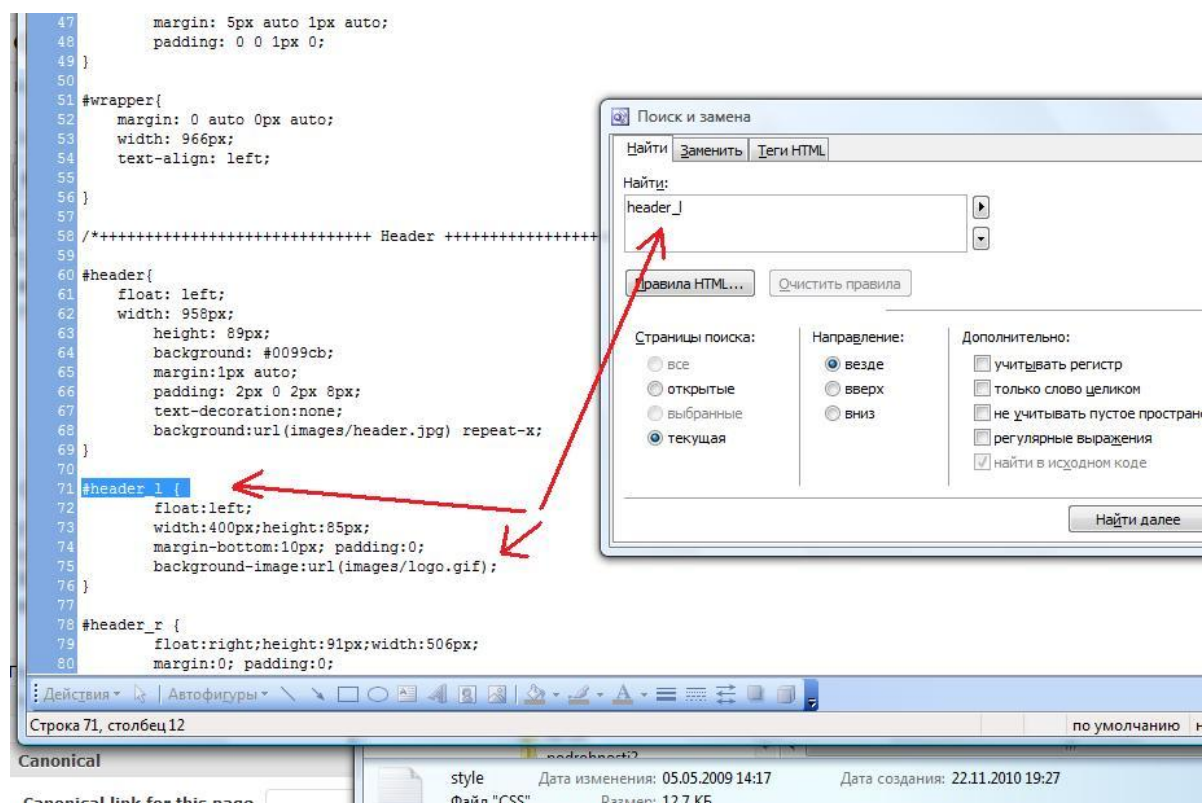
Такой код – стандартный при подключении файла шаблона в определенном месте (в разговорном языке еще называют инклюд). Имя файла говорит само за себя – подключается файл формы поиска на сайте. Давайте посмотрим главную страницу и поймем, где мы сейчас находимся:



Стрелками я показал, какой код мы сейчас рассмотрели: заголовок слева и окно поиска справа.

У нас в рассматриваемом шаблоне вместе текстового заголовка выводится Логотип-картинка. Значит, что функция вывода текстового заголовка скрыта картинкой. Давайте в этом убедимся. Еще раз посмотрим код (см. выше). В диве HEADER есть див HEADER\_L, который открывается и сразу закрывается. Это подсказывает, что он выводит картинку. Откроем файл style.css и найдем этот див.

Для просмотра и редактирования файлов CSS я привык использовать программу MS Front Page, она у меня открывается по умолчанию. Вы также можете использовать любой редактор, например тот же Notepad ++. Открыв программу, я через окно поиска ввожу нужный мне див – header\_l и программа подсвечивает его, смотрите картинку ниже:



Стрелками я показал окно поиска, найденный див и строку, в которой выводится картинка Логотипа.

Зачем так подробно я остановился на этом фрагменте кода файла header.php? Потому что я хочу вам показать как можно отказаться от использования функции и написать заголовок обычным текстом. Смотрите.

Я уже писал, что чем меньше запросов посылает PHP файл серверу, тем быстрее работает ваш сайт. Поэтому надо использовать любую возможность, чтобы отказаться от использования функций и перейти на обычный HTML.

Если мы перейдем на главную страницу и откроем исходный код, то увидим, что вывод заголовка оформлен так:

```

<body>
<div id="outer">
<div id="wrapper">

<div id="header"><div id="header_l"></div>
<a href="http://localhost/wp301" title="Коллекция Wordpress">
</a>

```

С помощью двух функций

```
<?php echo get_settings('home'); ?>" title="<?php bloginfo('name'); ?>
```

был послан запрос серверу, на что он ответил выводом текста заголовка – Коллекция WordPress.

Давайте откроем файл header.php и вместо этих функций напишем следующий код HTML:

```
<div id="header"><div id="header_1"></div>
```

```
<title>Коллекция WordPress</title>
```

Видите: вместо функций я поставил тег TITLE и внутри него написал заголовок сайта.

Теперь перегружаем главную страницу, открываем исходный код:

```
<body>
<div id="outer">
<div id="wrapper">

<div id="header"><div id="header_1"></div>
<title>Коллекция Wordpress</title>
```

Внешне ничего не изменилось. мы видим те же теги HTML, что и до внесения изменений. Но! Мы избавились от 2-х функций, а это даст нам чуть-чуть больше скорости работы нашего сайта.

Теперь что касается Логотипа. Мы можем заменить картинку на текст. Для этого достаточно в файле стилей style.css удалить строку вывода картинки (рассматривали выше). Но для вывода текста автор не прописал свойства. Поэтому здесь надо включать знания CSS и пробовать настроить цвет, стиль шрифта и его размер и т.д. Я это делать сейчас и здесь не буду. Я просто изучаю вместе с вами код файла header.php и рассказываю – что и как здесь работает и для чего.

В конце – пару слов о окне поиска на сайте. У нас сейчас там выводится английское слово search. Давайте его удалим и вставим русское – искать. Открываем файл searchform.php (который через инклюд был подключен для вывода в заголовке), найдем в коде формы это слово и заменим на русское – искать. Вот код файла:

```
<p style="text-align: justify;"><form id="search" method="get" action="<?php bloginfo('home');
?>/" ><input type="text" value="search" onfocus="if (this.value == 'search') {this.value = '');"
onblur="if (this.value == '') {this.value = 'search'};" size="18" maxlength="50" name="s" id="s"
/></form>
```

Меняем слово, получаем:

```
<form id="search" method="get" action="<?php bloginfo('home'); ?>/" ><input type="text"
value="искать" onfocus="if (this.value == 'искать') {this.value = '');" onblur="if (this.value == '')
{this.value = 'искать'};" size="18" maxlength="50" name="s" id="s" /></form>
```

Сохраняем изменения. Перегружаем главную страницу. Теперь у нас слово искать – на русском.

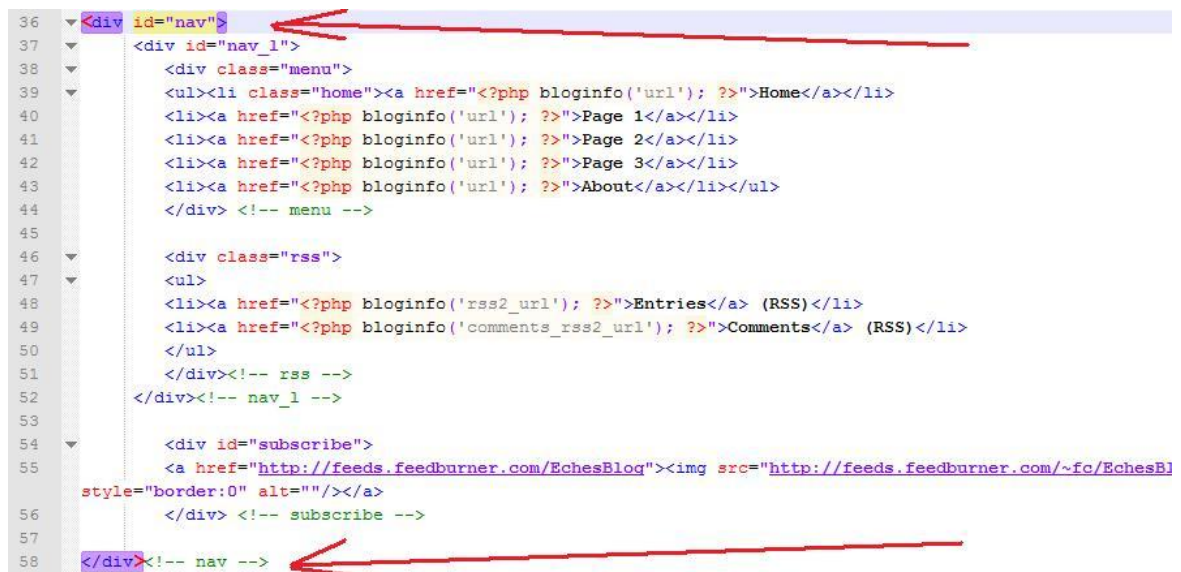


## Статья пятая – файл header.php, окончание

Всем доброго времени суток!

Сегодня я постараюсь закончить наше изучение файла header.php из шаблона o2. Напомню, что пишу цикл (или серию) статей на тему - изучаем файлы шаблона Wordpress. Кто только сейчас наткнулся на статью - советую начать с самого начала. Тогда мне не надо все повторно объяснять - зачем пишу и почему считаю, что это должно кому-то быть интересно.

Итак. У нас остался последний блок (div id="nav"). Судя по коду и имени "дива" - мы имеем дело с навигационным меню и блоком социальных закладок, расположенных в строке меню (см. главную страницу шаблона):



```

36 <div id="nav">
37   <div id="nav_1">
38     <div class="menu">
39       <ul><li class="home"><a href="<?php bloginfo('url'); ?>">Home</a></li>
40       <li><a href="<?php bloginfo('url'); ?>">Page 1</a></li>
41       <li><a href="<?php bloginfo('url'); ?>">Page 2</a></li>
42       <li><a href="<?php bloginfo('url'); ?>">Page 3</a></li>
43       <li><a href="<?php bloginfo('url'); ?>">About</a></li></ul>
44     </div> <!-- menu -->
45
46     <div class="rss">
47       <ul>
48       <li><a href="<?php bloginfo('rss2_url'); ?>">Entries</a> (RSS)</li>
49       <li><a href="<?php bloginfo('comments_rss2_url'); ?>">Comments</a> (RSS)</li>
50       </ul>
51     </div><!-- rss -->
52   </div><!-- nav_1 -->
53
54   <div id="subscribe">
55     <a href="http://feeds.feedburner.com/EchesBlog"></a>
57   </div> <!-- subscribe -->
58 </div><!-- nav -->
  
```

Но! Обратите внимание (советую смотреть не скриншот, а сам файл в редакторе Notepad++) - код вывода навигационного меню создан автором не как функция PHP, а именно как HTML. Почему я так решил? Потому что код PHP вызова меню навигации страниц в Wordpress выглядит так:

```
<?php dp_list_pages(); ?>
```

В Wordpress версии 3 - немного иначе, но сейчас мы не будем останавливаться на третьей версии. Вы видите, что автор вместо функции PHP написал код в формате HTML для вывода страниц в навигационном меню. Здесь вариант:

1. Мы можем оставить так, как написал автор. Только когда будем создавать свои страницы, их обязательно надо "прописать" в этом меню.
2. Можем удалить HTML - код автора и написать стандартную функцию PHP для вывода меню страниц.

### Мой комментарий вариантов:

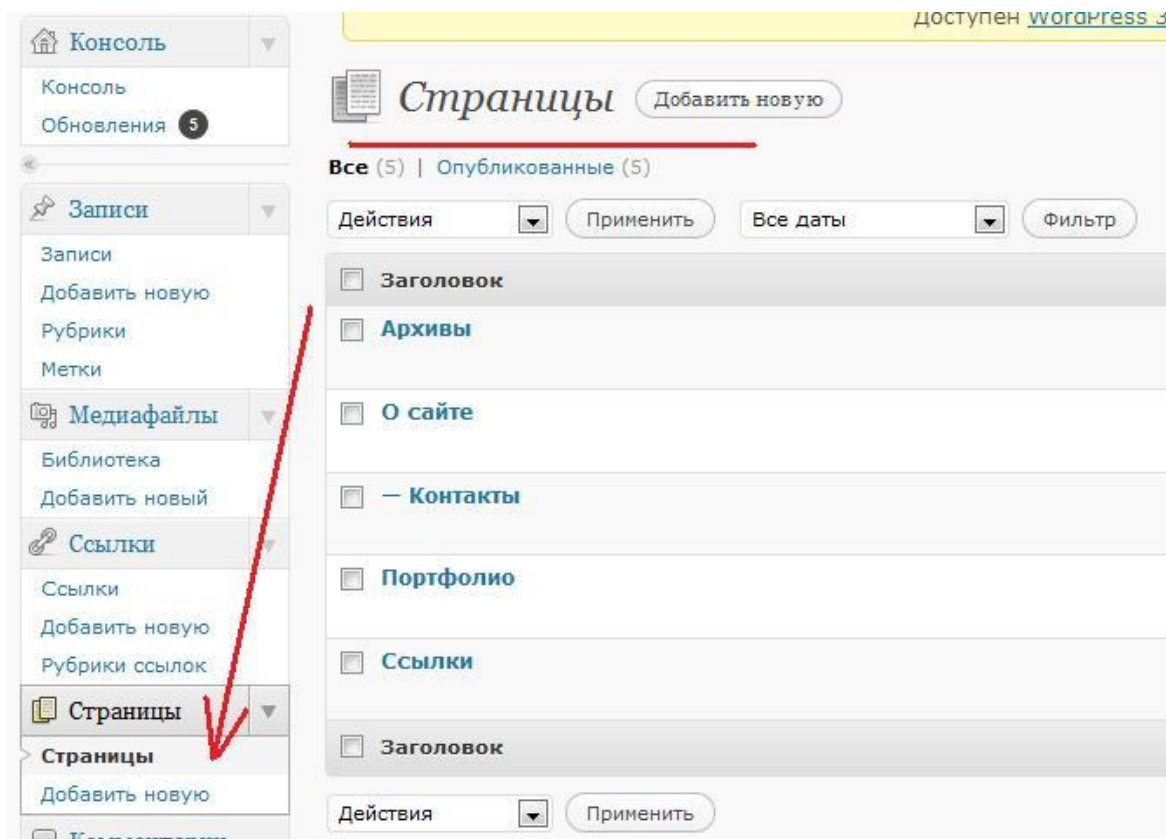
Первый вариант лучше. Он позволит нам отказаться от функции PHP и еще одного запроса к базе данных и серверу (я писал об этом раньше), что облегчит загрузку страниц нашего сайта и ускорит вывод ее в браузере посетителя.

Поэтому принимаю решение - оставить как есть, только внести коррективы, потому что автор написал пример кода:

```
<ul><li><a href="<?php bloginfo('url'); ?>">Home</a></li>
<li><a href="<?php bloginfo('url'); ?>">Page 1</a></li>
<li><a href="<?php bloginfo('url'); ?>">Page 2</a></li>
<li><a href="<?php bloginfo('url'); ?>">Page 3</a></li>
<li><a href="<?php bloginfo('url'); ?>">About</a></li></ul>
```

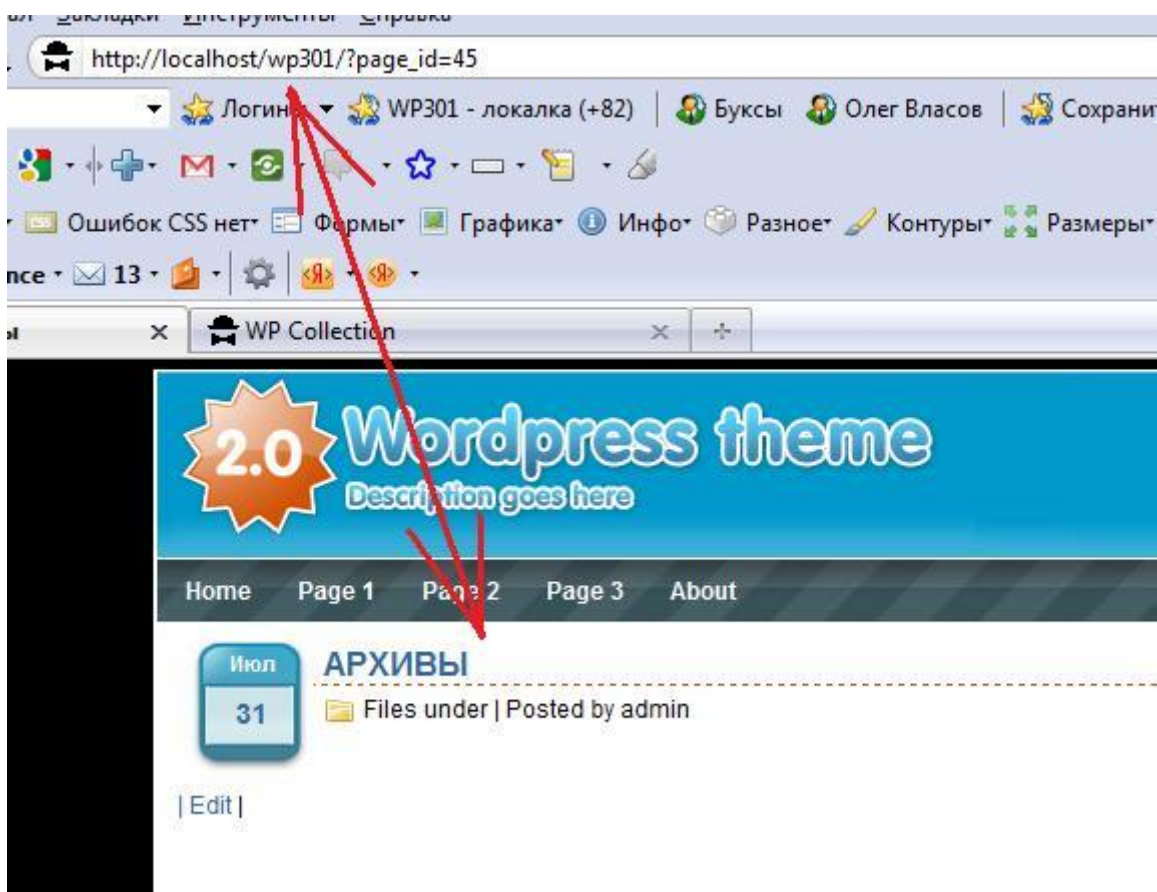
Первая строка вызывает загрузку главной страницы - home. Давайте напишем ее по-русски: Главная.

Теперь откроем список страниц нашего сайта. У меня он на локальном сервере сейчас такой:



Наведите курсор на страницу и увидите ссылку - Перейти. Откройте страницу в новом окне браузера (я открываю страницу Архивы). Откроется страница и вверху, в окне ссылок я вижу такой путь:





Полный путь, или еще говорят - URL страницы у меня такой - **http://localhost/wp301/?page\_id=45.**

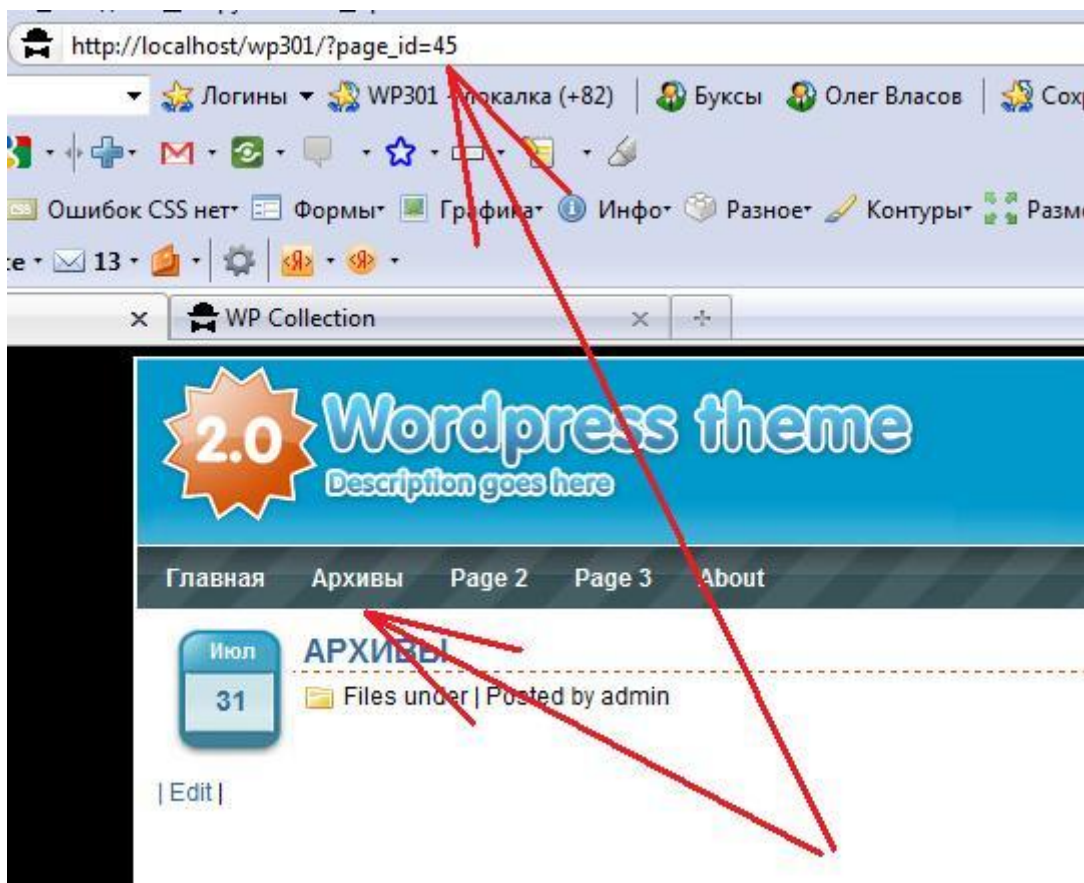
Теперь открываем файл header.php, строка 40:

```
<li><a href="<?php bloginfo('url'); ?>">Page 1</a></li>
```

и добавляем ссылку на страницу:

```
<li><a href="<?php bloginfo('url'); ?><strong>/?page_id=45</strong>">Page 1</a></li>
```

Заменяю слово **Page 1** словом **Архивы**. Сохраняю. Перегружаю главную страницу. Вижу:



Все работает. Главная страница и страница Архивы - "кликабельные", значит код написан верно.

Остальные страницы можно переделать таким же способом. Я у себя, вы - у себя.

Кстати, обратите внимание: это меню автора НЕ поддерживает вложенных страниц. Сам код мы можем изменить, но наверняка, что для дочерних страниц автор не предусмотрел в файле стилей style.css определенных свойств отображения, поэтому даже если мы напишем код для вывода вложенных страниц, все равно корректно работать не будет. Отдельно останавливаться на примере написания кода для внутренних страниц меню я сейчас не буду. Моя задача - рассказать о том шаблоне, который перед нами.

Следующий див, который мы рассмотрим - блок вывода подписки на статьи и комментарии:

```
<div>
<ul>
<li><a href="<?php bloginfo('rss2_url'); ?>">Entries</a> (RSS)</li>
<li><a href="<?php bloginfo('comments_rss2_url'); ?>">Comments</a> (RSS)</li>
</ul>
</div><!-- rss -->
```

Нам надо только заменить на русские слова: Entries - Записи, Comments - Комментарии. Функции вызова подписки написаны корректно.

Последний див и код в этом файле, которые мы рассмотрим, это вывод маленькой прямоугольной картинки, которая показывает количество подписчиков на ваш блог через систему Feedburner. Есть такой сайт, который примерно год назад выкупил под себя Google. Можно создать там аккаунт и приглашать посетителей подписаться на ваши новости и читать их не в формате RSS-ленты, а получать новости на почтовый ящик.

Процедура получения аккаунта немного запутана, как и все америкосовское, но есть русский интерфейс, т.ч. можно разобраться. Один из вариантов подписки: можно выводить по ссылке отдельное всплывающее окно, можно - просто поставить такую картинку с счетчиком подписавшихся и при кликании по картинке посетитель переходит на ваш подписной лист, где можно выбрать формат подписки. Код кнопки Feedburner дает сам Feedburner, ваша задача - изменить ссылку в коде и поменять аккаунт автора (сейчас там стоит именно его ссылка и подписчиков у него - 2350) на свой аккаунт. Итак, смотрим внимательно код:

```
<div id="subscribe">  
<a href="http://feeds.feedburner.com/<strong>EchesBlog</strong>"></a>  
</div> <!-- subscribe -->
```

Аккаунт автора, его ID - вот это: **EchesBlog (2 паза!)**. У меня есть такой же ID, но он выглядит так - **bestwp**. Я сейчас заменю на свой, сохраню и покажу вам скриншот. Если количество подписчиков изменилось (у меня сейчас по-моему 13), значит, я все верно сделал.

Было:



Стало:



Работает. Поверьте, я не нарисовал... :)

Все. С функциями и кодом PHP и HTML в файле header.php шаблона o2 я закончил. Теперь вы, надеюсь, немного лучше поймете как работает этот файл в вашем шаблоне и если есть похожие моменты - сможете что-то изменить к лучшему у себя в шаблоне. Не бойтесь пробовать - всегда у всех это бывает первый раз, и возможно, что-то сразу у вас не получится, пробуйте сделать еще и еще. Наверняка все выйдет.

Успехов!

Если у вас есть вопросы, советы, пожелания - пишите в комментариях, отвечу всем.

Хотите следить за новыми публикациями этой серии статей - **Изучаем файлы шаблона Wordpress** - подпишитесь. Кнопки подписки есть вверху справа, над сайдбаром.

### Статья шестая – Изучаем файлы шаблона: index.php

Я сегодня продолжаю серию статей, посвященную изучению файлов шаблона на примере темы o2. Свою серию статей я начал 17 ноября, и в пяти подряд статьях я рассмотрел с вами файл header.php, а также основные моменты макета нашего шаблона.

Советую тем, кому интересно: начните читать с самого начала, там вы найдете архив шаблона, а также много на мой взгляд полезной информации.

Сегодня у нас следующий файл, основной, и называется - index.php.

Кто-то знает, а кому-то из вас я скажу об этом впервые - это основной файл вашего шаблона. Именно он по умолчанию запускает все остальные файлы темы.

Поэтому в самом начале этого файла мы видим вот эту функцию:

```
<?php get_header(); ?>
```

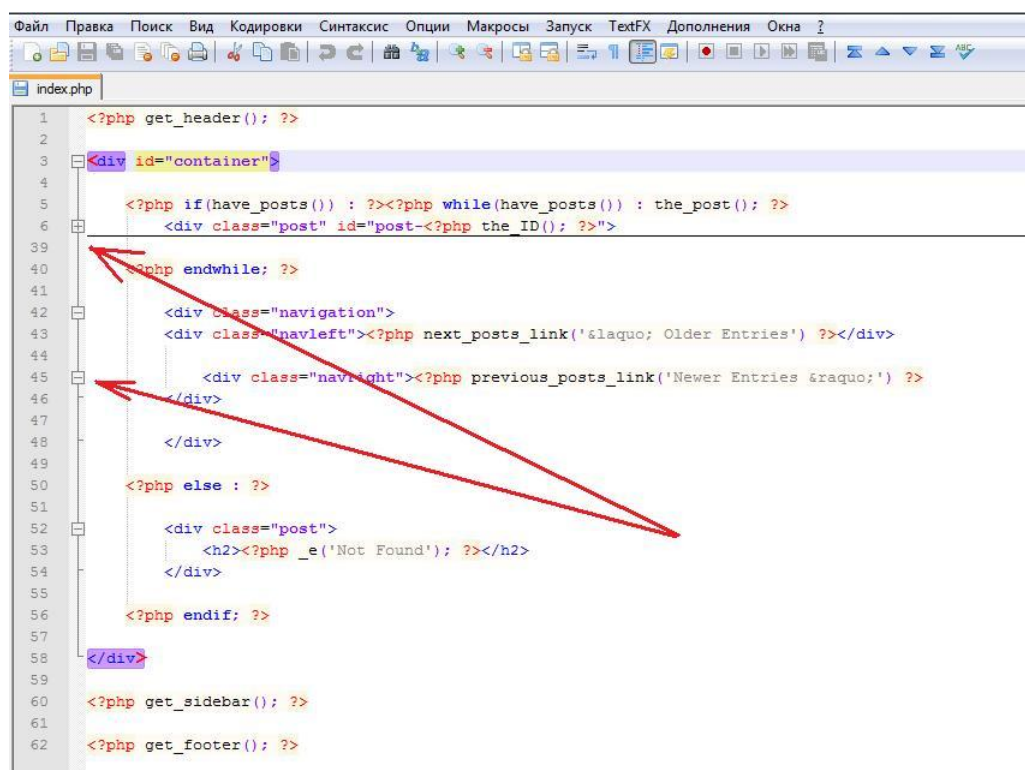
С ее помощью подключается наш файл заголовка (шапки) - header.php, который мы уже изучили и знаем, для чего он служит и какую информацию хранит.

Следующий блок - это начало (открытие) дива под именем 'container':

```
<div id="container">
```

О "дивах", свойствах и файле стилей я рассказывал в предыдущих статьях, повторяться не буду. Просто хочу вас попросить открыть этот файл в редакторе Notepad++ (или другом с подсветкой кода), поставить курсор в слово - div - и вы увидите, как подсветится весь контейнер этого "дива". Перейдите в его конец и вы поймете, что этот основной контейнер содержит в себе практически весь код файла index.php, за исключением функций в конце файла, о которых я скажу дальше.

На что еще хочу обратить ваше внимание. Если вы открыли файл в редакторе Notepad++ (не знаю, как у других редакторов), вы увидите слева в специальном столбце знаки "минус" и вертикальные линии. Если нажать курсором на "минус" - он закроется и вы увидите "плюс". Плюс - это закрытый "див". Закрывая любой див, вы уменьшаете количество выводимого в редакторе текста и так удобней работать с кодом файла. Также это помогает лучше понимать каждый "див" и его основные функции:



Ниже закрывающегося тега div id="container" записаны последние две функции файла - первая из них подключает вывод сайдбара, вторая - футера (подвала) нашего шаблона.

Таким образом, вы убедились, что наш файл index.php - основной. Он выводит все основные файлы нашего шаблона и в итоге мы получаем нашу страничку сайта.

А теперь давайте рассмотрим все функции, которые находятся внутри контейнера div id="container".



Первая, и главная, это цикл The Loop. Начинается он всегда вот так:

```
<?php if(have_posts()) : ?><?php while(have_posts()) : the_post(); ?>
```

Полный цикл The Loop выглядит следующим образом:

```
<?php if(have_posts()) : ?><?php while(have_posts()) : the_post(); ?>
```

```
<?php endwhile; ?>
```

```
<?php endif; ?>
```

Все, что вы найдете между первой строкой и второй - это касается вывода ваших записей.

Все, что вы найдете между второй и третьей строкой, может касаться вывода навигации и другой второстепенной информации.

Заканчивается цикл всегда как `<?php endif; ?>`.

Дословно, в [Документации Wordpress](#), вы можете прочитать, что:

***Цикл** используется в wordpress для отображения каждой записи. используя цикл, wordpress обрабатывает каждую из этих записей для вывода на текущей странице и форматирует ее в соответствии с указанными критериями внутри цикла. любой [html](#) или [php](#) код, расположенный внутри цикла, будет повторен для каждой записи. когда в документации wordpress говорится "этот тэг используется внутри цикла", как для конкретных [тэгов шаблона](#) или плагинов, тэг будет повторен для каждой записи.*

Вернемся к нашему файлу.

Из цитаты вы теперь понимаете, что все функции, которые находятся внутри цикла, выводят наши записи и будут повторяться столько раз, сколько записей на главной странице надо вывести. Если у вас в настройках шаблона (раздел Настройки -> Чтение) стоит число 10, значит, цикл выведет десять последних (по дате) записей.

В этом цикле нет никаких ограничений и дополнительных параметров. Поэтому он работает по умолчанию: выводит последние по времени создания записи, сверху вниз в порядке убывания даты публикации. Следовательно, вверху стоит самая последняя публикация, внизу - та, что вы писали ранее.

Следующая функция:

```
<?php the_ID(); ?>
```

Отображает числовое ID нашей записи.

Следующий контейнер - это див **post\_header**. Внутри него мы видим такой код:

```
<div>
<div><?php the_time('M') ?></div>
<div><?php the_time('jS') ?></div>
</div><!-- the_date -->
```

Функции вывода даты - месяц и день.

Дальше:

```
<div>
<div>
<h2><a href="<?php the_permalink(); ?>" title="<?php the_title(); ?>"><?php the_title();
?></a></h2>
</div>
```

Здесь функция вывода заголовка статьи, а также функция которая делает этот заголовок кликабельным. Зачем? Чтобы посетитель мог кликнув по названию, открыть страницу просмотра отдельной записи.

Совет: Иногда стоит эту функцию удалять, так как она фактически "дублирует" внутренние ссылки того же тега more.

Если у вас включен в записи тег more, или запись выводится отрывком и имеет дополнительную ссылку в виде - Читать Далее и тому подобное, стоит задуматься: а нужно ли для одной единственной записи две внутренние ссылки? Для SEO - точно не нужно. Поэтому иногда стоит функцию заголовка - <?php the\_permalink(); ?> - просто удалить.

Следующий контейнер:

```
<div>
<span><?php _e('Files under'); ?> <?php the_category(' ') ?></span> |
<span><?php comments_popup_link('Leave a Comment', '1 Comment', '% Comments');
?></span>
</div>
```

Первая строка - попытка автора воспользоваться конструкцией для корректного перевода (так и не работает) слов Files under, что можно вольно перевести как **Файл лежит в рубрике**, и дальше - функция вывода той рубрики, в которой лежит (прикреплена) эта статья. Давайте просто удалим эти слова. Сохраним страницу и посмотрим что там. Вместо вывода слов Files under теперь рядом с значком папки - имя рубрики. Давайте переведем следующую строку, где стоит функция comments\_popup\_link, которая сообщает нам (странице) ссылки на комментарии. Заменим английские слова на русские, примерно так:



```
comments_popup_link('Добавить комментарий', '1 комментарий', '% коммент.')
```

Сохраняем. Смотрим:



Стрелкой и линией подчеркивания я показал вам, где мы коснулись перевода.

На сегодня пока все. Мы остановились перед началом вывода основного контента.

Дальше будет еще интересней. Следите за публикациями. Попробую продолжить завтра. Лучше, если просто подпишитесь за новости моего сайта на RSS или на почту. ссылки есть вверху сайдбара.

### Статья седьмая – Продолжаем изучать index.php

Вчера я остановился перед началом кода, который выводит контент:

```
<div class="entry">
<?php the_content(); ?>
<p class="postmetadata">
<?php edit_post_link(' Edit', ' | '); ?><?php _e('Posted by'); ?> <?php the_author(); ?> |
<?php comments_popup_link('No Comments &#187;', '1 Comment &#187;', '% Comments &#187;'); ?>
</p>
</div>
```

Смотрим. Первая строка - открытия "дива" **entry**, который отвечает за свойства нашей записи (если интересно - можете открыть файл style.css и найти этот див там).

Затем - функция вывода самого контента, записи. Функция стандартная, выводит всю запись целиком (не отрывок), при этом в ее "конструкции" нет параметров (а бывают) типа "Далее". Значит, будет выводиться этот параметр по умолчанию из настроек самого Wordpress (тег more).

Далее - вывод линка **Редактировать** (Edit), затем - с помощью функции локализации языка текст **Posted by** (я писал - локализация автора не работает), затем - функция линка на автора статьи. И последняя конструкция - функция обработки комментариев.

Так как я обещал вам одновременно делать перевод, то все очень просто. Надо изменить английские слова и поставить русские варианты. Вот что должно получиться:

```
<div>
<?php the_content(); ?>
<p>
<?php edit_post_link('| Изменить', ' ' | '); ?><?php_e('Опубликовал'); ?> <?php the_author(); ?> |
<?php comments_popup_link('Нет комментариев &#187;', '1 комментарий &#187;', '% коммент. &#187;'); ?>
</p>
</div>
```

Следующий блок кода:

```
</div>
<?php endwhile; ?>
<div class="navigation">
<div class="navleft"><?php next_posts_link('&laquo; Older Entries') ?></div>
<div class="navright"><?php previous_posts_link('Newer Entries &raquo;') ?>
</div>
</div>
<?php else : ?>
```

Первая строка в этом блоке - закрывающийся тег "дива" **post**, затем - функция цикла The Loop (рассказывал о нем вчера), затем - "див" класса **навигация**, и в нем - два класса свойств для навигации слева и справа (Старые записи - Новые Записи).

Затем идет закрытие ранее открытых **дивов**. И в конце - функция цикла The Loop, которая буквально говорит, что если еще что-то есть - надо обработать. А под **еще** дальше мы видим такой код:

```
<div class="post">
<h2><?php_e('Not Found'); ?></h2>
</div>
```

Здесь понятно: если ничего не найдено - будет видны эти слова - "Не найдено". Давайте заодно и их переведем.

В конце файла index.php мы видим две последние функции:

- `get_sidebar` - вызывает (подключает) сайдбар, sidebar.php.
- `get_footer` - вызывает (подключает) footer.php

Все. Файл index.php на этом заканчивается. Мы сделали нужный нам перевод. Сохранили все правильно (без BOM!) и смотрим главную страницу:



Вчера наш перевод этого файла коснулся мета-данных сверху статьи, сегодня - внизу.

Завтра постараюсь продолжить. Мой рассказ пойдет о следующем важном файле нашего шаблона - sidebar.php.

Я обязательно рассмотрю все файлы, в первую очередь основные. Вы же понимаете, что цель моей серии статей о изучении файлов шаблона Wordpress - не изучать данный конкретный шаблон, который я дал вам скачать и мы вместе его "ковыряем". Я хочу помочь вам лучше понимать свой шаблон, который вы используете или планируете использовать.

В вашем шаблоне наверняка многое будет по-другому. Но основное - практически одно и тоже.

Поэтому советую вместе со мной двигаться дальше. В конце вы сами удивитесь, как много нового для вас открылось.

### Статья восьмая – Файл шаблона Wordpress: sidebar.php

Вчера я закончил изучать с вами файл index.php из шаблона o2. Пообещал продолжить с файла sidebar.php. Сегодня открыл его - а он оказался очень коротким. Лишний повод вам сделать вывод, что пишу я эту серию статей заранее к ней не готовясь.

Чтобы новичкам, кто открыл этот пост, было понятно, о чем идет речь - справа сверху в самом начале поста - серия статей. И начинается все с самого начала. Я пытаюсь внятно и доходчиво рассказать тем, кто этого хочет, из каких файлов состоит шаблон Wordpress, почему именно из них, зачем они нужны и какой код в себе несут, что делают, и почему именно так а не иначе.

Кто только пришел - читайте с самого начала, тогда вам станет понятно и я надеюсь - интересно.

А с остальными - продолжаю.

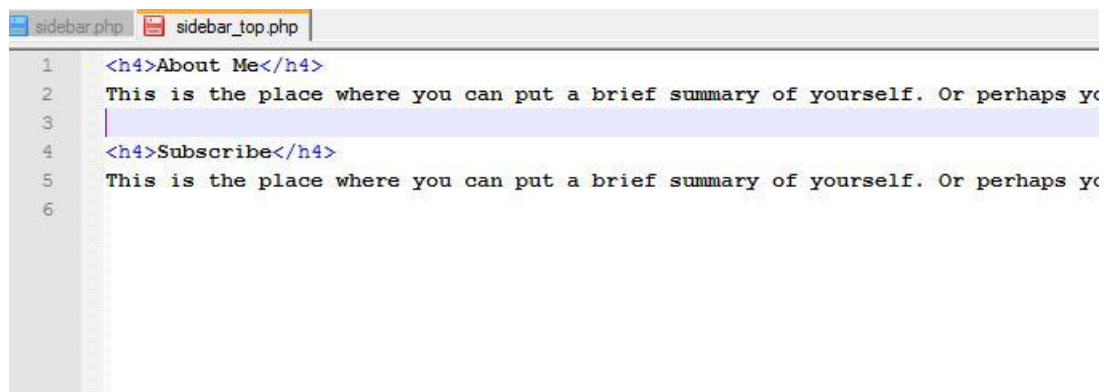
Откроем файл sidebar.php и увидим вот такой код:

```
<div class="sidebar">
<div id="sidebar_top">
<?php include (TEMPLATEPATH . "/sidebar_top.php"); ?>
</div>
<div class="lsidebar">
<?php include (TEMPLATEPATH . "/lsidebar.php"); ?>
</div>
<div class="rsidebar">
<?php include (TEMPLATEPATH . "/rsidebar.php"); ?>
</div>
</div>
```

Вы видите, что есть основной "див", называется **sidebar**, внутри которого - еще три "дива": sidebar\_top, lsidebar, rsidebar, что даже не посвященному становится понятно: в шаблоне есть три файла: сайдбар сверху (top), сайдбар справа и сайдбар слева.

Все три файла подключены с помощью стандартной функции подключения файла, по-русски часто пишут - **инклюд**, от слова в функции PHP - include.

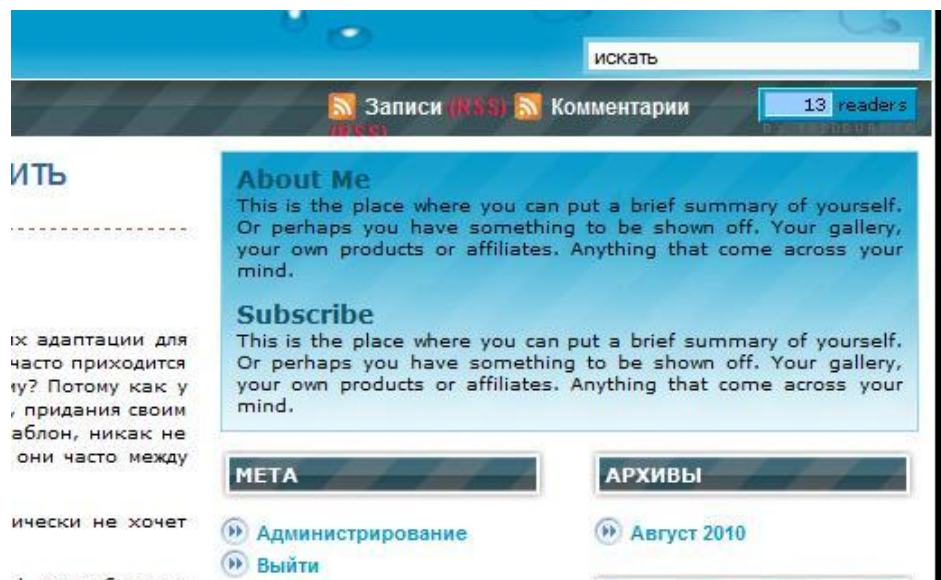
Теперь давайте по-порядку, начнем с файла sidebar\_top.php. Открываем, смотрим:



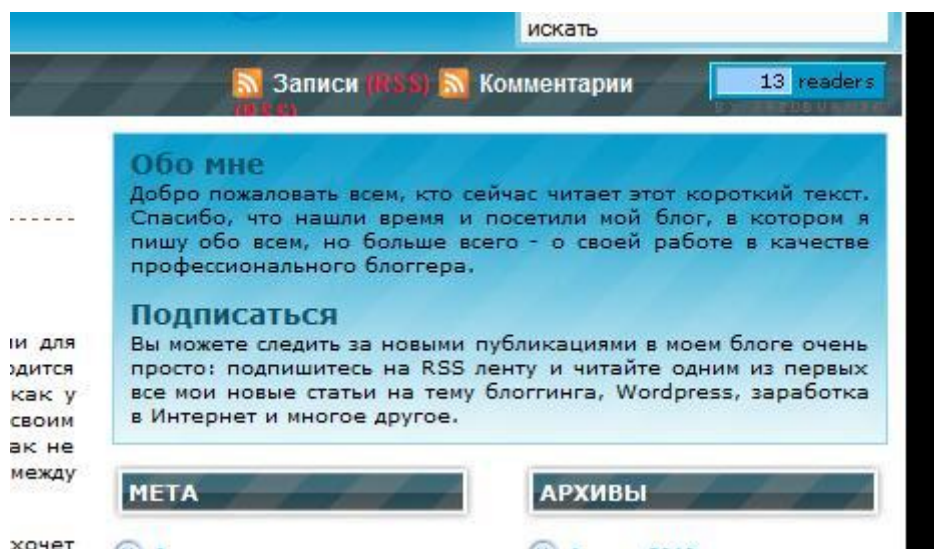
```
1 <h4>About Me</h4>
2 This is the place where you can put a brief summary of yourself. Or perhaps y
3
4 <h4>Subscribe</h4>
5 This is the place where you can put a brief summary of yourself. Or perhaps y
6
```

Здесь все просто как три копейки. В этом файле выводится небольшой блок, который озаглавлен как About Me, что значит - Обо мне, и идет небольшой кусок текста, слов на 30, и второй блок, который озаглавлен - Подписка. И тоже немного текста.

Давайте откроем главную страницу шаблона и посмотрим:



А вот и наш текст сверху над сайдбаром. Теперь можно немного подумать и написать текст на русском. Переходим в файл, пишем, сохраняем и перегружаем главную:



Где-то так. У вас может быть по-другому. Главное - не увлекайтесь и не пишите много. До 30 слов вполне достаточно.

Закрываем файл и переходим к следующему - `lsidebar.php`, или левый сайдбар. Открываем и смотрим:

```
<ul>
<?php if ( function_exists('dynamic_sidebar') && dynamic_sidebar(1) ) : else : ?>
<li><h2><?php_e('Categories'); ?></h2>
<ul>
<?php wp_list_cats('sort_column=name&optioncount=1&hierarchical=1'); ?>
</ul>
</li>
<?php wp_list_pages('depth=3&title_li=<h2>Pages</h2>'); ?>
<?php get_links_list(); ?>
<?php endif; ?></ul>
```

Отличный пример сайдбара! Ничего лишнего и все есть для того, что бы на примере этого файла немного рассказать вам о еще кое-что...

Во-первых, поговорим сначала о коде. Итак. Открытый тег `<ul>` - это начало списка. Можете сразу посматривать на главную, на левый сайдбар, чтобы понимать, о чем идет речь.

Этот тег расставляет наши слова в столбик. Второй тег - `<li>` - добавляет слева от слов значок. Если рубрика или ссылка вложенная (как на примере подрубрика 1) - у вложенной рубрики свое свойство в файле стилей и вы видите пунктирные линии.

На второй строке - функция, которая делает сайдбар динамичным, то есть этот сайдбар поддерживает динамично подключаемые виджеты из панели управления. Все, что находится между началом этой функции (вторая строка) и ее концом (третья снизу) - находится внутри динамично подключаемого контента, и в случае, если вы захотите добавить какой-нибудь виджет - все что внутри перестанет выводиться.

Если вы вынесете начало функции вниз, и поставите рядом с окончанием выше на строку, тогда у вас блоки. которые автор прописал по умолчанию, станут постоянными. Надеюсь, вы поняли меня.

Следующая строка - вывод заголовка Рубрики и ниже - функция вызова наших рубрик, в которой есть параметры вывода.

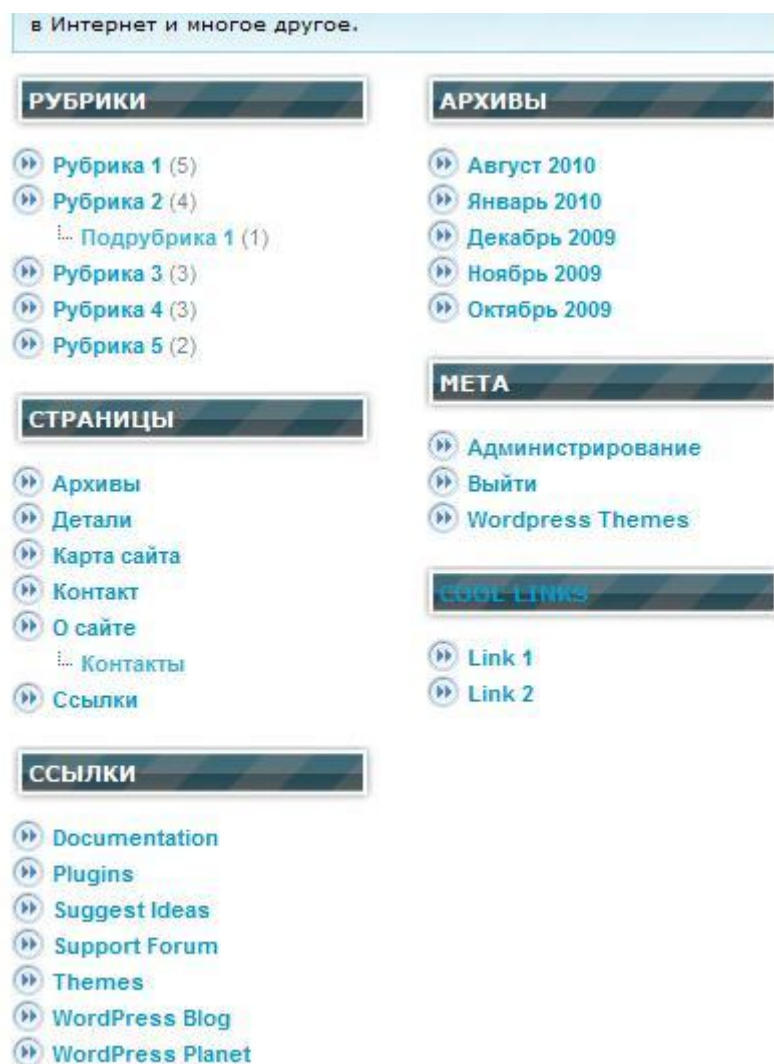
Ниже - тоже самое, только страницы.

Функция `get_links` - вызывает блок ссылок.

И последняя - это окончание функции динамичного сайдбара.

Давайте переведем оба слова на русский. Перевели, сохранили. Смотрим главную:





На сегодня все. О правом сайдбаре расскажу завтра или послезавтра. Там немного сложнее, чем было с левым.

## Статья девятая - Изучаем файлы шаблона: правый сайдбар

Всем привет!

Ну что, продолжим? Это уже девятая статья в серии - **Изучаем файлы шаблона Wordpress** - и разговор в ней пойдет о правой сайдбаре шаблона o2, который я взял в качестве примера изучения темы wordpress.

Зачем это надо? Кому? Где скачать архив шаблона и начать вместе с другими читать и изучать файлы? Найдите первую статью серии и начинайте читать.

Итак. Откроем файл rsidebar.php и посмотрим, что там (честное слово - сам еще не смотрел :)):



```
<ul>
<?php if ( function_exists('dynamic_sidebar') && dynamic_sidebar(2) ) : else : ?>
<li><h2><?php _e('Archives'); ?></h2>
<ul>
<?php wp_get_archives('type=monthly'); ?>
</ul>
</li>
<li><h2><?php _e('Meta'); ?></h2>
<ul>
<?php wp_register(); ?>
<li><?php wp_logout(); ?></li>
<li><a href="http://www.allmythemes.com">Wordpress Themes</li>
<?php wp_meta(); ?>
</ul>
</li>
<li><h2>Cool Links</h2><ul>
<li><a href="#">Link 1</a></li>
<li><a href="#">Link 2</a></li>
</ul>
</li>
<?php endif; ?>
</ul>
```

#### Замечание:

Я после копирования и вставки сюда кода из файла все теги и функции "подгоняю" под первый знак каждой строки. Когда вы открываете файл следом за мной - код располагается иначе. Такое расположение как в файле очень удобно для просмотра кода файла, особенно, если редактор не поддерживает подветку кода, но - такое расположение кода не годится для работы. Самый идеальный вариант - весь код расположить в одну строку.

Есть специальные программы, которые могут сгенерировать новый файл и расположить в нем весь код максимально сжато, что удобно для сервера и экономит немного его, сервера, время на выполнение команд PHP и MySQL.

Код файла правого сайдбара очень похож на код левого сайдбара. Та же функция которая делает сайдбар "динамичным", в смысле - дает возможность подключить динамичные виджеты из админки.

Затем - код вывода Архива - `wp_get_archives('type=monthly')`.

Затем - код вывода ссылок на управление сайтом (т.н. Мета).

Затем идет нестандартный для Wordpress метод вывода ссылок. Здесь автор использовал не встроенную функцию вывода ссылок как в файле `lsidebar.php - get_links_list()` - а обычные теги HTML. И при этом, похоже, посредник, где я скачал этот шаблон, воткнул сюда ссылку на свой ресурс:

```

<li><h2>Cool Links</h2>
<ul>
<li><a href="#">Link 1</a></li>
<li><a href="#">Link 2</a></li>
</ul>

```

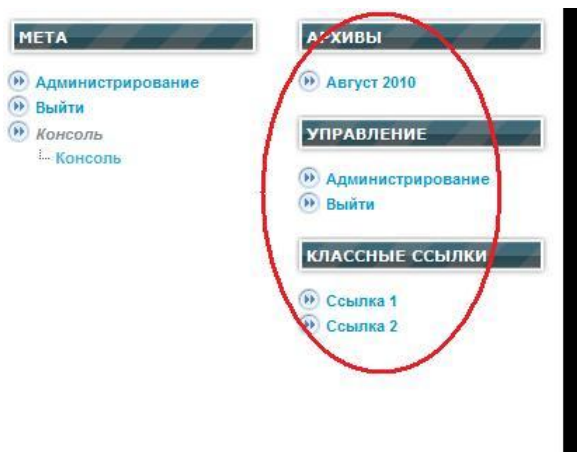
Я говорю вот об этих строках кода. Решетка в ссылке - # - частый прием разработчиков. Они ведь не знают, какую ссылку вы захотите здесь использовать.

Мы можем также перевести все английские слова, чтобы наш сайдбар заговорил с нами по-русски. Переводим:

- Archive - Архивы
- Meta - Управление
- Cool Links - Классные ссылки
- Link 1 - Ссылка 1
- Link 2 - Ссылка 2

Удаляем ссылку на спонсора, весь код строки с словами Wordpress Themes.

Сохраняем, открываем страницу. Смотрим:



Все в порядке.

Можно, конечно, заменить код HTML вывода ссылок на стандартный код Wordpress, но я предлагаю не трогать. Тем более, при подключении виджетов этот код исчезнет.

С файлами сайдбаров я закончил. Давайте быстро посмотрим файл footer.php, думаю, что он не займет у нас много времени.

Открываю. Вот как раз пример того, о чем я рассказывал выше: весь код одной строкой, что страшно неудобно для изучения, но очень хорошо для сервера, на котором все это работает.

Я немного отредактирую файл для удобства просмотра. Вы можете тоже попробовать. Сделал. Открываю:

```

<div id="footer">
<p>
Copyright &#169; <?php echo date("Y") ?> <a href="<?php bloginfo('url'); ?>"><?php bloginfo('info'); ?></a> <br />
<a href="http://blog.eches.net/themes">o2 2.0</a> : Designed by <a href="http://blog.eches.net">eches</a>
Brought by <a href="http://www.citizendia.org/">encyclopedie</a> & <a
href="http://www.debtconsolidation3.com">Debt Consolidation</a> &
<a href="http://www.myetymology.com">etymology</a>
</p>
</div><!-- footer -->

</div><!-- wrapper -->
</div><!-- outer -->
</body>
</html>

```

За что хочется сказать отдельное спасибо разработчику, так это за его корректное написание кода HTML. Очень удобно и все видно. Например, окончания "дивов" в конце он закомментировал и дал понять, какие дивы закрылись. Это как раз те дивы, о которых я рассказывал, когда мы смотрели и изучали файлы header.php и index.php. Как раз там эти дивы были открыты, и закрылись здесь.

Все-таки давайте по-порядку, сверху начнем.

Сначала идет строка вывода слова Copyright, функция вывода года, вывод названия сайта.

Следующая строка - похоже, это сайт самого автора.

Затем - несколько ссылок на спонсоров (или автора, или посредника).

Можете его удалить, если хотите. Я удаляю. Вот что у меня в итоге получилось:

```

<div id="footer">
<p> Copyright &#169; <?php echo date("Y") ?> <a href="<?php bloginfo('url'); ?>"><?php bloginfo('info'); ?></a>
<br />
Дизайн - <a href="http://blog.eches.net">eches</a>

</div><!-- footer -->
</div><!-- wrapper -->
</div><!-- outer -->
</body>
</html>

```

Открываем главную страницу, перегружаем ее и смотрим подвал:



Теперь все нормально.

На сегодня все. Продолжение скоро будет.

## Статья десятая - Продолжаем изучать файлы шаблона Wordpress

Всем доброго времени суток и с Новым Годом!

Надеюсь, что вы все отлично провели праздничные дни и неплохо отдохнули. Кто-то дома на диване, кто-то - в веселой компании. Кто-то - за компьютером.

Сегодня я продолжаю серию статей, посвященную изучению стандартных файлов шаблона Wordpress. В качестве примера я взял шаблон o2, автор - eches, сайт - [blog.eches.net](http://blog.eches.net).

Сейчас зашел в блог автора, действительно, лежит там такой шаблончик. Сделал он его еще в ноябре 2007 г. Но работает замечательно и на третьей версии Wordpress. Пока я глюков не заметил.

Кто не читал первый девять статей - советую начать сначала. А для остальных - продолжаю.

Итак, я рассмотрел следующие файлы шаблона:

- header.php
- index.php
- footer.php
- sidebar.php
- sidebar-top.php
- lsidebar.php
- rsidebar.php

Напомню, что эти файлы - основные. Они практически формируют главную страницу сайта.

Но у каждого Wordpress - шаблона есть еще несколько разного рода страниц. Например, страница просмотра отдельной записи, страница поиска, страница просмотра архивных записей, страница ошибки 404. С помощью каких файлов эти страницы выводятся?

Чтобы абсолютно качественно понимать и разбираться в структуре файлов шаблона Wordpress, советую поискать в Сети блоги, в которых очень грамотно и детально описана структура шаблона: какие файлы и за какие задачи отвечают. Тогда вам многое станет понятным. Практически все о своем шаблоне, который у вас установлен.

Ну, а прочитав мой мануал, вы сможете понимать, какой код PHP или HTML выводит ту или иную информацию на страницах вашего сайта, что можно подправить или исправить. Конечно, я не смогу и не собираюсь научить вас коду PHP или HTML, для этого есть много других источников в Сети.

Мне просто захотелось вам помочь лучше понимать файлы вашего Wordpress - шаблона.

Остальные файлы, которые я еще не рассмотрел, предлагаю изучать теперь в порядке английского алфавита, так, как вы их видите в папке.

Первый - это файл 404.php. Не открывая, сразу скажу, что задача этого файла - вывести на экран страницу ошибки 404. Эта ошибка случается тогда, когда посетитель попытался зайти на несуществующую страницу вашего сайта. Как это происходит, откуда берется несуществующая страница? Например, вы решили сменить у себя на сайте настройки постоянных ссылок. Было так: `http://www.мой_сайт.ru/?p=123`, а вы поменяли на - `http://www.мой_сайт.ru/archives/123`.

Яндекс или Гугл успели проиндексировать ваши страницы в первом варианте, а второй вариант для них не существует. Поэтому посетитель, переходя с поисковика к вам на сайт, попадет на страницу ошибки 404. И от того, как эта страница написана автором, зависят дальнейшие шаги посетителя.

Если он увидит пустое место и текст: "К сожалению, вы попали на пустую страницу." Он скорее всего, просто уйдет с вашего сайта. Если автор написал примерно так: "Извините, вы попали на страницу, которая скорее всего не существует или перемещена по новому адресу, потому что сменил постоянные ссылки. Советую воспользоваться поиском на сайте, или посмотреть страницу Архив, или посмотреть меню навигации. Возможно, это вам поможет найти то, что вы ищете." Есть вариант, что посетитель воспользуется одним из вариантов и попытается найти то, ради чего он сюда пришел.

А теперь давайте откроем файл 404.php и посмотрим, что написал здесь автор шаблона:

```
<?php get_header(); ?>
<div id="container">
<h2>Ooops: 404 page</h2><br />
The page you are looking for cannot be found. Please <b>contact</b> the author so that he can fix the problem. In
the meantime you might want to browse through this blog via <b><a href="<?php bloginfo('url'); ?>">main
page</a></b> to find latest posts.
<br /><br />
Sorry for the inconvenience.
</div>
<?php get_sidebar(); ?>
<?php get_footer(); ?>
```

Как видите, примерно так он и поступил. В вольном переводе фраза звучит примерно так:

Упс... Вы попали на страницу 404.

Страница, которую вы ищете, отсутствует. Вы можете сообщить автору, чтобы он исправил эту проблему, или перейти на главную и посмотреть последние публикации.

Чтобы увидеть эту страницу в работе, вы можете в адресной строке браузера, где сейчас открыта главная страница с этим шаблоном, написать некорректную ссылку, например: `http://localhost/wp301/?p=555` (я набрал несуществующий id 555), и вы увидите эту страницу в работе:



Вы можете заменить английский текст своим, взяв в качестве примера то, что я писал выше.

Следующий файл по списку - comments.php. Чаще всего, в нем больше всего кода, если говорить о стандартных шаблонах Wordpress.

Я сейчас начну его описание, а также покажу вам те куски кода, которые необходимо перевести на русский.

```
<?php // Do not delete these lines
if ('comments.php' == basename($_SERVER['SCRIPT_FILENAME']))
die ('Please do not load this page directly. Thanks!');
if (!empty($post->post_password)) { // if there's a password
if ($_COOKIE['wp-postpass_' . COOKIEHASH] != $post->post_password) { // and it doesn't match the cookie
?>
<p class="nocomments">This post is password protected. Enter the password to view comments.<p>
<?php
return;
}
}
/* This variable is for alternating comment background */
$oddc comment = 'odd';
?>
```

Сразу скажу, что абсолютное большинство файлов comments.php очень похожи один на другой. Из личного опыта. Поэтому наверняка код вашего файла очень похож на этот.

В самом начале выводится функция PHP, задача которой - сообщить посетителю, что просмотр комментариев защищен паролем, и чтобы их просмотреть - необходимо ввести свой пароль.

Здесь перевести надо коротких два предложения: **This post is password protected. Enter the password to view comments.**

Следующая функция - счетчик пингов и трекбеков:

```
<?php
/* Count the totals */
$numPingBacks = 0;
$numComments = 0;
/* Loop through comments to count these totals */
foreach ($comments as $comment) {
    if (get_comment_type() != "comment") { $numPingBacks++; }
    else { $numComments++; }
}
?>
```

Здесь ничего переводить не надо.

Далее:

```
<?php
/* This is a loop for printing comments */
if ($numComments != 0) : ?>
<h2 id="comments">
<br /> <?php comments_number('No Responses', 'One Response', '% Responses' );?>
to &#8220;<?php the_title(); ?>&#8221;</h2>

<ol class="commentlist">
<?php foreach ($comments as $comment) : ?>
<?php if (get_comment_type()=="comment") : ?>
```

Здесь функция определяет количество комментариев и выводит вам сообщение. Перевести надо те слова, которые находятся в круглых скобках (строка 05).

## Статья одиннадцатая - Файлы шаблона Wordpress: комментарии

Всем доброго времени суток и с Рождеством!

Сегодня я продолжу рассказ о файле comments.php. Кто читает не с самого начала - советую перейти на первую статью этой серии. Тогда вам будет понятно - о чем разговор, на примере какого шаблона я веду свой обзор. Там же, в начале, вы сможете скачать архив шаблона и вместе со мной следить за происходящим.

Следующий блок кода из файла comments.php



```

<ol>
<?php foreach ($comments as $comment) : ?>
<?php if (get_comment_type()=="comment") : ?>

<li"") echo 'mycomment'; else echo $oddccomment; ?>" id="comment-<?php comment_ID() ?>">

<p style="margin-bottom:5px; border-bottom:1px #fff dotted; padding-bottom:5px;">By <strong><?php
comment_author_link() ?></strong> on <a href="#comment-<?php comment_ID() ?>" title=""><?php
comment_date('M j, Y') ?></a> | <a href="#respond">Reply</a><?php edit_comment_link('Edit', ' | '); ?></p>
<?php if ($comment->comment_approved == '0') : ?>
<em>Your comment is awaiting moderation.</em>
<?php endif; ?>
<?php comment_text() ?>
</li>

```

Эта часть кода файла служит для вывода комментариев ваших посетителей. Обратите внимание на строку, где есть слова: By (автор), on (на), дата в кавычках (M j, Y) и Edit (изменить). Давайте посмотрим, где это расположено на странице сайта:



Стрелкой я показал код, о котором написал чуть выше. Подчеркнул вывод комментария. Вы можете перевести слова и написать вместо английских те, которые я указал выше.

Чуть ниже в файле идет код вывода трекбеков, нам он неинтересен.

Еще ниже - вот такой код (есть практически в каждом файле comments):

```

<li id="comment-<?php comment_ID() ?>">
<?php comment_date('M j, Y') ?>: <?php comment_author_link() ?>
<?php if ($comment->comment_approved == '0') : ?>
<em>Your comment is awaiting moderation.</em>
<?php endif; ?>
</li>

```

Обращаю внимание на два момента:

1. Дата в американском формате - M j, Y. Выглядит как - январь 7, 2011. Замените на - jS F, Y. Станет - 7 января 2011. Если у вас стоит плагин Макса - русские даты, будет именно "января" а не "январь".

Плагин можете скачать по этой [ссылке](#). Его надо только активировать.

2. Фразу - Your comment is awaiting moderation - замените на - Ваш комментарий ожидает проверки.

Дальше идет несколько строк кода, которые вам не интересны, т.к. касаются цикла вывода комментариев, если их несколько.

```
<?php if (get_option('comment_registration') && !$user_ID) : ?>
    <p id="comments-blocked">You must be <a href="<?php echo get_option('siteurl'); ?>/wp-
login.php?redirect_to=
    <?php the_permalink(); ?>">logged in</a> to post a comment.</p>
    <?php else : ?>

    <form action="<?php echo get_option('siteurl'); ?>/wp-comments-post.php" method="post"
id="commentform">

    <h2>Post a Comment</h2>

    <?php if ($user_ID) : ?>

    <p>You are logged in as <a href="<?php echo get_option('siteurl'); ?>/wp-admin/profile.php">
    <?php echo $user_identity; ?></a>. To logout, <a href="<?php echo get_option('siteurl');
?>/wp-login.php?action=logout" title="Log out of this account">click here</a>.
    </p>

    <?php else : ?>
```

В первой части этого кода вы видите слова: You must be... потом - logged in... to post a comment. Переведите примерно так: Вам надо.... войти.... чтобы написать комментарий.

Эта часть кода работает тогда, когда вы настроите в разделе Настройки (Параметры) условия комментирования на странице Обсуждение - "Пользователи должны быть зарегистрированы и авторизованы для комментирования".

Затем в тегах H4 фраза - Post comment - переведите как "Оставить (написать) комментарий".

Фраза: You are logged in as... To logout. Переведите: Вы вошли как... Выйти.

Я думаю, вы знаете, в какой части страницы выводятся эти слова и фразы. В области окна комментариев.

Следующий ниже код касается окна комментариев:

```
<p><label for="author">Name<?php if ($req) _e(' (required)'); ?></label>
    <input type="text" name="author" id="author" value="<?php echo
    $comment_author; ?>" size="22" tabindex="1" /></p>
```

```
<p><label for="email">E-mail (will not be published)<?php if ($req) _e('
(required)'); ?></label>
    <input type="text" name="email" id="email" value="<?php echo
    $comment_author_email; ?>" tabindex="2" size="22" /></p>
```

```
<p><label for="url">Website</label>
    <input type="text" name="url" id="url" value="<?php echo
    $comment_author_url; ?>" size="22" tabindex="3" /></p>
```

Здесь вам надо перевести:

☐ Name - Имя

☐ required - обязательно

☐ E-mail (will not be published) - Email (не публикуется)

☐ required - обязательно

☐ Website - Сайт

Устали? Осталось немного.

Последний кусок кода:

```
/****** Math Comment Spam Protection Plugin *****/
if ( function_exists('math_comment_spam_protection') ) {
    $mcsp_info = math_comment_spam_protection();
    ?>
    <p><input type="text" name="mcspvalue" id="mcspvalue" value="" size="22" tabindex="4" />
    <label for="mcspvalue"><small>Spam protection: Sum of <?php echo $mcsp_info['operand1'] . ' + ' .
    $mcsp_info['operand2'] . ' ?' ?></small></label>
    <input type="hidden" name="mcspinfo" value="<?php echo $mcsp_info['result']; ?>" />
</p>
<?php } // if function_exists... ?>
<?php endif; ?>
```

```
<p><textarea name="comment" id="comment" cols="5" rows="10"
tabindex="4"></textarea></p>
```

```
<p><input name="submit" type="submit" id="submit" tabindex="5" value="Submit
Comment" />
```

```
<input type="hidden" name="comment_post_ID" value="<?php echo $id; ?>" /></p>
```

В начале вы видите сообщение - Math Comment Spam Protection Plugin. Означает, что в шаблоне используется код плагина, который выведет сообщение для защиты от спам-

ботов. Вам надо указать сумму из результата чисел. Найдите и активируйте плагин, и тогда эта часть кода будет работать.

В конце фразы - Submit Comment - переведите как - Отправить комментарий.

И в предпоследней строке фразы - Sorry, comments for this entry are closed at this time - переведите как - "К сожалению, комментарии к этой записи сейчас закрыты. Будет работать если вы в настройках записи запретили комментарии к ней.

Все. Файл comments.php я закончил. Надеюсь, кому то из вас мой рассказ окажется полезной информацией и поможет немного разобраться в коде этого файла.

## Статья двенадцатая - Как сделать новый сайдбар с поддержкой виджетов

Сегодня я продолжаю публиковать серию статей, которую посвятил изучению вместе с вами файлам стандартного шаблона Wordpress.

Эта публикация - двенадцатая по счету. Постепенно я приближаюсь к окончанию серии. Как только я ее закончу - создам на основе этой серии короткую книгу или PDF - файл, чтобы те посетители, кто придет впервые ко мне на сайт, смогли сразу скачать и прочесть, а не "тыкаться" в поисках всех страниц этой серии.

Те посетители, кто читает эту серию сейчас, или только увидели некоторые новые публикации, могут воспользоваться содержанием, которое вы видите вначале поста справа.

Сегодня мой рассказ продолжится с описания файла 404.php. откройте его в редакторе Notepad ++ (или любом другом с подсветкой синтаксиса). Он короткий и в нем сразу все видно:

```
<?php get_header(); ?>
```

```
<div id="container">
```

```
<h2>Oops: 404 page</h2><br />
```

```
The page you are looking for cannot be found. Please <b>contact</b> the author so that he can fix the problem. In the meantime you might want to browse through this blog via <b><a href="<?php bloginfo('url'); ?>">main page</a></b> to find latest posts.
```

```
<br /><br />
```

```
Sorry for the inconvenience.
```

```
</div>
```

```
<?php get_sidebar(); ?>
```

```
<?php get_footer(); ?>
```

Первая строка, и две последние - это функции вызова заголовка, сайдбара и подвала (файлов header.php, siba.php, footer.php).

Затем открывается див container, и в нем выводится короткое сообщение с заголовком. Текст звучит примерно так:

#### Упс. Страница 404

Страница, которую вы ищите, не найдена. Пожалуйста, сообщите автору об этой проблеме. Вы можете пока посмотреть главную страницу (стоит ссылка), чтобы найти последние публикации. Извините за причиненные неудобства.

Вы можете сделать перевод по-своему, если вам не нравится мой беглый.

Напомню, что страница служит для вывода именно этого сообщения тогда, когда посетитель попадает на ваш сайт по несуществующей ссылке. По-моему, я уже писал вам об этом в предыдущих статьях, сто-то припоминаю :)

Следующий файл, который я сейчас рассмотрю - function.php. В этом шаблоне нет страницы настроек, поэтому этот файл мне и вам интересен может быть только одной функцией. Многие из вас еще не знают, как выглядит функция активации динамических виджетов в сайдбаре. В этом файле одна первая, смотрите:

```
<?php
if ( function_exists('register_sidebars'))
    register_sidebars(2);
?>
```

Откройте файл lsidebar.php, вот его код:

```
<ul>
<?php if ( function_exists('dynamic_sidebar') && dynamic_sidebar(1) ) : else : ?>

    <li><h2><?php _e('Рубрики'); ?></h2>
        <ul>
            <?php wp_list_cats('sort_column=name&optioncount=1&hierarchical=1'); ?>
        </ul>
    </li>

    <?php wp_list_pages('depth=3&title_li=<h2>Страницы</h2>'); ?>

    <?php get_links_list(); ?>

<?php endif; ?>

</ul>
```

А теперь - внимательно читаем.

В файле function.php функция, которую я показал, активирует динамические виджеты в сайдбаре. Если ее удалить - вы не сможете пользоваться странице Виджеты в разделе Дизайн (Внешний вид).

В левом сайдбаре (код выше) начало кода вывода динамических виджетов - строка 2.

Конец кода вывода динамических виджетов - строка 14. Все, что лежит между строками 2 и

14 - это виджеты по умолчанию, код которых написал автор шаблона.

В шаблоне есть файл сайдбара вверху, широкого, в котором выводится приветствие. Файл под именем `sidebar_top.php`. Откроем его:

```
<h4>Обо мне</h4>
```

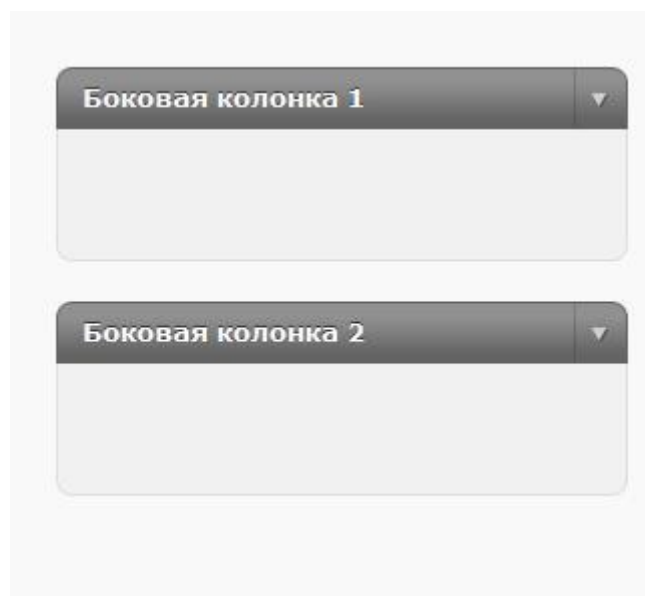
Добро пожаловать всем, кто сейчас читает этот короткий текст. Спасибо, что нашли время и посетили мой блог, в котором я пишу обо всем, но больше всего - о своей работе в качестве профессионального блоггера.<br /><br />

```
<h4>Подписаться</h4>
```

Вы можете следить за новыми публикациями в моем блоге очень просто: подпишитесь на RSS ленту и читайте одним из первых все мои новые статьи на тему блоггинга, Wordpress, заработка в Интернет и многое другое.

Видите: в этом файле никакого кода нет, и он - не динамичный, в смысле - к нему нельзя подключить динамичные виджеты. Его можно использовать только для вывода короткого приветствия и подписки.

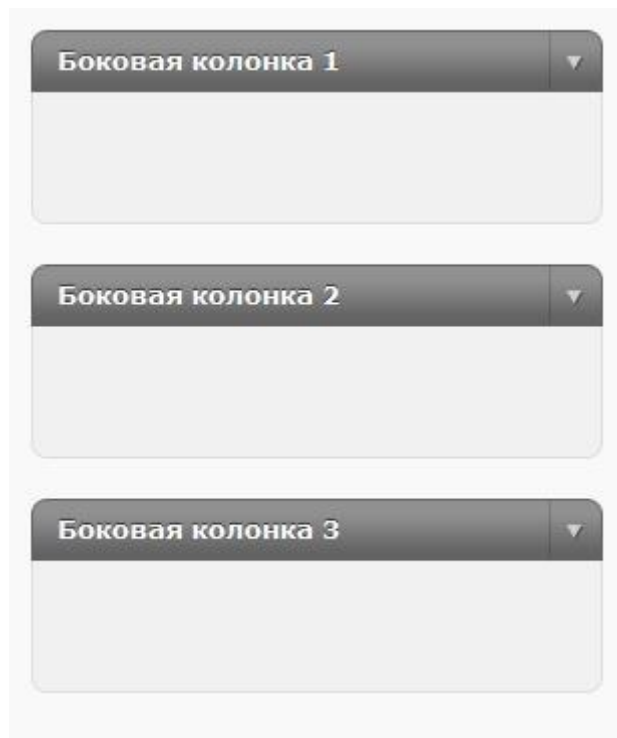
На странице Виджеты вы видите только два активных сайдбара:



В функции активации вывода динамичных виджетов (смотрите выше), которую я показал вам из файла `function.php`, есть цифра - 2. Она как раз и говорит нам о том, что в шаблоне - два сайдбара.

Давайте поэкспериментируем. Заменяем число 2 на три. Сохраним файл `function.php`. Перегрузим (F5) страницу Виджеты. Смотрим:





Видите? У нас теперь есть три сайдбара с динамичными виджетами. Но! Мы создали только панель для добавления виджетов в шаблоне. А теперь надо сделать так, чтобы в файле сайдбара тоже добавлялись динамичные виджеты. Откроем файл `sidebar_top.php` и добавим функцию подключения динамичных виджетов, сразу после нашего текста приветствия и ссылки на подписку. Получится вот так:

```
<h4>Обо мне</h4>
Добро пожаловать всем, кто сейчас читает этот короткий текст. Спасибо, что нашли время и посетили мой
блог, в котором я пишу обо всем, но больше всего - о своей работе в качестве профессионального
блоггера.<br /><br />

<h4>Подписаться</h4>
Вы можете следить за новыми публикациями в моем блоге очень просто: подпишитесь на RSS ленту и
читайте одним из первых все мои новые статьи на тему блоггинга, Wordpress, заработка в Интернет и
многое другое.
<ul>
<?php if ( function_exists('dynamic_sidebar') && dynamic_sidebar(3) ) : else : ?>

<?php endif; ?>
</ul>
```

Обратите внимание. Я вставил функцию вывода динамичных виджетов, в скобках указал число 3 - это наш новый сайдбар, он третий по счету (было два). Также добавил теги HTML для корректного отображения списка, как в других файлах сайдабаров (правом и левом). Теперь вы можете для примера вставить виджет в новый созданный сайдбар вверху страницы и посмотреть что получилось. Я для быстрого примера добавил виджет "Облака меток". Смотрим:



У нас верхний широкий сайдбар стал поддерживать вывод динамических виджетов.

Этот пример я привел только для того, чтобы вы понимали, где у вас в файле `function.php` есть функция активации динамических виджетов в сайдбаре, как она работает, какой код в сайдбаре отвечает за вывод динамических виджетов и как можно самому достаточно быстро модифицировать эти файлы.

Надеюсь, что кто-то из вас найдет для себя эту публикацию полезной.

### Статья тринадцатая - Файлы шаблона Wordpress: окончание

Сегодня речь пойдет о следующих файлах шаблона Wordpress, которые я еще не рассмотрел.

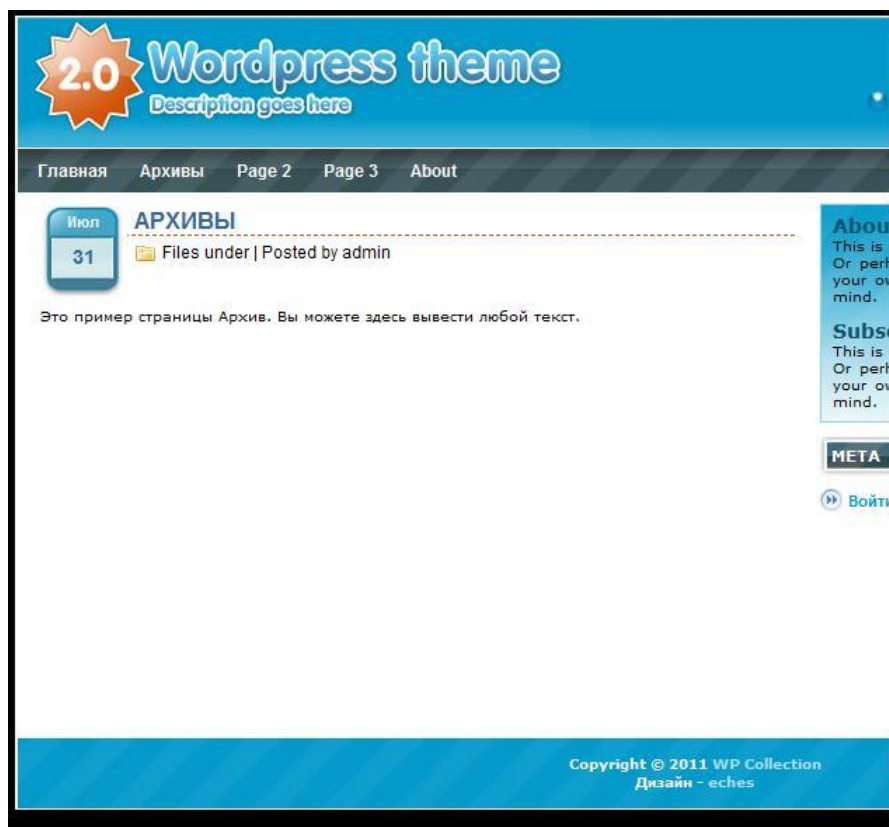
Идем по порядку. Перед нами - файл `page.php`.

Служит для вывода статичных страниц, таких как Детали, О сайте, Архив и так далее.

Код страницы похож на `index.php`, но немного отличается.

Напомню, что мы анализируем файлы шаблона o2. Скачать его можно было по ссылке в первых публикациях. Шаблон очень стандартен, и очень удобен для анализа кода.

Весь код файла лежит в контейнере - `container`. Этот див управляет всеми свойствами страницы. Выглядит она в окне браузера так:



Код файла:

```
<?php get_header(); ?>
```

```
<div id="container">
```

```
<?php if(have_posts()) : ?><?php while(have_posts()) : the_post(); ?>
```

```
<div id="post-<?php the_ID(); ?>">
```

Первая строка - подключаем файл заголовка, header.php

Третья строка - открываем контейнер дива.

Пятая и шестая - открываем цикл the Loop, который выводит нашу запись на этой странице.

```
<div><?php the_time('M') ?></div>
<div><?php the_time('jS') ?></div>
</div><!-- the_date -->
```

Здесь выводятся дата публикации, для месяца - одни свойства, для дня - другие.

```
<h2><a href="<?php the_permalink(); ?>" title="<?php the_title(); ?>"><?php the_title(); ?></a></h2>

<span><?php _e('Files under'); ?> <?php the_category(' ') ?></span> | <?php _e('Posted by'); ?> <?php
the_author(); ?>

</div><!-- post_headerr -->
</div><!-- post_header -->
```

Выводится заголовок страницы, а также мета-данные: ссылка на рубрику, в которой опубликована запись (непонятно, зачем эта функция для страницы, и ссылка на автора публикации).

Чаще всего при переводе шаблонов я такие мета-данные в файле page.php удаляю, так как не понимаю, какая рубрика? Ведь это - файл вывода страниц. И второй код - тоже часто лишний. У большинство сайтов на Wordpress автор - один, он же admin. Зачем везде повторять одно и то же?

```
<div>

<?php the_content(); ?>

</div>
```

Функция вывода текста записи.

```
<?php endwhile; ?>
<?php edit_post_link(' | Edit', ' | '); ?>

<?php else : ?>

<div>
<h2><?php _e('Not Found'); ?></h2>
</div>

<?php endif; ?>

</div>
```

Окончание цикла the Loop, ссылка на редактирование (Edit можете заменить на "Изменить"), свойство else - когда страница не найдена - посетитель увидит сообщение - Не найдено (Not Found). Здесь часто авторы шаблонов применяют инклюд и подключают файл 404.php.

И в конце файла - функции вызова сайдбара и подвала.

Следующий файл шаблона - search.php. Очень часто - почти копия файла index.php. В нашем случае файлы практически одинаковые, только в этом файле есть небольшой заголовок - Search Result, который вы можете перевести как **Результаты поиска**.

Эта страница отдается посетителю, который что-то решил найти в вашем блоге и воспользовался окном поиска на сайте.

Если ключевое слово нашло совпадения - посетитель увидит список записей. Если нет - увидит пустую страницу с сообщением - Ничего не найдено.

Сравните файлы index.php и search.php. Практически одинаковые. Поэтому я не буду повторяться.

Следующий файл - searchform.php. Это - именно окно (форма) поиска на сайте. Код файла:

```
<form id="search" method="get"
action="php bloginfo('home'); ?" >
<input type="text" value="искать" onfocus="if (this.value == 'искать') {this.value = '');"
onblur="if (this.value == '') {this.value = 'искать'}" size="18" maxlength="50" name="s" id="s" />
</form>
```

На странице сайта вы его видите справа вверху.

Объяснять здесь особенно нечего.

Последний файл шаблона 02, который остался, это single.php. Он отвечает за вывод отдельной записи на отдельной странице.

В нашем случае - очень похож на файл page.php, поэтому я не буду повторяться. Посмотри и сравните.

Бывают темы, когда код этого файла - еще меньше. Бывает - очень много функций. Но в основном, в стандартных и не сложных шаблонах, он именно такой, что-то среднее между index.php и page.php.

Вот и добрался я до окончания этой серии. Получилось 13 публикаций, какое-то магическое число. Наверное, к удаче.

Пожелаю ее всем, кто вместе со мной добрался до этих слов.

Если вам понравилась эта серия, - напишите в комментариях. Если не понравилась - тоже пишите, опубликую все, кроме мата :)