Каждая программа имеет свою пред-историю. Это то, что побудило написать ее.

Я знаком с несколькими CMS. Среди них Joomla, HostCMS и некоторые другие. Возникла необходимость познакомиться с CMS WordPress. Нет такого программиста, который бы не захотел заглянуть в исходный код каждой новой CMS. Например исходный код Joomla состоит примерно из 3000 файлов! Таков ее вес. CMS WordPress намного, намного легче. В этом ее несомненное достоинство перед Joomla.

Посмотрим например файлы шаблона WordPress. Для примера те файлы, которые находятся в каталоге:

wordpresswp-contentthemestwentythirteen>

Это так называемые темы. Это тема twentythirteen, которая включена в дистрибутив по умолчанию. Что мы там видим? В корне каталога мы видим порядка 30 php-файлов, которые расписывают тему. Имена этих файлов в разных темах в большинстве случаев совпадают. Раскрывая эти файлы по F4 или F3 мы видим "слепой" php-код с фрагментами html-кода. Неудобно.

Хорошо бы если бы код этих php-файлов был расцвечен, например html-код синим, комментарии - зеленым, отдельные php-команды другими цветами. Насколько легче рассматривать php расцвеченный код! Что касается меня, то я почти никогда не изучаю код в его слепом виде. Конечно, Notepad++ сделает все как надо, но не всегда хочется рассматривать файлы в Notepad.

Кроме этого гораздо удобнее было бы если бы каждый php-файл шаблона находился на отдельной странице, между страницами была бы навигация, отдельные страницы были бы собраны в главу, а у главы было бы оглавление страниц. Из глав можно было бы создать книгу. Я всегда, когда пишу программу или изучаю программу все свои действия, код функций, тесты, многочисленные варианты и тд. пишу в такую отдельную книгу. Естественно с рацвеченным кодом. Новая программа - новая книга. Такая тактика себя полностью оправдывает и окупается сторицей. Иначе можно безнадежно утонуть в ворохе файлов, функций, вариантов, миллионов строк кода. Через три месяца можешь не узнать собственный код, думая, а действительно ли это я написал?

Ну что-ж, было бы желание. PHP-язык имеет все возможности, чтобы претворить эти желания в жизнь. Тогда, нажав одну кнопку один раз мы будем иметь и книгу, и главы, и страницы с расцвеченным кодом всех php-файлов шаблона. Программа все это сделает за нас!

А начнем мы с расцветки кода:

```
<div class="entry-attachment">
        <div class="attachment">
                <?php twentythirteen_the_attached_image(); ?>

                <?php if ( has_excerpt() ) : ?>
                <div class="entry-caption">
                        <?php the_excerpt(); ?>
                </div>
                <?php endif; ?>
        </div><!-- .attachment -->
</div><!-- .entry-attachment -->

<?php if ( ! empty( $post->post_content ) ) : ?>
<div class="entry-description">
        <?php the_content(); ?>
        <?php wp_link_pages( array( 'before' '<div class="page-links">'
</div><!-- .entry-description -->
<?php endif; ?>
```

В этом нам поможет класс HtmlParser.

Вообще в программировании уже все задачи давно кем-нибудь решены. Как в поговорке: "О любви не говори. О ней все сказано.". Значит ли это что больше не надо писать новых программ? А вдруг они будут хуже тех, которые уже есть?

Класс HtmlParser написал человек по имени Jose Solorzano в 2003 году. Файл с этим классом был "пойман" мной в интернете. Посмотрел. А что? Вполне подходящий класс, чтобы парсить не только html, но и php-код. Надо его только чуть-чуть переделать под наши нужды. О, это мое любимое занятие, чуть-чуть переделывать чей-то код! Это как пить чай с малиновым вареньем.

Суть класса HtmlParser в том, чтобы разбить парсимый код на лексемы нескольких типов, к примеру: тип Комментарий, тип HTML-тег, тип PHP-тег, тип Текст. Каждую из этих лексем сохранить в структуре состоящей из трех полей - Тип, Имя, Значение. Сама лексема будет сохранена в поле Значение, а ее тип в поле Тип. Имя можно дать в соответствии с ее типом. Все эти структуры можно сохранять в простой массив, по ходу парсинга. Таким образом в массиве структур будет сохранен весь текст парсимого файла.

Далее просто выводя из массива в новый файл данные структура за структурой будем окрашивать значение каждой структуры (а это фрагмент текста) в свой цвет в зависимости от типа структуры.

Код класса при внимательном рассмотрении становится совершенно понятным. Сначала парсимый файл считывается в строку при помощи вызванной в файле index.php функции HtmlParser_ForFile(), в которой считанная строка передается конструктору класса HtmlParser() при создании объекта класса. Конструктор инициирует все необходимые переменные класса, записывает в переменную iHtmlText переданную в параметре строку, устанавливает указатель iHtmlTextIndex в нулевую позицию строки. Далее, в файле index.php, вызывается функция structOfTag(), которая в цикле вызывает метод parse() класса до тех пор, пока не будет найден конец строки.

Несколько слов о файле index.php. Этот файл и создает html-книгу и ее страницы, и записывает в эти страницы расцвеченный php-код.

Примечание: Парсимые файлы помещаются в директорию from2. Результат - книга в html-формате находится в директории booktest.

Кстати, ни в коем случае не стремился написать программу прекрасную во всех

отношениях. Довольно того, что программа выполняет поставленную перед ней задачу, и при этом избавляет программиста от рутинного труда, дарит драгоценное время! Много не тестировал ее. Проверил только на шаблонах включенных в дистрибутив WordPress. Ошибок не наблюдал.

Скачать файлы с исходным кодом можно здесь: WordPress и HtmlParser

index.php

{codecitation style="brush: php;" class="collapse:true;" }