Цель домашнего задания

* Отработать создание и подключение функций на практике.
* Выделить отдельный шаблон сайта.
* Создать новые страницы и реализовать навигацию по ним на сайте.

Что входит в задание

Продолжаем доделывать и развивать сайт, сделанный вами в предыдущем домашнем задании. Необходимо реализовать новые страницы на сайте и навигацию по ним, применив функции, созданные ранее. Домашнее задание состоит из трёх последовательно выполняемых задач.

1. Создание шаблона и страниц сайта.
2. Вывод меню.
3. Вывод заголовка страницы

Задание 1. Создание шаблона и страниц сайта

Что нужно сделать

Сначала необходимо создать новые страницы на сайте.

**1. Шаблон сайта**

Разделите вёрстку вашего сайта на хедер и футер. Файлы назовите соответственно (**header.php** и **footer.php**) и расположите их в директории templates в корне вашего сайта. Эти файлы должны подключаться из любой публичной страницы на вашем сайте.

**2. Ядро проекта и вспомогательные функции**

Создайте директорию src в корне сайта. Создайте в ней файл **core.php** — этот файл будет отвечать за подключение всех системных файлов проекта. Подключите файл **core.php** в самом начале файле **header.php**: таким образом весь исходный код будет доступен на всех страницах вашего приложения, где будет подключен header.

Скопируйте в директорию src файлы с объявлениями функций **cutString** и a**rraySort**, которые вы создали в предыдущем задании. Подключите эти функции в файле **core.php**.

**3. Страницы сайта**

Создайте новые страницы. Они должны располагаться в отдельном разделе route. Для этого создайте новый раздел в корне сайта — **/route/**.

Внутри него заведите не менее пяти подразделов с произвольными названиями, в каждом из них создайте индексные страницы, не забыв подключить **header** и **footer**.

Путь к разделам указывайте относительно раздела route (все URL должны начинаться с **/route/#ваш\_раздел#/**, например **/route/about/** или **/route/kak\_kupit\_kursy/**).

    4. Навигационное меню

Создайте файл **main\_menu.php**, в котором создайте массив с описанием пунктов меню, где каждый пункт — это ассоциативный массив следующего вида:

[  
   'title' => '...', // Название пункта меню  
   'path' => '...', // Ссылка на страницу, куда ведет этот пункт меню  
   'sort' => 0,  // Индекс сортировки  
]

Например:

[  
   'title' => 'Главная',  
   'path' => '/',  
   'sort' => 10,  
],  
[  
   'title' => 'О нас',  
   'path' => '/route/about/',  
   'sort' => 2,  
],

Заполните этот массив. В нём обязательно должна быть ссылка на главную страницу и ссылки на все разделы, которые вы создали внутри раздела **route**. Подключите файл в ядре вашего сайта.

Советы и рекомендации

* Каждая страница на сайте — это подключение хедера, затем тело страницы и в конце подключение футера. Именно так выглядит шаблон проектирования Page Controller. Каждая страница — это отдельная точка входа в приложение.
* Контент каждой страницы должен быть уникальным. Если она содержит повторяющиеся части кода, значит, эти части должны быть вынесены в хедер и футер.
* Раздел **route** в нашем задании — это лишь пример организации структуры ссылок на сайте и создан для того, чтобы потом решить дополнительную задачу в этом задании. Поэтому не воспринимайте его как цельный подход к организации страниц на сайте.

Что оценивается

* В каждом созданном разделе должен быть создан индексный файл.
* Общие части всех страниц должны быть вынесены в header и footer.
* В каждом индексном файле должны подключаться эти header и footer, а между ними должен остаться только уникальный для этой страницы контент.
* В файле header должно подключаться ядро проекта.
* Подключение всех функций должно быть осуществлено в ядре проекта.
* Файл с меню **main\_menu.php** должен содержать не менее пяти пунктов меню.

Задача 2. Вывод меню

Что нужно сделать

Для вывода меню реализуйте функцию function showMenu(...), которая будет, согласно вёрстке, выводить меню разделов в двух местах — шапке и футере. Каждый пункт меню должен быть ссылкой. Функцию разместите в разделе /src/. Эта функция не будет «чистой функцией».

В шапке пункты должны быть отсортированы в порядке возрастания параметра 'sort'. В футере пункты меню должны быть отсортированы в обратном алфавитном порядке (параметр 'title'). Для сортировки используйте функцию arraySort, которую вы написали в прошлом задании.

При этом у каждого места вывода в шаблоне будет своё дополнительное визуальное оформление:

* в шапке все пункты должны идти горизонтально и размер текста (font-size) должен равняться 16px;
* в подвале все пункты должны располагаться вертикально и размер текста (font-size) должен равняться 12px;
* если мы находимся в разделе, то необходимо выделять соответствующие пункты меню стилем text-decoration:underline;

Также должно выполняться следующее требование: если название пункта меню более 15 символов, то при выводе меню необходимо обрезать такое название до 12 символов и в конце установить троеточие (например: «Название раз...»), используйте вспомогательную функцию cutString, которую вы написали в прошлом задании.

Советы и рекомендации

* Подумайте и определите, какие параметры должны быть у этой функции: судя по условию, функция должна принимать массив меню, также направление сортировки и какой-то параметр, позволяющий менять стили.
* Одна функция подойдёт для вывода меню и вверху и внизу.
* Шаблон меню состоит и из тега ul, и из тегов li внутри него. Попытка создать шаблон без обрамляющего ul — это семантическая ошибка. Подключив такой шаблон вне тега ul, вы получите ошибку разметки.
* Не пытайтесь вшить ваши частные случаи (что у вас есть верхнее и нижнее меню) в функцию. Старайтесь мыслить шире. Этой функцией можно вынести сколько угодно видов меню, ведь внешний вид определяется css-классами и css-файлами.
* Разрывать теги внутри тела функции категорически запрещено. И как вы помните, выводить теги оператором echo или складывать их в переменную тоже недопустимо. Создайте отдельный файл — шаблон меню — и подключите его внутри вашей функции.
* Вам будет казаться, что подключать используемые функции только непосредственно перед их использованием с помощью require\_once — это хорошо с точки зрения экономии ресурсов. Возможно, но с точки зрения организации кода — это клубок с нитками. Все файлы с функциями (ядро вашего приложения) подключайте в шапке сайта. А если вы хотите задуматься об оптимальности, то нужно использовать механизмы автоматических подключений файлов при необходимости. Это будет рассмотрено на примере автозагрузки классов в ООП в следующем курсе.

Что оценивается

* Меню как в шапке, так и в футере должно выводиться функцией showMenu.
* В коде функций не должны быть зашиты два варианта отображения меню для хедера и футера. Функция должна быть универсальной.

Задача 3. Вывод заголовка страницы

Что нужно сделать

На каждой странице внутри раздела route нужно выводить корректный заголовок страницы, используя тег <h1>. При этом этот заголовок должен браться из названия текущего активного пункта меню из массива с меню в файле **main\_menu.php**.

Советы и рекомендации

* Создайте вспомогательную функцию, которая это будет делать.
* Не пробуйте передавать страницу GET-параметрами — это плохая мысль.
* Не определяйте текущий пункт меню по исполняемому файлу. Исполняемый файл не имеет значения и может быть другим, а ключевой параметр — это именно url.
* Для работы с url есть вспомогательная функция **parse\_url**.
* В переменной **$\_SERVER** есть подходящие данные.

Что оценивается

* Заголовок каждой страницы берётся из меню у соответствующего пункта меню.
* Для поиска подходящего пункта меню используется переменная $\_SERVER, и сравнение осуществляется по url, а не по $\_GET-параметрам или имени исполняемого файла.