Федеральное государственное бюджетное образовательное учреждение
высшего образования
«КАЛИНИНГРАДСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ
УНИВЕРСИТЕТ»

Г. В. Ломакина

В. А. Петрикин

СЕТЕВЫЕ ИНФОРМАЦИОННЫЕ ТЕХНОЛОГИИ

Учебно-методическое пособие по выполнению лабораторных работ
для студентов бакалавриата по направлению
подготовки 09.03.03 Прикладная информатика

Калининград
Издательство ФГБОУ ВО «КГТУ»
2022

Рецензент:
кандидат технических наук,
доцент кафедры цифровых систем и автоматики института цифровых технологий ФГБОУ ВО «Калининградский государственный технический университет» В. В. Капустин

Ломакина, Г. В., Петрикин, В.А.

Сетевые информационные технологии: учеб.-метод. пособие по выполнению лабораторных работ для студентов бакалавриата по направлению подготовки 09.03.03 Прикладная информатика / **Г. В. Ломакина, В. А. Петрикин**. – Калининград: Изд-во ФГБОУ ВО «КГТУ», 2022. – 68 с.

В учебно-методическом пособии даны методические указания по подготовке и выполнению лабораторных работ.

Пособие подготовлено в соответствии с требованиями утвержденной рабочей программы Профессионального модуля направления подготовки 09.03.01 Информатика и вычислительная техника.

Учебно-методическое пособие по изучению дисциплины рекомендовано к использованию в качестве локального электронного методического материала в учебном процессе методической комиссией института цифровых технологий ФГБОУ ВО «Калининградский государственный технический университет 6 декабря 2022 г., протокол № 10.

ОГЛАВЛЕНИЕ

ВВЕДЕНИЕ 4

Лабораторная работа № 1 «Создание Веб-страницы средствами языка разметки гипертекста HTML» 4

Лабораторная работа № 2 «Каскадные таблицы стилей CSS» 15

Лабораторная работа № 3 «Создание Веб-сайта с использованием конструктора сайтов» 31

Лабораторная работа № 4 «Создание и управление блогами» 37

Лабораторная работа № 5 «Установка и настройка локального веб-сервера» 45

Лабораторная работа № 6 «Установка CMS на локальный веб-сервер» 51

Лабораторная работа № 7 «Создание сайта при помощи современных CMS» 58

ЗАКЛЮЧЕНИЕ 66

ЛИТЕРАТУРА 66

ВВЕДЕНИЕ

Данное учебно-методическое пособие предназначено для студентов направления подготовки 09.03.03 Прикладная информатика, изучающих дисциплину «Сетевые информационные технологии».

Цель лабораторного практикума по дисциплине: получение теоретических и практических основ реализации и функционирования сетевых приложений; приобретение навыков разработки и поддержки веб-приложений.

Лабораторный практикум содержит семь лабораторных работ.

Лабораторные работы проводятся в лабораториях, оснащенных персональными компьютерами, объединенными в локальную сеть с доступом в Интернет.

В результате выполнения лабораторных работ ожидается, что студенты должны

знать:

- основные модели, методы и средства сетевых информационных технологий, способы их применения для решения задач в предметных областях;
- способы эффективной реализации Веб-интерфейсов к базам данных;
- протоколы обмена информацией Веб-серверов и клиентских браузеров;

- уметь:

- выбирать программные средства разработки Веб-приложений;
- разрабатывать и эксплуатировать Веб-приложения;

владеть:

- навыками выбора и применения методов и средств проектирования программного обеспечения Веб-сайтов;
- навыками разработки, отладки и эксплуатации Веб-приложений.

Лабораторная работа № 1 «Создание Веб-страницы средствами языка разметки гипертекста HTML»

Цель работы: изучение структуры HTML-документа, синтаксиса языка HTML (основных тегов и их атрибутов) и применение на практике языка разметки гипертекста HTML при создании простой Веб-страницы.

Материалы, оборудование, программное обеспечение: IBM PC-совместимый персональный компьютер, имеющий доступ в Интернет, редакторы для Веб-разметки и программирования, последняя версия любого из распространенных браузеров - Google Chrome, Mozilla Firefox, Microsoft Edge, Opera.

Критерии положительной оценки: выполнение типового задания, оформление отчета по работе, ответы на вопросы для самопроверки.

Планируемое время выполнения:

Аудиторное время выполнения (под руководством преподавателя): 6 часов.

Время самостоятельной подготовки: 2 часа.

Теоретическое введение

HTML (от англ. Hyper Text Markup Language — «язык гипертекстовой разметки») — стандартизированный язык гипертекстовой разметки веб-документов с целью их создания (верстки) и дальнейшего просмотра в браузере. HTML предоставляет средства для создания заголовков, абзацев, списков, ссылок, цитат и других элементов. С помощью HTML в веб-документ могут быть встроены различные конструкции (изображения, интерактивные веб-формы и другие объекты). Элементы HTML выделяются тегами, записанными с использованием угловых скобок. Браузеры не отображают HTML-теги, но используют их для интерпретации содержимого страницы. С помощью HTML можно легко создать относительно простой, но красиво оформленный документ. Помимо этого, в HTML внесена поддержка гипертекста и элементов мультимедиа.

Первым общедоступным описанием HTML был документ «Теги HTML», впервые упомянутый в Интернете Тимом Бернерсом-Ли в конце 1991 г.

24 декабря 2018 г. была представлена версия HTML 5.3.

При этом в HTML 5 были введен ряд новых тегов, например:

- тег электронной почты. Это совершенно новый тег, появившийся в HTML 5. Данный элемент ввода является тегом формы, который осуществляет проверку или аутентификацию введенного значения. Это дает уверенность в том, что введенные данные являются подлинным именем электронной почты.

- тег пароля. Данный тег является элементом ввода и предназначен для ввода пароля пользователя. При использовании этого тега пароль будет виден во время ввода и показан специальными символами. Этот тег защищает пароль символическим экраном.

- аудиотег. Данный тег был добавлен для вставки аудио на веб-страницы.

- семантические теги. Еще одно название структурных тегов. С помощью семантических тегов вы можете распределять и разделять веб-страницу HTML на различные структуры. Эти структуры объединяются, чтобы сформировать веб-страницу HTML.

- теги разделов. Эти теги позволяют разбивать HTML документ на разделы. Важными семантическими/структурными тегами являются пояснения к изображениям, шапка/заголовок и подвал.

Основная цель внедрения HTML 5 состояла в том, чтобы достичь две цели - улучшить язык и соответствовать новейшим разработкам в области мультимедиа. На HTML 5 удобнее писать, поддерживать, реструктурировать документ. Он лучше подходит для поисковой оптимизации (SEO), агрегаторов контента и систем чтения каналов, легко доступен на мобильных устройствах, может работать с более медленным подключением к Интернету, безопасный и простой путь для добавления мультимедийных элементов.

Эволюция и развитие HTML продолжаются более двух десятилетий. От

первой и начальной версии HTML 1.0 до самой продвинутой версии HTML 5.3, язык HTML модернизировался, улучшился и стал значительно продуктивнее. Консорциум W3C продолжает работать над тем, чтобы все следовали одному стандарту. В ближайшее время ожидается реализация HTML 6.

СИНТАКСИС HTML

Структура HTML-документа

HTML-документ представляет собой текстовый файл с расширением *.html или *.htm.

Каждый HTML-документ, отвечающий спецификации HTML любой версии, должен начинаться с строки объявления !DOCTYPE, которая необходима для переключения браузера в режим соответствия стандартам. Только в этом режиме можно рассчитывать на единообразное отображение HTML-страницы в разных браузерах. Объявление doctype не является тегом HTML и до HTML5 было довольно длинным и громоздким. Новый doctype HTML5 выглядит так:

```
<!DOCTYPE html>
```

Эта строка поможет браузеру определить, как правильно интерпретировать данный документ.

Документ состоит из двух основных блоков – "заголовка" и "тела документа". Заголовок определяется с помощью элемента head, а тело – элементом body. Рекомендуется использовать имена элементов и атрибутов нижнего регистра.

Заголовок содержит "техническую" информацию о документе, хотя чаще всего используется только для обозначения его названия (см. элемент title).

```
<html>
   <head>
   <title>Заголовок документа</title>
   </head>
<body>
Текст документа
</body>
</html>
```

Элементы и тэги

Код HTML-документа, состоит из элементов, а элементы состоят из тегов. Элементы определяют, каким образом данные должны быть отображены в браузере. Элемент состоит из открывающего и закрывающего тэгов:

<elementName>... some content ...</elementName>

Тег – часть элемента, представляющая из себя текст, заключенный в угловые скобки <>.

Большинство тегов HTML кодируются парами из открывающего и

закрывающего тегов. Они называются парными тегами. Парные HTML тэги являются контейнерами для других элементов страницы. Открывающий HTML тэг обозначает начало контейнера, а закрывающий HTML тэг обозначает окончание контейнера.

Эта пара контейнерных тегов охватывает данные, к которым применяется форматирование.

Смысл HTML-элемента состоит в применении форматирования к содержимому между начальным и конечным тэгами. Применяемое форматирование зависит от имени элемента.

Синтаксис языка HTML предусматривает опциональные парные тэги, например<html>, <head>, <body>. Эти тэги мы можем не писать и браузер сгенерирует их за нас.

Некоторые парные тэги предусматривают опциональный закрывающий тэг: практически все тэги для создания HTML таблиц, тэг списка <li>, тэг параграфа <p> и другие.

Наряду с парными в HTML существуют одиночные HTML тэги, эти тэги не имеют никакого содержимого и используются для того, чтобы сформировать графический элемент на странице. Для одиночных HTML тэгов закрывающий тэг запрещен стандартом. Среди одиночных HTML тэгов нет опциональных тэгов.

Такие пустые тэги кодируются специальным образом – они должны содержать символ наклонной черты (/) непосредственно перед закрывающей угловой скобкой.

Атрибуты тэгов

Тэги, помимо имени, могут содержать атрибуты — элементы, дающие дополнительную информацию о том, как браузер должен обработать текущий тэг. У любого HTML тэга могут быть атрибуты, причем, если вы даже не написали явно HTML атрибут, то браузер все равно определит какие-то значения по умолчанию для каждого тэга в документе. Атрибут состоит из двух частей: свойства и значения.

Наборы допустимых атрибутов индивидуальны для каждого тэга. Общие правила записи атрибутов заключаются в следующем. После имени тэга могут следовать атрибуты, которые отделяются друг от друга пробелами. Порядок следования атрибутов тэга произволен. Многие атрибуты требуют указания их значений, однако некоторые атрибуты не имеют значений или могут записываться без них, принимая значения по умолчанию. Если атрибут требует значения, то оно указывается после названия параметра через знак равенства. Значение атрибута может записываться в кавычках, так и без них. Единственным случаем, когда использование кавычек обязательно, является случай, когда в значении атрибута имеются пробелы. В значениях атрибутов (в отличие от названий тэгов и самих атрибутов) иногда важен регистр записи.

Пример записи тэга с атрибутами:

<table border align="left">

Здесь для тэга <table> задано два параметра. Первый параметр border

указан без значения. Второй параметр align имеет значение "left".

Текст комментария размещается между ограничителями <!--и --> и не отображается в окне браузера. Комментарии могут быть многострочными, например,

<!-- начало комментария .....
продолжение комментария ..... -->

Заголовок HTML-документа

Раздел документа head определяет его заголовок и не является обязательным тэгом. Задачей заголовка является представление необходимой информации для программы, интерпретирующей документ. Тэги, находящиеся внутри раздела head (кроме названия документа, описываемого с помощью тэга <title>), не отображаются на экране.

<head> ... </head> указывает на начало и конец заголовка документа. Помимо наименования документа (см. <title>), в этот раздел может включаться множество служебной информации.

Название документа

Тэг-контейнер <title> является единственным обязательным тэгом заголовка и служит для того, чтобы дать документу название. Оно обычно показывается в заголовке окна браузера. Все, что находится между метками <title> и </title>, интерпретируется браузером как название документа. Рекомендуется название не длиннее 64 символов.

По умолчанию текст, содержащийся в названии документа, используется при создании закладки (bookmark) для документа. Поэтому, для большей информативности, следует избегать безликих названий (Home Page, Index и т. д.). Название документа должно кратко характеризовать его содержание.

Связи между HTML-документами

Часто HTML-документы связаны между собой, то есть имеют ссылки друг на друга. Ссылки могут быть как абсолютные, так и относительные.

Абсолютные ссылки могут быть громоздкими и переставать работать, если перемещен младший по иерархии документ. Относительные ссылки легче вводить и обновлять, но и эта связь обрывается, если перемещен старший по иерархии документ. Оба вида связей могут нарушиться при переносе документа с одного компьютера на другой.

Для решения этой проблемы используются тэги <base> и <link>, которые включаются в заголовок для того, чтобы связь между документами не нарушалась.

Тэг <base> служит для указания полного базового URL-адреса документа. С его помощью относительная ссылка продолжает работать, если документ переносится в другой каталог или даже на другой компьютер.

Тэг <base> имеет один обязательный параметр href, после которого указывается полный URL-адрес документа.

Тэг <link> указывает на связь документа, содержащего данный тэг, и

другого документа или объекта. Он состоит из URL-адреса и параметров, конкретизирующих отношения документов. Заголовок документа может содержать любое количество тэгов <link>.

В раздел заголовка может быть добавлен еще один тэг <meta>, позволяющий авторам документа определять информацию, не имеющую отношения к HTML. Эта информация используется браузером для действий, которые не предусмотрены текущей спецификацией HTML.

Пример:

<meta http-equiv="refresh" content="60"/>

Этот тэг инструктирует браузер перезагружать страницу каждые 60 секунд, что может быть полезно, если данные на странице часто обновляются.

Кроме того, в качестве примера можно привести указание ключевых слов, используемых поисковыми системами. Этот способ позволяет включать в индекс документа дополнительные слова, которые могут явно не входить в его содержание. Для этого в тэге <meta> в качестве значения параметра name указывается имя некоторого свойства.

Еще одно назначение тэга <meta> — это указание кодировки текста. Например, для текста на русском языке в кодировке Windows используется следующий тэг:

<meta http-equiv="Content-Type" content="text/html; charset=Windows-1251"/>

Тело документа

Пара меток <body> ... </body> указывает на начало и конец тела HTML-документа, которое, собственно, и определяет содержание документа.

Контрольные вопросы для самопроверки

1. Структура HTML-документа.
2. Какая информация представлена в строке !DOCTYPE?
3. Что такое тэг? Как он записывается?
4. Парные и непарные тэги.
5. Какие тэги называются опциональными?
6. Правила записи атрибутов тэгов.
7. Какие тэги могут присутствовать в заголовке HTML-документа?
8. Объясните назначение тэгов <base> и <link>
9. Какие функции выполняет W3C?
10. Как задаются комментарии в HTML-документе?

Задание к лабораторной работе

Студент получает задание на выполнение работы. Типовыми заданиями являются:

1. Разработать HTML-страницу, содержащую резюме студента с краткой биографией, например, такую, как на рис. 1.1. В ходе разработки обязательно использовать следующие теги:

Текстовые блоки – <p>

Заголовки – <h1>, <h2>, <h3>

Форматирование текста – <font>, <i>, <b>, <u>, <small>,

Списки (нумерованный, маркированный) – , <ol>,

Изображения – <img>

Таблицы – <table>, <tr>, <td>

Ссылки – <a>

Прочие –
, <hr>, <title>, <sub>, <sup>

В ходе выполнения работы необходимо:

- изучить атрибуты тегов;
- уметь менять фон страницы, всей таблицы и отдельных ячеек;
- уметь объединять ячейки в таблице по горизонтали, вертикали; задавать ширину, высоту ячеек и таблицы, отступы внутри и между ячеек;
- уметь выравнивать текст в абзацах, заголовках, таблицах. Уметь устанавливать границу для таблиц и изображений;
- уметь изменять тип маркеров и тип нумерации для списков;
- уметь вставлять спецсимволы (€,©,®,¼, ±,÷ и др.);
- уметь использовать атрибут target для ссылок и вставлять ссылки на адрес электронной почты, сайт университета;
- уметь вставлять изображения.

Примерный вид резюме на следующей странице.

2. Подготовить в WORD идентичный документ "Резюме", сохранить его как веб-страницу. Сравнить размер и состав тегов в двух документах.

РЕЗЮМЕ

 Ф О Т О	Фамилия: Имя: Отчество: Дата рождения: Полных лет: Телефон: Электронная почта:

1. Образование	
2. Достижения	
3. Опыт работы	
4. Навыки, умения, личные качества	
• Общительный • Трудолюбивый • Отзывчивый • . . .	
5. Желаемое место работы	

Рис. 1.1. Примерный вид HTML-страницы с резюме

Методические указания и порядок выполнения работы

Необходимые инструментальные средства: текстовый редактор или текстовый процессор; редактор исходного кода веб-страниц, поддерживающий язык разметки HTML; веб-браузер для запуска и проверки написанных веб-страниц. ТОП 10 лучших HTML редакторов:

Visual Studio Code, Notepad ++, Sublime Text, WebStorm на базе IntelliJ, Vim, Eclipse, Atom, Adobe Dreamweaver CC.

Одним из самых простых и наиболее популярных редакторов исходного кода веб-страниц является Notepad++, который можно найти по адресу http://notepad-plus-plus.org/. К его преимуществам можно отнести бесплатность, подсветку тегов html. Рекомендуется ориентироваться именно на этот редактор.

Несколько большими возможностями, чем Notepad++, и, кроме того, кроссплатформенностью (работа не только в ОС Windows, но и в MacOS и в операционных системах на основе Linux) обладает редактор Visual Studio Code (доступен по ссылке https://code.visualstudio.com/). Для проверки поддержки HTML5 конкретным браузером можно использовать специальный сервис http://html5test.com.

Пример разработки простой веб-страницы.

1. Создать папку HTML.

2. Создать в текстовом редакторе текстовый файл. Сохранить его в папке HTML с именем index и с расширением на .html.

3. Открыть этот файл в редакторе исходного кода, например, в Notepad++. Добавить в файл следующий текст:

```
<!DOCTYPE html>
<html>

</html>
```

Для создания HTML- документа нужны в первую очередь два элемента: DOCTYPE и html. Элемент doctype или Document Type Declaration сообщает веб-браузеру тип документа. <!DOCTYPE html> указывает, что данный документ является документом html и что используется html5, а не какая-то другая версия языка разметки.

А между открывающим и закрывающим тегами элемента html будет размещено все содержимое документа.

4. Внутри элемента html разместить два других элемента: head и body. Элемент head содержит метаданные веб-страницы - заголовок веб-страницы, тип кодировки и т.д., а также ссылки на внешние ресурсы - стили, скрипты, если они используются. Элемент body определяет содержимое html-страницы.

Теперь изменим содержимое файла index.html следующим образом:

```
<!DOCTYPE html>
<html>
        <head>
   <title>Документ HTML5</title>
        </head>
        <body>
   <div>Содержание документа HTML5</div>
        </body>
</html>
```

Для корректного отображения символов браузером предпочтительно указывать кодировку. В данном случае с помощью атрибута charset="utf-8" в элементе head можно задать кодировку utf-8 <meta charset="utf-8">.

При создании веб-страницы с помощью редактора Notepad++ и выборе при его установке русского языка кодировка utf-8 выбирается автоматически.

5. Сохраните и откройте файл index.html в браузере. В основном поле браузера отобразится текст, определенный внутри элемента body:

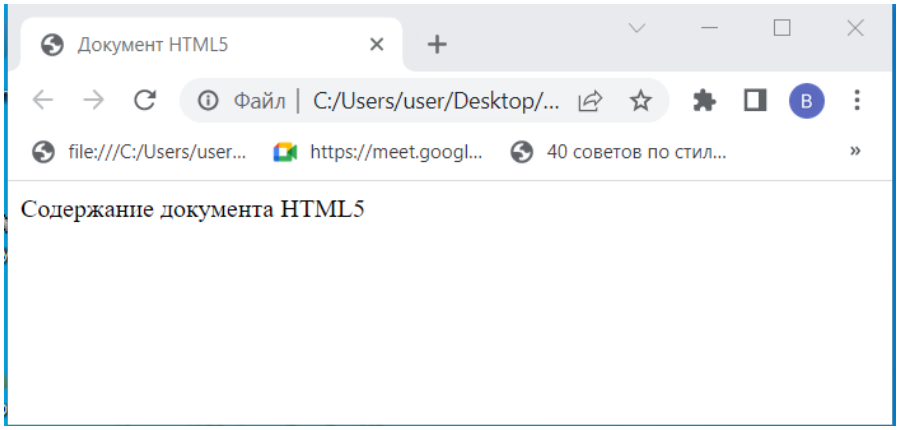

Рис. 1.2. Начало создания веб-страницы

Таким образом, создан документ HTML5. Так как в элементе title указан заголовок "Документ HTML5", то именно такое название будет иметь вкладка браузера.

Еще один пример:

```
<!DOCTYPE html>
<html>
<head>
<title>Название страницы</title>
</head>
<body>
<h1>Мой первый заголовок</h1>
<p>Мой первый абзац.</p>
</body>
</html>
```

Пояснения к примеру:
Декларация <!DOCTYPE html> определяет этот документ как HTML5
Элемент <html> является корневым элементом HTML-страницы
Элемент <head> содержит мета-информацию о документе
Элемент <title> задает заголовок документа
Элемент <body> содержит видимое содержимое страницы
Элемент <h1> определяет большой заголовок
Элемент <p> определяет абзац

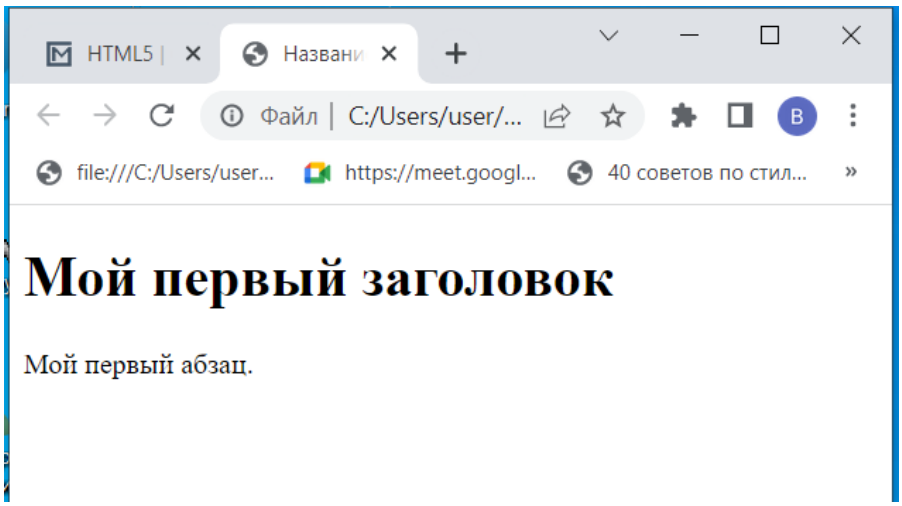

Рис. 1.3. Продолжение создания веб-страницы

С подробной информацией о языке разметки HTML и веб-разработке можно познакомиться в следующих источниках Интернет:

1. Основы HTML [Электронный ресурс]: - Режим доступа: https://developer.mozilla.org/ru/docs/Learn/Getting_started_with_the_web/HTML_basics (дата обращения 11.11.2022).

2. Справочник по HTML [Электронный ресурс]: - Режим доступа: http://htmlbook.ru/html (дата обращения 11.11.2022).

3. Изучение веб-разработки [Электронный ресурс]: - Режим доступа: https://developer.mozilla.org/ru/docs/Learn (дата обращения 11.11.2022).

4. HTML с нуля. Лучший HTML учебник для начинающих и чайников [Электронный ресурс]: - Режим доступа: https://html5css.ru/html/default.php (дата обращения 11.11.2022).

Требования к отчету и защите

В отчете указываются название, цель работы. Описание выполненных лабораторных заданий, HTML-код разработанной Веб-страницы с результатами в виде скриншотов и выводами по каждому заданию.

На защите проверяются приобретенные знания теоретического и практического материала по ответам на контрольные вопросы для самопроверки.

Лабораторная работа № 2 «Каскадные таблицы стилей CSS»

Цель работы: приобретение знаний и практических навыков в использовании каскадных таблиц стилей CSS.

Материалы, оборудование, программное обеспечение: IBM PC-совместимый персональный компьютер, IBM PC-совместимый персональный компьютер, имеющий доступ в Интернет, редакторы для Веб-разметки и программирования, последняя версия любого из распространенных браузеров - Google Chrome, Mozilla Firefox, Microsoft Edge, Opera.

Критерии положительной оценки: выполнение типового задания, оформление отчета по работе, ответы на вопросы для самопроверки.

Планируемое время выполнения:

Аудиторное время выполнения (под руководством преподавателя): 6 часов.

Время самостоятельной подготовки: 2 часа.

Теоретическое введение

CSS (Cascading Style Sheets — каскадные таблицы стилей) — технология описания внешнего вида документа, написанного языком разметки.

Преимущественно используется как средство оформления веб-страниц в формате HTML, но может применяться с любыми видами документов в формате XML.

CSS используется создателями веб-страниц для задания цветов, шрифтов, расположения и других аспектов представления документа. Основной целью разработки CSS являлось разделение содержимого (написанного на HTML) и представления документа (написанного на CSS). Это разделение может увеличить доступность документа, предоставить большую гибкость и возможность управления его представлением, а также уменьшить сложность и повторяемость в структурном содержимом. Кроме того, CSS позволяет представлять один и тот же документ в различных стилях или методах вывода, таких как экранное представление, печать, и даже чтение голосом.

Понятие стиля, таблицы стилей и CSS

Стиль представляет собой набор параметров, определяющих внешний вид документа HTML при его отображении в окне браузера: шрифты и цвета заголовков разных уровней, шрифт и разрядка основного текста, задаваемого в тэге абзаца <p>, и т. д. Стиль задается по определенным правилам

Таблица стилей - набор правил отображения, применяемых в документе, к которому присоединена соответствующая таблица стилей. Таблица стилей — это шаблон, который управляет форматированием тэгов HTML в веб-документе.

Термин "каскадные" используется потому, что возможно применять несколько таблиц стилей для управления форматированием одного документа

HTML, а браузер по определенным правилам выстраивает приоритетность применения этих таблиц.

Применение технологии CSS обеспечивает:

- гибкое размещение информации о стиле. Помещение таблиц стилей в отдельные файлы упрощает их повторное использование. Иногда полезно включать инструкции по представлению в документ, к которому они применяются, в начало документа или в атрибуты элементов в теле документа.

- каскады. Эта возможность обеспечивается некоторыми языками таблиц стилей, такими как CSS, для объединения информации о стиле из нескольких источников. Это могут быть, например, корпоративные положения о стиле, стили, общие для группы документов, а также стили, специфичные для одного документа. С использованием раздельного хранения эти таблицы стилей могут использоваться повторно, что упрощает работу авторов и повышает эффективность сетевого кэширования. Каскад определяет упорядоченную последовательность таблиц стилей, в которой правила более поздних таблиц имеют приоритет над более ранними.

- зависимость от устройств. HTML позволяет авторам разрабатывать документы, независимые от устройств. Это позволяет пользователям обращаться к веб-страницам с использованием различных устройств, например, графических дисплеев для компьютеров под управлением различных ОС, телевизионных устройств, телефонов и портативных устройств на базе PDA, речевых браузеров и тактильных устройств на базе азбуки Бройля. Таблица стилей, напротив, применяются к конкретным устройствам или группам устройств. Языки таблиц стилей могут включать функции описания зависимости от устройств в одной таблице.

- альтернативные стили. Авторы могут предлагать читателям несколько способов просмотра документа. Например, таблица стилей для представления компактных документов с мелким шрифтом, или таблица, задающая крупные шрифты для удобства чтения. Автор может указать предпочитаемую таблицу стилей, а также альтернативные таблицы для определенных пользователей или устройств. Агенты пользователей должны предоставлять пользователям возможность выбора одной из альтернативных таблиц или отключать все таблицы стилей.

- вопросы производительности. Загрузка внешней таблицы стилей может привести к задержке общего представления материала для пользователя. Подобные ситуации возникают и в том случае, если в заголовок документа включен длинный набор правил относительно стиля. При использовании общей таблицы стилей для группы документов снижение производительности произойдет только для первого документа, а для остальных документов таблицы стилей уже будет находиться в локальном кэше.

CSS при отображении страницы может быть взята из различных источников.

Авторские стили (информация стилей, предоставляемая автором страницы) в виде:

- внешних таблиц стилей, т. е. отдельного файла .css, на который делается ссылка в документе;

- встроенных стилей — блоков CSS внутри самого HTML-документа;

- Inline-стилей, когда в HTML-документе информация стиля для одного элемента указывается в его атрибуте `style`.

Пользовательские стили:

- локальный CSS-файл, указанный пользователем в настройках браузера, переопределяющий авторские стили, и применяемый ко всем документам;

- стиль браузера;

- стандартный стиль, используемый браузером по умолчанию для представления элементов.

Правила CSS

Таблица стилей состоит из набора правил.

Правило каскадных таблиц стилей состоит из двух частей: селектора и определения. Селектором может быть любой тэг HTML, для которого определение задает, каким образом необходимо его форматировать. Определение располагается в правой части правила. Оно помещается в фигурные скобки. Само определение, в свою очередь, также состоит из двух частей: свойства и его значения, разделенных знаком двоеточия (:). Пример:

```
h1 {color: blue; font-size: 16pt}
```

Назначение свойства очевидно из его названия. В приведенном правиле селектором является элемент h1, а определение, записанное в фигурных скобках, задает значения двух свойств заголовка первого уровня: цвет шрифта (свойство color) определен как синий (значение blue) и размер шрифта (свойство font-size) определен в 16 пунктов (значение 16pt). В одном правиле можно задать несколько определений, разделенных символом, точка с запятой (;).

Правило CSS, может задаваться с помощью атрибута style элемента, к которому оно применяется.

Для уменьшения размеров таблиц стилей можно группировать разные селекторы в виде списка элементов страницы HTML, разделенных запятыми, если для них задается одно правило. Например, следующие правила:

```
h1 {font-family: Arial}
h2 {font-family: Arial}
h3 {font-family: Arial}
```

можно сгруппировать и задать в виде одного правила со списком селекторов:

```
h1,h2,h3 {font-family: Arial}
```

Аналогично группируются определения, только в списке они разделяются точками с запятой (;). Следующие правила форматирования заголовка первого уровня:

```
h1 {font-weight: bold}
h1 {font-size: 14pt}
h1 {font-family: Arial}
```

можно задать в виде одного правила, сгруппировав определения:

```
h1 {font-weight: bold; font-size: 14pt; font-family: Arial}
```

Некоторые свойства имеют собственный синтаксис группирования, связанный с заданием значений нескольких свойств в одном. Например, предыдущий пример при использовании свойства font запишется так:

```
h1 {font: bold 14pt Arial}
```

При задании таблицы стилей можно свободно комбинировать все три правила группирования для уменьшения ее размеров.

Наследование

В HTML некоторые элементы могут содержать другие. Рассмотрим элемент, расположенный внутри другого элемента страницы, если для последнего задано правило форматирования, а для вложенного элемента нет.

Например, пусть цвет шрифта абзаца определен как синий (p {color:blue}). Рассмотрим выделенный элемент текста, задаваемый тэгом , если для него не определено правило форматирования.

В подобных случаях вложенный элемент наследует правила форматирования элемента-родителя.

Наследование полезно при задании значений свойств, применяемых к документу по умолчанию. Для этого достаточно задать все свойства для элемента, порождающего все остальные элементы страницы HTML. Таким элементом является тело документа, определяемое тэгом <body>:

```
body {
color: black;
font-family: "Times New Roman";
background: url(texture.gif) white;
}
```

Селекторы

Правила каскадных таблиц стилей, в которых в качестве селектора используются тэги HTML, влияют на отображение всех элементов заданного типа в документе. Следующее правило отображает без подчеркивания все ссылки (тэг <a>) в документе:

```
<style type="text/css">
    a { text-decoration: none; }
</style>
```

Если необходимо некоторые ссылки отобразить по-другому, то можно задать для них правило форматирования непосредственно в тэге, а можно применить параметр class, добавленный в HTML 4.0 в качестве стандарта для всех тэгов. Значением параметра class является ссылка на класс, задаваемый в таблице стилей.

Селектор class

В одной таблице стилей можно создать два и более различных классов одинаковых элементов с помощью селектора класса. К этим классам элементов можно затем будет применить различные таблицы стилей.

Пример. Необходимо создать страницу, на которой будет два вида абзацев <p>, причем оба вида будут постоянно чередоваться и часто

повторяться. Типичный пример такой страницы – интервью, в котором чередуются вопросы журналиста и ответы интервьюируемого. При создании такой страницы необходимо визуально отделить вопросы и ответы друг от друга.

```
<head>
<title>Интервью</title>
<style>
p.ask {
font-style: italic;
font-weight: bold;
font-family: Arial, sans-serif;
font-size: 10pt;
color: gray;
margin-left: 15px
}
p.answer {
font-family: 'Times New Roman', serif;
font-size: 12 pt; color: black;
}
</style>
</head>
<body>
<p class="ask">Вопрос</p>
<p class="answer">Ответ</p>
</body>
```

Селектор id

Параметр id, задает уникальное имя элемента, которое используется для ссылок на него в сценариях и таблицах стилей. Параметр id можно применять к любому элементу документа.

Правила таблиц стилей регламентируют использование уникального идентификационного имени элемента в качестве селектора, предваряя его символом #:

```
<head>
<title>Демо</title>
<style type="text/css">
<!-    #par24 { letter-spacing: lem; }
h1#form3 { color: red; background-color: blue}
-->
</style>
</head>
<body>
<p id="par24">Разреженные слова в абзаце</p>
<h1 id="form2">Черный шрифт</h1>
</body>
```

В этом примере абзац идентифицирован именем par24 в параметре id, поэтому к нему применимо правило с селектором #par24. Второе правило в таблице стилей должно применяться к заголовку первого уровня с идентификатором form3. Такого элемента в нашем фрагменте нет, и поэтому заголовок form2 отображается с применением правила по умолчанию.

Пример таблицы стилей:

```
p {
  font-family: "Garamond", serif;
}
h2 {
  font-size: 110 %;
  color: red;
  background: white;
}
.note {
  color: red;
  background: yellow;
  font-weight: bold;
}
p#paragraph1 {
  margin: 0;
}
a:hover {
  text-decoration: none;
}
#news p {
  color: blue;
}
```

Здесь приведено шесть правил с селекторами `p`, `h2`, `.note`, `p#paragraph1`, `a:hover` и `#news p`.

В первых двух правилах HTML-элементам p (абзацу) и h2 (заголовку второго уровня) назначаются стили. Абзацы будут отображаться шрифтом Garamond, или, если такой шрифт недоступен, каким-либо другим шрифтом с засечками («serif»). Заголовок второго уровня будет отображаться красным на белом фоне с увеличенным кеглем.

Связывание HTML с CSS

Чтобы таблица стилей могла воздействовать на внешнее представление документа, браузер должен знать о ее существовании. Для этого ее необходимо связать с HTML-документом. Существует четыре способа связывания документа и таблицы стилей:

1. Связывание — позволяет использовать одну таблицу стилей для форматирования многих страниц HTML.
2. Внедрение — позволяет задавать все правила таблицы стилей непосредственно в самом документе.

3. Импортирование — позволяет встраивать в документ таблицу стилей, расположенную на сервере.

4. Встраивание в тэги документа — позволяет изменять форматирование конкретных элементов страницы.

Связывание таблицы стилей

Связывание позволяет хранить таблицу стилей в отдельном файле и присоединять ее к документам с помощью тэга <link>, задаваемого в разделе <head>:

```
<link rel="stylesheet" type="text/css"
href="mystyles.css"/>
```

В этой строке указывается, что связываемый файл является таблицей стилей (rel="stylesheet"), формат этого файла – .css1 (type="text/css") и находится он в той же директории, что и файл html, либо имеет другой URL-адрес (href="mystyles.css").

Эта строка может быть указана в любом html-файле. Таким образом, единое оформление будет использовано для нескольких страниц.

При этом вся таблица стилей хранится в одном файле (расширение файла должно быть стандартным – .css).

Пример CSS-файла:

```
body {
font-family: 'Times New Roman';
font-size: 12pt;
color: green;
}
h1 {
font-family: Arial;
font-size: 16pt;
color: blue;
font-weight: bold;
}
h2 {
font-family: Helvetica;
font-size: 14pt;
color: yellow;
font-weight: bold;
font-style: italic;
}
```

Тэги <style> и </style> внутри файла таблицы стилей не используются – расширение .css явно указывает браузеру на то, что файл является таблицей стилей. Один такой файл может быть связан сразу с несколькими страницами (или импортирован сразу в несколько страниц).

Внедрение таблицы стилей

Для применения одинакового форматирования к нескольким одинаковым элементам станицы необходимо создать в заголовке страницы (в любом месте

между тегами <head> и </head>) внедренную таблицу стилей, в которой задаются требуемые правила оформления. Для этого создается тег-контейнер таблицы стилей, начинающийся открывающим тегом <style> и заканчивающийся закрывающим тегом </style>. Внутри этого тега-контейнера можно задать любое количество правил CSS, состоящих из селектора (названия тега HTML, к которому будет применяться правило) и его определения (непосредственно набора средств форматирования), заключенного в фигурные скобки.

Пример. Необходимо, чтобы все абзацы на странице выглядели, как в предыдущем примере, все заголовки первого уровня отображались шрифтом Arial синего цвета полужирного начертания размером 16 пунктов, а все заголовки второго уровня – шрифтом Helvetica размером 14 пунктов полужирного курсивного начертания желтого цвета.

```
<head>
...
<style>
<!-body {
font-family: 'Times New Roman';
font-size: 12pt;
color: green;
}
h1 {
font-family: Arial;
font-size: 16pt;
color: blue;
font-weight: bold;
}
h2 {
font-family: Helvetica;
font-size: 14pt;
color: yellow;
font-weight: bold;
font-style: italic;
}
-->
</style>
...
</head>
```

Этот способ связывания CSS и HTML называется внедрением. Его рекомендуется применять в тех случаях, когда необходимо задать какой-либо набор правил форматирования только для одной страницы сайта, а все остальные страницы должны выглядеть по-другому.

Импорт таблицы стилей

Для импортирования файла таблицы стилей (в том числе с другого сервера) необходимо указать в заголовке HTML-файла между тегами <head> и </head> внутри тега-контейнера <style> обращение к файлу таблицы стилей:

```
<head>
...
<style>
...
@import: url(my.css);
...
</style>
...
</head>
```

Помимо адреса импортируемой таблицы стилей, в теге-контейнере <style> можно указать любые правила CSS, которые будут дополнять или корректировать правила, заданные в импортируемой таблице – внедренные правила.

Встраивание CSS в теги документа

CSS позволяют назначить собственный стиль визуального представления любому тегу HTML, в том числе тегу <body>. Если стиль задан для тега <body>, он наследуется всеми элементами (абзацами, заголовками и т. д.), помещенными внутри этого тега-контейнера, в случае отсутствия собственных стилей для этих элементов. Таким образом, нет необходимости прописывать теги <font> и атрибуты color, size и т. п. для каждого абзаца на странице – достаточно задать стиль для тега <body>, и все абзацы на странице будут отображены в соответствии с этим стилем.

Пример. Необходимо, чтобы все абзацы отображались шрифтом Times New Roman размером 12 пунктов зеленого цвета. Для этого следует указать атрибут style тега <body>, присвоив ему соответствующее значение:

```
<body style="font-family: 'Times New Roman'; font-size: 12pt; color: green">
```

В примере используется встраивание стиля непосредственно в тег документа – так называемый inline-стиль. Этот способ связывания CSS с HTML-файлом рекомендуется в единичных случаях – если данный стиль планируется применить только к одному элементу только на одной странице сайта. Если же стиль должен быть применен к нескольким элементам на одной странице или к нескольким страницам сразу, рекомендуются другие способы связывания CSS и HTML.

Приоритеты использования таблиц стилей

Браузер расставляет приоритеты таблиц стилей следующим образом:

1. Встроенные (inline-) стили (встроенные с помощью атрибута style непосредственно в теги документа) – наивысший приоритет. Будут применены браузером в любом случае, даже если возникает конфликт с внедренными или внешними стилями;

2. Внедренные стили (перечисленные в теге-контейнере <style> в заголовке документа) – чуть меньший приоритет, будут применяться во всех случаях, кроме случаев возникновения конфликта с inline-стилями (при возникновении такого конфликта будут применены inline-стили);

3. Импортированные стили (стили внешнего файла .css, связанные с документом с помощью свойства @import в теге-контейнере <style>) будут применяться в тех случаях, когда отсутствуют аналогичные правила CSS среди встроенных и внедренных стилей;

4. Связанные стили (стили, присоединенные к html-файлу посредством тега <link>) – наименьший приоритет, будут применены только после того, как браузер убедится в отсутствии аналогичных правил во всех остальных типах стилей.

Контрольные вопросы для самопроверки

1. Для чего используется технология CSS?
2. Начиная с какой версии HTML была реализована технология CSS?
3. Какую смысловую нагрузку несёт термин "каскадные" в технологии CSS?
4. Какие преимущества обеспечивает технология CSS?
5. Из каких источников может быть взята таблица CSS при отображении WEB-страницы браузером?
6. Как задаётся правило CSS?
7. Как задаётся наследование в CSS?
8. Как можно задать селекторы?
9. Как можно связать HTML-документ с таблицей стилей?
10. Приоритеты использования таблиц стилей.

Задание к лабораторной работе

Студент получает типовые задания на выполнение работы.

Отредактировать HTML-страницу, созданную в лабораторной работе №1. Исключить из HTML-документа все теги и атрибуты, связанные с форматированием страницы. Разработать CSS-файл для документа, куда поместить всё форматирование. Использовать классы, идентификаторы и следующие правила CSS:

Параметры текста: Color Direction, Letter-spacing, Text-align, Text-decoration, Text-indent, Text-transform, White-space, Word-spacing.

Параметры фона: Background, Background-attachment, Background-color, Background-image, Background-position, Background-repeat.

Параметры границ: Border, Border-collapse, Border-color, Border-style, Border-width, Border-spacing.

Параметры шрифта: Font, Font-size, Font-family, Font-style, Font-variant, Font-weight.

Параметры списков: List-style, List-style-image, List-style-position, List-style-type.

Методические указания и порядок выполнения работы

Форматирование текста в HTML на основе тегов загромождает исходный код, усложняет его и повышает вероятность появления ошибок в нем. Избежать этого позволяет применение языка для стилевой разметки — CSS.

CSS является формальным языком разметки стилей. Стили – это набор свойств, которые управляют видом и положением элементов веб-страницы.

CSS — это язык описания внешнего вида документа, то есть он отвечает за то, как выглядят веб-страницы: цвет фона и декоративных элементов, размер и стиль шрифтов и ничего не представляет сам по себе, если не взаимодействует с другим языком разметки — HTML, который отвечает за размещение элементов на странице. При этом CSS выбирает элемент HTML (например, абзац), задаёт свойство для изменения (такое как цвет) и применяет определённое значение (например, красный):

```
p { color: red;}
```

Таблицы называются каскадными, потому что работают по принципу каскада — т. е. правило, прописанное ниже, считается приоритетным. Например, если в нашем примере под значением фонового цвета мы пропишем еще одно значение color: red, то цвет текста будет красным, а не черным.

```
p {
color: black
background: #eeeeee
color: red
}
```

Можно отметить следующие основные цели использования CSS:

- описание внешнего вида документа: цвет фона и декоративных элементов, размер и стиль шрифтов;

- упрощение создания веб-страниц и обслуживание сайтов за счет отделения внешнего вида (стиля представления документов) от содержимого документов (описания логической структуры веб-страницы);

- представление документа в различных стилях или методах вывода, таких как экранное представление, печатное представление, чтение голосом.

Перед выполнением работы ознакомьтесь на примерах с возможностями и назначением способов добавления стилей на веб-страницу. Для этого создайте предлагаемые варианты HTML-документов в редакторе кодов, например Notepad++. Последовательно в браузере откройте страницы без дизайна, только HTML-код и тот же самый документ, но уже с добавлением стилей. Сравните варианты представления.

Для добавления стилей на веб-страницу существует несколько способов, которые различаются своими возможностями и назначением.

Способ 1 Внутренняя таблица стилей

По своей гибкости и возможностям этот способ имеет ограничения, но часто применяется в ситуациях, когда речь идёт об одной веб-странице.

```
<!DOCTYPE html>
<html>
 <head>
  <meta charset="utf-8">
  <title>Стили</title>
  <link rel="stylesheet"
href="https://fonts.googleapis.com/css?family=Lobster&amp;subset=cyrillic">
  <style>
   h1 {
    font-family: 'Lobster', cursive;
    color: green;
   }
  </style>
 </head>
 <body>
  <h1>Заголовок</h1>
  <p>Текст</p>
 </body>
</html>
```

В данном примере задан стиль элемента <h1>, который затем можно повсеместно использовать на данной веб-странице. Обратите внимание, что мы можем спокойно комбинировать <link> со <style>.

Способ 2. Встроенный стиль

Встроенный стиль является по существу расширением для одиночного элемента, используемого на текущей веб-странице. Для определения стиля элемента к нему добавляется атрибут style, а значением атрибута выступает набор стилевых правил.

```
<!DOCTYPE html>
<html>
 <head>
  <meta charset="utf-8">
  <title>Стили</title>
 </head>
 <body>
  <p style="font-size: 1.2em; font-family: monospace; color: #cd66cc">Пример
текста</p>
 </body>
</html>
```

В данном примере стиль элемента <p> меняется с помощью атрибута

style, в котором через точку с запятой перечисляются стилевые свойства.

Встроенные стили не рекомендуется применять на сайте, поскольку это усложняет редактирование стилей и нарушает принцип разделения кода и оформления.

Способ 3. Внешняя таблица стилей

Стили располагаются в отдельном файле с расширением .css. Для связывания HTML-документа с CSS-файлом применяется элемент <link>. Он располагается внутри <head>.

```
<!DOCTYPE html>
<html>
 <head>
  <meta charset="utf-8">
  <title>Стили</title>
  <link rel="stylesheet"
href="https://fonts.googleapis.com/css?family=Lobster&amp;subset=cyrillic">
  <link rel="stylesheet" href="style.css">
 </head>
 <body>
  <h1>Заголовок</h1>
  <p>Текст</p>
 </body>
</html>
```

Значение атрибута rel у <link> всегда будет stylesheet и остаётся неизменным. В качестве значения href указывается путь к CSS-файлу; путь может быть задан как относительно, так и абсолютно. Таким образом можно подключать таблицу стилей, которая находится на другом сайте. В примере подключается кириллический шрифт Lobster с сайта Google Fonts.

Содержимое файла style.css:

```
h1 {
  font-family: 'Lobster', cursive;
  color: green;
}
```

Файл со стилем является обычным текстовым файлом и содержит только синтаксис CSS. HTML-документ содержит только указатель на файл со стилем, что реализует принцип разделения кода и оформления сайта. Использование внешней таблицы стилей — наиболее универсальный и удобный метод добавления стиля на сайт. Это позволяет редактировать файлы HTML и CSS независимо друг от друга.

Еще один пример использования внешней таблицы стилей.

Исходный код веб-страницы языке HTML и ее внешний вид:

```
<!DOCTYPE html>
<html>
 <head>
  <title>Высшее образование</title>
  <meta charset="utf-8">
  <link rel="stylesheet" href="/example/css/style.css">
 </head>
 <body>
  <h1>Университет</h1>
  <p> Калининградский государственный технический университет<br>
  Полное наименование образовательной организации:<br>
  Федеральное государственное бюджетное образовательное учреждение высшего
образования<br>
  «Калининградский государственный технический университет»<br>
 Сокращенное наименование образовательной организации:<br>
  ФГБОУ ВО «Калининградский государственный технический университет», ФГБОУ ВО
«КГТУ», КГТУ<br>
 Дата создания образовательной организации: 01.09.1959<br>
 Место нахождения образовательной организации (юридический и почтовый адрес):<br>
  236022, Северо-Западный федеральный округ, Калининградская обл., г. Калининград,
Советский проспект, д. 1.</p>
 </body>
</html>
```

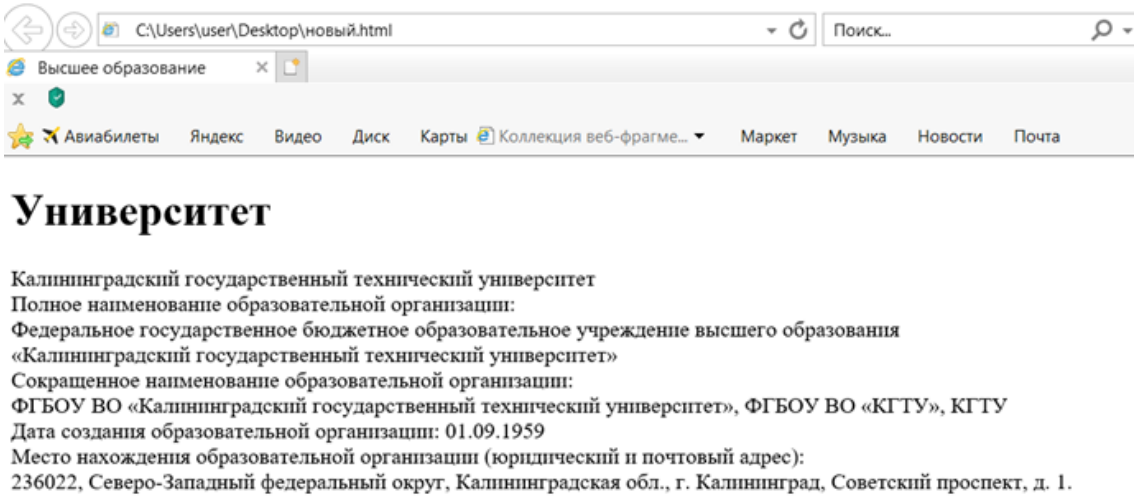

Рис. 2.1. Веб-страница, созданная на HTML

Это обычная веб-страница без дизайна, только чистый HTML.

Тот же самый документ, но уже с добавлением стилей выглядит совершенно иначе. Сам код HTML никаких изменений не претерпел и единственное добавление — следующая строка:

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

Она ссылается на внешний стилевой файл с именем style.css, который, в данном случае, необходимо поместить в одну папку с файлом .html.

```
body {
  font-family: Arial, Verdana, sans-serif; /* Семейство шрифтов */
  font-size: 11pt; /* Размер основного шрифта в пунктах  */
  background-color: #f0f0f0; /* Цвет фона веб-страницы */
  color: #333; /* Цвет основного текста */
}
h1 {
  color: #a52a2a; /* Цвет заголовка */
  font-size: 24pt; /* Размер шрифта в пунктах */
  font-family: Georgia, Times, serif; /* Семейство шрифтов */
  font-weight: normal; /* Нормальное начертание текста */
}
p {
  text-align: justify; /* Выравнивание по ширине */
  margin-left: 60px; /* Отступ слева в пикселях */
  margin-right: 10px; /* Отступ справа в пикселях */
  border-left: 1px solid #999; /* Параметры линии слева */
  border-bottom: 1px solid #999; /* Параметры линии снизу */
  padding-left: 10px; /* Расстояние от линии слева до текста */
  padding-bottom: 10px; /* Расстояние от линии снизу до текста */
```

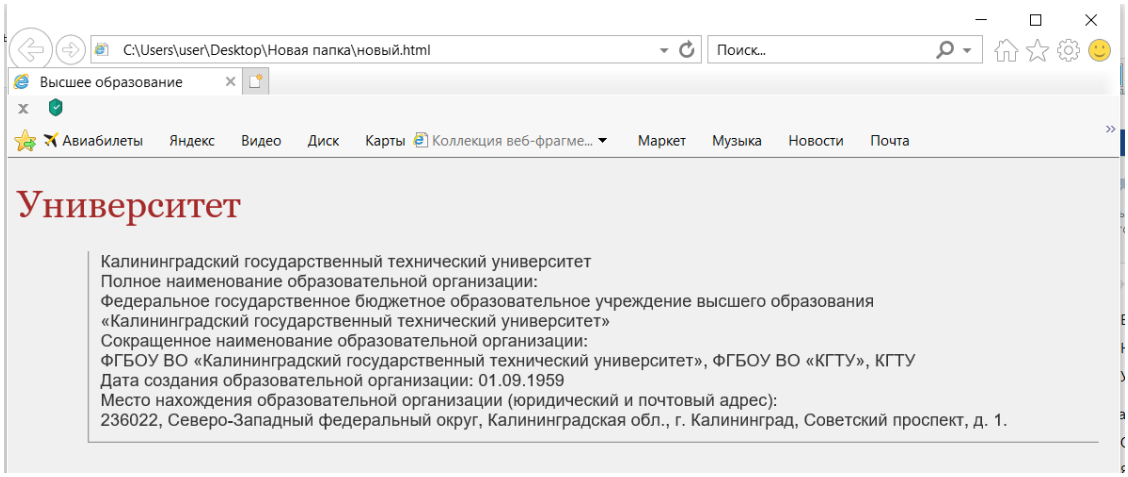

Рис. 2.2. Веб-страница, оформленная с помощью CSS

Все описанные способы добавления CSS могут применяться как самостоятельно, так и в сочетании друг с другом. В этом случае необходимо помнить об их иерархии. Первым имеет приоритет встроенный стиль, затем внутренняя таблица стилей и в последнюю очередь внешняя таблица стилей. В следующем примере используются одновременно два метода добавления стиля в документ.

```
<!DOCTYPE html>
<html>
 <head>
  <meta charset="utf-8">
  <title>Подключение стиля</title>
  <style>
   h1 {
    font-size: 1.2em;
    font-family: Arial, Helvetica, sans-serif;
    color: green;
   }
  </style>
 </head>
 <body>
  <h1 style="font-size: 36px; font-family: Times, serif; color: red">Заголовок 1</h1>
  <h1>Заголовок 2</h1>
 </body>
</html>
```

В данном примере для первого заголовка задан красный цвет и размер 36 пикселей с помощью атрибута style, для второго заголовка — зелёный цвет через элемент <style>.

С подробной информацией о CSS и веб-разработке можно познакомиться в следующих источниках Интернет:

1. Основы CSS [Электронный ресурс]: - Режим доступа: https://html5book.ru/osnovy-css/ (дата обращения 11.11.2022).

2. Основы современной верстки. Основы CSS. [Электронный ресурс]: - Режим доступа: https://ru.hexlet.io/courses/layout-designer-basics/lessons/css-intro/theory_unit (дата обращения 11.11.2022).

3. Основы CSS[Электронный ресурс]: - Режим доступа: https://developer.mozilla.org/ru/docs/Learn/Getting_started_with_the_web/CSS_basics (дата обращения 11.11.2022).

4. Основы CSS [Электронный ресурс]: - Режим доступа: https://wordpress.com/ru/support/css-basics/ (дата обращения 11.11.2022).

5. Основы CSS или как сделать свой сайт «самым обаятельным и привлекательным» [Электронный ресурс]: - Режим доступа: https://loftschool.com/blog/posts/osnovy-css-ili-kak-sdelat-svoj-sajt-samym-obayatelnym-i-privlekatelnym/ (дата обращения 11.11.2022).

Требования к отчету и защите

В отчете указываются название, цель работы. Описание выполненных лабораторных заданий, .html и .css файлы разработанной веб-страницы с результатами в виде скриншотов и выводами по каждому заданию.

На защите проверяются приобретенные знания теоретического и практического материала по ответам на контрольные вопросы для

самопроверки.

Лабораторная работа № 3 «Создание Веб-сайта с использованием конструктора сайтов»

Цель работы: приобретение знаний и практических навыков в использовании конструктора для создания и администрирования сайтов.

Материалы, оборудование, программное обеспечение: IBM PC-совместимый персональный компьютер, IBM PC-совместимый персональный компьютер, имеющий доступ в Интернет, последняя версия любого из распространенных браузеров - Google Chrome, Mozilla Firefox, Microsoft Edge, Opera.

Критерии положительной оценки: выполнение типового задания, оформление отчета по работе, ответы на вопросы для самопроверки.

Планируемое время выполнения:

Аудиторное время выполнения (под руководством преподавателя): 6 часов.

Время самостоятельной подготовки: 1 час.

Теоретическое введение

Сайт или веб-приложение можно разрабатывать при помощи одного из двух подходов:

- написать систему «с нуля» с применением языков разметки и программирования;
- использовать платформы для автоматизации создания сайтов.

На сегодняшний день актуальны следующие типы платформ для автоматизации создания сайта: SaaS-решения, CMS и фреймворки.

SaaS (Software as a Service – программное обеспечение как услуга) – это способ распространения программного обеспечения на арендной основе. SaaS-платформы для создания сайтов и веб приложений часто называют «конструкторами сайтов».

Конструктор сайтов – это специальный онлайн-сервис для создания веб-страниц и объединения их в единую структуру – сайт. Все файлы проекта хранятся на удалённом сервере – хостинге системы. Пользователь получает доступ через аккаунт сервиса, в который можно зайти из любого браузера.

Основные преимущества конструкторов сайтов:

- Простота в использовании.

Возможность создания сайта пользователем без опыта или с минимальными навыками. Интуитивно понятный интерфейс конструктора и визуальный редактор для изменения содержимого сайта и его дизайна. Любой конструктор по умолчанию предоставляет: хостинг, набор тематических шаблонов, виджеты, готовые контентные блоки, инструменты SEO/маркетинга

и множество других функциональных возможностей. После выбора сервис под задачу и регистрации на конструкторе можно сразу начинать работать.

- Наборы готовых шаблонов.

Конструкторы содержат в комплекте набор готовых макетов тем оформления под различные тематики и/или типы сайтов и инструменты для настройки и редактирования дизайна: смена структуры блоков и макета, цветов, шрифтов, фонов, добавление разнообразных элементов.

- Отсутствие необходимости в кодинге.

Владение им поможет настроить дизайн или добавить функциональность под индивидуальные требования, но не является ключевым фактором достижения результата.

- Высокая скорость разработки сайтов.

Быстрое создание проектов, особенно типовых.

- Независимость от сторонних разработчиков.

Пользователь сам распоряжается проектом, изначально в курсе всего происходящего с проектом и может самостоятельно решать вопросы.

- Быстрый редизайн.

Для смены оформления веб-ресурса, нужно лишь выбрать один из сотен шаблонов, которые предоставляет любой конструктор сайта.

- Ценовая политика.

В случае платной подписки, предоставляемые тарифные планы отражают возможности, обеспечиваемые за определённую плату. Это позволяет выбрать подходящий под задачу тариф.

- Автоматическая SEO-оптимизация.

Многие конструкторы сайтов предоставляют инструменты, которые упрощают поисковое продвижение созданных веб-ресурсов.

- Отсутствие этапа тестирования – весь функционал и шаблоны, предоставляемые конструкторами сайтов, проходят тестирование еще на этапе разработки. В результате вебмастер получает полностью рабочий сайт без багов.

- Гарантия результата.

Все конструкторы имеют специализацию. Выбирая профильный сервис, можно быть уверенным в достижении результата – там будет всё необходимое.

- Регулярные обновления.

Острая конкуренция в нише конструкторов определяет заинтересованность собственника в качестве своих продуктов. Выпускаемые обновления, которые автоматически внедряются в систему по мере их выпуска, обеспечивают преимущества перед конкурентами.

- Комфорт использования.

Конструкторы легко освоить и удобно использовать. Пользователь ведёте проект и достигает результатов, не отвлекаясь на побочные моменты. У многих конструкторов есть Базы знаний, раздел Помощи с инструкциями, FAQ (вопросы/ответы) и дружественная Техподдержка.

Недостатки конструкторов сайтов:

- Сайты на конструкторах похожи.

Сайты, созданные в одном конструкторе, могут быть похожи между собой. Это объясняется использованием готовых шаблонов и ограниченными возможностями редактора страниц.

- Ограничения при использовании.

Большинство функциональных ограничений связаны с бесплатными тарифными планами, предназначенными, скорее, для ознакомления с интерфейсом и тренировки начальных навыков разработки у новичков. Использование платных тарифов позволяет получить увеличение по дисковому пространству и объему трафика. доступ к дополнительным модулям, лучшим шаблонам и средствам их редактирования, SEO-инструментам.

- Сложности переноса сайта на другой хостинг.

Привязать домен к новому хостингу проблемы можно. Но автоматизированных средств переезда с конструкторов не существует. Большинство конструкторов не дают доступа к базе данных и прочим внутренним элементам. Контент нужно будет переносить вручную, что потребует навыков и дополнительных затрат.

- Ограничения по SEO и ранжирование.

Верстка блоков с точки зрения SEO не совсем корректна. Ранжирование сайта зависит от конструктора, на котором он создан, так как нет возможности детальной доработки сайта, чтобы он смог занимать высокие позиции в поисковой выдаче.

Условия целесообразности использования конструкторов:

- реализация проекта собственными силами;
- отсутствие опыта и минимальные навыки применения языков разметки и программирования;
- ограниченный бюджет, выделенный на разработку;
- разработка не ведется в высококонкурентных нишах;
- необходимость быстрой разработки и обеспечения доступа к сайту.

Кроме того, успех разработки может быть обеспечен безошибочным выбором конструктора, соответствующего задаче.

Наиболее популярные конструкторы сайтов:

Конструктор сайтов uKit

На сегодняшний день uKit – лучший конструктор сайтов для российской аудитории, а особенно – для малого и среднего бизнеса. Конструктор предоставляет большое количество адаптированных шаблонов в 40 различных категориях/тематиках, оптимизированных под поисковое продвижение и удобных для максимально быстрого запуска. При этом редактор поставляется с огромной коллекцией шрифтов и разными цветовыми схемами. Подходит для создания сайта-визитки, лендинга, блога и интернет-магазина.

Конструктор сайтов Nethouse

В конструкторе сделан упор на простоту функционала. В бесплатном тарифном плане сервис предоставляет SSL-сертификат и домен третьего

уровня для размещения сайта и несколько десятков шаблонов с адаптивным дизайном. Подходит для создания сайта-визитки, блога, интернет-магазина и лендинга.

Конструктор сайтов Tilda

Отечественный конструктор сайтов, основанный в 2014 году. Его особенностью является реализация блочной архитектуры построения веб-страниц. Конструктор позволяет развертывать сайты на основе уже заполненного и пустого шаблонов. Подходит для создания лендинга, блога, сайта-визитки и небольшого интернет-магазина.

Конструктор сайтов uCoz

Универсальный многофункциональный конструктор сайтов, который потенциально интересен всем: блогерам, бизнесу, креативщикам всех направлений, фрилансерам, веб-мастерам и просто новичкам, желающим попробовать свои силы в сфере создания сайтов. Подходит для создания сайта-каталога, крупного магазина, блога, форума.

Конструктор сайтов Insales

Платформа для развертывания полномасштабных интернет-магазинов. Предоставляет более сотни готовых модулей для расширения функционала создаваемого магазина. Поддерживает интеграцию со множеством сторонних сервисов и веб-приложений. Подходит для создания сложного интернет-магазина.

Конструктор сайтов Site123

Конструктор сайтов делает упор на простоту его использования. Отличается минималистичным набором инструментом и адаптивным дизайном всех предоставляемых шаблонов. Подходит для создания сайта-визитки, блога, интернет-магазина и лендинга.

Конструктор сайтов Mozello

Отечественный конструктор, в котором администрирование и настройка сайтов осуществляется в визуальном редакторе. При этом создаваемые веб-ресурсы обладают адаптивным дизайном и мультиязычностью. Подходит для создания блога, интернет-магазина, сайта-визитки и лендинга.

Конструктор сайтов Fo.ru

Платформа для создания сайтов любого типа. Предоставляет более 150 шаблонов, рассортированных по тематическим категориям. Кроме этого, клиенты данного сервиса могут приобрести уже готовые к запуску веб-ресурсы, развернутые на базе конструктора Fo.ru. Подходит для создания блога, сайта-визитки, интернет-магазина и лендинга.

Конструктор сайтов Mottor

Конструктор, предлагающий сразу несколько инновационных решений для создания сайтов и повышения эффективности продаж. Прежде всего, это редактор страниц, который позволяет пользователю самостоятельно редактировать CSS-код каждого блока. Можно автоматизировать реакцию сайта на определенные действия клиентов. Подходит для создания интернет-магазина и лендинга.

В целом огромной разницы между конструкторами сайтов нет.

Пользователю стоит самому посмотреть и понять, где более приемлем интерфейс, тарифный план, дизайн, поддержка и пр.

Контрольные вопросы для самопроверки

1. Назовите подходы, с помощью которых можно разрабатывать сайт или веб-приложение?
2. Какие типы платформ можно использовать для автоматизации создания сайта?
3. Каковы основные преимущества и недостатки конструкторов сайтов?
4. Охарактеризуйте целесообразность использования конструктора сайтов.
5. Дайте рекомендации по выбору конструктора сайтов.

Задание к лабораторной работе

Создать сайт резюме студента на основе самостоятельно выбранного конструктора сайтов с бесплатным тарифом.

Методические указания и порядок выполнения работы

Обратите внимание на то, что использование конструктора позволяет заметно облегчить создание сайта. В стандартном плане разработки исключаются этапы:

- выбор CMS. Внутри конструктора уже есть весь необходимый инструментарий и готовые компоненты для разработки;

- программирование. Сборка сайта на конструкторе происходит без программирования. У таких компонентов как кнопки, формы, анимации уже прописана логика их поведения;

- тестирование. Конструкторы сайтов предлагают готовые решения, нет нужды заниматься их тестами. Достаточно оптимизировать контент (сжать картинки, видео, упростить анимации и т.д.).

В стандартном плане разработки упрощаются этапы:

- составление ТЗ. Использование конструктора упрощает ТЗ. Можно не тратить время на прописывание многочисленных языков, фреймворков и инструментов. Вместо этого уделяется внимание более важным вещам (скорость загрузки сайта, структура, адаптивность и т.д.);

- вёрстка. Перенос дизайн-макета на конструктор происходит без использования HTML и CSS. Фактически, вёрсткой может заниматься сам дизайнер;

- публикация в Интернете. На конструкторах процесс запуска сайта заметно автоматизирован и упрощён: зарегистрировать домен, выбрать хостинг и выпустить сертификат можно внутри SaaS-платформы.

С одной стороны, применение конструкторов позволяет уделить больше внимания самым важным этапам разработки: проанализировать цели, провести исследования, посмотреть больше сайтов конкурентов, подготовить качественный контент, продумать прототип до мелочей и т.п. С другой,-

конструкторы сайтов — это самый простой и в то же время наименее гибкий способ разработки.

При выборе конструктора для создания сайта учитывайте следующие критерии:

- внимательно изучите лимиты, которые конструктор сайтов устанавливает на количество страниц, объем передаваемого трафика, дисковое пространство и т.д.;
- наличие большой коллекции шаблонов и возможности их редактирования позволит добиться уникальности дизайна создаваемого сайта;
- доступность расширения функционала сайта с помощью модулей;
- высокая скорость загрузки – конструкторы создают динамические сайты, страницы которых формируются в режиме реального времени и загружаются из разных источников. Поэтому важно, чтобы формирование происходило как можно быстрее;
- наличие многоканальной круглосуточной службы поддержки – позволяет оперативно решать возникшие проблемы;
- простотой и понятный функционал – гарантируют быстрое создание сайта.

Перечень рекомендованных для выполнения работы конструкторов:

Tilda (регистрация https://tilda.cc/ru/)

Бесплатный тариф с рекламой конструктора, ограничениями по количеству сайтов, серверному пространству и количеству страниц.

uCoz (регистрация https://www.ucoz.ru/)

Бесплатный период: Неограниченный.

uKit (регистрация https://ukit.com/ru)

14 дней бесплатного пробного периода. После завершения 14-дневного пробного периода сайт будет закрыт для посетителей до оплаты одного из тарифов.

Nethouse (регистрация https:// https://nethouse.ru/)

Бесплатный пробный период с рядом функциональных ограничений.

Кроме рекомендованных для выполнения лабораторной работы конструкторов сайтов можно выбрать любой другой доступный для регистрации.

После выбора конструктора перейдите к разработке сайта резюме, используя информационные материалы из лабораторных работ № 1 и 2.

Все конструкторы сайтов имеют оригинальные пошаговые инструкции по созданию сайта. Как правило, все они предлагают последовательное выполнение следующих основных этапов:

1. Регистрация.
2. Выбор шаблона.
3. Редактирование шаблона.
4. Наполнение.
5. Редактирование мобильной версии (выборочно).
6. Публикация (выбор доменного имени).

С подробной информацией о CSS и веб-разработке можно познакомиться в следующих источниках Интернет:

1. Создать сайт бесплатно. Конструктор сайтов Tilda Publishing [Электронный ресурс]: - Режим доступа: https://tilda.cc/ru/ (дата обращения 11.11.2022).
2. uKit — Конструктор сайтов для бизнеса [Электронный ресурс]: - Режим доступа: https://ukit.com (дата обращения 11.11.2022).
3. 14 лучших конструкторов или как сделать сайт бесплатно [Электронный ресурс]: - Режим доступа: https://kokoc.com/blog/konstruktor-sajtov/ (дата обращения 11.11.2022).
4. Бесплатный конструктор сайтов uCoz [Электронный ресурс]: - Режим доступа: https://www.ucoz.ru/ (дата обращения 11.11.2022).
5. Конструктор сайтов и его преимущества [Электронный ресурс]: - Режим доступа: https://loftschool.com/blog/posts/osnovy-css-ili-kak-sdelat-svoj-sajt-samym-obayatelnym-i-privlekatelnym/ (дата обращения 11.11.2022).
6. Конструкторы сайтов. Что это такое и как их выбирать [Электронный ресурс]: - Режим доступа: https://trinion.org/blog/konstruktory-saytovchto-eto-takoe-i-kak-ikh-vybirat (дата обращения 11.11.2022).

Требования к отчету и защите

В отчете указываются название, цель работы. Описание выполненных лабораторных заданий с результатами в виде скриншотов и выводами по каждому заданию.

На защите проверяются приобретенные знания теоретического и практического материала по ответам на контрольные вопросы для самопроверки.

Лабораторная работа № 4 «Создание и управление блогами»

Цель работы: ознакомиться с особенностью и разновидностями блогов, изучить методы создания и управления блогами и научиться создавать блог на блог-платформе Blogger.

Материалы, оборудование, программное обеспечение: IBM PC-совместимый персональный компьютер, IBM PC-совместимый персональный компьютер, имеющий доступ в Интернет, последняя версия любого из распространенных браузеров - Google Chrome, Mozilla Firefox, Microsoft Edge, Opera.

Критерии положительной оценки: выполнение типового задания, оформление отчета по работе, ответы на вопросы для самопроверки.

Планируемое время выполнения:

Аудиторное время выполнения (под руководством преподавателя): 6 часов.

Время самостоятельной подготовки: 1 час.

Теоретическое введение

Блог (англ. blog, от web log — интернет-журнал событий, интернет-дневник, онлайн-дневник) — веб-сайт, основное содержимое которого — регулярно добавляемые пользователем, ведущим блог (блогером), записи, содержащие текст, изображения или мультимедиа. Для блогов характерна возможность публикации отзывов (комментариев) посетителями; она делает блоги средой сетевого общения, имеющей ряд преимуществ перед электронной почтой, группами новостей и чатами. Информационный блок, размещённый пользователем в блоге, микроблоге, форуме, социальной сети и пр. – пост. Совокупность всех блогов сети принято называть блогосферой.

Особенность блогов заключается в структуре записей и в простоте их добавления. После обращения к веб-серверу и идентификации пользователь может добавлять новые записи к своей коллекции, структура которой напоминает последовательную структуру дневника или журнала с отображением информации в порядке обратной хронологии. Принцип, который действует: последний пост – станет первым.

Чтение блогов и авторство — два разных по содержанию процесса. Цели, чтения:

- получение информации;
- чтение-развлечение;
- отслеживание реакции публики на те или иные действия;
- чтение ради социализации, ощущения себя причастным к жизни известных людей.

Разновидности блогов:

1. По авторству:

- личный (персональный, авторский, частный) блог — ведётся одним лицом (как правило, его владельцем);
- коллективный или социальный блог — ведётся группой лиц по правилам, определяемым владельцем и модераторами;
- корпоративный блог — ведётся сотрудниками одной организации.

2. По тематической направленности.

3. По особенностям контента:

- контентный блог — блог, публикующий первичный авторский контент;
- мониторинговый (ссылочный) блог — блог, основным контентом которого являются откомментированные ссылки на другие сайты;
-цитатный блог — блог, основным контентом которого являются цитаты из других блогов;

- влог – это видеоблог, когда публикуются записи, созданные и отредактированные заранее;

- стриминг – это показ того, что вы делаете, в режиме реального времени.

- текстовый блог -в чистом виде встречается редко;

- аудиоблог;

- микроблог (немного текста + картинка или видео);

- тамблелог, Тамбллог, Тлог — блог, в котором запись может быть только определённого формата. Например, цитата, видео, ссылка, песня, разговор и так далее. Тамблелог — скорее не система типа дневника, а черновик или записная книжка;

- сплог — спам-блог;

- флоги или фейковые блоги. Наряду с обычными блогами существуют так называемые флоги (фейковые блоги). Во флогах публикуются оплаченные записи с рекламным содержанием, которые замаскированы под личные впечатления.

4. По технической основе:

- автономный блог — блог на отдельном хостинге и системе управления содержимым;

- блог на блог-платформе — блог, ведущийся на мощностях блог-службы (Живой Журнал, LiveInternet.ru, Blogger и др.).

5. По доменному имени:

- доменное имя второго уровня. Пример: domain.ru

- доменное имя третьего уровня (или дальше). Пример: subdomain.domain.ru.

В зависимости от уровня предоставляемого сервиса блог-платформы можно условно разделить на три группы:

- профессиональные: пользователю предоставляется индивидуальный движок блога, собранный (включая необходимые плагины) и настроенный согласно запросам пользователя. Доступа к коду движка пользователь, как правило, не имеет. Кроме того, предоставляется хостинг для файлов и ограниченная возможность запуска своих скриптов (или их подключения из готового перечня);

- полупрофессиональные: пользователю предоставляется возможность аренды движка (нередко возможен выбор одного из нескольких движков). Возможностей индивидуальной настройки нет. Для хранения файлов предоставляется хостинг;

- массовые: пользователю предоставляется учётная запись и аренда ресурсов сервера. Прямого доступа к данным у пользователя нет, он может использовать только штатные средства движка.

Профессиональные и полупрофессиональные блог-платформы обычно платные, поскольку используют модель предоставления хостинга, адаптированного для ведения блога, плюс аренда приложения и обслуживание. Массовые блог-платформы редко бывают платными, поскольку предоставляют, по сути, не хостинг, а массовый веб-сервис. Из-за этого в профессиональных блог-платформах социальных связей между пользователями меньше, но они более таргетированные (целевые).

Методы управления блогом (сайтом):

1. Регистрация на сервисах и поддержка блога без возможности управления сайтом;

2. Полное регулирование работы ресурса с использованием платного хостинга.

В первом случае вначале предлагается бесплатная регистрация на сервисе. Но при развитии блога возникают проблемы:

- домен третьего уровня затрудняет продвижение блога в поиске;

- домен не является вашей собственностью автора блога;

- отсутствие возможности бесплатного добавления функционала.

Такими сервисами являются специализированные блог-платформы:

Яндекс Дзен, Blogger, WIX, LiveJournal и др.

Второй метод обеспечивает полное управление блогом (сайтом). Существует два пути реализации метода:

- Регистрация бесплатного хостинга и создание блога без вложения денежных средств.

Пользователь получает настроенный сервер и начальные инструменты для создания блога. Как правило, бесплатные хостинги служат для ознакомления и тестовых испытаний.

- Приобретение платного хостинга. Такой подход предпочтительнее, так как пользователь получает арендованное выделенное место на сервере, защищенное, с гарантией безотказной и быстрой работы блога.

Платформы, требующие наличие веб-хостинга и доменного имени:

Wordpress, Joomla, Drupal и др.

Параметры оценки платформ для создания блога:

- соотношение цены и качества — затраты на сервис должны соответствовать полученным ресурсам;

- простота и легкость настроек — возможность использования полного функционала платформы без программирования;

- удобный редактор текстовых и визуальных материалов;

- возможность эффективного SEO-продвижения;

- гибкость сервиса для оформления блога и добавления функций.

Контрольные вопросы для самопроверки

1. Каковы основные функции блогов?
2. Какие различия между блогом и информационном сайтом?
3. Перечислите основные этапы создания блога.
4. Укажите особенности выбранной платформы для создания блога.
5. Что такое монетизация блога и способы ее реализации?

Задание к лабораторной работе

Студент получает типовые задания на выполнение работы.

Разработать блог, содержащий сведения об авторе, выбрать шаблон, стиль оформления, написать небольшие статьи и опубликовать свои посты. Обменяться ссылками на блог с одногруппниками и написать комментарии к их постам.

Методические указания и порядок выполнения работы

Рекомендуется выполнить работу с использованием платформы Blogger.

При желании можно самостоятельно выбрать другую платформу.

Особенности платформы Blogger:

- возможность создания до 100 блогов с одного аккаунта;
- можно включить HTTPS и бесплатно подключить собственный домен;
- обеспечивается функциональность для командной работы с несколькими авторами и встроенный поиск по блогу;
- по умолчанию блоги добавляются в каталоги Blogger и индексируются поисковыми системами, в настройках это можно запретить;
- есть своя система комментирования и можно подключить сторонние;
- выбор тем ограничен, но можно сделать внешний вид статей привлекательнее с помощью HTML и CSS;
- блог можно монетизировать.

1. Создайте или войдите в имеющийся аккаунт Google.
2. Зайдите на стартовую страницу Google http://www.google.ru/ в раздел "Приложения Google" (в правом верхнем углу) и найдите в списке приложений – Blogger.

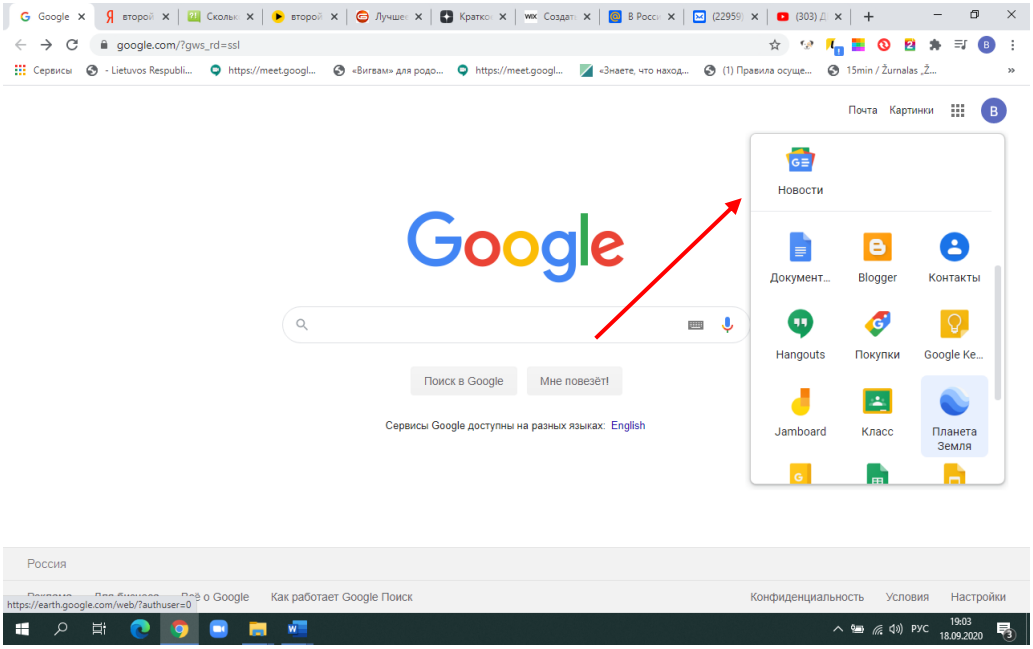

Рис. 4.1. Выбор приложения Blogger

3. В левом вертикальном меню выберите пункт «Политика в отношении содержания», ознакомьтесь с политикой Blogger в отношении содержания и перейдите к пункту «Создать блог».

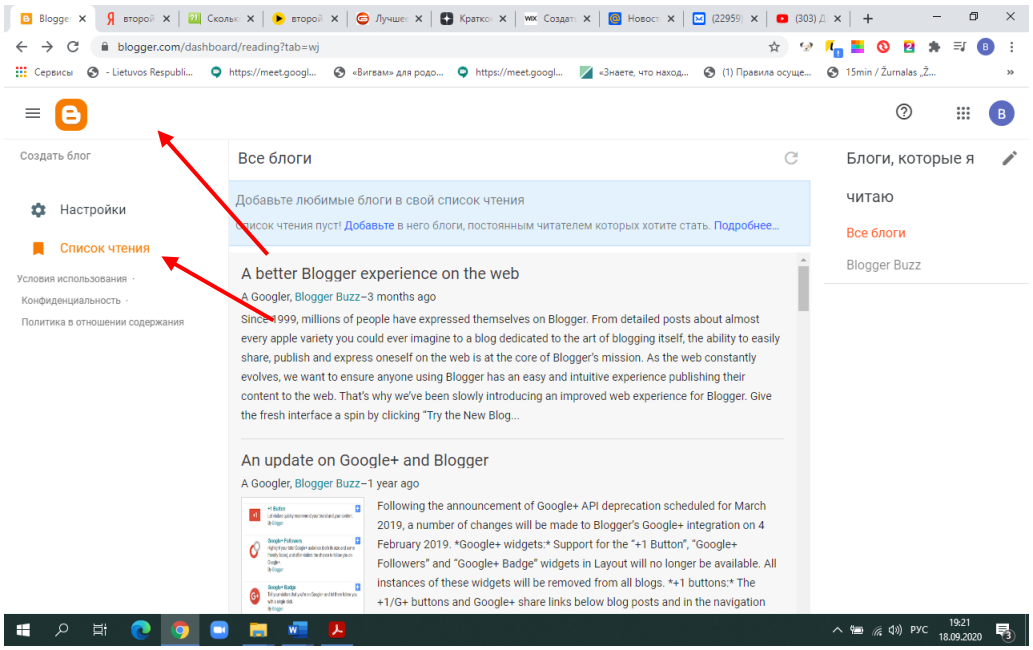

Рис. 4.2. Меню «Создать блог»

4. Выберите название блога. Креативное название поможет выделиться и запомниться читателям.

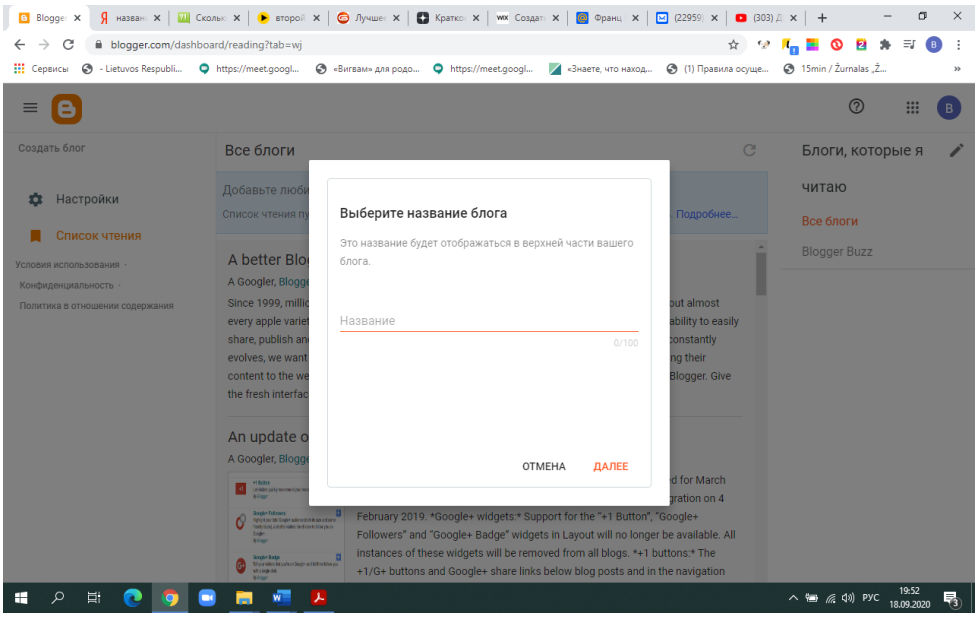

Рис. 4.3. Выбор имени блога

5. Выберите URL для вашего блога и нажмите на кнопку «Сохранить». Желательно задать доменное имя, созвучное названию блога, чтобы облегчить его запоминание и поиск.

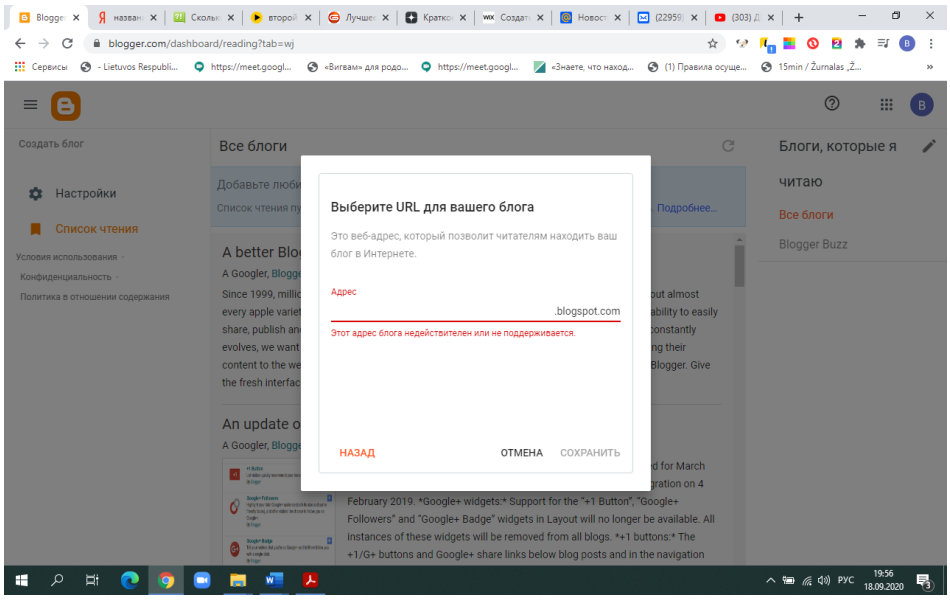

Рис. 4.4. Сохранение блога

6. После создания появится панель управления блогом.

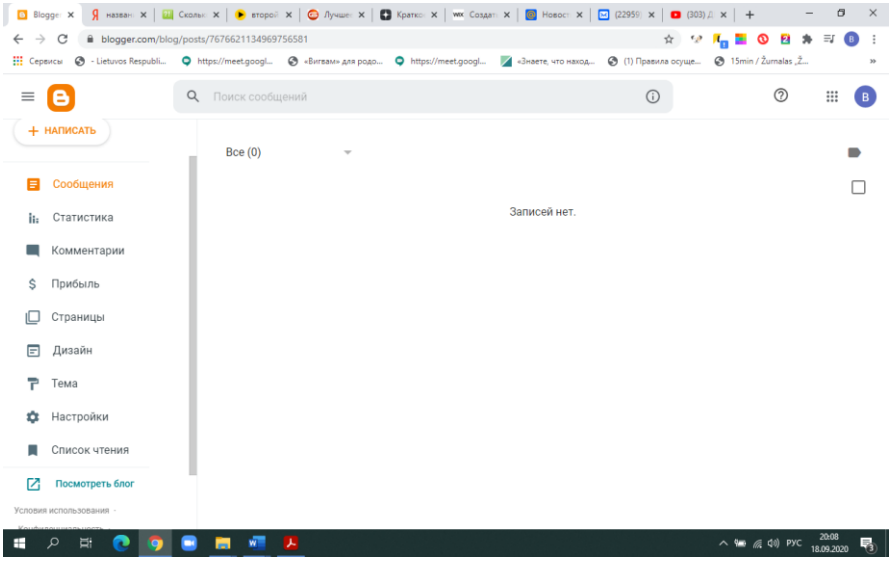

Рис. 4.5. Панель управления блогом

7. Для начальной настройки блога выберите пункт «Дизайн» на панели управления блогом. Добавляйте, удаляйте и настраивайте гаджеты в своем блоге. Перетаскивайте гаджеты, чтобы изменить их расположение и создать шаблон блога.

8. Переходя по пунктам «Тема», «Страницы», «Настройки», настройте блог под собственные задачи. В пункте «Настройки» уточните Основные и Общие настройки, создайте Профиль пользователя.

9. Пункт «Просмотреть блог» отображает полученный результат.

10. Выбрав пункт «+Написать», создаем и публикуем новое сообщение.

11. При необходимости удалить блог выбираем последовательно «Настройки», «Управление блогом», «Удаление блога».

Создавая блог, необходимо четко понимать его предназначение, целевую группу, запросы и потребности аудитории. Помните, что блог – отражение нашего внутреннего мира. Каждый блог индивидуален, как и его автор. Делитесь своими опытом и мыслями и станьте источником ценной информации для посетителей сайта.

С подробной информацией о разработке и управлении блогами можно познакомиться в следующих источниках Интернет:

1. Блог [Электронный ресурс]: - Режим доступа: https://ru.wikipedia.org/wiki/Блог (дата обращения 11.11.2022).

2. Лучшее Руководство По Настройке Своего Домена Blogger [Электронный ресурс]: - Режим доступа: https://tekhnologiya.blogspot.com/2020/02/rukovodstvo-po-nastrojke-svoego-domena-blogger.html (дата обращения 11.11.2022).

3. Краткое руководство по созданию сайта на Blogger [Электронный ресурс]: - Режим доступа https://zen.yandex.ru/media/it_flea/kratkoe-rukovodstvo-po-sozdaniiu-saita-na-blogger-5da9c6cb3d008800ac5cc588 (дата обращения 11.11.2022).

4. Создайте свой блог самостоятельно на платформе Wix [Электронный ресурс]: - Режим доступа: https://ru.wix.com/russianhtml/ru900_blog?yclid=5493342146876701382 (дата обращения 11.11.2022).

5. Социальная платформа для ведения блогов и каналов обо всем на свете [Электронный ресурс]: - Режим доступа: https://viewy.ru/blog/ (дата обращения 11.11.2022).

6. Как зарабатывать на блоге: полный список способов монетизации [Электронный ресурс]: - Режим доступа: https://texterra.ru/blog/kak-zarabatyvat-na-bloge-polnyy-spisok-sposobov-monetizatsii.html (дата обращения 11.11.2022).

Требования к отчету и защите

В отчете указываются название, цель работы. Описание выполненных лабораторных заданий с результатами в виде скриншотов и выводами по каждому заданию.

На защите проверяются приобретенные знания теоретического и практического материала по ответам на контрольные вопросы для самопроверки.

Лабораторная работа № 5 «Установка и настройка локального веб-сервера»

Цель работы: ознакомиться с назначением, функционированием и видами локальных веб-серверов, установкой и настройкой на локальном компьютере.

Материалы, оборудование, программное обеспечение: IBM PC-совместимый персональный компьютер, IBM PC-совместимый персональный компьютер, имеющий доступ в Интернет, последняя версия любого из распространенных браузеров - Google Chrome, Mozilla Firefox, Microsoft Edge, Opera.

Критерии положительной оценки: выполнение типового задания, оформление отчета по работе, ответы на вопросы для самопроверки.

Планируемое время выполнения:

Аудиторное время выполнения (под руководством преподавателя): 6 часов.

Время самостоятельной подготовки: 2 часа.

Теоретическое введение

Веб-сервер — сервер, принимающий HTTP-запросы от клиентов, обычно веб-браузеров, и выдающий им HTTP-ответы, как правило, вместе с HTML-страницей, изображением, файлом, медиа-потоком или другими данными. На этапе создания веб-сайтов и веб-приложений взаимодействие клиента и сервера необходимо для выполнения детального тестирования и подготовки сайта к публичному доступу.

Решение этой задачи возможно двумя способами:

- использования веб-сервера хостинг-провайдера;
- использование веб-сервера на локальном компьютере разработчика.

В первом случае разработка и тестирование сайта может проводиться в реальных условиях при высокой посещаемости, ограничениях, наложенных администрацией реального хостинг-провайдера. Но данный способ имеет следующие недостатки:

- возможная нехватка места на хостинге;
- влияние скорости доступа в Интернет;
- финансовые затраты на платный хостинг и оплату сетевого трафика.

Достоинства второго способа:

- практически равные с хостингом возможности
- отсутствует необходимость в подключении Интернета;
- отсутствие финансовых затрат;
- доступ к ресурсу имеют только лица участвующие в разработке. Это не даёт пользователям и поисковым системам взаимодействовать с недоработанным сайтом;
- возможность установки различных CMS;
- проверка корректной работы сайта, легкое и быстрое тестирование;
- обучение верстке и программированию;
- возможность создания учебного сайта без дальнейшей необходимости выставлять его в Интернет.

Реализуя второй способ, можно использовать два варианта:

- ручную самостоятельную установку отдельно веб-сервера, системы управления базами данных, серверного языка программирования (например, Apache, MySQL, PHP) и настройку их для совместной работы. Но, как правило, для неопытного пользователя это нелегко;
- использование готовых комплектов – локальных веб-серверов. Комплект содержит в себе все необходимые для веб-разработчика программы, легко настраивается и управляется. Сложность заключается в выборе комплекта, так как их существует большое количество и каждый из них обладает своими особенностями.

Локальный сервер – один из самых востребованных инструментов веб-разработчика.

Рассмотрим краткую характеристику популярных локальных серверов.

OpenServer имеет тщательно подобранный набор серверного программного обеспечения, дружелюбный продуманный интерфейс, простую установку, удобное управление добавленными сайтами и отсутствие необходимости долгой настройки. Распространяется OpenServer бесплатно.

Перейти к скачиванию этой программы для Windows можно на официальном сайте https://ospanel.io/.

Denwer - этот веб-сервер прост в установке и практически не занимает места на компьютере. С управлением программой разберется даже начинающий пользователь. Однако проект долгое время не обновлялся, что сказалось на его развитии в дальнейшем. Еще одним недостатком, который может стать решающим для некоторых пользователей, будет отсутствие графического интерфейса, из-за чего приходится все действия выполнять через консоль. Сейчас Denwer можно скачать с официального сайта http://www.denwer.ru/.

WampServer – это веб-сервер, преимущество заключается в простоте установки и нетребовательности к системе, что позволяет нормально взаимодействовать с программой на любом компьютере. Присутствует поддержка всех необходимых компонентов, поэтому с настройкой и запуском локального сервера проблем не возникнет. Распространяется этот инструмент бесплатно https://www.wampserver.com/ru/.

The Uniform Server - Одна из самых легких версий локального сервера, предназначенная для компьютеров под управлением Windows. Это ее главное преимущество, ведь пользователю не придется скачивать огромное количество файлов, которые занимают дисковое пространство. Бесплатная загрузка с официального сайта https://www.uniformserver.com/.

Winginx - веб-сервер, который функционирует в среде NGINX Это комплексный инструмент, поддерживающий разные языки программирования и несколько систем управления базами данных. Winginx доступен на сайте разработчика https://winginx.com/ru/.

Рекомендовать какую-то платформу, как наилучшую для локального сервера – некорректно, все зависит от круга задач, который необходимо решать в той или иной ситуации.

Контрольные вопросы для самопроверки

1. Назовите основное назначение и функции веб-сервера.
2. Охарактеризуйте возможные способы организации взаимодействия клиента и веб-сервера на этапе создания веб-сайтов и веб-приложений.
3. Укажите достоинства и недостатки использования веб-сервера хостинг-провайдера.
4. Каковы основные достоинства использования веб-сервера на локальном компьютере разработчика?
5. Что такое локальный веб-сервер?
6. Перечислите основные этапы установки и настройки локальных веб-серверов.

Задание к лабораторной работе

Произвести выбор, установку и настройку локального веб-сервера для его дальнейшего использования при разработке сайта с помощью CMS.

Методические указания и порядок выполнения работы

После выбора локального веб-сервера для загрузки его последней версии необходимо перейти на сайт разработчика. Там же находится документация по установке и настройке.

Далее рассмотрим порядок установки и настройки веб-сервера на примере Open Server.

Open Server относится к платформам типа WAMP (Windows, Apache, MySQL, PHP), но с очень расширенным функционалом и дополнительным программным обеспечением. Расшифровка аббревиатуры WAMP:

Windows — операционная система, для работы в которой предназначен данный локальный сервер;

Apache — web-сервер, который «поднимается» при запуске программы Open Server;

MySQL — очень популярная система управления базами данных, которая является обязательным условием для работы многих движков сайтов, в том числе таких популярный, как Joomla и WordPress;

PHP — интерпретатор серверного языка программирования, на котором написано множество веб-приложений.

Этот локальный сервер позволяет выбрать в настройках один из нескольких вариантов Apache, PHP, MySQL и других компонентов.

1. Перейдите на сайт разработчика по ссылке https://ospanel.io.

2. Ознакомьтесь с набором серверного программного обеспечения, которое тщательно подобрано и включает базовые модули, приложения, СУБД/NoSQL модули, PHP модули + доп. расширения, специальный пакет программ для быстрого старта. а также невероятно удобную и продуманную управляющую утилиту, которая обладает мощными возможностями по администрированию и настройке всех доступных компонентов.

3. Перейдя в меню «Скачать» можно выбрать вариант загрузки Обычная бесплатная (может продолжаться до 3 часов) или Быстрая загрузка (с оплатой финансовой поддержки для существования и успешного развития проекта.

4. Перейдите в меню «Документация» и ознакомьтесь архитектурой программного комплекса, установкой, настройкой и запуском.

5. Скачанный дистрибутив представляет из себя самораскрывающийся архив. После его запуска будет предложен выбор места распаковки. По умолчанию предлагается «диск С», но можно выбрать другое место (в том числе и внешний носитель, типа флешки или переносного диска).

6. Ярлыков на рабочем столе или в меню кнопки «Пуск» не создается, так как программа портативная. Для запуска сервера перейдите в папку OpenServer на указанном при установке диске и запустите файл Open Server.exe (можно поместить его ярлык на рабочий стол).

7. По окончании установки в трее появится новый значок в виде красного флажка, означающего, что локальный сервер пока еще не запущен.

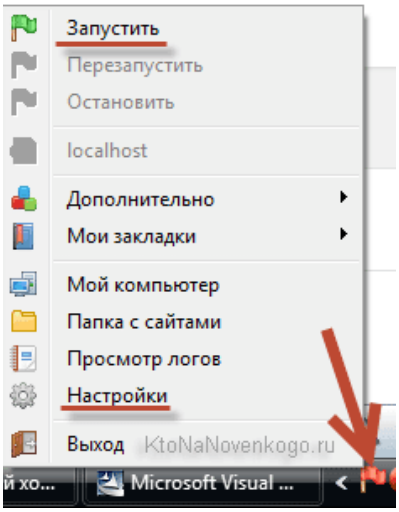

Рис. 5.1. Окно запуска локального сервера

8. Откройте контекстное меню, кликнув любой кнопкой мыши по красному флажку. Это меню является основным инструментом управления OpenServer. Оттуда можно будет запускать веб сервер, останавливать его или перезапускать, а также получить доступ к созданным на его базе сайтам и веб-приложениям (сейчас там только localhost доступен).

9. Проверка работы OpenSever возможна двумя способами: 1) нажмите на зеленый флажок. В открывшемся меню наводим стрелку на пункт Мои сайты. Появится подменю с единственным пунктом localhost. Нажмите на него. Второй: 2) выберите пункт «Запустить» и после перекраски флажка в зеленый цвет вставьте в адресную строку браузера:

http://localhost/

Появление страницы с приветствием означает успешную установку.

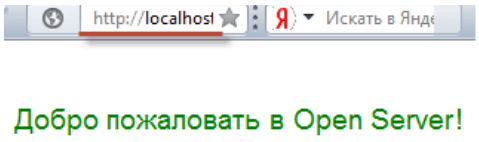

Рис. 5.2. Окно приветствия

10. Выполните настройку программы OpenServer, выбрав пункт «Настройка» в контекстном меню. В открывшемся окне на вкладке «Основные», можно отметить галочкой пункт «Запускать вместе с Windows», установив «Задержку», как и предлагается 20 секунд. Задержка нужна, чтобы не тормозить загрузку Windows. Сначала загрузятся все необходимые для работы компьютера компоненты, а потом запустится OpenServer. (Опция не обязательна).

Также отметьте пункт «Требовать учетную запись администратора». Некоторые функции работают только с правами администратора, поэтому обязательно ставим галочку.

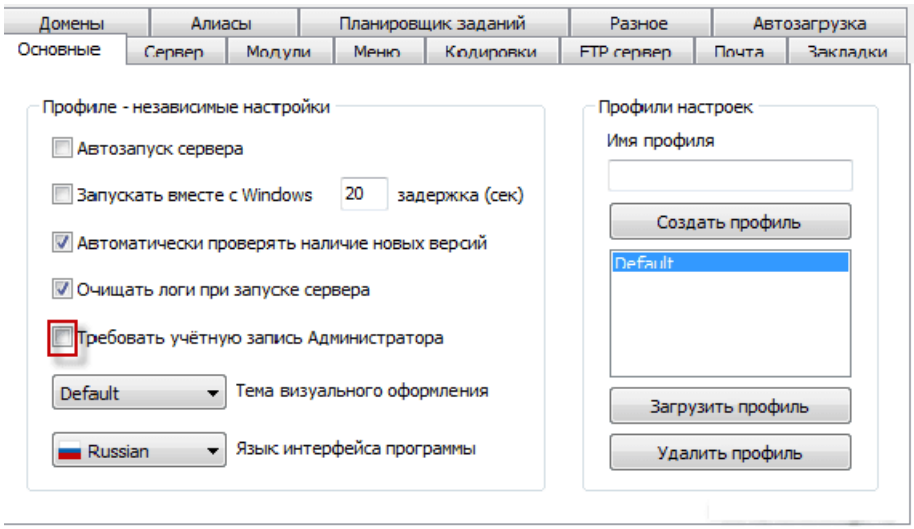

Рис. 5.3. Окно Настройка

На вкладке «Модули» можно выбрать нужные версии Apache, PHP, MySQL. Можно также на вкладке «Меню» поставить галочку в поле «Показывать сайты в главном меню» и назначить браузер, где будут открываться сайты после клика по их названию из контекстного меню.

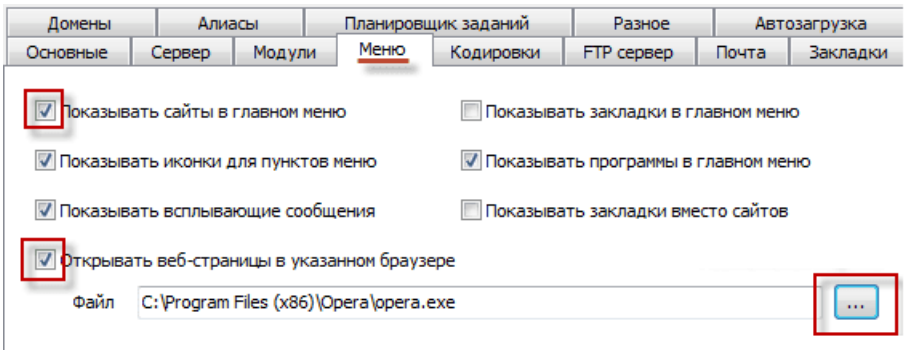

Рис. 5.4. Настройка отображения сайтов в главном меню и открытия их в нужном браузере

11. Для того чтобы начать работу со своим сайтом на данном локальном сервере (или установить движок сайта, т.е. CMS), перейдите из контекстного меню по пункту «Папка с сайтами». Внутри будет находиться папка «localhost». Рядом с ней создайте новую папку для сайта с любым названием, в котором можно использовать следующие символы [a-z0-9.-] (обратите внимание, что нижнее подчеркивание использовать нельзя).

12. Из контекстного меню значка OpenServer в трее выберите пункт «Перезапустить», после чего в списке сайтов появится новое название, кликнув по которому, откроется страница, отображающая сайт в заданном в настройках браузере.

С подробной информацией о установке и настройке локальных веб-серверов можно познакомиться в следующих источниках Интернет:

1. Что такое локальный сервер и для чего он нужен? [Электронный

ресурс]: - Режим доступа: https://handyhost.ru/help/term/chto-takoe-lokalnyij-server-i-dlya-chego-nuzhen.html (дата обращения 11.11.2022).

2. 8 лучших локальных серверов [Электронный ресурс]: - Режим доступа: https://timeweb.com/ru/community/articles/luchshie-lokalnye-servery (дата обращения 11.11.2022).

3. Локальный сервер на компьютере [Электронный ресурс]: - Режим доступа:
https://use-web.ru/news.php?id=37&tid=4 (дата обращения 11.11.2022).

4. Локальный сервер — что это? [Электронный ресурс]: - Режим доступа:
https://itelon.ru/blog/lokalnyy-server/ (дата обращения 11.11.2022).

5. Open Server Panel [Электронный ресурс]: - Режим доступа: https://ospanel.io/ (дата обращения 11.11.2022).

6. OpenServer — современный локальный сервер и пример его использования для установки WordPress на компьютер [Электронный ресурс]: - Режим доступа: https://ktonanovenkogo.ru/wordpress/openserver-lokalnyj-server-ustanovki-wordpress-na-kompyutere.html (дата обращения 11.11.2022).

7. Open Server, установка и настройка для работы [Электронный ресурс]: - Режим доступа: https://www.youtube.com/watch?v=SBTroONAzrU (дата обращения 11.11.2022).

8. Установка и настройка OPEN SERVER | Установка сайта на WordPress [Электронный ресурс]: - Режим доступа: https://www.youtube.com/watch?v=ON7d4DCBBlk (дата обращения 11.11.2022).

Лабораторная работа № 6 «Установка CMS на локальный веб-сервер»

Цель работы: ознакомиться с назначением, функционированием и видами систем управления контентом (CMS), установкой и настройкой на локальном веб-сервере.

Материалы, оборудование, программное обеспечение: IBM PC-совместимый персональный компьютер, IBM PC-совместимый персональный компьютер, имеющий доступ в Интернет, последняя версия любого из распространенных браузеров - Google Chrome, Mozilla Firefox, Microsoft Edge, Opera.

Критерии положительной оценки: выполнение типового задания, оформление отчета по работе, ответы на вопросы для самопроверки.

Планируемое время выполнения:

Аудиторное время выполнения (под руководством преподавателя): 6 часов.

Время самостоятельной подготовки: 2 часа.

Теоретическое введение

CMS (Content Management System — система управления контентом) –

инструмент для тех, кто хочет создать сайт своими руками и не обладает при этом обширными знаниями веб-программирования. По сути, любая CMS представляет собой программную площадку для создания веб-проектов.

Для разработки, настройки и редактирования сайта на основе CMS ее необходимо установить на хостинг: сервер хостинг-провайдера или локальный веб-сервер. Существуют три способа установки CMS-системы на сервер хостинг-провайдера: через встроенный каталог CMS, файловый менеджер и с помощью FTP-клиента.

Установка на сервер хостинг-провайдера в данной работе не рассматривается.

Многие популярные CMS устанавливаются на локальный сервер примерно одинаково:

1. Скачивание CMS с сайта разработчика.
2. Создание в папке с сайтами локального сервера папки с локальным именем домена.
3. Распаковываем в эту папку скачанную CMS.
4. Создание БД.
5. Установка.
6. В завершении установки важно запомнить: логин (может быть email для входа), пароль.

Контрольные вопросы для самопроверки

1. Дайте понятие CMS.
2. Охарактеризуйте возможные способы установки CMS.
3. Каковы способы установки CMS-системы на сервер хостинг-провайдера?
4. Рассмотрите этапы установки произвольной CMS на локальный веб-сервер.
5. Что такое локальный веб-сервер?

Задание к лабораторной работе

Произвести выбор, установку и настройку CMS на локальный веб-сервер.

Методические указания и порядок выполнения работы

После выбора CMS для загрузки ее последней версии необходимо перейти на сайт разработчика. Там же находится документация по установке и настройке.

Далее рассмотрим порядок установки и настройки CMS на примере установки Wordpress на OpenServer.

1. Скачайте с официального сайта https://ru.wordpress.org/download/ последнюю версию Wordpress (на 25.11.2022 это версия Wordpress 6.1.1).

2. Запускаем OpenServer, установка которого произведена в предыдущей лабораторной работе. Кликаем мышкой по красному флажку в трее в нижней правой его части. В открывшемся контекстном меню выбираем пункт

«Запустить». Флажок изменит цвет на желтый, а спустя несколько секунд станет зелёным. OpenServer запущен.

3. Ещё раз откройте контекстное меню нажатием на зеленый флажок в трее. В открывшемся меню выбираем пункт “Папка с сайтами”

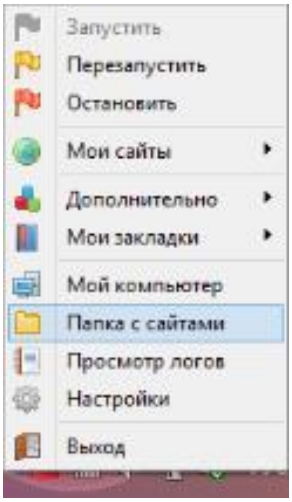

Рис. 6.1. Контекстное меню Openserver

4. Создайте в открывшейся папке domains. новую папку с названием будущего сайта (например mysite).

5. В эту папку распакуйте архив Wordpress. У вас распакуется папка с названием WordPress. Перенесите содержимое этой папки в папку mysite.

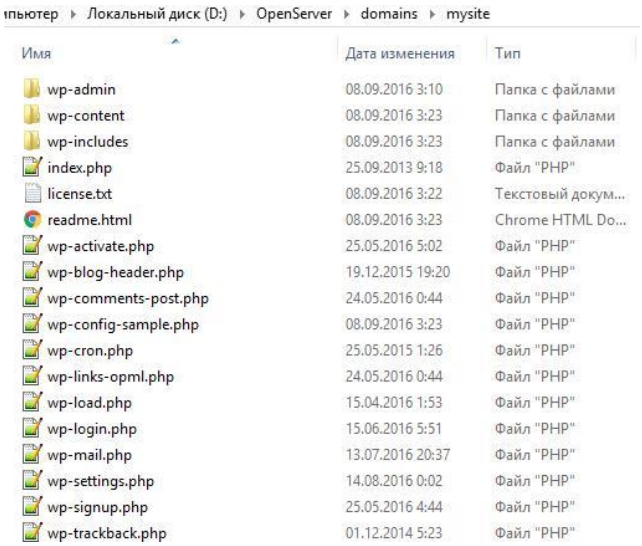

Рис. 6.2. Файлы Wordpress в папке mysite

6. Создайте базу данных. Нажмите на зеленый флажок WordPress. В контекстном меню выбираем пункт «Дополнительно» и в новом меню выбираем пункт «PhpMyAdmin». В браузере откроется программа для создания баз данных.

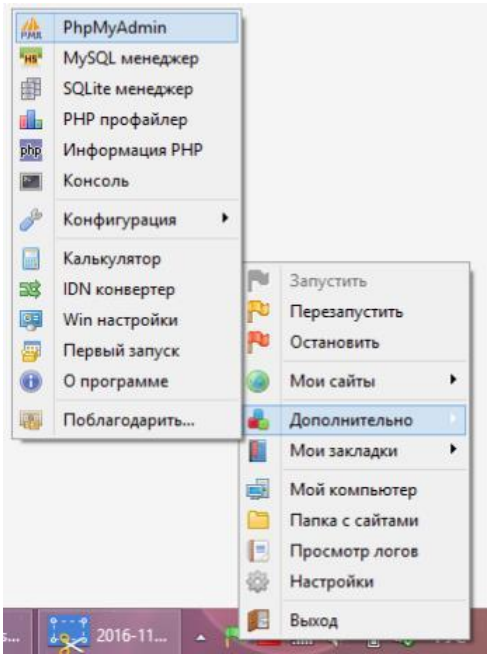

Рис. 6.3. Создание базы данных

7. Настройте вход в PhpMyAdmin, используя следующие данные: пользователь – root, пароль оставляем пустым, нажимаем кнопку ОК.

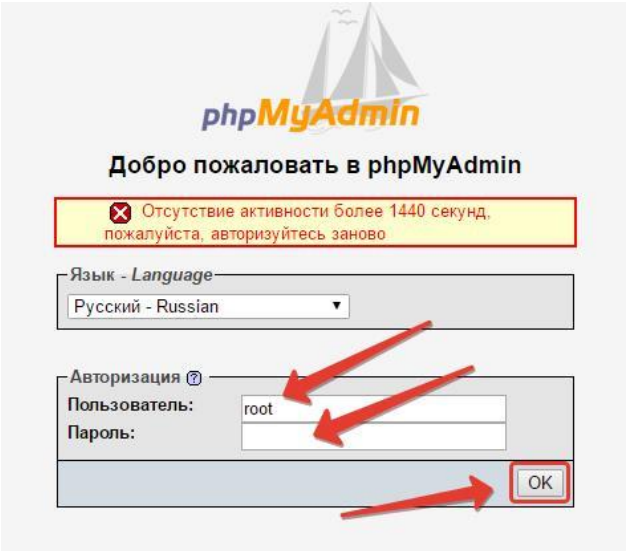

Рис. 6.4. Окно входа

8. В открывшемся окне перейдите на вкладку «Пользователи» и выберите «Добавить пользователя».

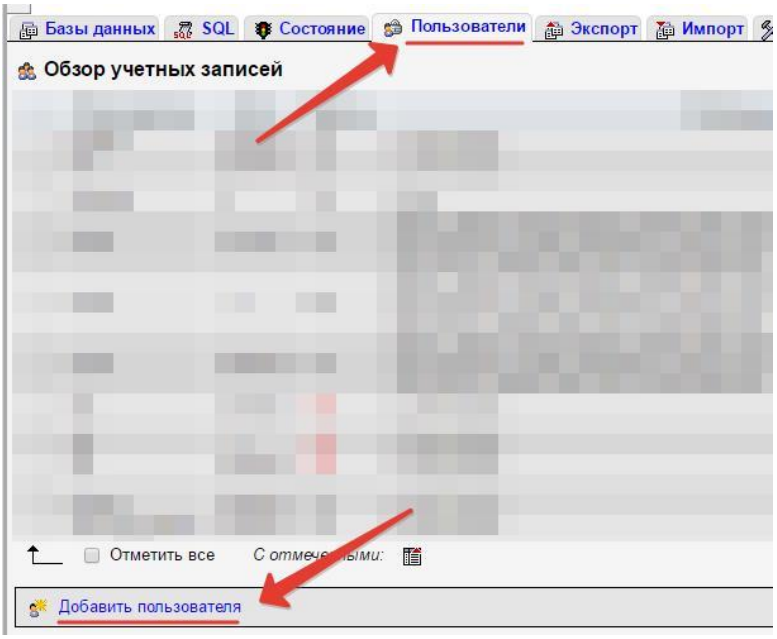

Рис. 6.5. Окно создание пользователя

9. Заполните поля: Имя пользователя (допустим mybaza), задаем и подтверждаем пароль, обязательно ставим галочку «Создать базу данных с именем пользователя», нажимаем кнопку ОК.

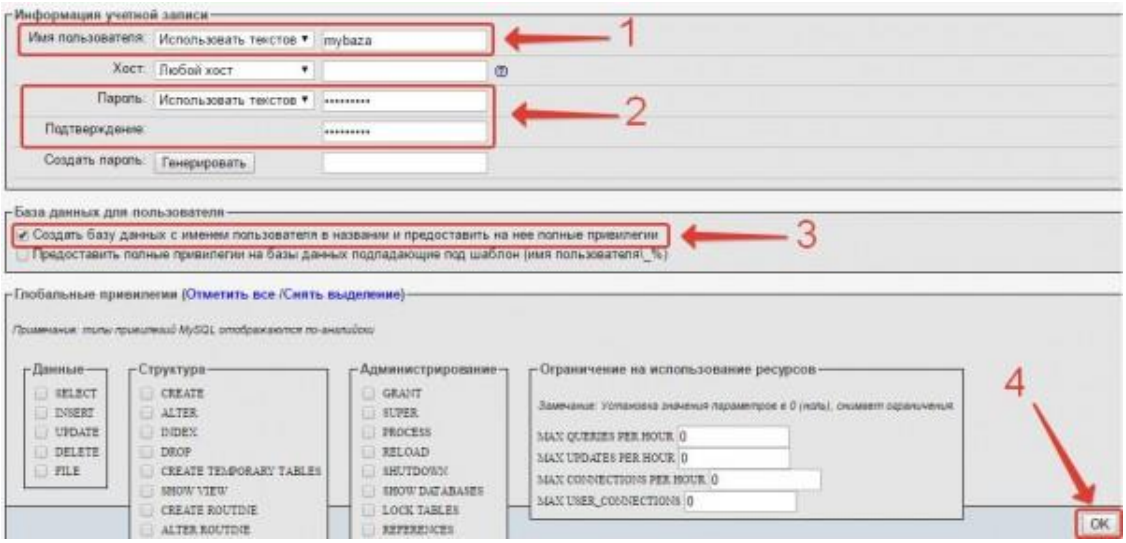

Рис. 6.6. Порядок создания пользователя и базы данных

10. Установка WordPress на Openserver. Нажмите на зеленый флажок Установка WordPress на Openserver. В раскрывшемся контекстном меню выбираем пункт «Мои сайты». В дополнительном меню со списком сайтов (скорее всего там будет два пункта: localhost и созданный mysite) нажмите на созданный сайт. В браузере откроется окно с установщиком WordPress. Нажмите кнопку «Вперед».

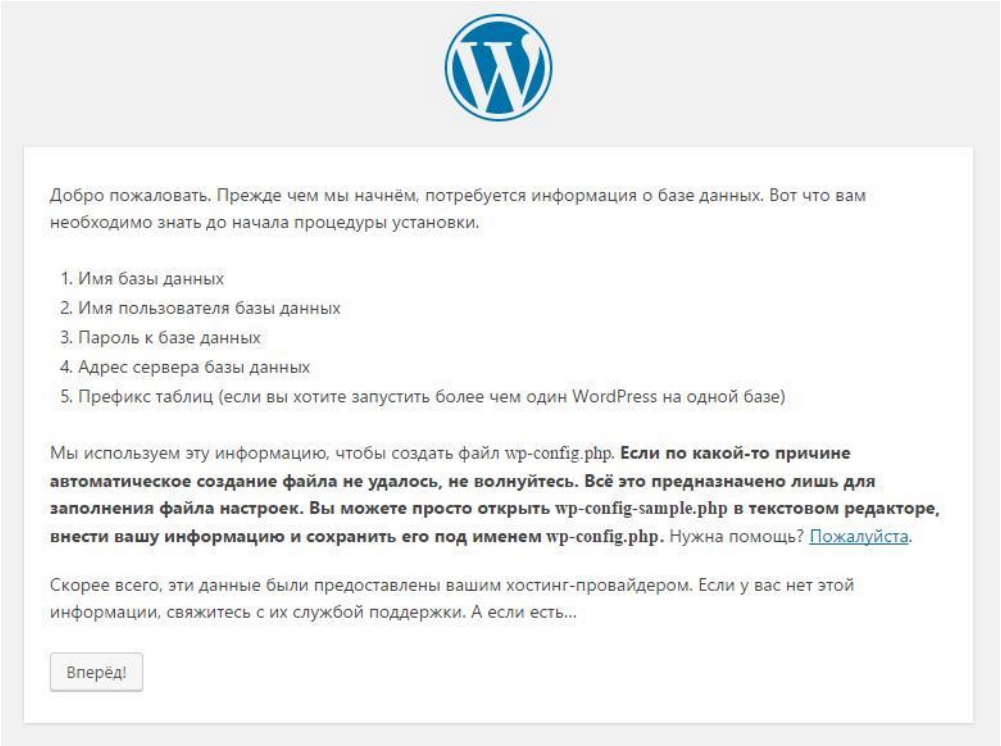

Рис. 6.7. Предустановка WordPress

11. В открывшемся окне заполните данные, которые были указаны при создании базы данных: имя пользователя и имя базы данных одинаковые (ставили галочку – создать базу с таким же именем, как у пользователя), пароль, который указывали там же, при добавлении нового пользователя и БД. Остальное оставляем без изменения. Нажмите кнопку «Отправить».

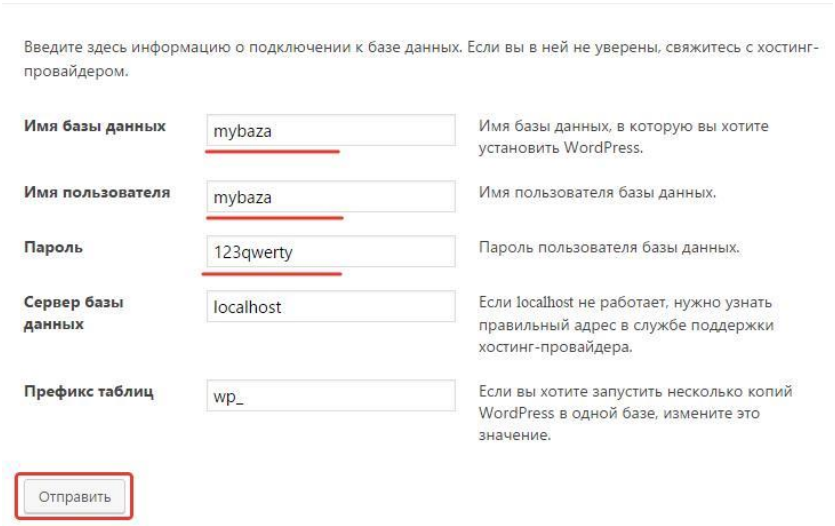

Рис. 6.8. Заполнение сведений о базе данных

12. В окне с сообщением, что все в порядке нажмите «Запустить установку».

13. В новом окне заполните нужные поля: название сайта, имя пользователя (это будет логин для входа в панель администратора сайта).

Запомните! Задайте пароль. WordPress подскажет на сколько он надежен. **Обязательно запомните**! Укажите адрес электронной почты. Нажмите «Установить WordPress».

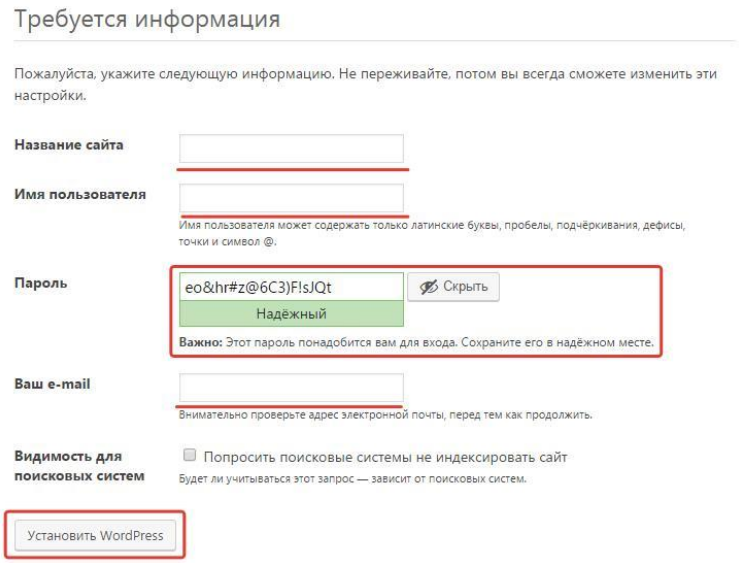

Рис. 6.9. Установка Wordpress на OpenServer

14. В окне появится сообщение с поздравлением об успешной установке!

15. Нажмите «Войти» для авторизации входа в административную панель WordPress. Введите данные, которые запоминали – логин и пароль. Нажмите «Войти».

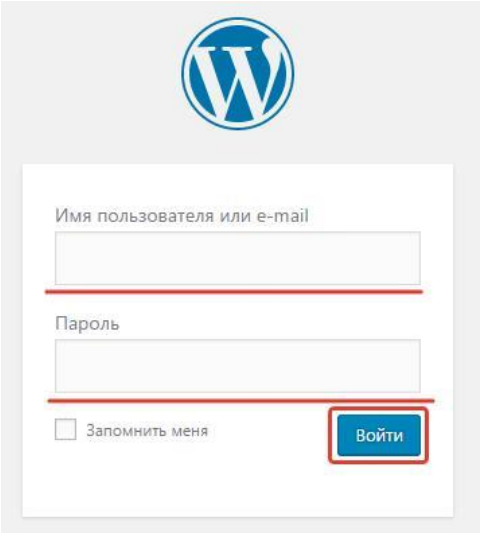

Рис. 6.10. Вход в административную панель Wordpress

С подробной информацией о установке и настройке CMS на локальный веб-сервер можно познакомиться в следующих источниках Интернет:

1. Установка WordPress на OpenServer – полное руководство в картинках [Электронный ресурс]: - Режим доступа: https://1akm.ru/sozdanie-sayta-na-wordpress/ustanovka-wordpress-na-openserver/ (дата обращения 11.11.2022).

2. Установка CMS WordPress на OpenServer: полная пошаговая инструкция от А до Я [Электронный ресурс]: - Режим доступа: https://wpcourses.ru/wordpress-install-openserver/ (дата обращения 11.11.2022).

3. Установка WordPress на OpenServer – полное руководство в картинках [Электронный ресурс]: - Режим доступа: https://1akm.ru/sozdanie-sayta-na-wordpress/ustanovka-wordpress-na-openserver/ (дата обращения 11.11.2022).

5. Автоматическая установка CMS [Электронный ресурс]: - Режим доступа: https://2domains.ru/support/saity/cms-autoinstall (дата обращения 11.11.2022).

6. Установка CMS [Электронный ресурс]: - Режим доступа: https://timeweb.com/ru/help/pages/viewpage.action?pageId=4358538 (дата обращения 11.11.2022).

7. Установка WordPress на Denwer пошаговая инструкция [Электронный ресурс]: - Режим доступа: http://normalnet.ru/wordpress/ustanovka-wordpress-na-denwer.html (дата обращения 11.11.2022).

8. Установка WordPress на локальный сервер Denwer [Электронный ресурс]: - Режим доступа: https://wp-lessons.com/ustanovka-wordpress-na-lokalnyiy-server-denwer (дата обращения 11.11.2022).

9. Youtube Видео: Установка Wordpress на Denwer [Электронный ресурс]: - Режим доступа: https://www.youtube.com/watch?v=yb0cdYI0_4U (дата обращения 11.11.2022).

10. Установка WordPress на локальный веб-сервер OpenServer [Электронный ресурс]: - Режим доступа: https://www.youtube.com/watch?v=fESUgt6yfrE (дата обращения 11.11.2022).

Требования к отчету и защите

В отчете указываются название, цель работы. Описание выполненных лабораторных заданий с результатами в виде скриншотов и выводами по каждому заданию.

На защите проверяются приобретенные знания теоретического и практического материала по ответам на контрольные вопросы для самопроверки.

Лабораторная работа № 7 «Создание сайта при помощи современных CMS»

Цель работы: ознакомиться с назначением, функционированием и видами систем управления контентом (CMS), создать сайт при помощи CMS.

Материалы, оборудование, программное обеспечение: IBM PC-совместимый персональный компьютер, IBM PC-совместимый персональный компьютер, имеющий доступ в Интернет, последняя версия любого из распространенных браузеров - Google Chrome, Mozilla Firefox, Microsoft Edge, Opera.

Критерии положительной оценки: выполнение типового задания, оформление отчета по работе, ответы на вопросы для самопроверки.

Планируемое время выполнения:

Аудиторное время выполнения (под руководством преподавателя): 8 часов.

Время самостоятельной подготовки: 2 часа.

Теоретическое введение

Система управления содержимым (англ. Content management system, CMS, система управления контентом) — информационная система или компьютерная программа, используемая для обеспечения и организации совместного процесса создания, редактирования и управления содержимым, сайта. CMS обычно состоит из двух основных компонентов: приложения для управления контентом (CMA) в качестве внешнего пользовательского интерфейса, позволяющего пользователю добавлять, изменять и удалять контент с веб-сайта без вмешательства веб-мастера, и приложение доставки контента (CDA), которое компилирует контент и обновляет веб-сайт.

Основное назначение CMS:

- предоставление инструментов для создания содержимого, организация совместной работы над содержимым;
- управление содержимым: хранение, контроль версий, соблюдение режима доступа, управление потоком документов;
- публикация содержимого;
- представление информации в виде, удобном для навигации, поиска.

Таким образом, CMS – инструмент для тех, кто хочет создать сайт своими руками и не обладает при этом обширными знаниями веб-программирования.

Сайты, организованные с помощью системы управления контентом, основаны на следующих технологиях: веб-сервер, хранилище данных (СУБД), веб-приложение для обеспечения работы самой системы, визуальный (WYSIWYG) редактор страниц, файловый менеджер с веб-интерфейсом для управления файлами сайта, система управления правами пользователей и редакторов сайта.

Большинство современных CMS обычно состоят из двух частей: back-office – инфраструктурная система, обеспечивающая функциональность и хранение информации, и front-office – часть, которая обеспечивает интерфейс с пользователем. При этом вся функциональность, сложность разработки и администрирования сосредоточены в back-office, а пользовательские свойства - во front-office.

Конструирование внешнего вида страниц сайта реализуется выбором понравившегося дизайн-шаблона из набора заранее заготовленных в CMS. Администратор системы управления контентом может расположить различные информационные блоки в рамках заданной страницы, задать их размеры, цвет

и прочие атрибуты и сохранить состояние страницы, чтобы пользователи сайта видели ее в заданном виде.

Ещё одно преимущество, которое дает CMS, это возможность использования технологии Drag&Drop при управлении содержанием сайта.

Основные функции CMS:

1. Разработка контента. В CMS-системах осуществляется поддержка совместной работы авторов, редакторов, программистов и менеджеров по обновлению содержимого сайта и предоставлению конечному пользователю своевременной информации. Все компоненты сайта, включая шаблоны и наполнение, хранятся в определенных местах хранилища данных;

2. Управление сайтом. На этом уровне происходит разработка самого сайта, предварительный просмотр и публикация подготовленного контента. Здесь разрабатывается внешний вид, подготавливаются шаблоны, распределяются роли пользователей и классификация необходимой информации;

3. Обеспечение доступа пользователей (посетителей, администраторов, редакторов) к информации, содержащейся в базе данных сайта. Система предоставляет средства для динамического формирования Веб-страниц в зависимости от конкретных пользователей. Обычно по умолчанию предлагаются следующие роли пользователей: администратор, редактор, автор, участник, подписчик. Можно добавить нового пользователя и назначить ему свои роли. Каждый пользователь получает только ту информацию, которая соответствует его роли.

Поскольку CMS-системы управляют информацией, а у информации есть свой жизненный цикл, то, естественно, эти системы должны иметь адекватные средства управления контентом на каждом из этапов его жизни (создание, модификация, публикация, передача в архив и т.д.).

Преимущества и недостатки:

- отсутствие необходимости в серьезных технических знаниях, продвинутых навыках программирования и верстки;
- высокая скорость создания сайта;
- удобство администрирования и добавления новых элементов;
- возможность без лишних сложностей создать красивый дизайн;
- простое наполнение, доступное даже людям без специальных знаний.

Единственным недостатком можно назвать сложность создания уникального сайта с нестандартными функциями, но такие ресурсы, как правило, требуются компаниям с соответствующими запросами (к примеру, работающим в IT-сфере). Подобные организации могут позволить себе штат программистов, которые самостоятельно разработают сайт.

В настоящее время существует множество систем управления контентом. От верного выбора CMS зависит успешность будущего сайта, а правильно подобранный функционал движка обеспечит удобное и простое развитие ресурса. Поэтому выбору CMS-системы уделяется особое внимание.

В табл. 7.1 приведены основные сведения о наиболее популярных CMS.

Таблица 7.1

CMS	Блог	Интернет-магазин	Лэндинг	Портал	Лицензия	Примерное время разработки, от дней
WordPress	+	+	+	+	бесплатно	2
1С-Битрикс	+	+	+	-	от 5 400 до 399 000	7
Joomla	+	+	+	+	бесплатно	2
OpenCart	-	+	-	-	бесплатно	30
Drupal	+	+	+	+	бесплатно	7
ModX	+	+	+	+	бесплатно	7

Основные этапы создания веб-сайта с применением практически любой CMS:

1. Установка и обновление версии CMS.
2. Установка шаблона.
3. Общие настройки сайта.
4. Настройка постоянных ссылок.
5. Установка плагинов.
6. Создание страниц, рубрик.
7. Добавление меню.

Контрольные вопросы для самопроверки

1. Дайте понятие CMS.
2. Каковы компоненты CMS?
3. Укажите основные функции CMS.
4. Перечислите основные технологии в составе сайтов, основанных на CMS.
5. Каковы достоинства и недостатки использования CMS?
6. Назначение back-office и front-office CMS.
7. Опишите последовательность создание веб-сайта с применением CMS.

Задание к лабораторной работе

Осуществить выбор CMS и создать веб-сайт при помощи CMS.

Методические указания и порядок выполнения работы

Выбор CMS, ее установка на локальный сервер и настройка рассмотрены в предыдущей лабораторной работе, поэтому можно непосредственно

приступить к разработке веб-сайта с произвольной тематикой и контентом. Рекомендуется создать многостраничный сайт с использованием информационных материалов из лабораторных работ № 1-3.

Далее рассмотрим порядок создания сайта при помощи CMS Wordpress.

Для создания сайта на WordPress нужно выполнить пять стандартных шагов:

1. Зарегистрировать доменное имя и приобрести CMS-хостинг.
2. Установить WordPress.
3. Настроить темы и шаблоны.
4. Добавить контент на готовый сайт.
5. Установить необходимые плагины.

Шаг 1-2: выполнены в прошлой лабораторной работе.

Шаг 3. Настраиваем темы

Темы в WordPress — это шаблоны для внешнего вида сайта. У каждого веб-сайта WordPress есть базовая тема, установленная по умолчанию. Персонализация сайта возможна заменой базовой темы на другую. Чтобы выбрать тему, в панели управления перейдите в раздел «Внешний вид» → «Темы». Нажмите кнопку «Добавить новую». Подберите тему WordPress из каталога, найдите нужную вам по названию или с помощью фильтра характеристик.

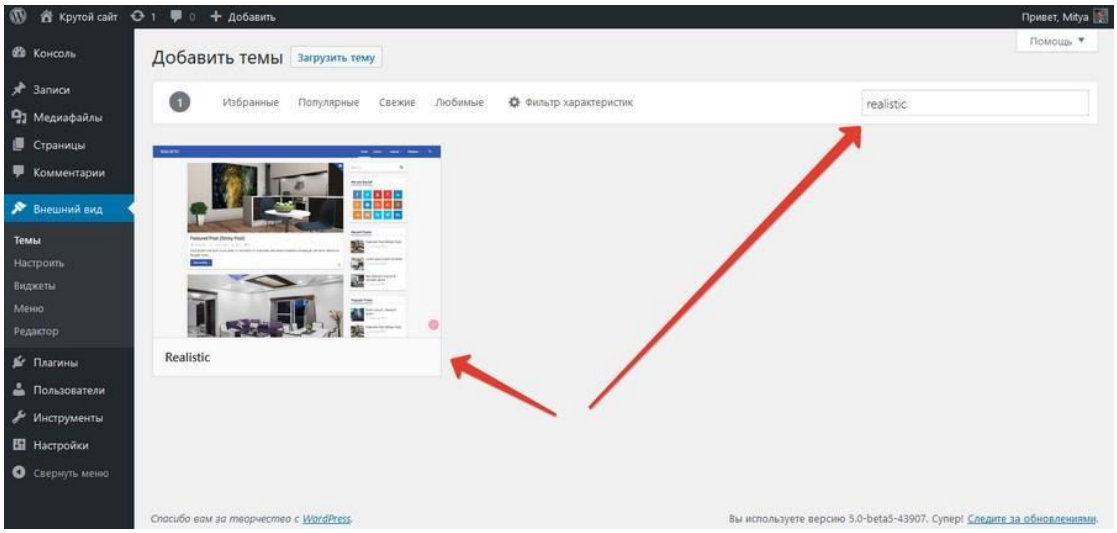

Рис. 7.1. Поиск нужной темы

Наведите курсор на выбранную тему и нажмите кнопку «Просмотреть». WordPress включит режим предпросмотра. Если внешний вид сайта вам нравится, установите тему, нажав кнопку «Установить».

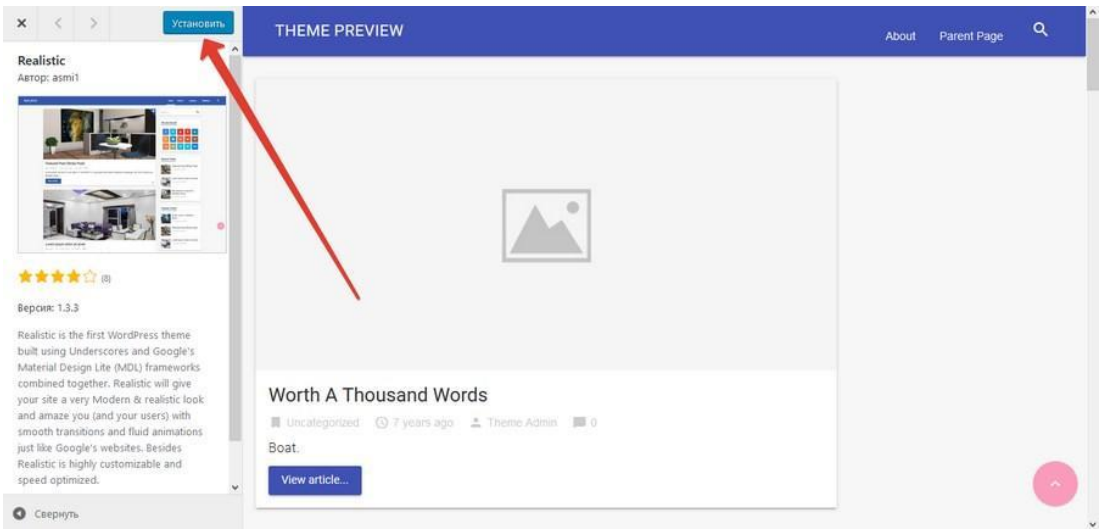

Рис. 7.2. Включение предпросмотра сайта

Более подробную инструкцию и FAQ по настройке темы можно найти в полном гиде от WordPress. (https://ru.wordpress.org/support/article/first-steps-with-wordpress-classic/).

Шаг 4. Добавляем контент

В WordPress есть две разновидности контента: страницы и записи.

Записи — часть блога, они показаны в обратном порядке: новые записи отражаются в начале.

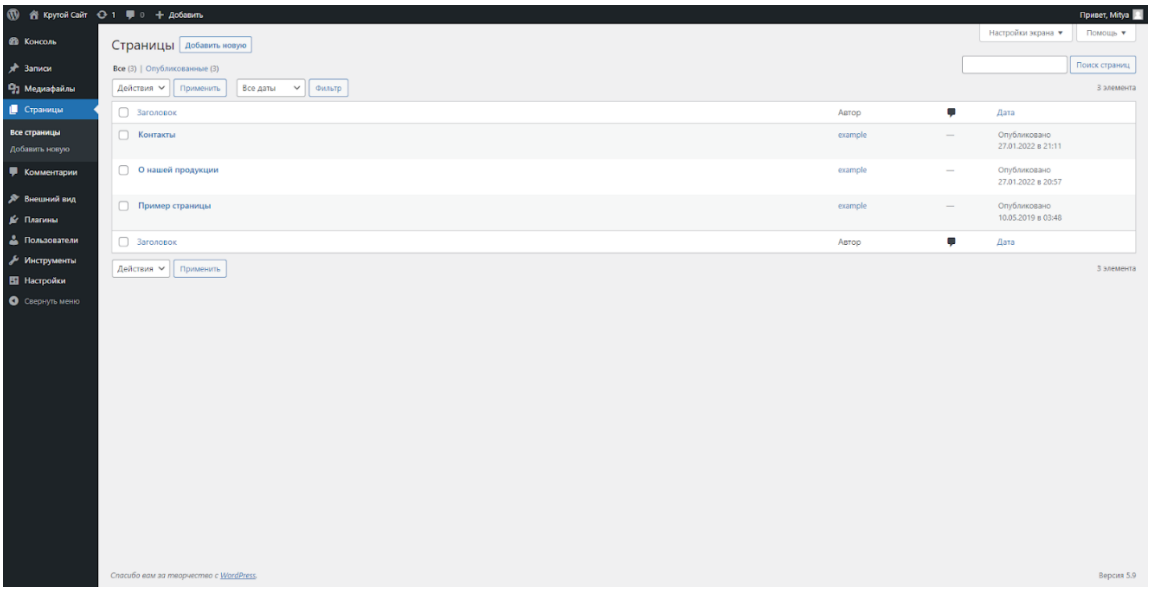

Рис. 7.3. «Записи» в WordPress

Страницы — это статический контент: например, страница контактов, страница с информацией о компании и т. д.

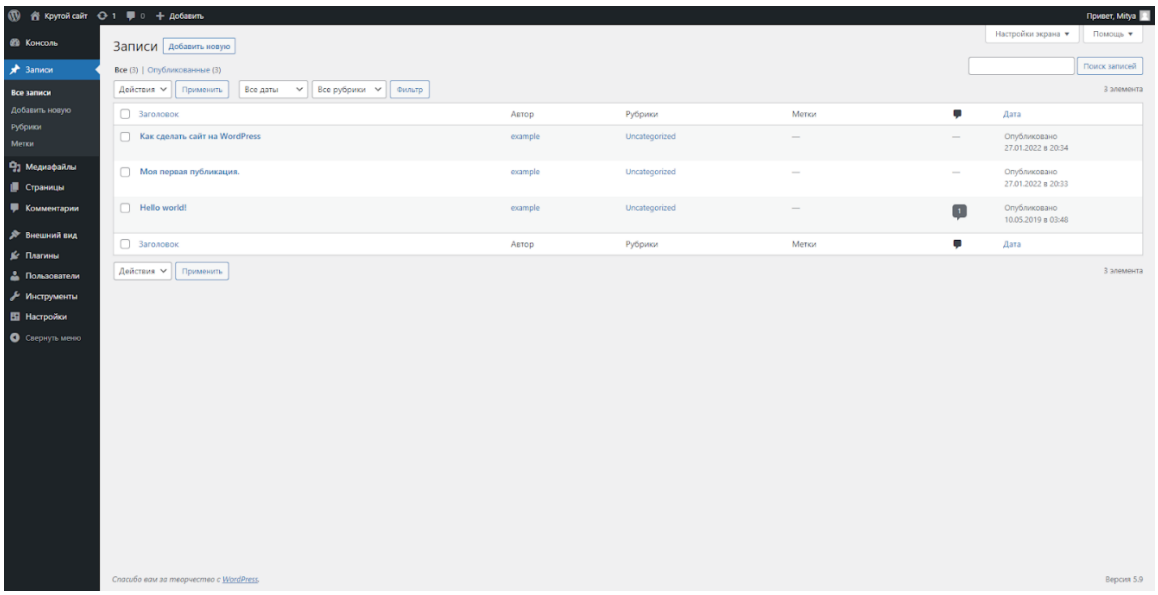

Рис. 7.4. «Страницы» в WordPress

Чтобы добавить контент на сайт, перейдите в раздел «Страницы» или «Записи» на панели управления, нажмите «Добавить новую». В редакторе страниц вы можете добавить текст, вставить видео, ссылку, картинку, аудио и т. п. Чтобы разместить готовый контент на сайт, нажмите кнопку «Опубликовать».

Шаг 5. Устанавливаем дополнительные плагины

Плагины — программные модули, которые подключаются к CMS. С их помощью можно делать страницы сайта безопаснее, увеличивать скорость загрузки страниц, добавлять на сайт форум, содержание, создавать опросы и многое другое.

Существует два наиболее безопасных источника, где можно найти плагины:

- на официальном сайте WordPress.org.
- в консоли WordPress.

Четыре типа плагинов, которые точно пригодятся даже новичкам:

- Бэкап — он предотвратит потерю всей информации сайта, создаст резервные копии.
- Безопасность — такие плагины защищают данные сайта и пользователей. Например, если ваши пользователи создают личный кабинет, то пригодится плагин двухфакторной аутентификации, генерации надежных паролей и reCAPTCHA.
- Дизайн и настройка — эти плагины помогут определиться с дизайном главной страницы и придумать свою пользовательскую тему.
- Плагины для повышения производительности и настройки кеширования контента — такие плагины помогут ускорить работу сайта. Самые популярные из них — W3 Total Cache, WP Super Cache, WP Fastest Cache.

С подробной информацией о создании веб-сайтов при помощи CMS можно познакомиться в следующих источниках Интернет:

1. Как сделать сайт на WordPress – очень полезный, полный гайд [Электронный ресурс]: - Режим доступа: https://texterra.ru/blog/kak-sozdat-sayt-na-wordpress-polnoe-rukovodstvo-dlya-novichkov.html (дата обращения 11.11.2022).

2. Как создать сайт на WordPress с нуля [Электронный ресурс]: - Режим доступа: https://timeweb.com/ru/community/articles/kak-sozdat-sayt-na-wordpress-s-nulya (дата обращения 11.11.2022).

3. Как сделать сайт на WordPress? Создать сайт на WordPress [С НУЛЯ ПО ШАГАМ 2021] [Электронный ресурс]: - Режим доступа: https://www.youtube.com/watch?v=4IATvixHm40 (дата обращения 11.11.2022).

4. Создание сайта на WordPress: основные этапы и настройки [Электронный ресурс]: - Режим доступа: https://gb.ru/blog/sozdanie-sajta-na-wordpress/ (дата обращения 11.11.2022).

5. Как создать сайт на WordPress: полезный полный гайд [Электронный ресурс]: - Режим доступа https://www.calltouch.ru/blog/kak-sozdat-sajt-na-wordpress-poleznyj-polnyj-gajd/ (дата обращения 11.11.2022).

6. Как создать сайт на WordPress в 2022 году. Пошаговое руководство с советами и ссылками [Электронный ресурс]: - Режим доступа: https://ichigarev.ru/sozdanie-saita/kak-sozdat-sayt-na-wordpress-poshagovoe-rukovodstvo.html (дата обращения 11.11.2022).

7. Создание сайтов на WordPress с нуля [Электронный ресурс]: - Режим доступа: https://glavhost.ru/sajt-na-wordpress (дата обращения 11.11.2022).

8. Как создать сайт на WordPress с нуля самостоятельно [Полная инструкция для новичков] [Электронный ресурс]: - Режим доступа: https://texterra.ru/blog/kak-sozdat-sayt-na-wordpress-polnoe-rukovodstvo-dlya-novichkov.html (дата обращения 11.11.2022).

9. Как создать сайт на WordPress: руководство для начинающих [Электронный ресурс]: - Режим доступа: https://wpschool.ru/howto-create-wp-site/ (дата обращения 11.11.2022).

10. Рейтинг ТОП 10 CMS сайтов: Какую лучше выбрать [Электронный ресурс]: - Режим доступа: https://habr.com/ru/post/645947/ (дата обращения 11.11.2022).

11. Что такое CMS и как ее использовать [Электронный ресурс]: - Режим доступа: https://wiki.rookee.ru/cms/ (дата обращения 11.11.2022).

12. Как создать сайт на WordPress с нуля [Электронный ресурс]: - Режим доступа https://www.nic.ru/info/blog/site-on-wordpress/?ref=vc.ru (дата обращения 11.11.2022).

13. Официальный сайт CMS WordPress на русском языке [Электронный ресурс]: - Режим доступа: https://codex.wordpress.org/ru:Main_Page (дата обращения 11.11.2022).

14. Документация, советы и примеры по реализации элементов CMS WordPress [Электронный ресурс]: - Режим доступа: https://wp-kama.ru/ (дата обращения 11.11.2022).

15. 9 уроков по рисованию макета сайта, его верстке и установке на CMS

WordPress [Электронный ресурс]: - Режим доступа: https://webformyself.com/minikurs/index-subscribe.html (дата обращения 11.11.2022).

16. Как быстро сделать интернет-магазина на WordPress [Электронный ресурс]: - Режим доступа: https://plugin-wp.ru/obzory/internet-magazin-na-wordpress/#%D0%9A%D0%B0%D0%BA_%D0%BD%D0%B0%D1%81%D1%82%D1%80%D0%BE%D0%B8%D1%82%D1%8C_%D0%BF%D0%BB%D0%B0%D0%B3%D0%B8%D0%BD (дата обращения 11.11.2022).

17. Как сделать сайт на Wordpress (2022) - урок в 23 ПРОСТЫХ шага [Электронный ресурс]: - Режим доступа https://www.youtube.com/watch?v=9AvU3nwsc5M (дата обращения 11.11.2022).

Требования к отчету и защите

В отчете указываются название, цель работы. Описание выполненных лабораторных заданий, разработанные Веб-страницы с результатами в виде скриншотов и выводами по каждому заданию.

На защите проверяются приобретенные знания теоретического и практического материала по ответам на контрольные вопросы для самопроверки.

ЗАКЛЮЧЕНИЕ

Выполнение лабораторных работ данного практикума должно способствовать формированию у студентов профессиональных знаний и целостное представление о сетевых информационных технологиях и основах реализации и функционирования сетевых приложений. Приобретенные практические навыки по разработке и поддержке сетевых веб-приложений обеспечивают закрепление теоретических знаний по дисциплине “Сетевые информационные технологии”.

ЛИТЕРАТУРА

1. Нагаева, И. А. Основы web-дизайна. Методика проектирования : учеб. пособие : [12+] / И. А. Нагаева, А. Б. Фролов, И. А. Кузнецов. – Москва ; Берлин : Директ-Медиа, 2021. – 236 с. : ил. – Режим доступа: по подписке. – URL: https://biblioclub.ru/index.php?page=book&id=602208 (дата обращения: 23.05.2022). – Библиогр. в кн. – ISBN 978-5-4499-1957-1. – Текст : электронный.

2. Зайцева, О. С. Технологии разработки web-ресурсов : учеб. пособие : [16+] / О. С. Зайцева ; Тюменский индустриальный университет. – Тюмень : Тюменский индустриальный университет, 2020. – 75 с. : ил., табл. – Режим доступа: по подписке. – URL: https://biblioclub.ru/index.php?page=book&id=611103 (дата обращения: 23.05.2022). – ISBN 978-5-9961-2274-5. – Текст : электронный.

3. Титов, В. А. Разработка WEB-сайта средствами языка HTML : учебное пособие / В. А. Титов, Г. И. Пещеров. – Москва : Институт мировых цивилизаций, 2018. – 184 с. : ил., табл. – Режим доступа: по подписке. – URL:

https://biblioclub.ru/index.php?page=book&id=598475 (дата обращения: 23.05.2022). – Библиогр. в кн. – ISBN 978-5-9500469-3-3. – Текст : электронный.

Дополнительная

4. Прохоренок, Н. А. HTML, JavaScript, PHP и MySQL. Джентльменский набор Web-мастера. — 5-е изд., перераб. и доп. / Н. А. Прохоренок, В. А. Дронов. — Санкт-Петербург: БХВ-Петербург, 2019. — 912 с.: ил. — (Профессиональное программирование) ISBN 978-5-9775-3986-9.

5. Васильева, М. А. Основы командной разработки. Основы командной разработки : учеб. пособие для вузов / М. А. Васильева, К. М. Филипченко. Санкт-Петербург : Лань, 2022. – 144 с. : ил. – Текст : непосредственный) ISBN 978-5-507-44630-8.

6. Хоффман, Э. Безопасность веб-приложений / Э. Хоффман. — Санкт-Петербург: Питер, 2021. — 336 с.: ил. — (Серия «Бестселлеры O’Reilly»). ISBN 978-5-4461-1786-4

7. Глушенко, С. А. Разработка мобильных приложений: учеб. пособие / С. А. Глушенко, А. И. Долженко – Ростов-на-Дону: Изд-во РГЭУ (РИНХ), 2018. – 221 с.

8. Парамонов, И. В. Разработка мобильных приложений для платформы Android: учеб. пособие / И. В. Парамонов; Яросл. гос. ун-т им. П. Г. Демидова. — Ярославль : ЯрГУ, 2013. — 88 с. ISBN 978-5-8397-0930-0.

9. Куроуз, Д. Компьютерные сети : Нисходящий подход / Д. Куроуз, К. Росс. 6-е изд. – Москва : «Эксмо», 2016. - 912 с. – (Мировой компьютерный бестселлер). ISBN 978-5-699-78090-7

Локальный электронный методический материал

Галина Владимировна Ломакина

Виктор Анатольевич Петрикин

Сетевые информационные технологии

Редактор Г. А. Смирнова

Уч.-изд. л. 5,0. Печ. л. 4,25

Издательство федерального государственного бюджетного образовательного учреждения высшего образования «Калининградский государственный технический университет».
236022, Калининград, Советский проспект, 1