УРАЛЬСКИЙ ГОСУДАРСТВЕННЫЙ ЛЕСОТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ
УГЛТУ

Е.В. Побединский
В.В. Побединский

ПРОЕКТИРОВАНИЕ ВЕБ-САЙТОВ С ИСПОЛЬЗОВАНИЕМ ТЕХНОЛОГИЙ PHP, HTML, CSS И WORDPRESS

МИНОБРНАУКИ РОССИИ

ФГБОУ ВО «Уральский государственный
лесотехнический университет»

Е.В. Побединский
В.В. Побединский

ПРОЕКТИРОВАНИЕ ВЕБ-САЙТОВ С ИСПОЛЬЗОВАНИЕМ ТЕХНОЛОГИЙ PHP, HTML, CSS И WORDPRESS

Учебное пособие

Екатеринбург
2018

УДК 004.4'24
ББК 32.973.26-018.2
П18

Рецензенты:
кафедра информатики и прикладной математики ФГБОУ ВО «Братский государственный университет;
доцент кафедры математики и информатики ФГБОУ ВО «Государственный аграрный университет Северного Зауралья», канд. пед. наук Л.И. Якобюк

Побединский, Е.В.
П18 Проектирование веб-сайтов с использованием технологий PHP, HTML, CSS и WordPress: учеб. пособие / Е.В. Побединский, В.В. Побединский. – Екатеринбург: Урал. гос. лесотехн. ун-т, 2018. – 115 с.

ISBN 978-5-94984-651-3

Излагаются методы и приемы проектирования веб-сайтов. Основное внимание уделено проектированию адаптивных элементов сайта. Описано содержание исходного кода элементов. Предназначено для обучающихся по направлениям 23.05.01 «Наземные транспортно-технологические средства», 23.03.03 «Эксплуатация транспортно-технологических машин и комплексов», профиля 23.04.03 «Сервис транспортных и транспортно-технологических машин автодорожно-строительного и лесного комплексов» по дисциплине «Современные информационные технологии в техническом сервисе».

Печатается по решению редакционно-издательского совета Уральского государственного лесотехнического университета.

УДК 004.4'24
ББК 32.973.26-018.2

ISBN 978-5-94984-651-3

ОГЛАВЛЕНИЕ

ВВЕДЕНИЕ

Еще недавно, после появления Интернета, в различных сферах предпринимательской деятельности, производстве, науке можно было обойтись без собственного сайта. Но на сегодня необходимость сайта не вызывает сомнений, потому что для любого участника рынка, общественных, государственных структур иметь свой сайт стало необходимостью. Основным фактором, который привел к такому положению дел, является, в первую очередь, развитие веб-технологий. Интернет распространился повсеместно, стал доступным по цене и с использованием современных компьютеров появился практически в каждом доме. Для большей части населения Интернет стал основным источником информации и постепенно вытесняет печатные СМИ, телевидение. Понимание необходимости обладания своим веб-ресурсом, переносить часть своего «оффлайнового» бизнеса в Интернет пришло повсеместно. В короткий срок количество разного рода коммерческих, частных, государственных сайтов в сети стремительно выросло и процесс переноса off-line бизнеса в Интернет продолжается с нарастающей динамикой.

Для создания сайтов существующие технологии с использованием языков программирования PHP, HTML и CSS являются достаточно сложными и применяются в практике профессиональных веб-студий или квалифицированными специалистами-разработчиками сайтов. Но, несмотря на сложность, процесс изучения, начинаемый именно с этих методов, представляется наиболее эффективным, так как обучающиеся получают фундаментальные знания языков программирования, выполнения программного кода, настройки элементов сайта и вообще понимание работы технологии.

С другой стороны, чтобы не перегружать специальной информацией обучающихся специальностей автодорожно-строительного и лесного комплекса, целесообразнее было предложить к рассмотрению более простой вариант, т.е. на примере какого либо одного сайта, выполненного такими же средствами. В качестве примера использован реально действующий сайт строительной компании «ООО “Флатирон”» (www.flatiron.ru). При этом методика процесса разработки предусматривает выполнение действий строго по пунктам, в которых поясняются только их последовательность и краткое назначение.

Такая технология была широко распространена и продолжает применяться, но ситуация изменилась с появлением систем

CMS (Content Management System – система управления контентом), с помощью которых любой пользователь может создавать полноценные сайты, по функциональности не уступающие разработанным профессионалами. Такую задачу в настоящем пособии предлагается выполнить в рамках конкретной CMS WordPress. Помимо освоения работы в WordPress решается весь основной спектр вопросов, связанный с разработкой и информационной поддержкой веб-сайта.

Предназначено для магистров и обучающихся по направлениям 23.05.01 «Наземные транспортно-технологические средства», 23.03.03 «Эксплуатация транспортно- технологических машин и комплексов», профиля 23.04.03 «Сервис транспортных и транспортно-технологических машин автодорожно-строительного и лесного комплексов», по дисциплине «Современные информационные технологии в техническом сервисе».

Глава 1

ВЕБ-ТЕХНОЛОГИИ. ОБЩИЕ ПОНЯТИЯ. ОПРЕДЕЛЕНИЯ. РАЗВИТИЕ

1.1. Назначение и основные функции сайтов

Прежде чем перейти к изучению функций сайта, рассмотрим понятие сайта в целом, назначение сайта и его возможности.

В целом, сайт представляет собой множество страниц, связанных между собой общей тематической направленностью, единым оформлением (дизайном) и системой навигации (ссылками). Главной задачей профессионально построенного сайта коммерческой компании является превращение посетителя, зашедшего на сайт, в потенциального клиента.

В первую очередь сайт представляет подробную информацию о владельце, например, предприятии или фирме, предоставляемых услугах, условиях заказов, контактах и реквизитах. Такая информация остается доступной круглый год днем и ночью. Любую информацию можно легко публиковать, размещать, изменять без разработки новых макетов, как в традиционной рекламе, а ссылки на сайт можно указывать в любых рекламных материалах, в визитках, в различных справочниках, каталогах и поисковых системах. Таким образом, веб-сайт становится инструментом для привлечения потенциальных клиентов и информационного обеспечения существующих.

Для территориально разобщенной фирмы все взаимодействие между сотрудниками и внутренний документооборот возможен только с помощью корпоративного сайта, на котором под соответствующей защитой будет находиться нужная информация. В процессе работы фирмы сайт может помогать сотрудникам и руководителям в качестве удобного и многофункционального бизнес - инструмента. При размещении на сайте документов, заданий, отчетов и другой информации, руководители компании получают возможность доступа к необходимой информации из любого географически удаленного места,

где есть подключение в Интернет. Также сотрудники, находящиеся в территориально рассредоточенных офисах, получают доступ к общей служебной информации. С помощь сайта появляется возможность разгрузить работников фирмы, а часть работы с клиентами (часто задаваемые вопросы, предоставление информации на прайс-листах, сертификаты производителей, тематические статьи и др.) организовать, разместив их на сайте. Хорошо спроектированный сайт является, в определенной степени, активным инструментарием, который может привлекать новых клиентов. Что касается процессов работы с информацией в сети Интернет, то она отличается скоростью приема-передачи, возможностью использования больших объемов данных, разнообразием и удобством использования.

Таким образом, в числе наиболее значимых функций сайта можно назвать следующие:

- является круглосуточным виртуальным офисом;
- является визитной карточкой предприятия, предоставляющей основную информацию, контактные данные, размещение, основные товары и услуги;
- обеспечивает конкурентное преимущество для любой фирмы;
- можно рассматривать как многоцелевой инструмент бизнеса;
- по совокупности своих функциональных свойств является самым уникальным средством массовой информации;
- является самым недорогим видом рекламы в сравнении с ее традиционными видами;
- является лицом любого современного предприятия и выполняет представительскую (имиджевую) функцию.

С технической точки зрения сайт должен быть информативным, современным, хорошо оформленным визуально, обладать необходимой функциональностью, быть защищенным от несанкционированного доступа, быть оптимальным для поисковых систем, иметь рациональную структуру.

1.2. Принятые в работе сокращения, обозначения, термины и определения к ним

Обзор технической литературы показывает, что за последние годы в области информационных технологий обозначилась проблема терминологии, которая имеет большое значение. Видимо, проблема гораздо шире и связана с развитием в ходе прогресса любой отрасли

знаний. Нужно отметить, вообще о роли терминов в науке говорит известный исторический факт, что представление об инерции было известно задолго до Галилея. Но только после введения Галилеем термина «инерция», точно определилось понятие инерции, и оно вошло в научный оборот.

Во многих случаях переход от предположений к точному знанию происходит после введения соответствующего термина. Поэтому все ученые в истории уделяли особое внимание развитию научной терминологии. «…Фиксируя время рождения того или иного термина, можно судить о развитии науки, ее проблематике, объектах исследования, о появлении новых научных направлений…» [1].

Нечто подобное сегодня наблюдается в любой отрасли. Поучительным становится опыт в тех случаях, когда вопросы научной терминологии достаточно проработаны. В этой связи в настоящем пособии разработан и рекомендован для использования терминологический аппарат, включающий следующие термины с соответствующими определениями.

Адаптивный дизайн - способ оформления веб-страниц, обеспечивающий корректное отображение веб-сайта под любые заданные размеры экрана.

Админ-панель - интерфейс управления сайтом.

Браузер - специальная клиентская программа, предназначенная для просмотра содержимого веб-узлов и отображения документов HTML. В браузеры встроен транслятор языка разметки гипертекста, компилирующий HTML-код в процессе открытия веб-страницы.

Веб-сервер - 1) сервер, принимающий HTTP-запросы от клиентов, обычно веб-браузеров, и выдающий им HTTP-ответы, как правило, вместе с HTML-страницей, изображением, файлом, медиапотоком или другими данными [2] в компьютерных сетях специализированное устройство, узел, выполняющий определенные функции по запросам других узлов и компьютерных устройств [3]; 3) комбинация аппаратных и программных средств, обеспечивающих обслуживание и хранение информации компьютеров, объединенных в сеть [3].

Единый указатель ресурса (англ. Uniform Resource Locator, URL) - ссылка на веб-ресурс.

Идентификатор - уникальное имя элемента, которое можно использовать не более одного раза на странице.

Индексирование в поисковых системах - процесс добавления сведений (о сайте) роботом поисковой машины в базу данных, впоследствии использующуюся для (полнотекстового) поиска информации на проиндексированных сайтах [2].

Интернет-провайдер (англ. internet service provider — поставщик Интернет-услуг) - организация, предоставляющая услуги доступа к сети Интернет и иные связанные с Интернетом услуги [2].

Каноническая ссылка - адрес, индексируемый при наличии страниц-дублей.

Класс (в CSS) – вид селектора, который может быть использован неограниченное количество раз на одной странице и в свойствах одного элемента.

Контейнер (в HTML) - дескрипторная пара, которая содержит HTML-элементы.

Контент – содержимое, информационное наполнение сайта [2].

Локальный сервер - сервер, имитирующий работу реального сервера хост-провайдера. Функционал состоит из аналогичных компонентов, например: веб-сервер Apache с поддержкой SSL, СУБД MySQL, утилита phpMyAdmin, PHP, Perl, FTP-сервер FileZilla, POP3/SMTP сервер.

Логическая структура сайта - набор тематических рубрик с распределенными по разделам документами и спроектированными гиперсвязями между всеми страницами ресурса.

Медиа-запрос (англ. media query) - метод создания набора правил, применяемых в зависимости от заданных параметров. Широко используется при разработке адаптивного дизайна.

Медиафайл – файл, содержащий аудио-, видео- и графическую информацию.

Метаданные – информация об HTML-документе, не отображаемая видимым образом на странице, но считываемая браузером и поисковыми роботами и впоследствии используемая для отображения страницы соответственно указанной информации.

Панель навигации - область веб-страницы, на которой в некотором упорядоченном виде расположены ссылки на разделы и (или) страницы сайта, и функция которой — предоставить пользователю удобное средство для перемещения по веб-сайту [4].

Плагин - (англ. plug-in, от plug in — «подключать») программная надстройка, расширяющая возможности программы.

Поисковая оптимизация (англ. search engine optimization) – комплекс мер по внутренней и внешней оптимизации для поднятия позиций сайта в результатах выдачи поисковых систем по определенным запросам пользователей с целью увеличения трафика (для инфоресурсов) и потенциальных клиентов (для коммерческих ресурсов) и последующей монетизации этого трафика [2].

Поисковая машина (searching engine) – веб-сервер, проводящий индексацию веб-страниц на доступных серверах (например, Yandex) [5].

Поисковый робот - программа, являющаяся составной частью поисковой системы и предназначенная для сканирования страниц Интернета с целью занесения информации о них в базу данных поисковой системы [6].

Проектирование веб-сайтов - процесс определения архитектуры, компонентов, интерфейсов и других характеристик системы или её части [8].

Псевдокласс (в CSS) – вид селектора, предназначенный для применения стилей к элементам в зависимости от их состояния.

Псевдоэлемент (в HTML) – тип элемента, позволяющий добавить контент на страницу с помощью CSS правила. Добавленный контент не отображается в структуре HTML-кода.

Сайт - упорядоченный набор взаимосвязанных HTML-страниц в Интернете [3, 8].

Селектор – формальное описание того элемента или группы элементов, к которым применяется указанное правило стиля [9].

Система управления содержимым (англ. Content management system, CMS) – информационная система или компьютерная программа, используемая для обеспечения и организации совместного процесса создания, редактирования и управления контентом веб-сайта [2].

Тег – элемент языка разметки гипертекста [2]. Отображение текста на странице зависит от свойств тега, внутри которого содержится текст.

Файл стилей – файл, содержащий в себе базу CSS правил.

Физическая структура сайта – схема расположения физических файлов в директории сайта.

Хостинг (англ. hosting) – услуга по предоставлению ресурсов для размещения информации на сервере, постоянно находящемся в веб-сети (обычно Интернет) [2].

Apache HTTP Server – кроссплатформенный веб-сервер.

CSS (англ. Cascading Style Sheets − каскадные таблицы стилей) − формальный язык описания внешнего вида документа, написанного с использованием языка разметки [2].

Footer – нижний колонтитул, блок в нижней части страницы веб-сайта, куда выносят, как правило, служебную или дополнительную информацию.

FTP (англ. File Transfer Protocol - протокол передачи файлов) стандартный протокол, предназначенный для передачи файлов по TCP-сетям. Часто используется для загрузки сетевых страниц и других документов с частного устройства разработки на открытые серверы хостинга [2].

FileZilla - свободный многоязычный FTP-клиент с открытым исходным кодом [2].

Header – верхний колонтитул, блок в верхней части страницы веб-сайта.

HTML (англ. Hyper Text Markup Language — язык гипертекстовой разметки) – стандартный язык разметки документов во всемирной паутине [2].

JavaScript (JS) – прототипно-ориентированный сценарный язык программирования. Является реализацией языка ECMAScript (стандарт ECMA-262) [2].

JQuery – библиотека JavaScript, фокусирующаяся на взаимодействии JavaScript и HTML [2].

MySQL – свободная реляционная система управления базами данных [2].

Perl (англ. Practical Extraction and Report Language – практический язык для извлечения данных и составления отчётов) высокоуровневый интерпретируемый динамический язык программирования общего назначения [2].

PHP (англ. Hypertext Preprocessor (PHP) – препроцессор гипертекста; первоначально Personal Home Page Tools — инструменты для создания персональных веб-страниц) скриптовый язык общего назначения, применяемый для разработки веб-приложений [2].

PhpMyAdmin - веб-приложение с открытым кодом, написанное на языке PHP и представляющее собой веб-интерфейс для администрирования СУБД MySQL [2].

SVG (англ. **S**calable **V**ector **G**raphics — масштабируемая векторная графика) – язык разметки масштабируемой векторной графики, предназначенный для описания двумерной векторной и смешанной векторно/растровой графики в формате XML [2].

Usability – степень оценки эффективности, трудоемкости и удовлетворенности, с которыми продукт может быть использован определенными пользователями при определенном контексте использования для достижения определенных целей/мотивов [9].

XML (англ. e**X**tensible **M**arkup **L**anguage − расширяемый язык разметки) – спецификация XML описывает XML-документы и частично

описывает поведение XML-процессоров (программ, читающих XML-документы и обеспечивающих доступ к их содержимому) [2].

XAMPP (**X** — все операционные системы, **A**pache, **M**ySQL, **P**HP, **P**erl) - программное обеспечение веб-сервера.

Учитывая насыщенность работы различной информацией, для удобства восприятия в тексте приняты следующие обозначения.

Жирным шрифтом – обозначены директории, каталоги и названия файлов.

*Курсивом шрифтом Times New Roman обозначен формат расширения файлов, например, *.sql.*

В английские двойные кавычки “ ” – заключаются названия кнопок или пункты меню визуальных форм интерфейса.

Во французские кавычки елочкой « » - заключаются сообщения при работе программ.

`Шрифтом Consolas красного цвета`, на сером фоне, например, `style` - обозначены атрибуты свойств или классы.

`Синим шрифтом Consolas, на сером фоне` в угловых скобках, например, `<ul>` обозначены теги кода, переменные или элементы, например, `<body>`.

`Шрифтом Consolas черного цвета на сером фоне`, например, `RewriteEngine on` написаны операторы кода PHP.

`Шрифтом Consolas темно-фиолетового цвета на сером фоне`, например, `image-caption` указываются значения параметров или свойства элементов.

`Надписи шрифтом Consolas зеленого цвета на сером фоне, например`, `/** Блок footer **/` обозначена служебная информация или комментарии в коде.

Шрифтом Times New Roman стандартного синего цвета, например, http://localhost/flatiron обозначены ссылки на ресурсы в Интернет.

1.3. Общая схема разработки сайта

Сложность и трудоемкость процесса разработки сайта зависит от его назначения, цели, задач, функциональных возможностей и других характеристик. В самом общем виде алгоритм разработки сайта имеет вид, как показано на рис. 1.

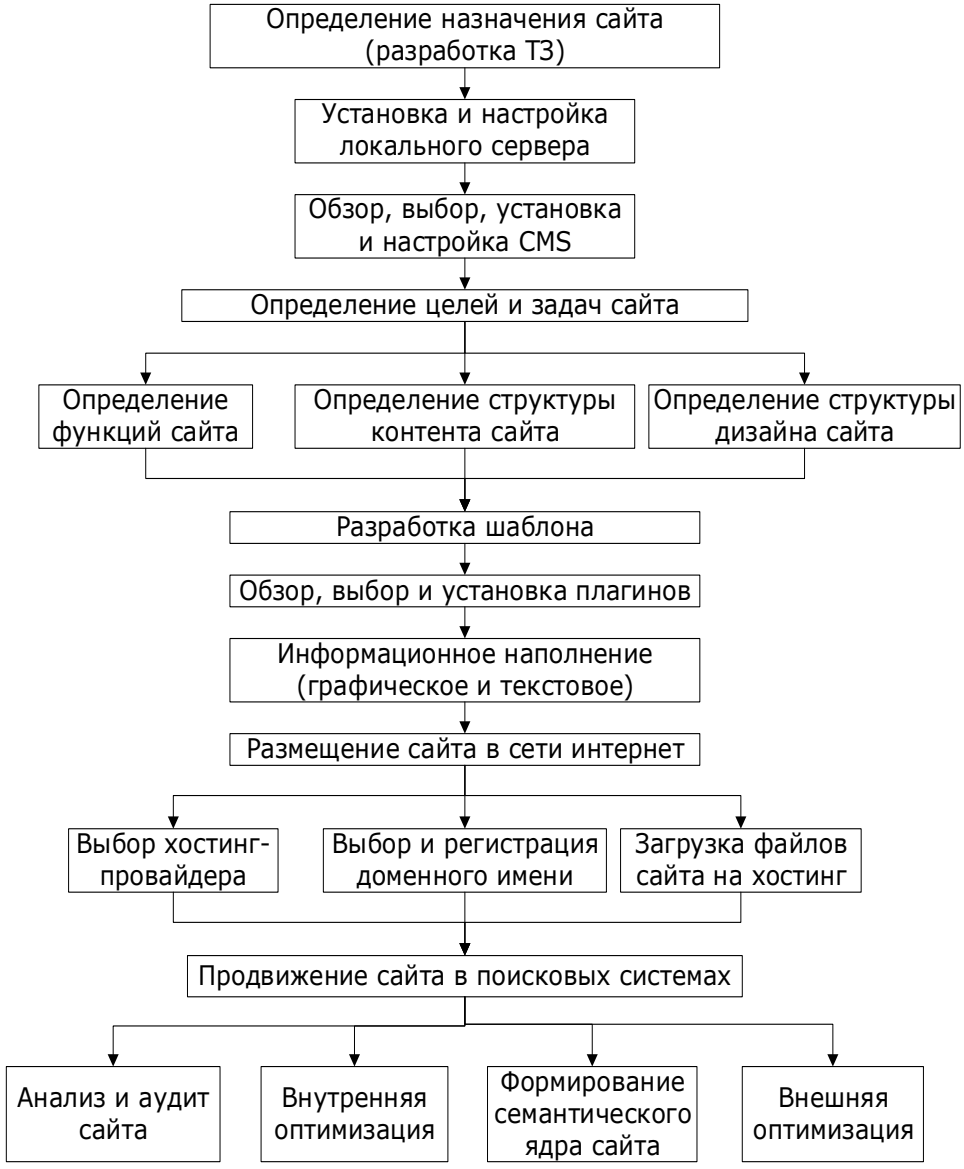

Рис. 1. Алгоритм разработки сайта

1.4. Средства для разработки сайтов

В настоящем пособии в той или иной мере используются практически все основные программные средства разработки сайтов - язык гипертекстовой разметки HTML, скриптовый язык программирования PHP, формальный язык CSS и система WordPress. Используются они только в самом необходимом объеме, тем не менее, для общего представления рассмотрим их подробнее.

1.4.1. Введение в HTML

Язык гипертекстовой разметки (HyperText Markup Language — HTML) предназначен для описания семантики и структуры содержимого веб-страницы и является одним из основных языков в веб-разработке, без знания которого невозможно создать веб-сайт. При верстке веб-страницы текст размечают тегами, поэтому HTML-элементом называется текст, заключенный между открывающим и закрывающим тегами. Теги делятся на открывающие и закрывающие.

Теги дают тексту дополнительное свойство и позволяют браузерам правильно отображать структуру страницы. К примеру, с помощью тегов можно определить, что текст «Введение в HTML» является заголовком первого уровня:

```
<h1>Введение в HTML</h1>
```

Отличие закрывающего тега от открывающего заключается в наличии символа «/», который несет информацию для браузера об окончании части структурированного кода, открытого тегом `<h1>`.

Существуют также элементы без содержимого и закрывающего тега, например `<br>`, который добавляет разрыв строки, или `<img>`, посредством которого на страницу добавляется изображение.

Элементы подразделяются на два типа: блочные и строчные. Блочные элементы отображаются в виде отдельного блока, перед и после которого добавляется разрыв строки. Синтаксис конструкции, которая отображает отдельный блок, выглядит следующим образом:

```
<blockquote>Информация в отдельном блоке</blockquote>
```

Строчные элементы могут быть расположены как отдельно, так и внутри абзаца информации. Добавление слова (цитаты) внутрь информации может быть выполнено, например, следующей записью:

```
<p>Информация, внутрь которой добавляется<q>слово
(цитата)</q>.</p>
```

Тегу могут быть заданы атрибуты, при помощи которых указывается дополнительная информация о содержимом HTML-элемента. Рассмотрим пример использования атрибута в HTML-теге для задания гиперссылки:

```
    <a href="http://example.org/">
```

Тег `<a>` определяет, что данный элемент является гиперссылкой, а атрибут `href` указывает браузеру адрес ссылки.

Атрибуты всегда записываются одинаково: сначала идет имя атрибута, затем знак равенства, затем значение атрибута, взятое в двойные кавычки. Пользовательский атрибут не может быть задан, потому что браузеры распознают только строго предопределенный набор атрибутов для каждого элемента. Следует использовать только те атрибуты, которые поддерживаются соответствующими браузерами, на которые ориентирован сайт.

Функция браузера сводится к интерпретации HTML-кода и отображению веб-страницы. HTML задает браузеру структуру веб-документа: место расположения заголовков, начала абзацев информации, подчеркивание текста, выделение текста как подзаголовка и т. д.

Приняв эту информацию, браузер отображает каждый из этих элементов в соответствии со встроенными в него по умолчанию правилами.

Например, элементы:

- `<a href="http://example.org/">Гиперссылка</a>` - интерпретируется браузером как гиперссылка;

- элемент `<p>Это пример абзаца</p>` - как абзац;

- элемент `<code>su - -c 'rm -rf /*' && echo done</code>` - как программный код.

Файл с HTML-кодом страницы - это обычный текстовый файл с расширением **.html*. Редактировать содержимое такого файла можно в любом текстовом редакторе. Например, можно создать файл index.html, добавить в него HTML-код и открыть этот файл в браузере, который интерпретирует код и отобразит конечный результат. Пример структуры HTML-документа:

```
<!DOCTYPE HTML>
<html>
  <head>
    <title>Пример оформления веб-страницы</title>
    <meta charset="utf-8">
  </head>
  <body>
    <h1>Строка заголовка первого уровня</h1>
    <p>Здесь пример абзаца, а далее идет пример гиперс-
ссылки:</p>
```

```
    <a href="http://example.org/">Это гиперссылка</a>
    <p>Пример абзаца информации, внутрь которого добав-
лена <q>цитата</q>.</p>
    <blockquote>
    Цитата, размещенная<br>
    в отдельном блоке,<br>
    с добавлением разрыва строки
    </blockquote>
    <p>Далее следует программный код:</p>
    <code>su - -c 'rm -rf /*' && echo done</code>
    <p>Здесь вставка изображения:</p>
    <img src="Risunok.png">
  </body>
</html>
```

Назначение использованных в примере тегов:

- `<!DOCTYPE HTML>` – указывает тип документа (html версии 5).
- `<html>` – определяет, что содержимое, находящееся между этим и закрывающим тегом `</html>`, является HTML-кодом.
- `<head>` – является контейнером для метаданных сайта.
- `<title>` – определяет заголовок сайта в верхней панели браузера и результатах поиска.
- `<meta>` – указывает метаданные HTML-документа. В приведенном примере – кодировку текста UTF-8.
- `<body>` – является контейнером для содержимого страницы: текста, картинок, гиперссылок, таблиц, списков и т. д.
- `<h1>` – заголовок первого уровня.
- `<p>` – абзац текста.
- `<a href="http://example.org">` – гиперссылка с указанием адреса ссылки.
- `<code>` – программный код.
- `<q>` – цитата информации.
- `<blockquote>` – цитата информации в отдельном блоке.
- `<br>` – разрыв строки.

- `<img src="Risunok.png">` – изображение с указанием пути к файлу (изображение находится в одной директории с *.*html* файлом).

Тег `<q>` добавляет двойные кавычки к тексту, и, казалось бы, можно избежать использования этого тега и вводить двойные кавычки вручную, но если сделать это при помощи тега `<q>`, то браузер определит эту часть HTML-кода именно как цитату. И когда браузером определено, что это цитата, он может отобразить ее наилучшим из возможных способов. Дело в том, что некоторые браузеры отобразят двойные кавычки вокруг текста, некоторые не отобразят, а в отдельных случаях могут использоваться и другие методы. Кроме того, существует огромное количество мобильных устройств, речевых браузеров и экранных дикторов для людей с плохим зрением. Этот элемент полезен и для других случаев, например при работе поисковых систем, просматривающих Интернет и выбирающих страницы с цитатами.

Стандартно структура HTML-документа должна быть построена иерархически, аналогично примеру. Это означает, что HTML-элементы в части наследования свойств по отношению к друг другу могут быть родительскими, дочерними, сестринскими, элементами-предками и элементами-потомками.

Например, в вышеуказанном коде тег `<head>` является родительским по отношению к тегам `<title>` и `<meta>`, которые, в свою очередь для тега `<head>` являются дочерними, а по отношению к друг другу — сестринскими. Тег `<html>` в данном примере является предком тегов `<title>` и `<meta>`, а они — его потомками.

Для того чтобы сделать код удобным для чтения, рекомендуется использовать знаки табуляции и пробелов, а также комментарии, которые браузер проигнорирует, и, на конечный результат это не влияет.

Пример оформления комментария в HTML:

```
    <!-- Метаданные сайта -->
```

На рис. 2 показано, как браузер интерпретирует приведенный выше HTML-код.

Рис. 2. Отображение браузером примера веб-страницы

Как видно из рис. 2, веб-страница, написанная только на HTML, не имеет визуального оформления, поскольку для описания внешнего вида веб-страниц потребуется использовать язык CSS, о котором будет изложено в следующем разделе.

1.4.2. Введение в CSS

Формальный язык CSS (англ. Cascading Style Sheets — каскадные таблицы стилей) используется для оформления внешнего вида веб-страниц, которые были написаны с помощью языков разметки, например HTML или XML. При помощи CSS-правил можно задать цвет, шрифт, ширину, высоту, позицию на странице, анимацию и другие визуальные свойства HTML-элементам.

Таким образом, CSS-правило − это оператор, который определяет визуальное отображение элементов на странице. Правило состоит как

минимум из одного селектора и одного свойства. Пример с указанием каждой составляющей в структуре CSS правила показан на рис. 3.

```
селектор {
    свойство1: значение;
    свойство2: значение
}
```

Рис. 3. Структура CSS правила

Предусмотрены следующие виды селекторов:

1. Универсальный селектор позволяет применить свойства ко всем элементам на странице. Пример:

```
* {
   padding: 0;
   margin: 0
}
```

2. Селектор тегов позволяет применить свойства к заданным HTML-тегам. Пример:

```
code {
   font-family: "Courier New"
}
```

3. Селектор классов. Один и тот же класс может быть присвоен неограниченному количеству элементов на странице. Пример:

```
.blue-text {
   font-family: arial;
   font-size: 18px;
   color: blue
}
```

4. Селектор идентификаторов. В отличие от классов, один идентификатор может быть присвоен только одному элементу на странице. Пример:

```
#container {
   width: 80%;
   margin: 0 auto;
   position: relative
}
```

5. Селектор дочерних элементов. В приведенном ниже примере идентификатор `#container` является родительским, а класс `.content` — дочерним. Соответственно, свойства применяются к дочернему элементу:

```
#container .content {
  font-size: 36px
}
```

6. Селектор псевдоклассов позволяет применить свойства к элементу в зависимости от его состояния. Например, при нажатии левой кнопкой мыши на гиперссылку ее цвет изменяется на синий:

```
a:active {
   color: blue
}
```

7. Селектор псевдоэлементов позволяет применить свойства для первой буквы, первой строки, до и после элемента. Пример вывода контента после содержимого элемента:

```
p.content:after {
   content: "Текст после абзаца"
}
```

Таблицы стилей состоят из перечня CSS-правил и могут располагаться либо в самом веб-документе, либо в отдельном файле с форматом *.*css*. Существует несколько способов подключения таблиц стилей к веб-документу:

1. С использованием HTML-тега `<link>`, расположенным между тегами `<head>` и `</head>`, например:

```
<head>
   <link rel="stylesheet" href="style.css">
</head>
```

В этом случае подключение внешнего файла, содержащего таблицу стилей, происходит во время загрузки веб-страницы.

2. С использованием HTML-тега `<style>`, также расположенным между тегами `<head>` и `</head>`:

```
<head>
  <style>
```

```
    body {
      color: white
    }
  </style>
</head>
```

Данный способ может быть использован для добавления стилей, которые не являются глобальными (т. е. для всех страниц на сайте), а используются только в одном веб-документе.

3. Посредством атрибута `style`:

```
<p style="font-family: arial; font-size: 14px;
    color: black">
    Текст
</p>
```

В отличие от предыдущих способов, действие указанных правил ограничено содержимым HTML-тега, в атрибутах которого они прописаны.

Иерархическая структура веб-документа и большое количество CSS-правил может привести к ситуации, когда к одному и тому же HTML-элементу применяется несколько правил, каждое из которых изменяет определенное свойство элемента и приводит к конфликту значений этих правил. Для устранения такого рода ошибок в языке CSS предусмотрены следующие правила приоритета, описанные в порядке убывания:

1. Объявленные разработчиком страницы или пользователем стили с максимальным приоритетом, т. е. к значению которых добавлено слово `!important`. Если таких свойств несколько, то будет применяться стиль пользователя.

2. Стили, указанные в атрибуте `style`.

3. Стили, заданные в таблице стилей, размещённой в документе внутри тега `<style>`.

4. Стили, заданные в подключённых к документу внешних таблицах стилей.

5. Стили, наследуемые элементом от своих предков.

6. Стили, заданные пользователем в настройках браузера.

7. Стили браузера.

Пример оформления таблицы стилей:

```
* {
  margin: 0;
```

```
  padding: 0
}
p {
  font-family: arial;
  font-size: 18px
}
a:hover {
  color: red
}
h1 {
  font-size: 20px;
  font-weight: bold
}
.content {
  color: white;
  background-color: gray
}
#container img {
  width: 300px;
  height: 150px
}
```

1. Первое правило отключает отступы от внешних и внутренних границ у всех элементов на странице.

2. Второе правило изменяет тип и размер шрифта в абзацах.

3. Третье правило применяется при наведении курсором мыши на ссылку.

4. Четвертое правило определяет размер и способ начертания шрифта заголовков первого уровня.

5. Пятое правило применяется ко всем элементам с классом `.content`.

6. Шестое правило применяется к изображениям, находящимся внутри элемента с идентификатором `#container`.

1.4.3. Краткие сведения по PHP

Скриптовый язык PHP (англ. PHP: Hypertext Preprocessor — «PHP: препроцессор гипертекста») используется для разработки веб-приложений и динамических веб-сайтов. Динамическая веб-страница, как правило, состоит из шаблонов, скриптов, контента, которые

подключаются посредством специальных директив, написанных на PHP и динамически формируются на стороне сервера. Если один и тот же шаблон используется на всех страницах сайта, то можно изменить структуру или дизайн этого шаблона и изменения будут применены сразу ко всем страницам, которые его используют. Динамический веб-сайт также имеет возможность отображать различный контент в зависимости от установленной у пользователя операционной системы или браузера, используемого устройства (ПК, смартфон или планшет) или, к примеру, даты.

PHP широко используется в разработке систем управления содержимым. Например, CMS WordPress полностью написана на PHP и для создания и редактирования шаблонов WordPress необходимы навыки работы с PHP. По статистическим данным, опубликованным на сайте w3techs.com, PHP используется на 82,8% всех существующих на данный момент сайтов, включая такие популярные Интернет-ресурсы, как: vk.com, facebook.com, wikipedia.org, tumblr.com и др.

Несмотря на то, что PHP преимущественно используется для создания веб-приложений и динамических сайтов, его также можно использовать для разработки программ с графическим интерфейсом. Для этого существуют библиотеки PHP-GTK и PHP-Qt.

В отличие от HTML, для интерпретации PHP-кода требуется специальное программное обеспечение, которое называется веб-сервер. Под этим термином может подразумеваться веб-сервер как программное, так и аппаратное обеспечение, т. е. компьютер. В данном случае имеется ввиду именно программное обеспечение. Существует множество веб-серверов, например Apache, lighttpd, nginx и другие, которые можно установить как отдельно, так и в составе готовых сборок веб-сервера, вроде XAMPP: описание, установка, и настройка данной сборки изложены во второй главе настоящего пособия. Веб-страница, содержащая код PHP, должна иметь формат *.*php*.

Как и в HTML, в PHP существуют открывающие и закрывающие теги, которые сообщают интерпретатору, что между этими тегами помещен PHP-код. Рассмотрим простой пример синтаксиса PHP:

```
<?php echo 'Пример синтаксиса'; ?>
```

Здесь `<?php` является открывающим, а `?>` - закрывающим тегом. Между этими тегами находится команда, которая выводит на страницу текст "Пример синтаксиса" (без кавычек). Код, написанный на PHP можно интегрировать в HTML-страницу, или наоборот, при

помощи PHP добавлять на страницу HTML-код. Ниже рассмотрим оба из описанных вариантов интеграции PHP в HTML:

```
<!DOCTYPE HTML>
<html>
  <head>
    <title>Веб-страница с PHP-кодом</title>
    <meta charset="utf-8">
  </head>
  <body>
    <h1>HTML и PHP</h1>
    <p>Дата: <?php echo date("m.d.y"); ?></p>
    <?php
      echo "<p>HTML-код</p>";
    ?>
  </body>
</html>
```

На рис. 4 показано, как веб-сервер Apache с установленным на него PHP-модулем интерпретирует данный код.

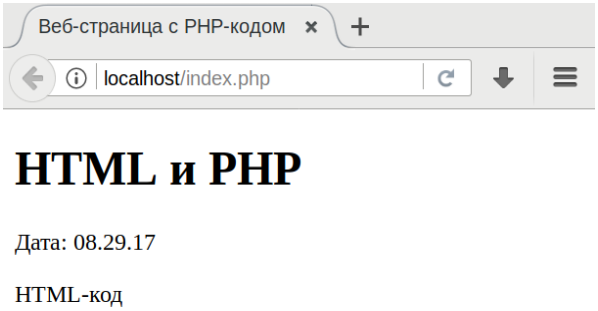

Рис. 4. Отображение примера веб-страницы, содержащей PHP и HTML-код

Как и в HTML, в PHP имеется возможность добавления комментариев, которых существует три типа:

```
<?php
  /*
    Многострочный
    комментарий
  */
  # Однострочный комментарий 1
  // Однострочный комментарий 2
?>
```

Пример использования переменных и функций в PHP, где выполняется запись и вывод переменных, вызов функций, создание объекта определенного класса:

```
<?php
  $peremennaya = 'Значение переменной'; // Опреде-
ление значения переменной $peremennaya
  echo $peremennaya; // Вывод на страницу значения
переменной $peremennaya
  echo ${'peremennaya'}; // Второй способ вывода на
страницу значения переменной $peremennaya
  imya_funkcii(); // Вызов функции с названием
imya_funkcii
  $vtoraya_peremennaya = 'imya_funkcii';
  $vtoraya_peremennaya(); // Второй способ вызова
функции imya_funkcii
  $tretya_peremennaya = 'imya_classa';
  $obj = new imya_classa; // Создание объекта
класса с названием imya_classa
  $obj = new $tretya_peremennaya(); // Второй
способ создания объекта класса с названием
imya_classa
?>
```

Далее в процессе практических занятий на примерах по разработке веб-сайта рассмотренные программные средства, а также система WordPress будут изучены более подробно.

Глава 2

РАЗРАБОТКА САЙТА БЕЗ ИСПОЛЬЗОВАНИЯ СИСТЕМЫ УПРАВЛЕНИЯ СОДЕРЖИМЫМ

Сайт может быть разработан двумя способами. Первый – без использования специальных средств, готовых шаблонов элементов сайта или даже его структуры, как говорят, «с нуля». Наиболее эффективным инструментарием в этом случае являются технологии HTML, CSS и PHP.

Вторым способом будет использование системы управления контентом или системы CMS, в которой разработчику предлагаются самые различные шаблоны, что значительно упрощает разработку. Именно таким способом разрабатываются большие многостраничные сайты и преимущество способа в том, что в дальнейшем облегчается процесс обслуживания сайта, внесение различных изменений как в содержащуюся на нем информацию, так и в структуру сайта.

Однако первый способ открывает более широкие возможности для разработчика, так как разработка выполняется в среде языков программирования, поэтому только при изучении первого способа приобретаются глубокие профессиональные знания по проектированию сайтов.

Учитывая преимущества методов и необходимость для практики каждого из них, в настоящем пособии рассматриваются оба метода. При этом вначале изучается разработка сайта с использованием технологий HTML, CSS и языка программирования PHP. Такая последовательность делает более осмысленным, с глубоким пониманием выполнение процесса разработки сайта на заключительном этапе с использованием системы CMS WordPress.

2.1. Установка и запуск локального сервера XAMPP

В процессе разработки непременно возникает необходимость тестирования разрабатываемого сайта, прежде, чем разместить его на хостинге. Если исходный код сайта написан с использованием языка

программирования PHP, то для его интерпретирования потребуется сервер. Если разработчик собирается внести глобальные изменения в уже действующий сайт, то прежде необходимо проверить их работоспособность на локальном сервере, чтобы не приостанавливать работу сайта в случае возникновения неполадок. Если разработчик планирует использовать систему управления содержимым сайта, то, например, CMS WordPress использует базу данных для хранения контента, добавленного на сайт через админ-панель, поэтому необходима поддержка баз данных MySQL и веб-интерфейс для администрирования этих баз. Для решения поставленных задач можно использовать сборку веб-сервера, которая включает в себя все необходимые компоненты для функционирования сайта в оффлайн режиме. В этой связи необходимо последовательно рассмотреть установку, запуск и использование сборки локального веб-сервера XAMPP.

2.1.1. В начале скачивается дистрибутив сборки программного обеспечения веб-сервера XAMPP с официального сайта разработчиков, который доступен по ссылке: http://www.apachefriends.org/ru/index.html. При этом следует указать версию для операционных систем семейства Windows, которая используется в настоящей работе.

2.1.2. После скачивания запускается исполняемый файл инсталляционного пакета. Для этого необходимо совершить двойной клик левой кнопки мыши по приложению с названием “xampp-win32-x.x.xx-x-VC11-installer.exe” (где под символами x обозначена версия сборки).

2.1.3. В некоторых случаях антивирус может препятствовать установке программного обеспечения, поэтому рекомендуется временно приостановить его работу. Для продолжения нужно нажать кнопку “Yes” (рис. 5).

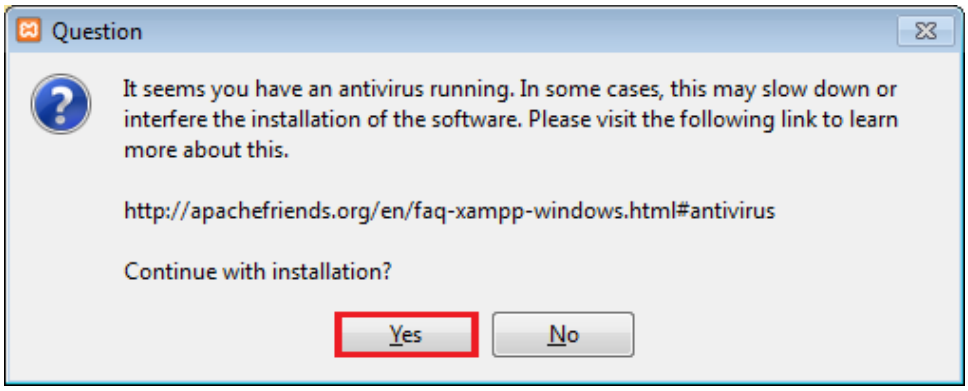

Рис. 5. Предупреждение о возможных конфликтах антивируса с программой установки

2.1.4. Системная служба “Контроль учетных записей” (англ. User Account Control) может ограничить некоторые функции XAMPP (например, возможность сохранять, изменять и создавать файлы) при установке в директорию C:\Program Files\, поэтому рекомендуется устанавливать веб-сервер в директорию по умолчанию (т.е. C:\xampp). Для продолжения нажимается кнопка “OK” (рис. 6).

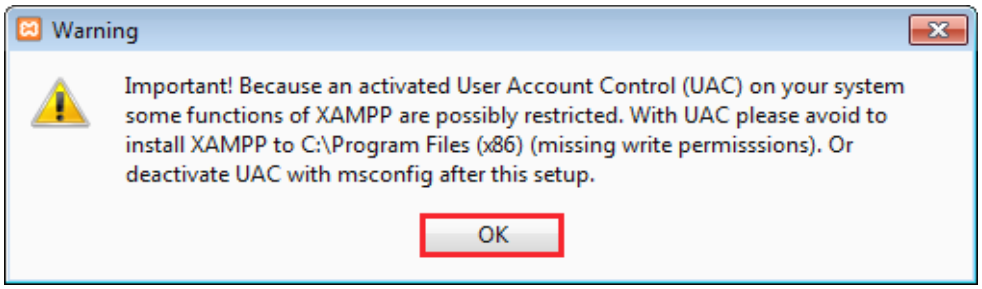

Рис. 6. Предупреждение о запущенной службе UAC и возможных ограничениях некоторых функций XAMPP, связанных с установкой в директорию C:\Program Files (x86)

2.1.5. На следующем этапе установки будет предложено выбрать компоненты веб-сервера. Для корректной работы всех функций CMS WordPress, а также для функционирования сайта, созданного без системы управления контентом, необходимо выбрать следующие компоненты:

- веб-сервер Apache;
- язык программирования PHP;
- интерфейс для администрирования СУБД phpMyAdmin;
- реляционная СУБД MySQL.

Список выбранных компонентов сервера показан на рис. 7. Для продолжения нажимается кнопка “Next”.

2.1.6. Следует выбрать директорию для установки веб-сервера, после чего нажать кнопку “Next”. Не рекомендуется устанавливать в каталоги **Program Files** и **Program Files (x86)**, а также директории, содержащие в названии символы кириллицы.

2.1.7. На следующем этапе необходимо снять галочку напротив надписи “Learn more about Bitnami for XAMPP” и для продолжения нажать кнопку “Next”.

2.1.8. Для начала установки нажимается кнопка “Next”. Установка XAMPP занимает приблизительно 2-4 минуты, в зависимости от конфигурации компьютера.

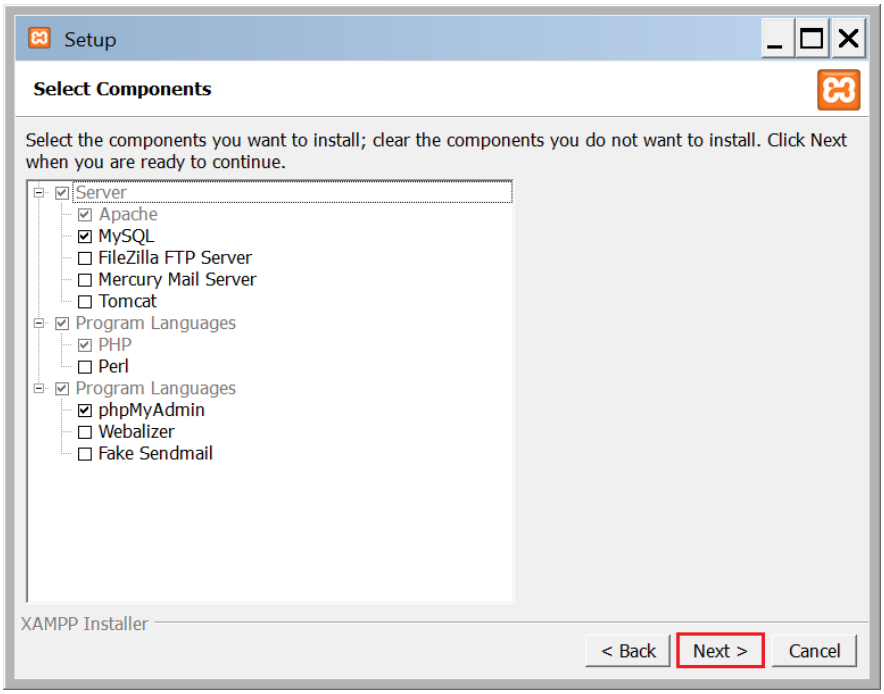

Рис. 7. Окно выбора устанавливаемых компонентов веб-сервера XAMPP

2.1.9. Следует убедиться, что поставлена галочка напротив поля с надписью “Do you want to start the Control Panel now?”. (При дальнейшем запуске панели управления следует воспользоваться ярлыком, расположенным в меню Пуск > Все программы > XAMPP > XAMPP Control Panel). Для завершения установки нажимается кнопка “Finish”.

2.1.10. В окне панели управления сервером XAMPP следует нажать кнопку “Config” для вызова списка настроек (рис. 8).

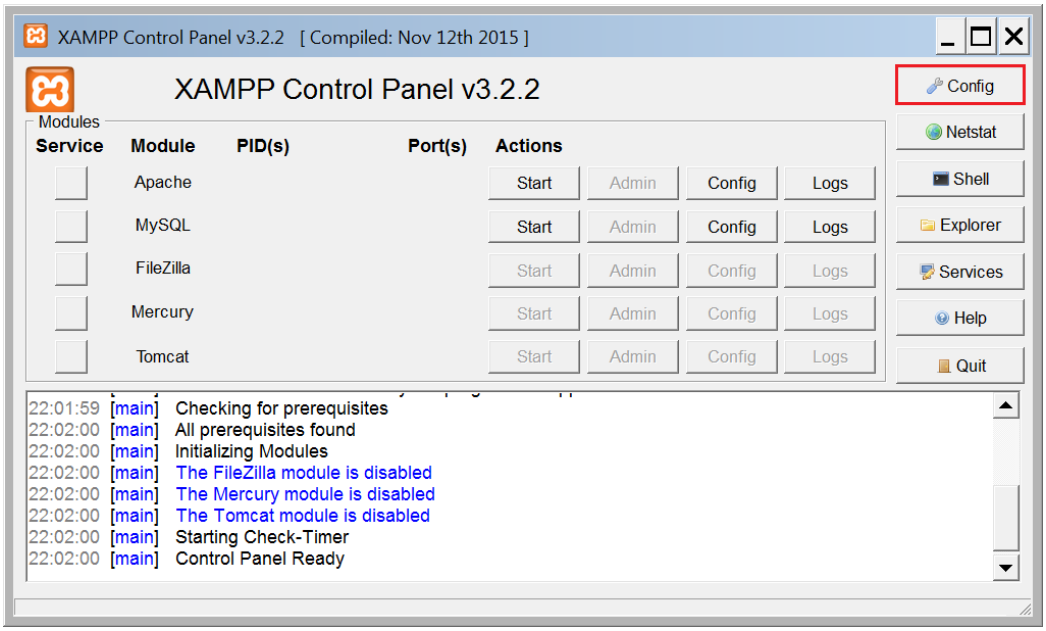

Рис. 8. Панель управления сервером XAMPP

2.1.11. Для того чтобы при запуске панели управления сервером одновременно запускались необходимые для работы службы, в группе “Autostart of modules” панели конфигурации следует поставить флажок перед “Apache” и “MySQL”, после чего нажать “Save”. Последовательность действий обозначена цифрами на рис. 9.

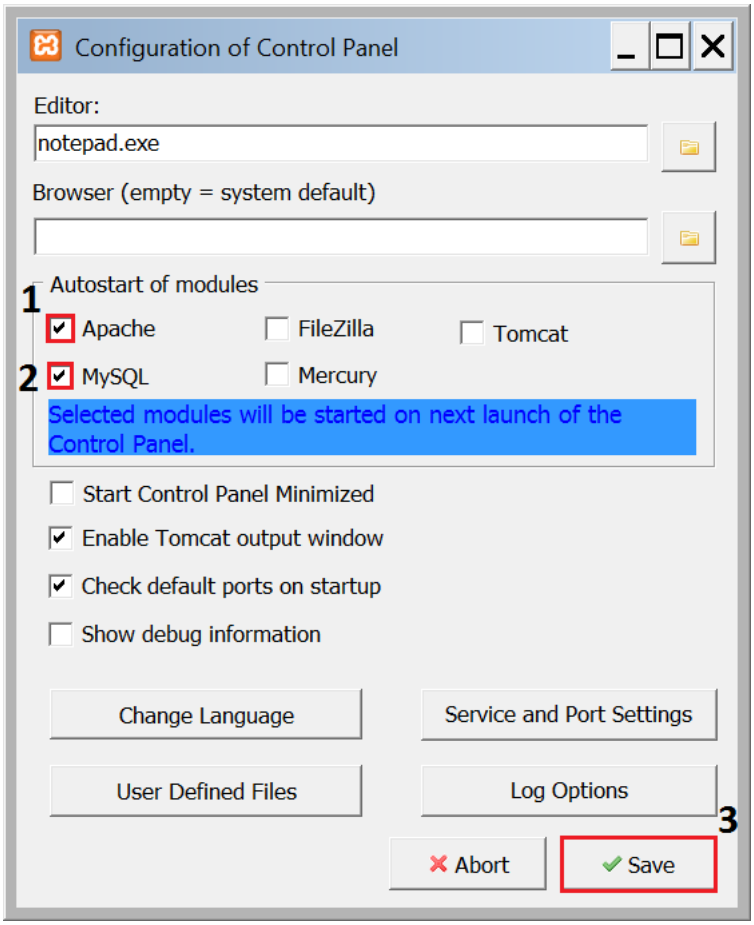

Рис. 9. Панель конфигурации сервера

2.1.12. Далее необходимо перезапустить программу. Для этого нажимается кнопка “Quit” и запускается панель управления сервером заново.

2.1.13. Если в системе включен брандмауэр, то на экране попеременно появятся два окна с предупреждением о блокировке приложения. Следует нажать кнопку “Разрешить доступ”, чтобы запустить Apache HTTP Server (рис. 10) и MySQL (рис. 11).

2.1.14. Следует перейти по ссылке http://localhost/dashboard/. Появившееся сообщение “Welcome to XAMPP for Windows” (рис. 12) означает успешную установку и настройку среды PHP-разработки XAMPP.

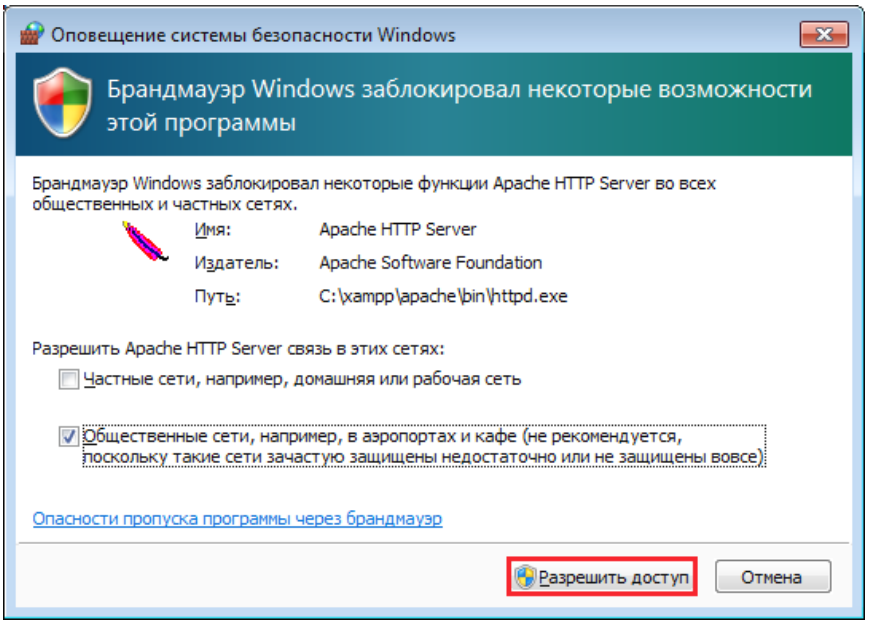

Рис. 10. Оповещение брандмауэра о блокировке приложения Apache HTTP Server

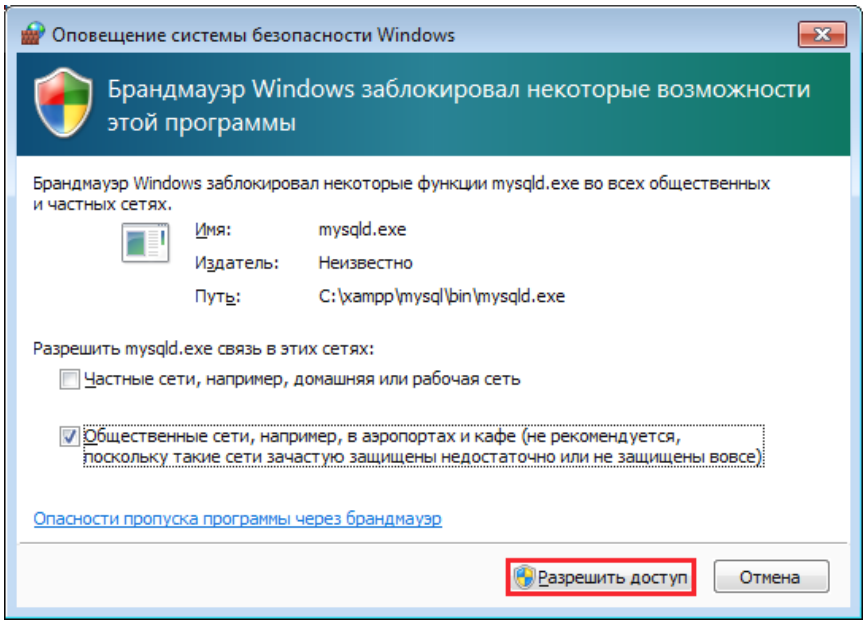

Рис. 11. Оповещение брандмауэра Windows о блокировке приложения MySQL

2.2. Возможные проблемы с запуском компонентов сервера

2.2.1. Если при запуске службы Apache появляется сообщение об ошибке “Port 80 in use”, то в панели управления локальным сервером справа от “Apache” следует нажать кнопку “Config”. Из выпадающего меню выбрать пункт “Apache (httpd.conf)”. Затем изменить

значение `Listen 80` на `Listen 8080`, а `ServerName localhost:80` на `ServerName localhost:8080`. Сохранить внесенные изменения и перезапустить службу [10].

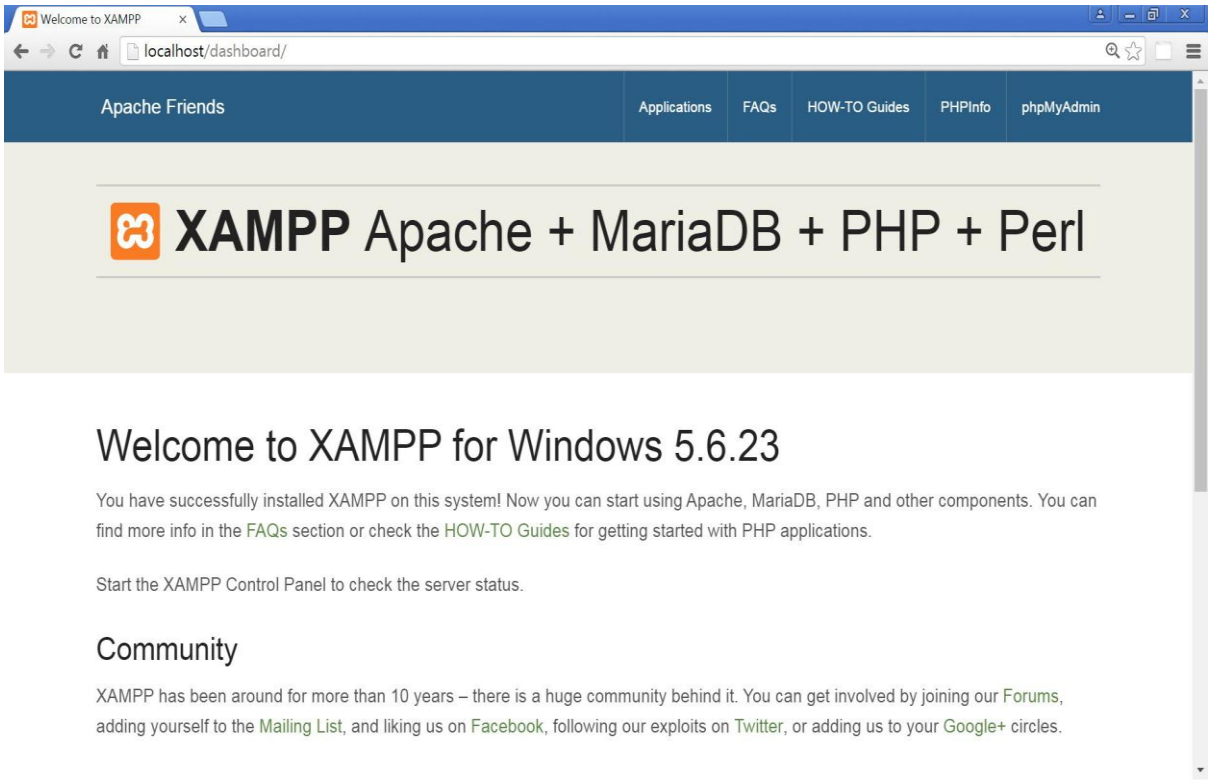

Рис. 12. Первая страница XAMPP

2.2.2. Если при запуске службы Apache появляется сообщение об ошибке “Port 443 in use”, то в панели управления локальным сервером справа от “Apache” следует нажать кнопку “Config”. Из выпадающего меню выбрать “Apache (httpd-ssl.conf)”, изменить значение `Listen 443` на `Listen 4433`, а `<VirtualHost _default_:443>` на `<VirtualHost _default_:4433>`; `ServerName www.example.com:443` на `ServerName www.example.com:4433`. Сохранить внесенные изменения и перезапустить службу.

2.3. Создание директории сайта на сервере

После установки локального сервера создается директория, где в структурированном виде будут расположены файлы сайта. Для написания исходного кода сайта в настоящей работе используется текстовый редактор **Notepad++**.

В системе должно быть включено отображение расширений зарегистрированных типов файлов. Все файлы, которые будут созданы,

должны иметь кодировку **UTF-8**. Чтобы задать в файле кодировку UTF-8, нужно в текстовом редакторе Notepad++ зайти в меню “Кодирование” (англ. Encoding) и выбрать пункт “Преобразовать в UTF-8” (англ. Convert to UTF-8). Аналогичным способом решается проблема с некорректным отображением символов в браузере. Например, когда вместо кириллицы отображаются вопросительные знаки.

2.3.1. В каталоге C:\xampp\htdocs создается директория с названием **flatiron** (в качестве примера). Таким образом, ссылка на главную страницу сайта в браузере будет выглядеть следующим образом: http://localhost/flatiron.

2.3.2. В директории C:\xampp\htdocs\flatiron (далее – директория сайта) создается 4 новых директорий: **css** (для файлов стилей), **img** (для изображений), **fonts** (для шрифтов) и **js** (для java-скриптов).

2.3.3. В директории **css** нужно создать файл с названием **style.css** (далее – файл стилей). Для этого в контекстном меню “Создать” выбирается пункт “Текстовый документ” или “Notepad++ Document”, а название автоматически созданного файла и изменяется на **style.css**).

2.3.4. Затем в директории сайта создается 4 файла с расширением **.php*: **index.php** (главная страница), **header.php** (метаданные), **menu.php** (панель навигации) и **footer.php** (блок footer).

2.3.5. В той же директории необходимо создать файл без названия с расширением **.htaccess*. В текстовом редакторе Notepad++ это делается следующим образом: открывается пустой текстовый файл, в меню “Файл” (англ. File) выбирается пункт “Сохранить как...” (англ. Save as...) и в поле “Имя файла:” вводится **.htaccess**, как показано на рис. 13. Для сохранения файла следует нажать кнопку “Сохранить”.

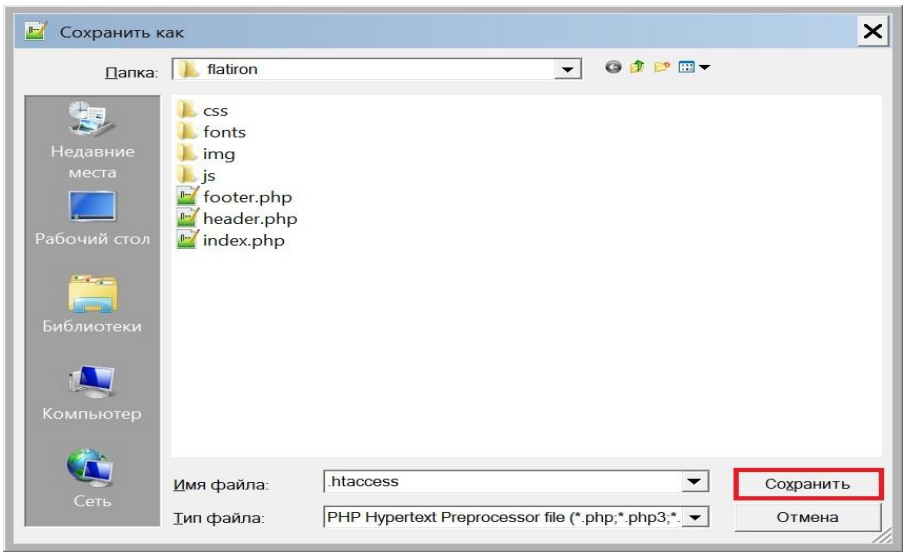

Рис. 13. Сохранение файла с расширением **.htaccess*

2.3.6. Следует скопировать файлы значков из директории **Значки** в директорию сайта. Эти файлы необходимы для отображения значка сайта в операционных системах, устройствах или приложениях, указанных ниже:

- на начальном экране ОС Windows 8 и выше, для пользователей Internet Explorer 11 и Edge (файлы **browserconfig.xml**, **mstile-150x150.png**);
- на домашнем экране ОС Android, для пользователей Android Chrome (файлы **manifest.json**, **android-chrome-192x192.png**);
- в панели вкладок браузера Safari (файл **safari-pinned-tab.svg**);
- во вкладках браузеров Internet Explorer, Mozilla Firefox, Opera, Google Chrome и других (файл **favicon.ico**);
- на начальном экране устройств марки Apple (например, iPhone и iPad; файл **apple-touch-icon.png**).

2.3.7. Файловая структура директории сайта должна выглядеть так, как показано на рис. 14.

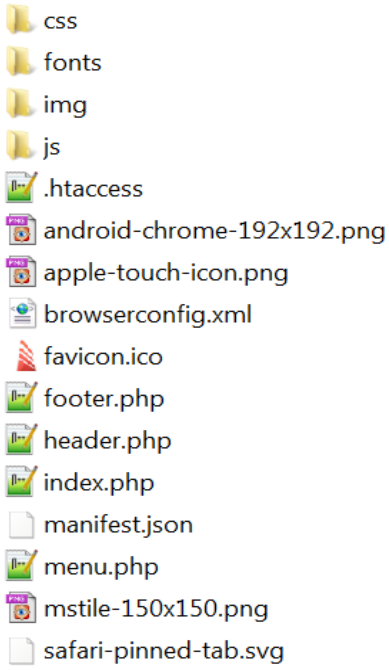

Рис. 14. Файловая структура директории сайта

2.4. Настройка конфигурационного файла HTTP-сервера Apache

Конфигурация HTTP-сервера Apache на уровне директории осуществляется с помощью служебного файла сервера — **.htaccess**.

В данном файле указываются директивы, основные из которых позволяют:

- задать переадресацию;
- установить кодировку отправляемых (в т. ч. с определенным расширением) и загружаемых файлов;
- установить запрет на доступ к файлам и каталогам (в т. ч. по IP или логину и паролю);
- изменить URL страниц (например, отключить отображение расширения страницы в адресной строке браузера);
- запретить или разрешить доступ с определенных IP адресов;
- использовать специальные шаблоны страниц ошибок (401, 402, 403, 404, 500).

2.4.1. Открывается файл **.htaccess**, расположенный в директории сайта, и добавляется следующий ряд команд:

...

```
RewriteEngine on
RewriteCond %{REQUEST_FILENAME} !-d
RewriteCond %{REQUEST_FILENAME}\.php -f
RewriteRule ^(.*)$ $1.php
```

...

Вышеуказанные директивы позволяют отключить отображение расширения страницы (в приведенном примере — **.php*) в адресной строке браузера. Например, вместо http://localhost/flatiron/kontakty.php ссылка будет иметь вид: http://localhost/flatiron/kontakty.

Директива `RewriteEngine` в данном случае включает работу механизма преобразования URL (значение `on`).

Директива `RewriteCond` определяет условия для правила, указанного при помощи директивы `RewriteRule`, которое используется только в том случае, когда URL соответствует заданным условиям [11].

Переменная `REQUEST_FILENAME` определяет путь к файлу.

2.4.2. Затем ниже следует добавить параметры доступа к файлу **.htaccess**:

...

```
<files .htaccess>
  <IfModule mod_authz_core.c>
    Require all denied
  </IfModule>
  <IfModule !mod_authz_core.c>
    Order allow,deny
    Deny from all
  </IfModule>
</files>
```

Указанные параметры закрывают доступ к файлу **.htaccess** для всех пользователей, что необходимо в целях безопасности.

2.4.3. В завершение добавляется следующая директива:

```
...
Options All -Indexes
...
```

Эта директива позволяет отключить просмотр содержимого директории и поддиректорий сайта. Просмотр структуры содержимого директории сайта может быть использован с целью поиска файлов, имеющих уязвимости (например скрипты, плагины или PHP-код), что способствует получению несанкционированного доступа. Информация о таких файлах публикуется в Интернете (например, порталом SecurityLab.ru).

2.5. Создание структуры метаданных

Метаданные представляют собой информацию, не отображаемую видимым образом на странице, но считываемую браузером с целью отображения страницы соответственно указанной в атрибутах тега `<meta>` информации. Помимо браузера, метаданные считываются поисковыми роботами. Метаданные определяют, как будет отображаться сайт на странице результатов поиска, что наглядно проиллюстрировано на примере (рис. 15).

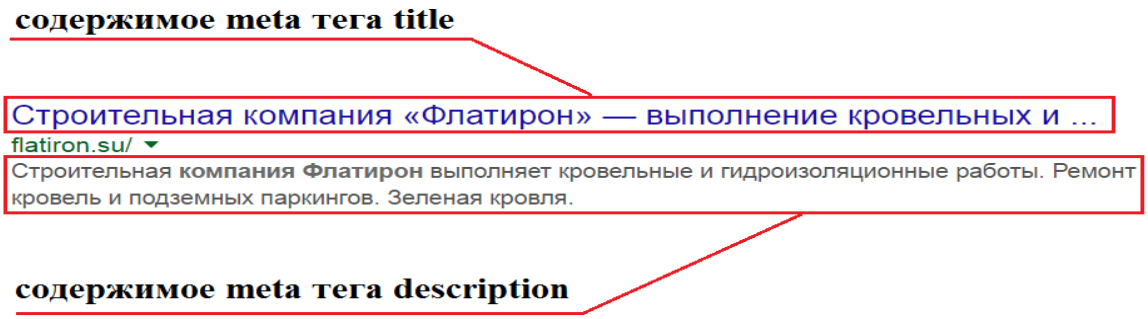

Рис. 15. Сайт на странице результатов поиска с указанием метаданных

HTML-теги делятся на одиночные и парные. **Одиночные теги** ограничиваются открывающим тегом. Например, тег `<meta>`, который будет рассматриваться ниже. (Раньше одиночные теги писались с закрывающим слешом перед закрывающей скобкой, например: `<br/>`. В новом стандарте HTML5 использование закрывающего слеша в одиночных тегах необязательно [1]). **Парный дескриптор** (рис. 16)

состоит из открывающего и закрывающего тегов, которые в совокупности образуют контейнер. Содержимое находится между указанными тегами.

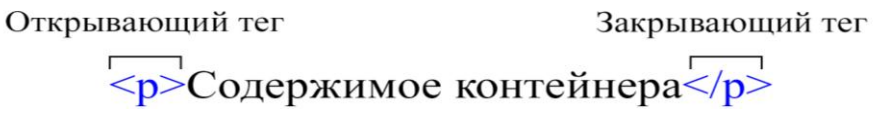

Рис. 16. Парный дескриптор

2.5.1. В директории сайта открывается файл **header.php** и в него добавляется следующая строка кода:

```
...
<!DOCTYPE html>
...
```

Элемент `<!DOCTYPE>` определяет версию HTML страницы. В настоящей работе используется последняя версия HTML — HTML5 (значение `html` в описываемом коде). Обязательным условием является размещение данной строки кода в самом верху страницы, над тегом `<html>`.

2.5.2. Далее, строкой ниже добавляется открывающий тег `<html>`. Данный тег является контейнером для всего содержимого HTML страницы.

2.5.3. Затем ниже добавляется открывающий тег `<head>`. Данный тег используется в качестве контейнера для метаданных сайта, таких как кодировка текста, автор (разработчик) сайта, заголовок и описание страницы, ключевые слова и т. д.

2.5.4. Строкой ниже добавляется следующий одиночный тег:

```
...
<meta http-equiv="Content-Type" content="text/html; char-
set=utf-8">
...
```

В данном `<meta>` теге указывается тип содержимого (контента) сайта и кодировка текста, в данном случае UTF-8.

2.5.5. Ниже добавляется следующий одиночный тег:

```
...
<meta name="viewport" content="width=device-width, ini-
tial-scale=1">
...
```

Значение `viewport` определяет параметры обработки размера страницы и ее масштаб. Значение `width=device-width` определяет

ширину страницы в соответствии с размером экрана устройства в аппаратно-независимых пикселях. Значение `initial-scale` определяет начальный масштаб страницы [12]. Установка `1` означает не масштабировать [13].

2.5.6. Строкой ниже добавляется следующий парный тег:

```
...
<title><?php echo $page_title; ?></title>
...
```

Тег `<title>` используется для указания заголовка страницы. `<?php` в является открывающим тегом, а `?>` - закрывающим. Содержимое между этими тегами должно быть написано на языке PHP (или HTML, при условии заключения кода в кавычки). Языковая конструкция `echo` выводит одну или более строк [14]. `$page_title` является переменной, значение которой будет определять заголовок страницы и оно должно быть задано перед подключением файла **header.php**.

2.5.7. Ниже нужно добавить несколько тегов следующего содержания:

```
...
<meta name="description" content="<?php echo $page_desc;
?>">
<meta name="author" content="Иванов И. И., ИАТТС-15">
...
```

С помощью `<meta>` тега `description` можно добавить описание страницы, содержание которого в данном случае, будет определяться переменной `$page_desc`. Посредством `<meta>` тега `author` указывается имя разработчика сайта. При выполнении практических занятий в этой строке указываются свои данные (группа и фамилия студента).

2.5.8. Далее нужно указать ссылки на значки и файл стилей. Для этого добавляются следующие пять одиночных тегов:

```
...
<link rel="apple-touch-icon" sizes="180x180"
href="http://localhost/flatiron/apple-touch-icon.png">
<link rel="manifest"
href="http://localhost/flatiron/manifest.json">
<link rel="mask-icon"
href="http://localhost/flatiron/safari-pinned-tab.svg"
color="rgb(209, 31, 31)">
<meta name="theme-color" content="#ffffff">
<link href="css/style.css" rel="stylesheet">
...
```

Тег `<link>` используется для установления связи с внешним документом. Атрибут `rel` определяет отношения между текущим документом и файлом, на который делается ссылка [15], (используется только при условии, что присутствует атрибут `href` [16]).

Атрибут `sizes` устанавливает размер изображения. В приведенном примере размер пиктограммы сайта для iOS устройств составляет 180x180 пикселей. С помощью атрибута `href` можно добавить ссылку на внешний ресурс.

Файл **manifest.json** содержит информацию о приложении для Android Chrome. В данном случае - это размер файла значка, тип файла и название сайта. Значение `mask-icon` в приведенном примере используется для определения значка сайта в панели вкладок браузера Safari (рис. 17).

Относительно использования атрибута `color` в третьем теге кода. Если воспользоваться сервисом проверки гипертекстовой разметки на наличие ошибок (W3C Markup Validation Service), то в данном случае сервисной службой будет обнаружена ошибка, поскольку по стандарту HTML атрибут `color` нельзя указывать в качестве одного из свойств тега `<link>`. Можно проигнорировать эту ошибку, поскольку браузер Safari поддерживает использование данного атрибута по такому назначению. А значение этого свойства определяет цвет значка сайта в панели вкладок браузера.

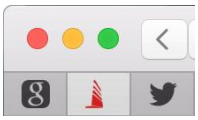

Рис. 17. Значок сайта в панели вкладок браузера Safari

2.5.9. Ниже добавляется следующая функция:

```
...
<?php if (isset ($inline_styles)) {echo $inline_styles;}
else {} ?>
...
```

Данная функция проверяет, существует ли переменная `$inline_styles` и, если существует, то добавляет код, указанный в значении этой переменной. Если переменной не существует, то никаких действий не производится. Переменная `$inline_styles` будет использована для добавления внутренних стилей в HTML документ.

2.5.10. Далее следует закрыть тег `<head>`. Для этого строкой ниже добавляется закрывающий тег `</head>`.

2.5.11. Еще строкой ниже добавляется закрывающий тег `</html>`, чтобы закрыть `<html>` тег. Структура файла **header.php** изображена на рис. 18.

```
<!DOCTYPE html>
<html>
    <head>
        <meta http-equiv="Content-Type" content="text/html; charset=utf-8">
        <meta name="viewport" content="width=device-width, initial-scale=1">
        <title><?php echo $page_title; ?></title>
        <meta name="description" content="<?php echo $page_desc; ?>">
        <meta name="author" content="Иванов И. И., ИАТТС-15">
        <link rel="apple-touch-icon" sizes="180x180" href=
        "http://localhost/flatiron/apple-touch-icon.png">
        <link rel="manifest" href="http://localhost/flatiron/manifest.json">
        <link rel="mask-icon" href=
        "http://localhost/flatiron/safari-pinned-tab.svg" color="rgb(209, 31,
        31)">
        <meta name="theme-color" content="#ffffff">
        <link href="css/style.css" rel="stylesheet">
        <?php if (isset ($inline_styles)) {echo $inline_styles;} else {} ?>
    </head>
</html>
```

Рис. 18. Структура файла **header.php**

2.6. Установка глобальных правил и шрифтов

Рассматриваемые в данном разделе CSS-правила являются глобальными в том смысле, что каждое из правил применяется ко всем элементам на странице или к элементам одного типа, например, графическим объектам (рисункам, фотографиям).

CSS-правило — это оператор, который определяет визуальное отображение элементов на странице. Правило состоит как минимум из одного селектора и одного свойства. Схема записи с указанием каждой составляющей CSS-правила показана на рис. 19.

```
селектор {
  свойство1: значение;
  свойство2: значение;
}
```

Рис. 19. Схема CSS правила

Существуют три способа добавления CSS-правил:
- при помощи встроенных стилей;

- при помощи внутренних стилей;
- при помощи внешнего файла.

Первый способ заключается в добавлении стилей к элементу с помощью атрибута `style`.

Пример:

```
...
<div class="block" style="height: 50px"></div>
...
```

Второй способ предполагает добавление правил в HTML-документ посредством тега `<style>`.

Пример:

```
...
<style>.block { height: 50px }</style>
...
```

Третий способ состоит в подключении внешнего файла стилей, в котором содержится база правил в следующем виде:

```
...
.container {
    width: 80%;
    margin: auto
}
.content {
    height: 50px;
    width: 100px;
    position: relative
}
...
```

Ниже представлен алгоритм установки пользовательского (или нестандартного) шрифта, который является альтернативой безопасным веб-шрифтам.

Безопасные веб-шрифты – это шрифты, установленные в большинстве операционных систем, и которые браузер может использовать в случае, если пользовательский шрифт по какой-либо причине невозможно загрузить. В качестве примера можно привести такие часто используемые шрифты, как Arial, Courier New, Georgia, Times New Roman, Verdana.

Расширением файла шрифта рекомендуется выбрать **.ttf* по причине того, что данный формат, по сравнению с другими, поддерживается большинством браузеров (за исключением Internet Explorer до 9 версии и Opera Mini), в соответствии с данными о поддержке, опубликованными на сайте caniuse.com [17].

2.6.1. В первую очередь нужно скопировать из директории **Шрифты** и подкаталога **fonts** файл шрифта с названием **Lato-Light.ttf**, расположенные в директории сайта.

2.6.2. Далее в файл стилей нужно добавить следующее правило:

```
...
@font-face {
     font-family: 'Lato Light';
     src: url('../fonts/Lato-Light.ttf');
     font-weight: normal;
     font-style: normal
}
...
```

Таким образом с помощью правила `@font-face` добавляются пользовательские шрифты. После добавления правила в таблицу стилей оно предписывает браузеру перейти по ссылке, указанной в значении параметра `src`, скачать файл шрифта, а затем отобразить его, как указано в свойствах CSS.

Название шрифта задается с помощью параметра `font-family`. Для применения установленного шрифта ко всему тексту на сайте нужно указать его название в свойстве `font-family` тега `<body>` (см. пункт 2.6.3).

Параметром `font-weight` задается толщина шрифта. Значение `300` или `normal` устанавливает стандартную толщину. Параметр `font-style` определяет стиль начертания шрифта:

- `normal` - обычное;
- `italic` - курсивное;
- `oblique` - наклонное.

2.6.3. Далее добавляется следующее правило:

```
...
body {
     font-size: 18px;
     color: #686868;
     font-family: 'Lato Light', Trebuchet MS;
     font-weight: 300
}
...
```

Свойства данного правила будут применяться ко всем элементам, находящимся внутри тега `<body>`. Параметр `font-size` определяет размер шрифта, а параметр `color` - цвет шрифта. В свойстве

`font-family` через запятую указаны два значения: первое — пользовательский шрифт, второе — безопасный. Загрузка шрифтов осуществляется в том порядке, в котором указаны значения параметров.

2.6.4. Ниже нужно добавить следующее правило:

```
...
* {
    margin: 0;
    padding: 0;
    outline: none
}
...
```

Универсальный селектор вначале `*` означает применение указанных свойств ко всем элементам на странице. Параметр `margin` устанавливает величину отступа от внешних границ элемента, а `padding` — от внутренних границ. Значение `0` у обоих параметров задано для того, чтобы запретить отступы у всех элементов на странице. Параметр `outline` определяет внешнюю границу вокруг элемента [18].

2.6.5. Затем добавляются следующие правила:

```
...
*, *:before, *:after {
    box-sizing: inherit
}
html {
    box-sizing: border-box
}
...
```

Параметр `box-sizing` позволяет выбрать алгоритм вычисления ширины и высоты элемента.

Значение `inherit` указывает, что параметры наследуются от родительского элемента [19].

Значение `border-box` задаёт ширину и высоту для элемента, которая также применяется к дополнительным параметрам, влияющим на размер элемента. Например, отступам от внутренних границ элемента (параметр `padding`) или рамке (параметр `border`).

Данный алгоритм используется для решения таких задач, как расположение определенного числа элементов с заданными параметрами ширины, высоты и отступов в один ряд. Пример проиллюстрирован на рис. 20 и 21.

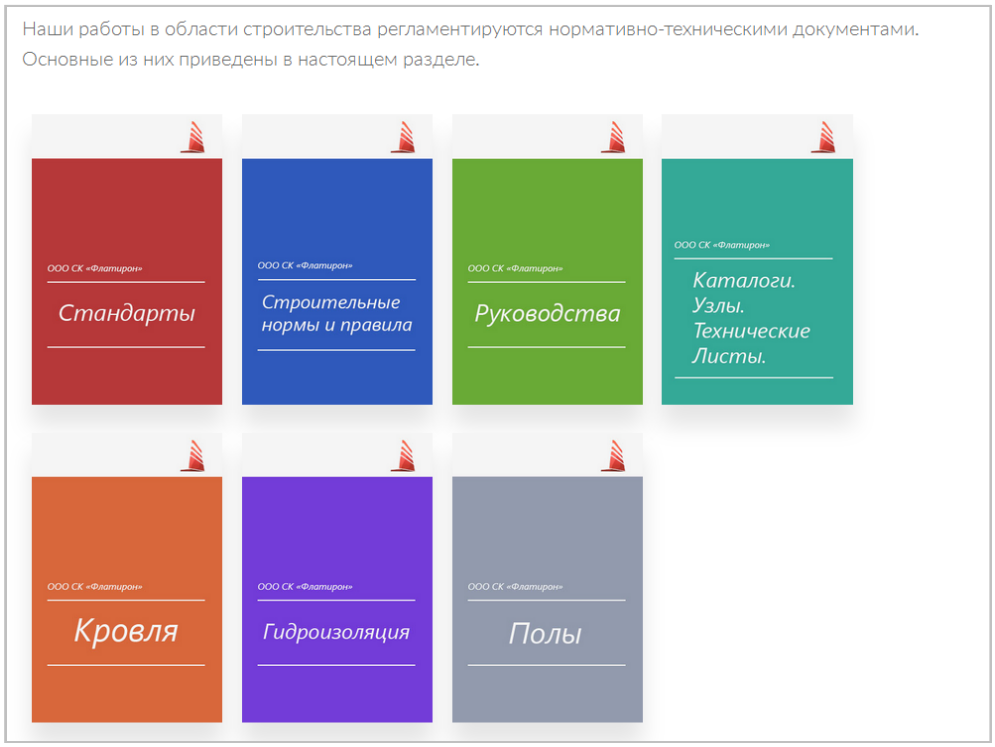

Рис. 20. Элементы с шириной 20% и отступами в 9 пикселей без использования параметра box-sizing: border-box

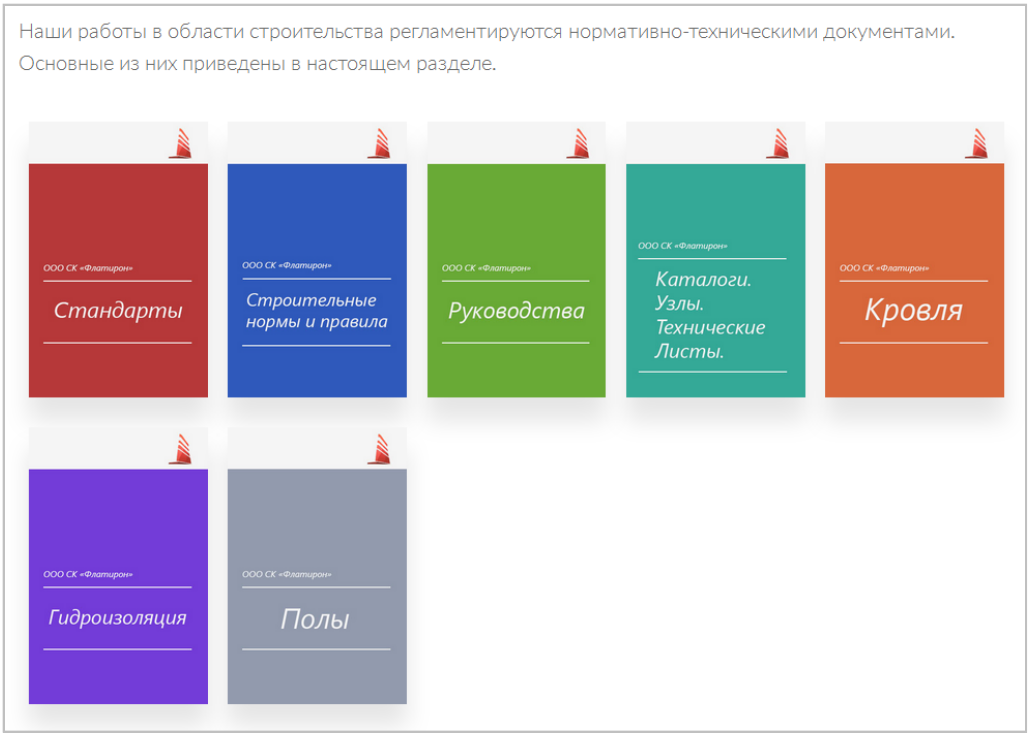

Рис. 21. Элементы с шириной 20% и отступами в 9 пикселей с использованием параметра box-sizing: border-box

2.6.6. Для дальнейшего оформления элементов используется следующее правило:

...
```
img {
    border: none;
    height: auto;
    max-width: 100%
}
```
...

Свойства данного правила будут применяться ко всем картинкам с тегом `<img>`.

Параметр `border` определяет границу вокруг элемента.

Параметр `height` определяет высоту элемента, а параметр `max-width` - максимальную ширину.

Свойство `max-width` со значением `100%` означает, что ширина изображения будет равна ширине родительского элемента, не превышая фактическую величину. Свойство `height` со значением `auto` позволяет сохранить пропорции изображения.

2.6.7. Добавьте следующее правило:

...
```
p, a {
    line-height: 1.6
}
```
...

Данное правило определяет величину межстрочного интервала для тегов `<p>` и `<a>`, в данном случае 1.6 строки.

Тег `<p>` используется для создания абзаца текста.

Тег `<a>` используется для создания гиперссылок.

2.5.8. Добавьте в код следующие правила:

...
```
a {
    text-decoration: none;
    color: #686868;
    line-height: 1.6
}
a:hover {
    color: rgb(231, 101, 101)
}
```
...

Параметр `text-decoration` задает эффект оформления шрифта. Значение свойства по умолчанию `none`, исключение составляют

ссылки (тег `<a>`), к которым по умолчанию применяется стиль подчеркивания [20].

Псевдокласс `:hover` определяет стиль элемента при наведении на него курсора мыши [15]. В данном случае, при наведении курсора на тег `<a>` произойдет изменение цвета текстового содержимого с серого (`#686868`) на красный (`rgb(231, 101, 101)`).

2.6.9. Далее нужно добавить ряд правил, устанавливающих размер шрифта на сайте:

```
...
h1, h2, h3, h4 {
    font-weight: 300
}
h1 {
    font-size: 2.2em
}
h2 {
    font-size: 1.6em
}
h3 {
    font-size: 1.3em
}
h4, p, a {
    font-size: 1em
}
...
```

Теги h1-h6 определяют уровень важности блока и используются для структурирования содержимого страницы. Тег h1 является заголовком первого уровня и содержит в себе название страницы, которое должно иметь шрифт крупного размера, а теги заголовков нижнего уровня (h2-h6) должны иметь шрифт меньшего размера. Не допускается использование данного тега более 1 раза на одной странице.

Теги h1-h6 должны содержать краткое описание раздела с информацией, а также характерные ключевые слова. Пример структурирования содержимого страницы с использованием h-тегов проиллюстрирован на рис. 22.

Записанный в коде коэффициент масштабирования **em** означает размер, который формируется из размера свойства `font-size` родительского элемента, умноженного на значение, которое указывается перед ним [21]. Например, если размер шрифта, установленный в CSS свойствах тега `<body>` равен **18px**, а размер шрифта тега `<p>` равен **1em**, то размер шрифта последнего составит 18 пикселей.

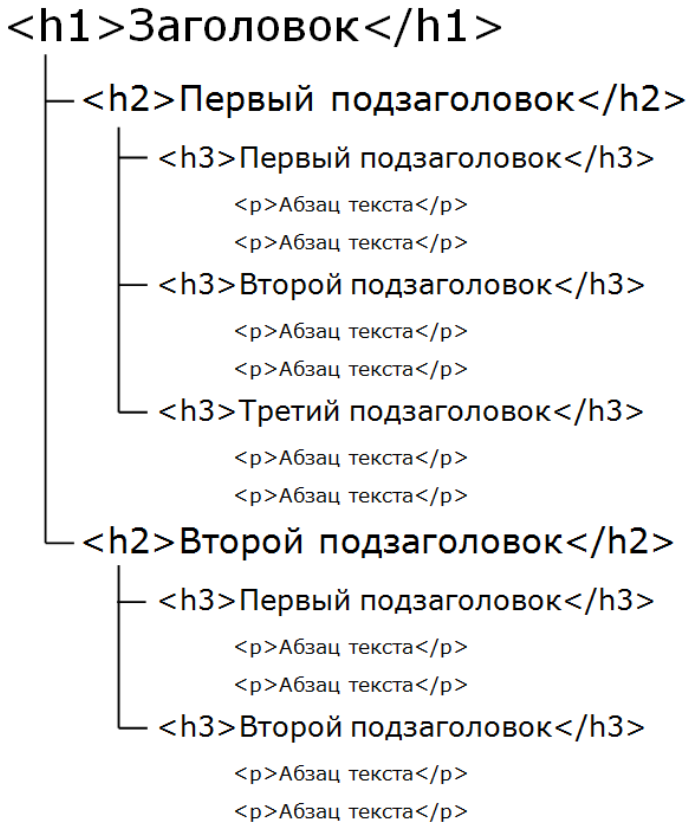

Рис. 22. Структурирование содержимого страницы с использованием тегов h1-h3

2.6.10. В завершение необходимо добавить следующие правила:

```
...
.clear:before,
.clear:after {
    content: "";
    display: table
}
.clear:after {
    clear: both
}
...
```

Вышеуказанные правила будут использованы для отключения свойства обтекания элементов со всех сторон.

Свойство `clear` устанавливает, с какой стороны элемента запрещено его обтекание текстом или другими элементами.

Значение `both` отменяет обтекание элемента одновременно с правого и левого края [15].

Псевдоэлементы `:before` и `:after` использованы для того, чтобы отключить обтекание перед элементом и после.

2.7. Разработка элементов главной страницы

2.7.1. Создание панели навигации сайта

Панель навигации или меню — это область веб-страницы (как правило, верхняя часть), на которой в упорядоченном виде расположены ссылки на разделы и/или страницы сайта. Функция панели навигации — предоставить пользователю удобное средство для перемещения по веб-сайту. С целью повышения удобства использования сайта (англ. usability) не рекомендуется добавлять в меню излишнее количество пунктов, что может наоборот затруднить поиск нужной информации. Рекомендуется указывать ссылки только на основные разделы, а страницы и подразделы, содержащие второстепенную информацию, размещать в виде подпунктов.

2.7.1.1. В самом низу файла стилей нужно добавить следующий комментарий: `/** Панель навигации сайта **/`. Использование таких комментариев облегчает чтение кода, обеспечивает быстрый поиск по файлу стилей, позволяет добавлять пояснения к тому или иному параметру, правилу или блоку правил. Комментарии не интерпретируются браузером. Комментарии могут быть расположены в несколько строк, например:

...
```
/**
    Панель
    навигации
    сайта
              **/
```
...

Комментарий начинается с `/*` и заканчивается на `*/`.

2.7.1.2. На одну строку ниже комментария добавьте следующее правило:

...
```
#menu {
     position: absolute;
     top: 0;
     z-index: 999;
     display: table;
     width: 100%;
     margin-top: 0.8em
}
```
...

Данное правило будет использовано для оформления основного элемента панели навигации. Все последующие правила будут определять визуальное отображение элементов, являющихся дочерними по отношению к элементу с идентификатором `#menu`.

Параметр `position` определяет способ позиционирования элемента, `absolute` - значение, при котором позиция элемента на странице определяется свойствами `left`, `top`, `right` и `bottom`.

Свойство `top` определяет расстояние от верхнего края родительского элемента (если в свойстве `position` установлено значение `absolute`, то окна браузера) до верхнего края дочернего элемента [15]. По аналогичному принципу применяются свойства `left` (левый край), `right` (правый край) и `bottom` (нижний край).

Параметр `z-index` определяет очередность отображения элементов на странице. Например, элемент с параметром `z-index: 2` будет отображаться поверх элемента с параметром `z-index: 1`.

Свойство `display` со значением `table` обозначает отображение элемента в виде таблицы. В приведенном примере оно использовано для того, чтобы впоследствии выравнивать содержимое блока по центру в вертикальной плоскости. Параметр `width` со значением `100%` задает максимальную ширину относительно родительского элемента.

Свойство `margin-top` (равно как и свойства `margin-left`, `margin-right` и `margin-bottom`) используется для определения конкретных значений свойства `margin`. Например, свойства `margin-top: 0.8em` и `margin: 0.8em 0 0 0` определяют величину отступа от верхнего края внешних границ элемента.

2.7.1.3. Далее добавляется следующее правило:

```
...
#menu .menu-inner {
    padding: 0 2.22em;
    display: table-cell;
    vertical-align: middle
}
...
```

Свойства данного правила будут применены к `.menu-inner` только при условии, что элемент с этим классом расположен внутри элемента с идентификатором `#menu`, т. е. следующим образом:

```
...
<div id="menu">
    <div class="menu-inner"></div>
</div>
...
```

Значение параметра `padding: 0 2.22em` определяет отступ в 40 пикселей с правого и левого краев панели навигации.

Значение `table-cell` свойства `display` в совокупности с параметром `vertical-align: middle` позволяют выровнять все дочерние элементы класса `.menu-inner` по центру вертикально при условии, что родительский элемент (`#menu`) имеет свойство `display: table` (см. пункт 2.6.1.2).

2.7.1.4. Из директории **Изображения** копируется файл **flat.png** в каталог **img**, расположенный в директории сайта. Затем добавляется следующее правило:

```
...
#menu .logo {
    background: url('../img/flat.png') no-repeat center
center;
    background-size: 1.38em 1.94em;
    position: absolute;
    width: 1.66em;
    height: 2em;
    top: -0.2em
}
...
```

С помощью данного правила будет добавлен логотип сайта с левого края меню. Параметр `background` определяет фон элемента, в данном случае логотипа **flat.png**.

Значение `no-repeat` устанавливает одно фоновое изображение элемента без его повторений [15].

Значение `center` позиционирует фоновое изображение по центру.

Параметр `background-size` определяет размер изображения.

Благодаря свойству `position: absolute` позиционирование объекта на странице определяется свойствами `top`, `bottom`, `right` и `left` и не зависит от параметров родительского элемента, влияющих на положение объекта. Но поскольку данный элемент будет размещен вне родительского элемента, то помимо размера фонового изображения нужно установить геометрические характеристики посредством параметров `width` и `height`.

2.7.1.5. Ниже добавляются следующие правила:

```
...
#menu .menu-inner nav.main-nav {
    float: right
}
#menu .menu-inner nav.main-nav ul li {
```

```
    display: inline-block;
    margin-right: 1em;
    text-align: center
}
#menu .menu-inner nav.main-nav ul li:last-child {
    margin-right: 0
}
```

...

Вышеуказанные правила определяют отображение списка с пунктами меню и контейнера `nav.main-nav`, внутри которого будет располагаться этот список.

Параметр `float` определяет, по какой стороне будет выравниваться элемент, при этом остальные элементы будут обтекать его с других сторон.

Значение `right` выравнивает элемент (в данном случае маркированный список) по правому краю [15].

Значение `inline-block` в данном случае используется для того, чтобы выровнять элементы маркированного списка по горизонтали.

Свойство `text-align` определяет положение текста в пределах родительского элемента.

Значения могут быть следующие:

- `center` - по центру;
- `left` - по левому краю;
- `right` - по правому краю;
- `justify` - по ширине страницы.

Свойство `margin-right` со значением `1em` устанавливает интервал между пунктами маркированного списка.

Селектор, имеющий окончанием псевдокласс `:last-child`, определяет параметры последнего по счету элемента. В указанном примере он обнуляет значение параметра `margin-right` применительно к элементу, находящемуся в конце маркированного списка.

2.7.1.6. Ниже добавляются следующие правила:

...

```
#menu .menu-inner nav.main-nav ul li a {
    display: inline-block;
    font-size: 0.93em;
    color: #FFF;
    text-decoration: none
}
#menu .menu-inner nav.main-nav ul li a:hover {
```

```
    color: rgb(231, 101, 101)
}
```

...

Записанные правила определяют визуальное отображение пунктов меню. При наведении курсором мыши цвет шрифта изменяется с белого на красный.

2.7.1.7. В директории сайта следует открыть файл **menu.php** и добавить следующий фрагмент кода:

...

```
<div id="menu">
  <div class="menu-inner">
    <a href="/flatiron" class="logo"></a>
    <nav class="main-nav">
      <ul>
        <li><a href="/flatiron/uslugi">Услуги</a></li>
        <li><a href="/flatiron/obekty">Объекты</a></li>
        <li><a href="/flatiron/dokumenty">Документы</a></li>
        <li><a href="/flatiron/portfolio.pdf">Портфолио
работ</a></li>
        <li><a href="/flatiron/o-kompanii">О компании</a></li>
        <li><a href="/flatiron/kontakty">Контакты</a></li>
      </ul>
    </nav>
  </div>
</div>
```

...

Тег `<div>` (англ. division — раздел) используется для создания блочных элементов и установления связи между элементом и внешним файлом стилей с помощью атрибутов `id` и `class`. Например, внешний вид элемента `<div id="menu">` определяется свойствами идентификатора `#menu` в файле стилей. То же самое относится к элементам с атрибутом `class`, за исключением того, что название классов в файле стилей начинается с точки, а не с «решетки» (например, `.menu-inner`).

Тег `<nav>` является одним из нововведений HTML5 и используется поисковыми роботами для определения местоположения панели навигации в исходном коде HTML-страницы. С помощью тега `<ul>` создается маркированный список, а каждый элемент этого списка обозначается тегом `<li>`.

2.7.1.8. С конечным результатом выполненной работы можно будет ознакомиться только после создания главной страницы (см. п. 2.7.4).

2.7.2. Создание блока footer

Блок footer (англ. footer – нижний колонтитул) располагается в нижней части сайта и используется для размещения второстепенной информации, например:

- информация о разработчике сайта;
- контактные телефоны, почтовые адреса ;
- карта сайта;
- ссылки на страницы в социальных сетях;
- упрощенная панель навигации.

В качестве примера будет описан алгоритм создания простого с точки зрения дизайнерского оформления блока footer, но с перспективой дальнейшего совершенствования, по мере развития у пользователя навыков работы с CSS.

2.7.2.1. В самом низу файла стилей рекомендуется записать следующий комментарий: `/** Блок footer **/`.

2.7.2.2. Ниже добавляются следующие правила:

```
...
footer {
    background-color: #f2f2f2
}
.footer-wrap {
    padding: 0.9em 3em
}
...
```

Эти правила определяют цвет фона блока footer и отступы текста от внутренних границ блока.

2.7.2.3. Добавляются следующие правила, определяющие внешний вид блока с информацией об интеллектуальных правах (копирайте):

```
...
.footer-copyright {
    width: 70%;
    float: right;
    margin-top: 0.4em
}
.footer-copyright p {
    text-align: right;
    font-size: 0.9em
}
...
```

Содержимое блока `footer` будет разделено на две части. Ширина первой части, содержащая информацию о копирайте, будет составлять 70 %, а ширина второй части, содержащая список значков социальных сетей — 30 %. Чтобы содержимое второй части не переносилось на следующую строку, использовано свойство `float` со значением `right`, устанавливающее обтекание элемента с левой стороны.

2.7.2.4. В завершении нужно добавить следующие три правила:

...

```
.footer-social-icons {
    width: 30%;
    float: left;
    height: 2.3em
}
.footer-social-icons a {
    display: inline-block
}
a svg:hover #facebook,
a svg:hover #vk,
a svg:hover #twitter,
a svg:hover #instagram {
    fill: rgb(231, 101, 101)
}
```

...

Правило `.footer-social-icons` определяет ширину, высоту и выравнивание контейнера для значков социальных сетей по левому краю с обтеканием других элементов с правой стороны.

Правило `.footer-social-icons a` упорядочивает расположение значков социальных сетей по горизонтали.

Последнее правило определяет цвет значков социальных сетей при наведении на них курсора мыши. Данное правило применяется только к идентификационным именам *.*svg* значков, указанных в названии правила.

2.7.2.5. В директории сайта нужно открыть файл **footer.php** и добавить следующий код:

...

```
<footer>
  <div class="footer-wrap clear">
    <div class="footer-copyright">
      <p>ООО Строительная компания «Флатирон» &copy; 2009-2017</p>
    </div>
```

...

Тег `<footer>` является одним из нововведений HTML5 и используется для того, чтобы поисковые роботы имели возможность определить расположение блока footer в исходном коде HTML-страницы.

2.7.2.6. Ниже добавляется следующий код (в коде файл **svg.txt** расположен в директории **Код**):

...

```
    <div class="footer-social-icons">
      <a href="https://www.facebook.com/groups/zaoflatiron/" tar-
get="_blank">
        Вставьте код SVG значка Facebook из файла svg.txt
      </a>
      <a href="#">
        Вставьте код SVG значка Twitter из файла svg.txt
      </a>
      <a href="#">
        Вставьте код SVG значка Instagram из файла svg.txt
      </a>
      <a href="#">
        Вставьте код SVG значка VK из файла svg.txt
      </a>
    </div>
  </div>
</footer>
```

...

Атрибут `target="_blank"` означает, что ссылка откроется в новой вкладке браузера.

В качестве формата значков социальных сетей рекомендуется выбрать формат SVG по следующим причинам:

- файлы с форматом *.*svg* можно просматривать и редактировать при помощи текстовых редакторов, так как код написан на языке XML;
- возможность масштабирования изображения без потери качества, независимо от размера файла;
- возможность использования анимации;
- визуальным отображением SVG-элементов можно управлять при помощи CSS-правил;
- простая интеграция с HTML и XHTML-документами.

2.7.2.7. С конечным результатом выполненной работы можно будет ознакомиться только после создания главной страницы (см. п. 2.7.4).

2.7.3. Добавление полноэкранной картинки с надписью

2.7.3.1. В самом низу файла стилей рекомендуется записать следующий комментарий: `/** Полноэкранная картинка с надписью **/`.

2.7.3.2. Ниже добавляются следующие правила:

```
...
.fullscreen-image {
    height: 100%;
    width: 100%;
    display: table
}
.fullscreen-image:before {
    content: ' ';
    position: absolute;
    top: 0;
    right: 0;
    left: 0;
    bottom: 0;
    background: rgba(0, 0, 0, 0.2)
}
...
```

Правило `.fullscreen-image` устанавливает максимальную ширину и высоту элемента, который будет содержать фоновое изображение.

Свойства правила `.fullscreen-image:before` определяют затемнение для фонового изображения, которое добавляется на страницу посредством псевдоэлемента `:before`.

2.7.3.3. Затем записываются следующие правила:

```
...
.image-caption {
    line-height: 1.5 !important;
    position: relative;
    text-align: center;
    display: table-cell;
    vertical-align: middle
}
.image-caption h1 {
    color: #FFF;
    font-size: 2.5em
}
...
```

Правило `.image-caption` определяет межстрочный интервал и положение на странице блока с надписью.

Правило `.image-caption h1` определяет цвет шрифта и размер заголовка первого уровня для надписи.

2.7.3.4. В завершении добавляется следующее правило:

...

```
.cover {
    background-size: cover;
    background-position: center center;
    background-repeat: no-repeat
}
```

...

Свойство `background-size: cover` позволяет масштабировать изображение с сохранением пропорций таким образом, что его ширина и высота соответствуют геометрическим характеристикам блока, внутри которого размещено изображение.

Свойство `background-position: center center` позиционирует изображение по центру, а свойство `background-repeat: no-repeat` отключает повторение изображения в том случае, когда размер блока превышает размер картинки.

2.7.3.5. С конечным результатом выполненной работы можно будет ознакомиться только после создания главной страницы (см. п. 2.7.4).

2.7.4. Размещение элементов главной страницы

Когда HTTP-клиент (как правило, браузер) запрашивает путь к директории сайта, то веб-сервер выдает ссылку на главную страницу, которую часто называют **main** или **index**. Распространенным названием для этой страницы является **index.html**, но большинство современных HTTP-серверов (например, Apache, используемый в настоящей работе) имеют конфигурируемый список с названиями файлов, который сервер может использовать в качестве главной страницы [2].

2.7.4.1. Открывается файл **index.php** и добавляется следующий код:

...

```
<?php
$page_title = 'Строительная компания «Флатирон» – выполнение кро-
вельных и гидроизоляционных работ';
$page_desc = 'Строительная компания Флатирон выполняет кровельные и
гидроизоляционные работы. Ремонт кровель и подземных паркингов';
```

```
$inline_styles = '<style>html, body { height: 100% }</style>';
include("header.php");
?>
```

...

Значение переменной `$page_title` определяет название страницы в теге `<title>`, а переменная `$page_desc` - описание страницы в теге `<meta name="description">` (см. п. 2.4).

Значение переменной `$inline_styles` определяет стили, встраиваемые в HTML-структуру страницы.

Конструкция `include` предназначена для включения файлов в код во время исполнения сценария PHP [22].

В данном случае, подключается файл **header.php**.

В п.п. 2.7.3.2 были указаны максимальные значения высоты и ширины элемента, внутри которого будет расположена полноэкранная картинка. Поскольку данная величина была задана относительно родительских элементов, которыми являются `<html>` и `<body>` , то необходимо задать максимальную высоту этих элементов, эквивалентной высоте окна браузера. В приведенном примере это реализовано с помощью внутренних стилей.

2.7.4.2. Копируется файл **home_background.jpg** из директории **Изображения** в каталог **img**, расположенный в директории сайта.

2.7.4.3. Затем необходимо подключить файлы **menu.php** и **footer.php**, а также добавить HTML-код полноэкранной картинки с надписью. Для этого записывается нижеуказанный код:

...

```
<body>
  <?php include("menu.php"); ?>
  <div class="fullscreen-image cover" style="background-
image: url(img/home_background.jpg)">
    <div class="image-caption">
      <h1><span style="opacity: 0.50">Строительная компа-
ния</span> Флатирон</h1>
    </div>
  </div>
  <?php include("footer.php"); ?>
</body>
```

...

В приведенном фрагменте кода использован один из способов добавления стилей в HTML — посредством встроенного стиля (атрибут `style`).

Тег `<span>` в рассматриваемом примере используется для применения CSS свойства к фрагменту текста.

Параметр `opacity` определяет процент прозрачности элемента. В данном случае - 50%.

2.7.4.4. Для просмотра конечного результата перейдите по ссылке http://localhost/flatiron в браузере.

2.7.5. Создание блока с данными о владельце сайта

При первом посещении сайта на пользователя, как правило, производит впечатление и, соответственно, мотивирует к более углубленному изучению содержимого сайта та информация, которая размещена на главной странице. Если пользователь сразу не находит того, что ему нужно, он покидает сайт. Поэтому целесообразнее в качестве первого блока с информацией, следующего после полноэкранной картинки, добавить блок с краткой информацией о владельце сайта. Таким образом, посетитель сразу же получает представление об авторе и общей характеристике содержимого сайта.

2.7.5.1. В самом низу файла стилей рекомендуется добавить следующий комментарий: `/** Блок с данными о владельце сайта **/`.

2.7.5.2. Ниже записывается следующее правило:

```
...
.about-us-content {
    width: 80%;
    margin: 0 auto;
    padding: 6em 0
}
...
```

Данное правило будет применено к контейнеру для текстового содержимого. Ширина контейнера будет составлять 80 % от ширины страницы (свойство `width`), выравнивание будет задано по центру страницы (свойство `margin`), отступы будут установлены от внутренних границ верхнего и нижнего края.

Значение свойства `padding` указано в сокращенном варианте; полный и сокращенный варианты проиллюстрированы на рис. 20. Размер их будет рассчитан путем умножения значения свойства `font-size` элемента `<body>` на число 6.

Например:

```
...
body {
    font-size: 18px
}
.about-us-content {
    padding: 6em 0
}
...
```

18 * 6 = 108. Следовательно, размер отступов будет составлять 108 пикселей.

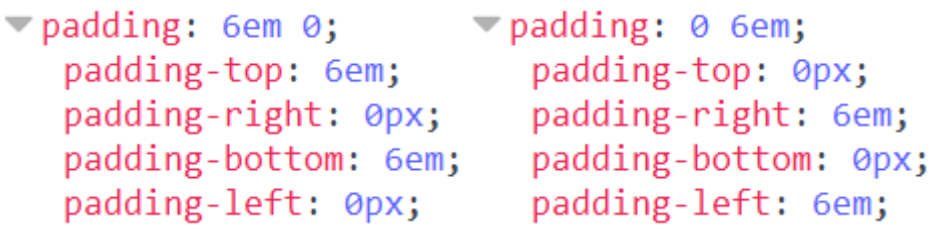

Рис. 23. Полный и сокращенный варианты написания значения свойства padding

2.7.5.3. Затем добавляется следующее правило:

```
...
.about-us-content p {
    text-align: center;
    font-size: 1.7em
}
...
```

Данное правило задает выравнивание содержимого тега `<p>` по центру внутри контейнера `.about-us-content`, а также устанавливает размер шрифта.

2.7.5.4. В файле **index.php** над директивой, подключающей блок footer (`<?php include("footer.php"); ?>`) следует разместить следующий HTML-код:

```
...
  <div class="about-us-container clear">
    <div class="about-us-content">
      <p>Строительная компания «Флатирон» имеет большой опыт работы на рынке строительных услуг. Мы предоставляем квалифицированную помощь и поддержку в воплощении наиболее смелых идей: благоустройство кровель и декоративное озеленение выполняется в полном соответствии с пожеланиями за-
```

```
казчиков.</p>
    </div>
  </div>
```

...

Дополнительный элемент с классом `.clear` добавлен с целью отключения обтекания блока перед и после отступов, установленных в свойствах класса `.about-us-content`.

2.7.5.5. Сохранить внесенные изменения в файлах **index.php** и **style.css**. Для просмотра конечного результата перейти по ссылке http://localhost/flatiron в браузере.

2.7.6. Создание группы блоков с разными визуальными параметрами

Далее будет описан алгоритм разработки группы блоков, расположенных в две колонки с разными параметрами высоты и фоновым изображением. При наведении курсора «мышки» на блок происходит затемнение фона, затем появляется контент, сопровождаемый анимацией.

2.7.6.1. В самом низу файла стилей рекомендуется добавить следующий комментарий: `/** Группа блоков с разными визуальными параметрами **/`.

2.7.6.2. Ниже записать следующие правила:

...

```
.section-header {
    background: #f2f2f2;
    padding: 3em 0
}
.section-header h2 {
    text-align: center;
    font-size: 2.2em
}
```

...

Вышеуказанные правила будут использованы для оформления заголовка разрабатываемого раздела. Фон блока, содержащего в себе текстовый контент, будет светло-серого цвета (свойство `background`). Одним из способов изменения цвета элемента является использование функции выбора цвета из цветовой палитры, реализованной в панели инструментов разработчика. Например, изменить цвет шрифта класса `body` (нижеприведенный алгоритм будет также подходить для изме-

нения цвета любого другого элемента) можно следующим образом:

1) в браузере нажать комбинацию клавиш CTRL+Shift+C;

2) в исходном HTML-коде, отображенном в левой части панели, совершить один клик левой кнопкой мыши по элементу `<body>`;

3) в правой части, в CSS свойствах `body`, слева от значения свойства `color` нажать на пиктограмму круглой формы. Цвет пиктограммы зависит от Hex (hexadecimal — шестнадцатеричный) кода. В приведенном примере пиктограмма имеет темно-серый цвет: ;

4) откроется цветовая палитра (рис. 24). С помощью курсора «мышки» нужно выбирать цвет;

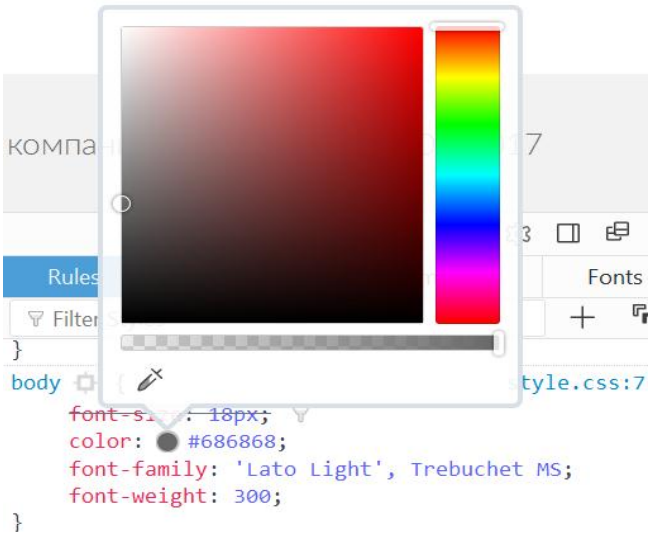

Рис. 24. Цветовая палитра в панели инструментов разработчика

5) для закрытия палитры совершить один клик левой кнопки «мышки» вне цветовой палитры. Hex код выбранного цвета в значении свойства `color` будет изменен. Полученный код можно скопировать с помощью комбинации клавиш CTRL+C и использовать в надлежащих целях.

6) помимо выбора цвета из цветовой палитры также существует функция выбора цвета из любого пикселя в рамках открытой веб-страницы. Для этого нужно открыть цветовую палитру и нажать на пиктограмму инструмента “пипетка” (англ. eyedropper tool) — . На рис. 25 проиллюстрировано использование данного инструмента;

7) для выбора цвета нужно навести инструмент на соответствующее место на странице и совершить один клик левой кнопкой «мышки».

Рис. 25. Выбор цвета с помощью инструмента “пипетка”

2.7.6.3. Затем добавить следующие правила:

```
...
.featured {
     width: 100%;
     overflow: hidden
}
.featured-item {
     width: 50%;
     float: left;
     text-align: center;
     z-index: 0;
     position: relative;
     display: table
}
...
```

Класс `.featured` будет применен к контейнеру для группы разрабатываемых блоков. В данном случае будет отключена прокрутка внутри контейнера (свойство `overflow`). Это необходимо по той причине, что анимация, которая впоследствии будет использована, предполагает увеличение контента при наведении на блок курсора «мышки», вследствие чего появляется полоса прокрутки в нижней части контейнера.

Свойства класса `.featured-item` определяют следующее:

- ширину блока (свойство `width` со значением `50%`);
- выравнивание блока по левому краю и обтекание других элементов (свойство `float` со значением `left`);
- выравнивание содержимого блока по центру (свойство `text-align` со значением `center`);
- очередность отображения (свойство `z-index` со значением `0`), в приведенном примере равна нулю для того чтобы контент, появляющийся при наведении «мышки», не был перекрыт содержимым блока, в рамках которого он появляется;

- относительное позиционирование элемента (свойство `position` со значением `relative`), в приведенном примере используется для отображения псевдоэлемента (затемнение фона) при наведении;

- тип элемента — таблица, чтобы впоследствии выровнять содержимое блока по центру вертикально (свойство `display` со значением `table`).

2.7.6.4. Ниже добавляются следующие правила:

```
...
.featured-item:nth-child(odd){
    clear: left
}
.featured-item:nth-child(even){
    float: none
}
.featured-item.h1 {
    height: 37em
}
.featured-item.h2 {
    height: 30em
}
.featured-item.w1 {
    width: 100% !important
}
...
```

Псевдокласс `:nth-child` используется для добавления стиля к элементам на основе нумерации в дереве элементов [15]. В приведенном примере для нечетных элементов (значение `odd`) будет применено обтекание с левого края элемента. При этом все другие элементы на этой стороне будут опущены вниз, и располагаться под текущим элементом [15].

Для четных элементов (значение `even`) будет отключено выравнивание по левому краю. Три последних правила определяют геометрические параметры элемента.

2.7.6.5. Затем нужно добавить следующее правило:

```
...
.featured-item-content {
    text-align: center;
    line-height: 1.5;
    position: relative;
    color: rgb(246, 246, 246);
    padding: 0 1.11em;
```

```
    z-index: 2;
    display: none
}
```

...

Указанные свойства определяют: выравнивание текста, межстрочный интервал, цвет шрифта, отступы от внутренних границ и очередность отображения. Свойство `display` со значением `none` отключает отображение элемента. В данном случае используется по той причине, что способ отображения элемента будет определен в последующих правилах.

2.7.6.6. Добавляются следующие правила:

...

```
.featured-item:hover:before {
    content: '';
    position: absolute;
    top: 0;
    right: 0;
    left: 0;
    bottom: 0;
    background: rgb(180, 180, 180);
    background: rgba(0, 0, 0, 0.5)
}
.featured-item:hover .featured-item-content {
    display: table-cell;
    vertical-align: middle
}
```

...

Посредством псевдоэлемента `:before` будет добавлено затемнение фона, появляющееся при наведении курсора на блок.

Второе правило включает отображение элемента при наведении и выравнивание содержимого вертикально по центру.

2.7.6.7. Затем следует добавить правила, определяющие параметры анимации:

...

```
@keyframes fadeAndScale {
    from {
        opacity: 0;
        filter: alpha(opacity=0);
        -ms-transform: scale(.9,.9);
        -webkit-transform: scale(.9,.9);
        -o-transform: scale(.9,.9);
```

```
            -moz-transform: scale(.9,.9);
            transform: scale(.9,.9)
        }
        to {
            opacity: 1;
            filter: alpha(opacity=100);
            -ms-transform: scale(1,1);
            -webkit-transform: scale(1,1);
            -o-transform: scale(1,1);
            -moz-transform: scale(1,1);
            transform: scale(1,1)
        }
}
.fadeAndScale {
        -webkit-animation-duration: .3s;
        animation-duration: .3s;
        -webkit-animation-name: fadeAndScale;
        animation-name: fadeAndScale;
        -webkit-animation-timing-function: cubic-
bezier(.71,.55,.62,1.57);
        animation-timing-function: cubic-
bezier(.71,.55,.62,1.57)
}
```

...

Правило `@keyframes` определяет параметры анимации на различных этапах ее воспроизведения, указанные в процентах или значениях `from` (эквивалентно 0 %) и `to` (эквивалентно 100 %). Время перехода от 0 % до 100 % определяется с помощью свойства `animation-duration`.

Свойство `opacity` определяет уровень прозрачности элемента. Аналогичную функцию выполняет свойство `filter` со значением `alpha`, используемое для обеспечения совместимости с браузером **Internet Explorer**, начиная с версии 6.

Свойство `transform` определяет тип двух- и трехмерного преобразования объекта. Значение `scale` задает масштабирование элемента. В приведенном примере происходит изменение размера элемента с 90 % (.9,.9) от исходного размера до 110% (1,1).

Приставки -moz (**Firefox**), -o (**Opera**), -webkit (**Chrome, Safari, Android и iOS**), -ms (**Internet Explorer**) используются для обеспечения совместимости с браузерами, в которых не поддерживается стандартное свойство.

Созданную анимацию можно впоследствии применить к любому элементу, предварительно указав в свойстве `animation-name` название анимации, заданное через пробел после правила `@keyframes`. Свойство `animation-timing-function` задает временную функцию воспроизведения анимации. Значение `cubic-bezier` задает движение анимации в соответствии с кривой Безье. В скобках указываются координаты движения по 4 точкам: cubic-bezier(p0[x], p1[y], p2[x], p3[y]).

Начальная точка (p0[x]) имеет координаты [0,0], конечная (p3[y]) — [1,1], при этом функция по оси ординат может превышать эти значения в большую или меньшую сторону [15]. Кривая Безье, с указанием точек p0[x], p1[y], p2[x] и p3[y] наглядно проиллюстрирована на рис. 26.

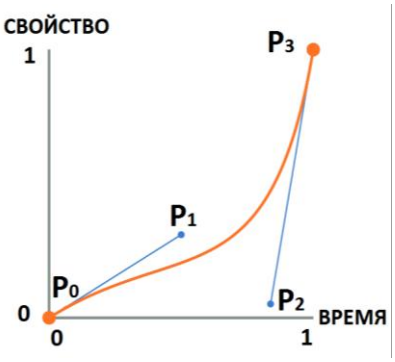

Рис. 26. Кривая Безье на оси ординат

Оранжевая линия обозначает кривую временной функции, а контрольные точки и линии синего цвета отвечают за формирование кривой. Ось y показывает изменение параметра скорости, а ось x — продолжительность процесса.

2.7.6.8. Ниже добавляются правила, определяющие визуальное оформление кнопок:

...

```
.button {
    text-align: center;
    text-decoration: none;
    display: inline-block;
    color: #FFF;
    font-size: 0.9em;
    margin: 0.88em 0;
    line-height: 1.3 !important;
    padding: 0.33em 1em 0.38em;
```

```
    -webkit-transition-duration: 0.4s;
    -o-transition-duration: 0.4s;
    -moz-transition-duration: 0.4s;
    transition-duration: 0.4s;
    cursor: pointer
}
.button1 {
    background-color: transparent;
    border: 0.11em solid rgb(240, 240, 240)
}
.button1:hover {
    background-color: rgb(125, 37, 37);
    background-color: rgba(221, 73, 73, 0.5);
    color: #F9F9F9;
    border: 0.11em solid transparent
}
```

...

Правило `.button` является определяющим для всех последующих вариаций кнопок, в свойствах которых будет изменяться только цвет заливки, границ и текста. Свойства этого правила задают:

- выравнивание текстового содержимого по центру (свойство `text-align` со значением `center`);
- отмену все эффектов оформления текста: мигание, подчеркивание, перечеркивание и добавление линии над текстом. Эффект подчеркивания по умолчанию применяется к содержимому тега `<a>`, а поскольку содержимое проектируемой кнопки будет включать в себя данный тег, то для отмены вышеупомянутого эффекта добавлено свойство `text-decoration` со значением `none`;
- отображение кнопки как встроенного элемента (возможность обтекания другими элементами) со свойствами блочного элемента (за исключением разрыва строки; свойство `display` со значением `inline-block`);
- цвет и размер шрифта (свойства `color` и `font-size`);
- отступы от внешних границ (свойство `margin`);
- межстрочный интервал (свойство `line-height`);
- отступы от внутренних границ (свойство `padding`);
- плавное изменение цвета заливки и текста при наведении курсора (свойство `transition-duration`);
- форму курсора (свойство `cursor` со значением `pointer`);

Правило `.button1` задает прозрачный фон и границы белого цвета. В данном случае при наведении курсора фон и границы принимают красный цвет с прозрачностью 50%.

2.7.6.9. В файле **index.php** над директивой, подключающей блок footer (`<?php include("footer.php"); ?>`) нужно разместить следующий HTML код:

```
...
  <div class="section-header">
    <h2>Избранные проекты</h2>
  </div>
...
```

С помощью приведенного фрагмента кода на страницу добавляется блок (`<div>`) с заголовком второго уровня (`<h2>`), описывающего содержимое разрабатываемого раздела.

2.7.6.10. Из директории **Изображения** скопировать в каталог **img** файлы, название которых начинается с “block” (например, block_01.jpg). Эти изображения будут использованы в качестве фона для каждого из блоков.

2.7.6.11. Затем добавляется следующий HTML код (файл **blocks.txt** расположен в каталоге **Код**):

```
...
  <div class="featured clear">
     Вставьте код из файла blocks.txt
  </div>
...
```

Данный элемент является контейнером для группы блоков, свойство обтекания для которых (перед и после) устанавливается с помощью класса `.clear`. Использование множества классов в свойствах одного элемента позволяет применить больше CSS-правил и использовать глобальные правила с целью сокращения кода. Например, если нужно применить анимацию к различным элементам на странице, не нужно прописывать параметры анимации в свойства каждого из этих элементов, а достаточно создать один класс с этими параметрами и добавить в свойства элемента, как это реализовано в приведенном примере.

Совокупность классов `.featured-item`, `.cover` и `h1` определяет визуальное оформление блока, параметры отображения фонового изображения и высоту блока.

Совокупность классов `.featured-item-content` и `.fadeAndScale` определяет оформление текстового содержимого блока и добавляют анимацию.

Название каждого фотоальбома заключено в тег заголовка третьего уровня (`<h3>`). Ниже добавлена кнопка для перехода на страницу соответствующего фотоальбома.

2.7.6.12. Сохранить внесенные изменения в файлах **index.php** и **style.css**. Для просмотра конечного результата перейти по ссылке http://localhost/flatiron в браузере.

2.7.7. Создание блока с информацией для связи с владельцем сайта

Для быстрой связи посетителей с владельцем сайта должен быть создан специальный блок, в котором размещаются соответствующие данные, телефонные номера и почтовые адреса. Организуется это следующим образом.

2.7.7.1. В самом низу файла стилей рекомендуется добавить следующий комментарий: `/** Блок с контактной информацией **/`.

2.7.7.2. Ниже добавляются следующие правила:

...

```
.contact-us-content {
    width: 80%;
    margin: 0 auto;
    padding: 3em 0
}
.contact-us-content h2 {
    font-size: 2.2em
}
.contact-us-content p {
    line-height: 2
}
.contact-us-content p a {
    color: rgb(231, 101, 101)
}
```

...

Первые три правила задают следующие параметры:

- ширину блока;
- выравнивание текста (в данном случае по центру);
- отступы содержимого блока от внутренних границ;
- размер шрифта заголовка второго уровня;
- межстрочный интервал абзацев.

Еще одно правило определяет цвет шрифта содержимого тега `<a>`, находящегося внутри тега `<p>`. Это сделано для того, чтобы данное правило не применялось к содержимому кнопки, которая также будет расположена внутри блока с классом `.contact-us-content`.

2.7.7.3. Ниже добавляется следующее правило:

```
...
.space {
    display: block;
    height: 0;
    clear: both;
    overflow: hidden
}
...
```

Это правило предназначено для задания пробелов фиксированной величины между абзацами.

2.7.7.4. В завершение добавляются следующие правила:

```
...
.button2 {
    background-color: rgb(225, 135, 135);
    border: 0.11em solid transparent
}
.button2:hover {
    background-color: transparent;
    color: rgb(216, 103, 103);
    border: 0.11em solid rgb(227, 151, 151)
}
...
```

Правило `.button2` задает светло-красный фон и толщину границ кнопки. В данном случае при наведении курсора фон становится прозрачным, шрифт и границы принимают светло-красный цвет.

2.7.7.5. В файле **index.php** над директивой, подключающей блок footer (`<?php include("footer.php"); ?>`) размещается следующий HTML код:

```
...
  <div class="contact-us-container clear">
    <div class="contact-us-content">
      <h2>Контакты</h2>
      <div class="space" style="height: 2.22em"></div>
      <p>ООО «СК «Флатирон» находится по адресу:<br>Россия, г.
Екатеринбург, ул. Ткачей 25, Бизнес Центр «Clever Park», офис
304</p>
      <div class="space" style="height: 1.11em"></div>
```

```
      <p>Телефон: <a href="tel:+73432877728">+7 (343) 28-777-
28</a> <br> Мобильный: <a href="tel:+79120458848">+7 (912) 045-
88-48</a></p>
      <div class="space" style="height: 1.11em"></div>
      <a class="button button2" href="/flatiron/kontakty" tar-
get="_blank">Посмотреть на карте</a>
    </div>
  </div>
```

...

Для задания величины интервала между абзацами в атрибут `style` было добавлено свойство `height`. Тег `<br>` используется для разрыва строки с целью переноса следующего за ним контента на строку ниже.

В значении атрибута `href` указывается телефонный номер. При нажатии на ссылку мобильное приложение запрашивает подтверждение вызова по указанному номеру. Если на компьютере установлено приложение, обеспечивающее телефонную связь по протоколу IP (например, Skype), система запросит подтверждение на открытие соответствующего приложения. Телефонный номер рекомендуется указывать в международном формате: знак плюс (+), код страны, код города и номер абонента.

2.7.7.6. Сохранить внесенные изменения в файлах **index.php** и **style.css**. Для просмотра конечного результата перейти по ссылке http://localhost/flatiron в браузере.

2.8. Разработка адаптивного дизайна

За последнее время резко повысилось использование мобильных устройств с целью выхода в Интернет. По некоторым данным не менее 80% пользователей для этих целей используют смартфоны, а больше половины всего Интернет-трафика приходится на мобильные устройства [26]. Отсюда становится очевидной необходимость оптимизации веб-сайтов под мобильные устройства, поэтому в настоящей работе изложены основные принципы создания адаптивного дизайна.

Существует несколько способов оптимизации сайта под мобильные устройства, но основные из них два:

1. Разработка мобильной версии сайта.
2. Разработка адаптивного дизайна.

Рассмотрим некоторые преимущества и недостатки указанных способов:

1. Сайты с адаптированным дизайном могут поддерживать различные устройства и размеры экрана путем добавления медиазапросов в файл стилей, в то время как мобильная версия предполагает разработку отдельного веб-сайта.

2. Сайты с адаптированным дизайном обычно загружаются дольше, чем мобильная версия. Это обусловлено загрузкой всего контента сайта, включая файлы стилей, плагины, картинки в высоком разрешении. Существуют методы повышения производительности сайта, включающие решение обозначенной проблемы, но их реализация требует дополнительных трудозатрат.

3. Так как мобильная версия является отдельным веб-сайтом, то потребуется разработка, а впоследствии и добавление нового контента для двух сайтов, что усложняет задачу администрирования.

4. В списках поисковиков Интернет позиция мобильной версии сайта в результатах поиска ниже, в сравнении с версией ПК. Это обусловлено тем, что URL мобильной версии сайта отличается от версии ПК (например, m.flatiron.su), и изначально не наследует высокую позицию в поисковой системе от родительского сайта.

5. Сайт с адаптивным дизайном имеет одинаковую функциональность и контент для всех разрешений экрана, в то время как мобильная версия в этом плане является значительно сокращенной.

6. Но с другой стороны, процесс разработки адаптивного дизайна требует больше трудозатрат, чем разработка мобильной версии и должен быть выше профессиональный уровень разработчика.

В целом, можно заключить, что оба рассмотренных способа используются на практике для оптимизации сайтов под мобильные устройства. При выборе разработчик может руководствоваться следующим:

- если мобильную версию сайта нужно разработать в короткие сроки, высокой производительности, но ограниченной функциональностью, то предпочтителен первый способ;
- если сохранение функциональных возможностей и контента сайта является первостепенным, а также необходимо сохранить высокую позицию в списках поисковых систем, то следует выбрать второй способ.

2.8.1. Установка и настройка плагина FlowType.JS

При создании адаптивного веб-дизайна разработчику необходимо определить набор CSS правил для элементов веб-сайта, с целью

оптимизации их под различный диапазон размеров экрана (медиа-запрос). Учитывая, как правило, большое количество элементов и медиа-запросов, а также время, затрачиваемое на тестирование, объем работ представляется большим. Для снижения трудоемкости можно использовать предназначенный для этой цели плагин FlowType.

Плагин состоит из формулы, написанной на JavaScript, по которой рассчитывается размер шрифта путем деления значения ширины элемента на коэффициент, указанный в свойстве `fontRatio`. Соответственно, при увеличении/уменьшении размера элемента изменяется размер шрифта. Если применить указанные параметры к элементу `<body>`, то размер шрифта будет изменяться во всех элементах на странице, имеющих свойство `font-size`.

Таким образом, плагин выполняет функцию адаптации размера шрифта под любое разрешение экрана и существенно сокращает время, затрачиваемое на оптимизацию сайта под размеры экрана, регламентированные в техническом задании. Здесь следует отметить, что помимо шрифта можно также адаптировать размер элементов и величину отступов, предварительно указав размеры в единице измерения em.

2.8.1.1. Прежде всего нужно скопировать файл библиотеки jQuery — **jquery.js**, необходимый для функционирования плагина, из директории **Скрипты** в каталог **js**, расположенные в директории сайта.

2.8.1.2. Далее, необходимо скопировать файл плагина FlowType — **flowtype.js** в каталог **js**.

2.8.1.3. В самом низу кода файла **flowtype.js** нужно добавить следующую строку:

```
...
$("body").flowtype({minFont:13,maxFont:999,fontRatio:65});
...
```

`$` (альтернативным вариантом записи будет оператор — `jQuery`) в данном случае означает, что указанная операция выполняется с использованием библиотеки jQuery.

Оператор `"body"` указывает, что последующие действия будут применяться к тегу `<body>`.

Также оператор `.flowtype` обозначает вызов функции `$.fn.flowtype` (2-я строка файла **flowtype.js**), и замену значений стандартных параметров (4-8 строки) теми, что указаны в скобках.

Свойство `maxFont` задает максимальное значение размера шрифта. При его исполнении значение ширины элемента делится на величину, указанную в свойстве `fontRatio`. Например, свойство `font-ratio` имеет значение `65`, а ширина элемента составляет 1200

пикселей. В этом случае расчитывается: 1200/65 = 18,46. Таким образом, размер шрифта элемента составит 18,46 пикселей.

Если ширина элемента `<body>` составляет 320 пикселей (как у некоторых мобильных устройств), то при делении на коэффициент 65, получится около 5 пикселей и, соответственно, шрифт такого размера будет не читабелен. Для устранения этой проблемы необходимо задать минимальное значение размера шрифта с помощью параметра `minFont`.

2.8.1.4. Далее, в конце текста файла **footer.php** записываются следующие строки кода:

```
...
<script src="js/jquery.js" defer="defer"></script>
<script src="js/flowtype.js" defer="defer"></script>
...
```

Здесь тег `<script>` предназначен для загрузки внешних скриптов, либо фрагментов кода функциональной части программы, написанных на языке JavaScript.

Атрибут `src` указывает путь к файлу скрипта.

Атрибут `defer` задает выполнение скрипта после полной загрузки страницы, а также определяет последовательность выполнения скриптов. Например, если скрипт **flowtype.js** загрузился раньше, чем **jquery.js**, то выполнение **flowtype.js** откладывается до загрузки и исполнения скрипта **jquery.js**.

Целесообразность использования этого атрибута обусловлена тем, что когда браузер в процессе загрузки страницы доходит до оператора обработки скрипта, то загрузка прочих элементов приостанавливается до завершения загрузки скрипта, и в этот промежуток времени пользователь лишен возможности просмотреть страницу целиком.

Атрибут `defer` устраняет эту проблему, путем загрузки скриптов одновременно с другими элементами страницы и выполнением их после полной загрузки этой страницы.

2.8.1.5. Далее нужно сохранить внесенные изменения в файлах **footer.php** и **flowtype.js**.

Чтобы удостовериться в правильности установки и настройки плагина, в браузере перейти по следующей ссылке: http://localhost/flatiron/. Далее необходимо вызвать панель инструментов разработчика путем нажатия следующей комбинации клавиш: CTRL+Shift+C (комбинация работает в Mozilla Firefox и браузерах на базе Chromium).

В открывшейся панели слева отображается структура HTML кода страницы. Справой части — CSS свойства выбранного элемента.

Если установка и настройка плагина были сделаны правильно, то к элементу `<body>` должен добавиться атрибут `style` со свойством `font-size` и значением, полученным путем деления ширины элемента на коэффициент `fontRatio`. Пример правильного выполнения этих условий изображен на рис. 27.

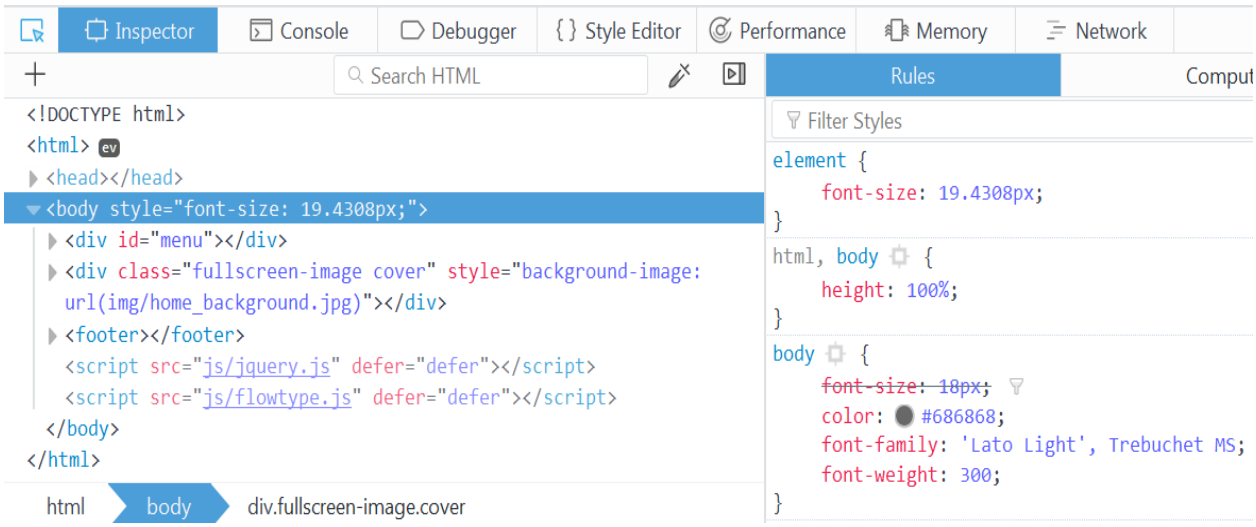

Рис. 27. Панель инструментов разработчика в Mozilla Firefox

Как видно из рис. 27, свойство встроенного стиля имеет приоритет по отношению к свойству внешнего стиля. (Свойство встроенного стиля можно отменить путем добавления оператора `!important` после значения свойства внешнего стиля).

Следует отметить необходимость использования панели инструментов разработчика в процессе разработки сайта, так как это позволяет вносить корректировки в дизайн сайта в режиме реального времени, однако при обновлении страницы все изменения сбрасываются, если не установлен плагин DevTools Autosave (Chrome) [23] или FireFile (Firefox) [24].

2.8.2. Создание мобильного меню

Если меню состоит из большого количества пунктов, то при низких разрешениях экрана оно будет занимать как минимум четверть высоты страницы, что затруднит пользование сайтом и исказит визуальное оформление. Для решения этой проблемы нужно создать

мобильное меню, содержимое которого будет появляться после нажатия соответствующей кнопки. Это позволит сэкономить пространство на странице и обеспечить удобство пользования сайтом.

2.8.2.1. В самом низу файла стилей рекомендуется добавить следующий комментарий: `/** Мобильное меню **/`.

2.8.2.2. Ниже добавляются следующие правила:

```
...
.menu_button {
    display: none;
    float: right;
    text-decoration: none;
    cursor: pointer
}
.menu_button svg:hover #menu_button {
    fill: rgb(231, 101, 101)
}
...
```

Пояснения к свойствам правила `.menu_button` изложены в п.п. 2.8.6.3 и 2.8.6.8.

Второе правило определяет цвет кнопки меню при наведении курсора.

Свойство `fill` в SVG-изображениях определяет цвет заливки элемента. Оператор `#menu_button` в приведенном примере является идентификатором SVG тега `<path>`, внутри которого расположен код блока кнопки.

2.8.2.3. Ниже добавляются следующие правила:

```
...
#overlay-content, #closebtn {
    display: none
}
#closebtn {
    cursor: pointer
}
.overlay {
    height: 0%;
    width: 100%;
    position: fixed;
    z-index: 999;
    top: 0;
    left: 0;
    overflow-y: auto;
```

```
    background-color: rgb(240, 240, 240);
    background-color: rgba(255, 255, 255, 0.9)
}
#closebtn, .overlay a {
    padding: 0.55em;
    text-decoration: none;
    font-size: 1.3em;
    color: #686868;
    display: block
}
#closebtn:hover, .overlay a:hover, .overlay a:focus {
    color: rgb(231, 101, 101)
}
```

...

Правило `.overlay` определяет визуальное оформление полноэкранного меню, а правила `#closebtn, .overlay a` — оформления пунктов и кнопки закрытия меню.

Правила `#overlay-content, #closebtn` отключают отображение меню и кнопки его закрытия с той целью, чтобы меню не открывалось автоматически, а только при нажатии на кнопку меню, что будет реализовано с помощью фрагмента кода на JavaScript.

Псевдокласс `:focus` определяет визуальное оформление элемента, выбранного с помощью клавиатуры (например, посредством клавиши табуляции).

Свойство `position` со значением `fixed` устанавливает положение элемента исходя из значений свойств `left`, `right`, `top` и `bottom` таким образом, что при прокрутке страницы положение элемента остается неизменным. Поскольку меню будет вызываться на каждой странице сайта, где могут быть расположены элементы с высоким значением параметра `z-index`, то для того чтобы визуальная форма меню не перемещалось на задний план, установлено значение `999`.

2.8.2.4. В каталоге **js** нужно создать файл с названием **mobile-menu.js**, а затем открыть его. Ниже будет описан алгоритм создания двух функций, написанных на языке JavaScript, определяющих визуальное отображение меню при открытии и закрытии.

2.8.2.5. Добавить следующий JavaScript код:

...

```
function openNav() {
  document.getElementById("mobile-nav").style.height="100%",
  document.getElementById("overlay-
content").setAttribute("style","display: block; text-align: center;
margin-top: 30px; width: 100%; top: 20%; position: relative;"),
```

```
  document.getElementById("closebtn").setAttribute("style","display:
block; position: absolute; top: 0; right: 5px; font-size: 40px")
}
```

...

Данная функция определяет визуальное отображение при нажатии на кнопку открытия меню. JavaScript функция определяется с помощью ключевого слова `function`, далее следует название функции (в приведенном примере — `openNav`), в круглых скобках задаются параметры функции, в фигурных — блок операторов. Параметры используются для передачи информации функции с целью изменения ее действия или для вычислительных операций.

В качестве примера рассмотрим модификацию функции `typeGroup`:

...

```
function typeGroup(groupName) {
  if (groupName ! = "")
  alert("Студент группы "+groupName);
  else
  alert("Пожалуйста, укажите группу");
}
```

...

Если вызвать функцию с помощью команды `typeGroup("ИАТТС-15");`, то результатом выполнения данной команды будет: *Студент группы ИАТТС-15*. Вызов функции с помощью команд `typeGroup("")` или `typeGroup()` приведет к появлению диалогового окна следующего содержания: *Пожалуйста, укажите группу*.

Команда `document` обеспечивает взаимодействие JavaScript с HTML страницей.

Команда `document.getElementById("mobile-nav").style.height="100%"` задает максимальную высоту элемента с идентификатором `mobile-nav`.

Команда `setAttribute` в отличие от команды `style` позволяет изменить большое количество CSS свойств элемента, что наглядно продемонстрировано в приведенном примере.

2.8.2.6. Далее нужно добавить следующую функцию:

...

```
function closeNav() {
  document.getElementById("mobile-nav").style.height="0%",
  document.getElementById("overlay-content").style.display="none",
  document.getElementById("closebtn").style.display="none"
}
```

...

Данная функция определяет визуальное отображение при нажатии на кнопку закрытия меню. Соответственно, отключается отображение кнопки закрытия и пунктов, а также происходит изменение высоты полноэкранного меню со 100% на 0%, что равносильно отключению отображения элемента.

2.8.2.7. Сохранить внесенные изменения. Далее нужно добавить JavaScript функцию, определяющую загрузку скрипта в зависимости от ширины экрана. Для этого в самом низу текста файла **footer.php** добавляется следующий код:

...

```
<script>
  if (screen && screen.width < 800) {
    document.write('<script src="js/mobile-menu.js" de-
fer="defer"><\/script>');
  }
</script>
```

...

Если ширина экрана меньше 800 пикселей, то в структуру HTML-документа добавляется указанный в круглых скобках код, вызывающий внешний скрипт.

2.8.2.8. В файле **menu.php** после закрывающего тега (`</nav>`) элемента `<nav class="main-nav">` добавить следующий фрагмент кода:

...

```
    <div class="menu_button" onclick="openNav()">
      Вставьте код кнопки меню из файла svg.txt
    </div>
```

...

Данный элемент добавляет на страницу кнопку меню (☰) в формате SVG. С помощью атрибута `onclick`, определяющего событие при нажатии на кнопку, происходит загрузка функции `openNav()`.

2.8.2.9. Ниже добавляется следующий код:

...

```
    <nav id="mobile-nav" class="overlay" style="display: none">
      <span id="closebtn" onclick="closeNav()">×</span>
      <div id="overlay-content">
        <ul>
          <li><a href="/flatiron/uslugi">Услуги</a></li>
          <li><a href="/flatiron/obekty">Объекты</a></li>
          <li><a href="/flatiron/portfolio.pdf">Портфолио
работ</a></li>
          <li><a href="/flatiron/dokumenty">Документы</a></li>
          <li><a href="/flatiron/o-kompanii">О компании</a></li>
          <li><a href="/flatiron/kontakty">Контакты</a></li>
```

```
        </ul>
      </div>
    </nav>
```

...

Данный код добавляет на страницу полноэкранное меню с расположенными вертикально по центру пунктами и кнопкой закрытия (×), расположенной в правом верхнем углу меню. При нажатии на кнопку закрытия происходит загрузка функции `closeNav()`.

2.8.2.10. Просмотреть конечный результат возможно только после создания медиа-запросов.

2.8.3. Создание медиа-запросов

Медиа-запрос представляет собой набор правил, предусмотренных для заданного диапазона разрешений экрана. Каждое правило должно определять визуальное оформление элемента таким образом, чтобы оно корректно отображалось под заданные размеры экрана. Ниже будут указаны медиа-запросы, но прежде рассмотрим пример, доказывающий важность установления порядка при расположении медиа-запросов. Если в двух медиа-запросах указано правило, применяемое к конкретному элементу, то предпочтение будет отдано правилу последнего медиа-запроса, в случае, когда в правиле первого медиа-запроса не указано значение `!important`.

В CSS коде это выглядит следующим образом:

...

```
@media screen and (max-width: 600px) {
     body {
          background: red
     }
}
@media screen and (max-width: 400px) {
     body {
          background: blue
     }
}
```

...

Если ширина окна браузера составляет 350 пикселей, то фон страницы будет синим, так как последний медиа-запрос имеет приоритет по отношению к первому.

Для медиа-запросов, как правило, создается отдельный файл, который подключается путем добавления дополнительной ссылки на

внешний ресурс в теге `<link>`. Однако в рассматриваемом примере размер базы CSS-правил сайта является небольшим, поэтому медиа-запросы будут добавлены в основной файл стилей (style.css).

2.8.3.1. В самом низу файла стилей рекомендуется добавить комментарий следующего содержания: `/** Адаптивный дизайн **/`.

2.8.3.2. Ниже добавляется следующий медиа-запрос:

```
...
@media screen and (max-width: 800px) {
     .featured-item {
          width: 100%
     }
     .featured-item.h1, .featured-item.h2 {
          height: 30em
     }
     .footer-wrap {
          padding: 0.9em 2em
     }
     #menu .logo {
          background-size: 1.6em 2.1em;
          top: 0
     }
     nav#mobile-nav, .menu_button {
          display: block !important
     }
     nav.main-nav {
          display: none
     }
}
...
```

В содержании кода правило `@media` используется для применения CSS свойств к заданному типу устройства.

Тип устройства `screen` означает экраны компьютеров, планшетов, смартфонов и т. п.

Оператор `and` позволяет комбинировать несколько условий. Параметр `max-width: 800px` определяет следующее условие: использование набора правил, входящих в состав медиа-запроса, происходит только в том случае, если ширина экрана не превышает 800 пикселей.

Первое правило устанавливает 100 %-ную ширину каждого блока с классом `.featured-item`, второе правило устанавливает одинаковую высоту для каждого блока. Третье правило изменяет величину отступов от внутренних границ блока footer. Два последних правила

отключают отображение панели навигации и включают отображение мобильного меню.

2.8.3.3. Ниже добавьте следующий медиа-запрос:

...

```
@media screen and (max-width: 600px) {
    .about-us-content p {
        font-size: 1.4em
    }
    .contact-us-content, .about-us-content {
        width: auto;
        padding: 3em 1.66em
    }
    .fadeAndScale {
        -webkit-animation-name: none;
        animation-name: none
    }
    .featured-item:hover:before {
        content: none
    }
    .featured-item-content {
        display: block;
        background: rgba( 0, 0, 0, 0.6 );
        padding: 0.55em 0.83em 0;
        text-align: left;
        width: 100%;
        bottom: 0;
        position: absolute
    }
    .footer-wrap {
        padding: 1em 1.66em
    }
    .footer-copyright, .footer-copyright p, .footer-
social-icons {
        width: auto;
        float: none;
        text-align: left;
        margin-top: 0
    }
    .footer-social-icons {
        margin-top: 1em
    }
    h3 {
```

```
            font-size: 1.2em
        }
        .image-caption, #menu .menu-inner {
            width: auto;
            padding: 0 1.66em
        }
}
```

Данный набор правил будет применяться к экранам шириной не более 600 пикселей.

Первое правило изменяет размер шрифта в блоке с данными о владельце сайта.

Второе правило сбрасывает ширину блока с контактной информацией и блока с данными о владельце сайта и устанавливает отступы от внутренних границ блоков.

Третье и четвертое правила отключают анимацию, а пятое правило определяет визуальное оформление для блоков с классом `.featured-item`.

Шестое, седьмое и восьмое правила определяют визуальное оформление блока footer.

Девятое правило определяет размер шрифта для заголовков третьего уровня.

Последнее правило сбрасывает ширину элемента и устанавливает отступы от внутренних границ с левой и правой стороны.

2.8.3.4. В завершение добавляется следующий медиа-запрос:

```
...
@media screen and (max-width: 400px) {
        .image-caption h1 {
            font-size: 1.8em
        }
}
...
```

Данное правило изменяет размер шрифта надписи, расположенной по центру полноэкранной картинки на экранах с шириной до 400 пикселей.

2.8.3.5. Для просмотра конечного результата нужно перейти по ссылке http://localhost/flatiron/. Далее, нажать комбинацию клавиш CTRL+Shift+M (в браузере Google Chrome нужно предварительно вызвать панель разработчика) для перехода в режим эмуляции мобильных устройств. Такой режим используется для тестирования адаптивного дизайна на различных размерах экрана и имеет интуитивно понятный интерфейс.

Поскольку при смене разрешения экрана в режиме эмуляции не происходит автоматической перезагрузки страницы, то для загрузки скрипта, необходимого для работы мобильного меню, нужно нажать клавишу F5.

Далее нужно нажать кнопку меню (☰). При этом должно открыться полноэкранное меню, как показано на рис. 28.

Далее необходимо протестировать адаптивный дизайн на следующих размерах экрана: 320x480, 600x400, 800x600. Для изменения существующих размеров их значения выделяются курсором и нажимается клавиша “Del” или “Backspace”, затем вводится требуемое значение. Для подтверждения нажать клавишу “Enter”.

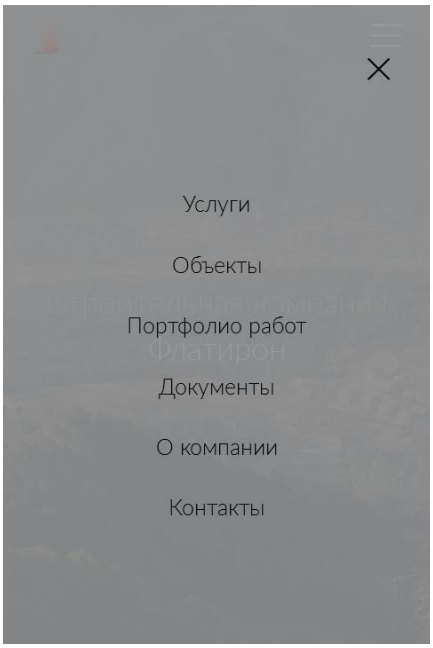

Рис. 28. Визуальная форма мобильного меню

Глава 3

ВВЕДЕНИЕ В СИСТЕМУ УПРАВЛЕНИЯ СОДЕРЖИМЫМ WORDPRESS

Система управления содержимым (англ. CMS — Content Management System) представляет собой веб-интерфейс, в котором реализована возможность редактировать информационное содержимое сайта. Созданный контент в совокупности с настройками CMS и плагинов хранится в базе данных. Также реализована возможность ограничения доступа к тем или иным разделам/страницам сайта для определенных групп пользователей. Здесь следует отметить, что в CMS имеется возможность регистрации пользователей, данные о которых будут также сохраняться в базе данных.

Системы CMS обладают как преимуществами, так и недостатками, а специфические особенности их приведены ниже.

1. Управление сайтом с большим количеством страниц (до нескольких сот или тысяч) невозможно осуществить без CMS. В качестве примера можно привести новостной портал, информация на котором должна регулярно обновляться. Если на сайте не установлена CMS, то для добавления, редактирования информации потребуется использование HTML и CSS, а также работа с ftp-сервером, все это значительно усложняет процесс управления сайтом.

2. В исходном коде CMS имеется большое количество функций, написанных на PHP, которые выполняются при загрузке сайта. Эти функции выполняют задачи разной степени сложности. Найти уязвимость в таком обширном количестве функций гораздо проще, чем, например, на сайте без CMS, в котором использование PHP может сводиться к выполнению простых операций, например, подключения файлов и т.п.

3. Владельцу дается возможность самостоятельно добавлять и редактировать контент на сайте без обращения к профессиональному разработчику.

4. Система управления контентом, особенно при наличии установленных плагинов, увеличивает время загрузки сайта.

Используемая в настоящей работе CMS WordPress была изначально предусмотрена для разработки блогов. Одной из функций данной CMS является хранение любого количества версий отдельно взятой страницы, что позволяет следить за изменениями на странице и при необходимости осуществлять переход к более ранним версиям («откат»). Для групп пользователей с ограниченными правами предоставляется возможность написания статей, но окончательная их публикация зависит от администратора.

В качестве недостатков данной CMS можно указать отсутствие функции создания директорий в библиотеке медиафайлов, что затрудняет поиск нужных файлов, особенно при большом объеме загруженного медиа-контента.

В разделе изложены основные принципы работы с системой управления контентом, включая установку CMS на локальный сервер, редактирование шаблона дизайна, создание и размещение информации, настройку CMS, а также установку и настройку плагинов.

3.1. Установка системы управления содержимым WordPress на локальный сервер

3.1.1. Скачивается дистрибутив WordPress. Список всех версий (включая бета-релизы) доступен по следующей ссылке: http://ru.wordpress.org/releases/. В настоящей работе используется версия 4.7.2.

3.1.2. Скачанный архив необходимо распаковать в директорию C:\xampp\htdocs\. Директория WordPress должна быть следующей: C:\xampp\htdocs\wordpress (далее — директория WordPress).

3.1.3. Далее создается база данных, куда ~~впоследствии~~ будет сохраняться информационное наполнение сайта, а также настройки CMS и плагинов. Создание и управление базами данных осуществляется при помощи веб-приложения **phpMyAdmin** (рис. 29). Для вызова этого приложения перейти по следующей ссылке: http://localhost/phpmyadmin.

После открытия приложения перейти на вкладку “Databases”. В поле “Database name” следует ввести “wordpress”, после чего нажать “Create”. Последовательность действий обозначена цифрами на рис. 30.

3.1.4. Перед установкой CMS необходимо указать данные для подключения базы данных MySQL в конфигурационном файле. Для этого в директории WordPress открывается файл **wp-config-sample.php** и в разделе “Параметры MySQL” изменяются стандартные значения по аналогии с рис. 31.

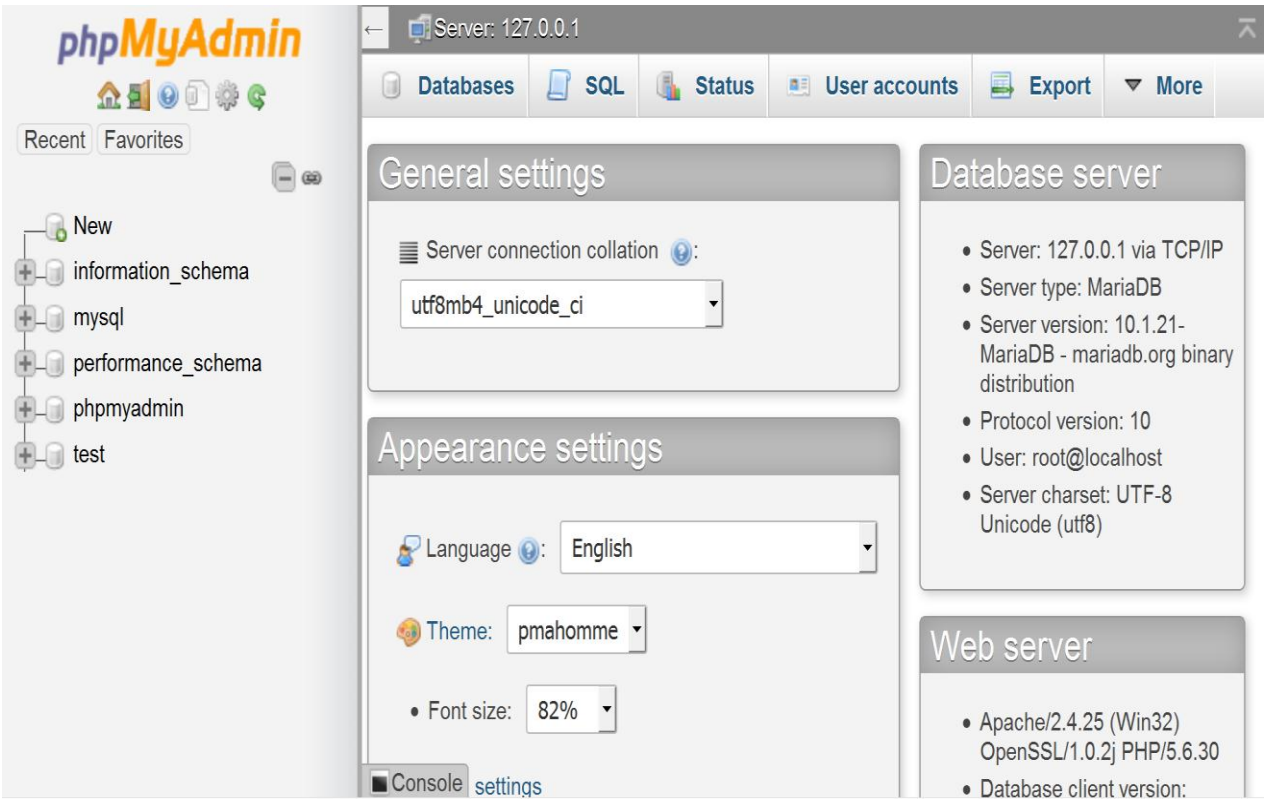

Рис. 29. Интерфейс веб-приложения phpMyAdmin

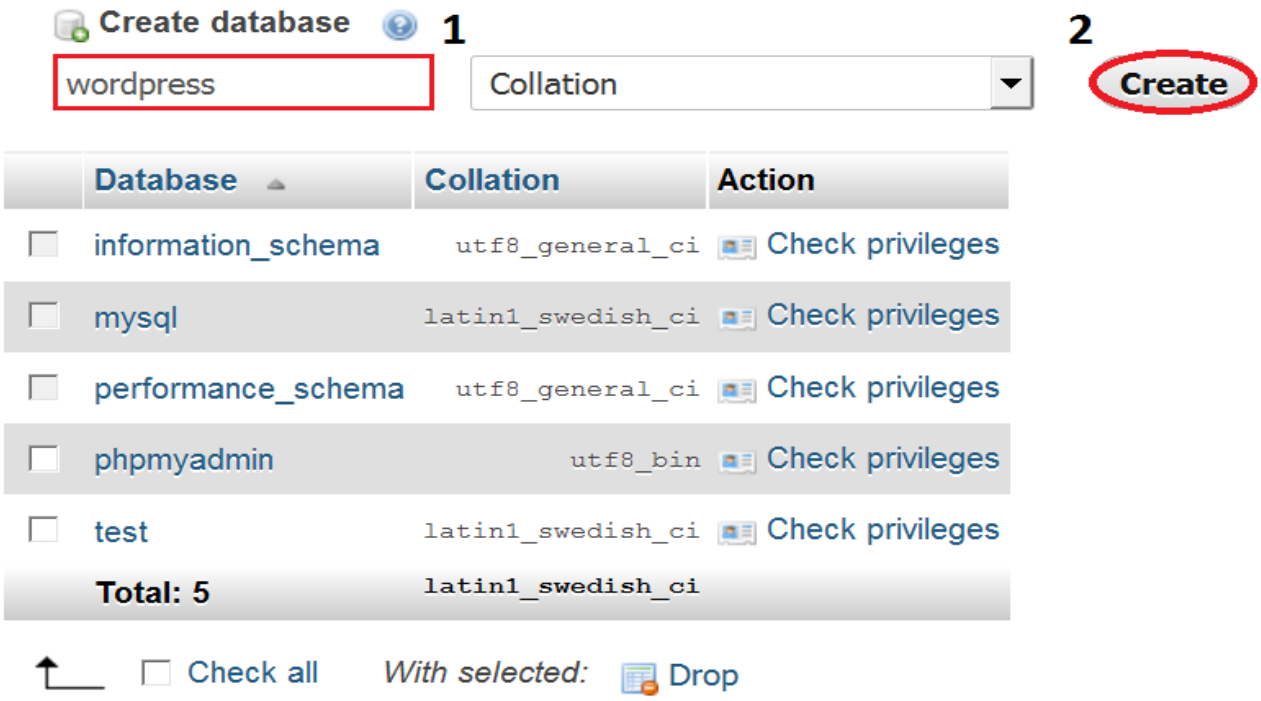

Рис. 30. Создание базы данных в приложении phpMyAdmin

```
22  /** Имя базы данных для WordPress */
23  define('DB_NAME', 'wordpress');
24
25  /** Имя пользователя MySQL */
26  define('DB_USER', 'root');
27
28  /** Пароль к базе данных MySQL */
29  define('DB_PASSWORD', '');
30
31  /** Имя сервера MySQL */
32  define('DB_HOST', 'localhost');
33
34  /** Кодировка базы данных для создания таблиц. */
35  define('DB_CHARSET', 'utf8');
```

Рис. 31. Данные для подключения базы данных MySQL
в конфигурационном файле **wp-config-sample.php**

Далее сохраняется этот файл под названием **wp-config.php** в той же директории, где расположен файл **wp-config-sample.php**. В текстовом редакторе Notepad++ это можно сделать следующим образом: в меню “Файл” (англ. File) выбрать пункт “Сохранить как” (англ. Save as) и в поле “Имя файла” изменить существующее имя на **wp-config.php**. Для сохранения нажать кнопку “Сохранить”.

3.1.5. Перейти по ссылке: http://localhost/wordpress. и запустить мастер установки CMS WordPress. В поля “Название сайта”, “Ваш e-mail” и “Имя пользователя” нужно ввести соответствующую информацию. В целях повышения безопасности сайта не рекомендуется менять автоматически сгенерированный пароль. Для завершения установки нажимается кнопка “Установить WordPress”.

3.1.6. Далее должна произойти переадресация на страницу входа в админ-панель. Если этого не произошло, то в адресной строке браузера следует ввести указанную ниже ссылку и нажать “Enter”: http://localhost/wordpress/wp-login.php. В поля “Имя пользователя” и “Пароль” вводятся данные, указанные в процессе установки.

3.2. Обзор плагинов

Краткий обзор плагинов, адаптированных под выполнение таких задач, как SEO оптимизация, безопасность, повышение производительности с указаниями некоторых функций управления системой

управления контентом, неосуществимых через стандартный (немодифицированный) интерфейс, приведен ниже.

Cyr to Lat enhanced – автоматическая транслитерация русского текста с кириллицы на латиницу в URL статей, страниц и рубрик. Транслитерация осуществляется в соответствии с международным стандартом ISO 9.

Disable Comments – позволяет глобально отключить комментарии на сайте WordPress для любого типа страниц.

WP-Optimize – оптимизация базы данных через удаление временных настроек, черновиков и спама в комментариях.

Shortcodes Ultimate – расширяет возможности стандартного редактора страниц.

Easy Updates Manager – контроль над обновлениями CMS и плагинов. Также позволяет отключить обновления на сайте WordPress.

Enhanced Media Library – упорядочивание содержимого библиотеки медиафайлов.

Hyper Cache - кэширование динамически формируемых страниц для последующего отображения обычных HTML-страниц. При повторном посещении страницы загружается кэшированная версия.

Autoptimize – повышает производительность сайта путем сокращения HTML, JS и CSS кода, а также конкатенирует (добавляет одну строку в конец другой) все установленные скрипты и файлы стилей.

All in One SEO Pack – автоматически добавляет мета-теги, оптимизирует заголовки для поисковых систем, предоставляет расширенные возможности использования канонических ссылок и др.

BackUpWordpress – резервное копирование базы данных и исходных файлов сайта.

3.3. Установка и активация плагинов

3.3.1. Перейти на страницу каталога плагинов WordPress, который доступен по ссылке http://ru.wordpress.org/plugins/ или из админпанели, в разделе Плагины > Добавить новый.

3.3.2. В поле “Поиск плагинов” следует ввести название плагина (рис. 32), затем нажать клавишу “Enter”.

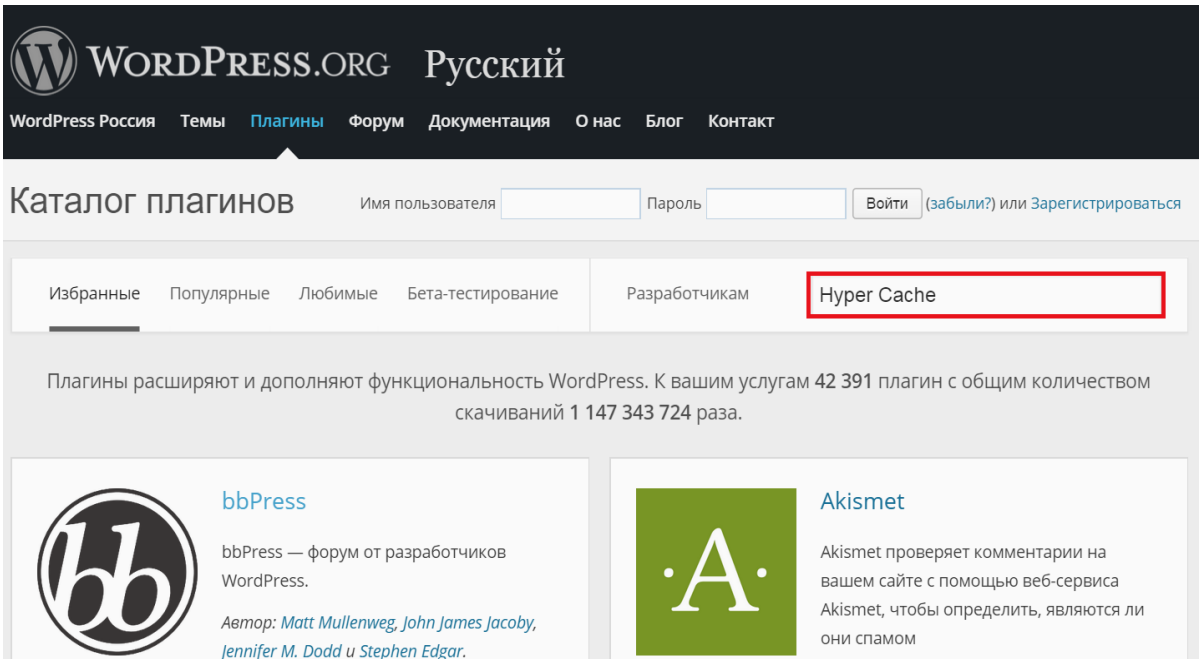

Рис. 32. Страница каталога плагинов WordPress

3.3.3. На странице результатов поиска следует выбрать плагин, название которого соответствует введенному запросу (рис. 33). Если поиск в каталоге осуществлялся через админ-панель, то для установки плагина нажимается кнопка “Установить” (в данном случае, два последующих пункта можно пропустить).

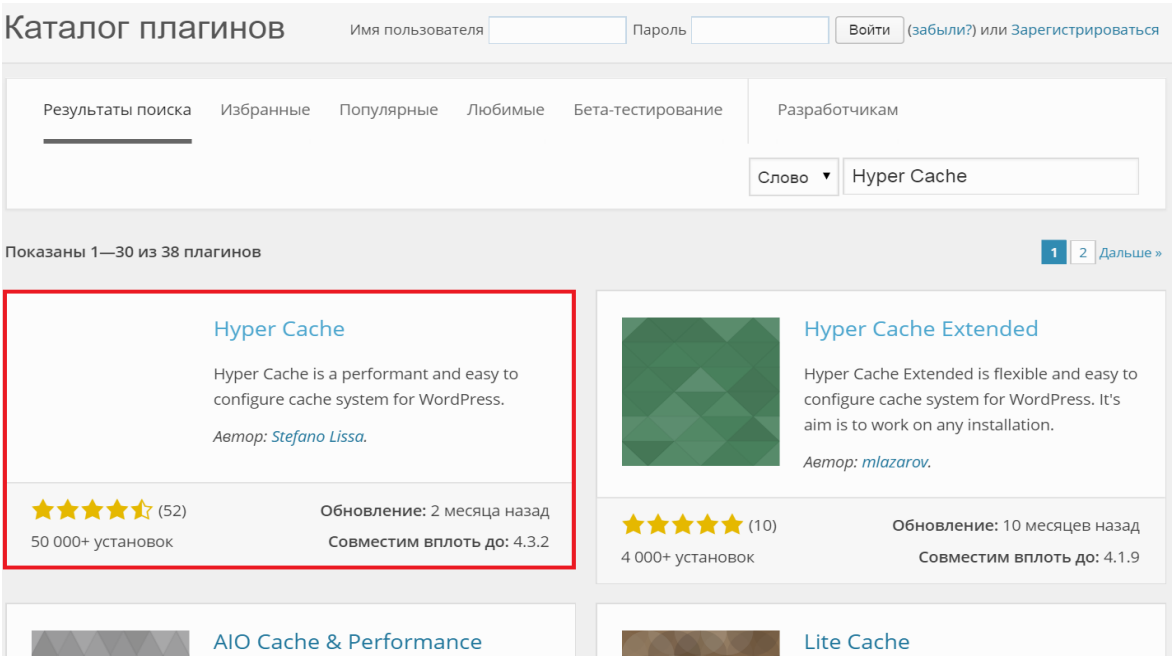

Рис. 33. Страница результатов поиска по каталогу плагинов WordPress

3.3.4. Для скачивания плагина нажимается кнопка “Скачать версию …”.

3.3.5. В админ-панели WordPress перейти в раздел “Плагины”, затем на пункт “Добавить новый”. После чего нажать кнопку “Загрузить плагин”. Далее нажимается кнопка “Выбрать файл”. В появившемся окне выбрать архив с файлами плагина в формате *.*zip* и нажать кнопку “Открыть”. Для установки плагина нажимается кнопка “Установить”.

3.3.6. Для активации плагина перейти в раздел “Плагины” и в меню выбора действий, расположенном под названием плагина, нажать кнопку “Активировать”.

3.4. Настройка постоянных ссылок

3.4.1. Устанавливается плагин **Cyr to Lat enhanced**. Алгоритм установки плагинов описан в п. 3.3.

3.4.2. В админ-панели перейти в пункт меню “Настройки”, затем в подпункт “Постоянные ссылки”.

3.4.3. Нажимается радиокнопка перед “Произвольно” и в поле для ввода вводится оператор `/%category%/%postname%/`, как показано на рис. 34. Таким образом, в URL страницы не будет указываться дата публикации, а только название и раздел, в котором помещена страница. Для сохранения внесенных изменений нажимается кнопка “Сохранить изменения”.

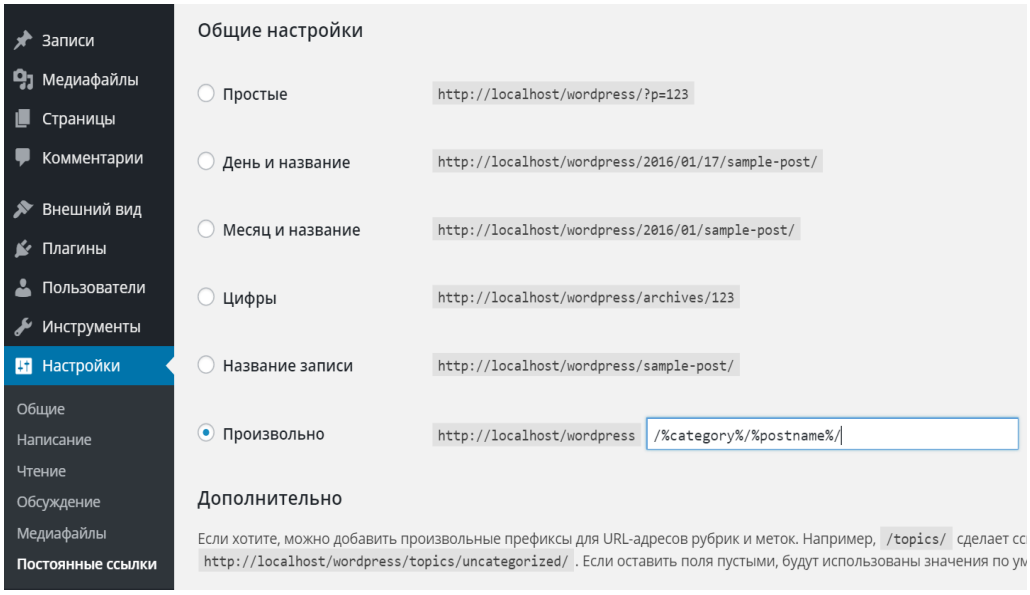

Рис. 34. Визуальная форма для общих настроек постоянных ссылок

3.5. Отключение комментариев и настройка библиотеки медиафайлов

3.5.1. Перейти в пункт меню админ-панели “Настройки”, затем в подменю “Медиафайлы”.

3.5.2. При загрузке изображения через библиотеку медиафайлов происходит обрезка этого изображения соответственно размерам, указанным в настройках, что приводит к созданию дополнительных файлов. Каждый из таких файлов занимает определенное пространство на диске и в базе данных, куда сохраняется информация о каждом загруженном файле. Если файлы, полученные в результате обрезки не используются на сайте, то в целях повышения производительности и экономии места рекомендуется отключить функцию обрезки.

Для этого вносятся изменения в настройки, как показано на рис. 35. Для сохранения изменений нажимается кнопка “Сохранить изменения”.

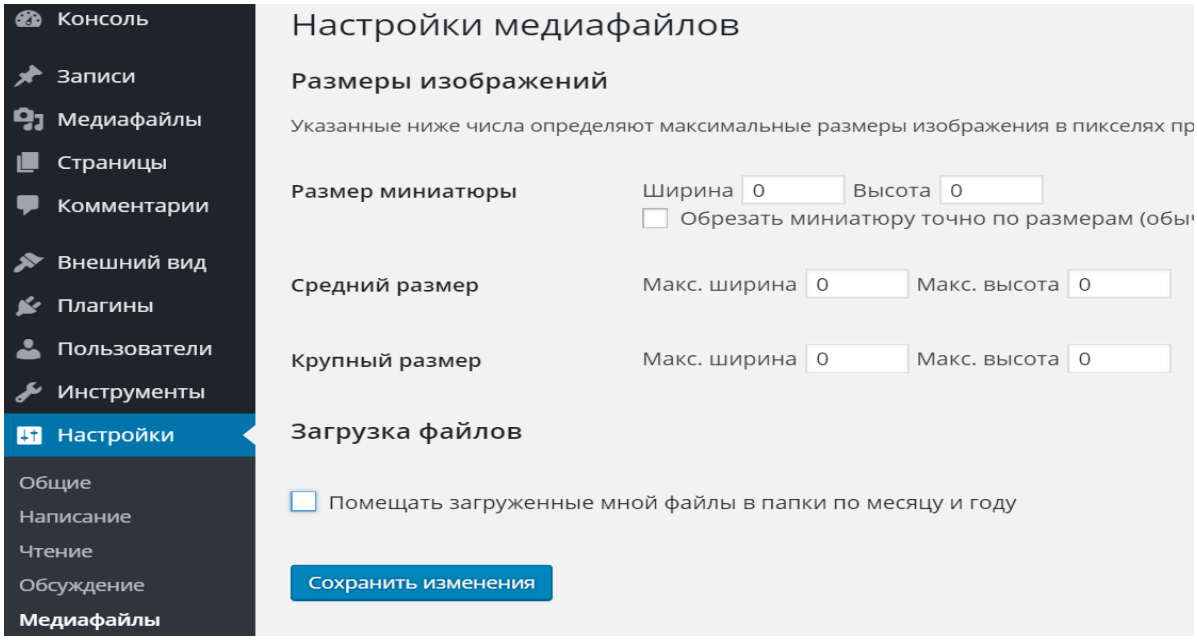

Рис. 35. Настройки медиафайлов

3.5.3. Перейти в подпункт “Обсуждение” и снять галочки с каждого параметра, как показано на рис. 36.

3.5.4. Снимается галочка напротив “Показывать аватары”. После чего нажимается кнопка “Сохранить изменения”.

3.5.5. Установить и активировать плагин **Disable Comments**. Алгоритм установки плагинов описан в п. 3.3.

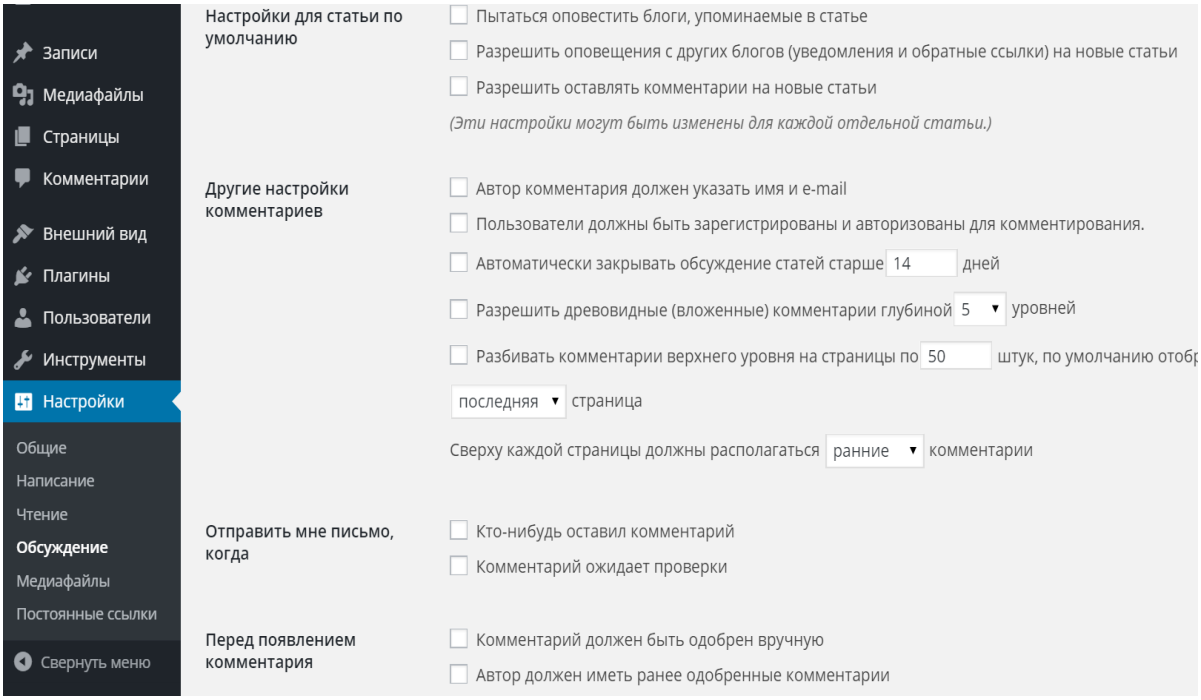

Рис. 36. Настройки комментариев

3.5.6. Для глобального отключения комментариев на сайте перейдите в пункт “Настройки” данного плагина (пункт “Настройки” появится в меню выбора действий после активации плагина), затем нажимается радиокнопка перед пунктом “**Везде:** Отключить все параметры и настройки WordPress, связанные с управлением комментариями”. Для сохранения изменений нажимается кнопка “Сохранить изменения”.

3.5.7. Перейти в директорию C:\xampp\htdocs\wordpress\wp-content\themes\twentysixteen и удалить файл **comments.php**.

3.5.8. Теперь нужно удалить все функции, которые вызывают шаблон комментариев. Для этого в файлах **image.php**, **page.php** и **single.php** удаляется код, изображенный на рис. 37 и сохраняются внесенные изменения.

```
if ( comments_open() || get_comments_number() ) {
    comments_template();
}
```

Рис. 37. Функция вызова шаблона комментариев

3.5.9. В директории C:\xampp\htdocs\wordpress\wp-content\themes\twentysixteen\inc открать файл **template-tags.php**. Найти и удалить функцию, которая определяет вид формы для добавления комментариев и условия, при которых эта форма размещается на сайте

(рис. 38). Для сохранения внесенных изменений нажимается комбинация клавиш “CTRL” +”S”.

```
if ( ! is_singular() && ! post_password_required() &&
 ( comments_open() || get_comments_number() ) ) {
    echo '<span class="comments-link">';
    comments_popup_link( sprintf( __( 'Leave a
    comment<span class="screen-reader-text"> on
    %s</span>', 'twentysixteen' ), get_the_title() )
    );
    echo '</span>';
}
```

Рис. 38. Функция, определяющая оформление и условия отображения формы комментариев

3.5.10. В файле **functions.php**, расположенном в директории C:\xampp\htdocs\wordpress\wp-content\themes\twentysixteen, удалить функцию вызова формы комментариев, приведенную в записи на рис. 39, и сохранить внесенные изменения.

```
if ( is_singular() && comments_open() && get_option(
'thread_comments' ) ) {
    wp_enqueue_script( 'comment-reply' );
}
```

Рис. 39. Функция вызова формы комментариев

3.6. Разработка внешнего вида (дизайна) сайта

В разделе излагается процедура разработки дизайна сайта на основе одного из стандартных шаблонов WordPress — “Twenty Sixteen”. Начиная с 4.7 версии, в качестве шаблона по умолчанию установлен “Twenty Seventeen”. Поэтому в меню Внешний вид > Темы нужно активировать шаблон “Twenty Sixteen”. Для этого курсор «мыши» наводится на блок с соответствующим названием и нажимается кнопка “Активировать”.

3.6.1. Создание меню выбора действий

3.6.1.1. В админ-панели перейти в меню “Страницы”, затем в подменю “Добавить новую”.

3.6.1.2. В поле “Введите заголовок” указать название “Страница 1”. Для сохранения страницы нажать кнопку “Опубликовать”. По такому же принципу создаются две страницы с названиями “Страница 2” и “Страница 3”. Созданные страницы будут использованы в качестве пунктов меню.

3.6.1.3. Перейти в меню “Внешний вид”, затем в подменю “Меню”.

3.6.1.4. В поле “Название меню” ввести “Меню”, затем нажать кнопку “Создать меню”. Алгоритм действий показан на рис. 40.

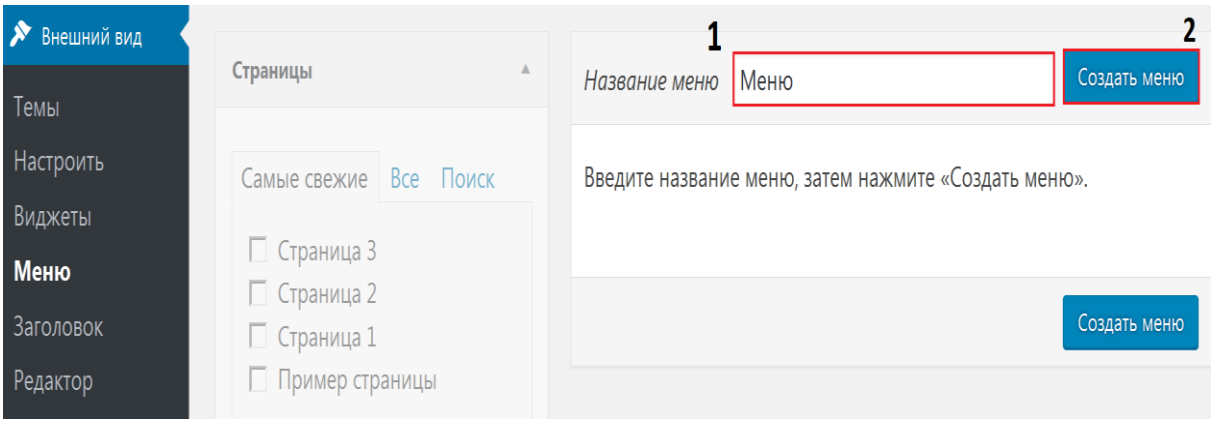

Рис. 40. Создание визуальной формы меню выбора действий

3.6.1.5. Поставить галочки напротив “Страница 1”, “Страница 2” и “Страница 3”. Для добавления выбранных страниц в качестве пунктов меню следует нажать кнопку “Добавить в меню”. Далее поставить галочку перед пунктами “Основное меню” и “Автоматически добавлять в это меню новые страницы верхнего уровня”. Для сохранения внесенных изменений в меню следует нажать кнопку “Сохранить меню”. Последовательность действий цифрами обозначена на рис. 41.

3.6.1.6. Открыть вкладку “Произвольные ссылки”. В поле “URL” ввести ссылку http://#. В поле “Текст ссылки” ввести “Подменю 1”. После чего нажать кнопку “Добавить в меню”. По такому же принципу добавить два подменю с названиями “Подменю 2” и “Подменю 3”.

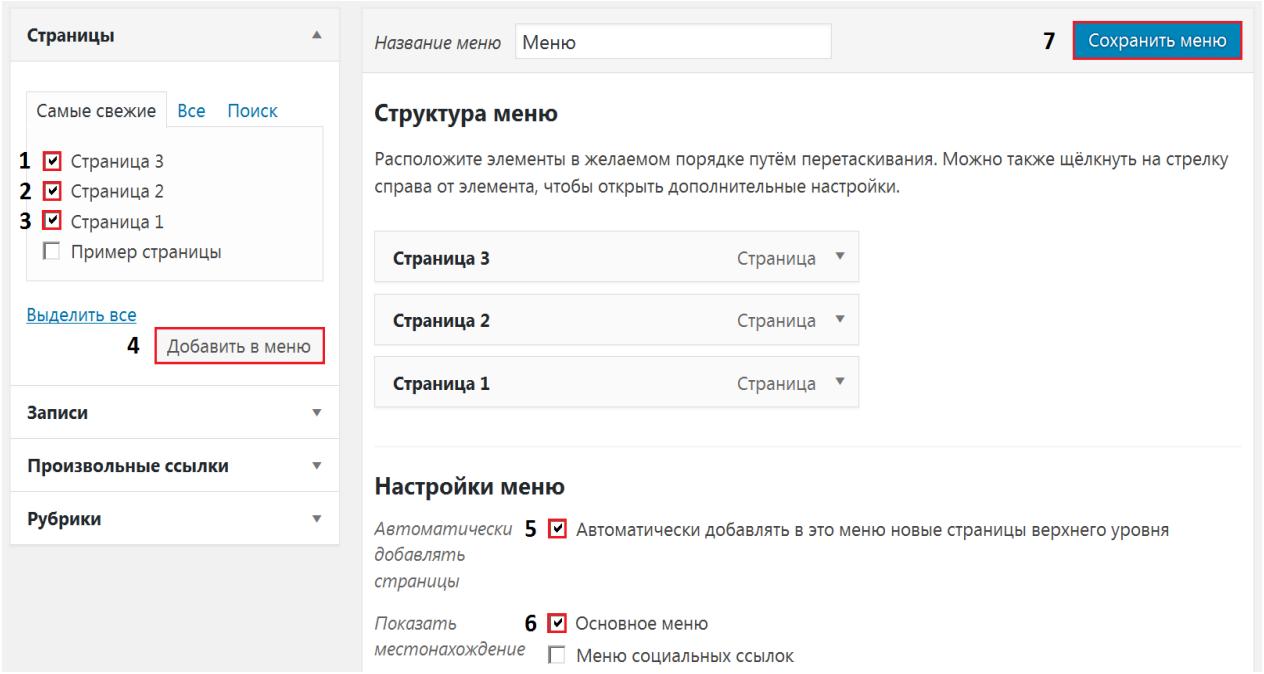

Рис. 41. Визуальная форма интерфейса управления созданием меню

3.6.1.7. Расположить элементы меню в порядке, изображенном на рис. 42. Для сохранения внесенных изменений нажать кнопку “Сохранить меню”.

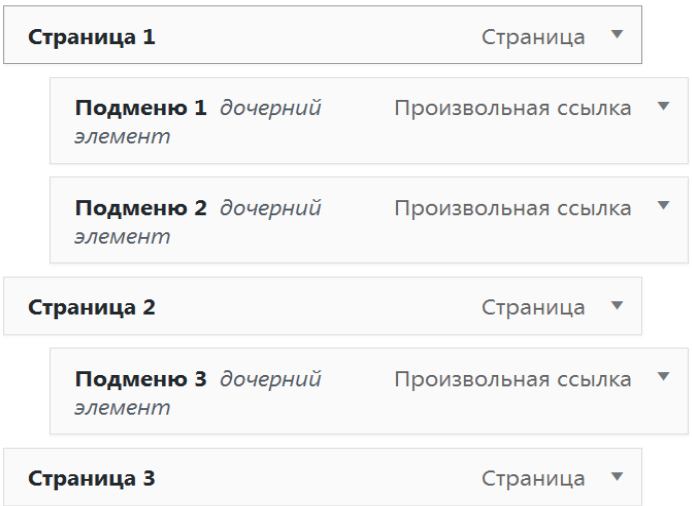

Рис. 42. Структура меню

3.6.2. Добавление фонового изображения в блок header

3.6.2.1. Для примера оформления рекомендуется найти и скачать в Интернете картинку с изображенным на ней автомобилем, размером примерно 1400 x 375 пикселей и с расширением **.jpg*.

3.6.2.2. Перейти в директорию C:\xampp\htdocs\wordpress\wp-content\themes\twentysixteen и создать в ней папку **images**, затем переместить скачанную картинку в эту папку и присвоить картинке имя **background.jpg**.

3.6.2.3. В вышеуказанной директории открыть файл **style.css** и в любом месте файла, кроме комментариев (т. е. между тегами `/*` и `*/`) добавить следующее правило:

```
...
.background {
    position: relative;
    z-index: 0;
    background-repeat: no-repeat;
    background-size: cover;
    background-image: url('images/background.jpg')
}
...
```

Свойства данного правила определяют визуальное отображение фонового рисунка в блоке “header”. Для сохранения изменений нажать комбинацию клавиш “CTRL”+”S”.

3.6.2.4. В той же директории открыть файл **header.php** и найти следующую строку кода:

```
...
<header id="masthead" class="site-header" role="banner">
...
```

В атрибут `class`, после `site-header` через пробел добавить оператор “background”:

```
...
<header id="masthead" class="site-header background"
role="banner">
...
```

Сохранить внесенные изменения. Конечный результат показан на рис. 43.

3.6.2.5. После добавления фонового рисунка в блок “header” замечаем, что меню, название и краткое описание сайта не имеет заливки. Разнообразить такое оформление можно следующим образом.

Открыть файл **style.css** и найти в коде следующее правило:

```
...
.main-navigation .primary-menu {
    border-bottom: 1px solid #d1d1d1;
}
...
```

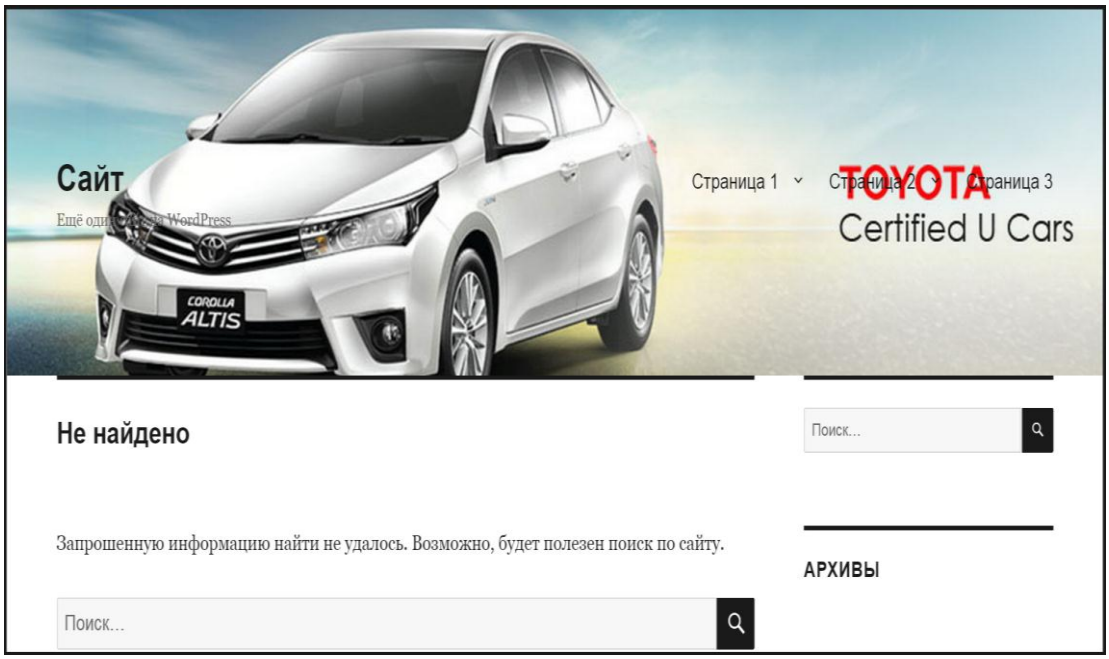

Рис. 43. Фоновый рисунок в блоке “header”

Изменить параметры в соответствии фрагментом кода на рис. 44. Здесь параметр `border-radius` устанавливает радиус закругления углов элемента.

```
.main-navigation .primary-menu {
    border-bottom: 1px solid #d1d1d1;
    background: #FFF;
    border-radius: 5px
}
```

Рис. 44. Фрагмент кода с заданием параметров меню

3.6.2.6. В этом же файле нужно найти правило, определяющее визуальное отображение блока с названием и кратким описанием сайта:

```
...
.site-branding {
    margin: 0.875em auto 0.875em 0;
    /* Avoid overflowing wide custom logo
    in small screens in Firefox and IEs  */
    max-width: 100%;
    min-width: 0;
    overflow: hidden;
}
...
```

Изменить параметры в соответствии с фрагментом кода на рис. 45.

```
.site-branding {
    margin: 0.875em auto 0.875em 0;
    padding: 15px;
    border-radius: 5px;
    background: #FFF
}
```

Рис. 45. Фрагмент кода с параметрами блока с названием и кратким описанием сайта

Конечный результат с учетом корректировки изображен на рис. 46.

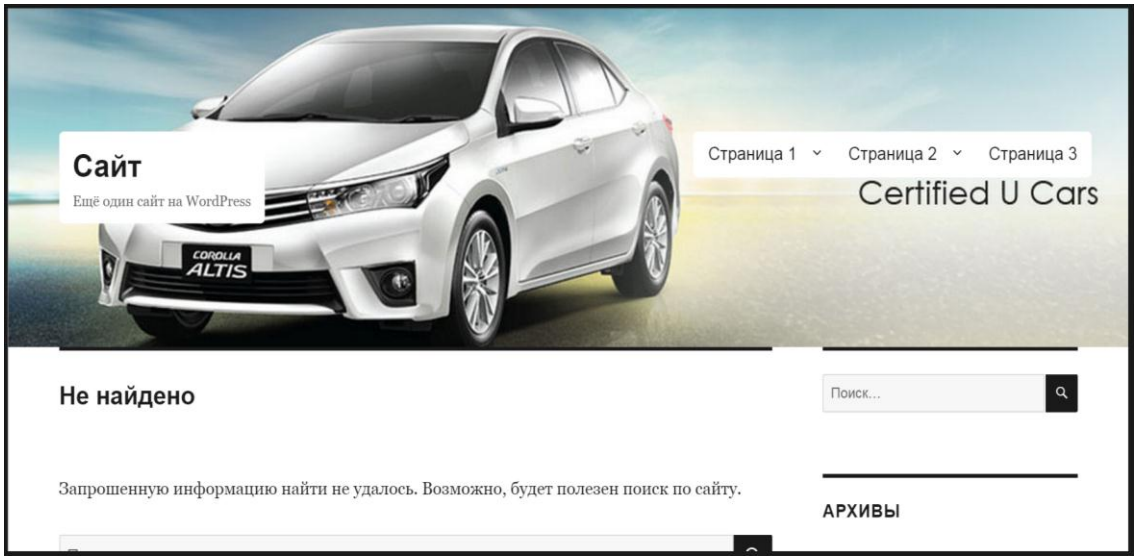

Рис. 46. Фоновый рисунок в блоке “header”

3.6.3. Редактирование элементов главной страницы

3.6.3.1. В директории C:\xampp\htdocs\wordpress\wp-content\themes\twentysixteen открыть файл **style.css**. и найти следующее правило:

```
...
.site {
    margin: 21px;
}
...
```

Изменить значение свойства `margin` с `21px` на `0`.

3.6.3.2. Далее найти следующее правило:

```
...
body:not(.custom-background-image):before,
body:not(.custom-background-image):after {
	background: inherit;
	content: "";
	display: block;
	height: 21px;
	left: 0;
	position: fixed;
	width: 100%;
	z-index: 99;
}
...
```

Удалить параметр `height` со значением `21px;` и сохранить внесенные изменения.

3.6.3.3. Открыть файл **index.php** и удалить элемент PHP кода следующего содержания (эта команда вызывает шаблон боковой панели сайта):

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

Сохранить внесенные изменения.

3.6.3.4. Снова в файле **style.css** найти следующее правило:

```
...
.content-area {
	float: left;
	margin-right: -100%;
	width: 70%;
}
...
```

Изменить параметры в соответствии с фрагментом кода на рис. 47. Таким образом, содержимое главной страницы будет выровнено по центру.

```
.content-area {
	margin: 0 auto;
	width: 70%;
}
```

Рис. 47. Фрагмент кода с параметрами класса “content-area”

3.6.3.5. Далее найти следующее правило:

```
...
.page-header {
    border-top: 4px solid #1a1a1a;
    margin: 0 7.6923% 3.5em;
    padding-top: 1.75em;
}
...
```

Удалите параметр `border-top` со значением `4px solid #1a1a1a;`.

3.6.3.6. Найти следующее правило:

```
...
.post-thumbnail {
    display: block;
    margin: 0 7.6923% 1.75em;
}
...
```

Измените параметры в соответствии с фрагментом кода на рис. 48. Сохранить внесенные изменения.

```
.post-thumbnail {
    display: block;
    margin: 0 7.6923% 1.75em;
    padding-top: 25px;
}
```

Рис. 48. Фрагмент кода с параметрами класса “post-thumbnail”

3.6.3.7. В директории C:\xampp\htdocs\wordpress\wp-content\themes\twentysixteen\template-parts\ открыть файл **content.php** и удалить код функции вывода заголовка страницы, изображенного на рис. 49.

```
<header class="entry-header">
    <?php if ( is_sticky() && is_home() && ! is_paged() )
     : ?>
        <span class="sticky-post"><?php _e( 'Featured',
        'twentysixteen' ); ?></span>
    <?php endif; ?>

    <?php the_title( sprintf( '<h2
    class="entry-title"><a href="%s" rel="bookmark">',
    esc_url( get_permalink() ) ), '</a></h2>' ); ?>
</header><!-- .entry-header -->
```

Рис. 49. Код функции вывода заголовка записи

3.6.4. Размещение информации на главную страницу сайта

3.6.4.1. В админ-панели перейти на вкладку “Записи”, затем навести курсор на запись с названием “Привет, мир!”. Появится меню с вариантами действий. Нужно выбрать пункт “Удалить”.

3.6.4.2. Найти и скачать в Интернете картинку с изображением, например, автомобиля размером примерно 1024 × 768 пикселей.

3.6.4.3. В той же вкладке выбрать подпункт “Добавить новую”. В поле “Введите заголовок” указать “Запись #1”. Ниже, в визуальном редакторе, в качестве примера оформления вставить 10-15 строк текста.

3.6.4.4. В правой части экрана, в блоке “Миниатюра записи”, нажать кнопку “Задать миниатюру”. В открывшемся окне перейти на вкладку “Загрузить файлы”, после чего нажать “Выберите файлы” и выбрать скачанную картинку. После нажатия кнопки “Открыть” начнется процесс загрузки картинки на сайт.

3.6.4.5. Выбрать загруженную на сайт картинку и нажать кнопку “Задать изображение”.

3.6.4.6. Для размещения записи нажать кнопку “Опубликовать”, расположенную в правой части экрана, в блоке с идентичным названием. Конечный результат изображен на рис. 50.

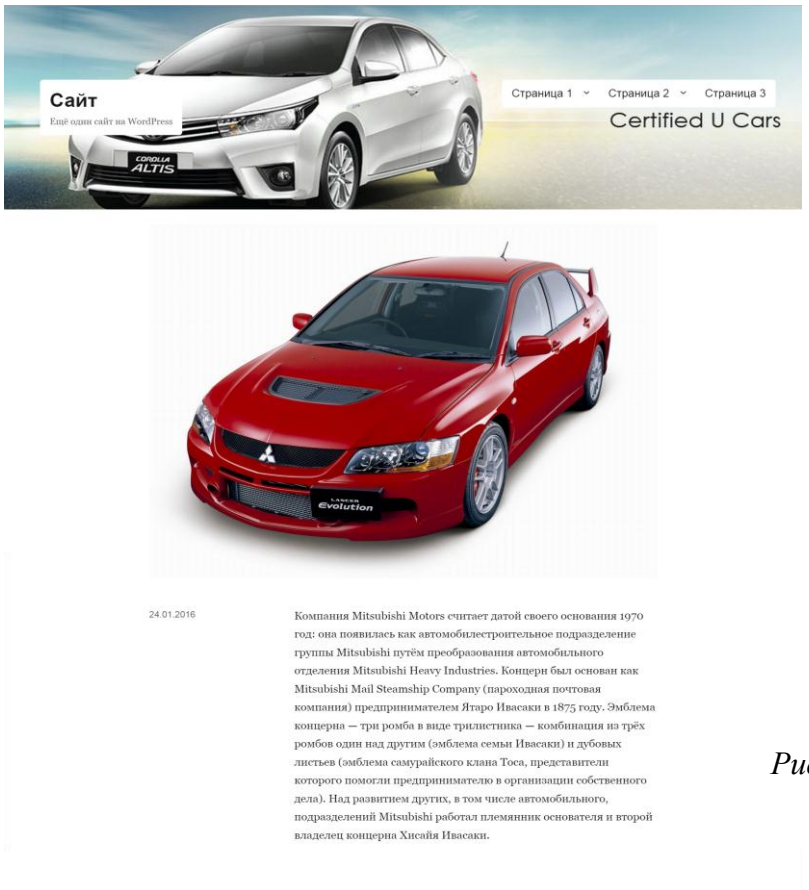

Рис. 50. Пример оформления главной страницы

3.6.5. Оформление блока footer

3.6.5.1. В директории C:\xampp\htdocs\wordpress\wp-content\themes\twentysixteen\ открыть файл **style.css** и найти следующее правило:

```
...
.site-footer {
    padding: 0 7.6923% 1.75em;
}
...
```

Изменить параметры в соответствии с фрагментом кода на рис. 51.

```
.site-footer {
    padding: 0 7.6923% 1.75em;
    box-shadow: inset 0 0 150px rgba(107, 111, 114, .3);
    background-color: #F9F9F9;
    border-top: 1px solid #F9F9F9
}
```

Рис. 51. Фрагмент кода с параметрами класса “site-footer”

3.6.5.2. Далее в коде найти следующее правило:

```
...
.site-info {
    color: #686868;
    font-size: 13px;
    font-size: 0.8125rem;
    line-height: 1.6153846154;
}
...
```

Изменить параметры в соответствии с фрагментом кода на рис. 52 и сохранить внесенные изменения.

```
.site-info {
    color: #686868;
    font-size: 18px;
    line-height: 1.6153846154;
    padding-top: 50px;
    text-align: center;
    margin: 0 auto !important
}
```

Рис. 52. Фрагмент кода с параметрами класса “site-info”

3.6.5.3. Открыть файл **footer.php**. Найти и удалить часть кода, (функции вывода информации о версии системы управления контентом WordPress) изображенного на рис. 53.

```
<a href="<?php echo esc_url( __(
'https://wordpress.org/', 'twentysixteen' ) ); ?>"><?php
 printf( __( 'Proudly powered by %s', 'twentysixteen' ),
 'WordPress' ); ?></a>
```

Рис. 53. Код функции вывода информации о версии CMS WordPress

3.6.5.4. В этом же файле найти следующую строку кода:

...

```
<span class="site-title"><a href="<?php echo esc_url(
home_url( '/' ) ); ?>" rel="home"><?php bloginfo( 'name'
); ?></a></span>
```

...

Изменить ее содержимое по аналогии с нижеприведенным кодом:

...

```
<span>Разработчик: Иванов И. И., группа ИАТТС-15 <br>
&copy; 2017</span>
<a href="<?php echo esc_url( home_url( '/' ) ); ?>"
rel="home"><?php bloginfo( 'name' ); ?></a>
```

...

При оформлении практических работ вместо записи “Иванов И. И., группа ИАТТС-15” указать свои данные. Сохранить внесенные изменения. Конечный результат работы изображен на рис. 54.

3.7. Настройка конфигурационного файла HTTP-сервера Apache применительно к системе управления содержимым WordPress

3.7.1. Открыть файл **.htaccess**, расположенный в директории C:\xampp\htdocs\wordpress\, и после строки кода `# END WordPress` добавить следующее правило:

...

```
<IfModule mod_rewrite.c>
RewriteEngine on
RewriteCond %{REQUEST_URI} ^(.*)?wp-login\.php(.*)$ [OR]
RewriteCond %{REQUEST_URI} ^(.*)?wp-admin$
```

```
RewriteCond %{REMOTE_ADDR} !^127.0.0.1$
RewriteRule ^(.*)$ - [R=403,L]
</IfModule>
```

…

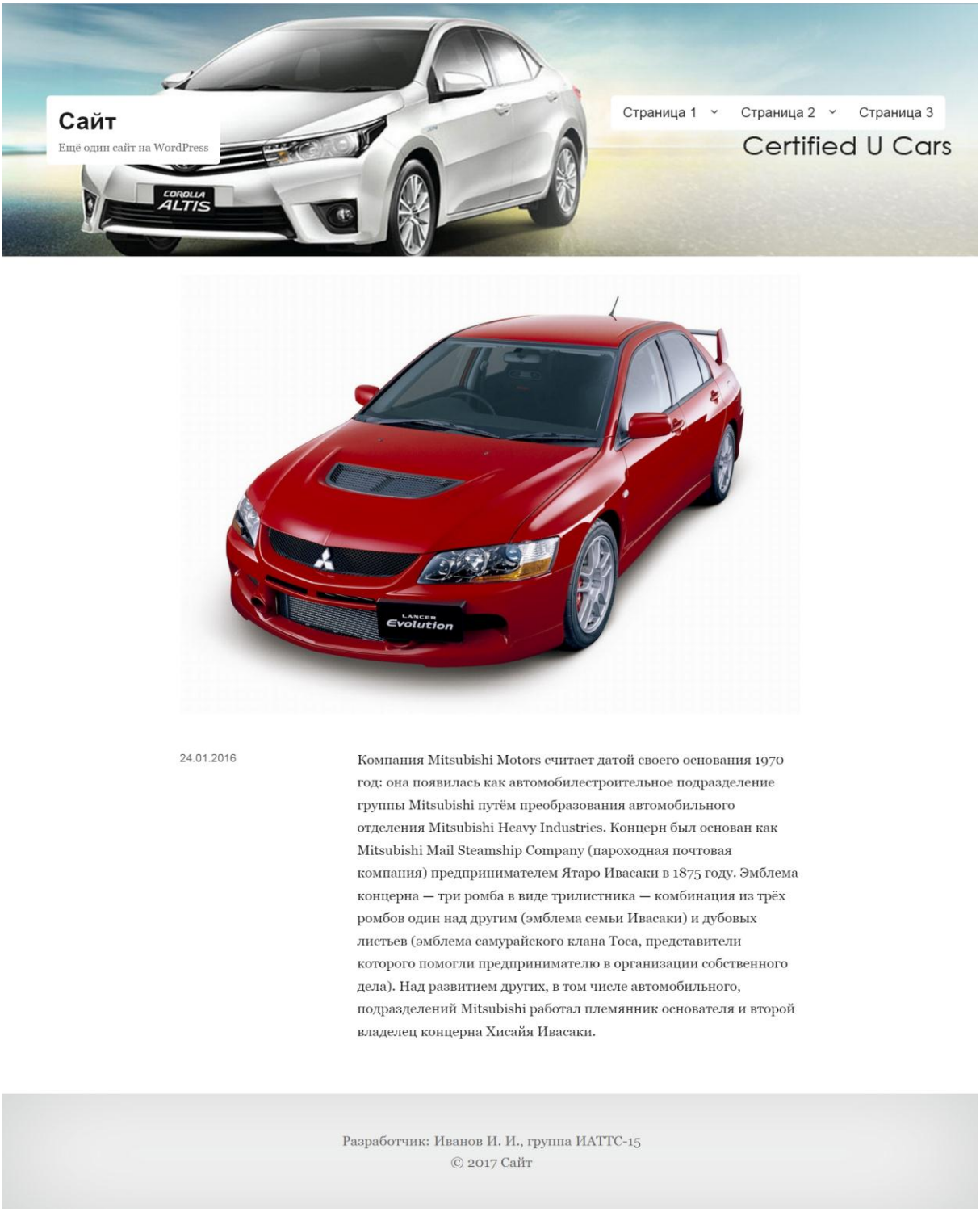

Рис. 54. Главная страница сайта в завершенном виде

Если сайтом управляет один или несколько администраторов, то для повышения безопасности рекомендуется ограничить доступ к админ-панели для всех IP, кроме адресов, с которых осуществляется вход в админ-панель администраторами (например, 127.0.0.1), что выполняет совокупность вышеуказанных команд.

3.7.2. Ниже добавить следующие параметры:

...

```
php_value upload_max_filesize 64M
php_value post_max_size 64M
php_value max_execution_time 300
php_value max_input_time 300
```

...

Максимальный размер загружаемого файла в библиотеке медиафайлов составляет 2 мегабайта. Добавление указанных параметров в конфигурационный файл позволяет изменить стандартное значение.

3.7.3. Ниже добавить следующую директиву:

...

```
Options All -Indexes
```

...

Описание данной команды приведено в п. 2.4.3.

3.7.4. Ниже следует добавить правила, ограничивающие доступ к конфигурационным файлам CMS:

...

```
<files .htaccess>
     <IfModule mod_authz_core.c>
          Require all denied
     </IfModule>
     <IfModule !mod_authz_core.c>
          Order allow,deny
          Deny from all
     </IfModule>
</files>
<files wp-config.php>
     <IfModule mod_authz_core.c>
          Require all denied
     </IfModule>
     <IfModule !mod_authz_core.c>
          Order allow,deny
          Deny from all
     </IfModule>
</files>
```

3.8. Размещение сайта в сети Интернет

Размещение сайта на хостинге является завершающим этапом разработки сайта. Помимо внесения изменений в структуру базы данных и некоторых конфигурационных файлов размещение сайта, прежде всего, предполагает приобретение доменного имени и хостинг площадки [27].

Для размещения сайта на хостинге требуются следующие сетевые реквизиты:

1. Данные для доступа к приложению phpMyAdmin для загрузки базы данных сайта.

2. Реквизиты для доступа к FTP (англ. File Transfer Protocol — протокол передачи файлов) серверу хостинга для размещения файлов сайта.

3. Данные для подключения к серверу баз данных.

4. Доменное имя веб-сайта.

Алгоритм размещения сайта на хостинг имеет такую последовательность:

3.7.1. В первую очередь следует отключить все установленные плагины во избежание возможных конфликтов, связанных с изменением названия базы данных и адреса веб-сайта.

3.7.2. Далее, необходимо изменить URL адрес на доменное имя сайта, указанное в приложении к договору о предоставлении web-услуг. Для этого нужно перейти в админ-панель WordPress, затем в раздел “Настройки”, после чего перейти на вкладку “Общие”. Доменное имя сайта следует указать в полях “Адрес WordPress (URL)” и “Адрес веб-сайта (URL)”, как показано на рис. 55.

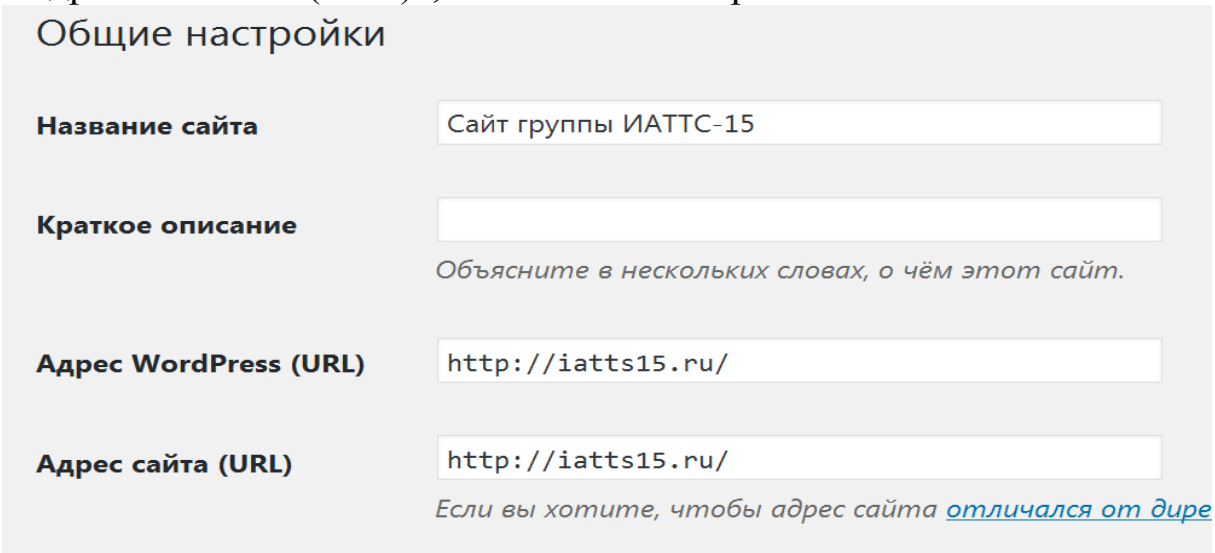

Рис. 55. Панель общих настроек для размещения сайта на хостинге

3.7.3. Сделать копию существующей базы данных. Для этого перейти по ссылке http://localhost/phpmyadmin и в списке баз данных, расположенном в левой части экрана, выбрать “wordpress”. В открывшейся вкладке “Structure” поставить галочку перед полем “Check All”, как показано на рис. 56.

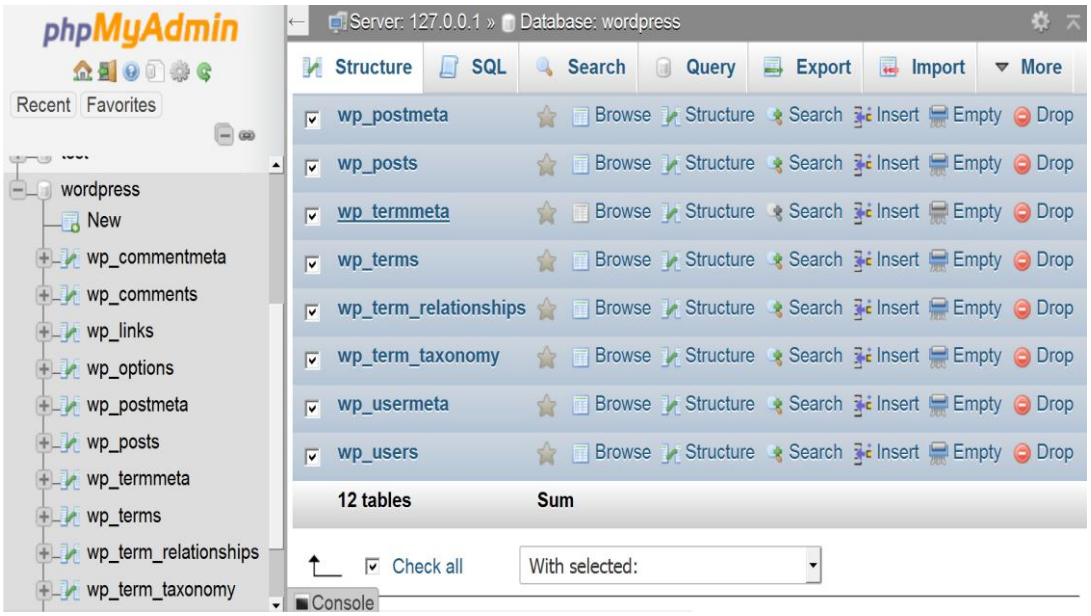

Рис. 56. Настройка структуры базы данных “wordpress”

После чего перейти на вкладку “Export”, нажать кнопку “Go”, как показано на рис. 57 и начнется скачивание файла базы данных в формате **.sql*.

3.7.4. Импортировать полученную базу данных в уже существующую базу данных на хостинге. Для этого перейти на страницу входа в приложение phpMyAdmin и ввести данные в поля “Login” и “Password”, которые указываются в приложении к договору о предоставлении web-услуг, а в учебном процессе при выполнении практических работ эти данные игнорируются. Для завершения операции нажать кнопку “Go” (рис. 58).

Далее открываем базу данных, которая в рассматриваемом примере названа “2017-iatts15”, и переходим на вкладку “Import”. После чего нажимается кнопка “Выберите файл”. В открывшемся окне выбирается файл **wordpress.sql** и нажимается кнопка “Открыть”. Запускается процесс импорта в БД путем нажатия на кнопку “Go”.

3.7.5. Теперь нужно внести изменения в конфигурационный файл **wp-config.php**, расположенный в директории C:\xampp\htdocs\wordpress (рис. 59).

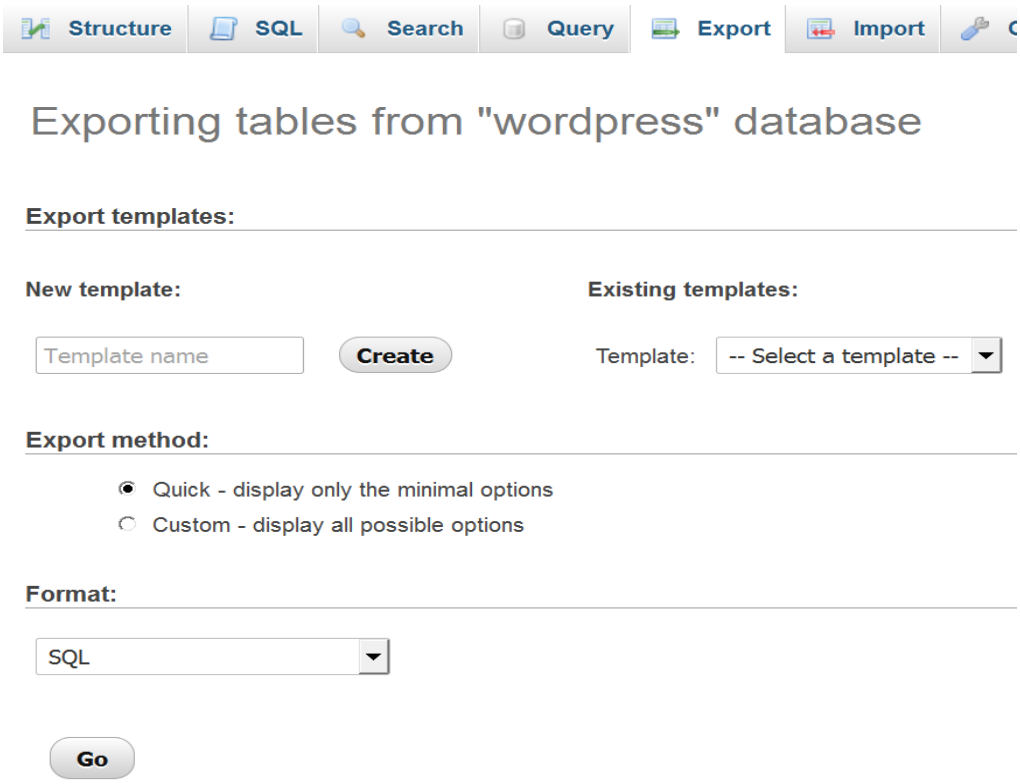

Рис. 57. Визуальная форма для экспортирования базы данных “wordpress”

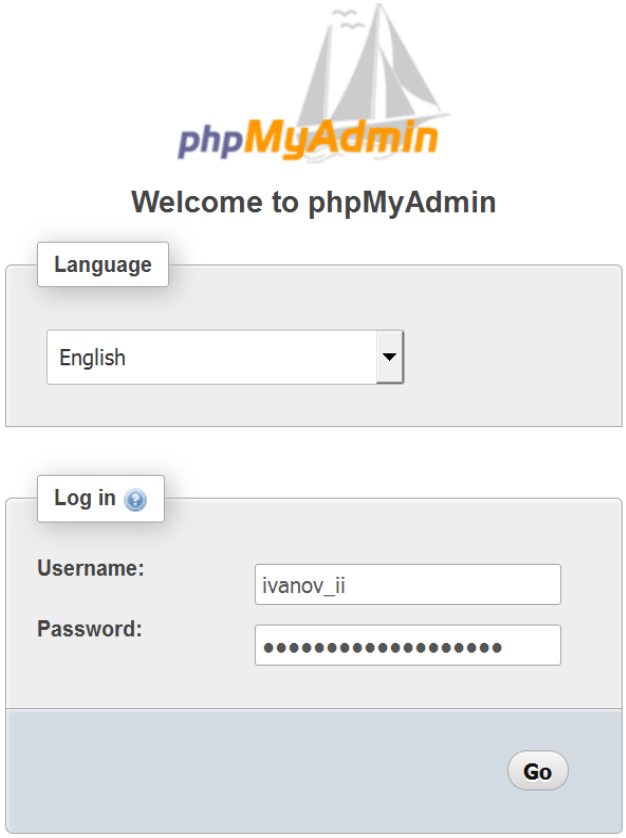

Рис. 58. Визуальная форма страницы входа в приложение phpMyAdmin

```
22  /** Имя базы данных для WordPress */
23  define('DB_NAME', '2017-iatts15');
24
25  /** Имя пользователя MySQL */
26  define('DB_USER', 'ivanov_ii');
27
28  /** Пароль к базе данных MySQL */
29  define('DB_PASSWORD', '12345678910111213l4');
30
31  /** Имя сервера MySQL */
32  define('DB_HOST', 'ns.sqlserver.ru');
33
34  /** Кодировка базы данных для создания таблиц. */
35  define('DB_CHARSET', 'utf8');
```

Рис. 59. Параметры конфигурационного файла **wp-config.php**

На этом процесс создания веб-сайта и размещения его в сети Интернет завершен.

ЗАКЛЮЧЕНИЕ

В результате проработки учебного пособия студент должен знать основные назначения и функции сайтов, освоить алгоритм проектирования, изучить технологии разработки сайтов и основы языков программирования PHP, HTML, CSS. Студент должен научиться разрабатывать структуру сайта и основные элементы главной страницы, программируя в среде языков. Кроме работы в коде, студент должен освоить систему WordPress, которая наиболее подходит для создания многостраничных сайтов, и выполнять настройку всех используемых шаблонов.

Настоящее учебное пособие позволяет расширить кругозор студентов относительно современных информационных технологий в контексте применения их для практических задач в области сервиса транспортных и технологических машин автодорожно-строительного и лесного комплексов.

БИБЛИОГРАФИЧЕСКИЙ СПИСОК

1. Интерактивные онлайн-курсы по HTML, CSS и JavaScript. – URL: http://htmlacademy.ru.

2. Свободная энциклопедия. - URL: http://ru.wikipedia.org.

3. Левин В.И. Информационные технологии в машиностроении: учебн. для студ. сред. проф. образования. – М.: Издательский центр «Академия», 2006. – 240 с.

4. Библиотека I2R. – URL: http://www.i2r.ru/static/255/out_13689.shtml.

5. ГОСТ Р 52872-2012. Интернет-ресурсы. Требования доступности для инвалидов по зрению. – М.: Стандартинформ, 2014. – 46 с.

6. Энциклопедия wikia.com на русском языке. – URL: http://ru.science.wikia.com.

7. ISO/IES/IEEE 24765:2010. Systems and software engineering vocabulary. International Organization for Standardization, Geneva, Switzerland.

8. Ремонт и обслуживание компьютеров. – URL: http:// avangardnt. ru.

9. Учебные материалы по HTML, CSS, JavaScript, PHP. – URL: http://vvz.nw.ru.

10. Сообщество пользователей сайта Treehouse. – URL: http:// teamtreehouse.com.

11. Руководство по настройке .htaccess с примерами. – URL: http://htaccess.net.ru.

12. Google Developers: справочная информация для разработчиков сайтов. - URL: http://developers.google.com.

13. Блог front end разработчика. – URL: http://frontender.com.ua.

14. Руководство по PHP. – URL: http://php.net.

15. Справочник по HTML, CSS, вёрстке, веб-разработке, мобильным приложениям. – URL: http://webref.ru.

16. W3Schools: онлайн веб-учебник. – URL: http://www. w3schools.com.

17. “Can I use” – данные о поддержке современными браузерами элементов HTML5, CSS3. – URL: http://caniuse.com.

18. Видеоуроки по созданию сайта для бизнеса. – URL: http:// ruseller.com.

19. Самоучитель, справочник HTML5, CSS, JavaScript и PHP. – URL: http://www.puzzleweb.ru.

20. Справочник CSS. – URL: http://hellohtml.ru.

21. Анатомия разработки сайтов. Эффективные технические решения создания и ведения сайтов. – URL: https://webgyry.info.

22. Ресурс по PHP, MySQL и другим веб-технологиям. – URL: http://www.php.su.

23. Интернет-магазин Chrome. Каталог расширений функционала браузера Google Chrome. – URL: http://chrome.google.com.

24. Расширения для браузера Mozilla Firefox. - URL: https:// addons.mozilla.org.

25. Справочник по HTML, CSS, вёрстке, веб-разработке, мобильным приложениям. – URL: http://webref.ru.

26. Smart Insights Digital Marketing. Советы по стратегии цифрового маркетинга. – URL: http://www.smartinsights.com.

27. Ремонт и обслуживание компьютеров. – URL: http:// avangardnt.ru.

Учебное издание

Егор Владимирович Побединский
Владимир Викторович Побединский

ПРОЕКТИРОВАНИЕ ВЕБ-САЙТОВ С ИСПОЛЬЗОВАНИЕМ ТЕХНОЛОГИЙ PHP, HTML, CSS И WORDPRESS

ISBN 978-5-94984-651-3

Редактор Р.В. Сайгина
Оператор компьютерной верстки О.А. Казанцева

Подписано к использованию 19.02.2018
Формат 60x84 1/16
Уч.-изд. л. 5,19 Усл. печ. л 6,74
Тираж 300 экз. (Первый завод 50 экз.)
Заказ №

ФГБОУ ВО «Уральский государственный лесотехнический университет»
620100, Екатеринбург, Сибирский тракт, 37
Тел.: 8(343)262-96-10. Редакционно-издательский отдел

Типография ООО «ИЗДАТЕЛЬСТВО УЧЕБНО-МЕТОДИЧЕСКИЙ ЦЕНТР УПИ»
620062, РФ, Свердловская область, Екатеринбург, ул. Гагарина, 35а, оф. 2
Тел.: 8(343)362-91-16