Л.Е. Рамазанова, К.М. Аубакирова
А.Б. Тойшибекова, А.Н. Токарев

Создание WEB-страниц, сайтов с применением WEB-технологий

* * * * *

Л.Е. РАМАЗАНОВА, К.М. АУБАКИРОВА
А.Б. ТОЙШИБЕКОВА, А.Н. ТОКАРЕВ

СОЗДАНИЕ WEB-СТРАНИЦ, САЙТОВ С ПРИМЕНЕНИЕМ WEB-ТЕХНОЛОГИЙ

Разработано в качестве учебного пособия системы технического и профессионального образования по специальности «Вычислительная техника и программное обеспечение»

Астана
2018

УДК 004.4 (075)
ББК 32.973.202
С58

«Создание web-страниц, сайтов с применением web-технологий» Учебное пособие / Рамазанова Л.Е., Аубакирова К.М., Тойшибекова А.Б., Токарев А.Н. -Астана: Некоммерческое акционерное общество «Холдинг “Кәсіпқор”», 2018 г.

ISBN 978-601-333-510-0

Данное учебное пособие разработано в соответствии с типовым учебным планом технического и профессионального образования по специальности 1304000 «Вычислительная техника и программное обеспечение», с целью достижения результатов обучения по профессиональному модулю «Создание web-страниц, сайтов с применением web-технологий» и обеспечения обучающихся и преподавателей учебным материалом по основным теоретическим концепциям технологии разработки web-страниц и практическим примерам.

Учебное пособие содержит необходимые теоретические и практические материалы для создания web-страниц с использованием гипертекстовой разметки и каскадных таблиц стилей CSS, знакомит с основными web-технологиями, такими как PHP и MySQL, а также современными принципами и инструментальными средствами разработки web-сайтов и серверным программным обеспечением.

Представленный в учебном пособии материал позволяет освоить необходимые компетенции для создания web-сайта с использованием основных технологий разработки web-страниц.

УДК 004.4 (075)
ББК 32.973.202

Рецензенты:
Айтимов А.С. - кандидат технических наук, профессор.
Дудниченков М.М. – генеральный директор ТОО «IT Group Kazakhstan»
Ахметова А.Х.- преподаватель специальных дисциплин Высшего аграрно- технического колледжа

Одобрено Научно-методическим советом НАО «Холдинг “Кәсіпқор”», Протокол № 2 от 26.09.2018 г.

ОГЛАВЛЕНИЕ

Введение

Настоящее учебное пособие разработано в соответствии с актуализированным типовым учебным планом и программами по специальности 1304000 «Вычислительная техника и программное обеспечение», квалификации 1304012 «Оператор электронно-вычислительных машин», изучающих модуль (ПМ5) «Создание web-страниц, сайтов с применением web-технологий».

В современном мире учебные заведения, предприятия, компании, которые работают в сфере рекламы, образования, продвижения того или иного товара, стремятся иметь свое место во Всемирной паутине. Если рассматривать образовательную сферу, то на сегодняшний день все учреждения, начиная с детских садов, школ, колледжей и заканчивая высшими учебными заведениями, имеют свои web-сайты (от англ. web-паутина, сеть и site - место). Сайт – виртуальная визитная карточка организации, возможность рассказать всему миру о деятельности, структуре и об оказываемых услугах. Главной его миссией является предоставление качественной информации посетителю. При этом грамотно разработанный контент, хорошо подобранный дизайн, интерфейс сайта играют немаловажную роль.

Учебное пособие по дисциплине «Web-программирование и интернет-технологии» подготовлено и структурировано так, чтобы обучающийся в итоге получил знания, которые помогут ему разработать полноценно функционирующий сайт с интересным контентом и дизайном. Учебное пособие разработано для специальностей, которые изучают программирование, а также оно поможет тем, кто самостоятельно желает ознакомиться с современными инструментальными средствами разработки web-сайта, изучить основы гипертекстовой разметки HTML, каскадных таблиц стилей CSS, разобраться с серверными программными обеспечениями. Пособие в полном объеме содержит необходимый теоретический и практический материал, примеры фрагментов кода для качественного освоения материала. Первая и вторая главы пособия отведены изучению интернет-протоколов, безопасности работы в сети, основам HTML, CSS, а также построению базы данных MySql и языку скриптов JavaScript. Третья глава пошагово описывает создание web-сайта при помощи панели Plesk.

Разработчик web-сайта — одна из распространенных и достойно оплачиваемых профессий на сегодняшний день. Одним из ее преимуществ является гибкий график работы или фриланс, где минимальная необходимость — это ноутбук и Интернет.

1. Представление и передача информации в сети Интернет

1.1 Распределенная обработка информации на основе web-технологий

Использование распределенных систем обработки данных стало актуальным с появлением многомашинных вычислительных комплексов с распределенными ресурсами в пределах одного компьютера (metacomputer), локальных и глобальных компьютерных сетей, технологий поиска и многомерного анализа данных, а также с развитием web-технологий.

Распределенная обработка данных – это система обработки данных, которая ведется несколькими, рассредоточенными в организации компьютерами, а не одним центральным. Компьютеры могут быть связаны между собой в единую сеть (network), что позволяет им работать совместно, либо подключены к более мощному центральному компьютеру, хотя значительная часть обработки информации происходит без обращения к нему. Это отличается от единого централизованного управления сервером и обеспечения возможности обработки для всех подключенных систем. Компьютеры, которые составляют распределенную сеть обработки данных, расположены в разных местах, но взаимосвязаны посредством беспроводных или спутниковых каналов.

Пример. Web-сайт размещается на онлайн-сервере. Когда посетитель приходит на сайт, страницы загружаются с сервера, который находится рядом с пользователем. Google также использует распределенную обработку и имеет серверы баз данных во всех основных странах. Когда пользователь компьютера посещает сайт Google из Китая, сайт загружается с китайского сервера.

Рисунок 1.1 Схема распределенной обработки данных
Адаптировано с сайта http://www.itrelease.com/2018/07/advantages-and-disadvantages-of-distributed-data-processing/

В распределенной обработке существует один главный сервер, который контролирует все остальные компьютеры в сети. Распределенная обработка выполняется с высокой скоростью Интернета, например, при запросе базы данных.

Целью распределенной обработки информации является **оптимизация использования ресурсов** и **упрощение работы пользователя**.

Преимущества распределенной обработки информации:

1. Низкая стоимость. Большие организации инвестируют в дорогие мейнфреймы и суперкомпьютеры, которые будут функционировать как централизованные серверы. По данным Университета Нью-Мексико, каждая машина мейнфрейма стоит сотни тысяч долларов против нескольких тысяч долларов за миникомпьютеры. Распределенная обработка значительно снижает затраты на совместное использование данных и сетевое взаимодействие в организации путем включения нескольких миникомпьютеров, которые стоят значительно меньше, чем мейнфреймы.

2. Надежность. Аппаратные сбои и аномалии программного обеспечения могут привести к ошибкам в работе одного сервера или к полному сбою системы. Распределенная обработка данных является более надежной, поскольку несколько центров управления распределены между разными машинами. Сбой на одной машине не влияет на сеть, поскольку другая машина использует свои возможности обработки. Неисправные устройства быстро изолируются и ремонтируются. Это делает распределенную обработку данных надежнее односерверных систем.

3. Повышение производительности и сокращение времени обработки. Отдельные компьютеры ограничены по производительности и эффективности. Легкий способ повысить производительность - добавить другой компьютер в сеть. Распределенная обработка данных действует по этому принципу и считает, что работа выполняется быстрее, если несколько машин выполняют ее параллельно или синхронно. Например, сложные статистические проблемы разбиваются на модули и распределяются на разные машины, где они обрабатываются одновременно. Это значительно сокращает время обработки и повышает производительность.

4. Гибкость. Отдельные компьютеры, которые содержат распределенную сеть, присутствуют в разных географических точках. Например, организационно-распределенная сеть, состоящая из трех компьютеров, может иметь каждую машину в другом филиале. Эти три машины взаимосвязаны через Интернет и могут обрабатывать данные параллельно даже в разных местах. Это делает сети распределенной передачи данных более гибкими. Система является гибкой и с точки зрения увеличения или уменьшения вычислительной мощности. Например, добавление большего количества узлов или компьютеров в сеть увеличивает вычислительную мощность и общую способность системы, а сокращение компьютеров в сети снижает эти показатели.

Недостатки распределенной обработки информации:

1. Сложность. Компьютеры, подключенные к протоколу доставки дейтаграмм DDP (Datagram Delivery Protocol), трудно устранить, спроектировать и администрировать.

2. Синхронизация данных планирования затруднена. Невозможно разработать правильную синхронизацию данных, так как иногда они обновляются в неверном порядке. Поэтому администраторы должны

сосредоточиться на порядке обновления данных, прежде чем создавать распределенную сеть.

3. Безопасность данных. Если несанкционированный компьютер подключен к распределенной сети, он может повлиять на другие характеристики компьютера, и данные могут быть потеряны.

Примеры распределенной обработки данных:

- Инструменты для редактирования фотографий в Интернете,
- Система бронирования авиабилетов,
- Обработка пользовательских данных мобильными компаниями,
- Dropbox, диск Google, накопитель MSN, фотографии Google,
- Генерация отчетов со спутника,
- Система прогноза погоды.

Контрольные вопросы

1. В чем суть распределенной обработки данных?
2. В чем преимущества распределенной обработки данных?
3. Перечислите недостатки распределенной обработки информации.
4. В чем заключается надежность распределенной обработки информации? Приведите примеры.

1.2 Функции сетевых служб

Сетевая служба предоставляется сервером, который может запускать одну или несколько служб на основе сетевых протоколов, работающих на уровне приложения в модели сетевого взаимодействия Open Systems Interconnection (OSI).

Сетевая услуга — это процесс обслуживания объектов сети, обычно связанный с распределенной обработкой данных и информационным обменом. Состав сетевых функций услуг определяется его функциональной и поведенческой спецификацией.

Примечание. Сетевая услуга способствует поведению службы более высокого уровня, которая характеризуется производительностью, надежностью и спецификациями безопасности. Поведение сквозных сетевых услуг является результатом сочетания поведения отдельных сетевых функций, а также поведения механизма компоновки сетевой инфраструктуры одной или нескольких служб на основе сетевых протоколов, работающих на прикладном уровне в модели сети Open Systems Interconnection (OSI).

За последние два десятилетия появилось множество инноваций в устройствах, которые используются для доступа к сети, приложениям и сервисам, а также для вычислений и хранения «больших данных». Однако базовая сеть, которая соединяет все перечисленное, практически не изменилась.

В компьютерных сетях сетевая служба - это приложение, работающее на сетевом уровне и выше, которое обеспечивает хранение данных, манипулирование, представление, обмен данными и другие возможности.

Эти возможности часто реализуются с использованием клиент-серверной или одноранговой архитектуры на основе сетевых протоколов прикладного уровня.

Каждая служба обычно предоставляется серверным компонентом, работающим на одном или нескольких компьютерах, и доступными по сети клиентскими компонентами, функционирующими на других устройствах. Однако клиентский и серверный компоненты могут выполняться на одной машине.

Примерами сетевых служб являются система доменных имен (DNS), которая преобразует доменные имена в адреса интернет-протокола (IP) и протокол DHCP (Dynamic Host Configuration Protocol) для назначения информации о конфигурации сети для сетевых узлов. Серверы проверки подлинности идентифицируют и аутентифицируют пользователей, предоставляют профили их учетных записей и могут регистрировать статистику использования.

Функции IP. IP – это интернет-протокол, отвечающий за адресацию хостов, инкапсуляцию данных в дейтаграммы (включая фрагментацию и повторную сборку) и маршрутизацию дейтаграмм с исходного хоста на конечный хост в одной или нескольких IP-сетях. Для этих целей интернет-протокол определяет формат пакетов и обеспечивает систему адресации.

Каждая дейтаграмма состоит из двух компонентов: заголовка и полезных данных. IP-заголовок содержит IP-адрес источника, IP-адрес назначения и другие метаданные, необходимые для маршрутизации и доставки дейтаграммы. Полезная нагрузка - это передаваемые данные. Этот метод вложения полезных данных в пакет с заголовком называется инкапсуляцией.

IP-адресация подразумевает назначение интерфейсам хоста IP-адресов и связанных с ними параметров. Адресное пространство разделено на подсети, включающие назначение сетевых префиксов. IP-маршрутизация выполняется всеми хостами, а также маршрутизаторами, основной функцией которых является транспортировка пакетов через границы сети.

Функции DNS. Система доменных имен служит «телефонной книгой» для Интернета, переводя удобные для человека имена хостов компьютеров в IP-адреса. Например, доменное имя www.example.com переводится в 93.184.216.34-адреса (IPv4) и 2606:2800:220:1:248:1893:25c8:1946 (протокол IPv6). При обращении пользователя к распределенной интернет-службе с помощью URL-адреса, доменное имя URL-адреса преобразуется в IP-адрес сервера, который находится ближе к пользователю. DNS может быть быстро обновлен, позволяя местоположению сервиса в сети измениться, не затрагивая конечных пользователей, которые продолжают применять то же имя хоста. Пользователи владеют этим преимуществом, когда они используют значимые унифицированные локаторы ресурсов (URL-адреса) и адреса электронной почты без необходимости знать, как компьютер фактически находит службы.

Важной и повсеместной функцией DNS является ее центральная роль в распределенных интернет-службах, таких как облачные службы и сети доставки контента. Ключевая функциональность DNS здесь заключается в том, что разные пользователи могут одновременно получать разные переводы для одного и того же доменного имени, что является ключевым моментом расхождения с традиционным представлением DNS как «телефонной книги». Этот процесс использования DNS для назначения проксимальных серверов пользователям является ключевым для обеспечения более быстрых и надежных ответов в Интернете и широко применяется большинством основных интернет-служб.

DNS отражает структуру административной ответственности в Интернете. Для зон, управляемых реестром, административная информация часто дополняется службами RDAP (протокол регистрации доступа к данным) и WHOIS (сетевой протокол прикладного уровня, базирующийся на протоколе TCP порт 43) реестра. Эти данные могут быть использованы для получения информации и отслеживания ответственности за определенный хост в Интернете.

Функции DHCP. DHCP (протокол динамической конфигурации хоста) - это протокол управления сетью, используемый для динамического назначения IP-адреса любому новому узлу, входящему в сеть. DHCP позволяет настраивать узел автоматически, тем самым избегая необходимости участия сетевого администратора.

DHCP выполняет следующие действия:

1. Управляет предоставлением всех узлов, добавленных в сеть или удаленных из нее.

2. Поддерживает уникальный IP-адрес хоста с помощью DHCP-сервера.

3. Отправляет запрос на сервер DHCP всякий раз, когда клиент/узел, настроенный для работы с DHCP, подключается к сети. Сервер подтверждает запрос, предоставляя IP-адрес клиенту/узлу.

DHCP - это автоматизированный метод, с помощью которого любому вновь добавленному или переданному узлу в сети мгновенно может быть назначен или переназначен IP-адрес. Без DHCP сетевые администраторы будут вынуждены назначать IP-адрес вручную для каждого узла в сети.

Функции DHCP-сервера:

1. DHCP-сервер настроен для управления предоставлением IP-адресов и является важным требованием для запуска протокола DHCP. Сервер управляет записью всех IP-адресов, которые он назначает узлам. Если узел присоединяется к сети или перемещается в ней, сервер идентифицирует узел, используя его MAC-адрес (физический адрес). Это помогает предотвратить случайную настройку одного и того же IP-адреса на два разных узла.

2. Когда клиент, поддерживающий DHCP, подключается к сети, он передает запрос на сервер DHCP для сетевых настроек.

3. Сервер отвечает на запрос клиента, предоставляя необходимую информацию о конфигурации IP.

4. DHCP-сервер идеально подходит для сценариев, где есть регулярное включение и исключение сетевых узлов, таких как беспроводные точки доступа. В этих случаях сервер DHCP также назначает время аренды каждому клиенту, после чего назначенный IP-адрес становится недействительным.

Службы электронной почты, печати и распределенной (сетевой) файловой системы являются общими в локальных сетях. При описании сетевых служб применяется стек языков описания, в котором каждый элемент более высокого уровня использует и уточняет описание нижнего уровня. В качестве общего метаязыка в настоящее время используется XML, который широко распространен, имеет гибкий синтаксис и позволяет определять языки описания и протоколы служб. При описании интерфейсов сетевой службы дополнительно указывается ее адрес и транспортный протокол.

Решения об использовании службы принимаются на основании их описаний, доступных благодаря спецификации универсальных средств описания, обнаружения и интеграции. В этой спецификации показано, как организуется информация о сетевой службе и как строить репозитории, где эта информация может регистрироваться. Для обеспечения доступности сетевых служб их описания заносятся в справочники, которые позволяют разработчикам регистрировать новые службы, а пользователям отыскивать их. Поиск службы может осуществляться во время разработки или динамически. Для взаимодействия с единой справочной службой или между локальными справочниками определяются протоколы и программные интерфейсы опубликования и поиска информации.

Каждый уровень стандартов взаимодействия сетевых служб характеризуется одним или несколькими протоколами, которые применимы ко всем сетевым службам.

Сетевые службы, получающие запросы и передающие их другим системным программам, относятся **к внутренней** архитектуре. Сетевые службы, обеспечивающие интеграцию различных служб между собой, относятся **к внешней** архитектуре. Сетевые службы играют ту же роль, что и традиционные промежуточные слои программного обеспечения, но имеют другой масштаб. Они являются оболочками, которые обращаются к внутренним службам, реализующим нужную прикладную логику. Системная поддержка сетевых служб связана с упаковкой и распаковкой сообщений, пересылаемых между ними, а также с преобразованием их в формат внутреннего программного обеспечения.

Чтобы сетевые службы имели глобальный характер, процесс публикации сведений о них и их поиск должны быть стандартизованы. Для этого используется **универсальная система описания, поиска и взаимодействия UDDI (Universal Difinition Detection Integration)**, состоящая из реестра и прикладного программного интерфейса. Реестр эквивалентен серверу именования. Прикладной интерфейс UDDI определяет, как опубликовать службу, что необходимо для регистрации, как делать запросы к службе. Информация в реестре используется для написания

клиентских программ и для обеспечения динамического поиска службы. Информация, занесенная в реестр UDDI, группируется в алфавитном порядке имен предприятий, поставляющих сетевые службы для использования, по тематике, к которой относится как деятельность поставщиков, так и сами сетевые службы, и по способам вызова сетевых служб (в виде ссылок на документы, хранящиеся вне реестра).

Работу с реестром могут проводить поставщики служб, клиенты и другие реестры. Для разных типов пользователей реестры поддерживают разные точки входа, взаимодействие с которыми осуществляется посредством обмена XML-документами. Реестр имеет разные виды прикладного программного интерфейса:

1. **Интерфейс запросов реестра** содержит операции для поиска записей и операции, позволяющие получить описания объекта. Интерфейс используется разработчиками и клиентами для динамической привязки.
2. **Интерфейс публикации реестра** предназначен для поставщиков служб.
3. **Интерфейс безопасности реестра** позволяет пользователям проходить аутентификацию.
4. **Интерфейс надзора и передачи прав владения реестра** позволяет передавать часть своих прав от одного поставщика служб другим. В любых условиях владельцем записей в реестрах всегда является реестр, в котором запись была первоначально создана. Модификации записей проводятся только в реестрах-владельцах, но права можно передать другим реестрам в явном виде.
5. **Интерфейс подписки на реестр** позволяет подписываться на новые или модифицированные службы.
6. **Интерфейс репликации** помогает реплицировать информацию, обеспечивая синхронность различных реестров.

Контрольные вопросы

1. Дайте определение понятию «сетевая служба».
2. Представьте общую характеристику и архитектуру сетевых служб.
3. Опишите работу сетевой службы DNS.
4. Перечислите функции DHCP-сервера.

1.3 Функции клиент/сервер в информационных системах

Клиент/сервер - это отношение к программе, в котором одна программа (клиент) запрашивает услугу или ресурс из другой программы (сервера).

Хотя модель клиент/сервер может использоваться программами на одном компьютере, это более важная концепция для сетей. В этом случае клиент устанавливает соединение с сервером через локальную сеть (LAN) или глобальную сеть (WAN), такую как Интернет. Как только сервер выполнит запрос клиента, соединение будет прекращено. Web-браузер - это клиентская программа, которая запросила услугу с сервера; на самом деле, услуга и повторный доступ к серверу - это доставка web-страницы.

Компьютерные транзакции, в которых сервер выполняет запрос, сделанный клиентом, очень распространены, и модель клиент / сервер стала одной из центральных идей сетевых вычислений. Большинство бизнес-приложений используют модель клиент/сервер, как и основная программа Интернета TCP / IP. Например, при проверке банковского счета со своего компьютера, клиентская программа на устройстве отправляет запрос на серверную программу в банке. Она, в свою очередь, направляет запрос собственной клиентской программе, которая затем аналогично связывается с сервером базы данных на другом банковском компьютере. После того, как баланс учетной записи был извлечен из базы данных, он возвращается обратно клиенту банковских данных, который направляет его клиенту на персональном компьютере, отображающем информацию.

Как клиентские, так и серверные программы нередко являются частью более крупной программы или приложения. Поскольку несколько клиентских программ совместно используют службы одной и той же серверной программы, специальный сервер, называемый демонами, может быть активирован только для ожидания запросов клиентов. В маркетинге клиент / сервер когда-то использовался, чтобы отличать распределенные вычисления персональных компьютеров (ПК) от монолитной, централизованной вычислительной модели, используемой мэйнфреймами. Однако это различие в значительной степени исчезло, поскольку мэйнфреймы и их приложения также обратились к модели клиент / сервер и стали частью сетевых вычислений.

Другие модели отношений с программами включали master/slave и peer-to-peer (P2P). В модели P2P каждый узел в сети может функционировать как сервер и клиент. В модели ведущий / ведомый одно устройство или процесс (известный как ведущий) управляет одним или несколькими другими устройствами или процессами (известными как подчиненные). После установления отношения ведущий/ведомый направление управления всегда одностороннее: от ведущего к ведомому.

Контрольные вопросы:

1. Что такое «сервер»? Что такое «клиент»?
2. Охарактеризуйте основные функции клиент/сервер.
3. Опишите модель peer-to-peer (P2P).

1.4 Службы и протоколы сети Интернет

Когда речь идет о работе в Интернете или об использовании Интернета, то подразумевается не Интернет в целом, а только одна или несколько из его многочисленных служб. В зависимости от конкретных целей и задач клиенты сети используют те службы, которые им необходимы. Разные службы имеют разные протоколы. Их соблюдение обеспечивается и поддерживается работой специальных программ. Таким образом, чтобы воспользоваться одной из служб Интернета, необходимо установить на компьютере программу,

способную работать по протоколу данной службы. Такие программы называют клиентскими или просто клиентами.

Сетевые протоколы фактически управляют сетью, указывая сетевым устройствам, что они должны делать. Сетевые протоколы - это набор правил, по которым работает сеть. Для передачи информации по сети компьютеры должны использовать один и тот же набор правил, т.е. единый протокол.

Сетевые службы предназначены для выполнения определенных функций в рамках действующего протокола, например, существует служба разрешения имен, служба автоматического выделения адресов и др. Среди множества типов сетевых протоколов, работающих в разных сетях и на разных уровнях модели OSI, можно выделить следующие:

1. TCP/IP.
2. NetBEUI.
3. IPX/SPX.
4. NWLink.
5. Apple Talk.
6. DLC.

Протоколы удаленного доступа

Служба маршрутизации и удаленного доступа (RRAS) представляет собой набор сетевых услуг в семействе Windows Server, который позволяет серверу выполнять услуги обычного маршрутизатора. RRAS включает в себя интерфейс прикладного программирования (API), который облегчает разработку приложений и процессов для администрирования целого ряда сетевых сервисов. Служба RRAS поддерживает три протокола удаленного доступа:

Point-to-Point Protocol (PPP) - это протокол компьютерной сети, используемый для передачи дейтаграммы между двумя напрямую связанными («точка-точка») компьютерами. Этот протокол используется для базового уровня подключения, обеспечивающего связь данных между компьютерами, и широко применяется для более сложных и быстрых соединений, необходимых для широкополосной связи.

Serial Line Internet Protocol (SLIP) — это протокол TCP / IP, используемый для связи между двумя компьютерами, которые предварительно настроены для связи друг с другом. Например, поставщик интернет-сервера может предоставить пользователю SLIP-соединение, чтобы сервер провайдера мог отвечать на запросы, передавать их в Интернет и перенаправлять запрошенные ответы в Интернете.

Asynchronous NetBEUI (AsyBEUI) - протокол службы удаленного доступа Microsoft, известный также как асинхронный NetBEUI; применяется устаревшими клиентами удаленного доступа под управлением Windows NT, Windows 3.1, Windows for Workgroups, MSDOS и LAN Manager.

Стек протоколов TCP/IP

Стек TCP/IP - широко используемый набор протоколов для межсетевого взаимодействия, разработанных для обеспечения взаимосвязи различных устройств в сети Интернет. Стек состоит из следующих протоколов (Рис 1.2):

Протокол ARP (Address Resolution Protocol) - вспомогательный протокол стека TCP/IP для сопоставления IP-адреса с физическим адресом компьютера, который распознается в локальной сети. Например, в IP-версии 4, IP-адрес имеет длину 32 бита. Однако в локальной сети Ethernet адреса для подключенных устройств имеют длину 48 бит. Таблица, обычно называемая ARP-кешем, используется для поддержания корреляции между каждым физическим адресом и соответствующим IP-адресом. ARP предоставляет правила протокола для выполнения этой корреляции и обеспечения преобразования адресов в обоих направлениях, предназначенных для определения аппаратного адреса узла назначения по заданному IP-адресу.

Протокол ICMP (Internet Control Message Protocol) - вспомогательный протокол сетевого уровня TCP / IP, который предоставляет службы устранения неполадок, управления и сообщений об ошибках. ICMP чаще всего используется в операционных системах для сетевых компьютеров, где он передает сообщения об ошибках. В частности, утилита PING использует этот протокол для посылки так называемого «эхо-запроса». Сообщение ICMP создается в результате ошибок в IP-дейтаграмме или для целей диагностической маршрутизации. Эти ошибки сообщаются исходному IP-адресу исходной дейтаграммы. Сообщение ICMP инкапсулируется непосредственно в одну дейтаграмму IP и информирует об ошибках в обработке дейтаграмм.

Протокол IGMP (Internet Group Management Protocol) — это протокол, отвечающий за организацию групп, которые позволяют передавать потоки данных IP нескольким получателям. Это означает, что протокол IGMP управления группами Интернета автоматически реализуется на всех узлах, поддерживающих многоадресную рассылку IP. IGMP работает между маршрутизатором и узлом, который позволяет выполнять следующие действия:

– маршрутизаторы запрашивают узлы, если им нужен конкретный многоадресный поток (IGMP-запрос);

– узлы реагируют на маршрутизатор, если они ищут конкретный многоадресный поток (отчеты IGMP).

Протокол TCP (Transmission Control Protocol) - это протокол транспортного уровня, который используется для создания соединения между удаленными компьютерами путем транспортировки и обеспечения доставки сообщений через поддерживающие сети и Интернет. Протокол управления передачей TCP является одним из наиболее используемых протоколов в цифровой сети связи и является частью пакета интернет-протокола, широко известного как стек TCP/IP. В первую очередь TCP обеспечивает сквозную доставку данных между отдельными узлами. TCP работает в сотрудничестве с Internet Protocol, который определяет логическое расположение удаленного узла, тогда как TCP передает и обеспечивает доставку данных в правильное место назначения.

Перед передачей данных TCP создает соединение между исходным и конечным узлами и сохраняет его в реальном времени до тех пор, пока связь не будет активна. TCP разбивает большие данные на более мелкие пакеты, а также гарантирует целостность данных после их повторной сборки на узле назначения.

Протокол UDP (User Datagram Protocol) — это протокол транспортного уровня для сетевых приложений клиент/сервер. UDP применяет простую модель передачи, но не использует диалог установления связи для надежности, упорядоченности и целостности данных. Протокол UDP - это альтернативный протокол связи для протокола управления передачей TCP, используемый в основном для установления соединений с низкой задержкой и потерей данных между приложениями в Интернете. В тех случаях, когда протокол UDP обеспечивает взаимодействие между процессами, TCP поддерживает связь между узлами. TCP отправляет отдельные пакеты и считается надежным транспортным средством; UDP отправляет сообщения, называемые дейтаграммами, и считается наилучшим способом связи.

UDP широко используется в видеоконференциях и компьютерных играх реального времени. Протокол позволяет упаковывать отдельные пакеты и получать пакеты UDP в другом порядке, чем тот, в котором они были отправлены, что позволяет повысить производительность.

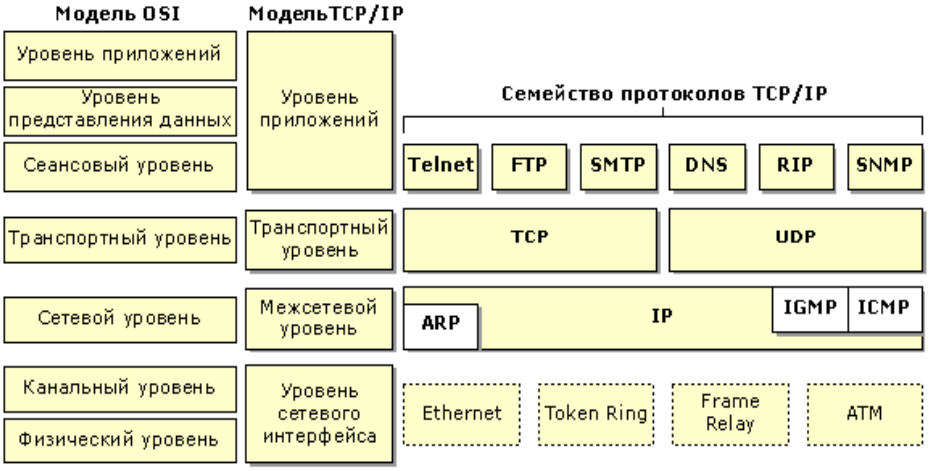


Рисунок 1.2 Стек протоколов TCP/IP
Адаптировано с сайта
http://www.xnets.ru/plugins/content/content.php?content.103

Компоненты прикладного уровня

На прикладном уровне работает множество стандартных утилит и служб TCP/IP, к числу которых относятся:

1. Протокол HTTP (HyperText Transfer Protocol) - это протокол прикладного уровня, используемый, в первую очередь, во Всемирной паутине. HTTP использует модель клиент/сервер, где web-браузер является клиентом и взаимодействует с web-сервером, на котором размещен web-сайт. Браузер использует HTTP, который переносится через TCP / IP для связи с

сервером и получения web-контента для пользователя. HTTP используется для организации доступа к общим данным, расположенным на web-серверах, с целью публикации и чтения общедоступной информации.

2. Протокол FTP (File Transfer Protocol) - это протокол прикладного уровня, который перемещает файлы между локальными и удаленными файловыми системами. Для передачи файла два TCP-соединения используются FTP параллельно: в управлении соединением и в подключении к данным.

3. Протокол SMTP (Simple Mail Transfer Protocol) - это стандартный протокол для почтовых служб в стеке TCP/IP. SMTP как протокол прикладного уровня предоставляет возможность отправлять и получать сообщения электронной почты через Интернет.

4. Протокол Telnet (telecommunication network) - протокол эмуляции терминала, который применяется для подключения к удаленным узлам сети. Telnet позволяет клиентам удаленно запускать приложения. Кроме того, он упрощает удаленное администрирование. Реализации Telnet, доступные практически для всех ОС, облегчают интеграцию в разнородных сетевых средах. При помощи Telnet можно:

– подключаться к удаленным компьютерам;

– проверить порт на наличие доступа;

– использовать приложения, которые доступны только на удаленных машинах;

– использовать различные каталоги, к которым получить доступ можно только таким образом;

– отправлять электронные письма без использования специальных программ (клиентов);

– обеспечивать другим пользователям доступ к данным, размещенным на персональном компьютере.

5. Службы имен — это набор протоколов и служб, который позволяет управлять именованием компьютеров в сети.

6. Протокол SNMP (Simple Network Management Protocol) позволяет централизованно управлять узлами сети. Он поддерживается многими типичными сетевыми устройствами, такими как маршрутизаторы, концентраторы, мосты, коммутаторы, серверы, рабочие станции, принтеры, модемные стойки и другие сетевые компоненты и устройства. Кроме того, SNMP можно использовать для конфигурирования удаленных устройств, мониторинга производительности сети, выявления в ней ошибок и попыток несанкционированного доступа, а также для аудита использования сети. Стандарты SNMP включают протокол прикладного уровня, набор объектов данных и методологию хранения, обработки и использования объектов данных в схеме базы данных.

Протоколы можно добавлять, удалять и выборочно привязывать ко всем сетевым интерфейсам сервера. Порядок привязки протоколов определяется последовательностью, в которой они были установлены, но при этом

администратор всегда может изменить этот порядок для отдельных интерфейсов, что делает процесс управления более гибким. Например, к одному интерфейсу могут быть привязаны протоколы TCP/IP и IPX/SPX с приоритетом протокола TCP/IP, а к другому - те же протоколы, но с приоритетом IPX/SPX. Для отдельных сетевых интерфейсов, протоколов и их комбинаций можно произвольно включать или отключать сетевые службы. Это позволяет администраторам легко создавать защищенные конфигурации сети (например, отключить все сетевые службы для общедоступных интерфейсов с прямым подключением к Интернету).

Контрольные вопросы:

1. Объясните назначение протокола TCP.
2. Объясните назначение протокола IP.
3. В чем состоит сущность работы по протоколу Telnet?
4. Протокол UDP: принцип работы и применение.
5. Point-to-Point Protocol: принцип работы и применение.

1.5 Основы безопасности в сети Интернет

В мире высоких технологий сложно представить свою жизнь без компьютера. Это помощник и в работе, и в частной жизни. Интернет предоставляет ресурсы, которые 15 лет назад не считались возможными. Обширная сеть Интернет делится между сотнями миллионов пользователей. Эта особенность сети часто используется для совершения правонарушений, направленных против личности, частной собственности, нравственных устоев. В Интернете эти правонарушения совершают хакеры и создатели вирусов. Их единственная цель - вызвать хаос или нанести вред определенной компьютерной системе, а нередко и миллионам систем по всему миру. Последствия таких вмешательств могут быть разными: исчезновение важных файлов, утечка конфиденциальной информации к конкурентам, падение скорости передачи данных из-за несанкционированного подключения к сети и многое другое.

Набор основных правил безопасности в Интернете:

1. Не отключать средства защиты (firewall, антивирус) на компьютере, даже если они замедляют его работу.

2. Пользоваться только лицензионными антивирусными программами. Если нет возможности купить лицензионную программу, лучше поставить бесплатный антивирус.

3. Регулярно обновлять антивирусную программу. Обычно это происходит автоматически, но некоторые прерывают этот процесс из-за замедления в работе операционной системы. Это неправильно, новые версии компьютерных вирусов множатся каждый час, антивирус должен быть готов их обнаружить.

4. Не посещать страницы Интернета, ссылки на которые получены из неизвестных источников.

5. Быть осторожными с файлами, прикрепленными к письмам.

6. Не запускать программы, загруженные из Интернета, без проверки антивирусной программой.

7. Нужно учесть, что «генераторы ключей» имеют двойное предназначение, одно из которых невыгодно в той же мере, в какой выгодно второе.

8. Использовать разные пароли для почтового ящика, блога, социальных сетей. Для регистрации на маловажных сайтах стоит придумать дополнительный пароль.

Неприемлемые пароли:

- нельзя создавать пароль на основе личной информации, например, даты рождения;

- пароль не должен являться осмысленным словом, даже если он набит в другой раскладке;

-пароль не должен создаваться путем нажатия тривиальной последовательности клавиш - qwertyuiop123 или 1йфя2цыч3увс.

9. Не открывать файлы из подозрительных писем, в том числе от знакомых. Никогда не запускать исполняемые файлы (*.exe) из писем.

10. При переходе по ссылке в письме обязательно удостовериться в наличии «безопасного соединения». Дважды нажать на значок с изображением желтого замка и проверить, кому именно выдан сертификат безопасности. Для наибольшей надежности напечатать интернет-адрес необходимой организации в адресной строке браузера и, тем самым, войти на официальный сайт. Таким образом, можно избежать заманивания с помощью писем «от администрации» на подложные сайты, в точности копирующие официальные сайты солидных организаций, с целью незаконного сбора личной информации.

2. Основные технологии разработки web-страниц

2.1 Основы языка HTML

Глобальная сеть состоит из множества web-страниц, а язык гипертекстовой разметки HTML **(**Hyper Text Markup Language) является одним из языков для их создания. HTML необходим для правильного размещения элементов сайта на странице, разметки текстового документа. Эта разметка определяет стиль, который будет использован при выводе текста на экран монитора.

2.1.1 Синтаксис HTML

Страница web-сайта — это текстовый файл с расширением *.htm или *.html. Внутри каждой страницы хранится текст HTML вместе с тегами. **Тэг или тег** (от англ. tag) – специальное зарезервированное слово, помещенное (заключенное) внутри угловых скобок < >. При помощи тега можно определять название страницы, заголовки, абзацы, размещать картинки, а также строить гиперссылку из одной страницы в другую и т.д. Редактирование HTML-кода происходит в текстовом редакторе, например, в Блокноте, а просмотр - в браузере.

Структура тега выглядит следующим образом:

<имя тега атрибут1=” значение” атрибут2=” значение”>

Теги являются элементом HTML-языка. Они бывают **парными и одиночными**. Парными называются теги, которые имеют открытую и закрытую форму. Разница между ними в том, что в закрывающем теге после угловой скобки «<» стоит знак слеш /.

Например:

 Список (устанавливает жирное начертание шрифта), где

 - открытый тег,

 - закрытый тег.

Если не закрыть парный тег , то весь текст на странице за этим тегом будет иметь жирное начертание. При вложении тегов друг в друга закрывать каждый из них необходимо, начиная с последнего, в обратном порядке. Следующий тег, который определяет начертание, - это . Он предназначен для курсивного текста.

Пример правильного закрытия вложенного тега:

  курсивный и жирный текст

Последний открытый тег

Первый открытый тег

Все, что находится между открытым и закрытым тегом, называется **областью действия тега.** Существуют теги, которые не требуют закрытия.

Например, тег
 предназначен для перевода строки, так как браузер игнорирует обычный перевод клавишей Enter. Правильная форма перевода строки выглядит следующим образом:

Первая линия
 Вторая линия

В браузере будет иметь вид:

Первая линия

Вторая линия

Тег состоит из названия, за которым следует список атрибутов. **Атрибуты** — это специальные команды, которые расширяют действия тега. Значения атрибутов заключаются в двойные или одиночные кавычки, не чувствительны к регистру, отделяются друг от друга пробелом и указываются только внутри открытого тега.

Например, <h1 align = left> выравнивает текст заголовка первого уровня по левому краю.

2.1.2 Структура HTML-документа

Каждый HTML-документ должен начинаться с объявления типа документа. Это необходимо для того, чтобы браузер определил версию HTML-документа и корректно отобразил страницу. Для последней версии языка разметки достаточно описать так:

```
<!DOCTYPE html>
<html>
    <head>
        Заголовок HTML-документа
    </head>
    <body>
        Тело HTML-документа
    </body>
</html>
```

Гипертекстовый документ разметки состоит из заголовка и тела, которые заключены в парный тег HTML**.** Заголовок документа и некоторые другие параметры размещены внутри тега <head>. В служебное содержимое, которое располагается внутри тега <head>, входит множество различных тегов. Рассмотрим некоторые из них:

<title> - предназначен для определения названия страницы,

<meta> - задает кодировку страницы.

Тег <body> служит для основного текста и тега страницы, который виден в браузере.

Пример простого HTML-документа:

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

```
<head>
        <title> Название
        </title>
</head>
<body>
<b> Моя первая страница! </b>
</body>
</html>
```

Чтобы посмотреть, как документ выглядит в браузере, необходимо набрать исходный код документа с основными тегами в текстовом редакторе Блокнот и сохранить файл с расширением .**html.** В большинстве случаев первую страницу сохраняют как **index.html**.

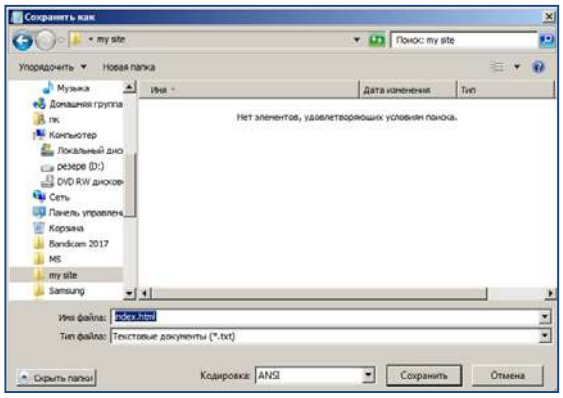


Рисунок 2.1. Как сохранить первую страницу сайта

При работе с изменениями в файле index.html помогут «горячие» клавиши: Ctrl +S (сохранение изменений), Alt+Tab (переключение в окно браузера) и F5 (обновление страницы).

При создании простейшей страницы внутри тега **<body> также** прописываются теги для заголовков, абзацев, списков, таблиц, ссылок и многие другие.

Задание:

1.В созданном документе index.html изменить начертание текста с жирного на курсив.

2.Добавить в документ фамилию, имя и отчество.

2.1.3 Работа с цветом

Многие сайты привлекают своей красочностью, гармонично подобранной цветовой гаммой, которая приятна для восприятия контента. Задать тексту определенный цвет можно двумя способами: использовав в HTML коде английское название цвета или указав его в цветовой гамме RGB (от англ. Red – красный, green – зеленый, blue – синий).

Имя цвета	Цвет	Описание	Шестнадцатеричное значение
black		Черный	#000000
blue		Синий	#0000FF
fuchsia		Светло-фиолетовый	#FF00FF
gray		Темно-серый	#808080
green		Зеленый	#008000
lime		Светло-зеленый	#00FF00
maroon		Темно-красный	#800000
navy		Темно-синий	#000080
olive		Оливковый	#808000
purple		Темно-фиолетовый	#800080
red		Красный	#FF0000
silver		Светло-серый	#C0C0C0
teal		Сине-зеленый	#008080
white		Белый	#FFFFFF
yellow		Желтый	#FFFF00

Таблица 2.1 Названия некоторых цветов и их шестнадцатеричные значения. Адаптировано с сайта (http://htmlbook.ru)

Последний способ преимущественно используется как наиболее универсальный, в основе которого лежит смешение основных цветов – красного, зеленного и синего. Перед кодом цвета обязательно ставится знак #, при этом регистр не имеет значения.

Например,

<Font color = #00FF00> Мои интересы </Font>

В языке HTML для задания основного цветового фона используется атрибут **bgcolor** тега <body>, который определяет цвет фона, указанный в значении. Например, <body bgcolor = “red”> или <body bgcolor =#FF0000> означает, что фоновый цвет будет красным.

Задание:

1. В документе index.html изменить цвет текста «**Моя первая страница и ФИО».**

2. Изменить цвет фона страницы.

2.1.4 Форматирующие теги HTML-языка.

Грамотно разработанный сайт облегчает восприятие информации пользователем. Поэтому качественное оформление текста на странице крайне необходимо. В языке HTML текст может иметь любой цвет, размер и вид. Помимо тегов <b> (от англ. bolt - жирный) и <i> (от англ. italic — курсивный) можно использовать следующие теги для работы со шрифтом и выравниванием текста:

<u> ... </u> подчеркнутый текст (от англ. underline — подчеркнуть),
_{...} отображает текст в виде нижнего индекса (от англ. subscript — индекс),
^{...} отображает текст в виде верхнего индекса (от англ. superscript – верхний индекс),
 ... выравнивает текст по центру.

Примеры использования нижнего и верхнего индексов:

X ₂
H <sup>2 <|sup.> O

Тег <Font> … </Font> - парный тег для работы с текстом, который является «контейнером» для определения характеристики шрифта и имеет атрибуты:

size – задает размет шрифта,
color – задает цвет шрифта,
face – определяет шрифт.

Пример:
<Font size=”16”>

Несмотря на то, что данный тег поддерживается браузерами, он считается устаревшим.

Абзац является одним из основных элементов страницы. Для определения начала или конца строки в языке HTML существует специальный тег (парный). Он служит для обозначения красной строки, имеет ряд атрибутов, расширяющих его действие, и описывается следующим образом:

 Новый абзац
Атрибут align = “…” – определяет режим выравнивания текста
Left – по левому краю (установлен по умолчанию),
Right - по правому краю,
Center – по центру,
Justify – по ширине.

При создании web-страницы требуется правильное оформление заголовков. Теги <h1>, <h2>, <h3>, <h4>, <h5>, <h6> служат для создания заголовка, имеющего по умолчанию жирное начертание. Теги различаются размером устанавливаемого шрифта.

Пример:
<body>
<h1> Рубрика 1 </h1>
<h2> Рубрика 2 </h2>
<h3> Рубрика 3 </h3>
<h4> Рубрика 4 </h4>
<h5> Рубрика 5 </h5>
<h6> Рубрика 6 </h6>
</body>

Тег <HR> (от англ. horizontal rule) служит для создания горизонтальной разделительной линии. Этот тег начинает строить линию с новой строки, относится к блочным элементам и не требует закрывающего тега. <HR> имеет следующие атрибуты:

align = “…” – определяет режим выравнивания текста

Left – по левому краю,

Right - по правому краю,

Center – по центру,

color – цвет линии,

noshade – рисует без трехмерных эффектов,

width – ширина линии, значение задается в процентах или в пикселях,

size – толщина линии, указывается в пикселях.

Пример: <HR noshade size= 4 width =”50%”>

Размеры объектов в HTML часто указываются в пикселях. Пиксель – это точка, логический размер дисплея. Размер задается именно в пикселях и называется разрешением экрана, например, 1024 по горизонтали и 768 по вертикали.

Задание:

1. Используя теги, приписать заголовок «Моя первая страница».
2. Изменить начертание текста «Моя первая страница» на курсив.
3. ФИО изменить на жирное начертание, подчеркнуть текст и расположить по правому краю.
4. Под текстом «Моя первая страница» прописать горизонтальную линию толщиной равной 2, установить цвет, который гармонирует с фоном.

2.1.5 Работа со списками

Списки так же, как заголовки и абзацы, являются важным атрибутом при построении web-страницы. Использование нумерованных и маркированных списков часто встречается при разработке сайта, когда есть необходимость создать какой-либо перечень или перечислить что-то по пунктам. Маркерным называют список, где вместо цифр используется закрашенный круг или квадрат, который часто именуется маркером.

Маркированные или неупорядоченные списки создаются при помощи парного тега <ul> (от англ. unordered list), который задает список. Внутри этого тега прописывается элемент списка - тег <li> (от англ. list item), который в свою очередь отвечает за пункты перечня. Каждому пункту необходимо прописать отдельный элемент списка. Вложенный тег <li> имеет следующие атрибуты:

type="…" – формат номера или маркера (см. описание <ol> и <ul>), value="N" – задает номер элемента списка.

Пример:

```
<ul>
<li type=square> квадратный маркер </li>
<li type=circle> круглый маркер </li>
</ul>
```

Атрибутами ненумерованного тега **<ul>** являются:

type=” …” – определяет формат маркера

disk – диск (установлен по умолчанию),
square — квадрат,
circle – окружность.

В языке гипертекстовой разметки существуют еще и упорядоченные списки с нумерованными элементами. Такие списки создаются при помощи тега <ol> (от анг. ordered list), а пункты так же, как и в неупорядоченных, описываются через тег <li>.

Пример:

```
<body>
    <ol>
        <li> Первый </li>
        <li> Второй </li>
        <li> Третий </li>
    </ol>
</body>
```

Атрибутами ненумерованного тега <ol> являются:

Start= “N” – когда есть необходимость начать нумерацию с определенной цифры,

type=” …” – определяет формат нумерации.

1 – арабские цифры (установлены по умолчанию),

A – заглавные буквы (A, B, C),

a – строчные буквы,

I – заглавные римские цифры (I, II, III, IV),

i – строчные римские цифры (i, ii, iii, iv).

Преимущество упорядоченных списков в том, что нумерация перестраивается автоматически, т.е. нет необходимости прописывать ее вручную, и исключена возможность ошибиться в последовательности.

Пример:

```
<ol type =” I”>
<li> Техника </li>
<li> Бытовая Техника </li>
</ol>
<ul>
<li type=” square”> Холодильник </li>
<li type=” square”> Стиральная машина </li>
</ul>
```

Задание:

1.Создать новый HTML-документ и назвать его interes.html.

2.Создать список интересов или увлечений, используя нумерованный и маркированный теги.

3.Изменить нумерованный список на буквенный.

4.Используя разные типы маркера, изменить список на маркированный.

2.1.6 Создание ссылок

Ссылки являются элементами, которые дают возможность перемещаться по сайту, создавая связь между его страницами. Интернет без ссылок был бы набором страниц, не связанных друг с другом. Для создания ссылки используется парный тег <a>**.** Внутри тега необходимо прописать обязательный атрибут - **href**, который хранит в себе относительный или абсолютный адрес страницы, на которую ведет ссылка. Абсолютный адрес состоит из имени хоста и полного пути ресурса, что дает возможность ссылаться на любой открытый интернет-ресурс.

Пример создания абсолютной ссылки:

 Ссылка на сайт mywebsite.com

Также для соединения двух HTML–документов можно использовать относительный адрес. Если два документа находятся в одной и той же папке, код будет выглядеть следующим образом:

 Ссылка на страницу - ссылка на первый документ

 Ссылка на страницу - ссылка на второй документ

Ссылки делятся на абсолютные и относительные. Ссылаться можно, как на внешние, так и на внутренние страницы сайта.

_self– открывает страницу в текущем окне,

_parent – открывает страницу во фрейме,

_blank – открывает страницу в новом окне,

_top – открывает страницу в новом окне, при этом игнорируя созданные фреймы.

В роли ссылки может выступать не только текст, но и **картинка.** Для этого необходимо в теге **<a>** прописать тег **<img>. Ч**тобы просмотреть, как это выглядит, достаточно в браузере набрать следующий код:

 <img src=" winter.jpg ">

В этом случае вокруг изображения автоматически появляется рамка. Толщина рамки задается атрибутом border. Обычно рамку убирают, указывая border="0" в теге <img>.

Задание:

1.Создать гиперссылку со страницы index.html на interes.html .

2.На странице interes.html добавить гиперссылку на страницу index.html.

2.1.7. Работа с рисунками и графикой

Изображения на странице делают сайт более насыщенным, приятным к восприятию, придавая ему особые очертания и окрас. Разместить фото, картинку, рисунок на странице при помощи языка HTML можно, используя тег <img>. При этом путь к файлу, где размещено изображение, хранится в обязательном атрибуте src.

Пример:

<img src="winter.jpg">

На web–страницах чаще всего представлены следующие форматы изображений: GIF, JPG, PNG. Рисунки могут использоваться как фон страницы или отдельно взятых ее элементов, в том числе гиперссылок. Все графические файлы рекомендуется хранить в той же папке, что и сайт. Если изображения хранятся в другой папке, необходимо прописать полный путь файла. Для того чтобы расширить возможности работы с изображениями, необходимо учитывать особенности форматов графических файлов, которые представлены в следующей таблице:

Формат	**Достоинства**	**Недостатки**	**Применение**
GIF	- Обеспечивает высокую степень сжатия, - может хранить анимацию, - позволяет оставлять участки рисунка прозрачными.	- максимальное количество цветов 256, - прозрачность в GIF однобитная, т.е. каждый пиксель изображения либо прозрачен, либо виден; это не дает возможности создавать переход между фоном и рисунком, - значительный вес файла с анимацией, - плохо масштабируется.	Баннеры, кнопки, небольшие изображения с прозрачным фоном, разделительные линии и т.п.
PNG	- минимальные потери при высокой степени сжатия, - большая глубина цвета, - неограниченные возможности при работе с прозрачными областями.	- не поддерживает анимацию.	- самое широкое при создании полупрозрачных эффектов: градиентной заливке, наложении полупрозрачных областей на рисунок, водяных знаков, трафаретных изображений.
JPG	- поддерживает высокую степень сжатия, - позволяет	- высокая степень сжатия может отрицательно отразиться на	- применяется в основном для хранения фотографий.

	хранить изображения с количеством цветов – 16.7 млн.	возможностях редактирования (нет доступа к отдельным пикселям), - хорошо масштабируется, - не поддерживается прозрачность.	

Таблица 2.2 Особенности графических форматов
(Адаптировано из учебника Информатика, Е.А. Вьюшкова, Н.В. Параскун)

Атрибуты тега <img>:
Align = "…" определяет режим выравнивания изображения
bottom – по нижнему краю (по умолчанию),
top – по верхнему краю,
middle – по центру строки,
left – по левому краю окна,
right – по правому краю окна.
alt="…" – определяет текст, описывающий изображение, для браузеров без
поддержки графики (или с отключенной графикой), поисковых машин и т.п.
border="N" – устанавливает толщину рамки вокруг изображений, равной N пикселей, 0 – для отключения рамки;
height="N" – высота изображения в пикселях или процентах;
width="N" – ширина изображения в пикселях или процентах.

В языке HTML существует способ установки основного фона путем вставки рисунка. Тег **<body>** имеет атрибут, который работает с фоном страницы: **Background.** Он устанавливает фоновое изображение, которое выводится в полную величину. Например, <body background = "1.jpeg">
Изображение будет повторяться в окне браузера в том случае, если его размер меньше размера окна браузера. Поэтому необходимо заведомо выбрать изображение с соответствующими параметрами.

Пример:

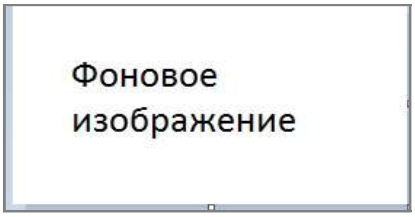


Рисунок 2.2 Файл 1.jpeg

<html>
<head>
<title> Тестирование фонового изображения </title>
</head>
<body background="1.jpeg">
</body>
</html>

Результат в браузере будет следующим:

Рисунок 2.3 Вид фонового изображения в браузере

Задание:

1. В документ index.html под ФИО добавить свою фотографию.
2. Используя атрибуты тега <img>, изменить размер рисунка.
3. Добавить фоновое изображение в документ interes.html.

2.1.8 Комментарии HTML

Комментарии - это текст, который игнорируется браузером, на экране он не виден, но останется в исходном коде страницы.

Комментарии необходимы для того, чтобы:

- оставлять заметки в исходном коде для более удобного понимания текста,
- с их помощью из кода можно убирать то, что необходимо удалить, но потом вернуть при надобности.

Комментарии в HTML оформляются следующим образом:

<! -- текст комментария --->

```
<body>
        <! -- комментарий HTML --->

        Моя первая страница
</body>
```

2.1.9 Работа с таблицами

Таблица состоит из строк и столбцов, на их пересечении находятся ячейки.

При построении таблицы необходимыми тегами являются:

<table> является основным тегом,

<tr> - создает ряды или строки таблицы,

<td> - парный тег, который создает ячейки и работает в паре со следующими атрибутами:

Nowrap - служит для размещения содержимого ячейки в одной строке,

Colspan – служит для горизонтального объединения ячеек (пример: Colspan =3 - означает, что необходимо объединить 3 ячейки),

Rowspan – вертикальное объединение ячеек,

Align – выравнивание текста по горизонтали,

Valign - выравнивание текста по вертикали,

Width – задает значение ширины ячейки в пикселях,

Height– задает значение высоты ячейки в пикселях.

В языке HTML таблица создается рядами (первый, второй), не существует тега, который создавал бы столбцы.

Тег <table> имеет атрибут border, который определяет толщину линии границы таблицы.

```
<table border="1">
<! -- Первый ряд таблицы: -->
        <tr>
            <td>Строка 1</td>
            <td>Строка 2</td>
            <td>Строка 3</td>
        </tr>
<! --Второй ряд таблицы: -->
        <tr>
            <td> Строка 4</td>
            <td> Строка 5</td>
            <td>Строка 6</td>
        </tr>
</table>
```

При создании ячеек-заголовков применяется специальный тег <th>, который, в отличие от <td>, предназначен именно для заголовков.

Следующий пример показывает различие между двумя тегами для создания ячеек таблицы:

```
<table border="2">
        <tr>
            <th> Фамилия </th>
            <th> Имя </th>
            <th> Отчество </th>
        </tr>
        <tr>
```

```
            <td>Ахметов </td>
            <td>Аскар </td>
            <td> Маратович </td>
        </tr>
</table>
```

В браузере данная таблица выглядит следующим образом:

Фамилия	**Имя**	**Отчество**
Ахметов	Аскар	Маратович

Как видно, по умолчанию текст в ячейках <th> выровнен по центру и имеет начертание жирным шрифтом.

Атрибуты cellpadding и cellspacing тега <table>

Важно, что каждая ячейка в таблицах имеет границу, а между ячейками существует некоторое расстояние. Это расстояние регулируется атрибутом **cellspacing**. По умолчанию данный атрибут имеет ненулевое значение, из-за чего ячейки не сливаются.

Следующий пример показывает использование атрибута cellspacing и изменение его значения на 15 пикселей:

```
<table border="1" cellspacing="15">
    <tr>
        <th>Фамилия</th>
        <th>Имя</th>
        <th>Отчество</th>
    </tr>
    <tr>
        <td>Ахметов</td>
        <td>Аскар</td>
        <td> Маратович </td>
    </tr>
</table>
```

Необходимо задать нулевое значение cellspacing, если значение по умолчанию мешает созданию корректной таблицы. Атрибут cellspacing в версии HTML5 считается устаревшим. Однако большинство сайтов созданы в более ранних версиях HTML, и разработчику необходимо знать, как работать с такими атрибутами.

Атрибут **cellpadding** создает отступ между границей ячейки и текстом. По умолчанию атрибут имеет некоторое присвоенное ему значение. Если есть потребность — это игнорировать, то необходимо обнулить значение. В следующем примере рассматривается работа атрибута, где его значение равно 15px:

```
<table border="1" cellpadding="15">
    <tr>
```

```
        <th>Фамилия</th>
        <th>Имя</th>
        <th>Отчество</th>
    </tr>
    <tr>
        <td>Ахметов</td>
        <td>Аскар</td>
        <td> Маратович </td>
    </tr>
</table>
```

В браузере данная таблица выглядит следующим образом:

Фамилия	**Имя**	**Отчество**
Ахметов	Аскар	Маратович

Обнуление значений атрибутов cellpadding и cellspacing выглядит так:
<tableborder="2" cellpadding="0" cellspacing="0">

Работа со значениями ширины и высоты таблицы

Существуют также атрибуты **width** и **height**, которые задают соответственно **ширину** и **высоту** таблицы. При отсутствии данных параметров размер регулируется содержимым ячеек, т.е. чем больше текста, тем крупнее размер ячейки. Значения атрибутов могут быть заданы как в пикселях, так и в процентах. В процентном соотношении значения задаются следующим образом:

height="40%" - в данном случае таблица займет **40%** высоты родителя.

Пример:

```
<table border="2" cellpadding="0" cellspacing="0" width="400" height="400">
    <tr>
        <th>Фамилия</th>
        <th>Имя</th>
        <th>Отчество</th>
    </tr>
    <tr>
        <td>Ахметов</td>
        <td>Аскар</td>
        <td> Маратович </td>
    </tr>
</table>
```

Группировка ячеек таблиц

Объединять ячейки можно по горизонтали и по вертикали. Атрибут **colspan** служит для объединения столбцов, хотя на самом деле этот атрибут расширяет ячейку по горизонтали, а не объединяет. Например, значение

colspan="3" расширит размер ячейки по ширине как три, а colspan="4" - как четыре.
Наглядный пример в виде таблицы представлен ниже:

<table>
<tr><td rowspan="3">Cell 1</td><td colspan="2">Cell 2</td></tr>
<tr><td>Cell 4</td><td>Cell 6</td></tr>
<tr><td>Cell 7</td><td>Cell 9</td></tr>
</table>

Расширение ячейки Cell 1 на два столбца

Расширение ячейки **Cell1** на два столбца при помощи атрибута **colspan** в значении **2:**

```
<table>
    <tr>
        <td colspan="2">Cell 1</td>
        <td>Cell 2</td>
        <td>Cell 3</td>
    </tr>
    <tr>
        <td>Cell 4</td>
        <td>Cell 5</td>
        <td>Cell 6</td>
    </tr>
</table>
```

При таком кодировании видно, что Cell 1 вытеснила ячейки справа, причем Cell 3 вышла за рамки таблицы.

Результат объединения выглядит в браузере следующим образом:

<table>
<tr><td colspan="2">Cell 1</td><td>Cell 2</td><td>Cell 3</td></tr>
<tr><td>Cell 4</td><td>Cell5</td><td>Cell 6</td><td></td></tr>
</table>

Расширение Cell 1 на два столбца с сохранением целостности таблицы

Чтобы поправить проблему с целостностью таблицы из предыдущего примера, необходимо удалить одну из близстоящих ячеек. В данном примере это ячейка:

```
<table>
    <tr>
        <td colspan="2">Cell 1 </td>
        <td>Cell 2 </td>
        </tr>
    <tr>
        <td>Cell 4 </td>
        <td>Cell 5 </td>
        <td>Cell 6 </td>
```

```
    </tr>
</table>
```

Таблица в браузере выглядит следующим образом:

Cell 1		Cell 2
Cell 4	Cell 5	Cell 6

Объединение строк

Атрибут rowspan аналогичен атрибуту colspan. Принцип их работы одинаков, за исключением того, что rowspan расширяет ячейки по вертикали. В следующем примере с помощью задания атрибуту rowspan значения равного двум объединится Cell1, при этом разрушится целостность таблицы:

```
<table>
    <tr>
        <td rowspan="2">Cell 1 </td>
        <td>Cell 2 </td>
        <td>Cell 3 </td>
    </tr>
    <tr>
        <td>Cell 4 </td>
        <td>Cell 5 </td>
        <td>Cell 6 </td>
    </tr>
</table>
```

Таблица потеряла целостность:

Cell 1	Cell 2	Cell 3	
	Cell 4	Cell 5	Cell 6

Расширение ячейки Cell1 с сохранением целостности ряда

Чтобы исправить проблему с разрывом таблицы из предыдущего примера, необходимо удалить одну ячейку из второго ряда, в данном случае это ячейка под номером 5:

```
<table>
    <tr>
        <td rowspan="2"> Cell 1</td>
        <td> Cell2</td>
        <td> Cell 3</td>
    </tr>
    <tr>
        <td>Cell 4</td>
        <td>Cell 6</td>
```

</tr>
</table>

Так таблица выглядит в браузере:

Cell 1	Cell 2	Cell 3
	Cell 4	Cell 6

Задание:

1. В документ index.html добавить таблицу распорядка дня на неделю.
2. Установить ширину границы на 2.
3. Задать заголовок таблицы.
4. Изменить фон ячеек, содержащих дни недели.

2.1.10 Создание элементов формы

Теги **<input>, <select> и <textarea>** создают поля формы**.** При обработке формы на сервер передаются значения. Name – определяет имя поля, а Value -задает значение поля. Атрибутом тега <input> является type. Он определяет тип поля и может иметь следующие значения:

File – дает возможность приложить файл,

Password – создает поле, где вводимые пользователем значения отображаются в виде * (звездочки), тем самым скрывая его значение.

  Surname: <input type=”text” name= “name”>

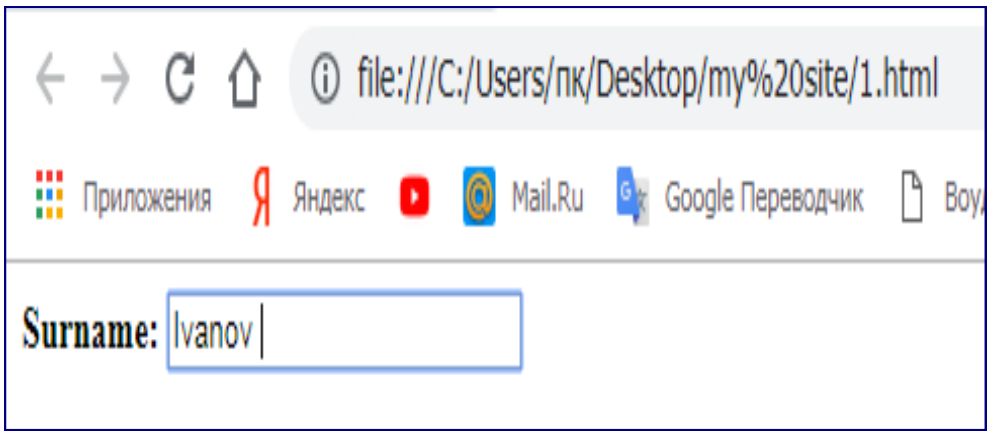


Рисунок 2.4 Поле формы

Переключатели (радиокнопки) также создаются тегом input. Под радиокнопкой подразумевается единственный выбор из всех возможных предложенных вариантов ответа. Например, в одном вопросе, где предлагаются варианты, присваивается одинаковое имя (name), а значение Value будет разным. Атрибут checked будет установлен по умолчанию:

  Укажите Ваш пол:
<input name="gender" type="radio" value="female"> Женский
<input name="gender" type="radio" value="male" checked> Мужской

Списки при создании формы задаются тегом <select name = “list” size=” n”>, если значение атрибута size равно 1, иначе список будет выпадающим. Тегом <option> задается значение элементов списка, которое может быть неограниченным.

```
<p> <b> Укажите Ваш возраст: </b>
<select name =” select”>
<option> 10-18 </option>
<option> 18 -25 </option>
<option> 25-30 </option>
</select>
```

Для создания **флажков** используется значение checkbox атрибута type. Пользователи таких форм могут одновременно установить несколько флажков, как и для переключателей, основными свойствами являются Name, Value и Checked

```
<p> <b>Ваши интересы:
<p><input type="checkbox" name="a" value="Computer"> Компьютер</p>
<p><input type="checkbox" name="a" value="Math" checked> Математика</p>
<p><input type="checkbox" name="a" value="Languages" checked> Языки</p>
<p><input type="checkbox" name="a" value="Education"> Образование</p>
<p><input type="checkbox" name="a" value="Nothing"> Ничего из этого списка</p>
```

Существует четыре вида кнопок в языке гипертекстовой разметки:

submit - отправка содержимого формы на сервер,

reset - сброс введенных данных,

image - графическая кнопка.

button - действие этой кнопки назначается с помощью сценария на языке JavaScript.

```
<p><input type="button" value=" отправить ">
<input type="button" value=" очистить "> </p>
```

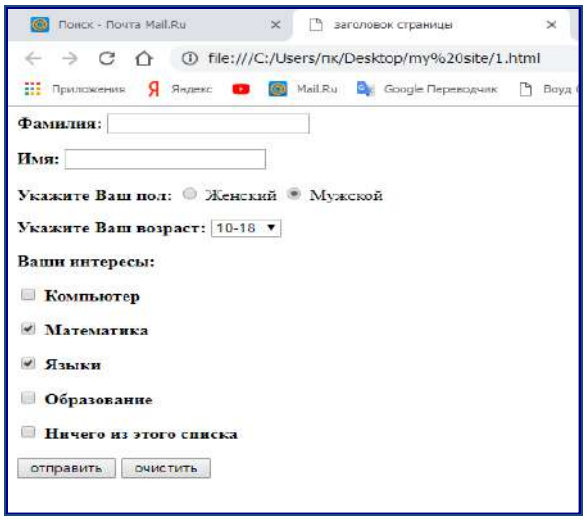


Рисунок 2.5 Работа с формами

Задание:

1.В текстовом документе interes.html добавить форму Хобби.

2.Создать текстовые поля для ввода ФИО.

3.Создать выпадающий список для перечня хобби.

2.2 Технология каскадных таблиц стилей CSS

После знакомства с языком HTML большинство начинающих разработчиков считает, что этого вполне достаточно для создания полноценного сайта. Однако инструментов языка разметки недостаточно для оформления полномасштабного web–документа. Действительно, HTML это начало процесса по созданию web–страниц. **С**ледующим шагом является изучение **стилей** или **CSS (Cascade Style Sheets).** Параметры CSS дают возможность управлять положением, видом и используются для изменения стиля документа.

Разработка сайта с использованием языка разметки HTML позволяет создать статические сайты, а технология каскадных стилей CSS, в свою очередь, предоставляет возможность оформить их по задуманному дизайну. Знание этих двух технологий составляет основу для дальнейшего изучения серверных языков программирования.

CSS - Cascading Style Sheets («Каскадные таблицы стилей») позволяют задавать различные визуальные свойства HTML-элементам. Каскадные таблицы стилей CSS предназначены для создания HTML-страниц сайтов с красочным дизайном. Обычно в HTML содержимое страницы отображается на экране сверху вниз, один элемент следует за другим. CSS позволяют помещать изображения или текст в любом месте на странице рядом друг с другом, поэтому web-страницы имеют более интересный внешний вид.

Термин «каскадные таблицы стилей» был предложен Хокон Виум Ли в 1994 году. Совместно с Бертом Босом он стал развивать CSS.

Стиль в CSS - это правило, устанавливающее внешний вид какого-либо элемента или фрагмента web-страницы. Это дает возможность форматирования: изменения цвета шрифта заголовка на нужный, выделения фотографии рамкой, создания меню.

Таблица стилей – это шаблон, управляющий форматированием тегов HTML в web-документе и состоящий из набора правил описания стиля.

Стили представляют собой набор параметров, которые определяют вид и положение элементов web-страницы. Стили обычно хранятся в таблицах: они могут быть определены как внутри HTML-документа, так и в специальном файле с расширением ***.css**. Используя отдельные файлы для хранения таблиц стилей можно существенно сократить объем работы при написании кода программы.

Определение стиля состоит из двух основных элементов: **селектора** и **блока объявления**. Селектор сообщает браузеру, к какому элементу (заголовку, абзацу, изображению или гиперссылке) web-страницы применяется стиль. Блок объявления стиля - это код, расположенный сразу за селектором и содержащий все форматирующие команды, которые можно применить к этому селектору. Блок пишется внутри фигурной скобки { }.

Например, цвет текста абзаца может быть задан с помощью стиля как:

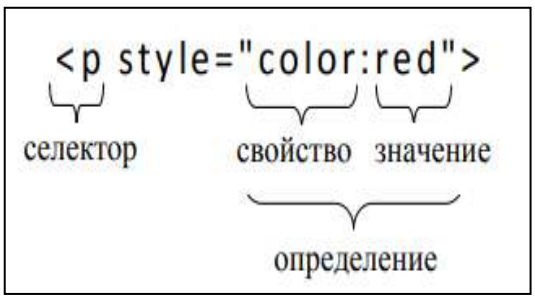


Рисунок 2.6 Правила таблиц стилей

Данная запись означает, что все абзацы будут набраны красным шрифтом. Здесь «p» — это атрибут, слово color является свойством объявления, а red— значением.

После каждого свойства необходимо ставить двоеточие. После каждого значения — точку с запятой.

Различают **простые селекторы,** которые будут применены к указанному элементу (в примере к любому заголовку h1, h2, h3):

h1 { font-family: sans-serif }
h2 { font-family: sans-serif }
h3 { font-family: sans-serif }

Группы селекторов (равно сильный выше приведенному фрагменту):

h1, h2, h3 { font-family: sans-serif }

Селекторы класса: *.pastoral {color: green} /* все элементы, имеющие class=pastoral */

Шрифты. При оформлении страницы в CSS доступны следующие семейства шрифтов:

- Serif - шрифты с засечками. Применяется для бумажной печати. Используется шрифт – Times;

- Sans-serif - шрифты без засечек. Используется для оформления заголовков. Наиболее используемые шрифты– Arial, Helvetica, Verdana;

- Monospace - шрифт, обеспечивающий одинаковую ширину символов. Используется для вывода примеров кода, поскольку внешний вид этого текста будет соответствовать текстовой консоли. Наиболее используемый шрифт – Courier;

- Fantasy, Cursive – применяется для декоративных и курсивных шрифтов и не рекомендуется к использованию, поскольку шрифты этой группы редко присутствуют на компьютере, на котором будут просматривать HTML-страницу. Выбор семейства шрифта осуществляется свойством font-family. Примеры отображения шрифтов:

Serif: Пример
Sans-serif: Пример
Cursive: Пример
FaNTaSY: Пример

Monospace: Пример

Цвет. Для создания дизайна сайта очень важным является выбор цвета и цветовых схем. Для назначения цвета шрифта используется свойство color. Цвет можно указать по названию (red, green, lime и др.), а также по его значению в системах RGB, HSL.

Пример:

body {color:blue;}

h1 {color:#00ff00;}

h2 {color:rgb(255,0,0);}

Фон элементов может быть задан однородным цветом, отдельным или мозаично расположенным изображением. Для этого используются следующие свойства: background-color – однородный цвет с константой или кодом в одной из допустимых систем цветности.

Пример: div {background-color:#b0c4de;}

background-image – фоновое изображение.

Пример: body {background-image:url('paper.gif');}

background-repeat – флаг узорчатое размножения изображения.

Пример: body {background-image:url('gradient2.png'); background-repeat:repeat-x; }

background-attachment – указывает, будет ли изображение смещаться или останется на месте.

Пример: background-attachment: fixed; 28

background-position – указывает позицию размещения изображения на устройстве отображения.

Типы каскадных таблиц стилей

Каскадные таблицы стилей используются при задании единого стиля оформления разных страниц web-документа, а также для его быстрого редактирования. В зависимости от того, где определена стилевая информация, связанная с web-страницей, таблицы стилей разделятся на два вида: внутренние (в самой web-странице) и внешние (в отдельном файле с расширением .css).

Каскад - это иерархия применения правил или процесс, с помощью которого сбиваются конфликтующие объявления. Рассмотрим типы таблиц стилей и способы применения стиля к одному и тому же web-документу.

1. Встроенный стиль (inline CSS)

CSS позволяют назначить собственный стиль визуального представления любому тегу HTML, в том числе тегу <body>. Если стиль назначен для тега <body>, он наследуется всеми элементами (абзацами, заголовками), распределенными внутри этого тега-контейнера, в случае отсутствия собственных стилей для этих элементов. Таким образом, не нужно 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 ним.

Встроенный стиль определяется непосредственно в теге с использованием атрибута style. Такой подход полезен, только если стиль одновременно применяется к элементу.

Каждый тег может иметь параметр style, который определяет свойства тега. Пример написания абзаца с зеленым текстом и 18 кеглем выглядит так:

 Green text

Название	*Описание*	*Значения*
font-family	гарнитура шрифта	
font-style	начертание шрифта	обычный: normal курсивный: italic наклонный: oblique
font-weight	толщина шрифта	100, 200, ... 900
font-size	размер шрифта	
text-align	выравнивание текста относительно элемента, в котором он находится	по левому краю: left по правому краю: right по центру: center по ширине: justify
vertical-align	положение элемента по вертикали относительно родительского элемента	

Таблица 2. 3. Свойства форматирования текста

Такое применение стилей дает возможность сокращения тегов <font> вместе с атрибутами, что уменьшает размер файла. В приведенном примере используется вставка стиля непосредственно в тег документа - так называемый inline-стиль. Этот способ связывания CSS с HTML-файлом рекомендуется в редких случаях: если данный стиль планируется применить только на одной странице сайта и только к одному элементу.

Недостаток такого способа - трудоемкость создания и коррекции документа, его необходимо использовать только для ограниченного числа элементов.

Задание №1

1. Создайте внешний CSS файл. Подключить его к страницам index.html и interes.html.
2. Увеличить размер шрифта, задать для тега BODY фон свойством background-color и границу толщиной 5px.
3. Проверить созданный документ.

2. Внутренняя или внедренная таблица стилей (Internal CSS). Внедренный стиль управляет представлением одного документа и

размещается внутри тега <style>...</style> в разделе head HTML-документа. Например:

```
<head>
...
<style type="text/css">
p {background-color: blue; color: green; text-align: left;}
</style>...
</head>
```

В результате во всем документе будет изменен цвет фона, цвет текста и выравнивание текста в теге <p>.

Задание №2

1.В документе index.html создать внедренную таблицу стилей, изменить свойства тега <p>: размер шрифта – 12 pt, полужирный, выравнивание по центру, цвет - красный, фон текст - голубого цвета.

2. Убедиться в том, что встроенный стиль имеет максимальный приоритет.

3. Внешняя или связанная таблица стилей (External CSS). Внешняя таблица стилей связана с HTML-документом при помощи тега <link>, размещенного в разделе документа head. Любой документ, связанный с данным типом таблицы стилей, получает все стили, определенные в ней. В этом заключается преимущество управления языка CSS.

Такая таблица стилей находится в отдельном текстовом файле с расширением CSS. Документ CSS - это отдельный документ, который создается с помощью текстового редактора и сохраняется как стандартный текстовый файл с расширением .css (например, styles.css). Он должен находиться в той же папке, что и web-документ. Тогда таблица стилей подключается с помощью тега <link> в разделе <head>:

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

Параметр href определяет путь к файлу CSS. Если таблица находится в той же папке, что и web-документ, то это просто href="styles.css", где styles – имя файла CSS.

Эту строку можно прописать в любом из HTML-файлов. Таким образом, единое стилевое оформление будет прописано для нескольких страниц сразу.

Создание файла CSS. Для связи CSS и HTML помимо встраивания и внедрения используются способы импортирования и связывания таблиц стилей. Это наилучшие способы для придания единого стилевого оформления нескольким или всем страницам одного сайта. При этом вся таблица стилей хранится в одном файле (его расширение должно быть стандартным - .css).

В текстовый документ информация вводится следующим образом:

Имя тега {свойство: значение;}

Например, файл такого содержания определяет серебристый фон страницы:

body {background-color: silver;}

Название	*Описание*	*Значения*
color	цвет текста	любое соответствующее стандарту значение цвета (название цвета в цветовой модели RGB)
background-color	цвет фона	любое соответствующее стандарту значение цвета
background-image	рисунок в качестве фона	указывается имя файла background-image: url(имя файла)
background-repeat	повторяемость для фона, заданного изображением	по горизонтали: repeat-x по вертикали: repeat-y не повторяется: no-repeat по умолчанию повторение и по горизонтали, и по вертикали.
background-attachment	неподвижность фона при прокрутке (по умолчанию фон прокручивается)	fixed (неподвижность фона)
background-position	положение изображения относительно верхнего левого угла элемента	задается в процентах от размера элемента (первое число – смещение по горизонтали, второе - по вертикали)

Таблица 2.4 Свойства цвета и фона

4. Импортированная таблица стилей

Импортированная таблица стилей *похожа на внешнюю таблицу стилей и имеет код, записанный в отдельном текстовом файле с расширением* **.css**. *Однако импортированная таблица связывается с помощью инструкции @ import: url(styleimport.css); либо с внешней таблицей стилей, либо непосредственно с HTML-документом в начале блока* <style>.

Задание №3

1. В файле interes.html задать в таблице стилей (в файле CSS) фоновый рисунок, определить повторение рисунка по вертикали.
2. Указать цвет фона страницы подходящего к фоновому рисунку
3. Посмотреть получившийся результат в браузере.

Свойства стиля оформления полей. Для определения внешних отступов элементов страницы от внешней границы web-сайта используется свойство margin.

Например:

margin-top
margin-right
margin-bottom
margin-left

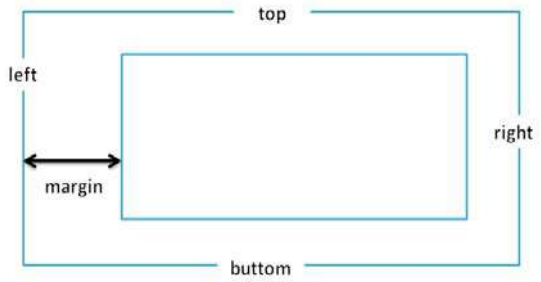


Рисунок 2.7 Отступы и рамки

Селекторы CSS

При необходимости изменения внешнего вида web-страницы, например, ее шрифта или цвета всех тегов <h1> ,нужно создать стиль. Написав один такой стиль, можно изменить шрифт сразу для всех тегов <h1> на странице вне зависимости от их количества.

Селекторы класса позволяют создавать стиль для одного конкретного элемента либо определенной группы элементов web-страницы. Название класса указывается после наименования элемента и отделяется точкой. Рассмотрим пример такой страницы с интервью, где шрифты вопросов журналиста будут отличаться от ответа, интервьюируемого:

```
<html>
<head>
...
<style>
...
p.vopros {font-style: italic;
font-weight: bold;
font-family: Arial, sans-serif;
font-size: 12pt;
color: green;
margin-left: 18px;}
p.otvet {font-family: 'Times New Roman', serif;
font-size: 14 pt; color: blue;}
...
</style>
...
</head>
<body>
```

```
...
<p class=" vopros ">Вопрос журналиста</p>
<p class=" otvet ">Ответ интервьюируемого</p>
...
</body> </html>
```

В данном примере вопросы будут отображаться шрифтом Arial зеленого цвета, полужирным, курсивного начертания, размером 12 пунктов с отступом 18 пикселей от левого края страницы, а ответы - шрифтом Times New Roman синего цвета, размером 14 пунктов.

Селектор id

Селектор id используется при необходимости создания стиля для уникального элемента web-страницы, обращаясь к нему через ID (идентификатор). Идентификатор – уникальное название элемента web-страницы, которое присваивается с помощью атрибута id. Его удобно использовать в том случае, если элементы имеют одинаковое оформление посредством CSS и могут повторяться на других страницах. Он вставляется после символа решетка «#». Пример назначения идентификатора и правил CSS таким элементам:

```
<html>
<head>
...
<style>
...
input#green {color: green;}
input#red {color: red;}
...
</style>
...
</head>
<body>
...
<form ...>
<p>Текст будет отображен зеленым цветом:
<input type="text" id="green" name="info1" size="20"></p>
<p>Текст будет отображен красным цветом:
<input type="text" id="red" name="info2" size="20"></p>
</form>
...
</body>
</html>
```

Аналогичным образом уникальные идентификаторы могут быть назначены любому количеству разных элементов страницы. Это может быть полезно для обращения к элементу из программы на языке JavaScript и

изменения стиля его отображения в ответ на действия пользователя, что позволяет создавать различные динамические эффекты.

Селекторы групп

Селекторы групп используются при необходимости изменения шрифта на курсив для нескольких элементов web-страницы – h1, h2, h3,p. Если записывать стиль к каждому тегу по отдельности, на это уйдет больше времени и места. В таком случае удобно использовать групповой селектор, например:

h1, h2, h3, p {font-weight: italic;}

Практическая работа № 1

Тема: технология HTML и каскадные таблицы стилей CSS.

Цель: создание и подключение каскадной таблицы стилей к странице HTML, закрепление полученных знаний по использованию технологий HTML и CSS при создании web-сайта.

Ход работы:

1. Создать папку «css», в которой будут HTML-страница и каскадная таблица стилей.
2. В данной папке «css» с помощью Блокнота создать текстовый документ index.txt.
3. Создать простой текстовый документ (аналогично index.txt), но назвать его style.css.
4. Для подключения созданной таблицы стилей к файлу index.html в этом файле в теге <head> прописать следующий тег:

<linк href="style.css" rel="stylesheet" type="text/css">

В атрибуте href="style.css" прописан путь к самому файлу со стилями CSS. В данном случае файл CSS и index.html находятся в одной папке. В результате должно получиться так:

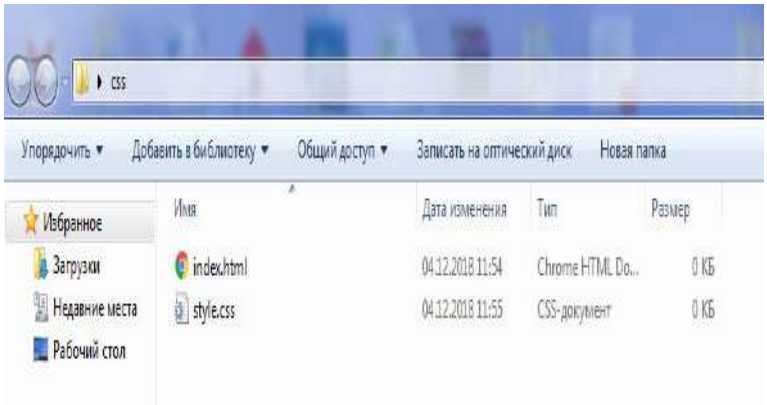


Рисунок 2.8 Создание web-страниц

Например, файл index.html будет иметь такой код:

<html>
<head>

```
<meta http-equiv="Content-Type" content="text/html; charset=windows-1251">
<title> Каскадные таблицы стилей css</title>
<meta name="description" content="Каскадные таблицы стилей">
<meta name="keywords" content="стили, таблицы, css">
<link href="style.css" rel="stylesheet" type="text/css">
</head>
<body>
<h2>Web страница для изучения каскадных таблиц стилей css </h2>
</body>
</html>
```

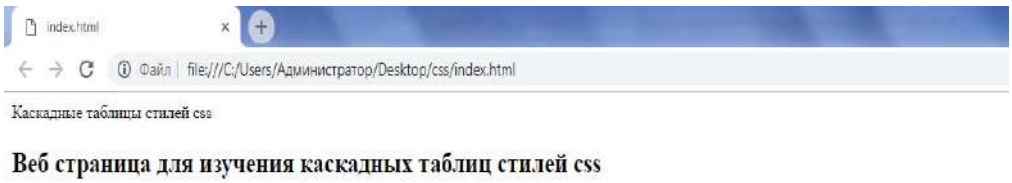


Рисунок 2.9 Результат index.html файла

Если страница, показанная на рисунке 2.8, отобразится, значит таблицы стилей css подключены к файлу index.html.

5. Задать тегу <body> новый стиль, определяющий фоновое изображение HTML-страницы, цвет фона страницы, верхние и нижние отступы, шрифт по умолчанию, его размер и цвет. Тем самым проверить работу таблиц стилей CSS.

```
body {
background-image:url(bg.jpg);
background-color:yellow;
margin-top:0px;
margin-bottom:0px;
font-family:Times New Roman;
font-size:12px;
color:#000066;
}
```

Здесь:

body {...} - задание стилей css для тега <body>.

background-image:url(bg.jpg); - фоновое изображение, где в параметре url(...jpg) указывается полный путь к изображению, которое будет фоновым.

background-color: yellow; - фон тела документа желтый.

margin-top:0px; - расстояние от тела документа до верхней части браузера 0 px.

margin-bottom:0px; - расстояние от тела документа до нижней части браузера 0 пикселей.

font-family: Times New Roman; - установка шрифта для документа.

font-size:12px; - размер шрифта документа 12 пикселей.
color:#000000; - цвет текста темно-синий.

6. Установить фоновое изображение, разместив его в папке с файлами.

Задание для самостоятельной работы: Создать сайт «Визитная карточка», на котором главная страница содержит информацию о личности и меню со следующими составляющими: данные о персоне, хобби, расписание, контакты и др. Навигация по сайту должна осуществляться с помощью гиперссылок.

Контрольные вопросы:

1. Что такое стиль в HTML-документе?
2. Что обозначает термин «каскадные таблицы стилей»?
3. Что называется, селектором?
4. Поясните принцип наследования в CSS.
5. Какие существуют способы создания стилей в HTML-документе?
6. В каком случае удобно использовать каждый из стилей?
7. В чем состоят преимущества использования стилей?

2.3 Технологии серверного программирования.

Языки web-программирования - это специализированные языки, которые предназначены для создания программ с использованием интернет-технологий и обработки текстовых массивов данных. Языки web-программирования делятся на две группы: клиентские и серверные. Код, написанный на клиентском языке, выполняется в браузере, а код, выполняющий на сервере клиентский запрос, пишется на серверном языке. В web-приложениях клиентским языком чаще всего является JavaScript, а серверным - язык программирования PHP. Основу web-приложения на клиентской части составляет язык разметки HTML и каскадные таблицы стилей CSS.

Клиентское программирование - это использование технологий JavaScript и VBScript для динамического изменения внешнего вида web-страницы при ее просмотре и выполнении обработки информации, введенной пользователем в формы.

Серверное программирование – это создание CGI-приложений. CGI (Common Gateway Interface) — технология, позволяющая запускать на web-сервере программы, имеющие возможность получать данные от посетителей сайтов, поддерживаемых этим web-сервером, и, в свою очередь, выдавать им обработанные данные в виде web-страниц или других файлов.

При помощи гипертекстового транспортного протокола HTTP web-браузеры взаимодействуют с web-серверами . Когда запускается поиск, HTTP-запрос отправляется из браузера на целевой сервер. Этот запрос включает: **путь**, определяющий целевой сервер и ресурс (например, файл, определенная точка данных на сервере, запускаемый сервис) и **метод**, который определяет необходимое действие (например, получить файл,

сохранить или обновить некоторые данные). При работе с сервером используются следующие виды методов:

- GET – получение определенного ресурса (например, HTML-файл, содержащий информацию о товаре или список товаров);
- POST – создание нового ресурса (например, добавление статьи на Википедию или нового контакта в базу данных);
- HEAD – получение метаданных об определенном ресурсе без получения содержания, как делает GET;
- PUT – обновление существующего ресурса;
- DELETE – удаление указанного ресурса.

Web-серверы ожидают сообщений с клиентскими запросами, обрабатывают их по прибытии и отвечают web-браузеру при помощи HTTP-сообщения. После возвращения HTML-страница обрабатывается браузером, который далее может исследовать ссылки на другие ресурсы.

Статический сайт (пассивный) - это тот сайт, который возвращается с тем же кодированным содержанием с сервера всякий раз, когда запрашивается конкретный ресурс. Он способен выдавать web-страницы только по запросам пользователей. Например, если есть сайт о книге, эта страница будет возвращена каждому пользователю. Для того чтобы разместить еще одну книгу на сайте, необходимо добавить страницу (например, mybook.html). При внесении каких-либо корректировок нужно менять каждую страницу отдельно, что неэффективно.

Статические сайты удобны тем, что можно отправить один и тот же контент нескольким пользователям с небольшим количеством web-страниц.

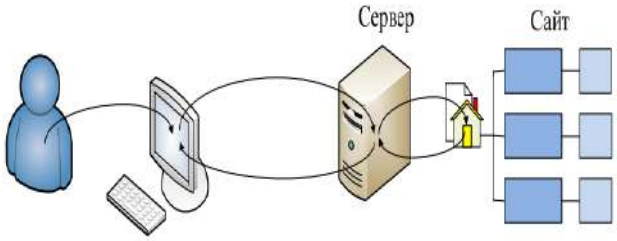


Рисунок 2.10 Статический сайт

Динамический сайт (активный) может генерировать и возвращать содержание сайта на основе конкретного URL-адреса - запроса и данных. Все данные сайта будет хранить сервер. Динамический сайт вступает в диалог с пользователем, запрашивая и принимая от него информацию, при формировании динамических страниц обращается к базам данных сервера, извлекает оттуда информацию и включает ее в формируемую web-страницу.

Рисунок 2.11 Динамический сайт

Использование базы данных позволяет эффективно хранить информацию о данных с помощью легко расширяемого, изменяемого и доступного для поиска способа. Использование HTML-шаблонов позволяет очень легко изменять структуру HTML, потому что это нужно делать только в одном месте, в одном шаблоне, а не через множество статических страниц.

Web-сервер — программа, устанавливаемая на узле сети Интернет. Она выдает посетителям этого узла web-страницы по запросам. Web-сервером часто называется узел, на котором запущена программа, или компьютер, являющийся таким узлом.

Для организации операций приема и сохранения, отправляемых пользователями сообщений используется *серверное программирование*. Чаты, опросы/голосования, счетчики, гостевые книги, форумы, посещений или программные компоненты, взаимодействующие с базами данных на сервере, создаются с помощью серверных скриптов. Серверное программирование позволяет решать такие задачи, как регистрация, авторизация пользователей на сайте и управление аккаунтом в почтовых web-системах и социальных сетях, работа интернет-магазина, поиск информации по базе данных.

Серверные программы позволяют решать ряд важных задач, с которыми невозможно справиться с помощью базовых технологий разработки web-страниц и клиентского программирования. К таким задачам относится, например, сохранение полученной от пользователей информации, взаимодействие с внешней базой данных. Обращение браузера к web-серверу может осуществляться и в рамках одного компьютера. В таком случае установка и функционирование web-сервера может использоваться для отладки серверных программ. Чтобы гарантировать доступ к данным на компьютере для других пользователей, одного web-сервера недостаточно, потребуются также выделенный IP-адрес и запись на сервере. Установка и настройка на компьютере программного обеспечения web-сервера дает возможность разработать и отладить серверную часть web-приложения. Кроме того, необходимо установить транслятор языка серверного программирования, систему управления базой данных (СУБД), с которой будет взаимодействовать web-приложение, и обеспечить взаимодействие всех этих компонентов. Для облегчения этого процесса часто используются

известные пакеты для связки Apache-PHP-MySQL – XAMPP, Denwer, EasyPHP. Альтернативой является разработка с использованием готовых систем управления контентом (CMS), таких как Drupal, Joomla!, WordPressMS, MS WebMatrix, которые не требуют знания языков программирования. Серверные программы классифицируются по следующим видам:

1. Исполняемые программы, работающие через интерфейс CGI (Common Gateway Interface — общий интерфейс обмена), так называемые CGI-nрограммы. Они представляют собой обычные исполняемые файлы, написанные на любом языке программирования и откомпилированные в машинный код процессора. Они не имеют интерфейса пользователя, а взаимодействуют с web-сервером, получают от него входные данные и пересылают обратно результаты работы. Запускаются они самим web-сервером, когда необходимо обработать полученные от пользователя данные, и управляются операционной системой серверного компьютера.

2. Расширения web-сервера (приложения формата ISAPI, NSAPI, модули расширения Apache) - новый способ, позволяющий встраивать серверные программы в сам web-сервер, делая их его составными частями.

3. Активные серверные страницы (ASP, JSP и др.). Фактически это обычные статические web-страницы, сохраненные в файлах. Они включают в себя HTML-код и команды, обрабатываемые либо самим web-сервером, либо его расширением.

4. Серверные сценарии - это написанные на интерпретируемом языке (Perl, Python, VBScript, JavaScript и др.) обычные сценарии, работающие через интерфейс CGI или ISAPI на стороне сервера.

При выборе инструментов разработки web-приложения нужно обращать внимание на технологии, обеспечивающие взаимодействие с базами данных, работу на стороне клиента и сервера, возможности технической поддержки, стоимость программного обеспечения и сопровождения, масштабы web-приложения, вопросы безопасности. Благодаря своей простоте, скорости выполнения, богатой функциональности и открытому исходному коду (Open Source), распространяющемуся под собственной лицензией, одним из лидеров среди языков разработки динамических web-сайтов является PHP.

На сегодняшний день PHP (Hypertext Preprocessor - препроцессор гипертекста) - наиболее распространенный язык web-программирования. Применяя его, можно быстро и легко создавать сайты и порталы различной сложности. С помощью PHP написано большинство сайтов и web-сервисов в Интернете. В их числе facebook.com, vk.com, baidu.com и другие.

PHP был создан в 1994 году датским программистом Расмусом Лердорфом и изначально состоял из набора скриптов на языке Perl. Позже этот набор был переписан в интерпретатор на языке Си. С самого формирования PHP представлял удобный набор инструментов для создания адаптированных web-сайтов и web-приложений. PHP имеет следующие преимущества:

- Возможность создавать web-сайты для всех типов операционных систем (Windows, MacOS, Linux);

- Умение работать с различными web-серверами, такими как: Apache, Nginx, IIS;

- Простота и легкость усвоения;

- Поддержка работы с множеством систем баз данных (MySQL, MSSQL, Oracle, Postgre, MongoDB и другими);

- Доступность хостинговых услуг.

Контрольные вопросы:

1. Дайте определение понятию «Языки web-программирования».
2. Что такое клиентские программы?
3. Серверное программирование – это...?
4. Чем отличается статический сайт от динамического?
5. Объясните назначение web-сервера.

2.4 Технология клиентского программирования JavaScript

JavaScript является объектно-ориентированным языком для создания интерактивных web-страниц и web-приложений. Он обычно используется в клиентской части web-приложений и позволяет управлять браузерами, элементами HTML-документов и стилями каскадных таблиц CSS. Сценарии (скрипты) JavaScript являются основой клиентской части web-приложений. Они загружаются с сервера вместе с web-страницами и выполняются браузером на компьютере пользователя. Сценарии JavaScript обрабатываются с помощью встроенного интерпретатора в браузере.

Технология размещения кода JavaScript на web-странице. Программный код размещается на HTML-странице. Рассмотрим пять способов функционального применения JavaScript:

1. **Размещение кода внутри HTML-документа.** Код JavaScript можно размещать как внутри контейнера <HEAD> ...</HEAD>, так и в теле документа <BODY> ...</BODY>. Код в заголовке документа размещается внутри тегов SCRIPT. Пример вывода окна с надписью: «Изучаем JavaScript»

```
<body>
<script type =″text/javascript″>
Alert (“Изучаем JavaScript”);
<\script>
<\body>
```

2. Размещение во внешнем файле с расширением js. Сценарий программы можно разместить с помощью отдельного текстового файла с расширением js. Подключение внешнего файла скриптов производится с помощью параметра src тега <script>, значением которого является URL или относительный адрес файла со сценарием. Например:

<SCRIPT type=”text/javascript" src="имя_файла">.

3. Гипертекстовая ссылка (схема URL). Схема URL (Uniform Resource Locator). Каждый информационный ресурс в Интернете имеет свой уникальный URL. URL указывают в атрибуте HREF контейнера A, в атрибуте SRC контейнера IMG, в атрибуте ACTION контейнера FORM.

<A HREF="JavaScript:код_программы">...</A>
<FORM ACTION="JavaScript:код_программы" ...> ... </FORM

4.Обработчик события (handler). В JavaScript сценарии выполняются с помощью определенного события. В качестве событий выступают внутренние события браузера, например, загрузка страницы (load) или любые действия пользователя: щелчок мыши (click), наведение указателя на элемент (mouseover), перемещение указателя за пределы элемента (mouseout), перемещение курсора мыши относительно элемента страницы, изменение значения поля формы (change), нажатие пользователем кнопки отправки данных формы (submit).

Пример: при нажатии на кнопку будет выходить окно предупреждения с текстом «Спасибо!»:

<input type = ″button″ value= ″Нажать″ onClick =″alert(‘Спасибо!’)″/>

Типы данных и операторы

Оператор присваивания возвращает результат вычисления. Последовательность операторов присваивания выполняется справа налево:

var hello;
var hello = "Здравствуйте";
hello=" Здравствуйте"

Переменные в JavaScript объявляются с помощью оператора var, и им даются начальные значения.

var к;
var h='Моя первая!';
var к, h='Моя первая!';

Тип переменной определяется по присвоенному ей значению. В JavaScript в разных частях программы можно присваивать одной и той же переменной значения различных типов, и интерпретатор будет менять тип переменной. Узнать тип переменной можно с помощью оператора typeof():

var i=7; alert(typeof(i));
i= new Array(); alert(typeof(i));
i= 2.22; alert(typeof(i));
i= 'Hello!'; alert(typeof(i))

Строковый тип (string). Строка — набор символов, обрамляется **""**.

myString = "Привет";

объединение строк:

var x="При";

y="вет";

s=x+y; *//"Привет "*

Задание №1

С помощью JavaScript метода document.write вывести в окно браузера строку: *Как твои дела?* (с пробелами между словами). Для этого необходимо:

1.Создать 3 переменных с использованием ключевого слова var с именами str1, str2, str3.

2.Переменной str1 присвоить фразу *'Как '*, str2 – *'твои '*, str3 – *'дела? '*- х строк (str1, str2, str3).

В JavaScript реализовано 3 **метода**, которые позволяют выводить пользователю **диалоговые окна**:

- alert,
- confirm,
- prompt.

Alert (строка). Метод alert используется для вывода простейшего диалогового окна, содержащего текст сообщения и единственную кнопку «ОК». Формат вызова данной функции:

alert("Текст сообщения");

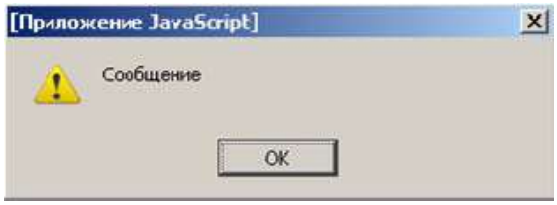


Рисунок 2.12 Использование метода alert
Адаптировано с сайта (http://labs.org.ru/javascript-2/)

Confirm. Функция confirm позволяет вывести диалоговое окно, содержащее текст сообщения и кнопки «ОК» и «Cancel», и применяется в тех случаях, когда пользователь должен сделать выбор. Формат вызова данной функции:

var result=confirm("Текст вопроса");

if(result) {

/* действия */}

Функция confirm возвращает логическое значение в зависимости от нажатой пользователем кнопки: «ОК» соответствует значению true, «Cancel» - значению false. Как правило, результат работы функции присваивают

переменной для дальнейшего анализа, как это показано в примере выше. Confirm("Вы действительно хотите завершить работу?");

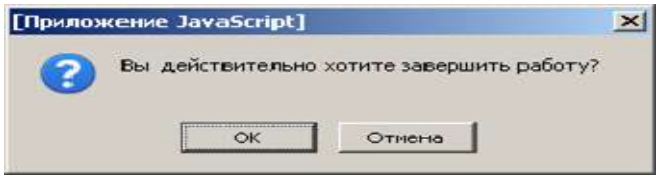


Рисунок 2.13 Использование метода confirm
Адаптировано с сайта (http://labs.org.ru/javascript-2/)

Prompt. Функция prompt позволяет вывести диалоговое окно запроса на ввод данных и применяется в тех случаях, когда пользователь должен ввести строку текста. Формат вызова данной функции:

Объявление переменной

var str=prompt("Запрос на ввод данных", значение_по_умолчанию);

Функция prompt возвращает результат строкового типа. Поэтому, прежде чем его использовать в арифметических выражениях, необходимо выполнить преобразование типов к числовому. Это можно сделать при помощи следующих функций:

- parseInt("строка") - преобразует строку в целое число;
- parseFloat("строка") - преобразует строку в число с плавающей точкой.

Рассмотрим комплексный пример, в котором для реализации взаимодействия с пользователем используются все три метода, описанные выше. Обратите внимание, что для ввода числового значения, метод prompt используется в комбинации с функцией parseInt. Пример:

prompt("Как Вас зовут?", "Иван");

Метод prompt принимает два аргумента: первый выводится в качестве простой строки в модальном окне; второй — это значение по умолчанию в текстовом поле для ввода. Оба аргумента заключаются в кавычки.

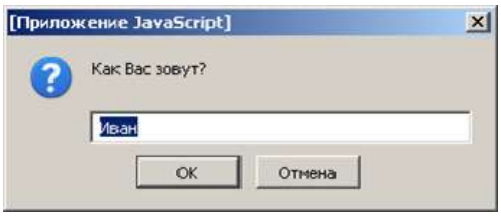


Рисунок 2.14 Результат примера использования метода prompt
Адаптировано с сайта (http://labs.org.ru/javascript-2/)

Задание №2.

Вычислить значение выражения по формуле (все переменные принимают вещественные значения):

$$\frac{7x}{12x^2 + 2x - 5}$$

- Запрос значения x;
- Вычисление выражения;
- Вывод результата при помощи метода alert.

Условные операторы языка JavaScript

BJavaScript условие осуществляет **оператор if**. Рассмотрим синтаксис условного оператора:

```
if (условие)
{ // если истина
  operator1;
  operator2;
}
else
{// если ложь
  operator3;
  operator4;
}
```

Пример: выводить в модальное окно *«a больше 1»*, если переменная, a>1, иначе выводить *«a не больше 1»*

```
var a=1;
if (a>1)
  alert("a больше 1")
else
  alert("a не больше 1");
```

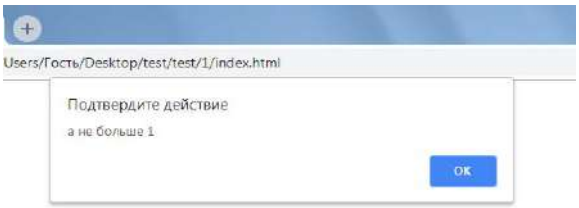


Рисунок 2.15 Результат примера использования оператора if

Оператор переключения в JavaScript – **switch**. Он служит для проверки переменной на множество значений:

```
switch (выражение) {
case вариант1:
  //..блок операторов..
  break
case вариант2:
```

```
  //..блок операторов..
  break
  [default:
  //..блок операторов..]
}
```

Блок, начинающийся со служебного слова **default,** можно опустить. Операторы блока будут выполнены в случае, если ни одно из перечисленных значений не подходит.

Пример: Запрашивать у пользователя ввести цвет. Выводить перевод на английский язык введенного цвета.

```
var color = prompt("Какой цвет?");
switch (color) {
 case "красный" :
 alert("red");
 break;
 case "зеленый":
 alert("green");
 break;
 case "синий":
 alert("blue");
 break;
 default:
 alert("у нас нет документа на таком языке")
}
```

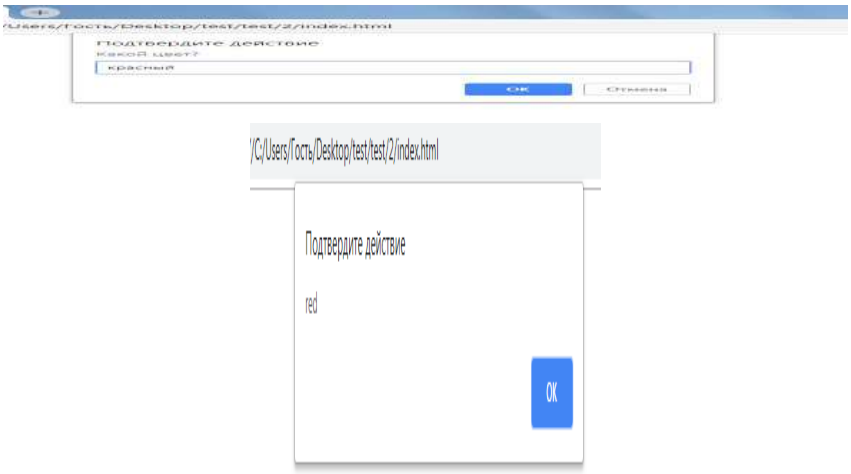


Рисунок 2.16 Результат примера использования оператора switch

Задание №3.

У пользователя запрашивать число – количество яблок на ветке. В зависимости от введенного числа (не более 10) выводить сообщение. Для проверки использовать оператор Switch javascript:

- На ветке 1 яблоко;
- На ветке 5 яблок;
- На ветке 10 яблок.

Циклические операторы языка JavaScript – for. Цикл в Javascript for используется, когда заранее известно, сколько раз должны повториться циклические действия.

for(начальное значение счетчика итераций (повторения); условие; прибавление счетчика)

{ *//..блок операторов..*}

В качестве начального значения счетчика итераций используется выражение присваивания: например, i=0 - счетчик цикла начинается с нуля.

В качестве прибавления счетчика указывается шаг, с которым должен увеличиваться счетчик: например, i++ указывает на то, что каждая итерация цикла будет сопровождаться его увеличением на 1.

Условие цикла - это и есть конечное значение счетчика: например, i<10 - счетчик, достигнув значения 10, останавливает цикл.

Рассмотрим пример использования цикла for в Javascript:

В примере на экран выводятся значения счетчика цикла, так как приращение счетчика i++, соответственно на экране будут появляться *0 1 2 3 ... 9*, причем каждая цифра - с новой строки (тег br).

```
for (var i=0;i<10;i++)
  document.write(i+"<br>")
```

Задание №4.

Вывести сумму всех целых чисел от 1 до 10.

Операторы выхода из цикла break и continue в Javascript. Оператор exit.

Оператор **break** прерывает выполнение всего тела цикла и в итоге осуществляет выход из цикла в JavaScript, в то время как оператор continue прерывает выполнение текущей итерации цикла, продолжая при этом выполнение цикла со следующий итерации.

Рассмотрим работу операторов break и continue на примере:

```
for (var i=0;i<10;i++)
{
  if (i==4) continue;
  document.write(i+"<br>");
  if (i==8) break;
}
```

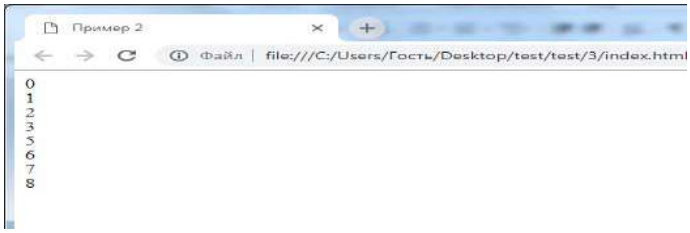


*Рисунок 2.17 Результат примера использования операторов **break** и **continue***

Циклические операторы языка JavaScript - While

Синтаксис оператора while:

```
while (условие)
{
  //..блок операторов..
};
```

Пример: Выводить в диалоговое окно степени двойки до 1000

```
var a = 1;
while (a<1000){
  a*=2;
  alert(a);
}
```

Циклические операторы языка JavaScript - цикл с постусловием do..while

```
do
{
  //..блок операторов..
}
while (условие);
```

В примере использования **цикла do while** необходимо самостоятельно выяснить, что будет выводиться в диалоговое окно:

```
var a = 1;
do
 {
  a*=2;
  if (a==64)
          continue;
  alert(a);
  if (a==256)
          break; }while(a<1000);
```

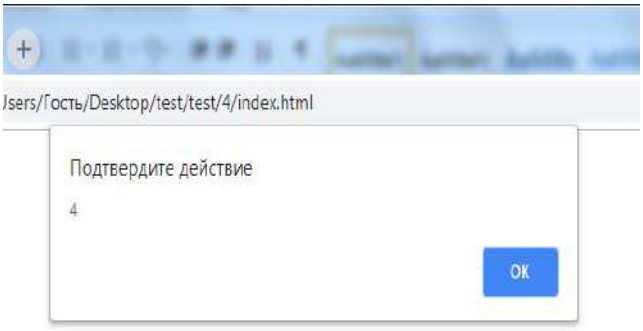


Рисунок 2.18 Результат примера использования цикла с постусловием do..while

Массив — это упорядоченный набор данных. Доступ к элементам массива осуществляется с помощью порядкового номера — индекса. Таким образом, массив — объект, представляющий собой проиндексированный набор элементов.

```
var arr = new Array();
arr[0] = "element1";
arr[1] = "element2";
arr[2] = "element3";
alert(arr[2]);
alert("Число элементов" + arr.length);
```

В Javascript длина массива — свойство length.

Javascript создание массива

Создание элементов массива возможно несколькими способами:

1 способ:

```
var earth = new Array(4); /* массив из 4-х элементов*/
earth[0] = "Планета";
earth[1] = "Земля";
earth[2] = 24 часа;
earth[3] = 365.25;
```

2 способ:

```
var earth = new Array("Планета", "24 часа", 6378, 365.25);
```

3 способ:

```
var earth = new Array(); // пустой массив
earth.xtype = "Планета";
earth.xday = "24 часа";
earth.radius = 6378;
earth.period = 365.25;
```

Обращение или доступ к элементам массива в JavaScript происходит так:

```
var mas=new Array(1,25,'Моя первая ');
```

```
mas[0]='Пока';
mas[1]=35;
```

Вывод массива с использованием обычного цикла for осуществляется в Javascript так:

```
var mas=new Array(1,25,'Моя первая ');
mas[0]='Пока';
mas[1]=35;
function showElement(){
  for(i=0;i<3;i++)
  alert(mas[i]);
}
showElement();
```

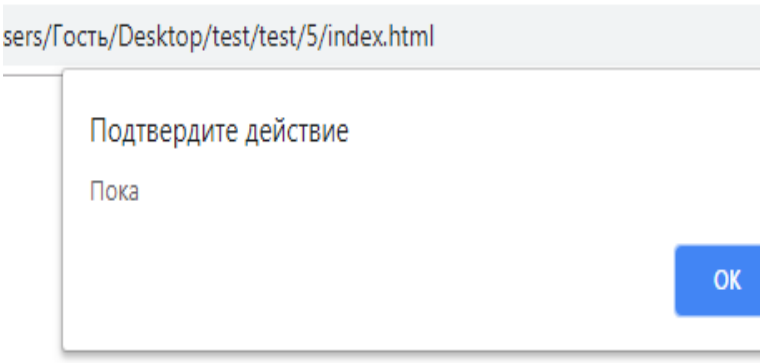


Рисунок 2.19 Результат примера использования массива

Задание №5.

Создать два массива: countries – с названием стран, currency – с национальной валютой этих стран. Вывести название страны и ее денежной валюты (использовать метод document.write).

JavaScript объекты. В JavaScript существует несколько видов объектов:

- встроенные объекты,
- объекты браузера,
- объекты, которые программист создает самостоятельно (пользовательские).

Встроенные объекты — это предопределенные объекты number, string, array.

Пользовательские объекты в JavaScript:

- Object(объекты),
- Number (обработка чисел),
- String (обработка строк),
- Array (массивы),
- Math (математические формулы, функции и константы),

- Date (работа с датами и временем),
- RegExp,
- Global (его свойства Infinity, NaN, undefined),
- Function.

Создание объектов

Существует 2 способа создания объектов:

1. Использование инициализатора объекта (создание объектов-коллекций);
2. Использование конструктора объектов (создание классов-конструкторов).

1 способ: Создание объектов-коллекций

```
var имя_объекта = new Object();
имя_объекта. свойство = значение;// точечная нотация
имя_объекта["свойство"] = значение;// скобочная нотация
```

Пример: Создать объект myBrowser со свойствами name (значение *"Microsoft Internet Explorer"*) и version (значение «*9.0*»)

```
var myBrowser = {name: "Microsoft Internet Explorer", version: "9.0"};
alert(myBrowser.name);
alert(myBrowser.version);
```

2 способ: Создание класса объектов с помощью конструктора (создание классов-конструкторов):

```
function Имя_класса_объектов (св-во1, св-во2){
  this.св-во1 = значение;
  this.св-во2 = значение;
}
```

Пример: Создание конструктора для класса объектов и объекта на основе этого класса. Чтобы создать объект myBrowser со свойствами name (значение *"Microsoft Internet Explorer"*) и version (значение «*9.0*»):

```
function Browser (name, version){
  this.name = name;
  this.version = version;
}
var myBrowser = new Browser("Microsoft Internet Explorer","9.0");
alert(myBrowser.name);
alert(myBrowser.version);
```

Типы событий JavaScript

Событие — это реакция программы на действие пользователя (щелчок мышью по кнопке, уменьшение мышкой окна браузера, ввод текста с клавиатуры и т. д.):

События	**Поддерживающие HTML- элементы и объекты**	**Описание**
onBlur	a, area, button, input, label, select, textarea	Потеря текущим элементом фокуса Возникает при щелчке мышью вне элемента либо нажатии клавиши табуляции
onChange	Input, select, textarea	Изменение значений элементов формы. Возникает после потери элементом фокуса, т.е. после события blur
onClick	Практически все	Одинарный щелчок (нажата и отпущена кнопка мыши)
onFocus	a, area, button, input, label, select, textarea	Получение элементом фокуса
onLoad	body, frameset	Закончена загрузка документа
onMouseDown	Практически все	Нажата кнопка мыши в пределах текущего элемента
onMouseOut	Практически все	Курсор мыши выведен за пределы текущего элемента
onMouseOver	Практически все	Курсор мыши наведен на текущий элемент
onMouseUp	Практически все	Отпущена кнопка мыши в пределах текущего элемента
onMove	window	Перемещение окна
onResize	window	Изменение размеров окна
onSelect	textarea, input	Выделение текста в текущем элементе
onSubmit	form	Отправка данных формы
onUnload	body, frameset	Попытка закрытия окна браузера и выгрузки документа

Таблица 2.5 Стандартные события JavaScript

Событие **onClick** происходит **во время одинарного щелчка кнопкой мыши**.

```
<body>
<form>
<input type="button" name= "myButton" onClick="message()" value="Щелкни!">
</form>
```

В данном примере в HTML-коде можно увидеть кнопку. У нее присутствует атрибут onClick (*«по щелчку»*), в значении которого стоит вызов функции с названием message (). Это пользовательская функция, описанная выше в скрипте. В самой функции выводится диалоговое окно, что и требуется в задании.

События **onMouseOver и onMouseOut** наступают при наведении (**onMouseOver**) на объект и при выведении (**onMouseOut**) за пределы объекта курсора мыши.

Назначение обработчика событий осуществляется точно таким же образом, как и в случае с событием onClick.

Задание №6.

По наведению курсора мыши на гиперссылки закрашивать задний фон страницы в разные цвета. Дополните код:

 Зеленый

 … seagreen

 … magenta

 … purple

 … navy

 … royalblue

Контрольные вопросы:

1. В каком месте HTML-документа находится стандартное расположение скрипта

JavaScript?

2. С помощью каких методов осуществляется вывод диалоговых окон?
3. Что такое типы данных? Какие типы поддерживает JavaScript?
4. При помощи каких операторов создаются комментарии?
5. Перечислить функции преобразования данных.
6. Какие методы для вывода модальных окон в Javascript известны?
7. Какой метод позволяет вывести модальное окно для ввода данных?

2.5 Общее понятие Bootstrap (Бутстрап) верстки

Bootstrap – это разработанная компанией «Twitter» система управления сайтом (фреймворк), которая содержит готовые CSS, HTML и JavaScript компоненты. Фреймворк - программный структурированный каркас, включающий программное обеспечение для создания web-сайтов и облегчающий их разработку.

Bootstrap используется для самостоятельного конструирования, верстки сайтов любой сложности. Он содержит стили для основных элементов, которые применяются в верстке. Использование такого фреймворка значительно ускоряет процесс создания web-страниц. Стандартные стили легко менять, что обеспечивает гибкий и простой процесс создания макетов

сайтов. С помощью Bootstrap в кратчайшие сроки можно создать web-сайт на высоком уровне.

Фреймворк Bootstrap состоит из следующих папок: css, js и fonts. В папке css хранятся готовые CSS-стили; в js - файл с набором готовых js-сценариев; в fonts - шрифты.

Компоненты и инструменты Bootstrap:

1. Сетка. При создании сайта в Bootstrap каркас web-страницы строится с помощью сетки. Сеткой называют адаптивную 12-колоночную сетку с фиксированными размерами колонок.

Прежде чем создать колонки, нужно прописать строку. Для этого достаточно тега «row».

```
<div class="row">
... здесь будут колонки...
</div>
```

Вначале пишется строка, а в ней строятся колонки.

1. **Создание строки:**

```
<div class="row">
... пишутся колонки...
</div>
```

2. Создание колонки (сетки):

Сетка Bootstrap состоит из 12 одинаковых по ширине колонок. При необходимости колонки можно объединить и разместить в нужной позиции.

Чтобы создать колонки, нужно прописать внутри «row» класс «col-(*1)-(*2)».

Там, где (*1), указываем, для каких групп устройств нужно выполнить.

Там, где (*2), указываем число (количество колонок).

	Очень маленькие устройства Телефоны (<768px)	Малые устройства Планшеты (≥768px)	Средние устройства Настольные (≥992px)	Большие устройства Настольные (≥1200px)
Ширина контейнера	Нет (автом.)	750px	970px	1170px
Класс префикса	.col-xs-	.col-sm-	.col-md-	.col-lg-
Количество колонок	12			
Ширина колонки	Авто	60px	78px	95px

Таблица 2.6 Разметки Bootstrap для различных устройств

С помощью таблицы можно построить сетку для различных групп устройств (мобильных телефонов, планшетов, ПК и т.д.).

Пример:

1.
2. <div class="col-lg-12 col-md-8 col-sm-6 col-xs-12»> блог StepkinBLOG.RU
3. <div class="col-lg-12 col-md-4 col-sm-6 col-xs-12»> Таблица разметки Bootstrap
4.

В строках №2 и №3 col-md-8 и col-md-4 (размер экрана ≥992px). На экране будет две колонки, так как 8 + 4 =12 колонок в сетке. Левая часть объединит в себе 8 колонок, а правая - 4.

Задание№1.

Сделать адаптивную верстку на формирование ячеек.

<div class=″row myrow″>
<\div>
<\div>
<\div>
<\div>
<\div>
<\div>
<\div>
<\div>
<\div>

Результат:

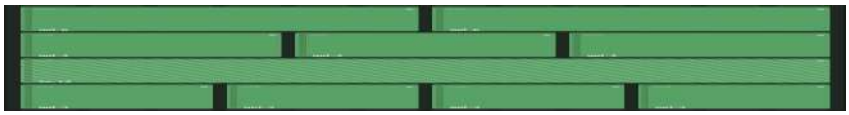


Рисунок 2.20 Верстка на формирование ячеек

2. Шаблон. Для Bootstrap можно скачать готовые адаптированные шаблоны. Они бывают резиновыми или фиксированными. В случае если общему контейнеру задать класс container, его максимальная ширина будет ограничена 1170 пикселями. Если указать container-fluid, ширину сайта ничто не будет ограничивать. Например, на мониторах шириной 1920 пикселей его ширина будет такой же – на все 100% окна. Перечень бесплатных HTML-шаблонов на Bootstrap:

Codester - Супер-шаблон на Bootstrap с качественным HTML.

Stylish Portfolio - адаптивный шаблон для портфолио.

– Modern Business

Weddo- Адаптивный HTML-шаблон на Bootstrap свадебной тематики.

– Brushed

Colorsy-Адаптивный Bootstrap-шаблон галереи изображений с боковой колонкой навигации.

Business Casual

– SofiaPenny- Адаптивная тема в голубых тонах на фреймворке Bootstrap.

Joey- Простой адаптивный бизнес-шаблон в голубых тонах.

ResponsiveWebInc- для сайта-визитки.

Resume- Шаблон на Bootstrap для создания личной странички.

3. **Типографика** дает возможность оформить код, цитаты, абзацы, заголовки, заголовки со вторичным текстом, подзаголовки, выравнивание текста, аббревиатуры.

Заголовки в Bootstrap обозначаются тегами <h1> - <h6>.

 Полужирный текст

  Жирное начертание

 Курсивное начертание текста

 Нижний индекс

 Верхний индекс

<ins></ins> Подчеркнутый тест

<big></big> Текст с увеличенным размером шрифта

<small> </small> Текст с уменьшенным размером шрифта

Выравнивание текста и заголовка:

"text-justify" - Выравнивание текста по ширине,

"text-left" - Выравнивание по левому краю,

"text-right" - Выравнивание по правому краю,

"text-center" - Выравнивание по центру.

Для параграфа

 Выравнивание по левому краю.

 Выравнивание по центру.

 Выравнивание по правому краю.

 Выравнивание текста по ширине

Для заголовка

<h1 class="text-left"> Выравнивание по левому краю. </h1>

<h2 class="text-center"> Выравнивание по центру. </h2>

<h3 class="text-right"> Выравнивание по правому краю. </h3>

<h4 class="text-justify"> Выравнивание текста по ширине</h4>

4**. Медиа** — позволяет художественно оформлять картинки и видео.

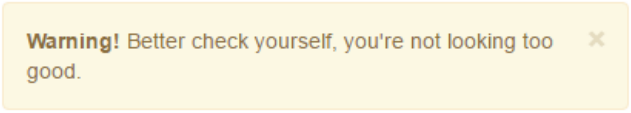
5. **Таблицы** — можно добавить таблицу.

6. **Навигация** — используется для создания навигационного меню на сайте.

7. **Формы** — позволяет создавать различные формы: в одну или несколько строк, с подсказками и выпадающими кнопками.

Рисунок 2.21 Создание формы
Адаптировано с сайта (https://blogwork.ru/chto-takoe-bootstrap/)

8. **Алерты** применяются для оформления диалоговых окон. В основном, это всплывающие окна или подсказки: ошибка (danger), подсказка (info), предупреждение (warning). При использовании достаточно дописать необходимый класс.



9. ***Кнопки*** — можно добавлять кнопки и выпадающие кнопки.

Рисунок 2.22 – Выпадающее меню
Адаптировано с сайта (https://blogwork.ru/chto-takoe-bootstrap/)

10. ***Прогресс-бары***

Рисунок 2.23 Прогресс бары
Адаптировано с сайта (https://blogwork.ru/chto-takoe-bootstrap/)

11. **Шрифты из иконок** — это огромная возможность добавить художественные элементы в оформление сайта. Можно даже отказаться от формирования и использования иконочных спрайтов.

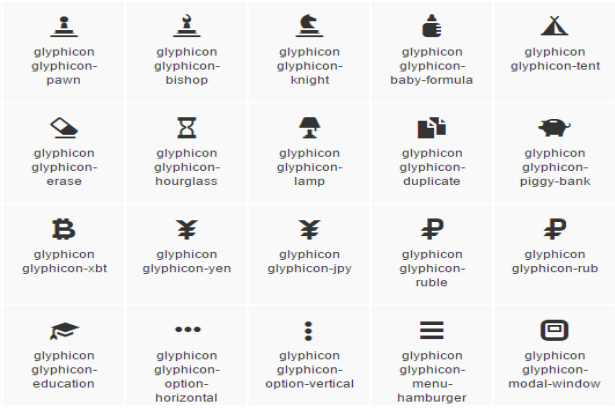


Рисунок 2.24 Шрифты из иконок
Адаптировано с сайта (https://blogwork.ru/chto-takoe-bootstrap/)

Фреймворк Bootstrap3 содержит 200 качественных иконок из набора Glyphicons Halflings. Чтобы использовать иконки Glyphicons на сайте, нужно добавить тег или <i>, прописать базовый класс значков «glyphicon» и через пробел добавить класс нужной иконки «glyphicon-название_иконки»:

<span class="glyphicon glyphicon-название_иконки»>

Можно указать размер иконкам:

btn-lg - большая,

btn-sm - маленькая,

btn-xs - очень маленькая.

Для определения красного цвета к иконке вставляем код «style="color: red;"».

В Bootstrap, можно задавать различные состояния для кнопок. Эти стили можно применять к таким элементам, как <a>, <input>, <button>.

Вид кнопки	Класс
Default	btn btn-default
Primary	btn btn-primary
Info	btn btn-info
Success	btn btn-success
Warning	btn btn-warning
Danger	btn btn-danger
Link	btn btn-link

Таблица 2.7 Стили доступные в Bootstrap 3

Пример для кнопки button:

```
<button type="button" class="btn btn-default"> Default</button>
<button type="button" class="btn btn-primary"> Primary</button>
```

Для ссылки

```
<a href="#" class="btn btn-default"> Default</a>
<a href="#" class="btn btn-primary">Primary</a>
<a href="#" class="btn btn-info">Info</a>
```

Создание стандартного маркированного списка

Маркированные списки создаются стандартным путем с использованием HTML-тегов …… :

```
<ul>
<li>Элемент маркированного списка</li>
<li>Элемент маркированного списка</li>
</ul>
```

Результат:

✓ Элемент маркированного списка

✓ Элемент маркированного списка

Нумерованные списки создаются стандартным путем с использованием HTML-тегов <ol> …… :

```
<ol>
<li>Элемент нумерованного списка</li>
<li>Элемент нумерованного списка</li>
</ol>
```

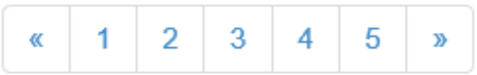
Результат:

1. Элемент нумерованного списка
2. Элемент нумерованного списка

Пагинация — это порядковая нумерация web-страниц. Для создания пагинации достаточно прописать класс «pagination» для тега <ul>:

```
<ul class="pagination">
<li> <a href="#"> «</a> </li>
<li> <a href="#">1</a> </li>
<li> <a href="#">2</a> </li>
<li> <a href="#">3</a> </li>
<li> <a href="#">4</a> </li>
<li> <a href="#">5</a> </li>
<li> <a href="#">» </a> </li>
</ul>
```

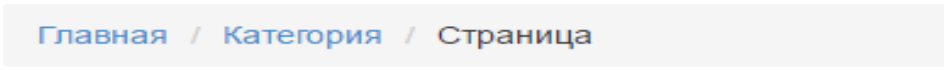
Выглядит она так:



«Хлебные крошки» — это навигационная цепочка, с помощью которой пользователю легко ориентироваться, в какой категории он находится. Чтобы в Bootstrap3 создать «хлебные крохи», достаточно прописать класс «breadcrumb» для тега

```
<ul class="breadcrumb">
<li> <a href="#"> Главная</a> </li>
<li> <a href="#"> Категория</a> </li>
<li>Страница</li>
</ul>
```

Результат:



Метки — это дополнительный способ навигации по сайту. Выглядят метки, например, так:

Рисунок 2.25 Метки
Адаптировано с сайта (https://blogwork.ru/chto-takoe-bootstrap/)

Чтобы в Bootstrap3 создать метки, достаточно прописать класс «label» для тега :

```
<span class="label label-default"> Default</span>
<span class="label label-primary"> Primary</span>
<span class="label label-success"> Success</span>
<span class="label label-info"> Info</span>
<span class="label label-warning"> Warning</span>
<span class="label label-danger"> Danger</span>
```

Результат:

Практическая работа №1

Тема: подключение Bootstrap.

Цель: закрепить полученные знания по изучению фреймворка Bootstrap.

Ход работы:

Предварительно установить текстовый редактор Sublime Text.

1. Создать на рабочем столе папку bootstrap.
2. Запустить текстовый редактор Sublime Text. Открыть ранее созданную папку. Выбрать команду New folder и создать новую папку assets. В этой папке будут храниться CSS, Java Script, рисунки.
3. В корневой папке bootstrap создать главный файл сайта index.html с помощью команды File -New File (Ctrl+S).
4. Скачать Bootstrap с официального сайта https://getbootstrap.com/docs/3.3/getting-started/. Для этого нужно зайти на сайт и нажать Download Bootstrap. После загрузки папки открыть ее, выделить в ней 3 других папки - css, fonts, js — и скопировать (Ctrl+C) их. Открыть папку vizitkabootstrap, найти в ней папку assets, в нее вставить (Ctrl+V) скопированные css, fonts, js.
5. Зайти в папку css, оставить в ней только файлы bootstrap.min и bootstrap-theme. min., а остальные - удалить.
6. Вернуться в папку assets. Открыть в ней js. Здесь оставить только файл bootstrap.min, а остальные удалить. Закрыть все папки.
7. Скачать текст базового шаблона (код указан ниже) для Bootstrap и вставить его в файл index.html. Базовый шаблон (Basic template) можно скопировать с сайта, с которого был загружен Bootstrap.

Базовый шаблон:

```
<!DOCTYPE html>
<html lang="ru">
<head>
<meta charset="utf-8">
<meta http-equiv="X-UA-Compatible" content="IE=edge">
<meta name="viewport" content="width=device-width, initial-scale=1"> <!- - The above 3 meta tags *must* come first in the head; any other head content must come *after* these tags -->
<title> Первая программа</title> <!-- Bootstrap -->
<linк href=" assets/css/bootstrap.min.css" rel="stylesheet"> <!-- HTML5 shim and Respond.js for IE8 support of HTML5 elements and media queries --> <!-- WARNING: Respond.js doesn't worк if you view the page via file:// --> <!--[if lt IE 9]> <script src="https://oss.maxcdn.com/html5shiv/3.7.3/html5shiv.min.js"></script> <script src="https://oss.maxcdn.com/respond/1.4.2/respond.min.js"></script> <![endif]--> </head>
<body>
<h1>Hello, world! </h1> <!-- jQuery (necessary for Bootstrap's JavaScript plugins) --> <script src="https://ajax.googleapis.com/ajax/libs/jquery/1.12.4/jquery.min.js"></script> <!-- Include all compiled plugins (below), or include individual files as needed --> <script src=" assets/js/bootstrap.min.js"></script>
</body>
</html>
```

Тег <html lang="ru"> означает, что контент сайта будет на русском языке.

Строка <meta name="viewport" content="width=device-width, initial-scale=1"> дает возможность сайту отображаться в мобильных устройствах.

8. Откорректировать данный шаблон: изменить название страницы Title на «Первая программа». Далее идет подключение css: добавить библиотеку в строке <assets/link href="css/bootstrap.min.css" rel="stylesheet"> название папки assets, где находится css.

9. Нужно изменить путь Java Script файла для работы с фреймворком. Здесь так же добавить название папки <script src=" assets/js/bootstrap.min.js"> </script>.

10. После редактирования кода программы нажать «Сохранить файл» (Ctrl+S). Для проверки открыть данный файл через браузер Google Chrome.

Практическая работа № 2

Тема: создание сайта визитки в Bootstrap 3.

В качестве примера рассмотрим процесс создания сайта фирмы Business, которая будет содержать информацию о компании, услугах, портфолио, а также контактную информацию.

Ход работы:

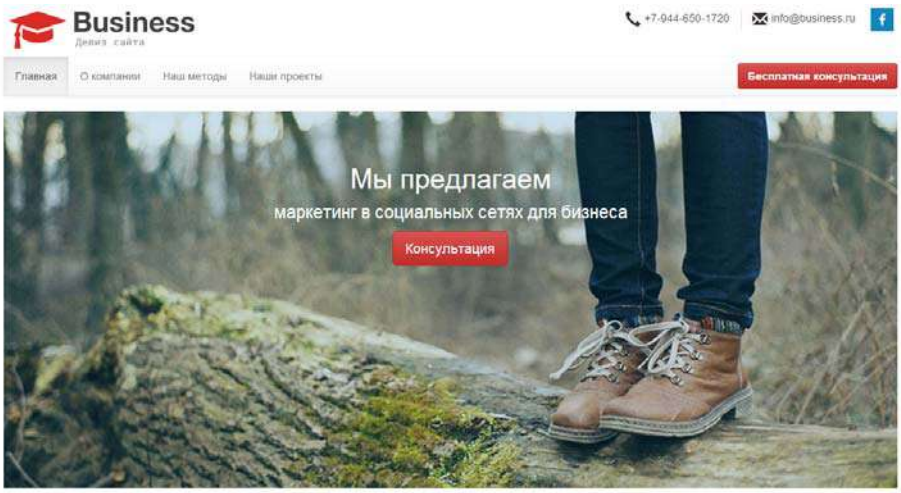


Рисунок 2.26 Главная страница сайта
Адаптировано с сайта (https://itchief.ru/lessons/bootstrap-3/website-creation-business-cards-(part-1))

1. Создание сайта нужно начать со скачивания Bootstrap с официального сайта https://getbootstrap.com/docs/3.3/getting-started/ или архива с сайта https://itchief.ru/lessons/bootstrap-3/website-creation-business-cards-(part-1).

2. После скачивания и распаковки архива, создать HTML-файл с именем index.html, к которому подключатся стили и скрипты платформы Bootstrap 3 и стили для использования иконок Font Awesome.

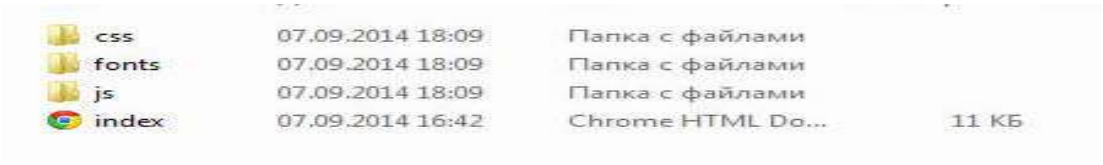


Рисунок 2.27 Структура папки сайта

3. Прописать код в файле index.html:

```
<!DOCTYPE html>
<html lang="ru">
<head>
<!-- Кодировка документа -->
<meta charset="utf-8">
<!-- Заголовок страницы -->
<title> Маркетинг в социальных сетях для бизнеса | Bussines.ru</title>
<!-- Подключения таблицы стилей Bootstrap 3 -->
<link rel="stylesheet" type="text/css" href="css/bootstrap.min.css">
<!-- Подключение темы со стилями Bootstrap 3 -->
<link rel="stylesheet" type="text/css" href="css/bootstrap-theme.min.css">
<!-- Подключение таблицы стилей иконок Awesome -->
<link rel="stylesheet" type="text/css" href="css/font-awesome.min.css">
<!-- Подключение таблицы стилей, в которой будем прописывать свои стили -->
<link rel="stylesheet" type="text/css" href="css/style.css">
<!-- Подключение библиотеки jQuery для работы скриптов Bootstrap 3 -->
<script src="js/jquery-1.11.1.min.js"></script>
<!-- Подключение скриптов Bootstrap 3 -->
<script src="js/bootstrap.min.js"></script>
</head><body>
<!-- Основное содержимое страницы -->
</body> </html>
```

4. Создание макета. На изображении видно, что web-страница состоит из 2 главных частей: основного контейнера (container) и подвала (footer). Основной контейнер (container) выровнен в горизонтальном направлении по центру и содержит следующие части: шапку страницы (header); горизонтальное навигационное меню (nav); нижнюю часть шапки (header-bottom); блок об услугах компании (main), состоящий из 3 колонок; блок об основных методах (method), применяемых в компании; блок, содержащий портфолио (work) компании.

Рисунок 2.28 Макет сайта Адаптировано с сайта (https://itchief.ru/lessons/bootstrap-3/website-creation-business-cards-(part-1))

```
<body>
 <!-- Основной контейнер -->
 <div class="container">
 <!-- "Шапка" сайта -->
 <header></header>
 <!-- Навигационное меню -->
 <nav></nav>
 <!-- Нижняя часть "шапки" сайта -->
 <div id="header-bottom"></div>
 <!-- Блок об услугах компании -->
 <div id="main">
 <!-- Ряд, состоящий из 3 блоков -->
 <div class="row">
 <!-- 1 блок, состоящий из 4 колонок Bootstrap -->
 <div class="col-md-4"></div>
 <!-- 2 блок, состоящий из 4 колонок Bootstrap -->
 <div class="col-md-4"></div>
 <!-- 3 блок, состоящий из 4 колонок Bootstrap -->
 <div class="col-md-4"></div>
 </div>
 </div>
 <!-- Блок, содержащий информацию об основных методах компании --
>
 <div id="method">
 <div class="row"></div>
 </div>
```

```
<!-- Блок, содержащий портфолио компании -->
<div id="workк">
<div class="row"></div>
</div>
</div>
<!-- Подвал сайта -->
<footer></footer>
</body>
```

Задание для самостоятельной работы:

Новый блок. Добавить еще один блок синего цвета. Расположить его на всю ширину экрана после блока «Последние работы». Внутри блока разместить еще два других так, чтобы в каждом были заголовок и текст.

2.6 Основы MySQL

MySQL - это популярная система управления базами данных с открытым исходным кодом, обычно используемая в web-приложениях благодаря своей быстроте, гибкости и надежности. MySQL использует SQL (Structured Query Language) для доступа и обработки данных, содержащихся в базах данных. Каждый, кто поставил перед собой цель освоить web-технологии, должен знать SQL — язык структурированных запросов, применяемый для создания и управления данными в реляционных базах данных. Если говорить о современных web-приложениях, то практически все из них взаимодействуют с СУБД — системой управления базой данных.

MySQL является самым популярным СУБД. Поэтому разработчикам web-приложений необходимо освоить СУБД MySQL и язык SQL.

MySQL – компактный многопоточный сервер баз данных, который характеризуется большой скоростью, устойчивостью и легкостью в использовании. MySQL является идеальным решением для малых и средних приложений. Исходники сервера компилируются на множестве платформ. Наиболее полно возможности сервера проявляются на Unix-серверах, где есть поддержка многопоточности, что дает значительный прирост производительности.

Возможности MySQL.

MySQL поддерживает язык запросов SQL в стандарте ANSI 92 и имеет множество расширений к этому стандарту, которых нет ни в одной другой СУБД.

Краткий перечень возможностей MySQL:

1. Поддерживается неограниченное количество пользователей, одновременно работающих с базой данных.
2. Количество строк в таблицах может достигать 50 млн.
3. Быстрое выполнение команд. Возможно, MySQL самый быстрый сервер из существующих.
4. Простая и эффективная система безопасности.

MySQL, действительно, очень быстрый сервер, но для достижения этого разработчикам пришлось пожертвовать некоторыми требованиями к реляционным СУБД.

В MySQL отсутствуют:

1. Поддержка вложенных запросов типа SELECT * FROM table1 WHERE id IN (SELECT id FROM table2).
2. Не реализована поддержка транзакций. Взамен предлагается использовать LOCK/UNLOCK TABLE.
3. Нет поддержки триггеров и хранимых процедур.

По словам создателей, именно эти пункты дали возможность достичь высокого быстродействия. Их реализация существенно снижает скорость сервера. Эти возможности не являются критичными при создании web-приложений, что в сочетании с высоким быстродействием и малой ценой позволило серверу приобрести большую популярность.

Установка MySQL в Windows:

1. Загрузить пакет установщика.
2. Распаковать его в любом месте.
3. Запустить файл setup.exe.

Установщик setup.exe по умолчанию установит все под C: \ mysql.

Протестировать сервер, запустив его из командной строки в первый раз. Перейти в расположение сервера mysqld, который находится на диске C: \ mysql \ bin, и ввести - mysqld.exe ---concole

Шаги после установки

MySQL поставляется с пустым паролем для пользователя root MySQL. Как только установлена база данных и клиент, необходимо установить пароль root так, как указано в следующем блоке кода:

[root@host]#mysqladmin -u root password "new password"

Чтобы установить соединение с сервером MySQL, необходимо использовать следующую команду:

[root@host]#mysqladmin -u root-p
Enter password:****

Запуск и выключение сервера MySQL

Проверить, работает ли сервер MySQL, можно следующей командой:

- ps-ef| grep mysqld

Если MySql запущен, то процесс mysqld будет указан в результате. Если сервер не запущен, можно запустить его командой:

root@host#cd/usr/bin
/safe_mysqld &

Чтобы закрыть уже запущенный сервер MySQL, нужна команда:

root@host#cd/usr/bin
/ mysqladmin -u root-p shutdown
Enter password:****

Настройка учетной записи пользователя MySQL

Для добавления нового пользователя в MySQL нужно внести новую запись в пользовательскую таблицу в базе данных mysql.

Пример. Добавление нового пользователя - гостя с привилегиями SELECT, INSERT и UPDATE с паролем guest123:

```
root@host# mysql -u root -p
Enter password:*******
mysql> use mysql;
Database changed

mysql> INSERT INTO user
 (host, user, password,
 select_priv, insert_priv, update_priv)
 VALUES ('localhost', 'guest',
 PASSWORD('guest123'), 'Y', 'Y', 'Y');
Query OK, 1 row affected (0.20 sec)

mysql> FLUSH PRIVILEGES;
Query OK, 1 row affected (0.01 sec)

mysql> SELECT host, user, password FROM user WHERE user = 'guest';
+-----------+---------+------------------+
| host | user | password |
+-----------+---------+------------------+
| localhost | guest | 6f8c114b58f2ce9e |
+-----------+---------+------------------+
1 row in set (0.00 sec)
```

Заявление FLUSH PRIVILEGES говорит серверу перезагрузить таблицы предоставления. Если его не использовать, то будет невозможным подключение к MySQL с применением новой учетной записи пользователя, по крайней мере, до перезагрузки сервера.

Создание базы данных с помощью mysqladmin

Для создания или удаления базы данных MySQL понадобятся специальные привилегии. Если есть доступ к пользователю root, можно создать любую базу данных с помощью MySQL mysqladmin.

Пример:

Создание базы данных под названием WORK -

```
[root@host]# mysqladmin -u root -p create WORK
Enter password:******
```

Описание СУБД MySQL

Клиентская часть СУБД MySQL названа **mysql**. Она обеспечивает интерфейс командной строки с СУБД MySQL и возможность неинтерактивной пакетной обработки.

Опции, поддерживаемые программой mysql, представлены в таблице. Можно использовать «короткий» одиночный символ или более подробную версию.

Опция	Описание
-\?, --help	Справка.
-d, --debug=[options]	Вывести в протокол отладочную информацию. В общем виде *'d:t:o,filename'*.
-d, --debug-info	Вывести отладочную информацию при выходе из программы.
-e, --exec	Выполнить команду и выйти, неявная форма опции *—batch.*
-f, --force	Продолжить, даже если сталкиваемся с SQL ошибкой.
-h, --hostname=[hostname]	Задает имя сервера, с которым нужно соединиться.
-P, --port=[port]	Порт для соединения с сервером MySQL.
-p, --password=[password]	Пароль пользователя для соединения с сервером MySQL. Важно, что не должно быть пробела между -p и паролем.
-q, --quick	Быстрый (небуферизованный вывод), может замедлить сервер, если вывод приостановлен.
-s, --silent	Работать молча (подавить вывод).
-u, --user=[user]	Имя пользователя для соединения с сервером MySQL. Необязательно, если имя пользователя такое же, как *логин*. По умолчанию *логин* используется в качестве имени пользователя, что облегчает настройку.
-v, --verbose	Подробный вывод. -v опция может быть удвоена или утроена для более подробного вывода.
-w, --wait	Если подключение не удалось, то подождать и повторить попытку.
-B, --batch	Выполнить в пакетном режиме. Никаких запросов и никаких ошибок в STDOUT. Устанавливается автоматически при чтении из записи в канал (пайп). Результаты будут выведены в формате с разделением табуляцией. Одна строка результата соответствует одной строке вывода.

-I, --help	Справка, эквивалент -\?
-V, --version	Вывести информацию о версии пакета.

Таблица 2.8 Опции mysql

(Адаптировано из методических указаний к выполнению практического задания №9 для студентов специальности 071900 «Информационные системы и технологии»/ сост. Г.К. Конопелько, Д.Г. Конопелько – Хабаровск: Изд-во Хабар гос. техн. унт-та)

Особенности СУБД MySQL

1. Ядро MySQL само удаляет пробелы, находящиеся в конце строки.

2. При сравнениях и сортировке регистр символов в колонках CHAR и VARCHAR не учитывается, если только не задать атрибут BINARY, например:

CREATE TABLE foo (A VARCHAR(10)BINARY);

3. В версиях MySQL после 3.23 можно осуществлять сравнение строк без учета регистра с помощью модификатора BINARY:

*SELECT * FROM table WHERE BINARY column = "A"*

4. Национальные символы обрабатываются в сравнениях соответственно системе кодировки, указанной во время компиляции, по умолчанию ISO-8859-1. Системы кодировки, не соответствующие ISO, в частности UTF-16, не поддерживаются.

5. Конкатенация строк производится с помощью функции SQL CONCAT(s1, s2, ...).

Контрольные вопросы:

1. Что такое MySQL? Возможности MySQL.
2. Как создать базу данных в MySQL?
3. Опишите команду создания пароля пользователя для соединения с сервером.

2.7 Работа с Php Myadmin

База данных – это совокупность структурированных данных. SQL – является языком запросов базы данных. MySQL - это система управления базами данных. При работе в MySQL нужно создать пользователя и указать для него логин, пароль. Затем базе данных нужно дать соответствующие права пользования.

PHPMyAdmin — web-приложение для управления базами данных СУБД MySQL на сервере, написанное на языке web-программирования PHP. При работе с базой данных в PHPMyAdmin для передачи на сервер команд используется браузер. В качестве языка работы с БД используется язык SQL.

PHP позволяет использовать различные системы управления базами данных, но наиболее популярной на сегодняшний день в связке с PHP является MySQL. Он представляет бесплатное программное обеспечение, позволяющее взаимодействовать с базами данных с помощью команд языка

SQL. Среда PHPMyAdmin является одной из самых популярных оболочек для работы с MySQL. Он позволяет:

- Создавать базу данных;
- Создавать в базе данных таблицы и добавлять, удалять, редактировать данные в них;
- Осуществлять поиск данных;
- Устанавливать привилегии на базу данных, таблицу;
- Делать копию и восстанавливать базу данных.

Практическая работа № 1

Установка phpMyAdmin на компьютере с помощью существующего сервера Apache

1. Для установки PhpMyAdmin необходимо настроить Apache, PHP и MySQL на компьютере. PhpMyAdmin можно скачать со страницы https://www.phpmyadmin.net/
2. После щелчка по «Download» (Скачать) запустится процесс скачивания архива (ZIP-файла).
3. Щелкнуть «Close» (Закрыть), когда появится запрос.
4. Распаковать загруженный архив. Скопировать папку «phpMyAdmin».
5. Открыть папку «htdocs». Она находится в папке «Apache», которая расположена на диске «C:». В этой же папке есть текстовый файл «index.php».
6. Вставить скопированную папку «phpMyAdmin» в папку «htdocs».
7. Переименовать скопированную папку на phpmyadmin.
8. Открыть папку «PHP». Эта папка находится на диске «C:». Найти файл «php.ini-production» и переименовать на php.ini
9. Открыть файл «php.ini» в Блокноте. Найти строку с текстом «extension=php_mbstring.dll» и удалить точку с запятой.
10. Найти строку с текстом «extension=php_mysqli.dll» и удалить точку с запятой. Так происходит настройка сервера phpMyAdmin.
11. Сохранить изменения и закрыть Блокнот.
12. Запустить сервер Apache. Открыть командную строку от имени администратора: щелкнуть правой кнопкой мыши по «Пуск».
13.  , выбрать «Командная строка (Администратор)» и нажать «Да», когда появится запрос.
14. Ввести cd /Apache24/bin и нажать «Enter» (заменить «Apache24» на имя папки с Apache);
15. Ввести httpd -к restart и нажать «Enter».
16. Протестировать работу phpMyAdmin: открыть web-браузер, в адресной строке ввести http://localhost и нажать «Enter». Должна открыться страница авторизации phpMyAdmin.

Практическая работа № 2
Создание базы данных MySQL в phpMyAdmin

Чтобы приступить к созданию базы данных, нужно перейти в меню «Базы данных». В поле ввести название базы и нажать «Создать».

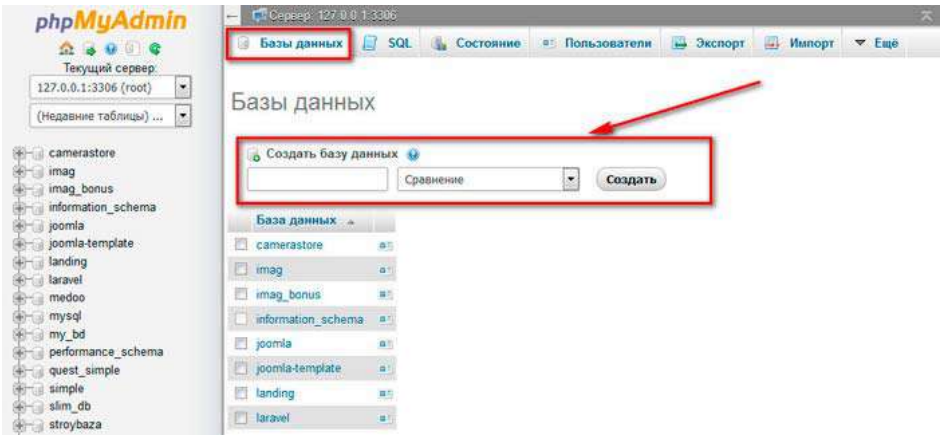


Рисунок 2.29 Создание базы данных в phpMyAdmin

Удаление базы данных. Для того чтобы удалить базу данных, следует в окне баз данных выбрать необходимую базу и нажать «Удалить».

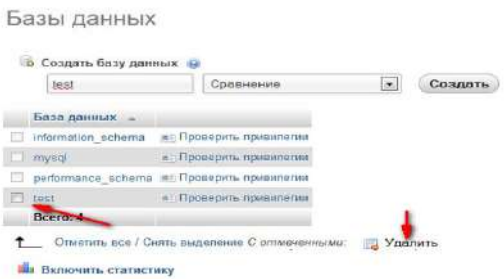


Рисунок 2.30 Удаление базы данных в phpMyAdmin

Для удаления базы данных необходимо подтвердить запрос.

Для того чтобы начать работу, следует зайти в базу данных, выбрав необходимую в левой части или в окне баз данных щелкнуть по требуемой ссылке.

Чтобы создать таблицу в новой базе данных test, в окне новой базы данных в поле «Имя» нужно ввести название таблицы, а в поле «Количество столбцов» поставить, например, 2 и нажать «ОК».

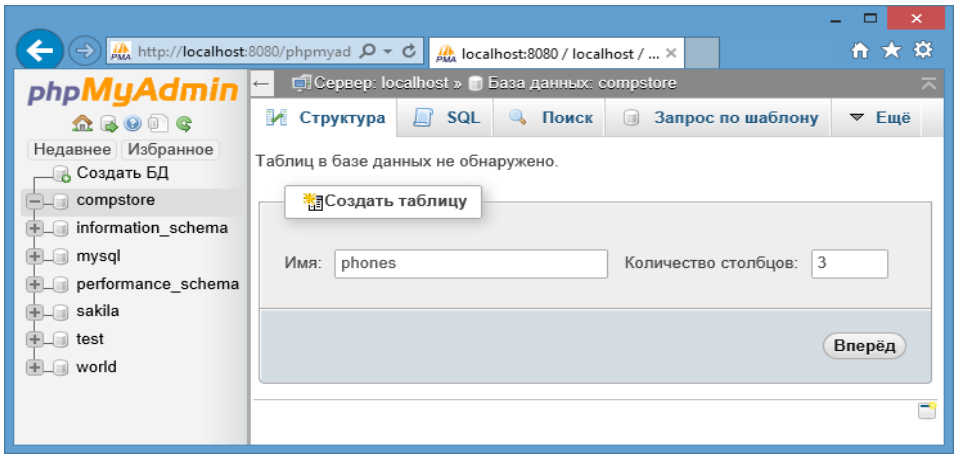


Рисунок 2.31 Создание таблицы в базе данных phpMyAdmin

В следующем окне программа предложит заполнить данные о столбцах:

Имя — имя столбца,

Тип — тип столбца,

Длина — длина столбца,

Сравнение — как будет осуществляться поиск данных,

Атрибуты — атрибуты столбца,

Null — может ли столбец быть пустым,

Индекс — индекс поля,

Комментарий — комментарий к данному столбцу.

Также нужно указать тип столбцов. После ввода всех необходимых данных нажать «Сохранить».

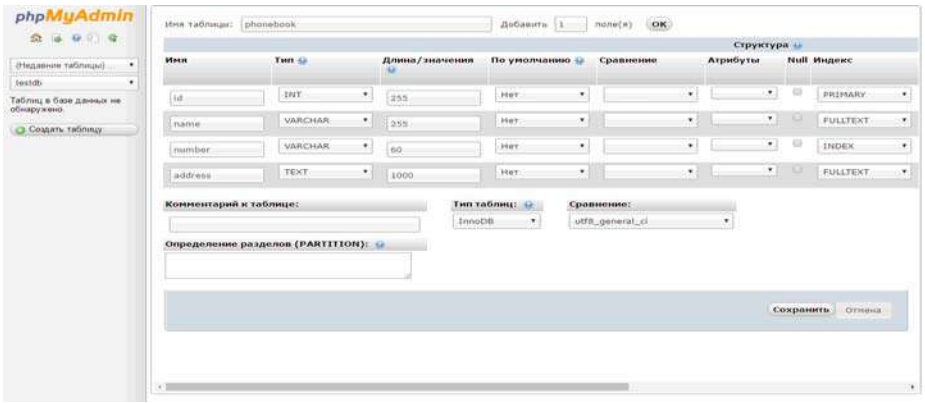


Рисунок 2.32 Таблица phonebook

После создания таблицы можно:

– выполнить SQL-запрос — это делается в меню «SQL»;

– осуществить поиск данных в базе — это делается в меню «Поиск»;

– осуществить запрос по шаблону, который можно сделать в меню «Запрос по шаблону»;

– экспортировать данные базы в различные форматы —выполняется в меню «Экспорт»;

– импортировать данные в базу в меню «Импорт»;

– установить привилегии на базу данных: создать пользователей для данной базы и настроить их доступ к данным — это выполняется в меню «Привилегии»;

– можно удалить таблицу. Для удаления нужно выбрать необходимую таблицу и нажать «Удалить».

Работа с данными

Перейдя по ссылке с названием таблицы, можно открыть созданную таблицу для введения в нее данных.

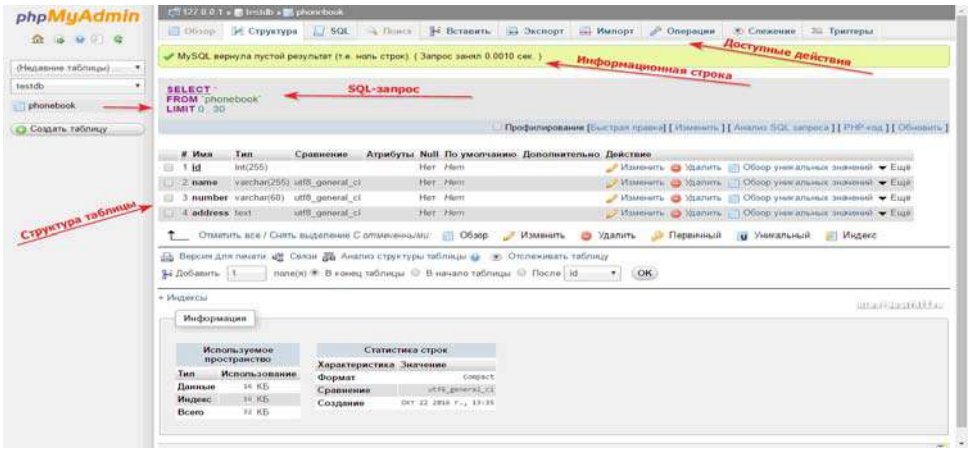


Рисунок 2.33 Структура таблицы

Чтобы добавить данные в таблицу, необходимо перейти в меню «Вставить». Также данные в таблице можно:

- Просматривать,
- Добавлять,
- Удалять,
- Изменять,
- Копировать,
- Осуществлять поиск по различным критериям.

В phpMyAdmin можно добавлять пользователей, назначая им определенные привилегии. Пользователя можно создать как для сервера целиком, так и для отдельной базы данных. На примере тестовой базы данных test можно создать пользователя и назначить ему определенные привилегии. Для этого нужно перейти в базу данных test и в меню выбрать «Привилегии». В следующем окне выбрать «Добавить пользователя».

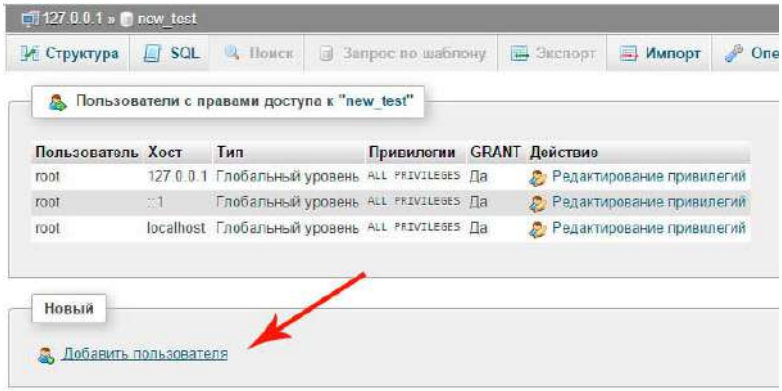


Рисунок 2.34 Добавление пользователя

В открывшемся окне нужно заполнить все поля:

Имя пользователя — логин;

Хост — выбирается ограничение доступа: с любого компьютера, с локальной машины, использовать таблицу хостов или использовать текстовое поле;

Пароль — вводится пароль для данной учетной записи;

Подтверждение — повторяется пароль;

Создать пароль — при нажатии на кнопку «Генерировать» phpMyAdmin автоматически сгенерирует пароль.

После заполнения всех полей нажать «Добавить пользователя».

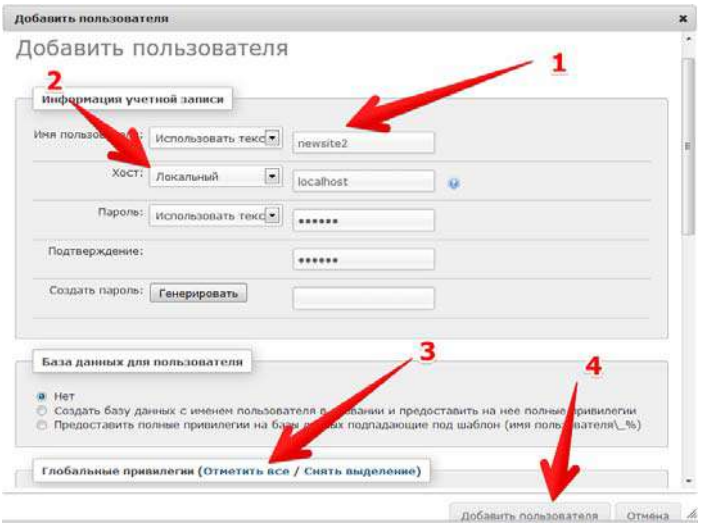


Рисунок 2.35 Настройка добавления пользователя

Для изменения привилегий выбрать «Редактирование привилегий» напротив необходимого пользователя.

2.8 Современные системы управления сайтом: WordPress, Typo3, Joomla!

Система управления контентом (содержимым) или CMS (англ. Content management system) - это набор программ, позволяющий добавлять,

редактировать, удалять содержимое web-ресурсов и сайтов. Такие системы очень удобны для интернет-пользователей, желающих иметь свой ресурс, но мало знакомых с программированием. Для пользователя CMS – идеальный инструмент сайтостроения. У разработчика или владельца сайта есть возможность производить любые действия с его содержимым.

Существует множество разновидностей CMS, у каждой из которых свои особенности, преимущества и недостатки. Некоторые из них предназначены для выполнения конкретных действий, другие универсальны. С их помощью можно производить практически любые изменения своего ресурса, добавлять и удалять данные, изменять контент, редактировать разделы и т.п. Некоторые системы состоят из отдельных блоков, а некоторые представляют собой единый комплекс для управления всеми ресурсами сайта. Существуют платные системы и находящиеся в свободном доступе. Рассмотрим некоторые из них.

Система управления сайтом WordPress

WordPress - это многофункциональная платформа с открытым исходным кодом, ориентированная на создание новостных блогов и различных онлайн-публикаций. Также на базе Wordpress можно создать портфолио, сайты-визитки, бизнес-сайты, интернет-магазины, форумы и фотогалереи. Главным преимуществом WP является доступность множества тематических шаблонов и плагинов, которые делают платформу универсальной.

Система управления сайтом WordPress разработана в 2003 году на базе блогового движка «b2» студентом из города Хьюстона Мэтью Чарльзом Мюлленвегом, в дальнейшем (2005г.) основателем Компании «Automattic».

CMS написана на скриптовом языке программирования общего назначения PHP, интенсивно применяемом для разработки web-сайтов и разнообразных web-приложений; имеет открытый исходный код, доступный для просмотра и изменения любому пользователю, в качестве баз данных использует MySQL.

Лицензия GNU GPL (Универсальная общедоступная лицензия GNU), под которой распространяется WordPress, позволяет не только бесплатно пользоваться платформой, но и свободно копировать, модифицировать и распространять ее.

CMS WordPress является программой, состоящей из набора файлов, содержащих программный код, последовательно объединенных в общую систему, формирующую каркас платформы. За визуальное оформление (дизайн) системы отвечают шаблоны (темы оформления), а за функциональность – разнообразные плагины (дополнительные модули).

Так выглядят сайты на WordPress со стандартными шаблонами «Twenty Ten» и «Twenty Eleven», идущими вместе с движком:

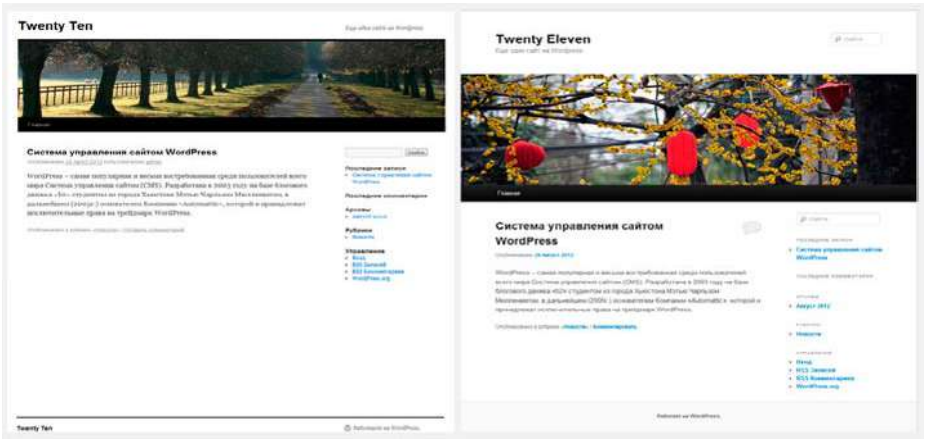


Рисунок 2.36 Пример сайтов на WordPress со стандартными шаблонами «Twenty Ten» и «Twenty Eleven» Адаптировано с сайта (https://ru.wordpress.org/themes/twentyten/)

CMS не требует навыков в области web-программирования. Благодаря простой, удобной и понятной панели управления WordPress вести сайты могут даже неопытные пользователи, не разбирающиеся в PHP, CSS, MySQL, HTML.

Основным достоинством WordPress является гибкость и легкость использования. Удачная конфигурации платформы вместе с бесчисленными шаблонами и плагинами позволяет реализовывать специфичные сайты, например, для поиска работы или продажи недвижимости. WP предназначен для журналистов, блогеров, людей, заинтересованных в написании текстов и желающих опубликовать свои работы. Также платформа полезна представителям малого и среднего бизнеса и индивидуальным предпринимателям, создающим или расширяющим свой бизнес в сфере торговли и услуг. WordPress предлагает бесплатный тариф с необходимыми базовыми инструментами для создания сайта и размещения текстов, блогов или любых других записей. При необходимости можно перейти на платную подписку с дополнительными опциями. Функционал WP позволяет добавлять кнопки социальных сетей, создавать скидочные купоны, строить графики, размещать видео, проводить голосование и опросы.

Преимущества WordPress:

- Абсолютная доступность. Движок WordPress можно скачать с официального сайта ru.wordpress.org и установить бесплатно;
- Открытый исходный код. WordPress является бесплатной популярной системой управления сайтом (CMS) с открытым исходным кодом (OpenSource);
- Большой выбор платных и бесплатных шаблонов и плагинов;
- Понятный интерфейс. С помощью Админ панели можно осуществлять все настройки, включая редактирование шаблонов.

OpenSource — программное обеспечение с открытым исходным кодом. Любой пользователь на свое усмотрение может изменять, дополнять и использовать исходный код абсолютно бесплатно.

Для того чтобы была возможность создавать сайты на CMS на локальном компьютере, необходимо установить локальный сервер. В основном используется база Denwer.

Локальный сервер - специальная программа, позволяющая разрабатывать сайт на локальном (домашнем) компьютере, без необходимости выхода в Интернет. Работа локального сервера полностью имитирует работу реального сервера хост-провайдера. Он состоит из аналогичных компонентов, таких как базы данных MySQL, сервер, поддержки PHP и скрипты для работы с базами данных. Установка на компьютер локального сервера дает возможность запускать web-сервер в любой момент и там, где необходимо.

Также CMS -системы управления сайтом можно устанавливать сразу на хостинг, но это не всегда удобно. Иногда отладка сайта и доработка CMS под конкретные нужды может занимать некоторое время.

Кроме того, процесс отладки CMS на реальном хостинге потребует работы через Интернет, на что понадобится большое количество трафика.

Практическая работа №1
Создание сайта на платформе WordPress

Для создания сайта на базе WordPress перейти на официальный сайт WordPress https://ru.wordpress.com/.

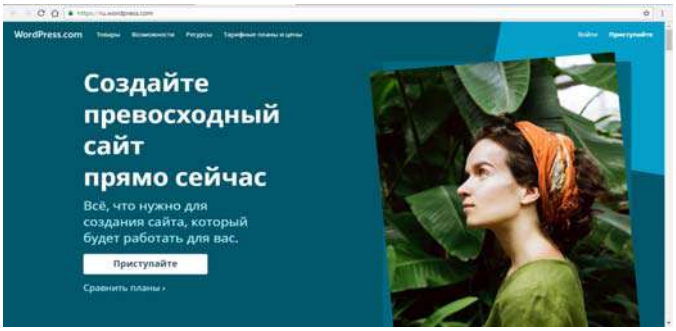


Рисунок 2.37 Главная страница сайта WordPress
*Адаптировано с сайта (*https://ru.wordpress.com/*)*

На главной странице можно увидеть заголовок, в котором предлагается создать сайт на WordPress бесплатно. Для этого нужно нажать на кнопку «Приступайте». Далее откроется страница для описания сайта (рисунок 2.32).

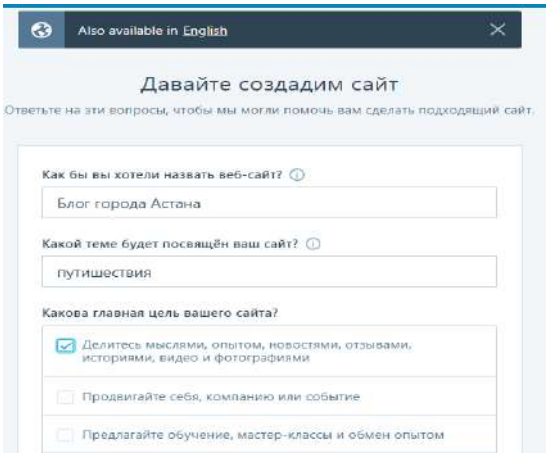


Рисунок 2.38 Описание сайта
*Адаптировано с сайта (*https://ru.wordpress.com/*)*

После этого следует переход на страницу, которая позволит привязать сайт к домену. Домен — это уникальное имя сайта, то самое, которое можно ввести и увидеть в адресной строке. Например, адрес (домен) официального сайта WordPress — wordpress.com. Именно по домену сайты и отличаются друг от друга. В строку поиска вводится желаемое название сайта, например, kazastana. Поскольку платформа позволяет создавать сайт бесплатно, вполне возможно, что желаемый адрес может быть уже занят или просто недоступен. В данном примере WordPress предложил такой вариант домена: kazastana.wordpress.com.

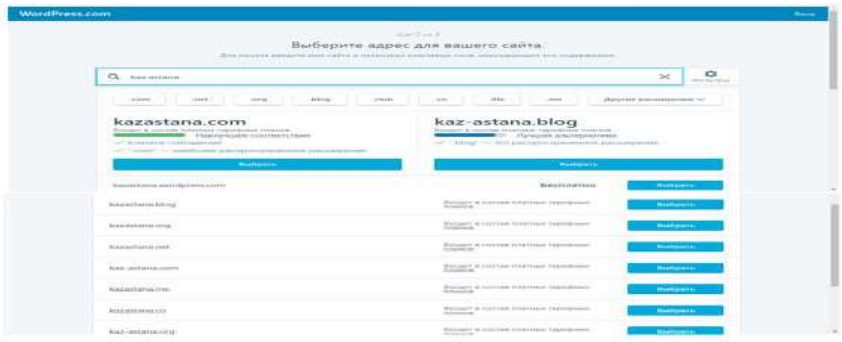


Рисунок 2.39 Выбор домена для сайта
*Адаптировано с сайта (*https://ru.wordpress.com/*)*

На новой странице выбирается тарифный план «Бесплатно».

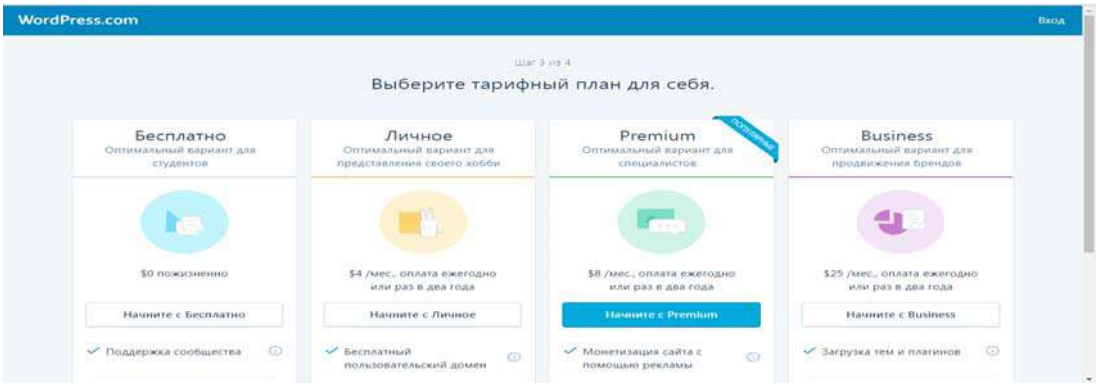


Рисунок 2.40 Выбор тарифного плана
*Адаптировано с сайта (*https://ru.wordpress.com/*)*

Далее путем заполнения полей формы создается учетная запись.

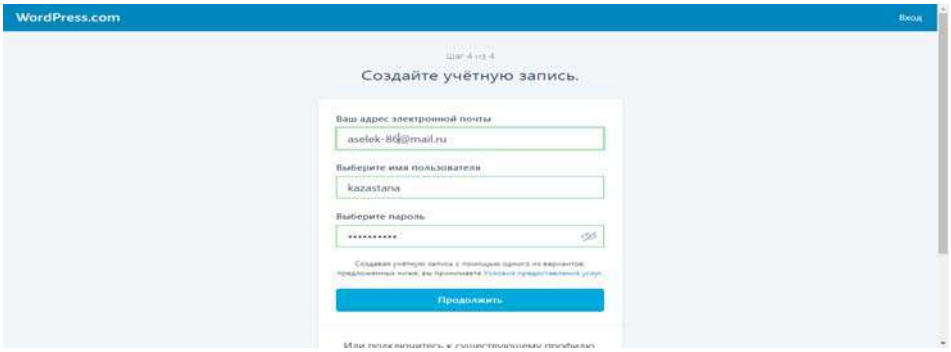


Рисунок 2.41 Создание учетной записи
*Адаптировано с сайта (*https://ru.wordpress.com/*)*

Как и для многих других продуктов с регистрацией, требуется подтвердить факт регистрации, перейдя по ссылке в письме, которое отправлено на указанную электронную почту. Нужно зайти в почтовый ящик (e-mail) и перейти по ссылке подтверждения. После чего выйдет сообщение о подтверждении регистрации.

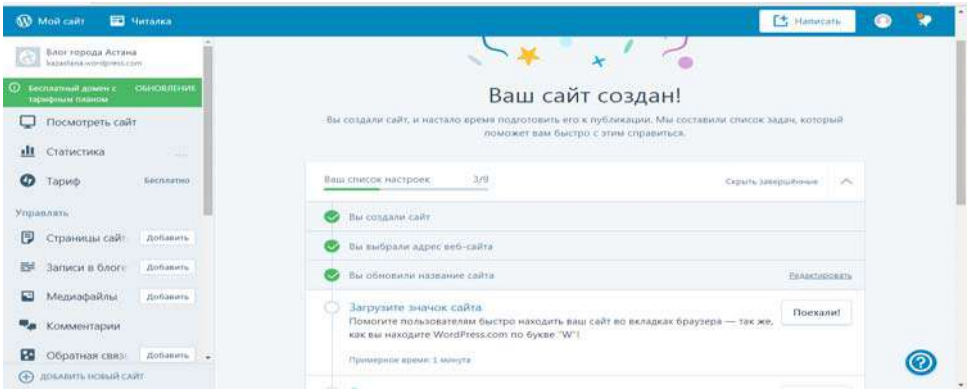


Рисунок 2.42 Подтверждение регистрации
*Адаптировано с сайта (*https://ru.wordpress.com/*)*

Для публикации первой записи в блоге нужно нажать кнопку «Добавить» рядом с «Записи в блоге». Для добавления на сайт разных элементов (текста) необходимо перейти в окно редактирования.

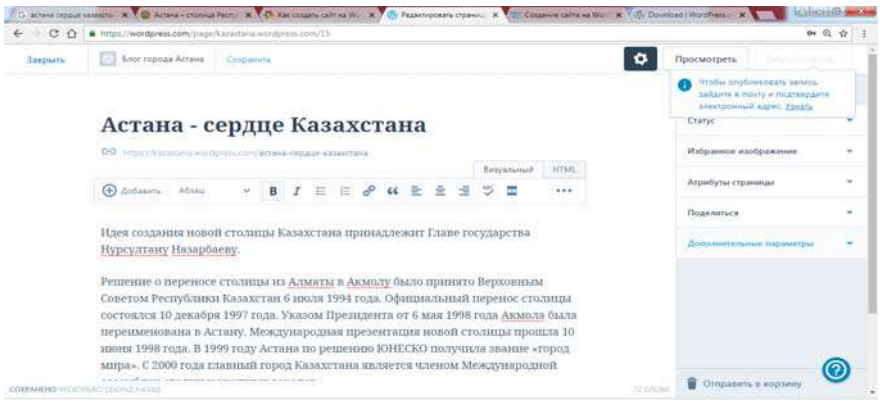


Рисунок 2.43 Окно редактирования
*Адаптировано с сайта (*https://ru.wordpress.com/*)*

После публикации можно будет перейти на сайт — kazastana.wordpress.com — и посмотреть написанную статью.

Рисунок 2.44 Главная страница сайта kazastana.wordpress.com
*Адаптировано с сайта (*https://ru.wordpress.com/*)*

Задание для самостоятельной работы:

На основе вышеприведенного примера создать свой сайт на платформе Wordpress.

TYPO3 - корпоративная система управления сайтом

TYPO3 — система управления сайтом (CMS/CMF*) с открытым исходным кодом и свободной лицензией, написанная на языке PHP. Начало разработки TYPO3 было положено в 1998 году датским программистом Каспером Скархей.

TYPO3 — гибкая и расширяемая система с удобным интерфейсом, большим количеством модулей и функций. Система достаточно универсальна и легко поддается изменению для достижения требуемых целей. Как и многие другие системы с открытым исходным кодом, TYPO3

распространяется под бесплатной лицензией GPL и свободно доступна через Интернет.

Работа системы TYPO3. Система состоит из двух режимов работы. Первый режим Frontend - это шаблон для внешнего вида сайта, который видят пользователи. Второй режим Backend работает только под администратором. В нем можно корректировать работу сайта.

Структура сайта в TYPO3 представлена иерархией страниц. На каждой странице могут быть размещены элементы содержимого: небольшие блоки информации, такие как текст, изображение, таблица, html, плагин и др. TYPO3 основан на множестве шаблонов. Существуют готовые шаблоны, но зачастую их создают с нуля на специальном языке TypoScript. TypoScript не является процедурным языком. Он используется для сочетания и отображения web-сайта и является альтернативой XSLT.

Возможности TYPO3:

- Редактирование текста, редактирование содержимого из frontend и backend,
- Внутренние ссылки и поисковик,
- Одна инсталляция системы для многих сайтов,
- Templavoila,
- Поддержка WML, XML, импорта и экспорта RSS,
- Экспорт в PDF, в статический HTML,
- Кэширование страниц.

Templavoila - альтернативный шаблонизатор для TYPO3. С помощью Templavoila из HTML-шаблона генерируется TYPO3-шаблон без изменений в структуре HTML.

Расширения:

- Гибкая новостная система,
- Форум,
- Интеграция с существующими форумами,
- Галереи изображений,
- Интернет-магазин и каталог,
- Голосования,
- Блог,
- Рассылки,
- Чат,
- Календарь.

Системные требования:

Web-сервер: Apache, IIS.

База данных, ядро системы: MySQL.

ОС: *NIX, MacOSX, Win32.

Размер дистрибутива: около 40 Mb.

Статус: бесплатная.

Тип лицензии: GNU General Public License.

Интерфейс: мультиязычный, в т.ч. русский.

Официальные сайты: typo3.org, typo3.com

*Web Content Management Framework

TYPO3 CMS работает под всеми операционными системами: Windows, Linux, Mac OS X, OS/2 и FreeBSD.

На сегодняшний день TYPO3 работает более чем на 245 000 сайтов. Среди наиболее крупных и известных пользователей данной системы Epson, UNICEF, Philips, Cisco, Konica и другие. Примечательно, что система отвечает очень высоким стандартам безопасности и активно используется большим количеством европейских банков, крупными международными организациями, а также сайтами государственных учреждений.

Признаком того, что интернет-ресурс разработан с помощью TYPO3, является наличие в исходном коде комментария с указанием системы управления содержанием.

TYPO3 считается самой гибкой CMS/CMF из всех существующих, а также входит в первую десятку самых безопасных систем управления сайтами. Также она отличается предельной простотой работы и не требует каких-либо специальных знаний.

Система управления контентом Joomla!

Joomla! представляет собой набор скриптов, написанных на языке программирования PHP. Каждая функция Joomla! выполняется с помощью компонента. Joomla! состоит из основных модулей, которые дают возможность быстро создать сайт, содержащий текстовые материалы, баннеры, каталог ссылок и контактную информацию о какой-либо компании. Материалы группируются по категориям, представляющим собой дерево с неограниченным уровнем вложенности.

Модульность Joomla! позволяет расширять ее функционал практически до бесконечности, устанавливая сторонние компоненты, модули и шаблоны, называемые общим термином расширения.

Внешний вид сайта, основанного на Joomla! можно изменять, устанавливая новые шаблоны оформления. Система состоит из многих простых шаблонов. Найти их можно в Интернете. Разработкой шаблонов занимается множество самостоятельных дизайнеров и web-студий по всему миру. В самой большой галерее содержится несколько сотен бесплатных шаблонов для Joomla!. При этом профессиональных платных шаблонов, разрабатываемых специализирующимися на этом студиями из разных стран, намного больше.

Рисунок 2.45 Примеры сайтов созданных на Joomla!

В web-приложениях выполнение кода происходит в двух сторонах – в серверной и клиентской. К клиентской части относятся HTML, CSS, Javascript, а к серверной - ASP, JAVA, PHP и т.д. Поэтому для работы (выполнения) скриптов Joomla! необходимо наличие web-сервера с поддержкой PHP и MySQL (рекомендуется Apache версии 1.3 и старше) и web-браузера (самые популярные - Internet Explorer, Mozilla Firefox, Opera) у пользователя.

Характеристики Joomla!:

- Полностью основанный на БД движок с использованием PHP/MySQL;
- Модуль безопасности для многоуровневой аутентификации пользователей/администраторов;
- Полностью настраиваемые схемы расположения элементов, включая левый, правый и центральный блоки меню;
- Скачивание изображений при помощи браузера в собственную библиотеку для последующего использования с любого места сайта;
- Форум/Опросы/Голосования для эффективной обратной связи;
- Работа под Linux, FreeBSD, MacOSX, Solaris, AIX, SCO, WinNT, Win2K.

Joomla! настолько универсальна, что позволяет делать сайты любой сложности и для разных целей. Несколько примеров использования CMS Joomla!:

- Корпоративные сайты или информационные порталы,
- Онлайн-газеты, журналы, публикации,
- Сайты для малого бизнеса, некоммерческие и организационные сайты,
- Сайты, основанные на сообществе,
- Персональные или домашние страницы.

CMS Joomla! – бесплатная CMS с открытым исходным (Open Source CMS) кодом. Открытый исходный код означает, что любой желающий может писать под нее свои расширения. Joomla! появилась над ее знаменитой предшественницей Mambo (в прошлом MOS).

Название Joomla! было выбрано на конкурсе, результаты которого оценивали специалисты в области брендинга и маркетинга. Основой для названия послужило слово «Jumla», которое в переводе с суахили означает «все вместе» или «в целом», так как многие команды, участвовавшие в проекте Mambo, были единогласны в стремлении защитить интересы создателей и сообщества, которое и было истинной причиной успеха Mambo.

Главное отличие Joomla! от других систем в том, что она старается сохранить вещи настолько простыми, насколько это возможно, в то же время предоставляя большие возможности.

В качестве web-сервера на компьютере достаточно установить пакет DENWER.

Практическая работа №2
Установка Joomla! на локальном компьютере

Установка Denwer

Для установки Joomla! на локальном компьютере понадобится виртуальный сервер. Чаще всего для этих целей используют программную оболочку под названием «Denwer», которую необходимо скачать с официального сайта denwer.ru.

После перехода на сайт нужно нажать кнопку с надписью: «Скачать Денвер 3»: будет предложена программа для PHP 5.3 или для старой версии PHP 5.2, которая включает в себя больше различных модулей и специальную утилиту Zend Optimizer. Следует выбрать PHP 5.3 и нажать «Скачать».

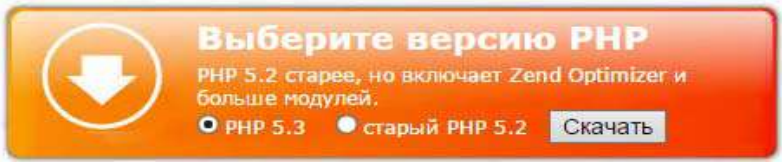
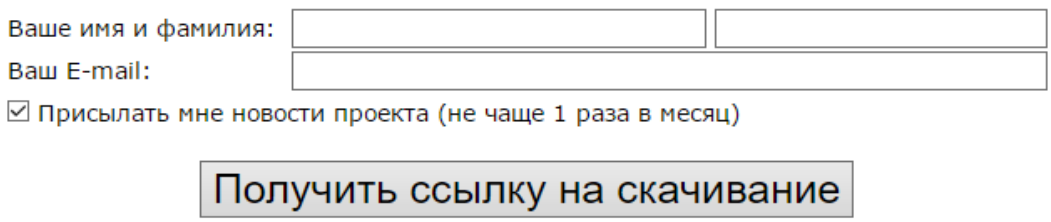


Рисунок 2.46 Загрузка Denwer

На следующем этапе предстоит указать имя, фамилию и действующий электронный адрес, на который будет отправлена персональная ссылка для скачивания программы Denwer. После заполнения всех полей нужно нажать «Получить ссылку на скачивание».



Если все заполнено правильно, появится сообщение: «Проверьте почту. На Ваш E-mail «*****@gmail.com» выслана ссылка для скачивания Денвера».

Письмо обычно приходит в течение нескольких минут.

Далее необходимо перейти в почтовый ящик, куда придет ссылка «Denwer download link for package Base». Перейдя по этой ссылке, нужно скачать программу на локальный компьютер. Установить Joomla! можно не только на Denwer, но и, например, на OpenServer.

После скачивания программы в папке с загрузками появится EXE-файл «Denwer3_Base», который теперь предстоит установить.

Следует запустить установщик и ждать, пока он извлечет нужные файлы.

После извлечения файлов в браузере появится окно «Инсталляция Денвера», которое необходимо закрыть.

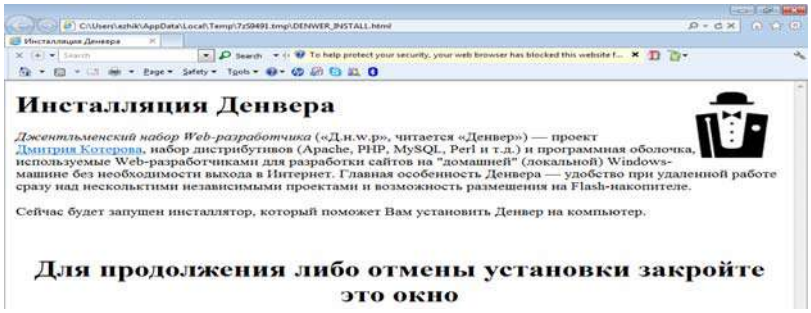


Рисунок 2.47 Инсталляция Denwer

Когда браузер будет закрыт, в окне командной строки на черном фоне появится сообщение с текстом «Для продолжения нажать Enter».

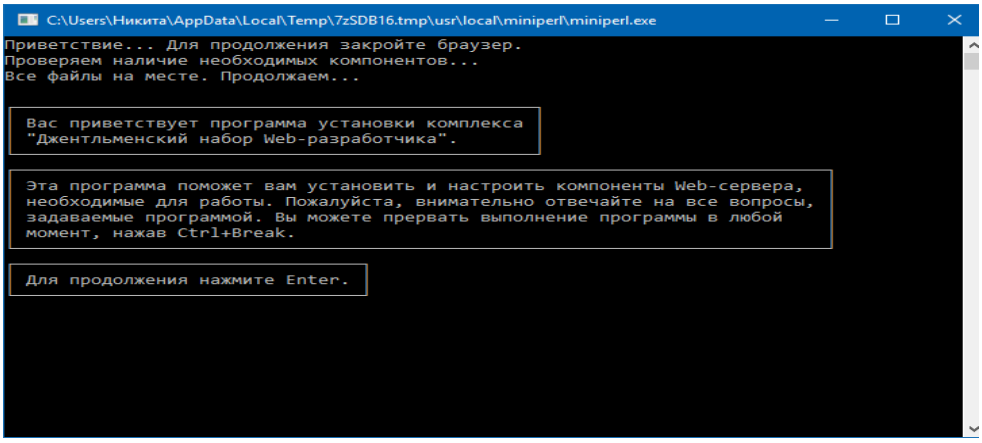


Рисунок 2.48 Установка Denwer

На следующем этапе предстоит указать полный адрес для папки, в которую нужно установить программу Денвер. По умолчанию эта папка называется «WebSerwers» и располагается на диске «C». Здесь не требуется ничего менять, поэтому нужно нажать «Enter».

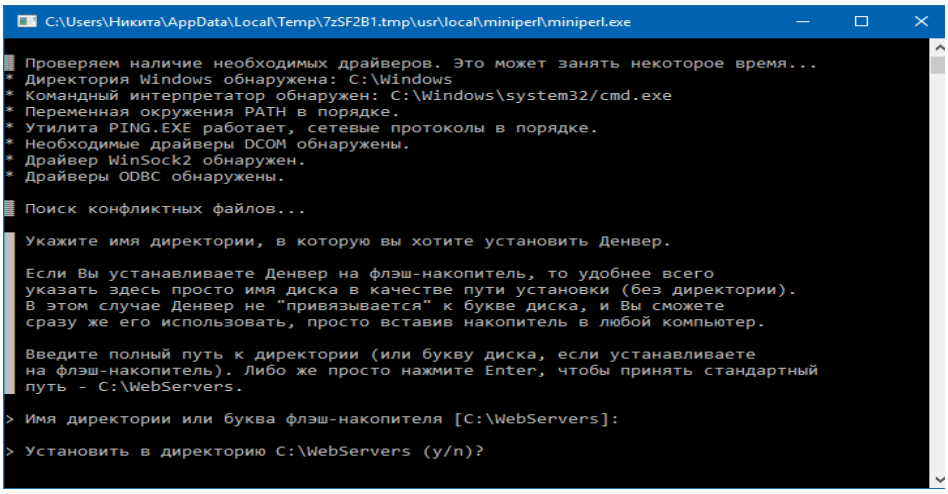


Рисунок 2.49 Выбор директории для сохранения Denwer

Чтобы подтвердить это действие, необходимо нажать клавишу «Y» (от англ. - Yes) на клавиатуре и еще раз «Enter»: установщик создаст виртуальный диск, который необходим для полноценной работы программы. Для продолжения нажать «Enter». На данном этапе предстоит выбрать букву будущего виртуального диска, по умолчанию это буква Z, и в очередной раз нажать «Enter».

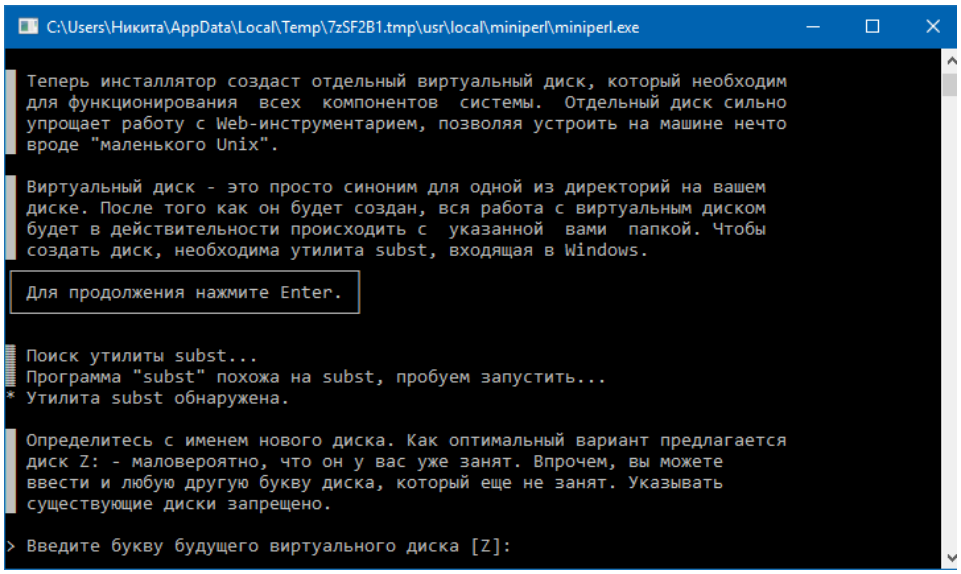


Рисунок 2.50 Установка Denwer

Осталось скопировать все файлы и настройки в созданную директорию C:/WebServers. Для продолжения нажать «Enter». Необходимо дождаться завершения переноса файлов. Программа Denwer может запускаться в двух режимах:

1. Виртуальный диск создается автоматически при загрузке Windows.
2. При загрузке Windows виртуальный диск не создается, но на рабочем столе появятся ярлыки для ручного старта и остановки.

Рекомендуется выбрать вариант 1, потому что он наиболее удобен.

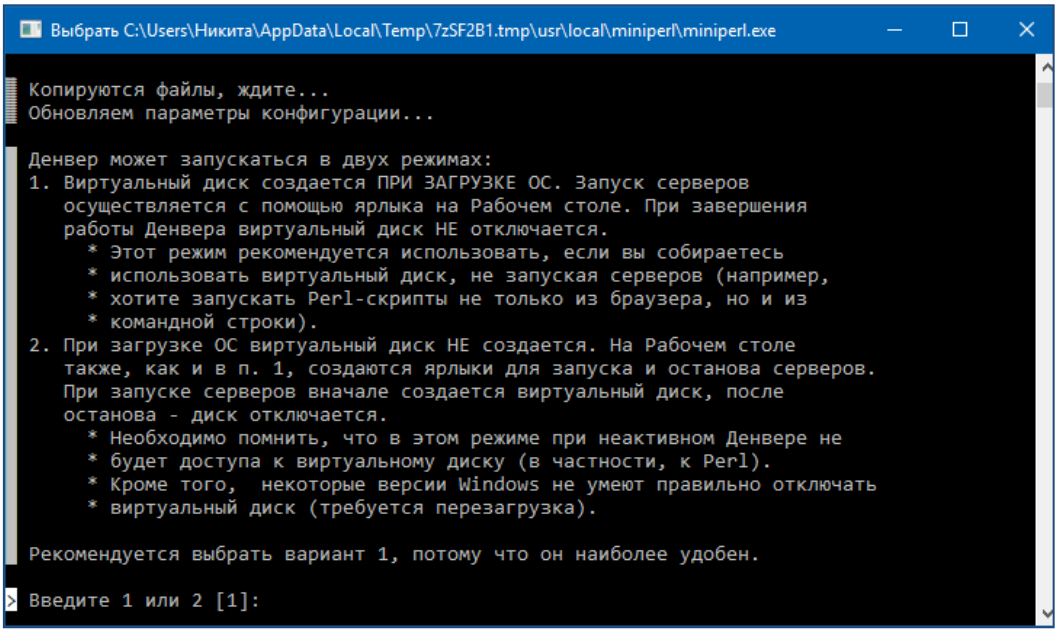


Рисунок 2.51 Выбор запуска Denwer

Вводится цифра 1, и нужно нажать «Ввод» на клавиатуре, после чего подтверждается создание ярлыков для запуска программы на рабочем столе – ввести «Y» и снова нажать «Enter». Если все сделано правильно, откроется окно браузера с заголовком «Денвер успешно установлен». Здесь можно ознакомиться с первоначальными действиями по использованию программы.

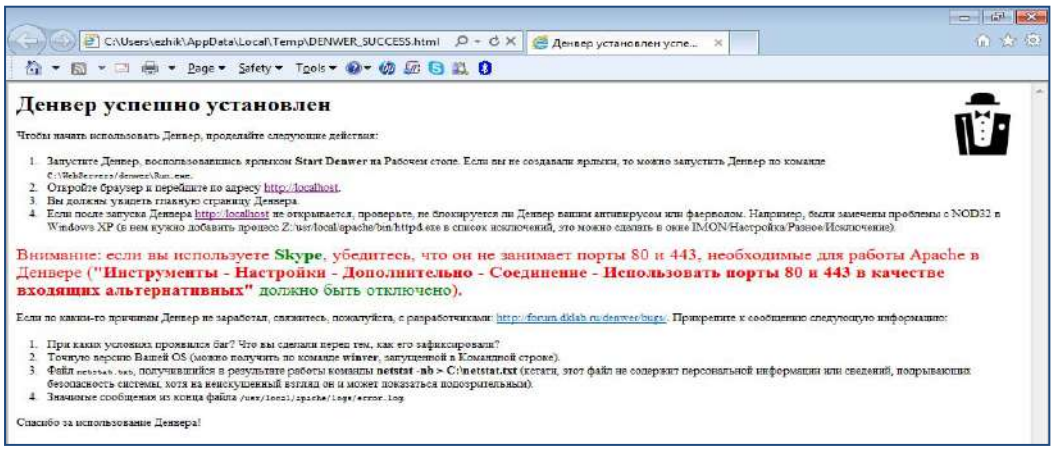


Рисунок 2.52 Окно подтверждения установки программы Денвер

Если не запустится локальный сервер (Денвер), нужно запустить программу Skype, перейти во вкладку Инструменты – Настройки – Дополнительно – Соединения и здесь убрать галочку «Для дополнительных соединений следует использовать порты 80 и 443» (потому что в двух местах идет совпадение по портам). Далее нажать «Сохранить».

Рисунок 2.53 Ярлыки для «Запуска», «Перезагрузки», «Остановки» сервера

Нужно запустить Денвер, нажимая на ярлык Start Denwer. Теперь следует проверить, работает ли Денвер. Для этого в браузере открыть окно с адресом http://Localhost/ и нажать «Enter». Если Денвер установлен правильно, появится следующее окно:

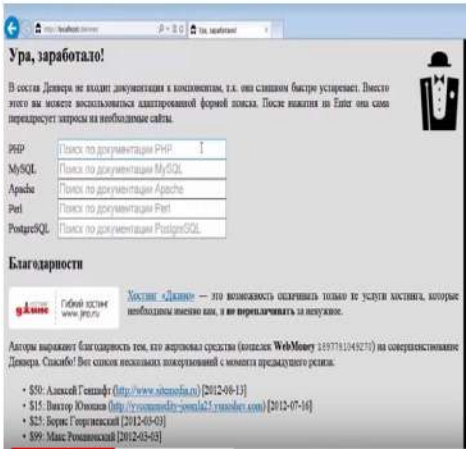


Рисунок 2.54 Окно проверки работы Денвер

Установка Joomla!

Чтобы установить систему управления Joomla! на локальном компьютере, ее необходимо скачать с официального сайта joomla.org.

После перехода на сайт в правой верхней его части необходимо нажать оранжевую кнопку «Download». На следующей странице загрузки нужно кликнуть по синей кнопке «Download Joomla! 3.6.2 English (UK), 3.6.2 Full Package».

Когда скачивание будет завершено, на компьютере в папке с загрузками появится архив «Joomla_3.6.2-Stable-Full_Package», который необходимо распаковать в директорию локального web-сервера.

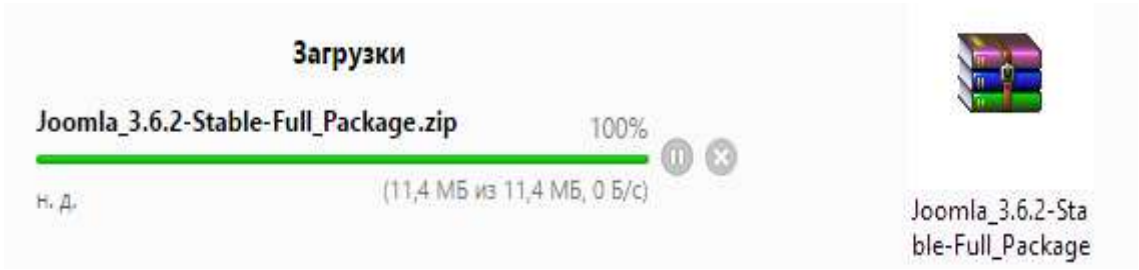


Рисунок 2.55 Скачивание Joomla

Но прежде требуется создать папку с названием будущего сайта. Для этого необходимо перейти по следующему адресу: C:\WebServers\home\localhost\www\ на компьютере и внутри создать папку с названием, например, «mysite.ru».

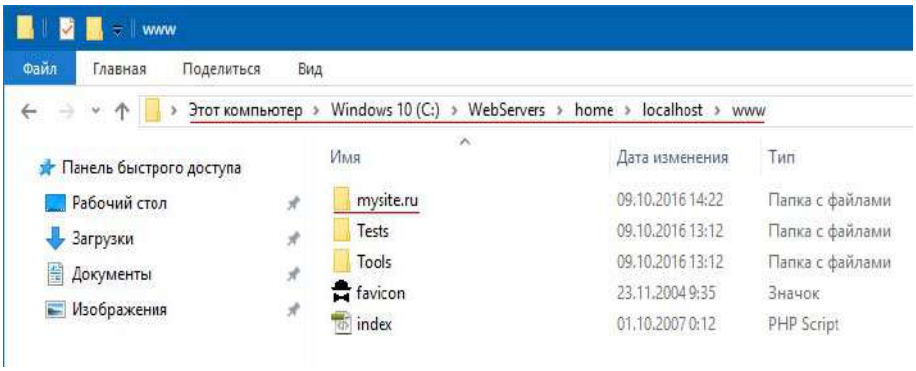


Рисунок 2.56 Файлы сайта

Удерживая курсор на скачанном ранее архиве, нажать правую кнопку мыши и выбрать из появившегося списка пункт «Извлечь файлы».

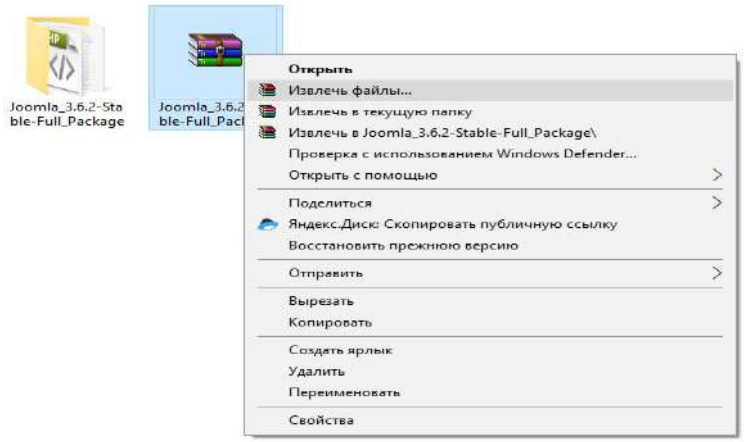


Рисунок 2.57 Извлечение файлов

В следующем окне выбрать расположение недавно созданной папки с будущим сайтом. Нажать «ОК» и дождаться завершения распаковки файлов.

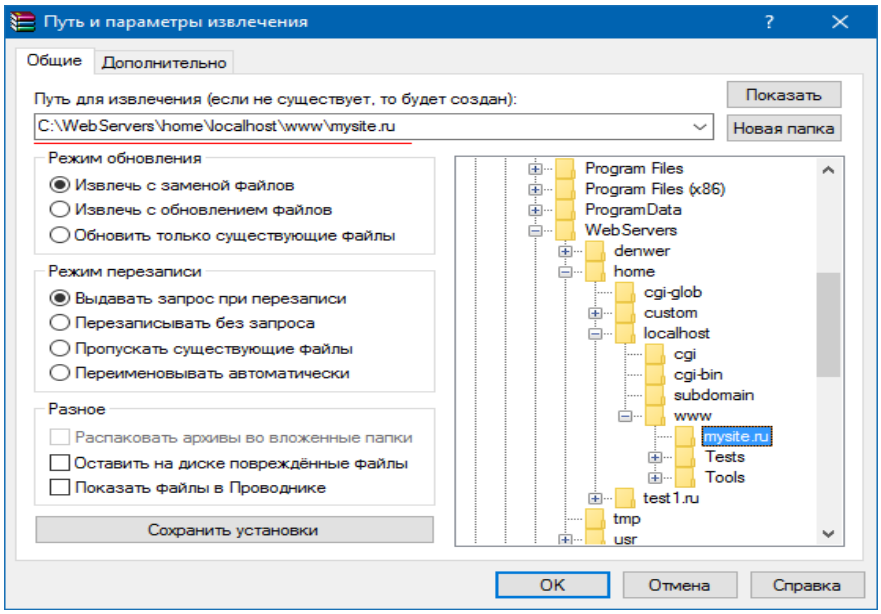


Рисунок 2.58 Путь и параметры извлечения

Настройка Joomla!

После извлечения необходимых файлов сайт почти готов к работе. Осталось настроить подключение к MySQL, распаковать сайт и выбрать для него подходящий шаблон.

Прежде всего следует открыть страницу http://localhost/mysite.ru в браузере и завершить конфигурацию сайта, заполнив все необходимые поля: выбрать язык, придумать название и краткое описание для сайта.

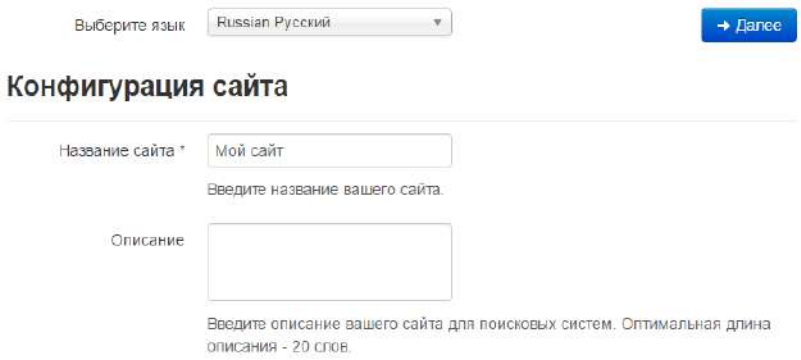


Рисунок 2.59 Конфигурация сайта

Также указать действующий e-mail в качестве электронного адреса администратора, придумать логин и пароль. Нажать «Далее» в нижней части страницы.

Следующий этап — это конфигурация MySQL базы данных. Чтобы создать базу данных для Joomla!, в браузере необходимо открыть страницу http://localhost/tools/phpmyadmin/ и перейти в соответствующую категорию под названием «Базы данных».

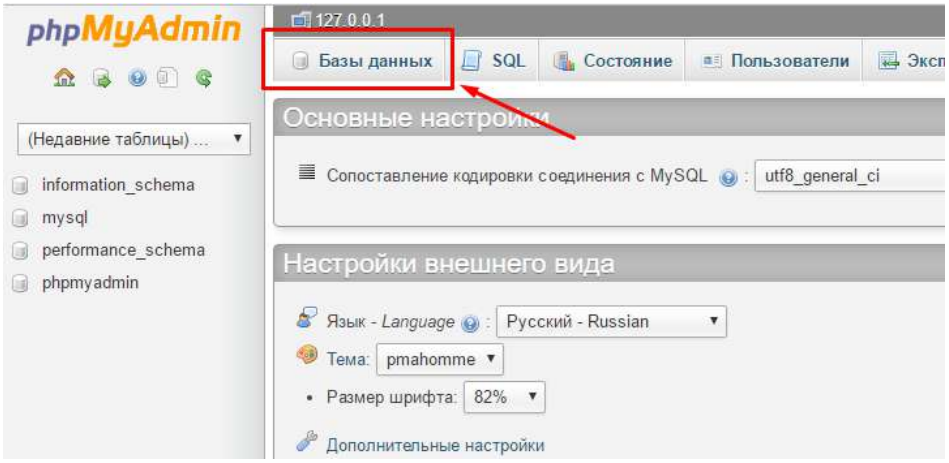


Рисунок 2.60 Конфигурация MySQL базы данных

На следующей странице указать имя для новой базы данных – joomla **и** выбрать кодировку сайта (чаще всего используется utf8_general_ci)**.**

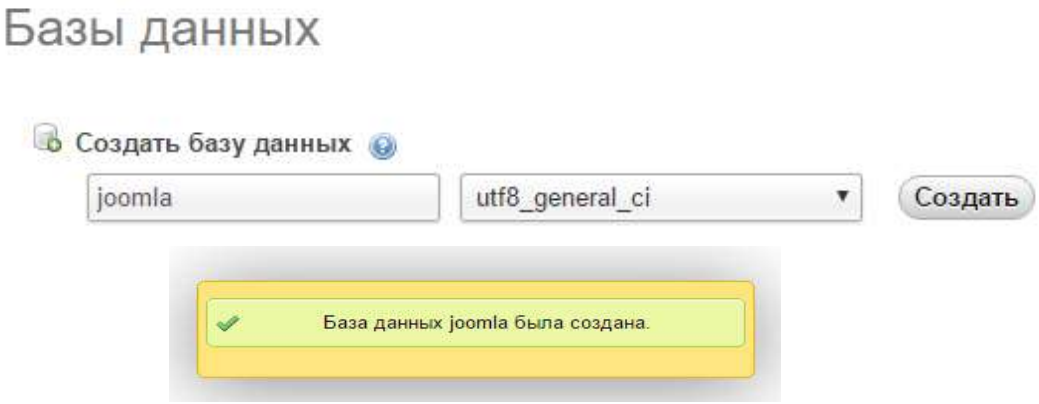


Рисунок 2.61 Создание базы данных MySQL

Далее нажать кнопку «Создать». Если конфигурация проходит правильно, появится сообщение «База данных joomla была создана».

Затем следует перейти во вкладку «Пользователи» и кликнуть по ссылке «Добавить пользователя».

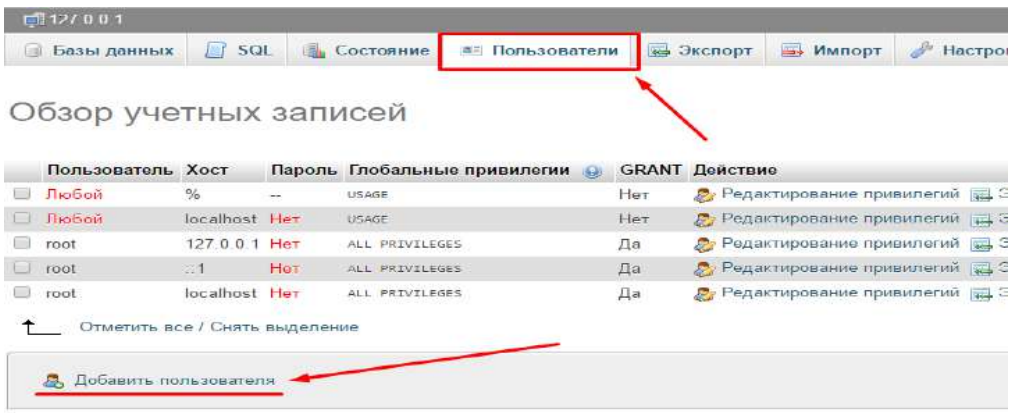


Рисунок 2.62 Добавление пользователей базы данных MySQL

В появившемся окне необходимо заполнить информацию для новой учетной записи. Придумать имя пользователя, пароль и выбрать локальный хост. При желании пароль можно сгенерировать.

- Имя пользователя: joomla

- Хост: Локальный (localhost)

- Пароль: *********

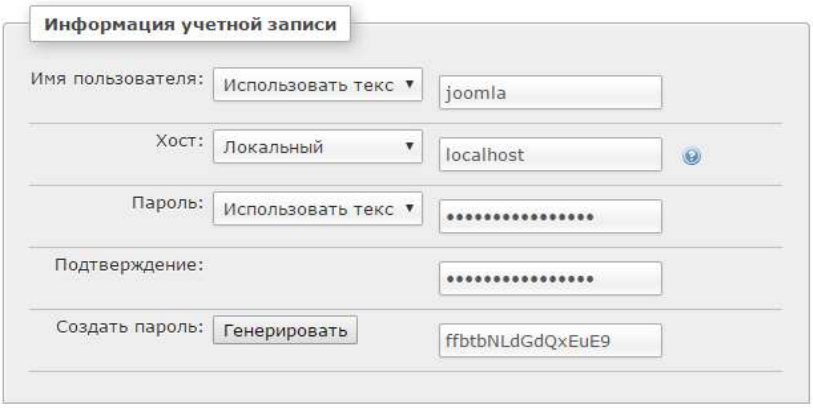


Рисунок 2.63 Настройка учетной записи

После заполнения полей необходимо найти блок «Глобальные привилегии» и выбрать «Отметить все», чтобы установить для пользователя максимальный набор привилегий.

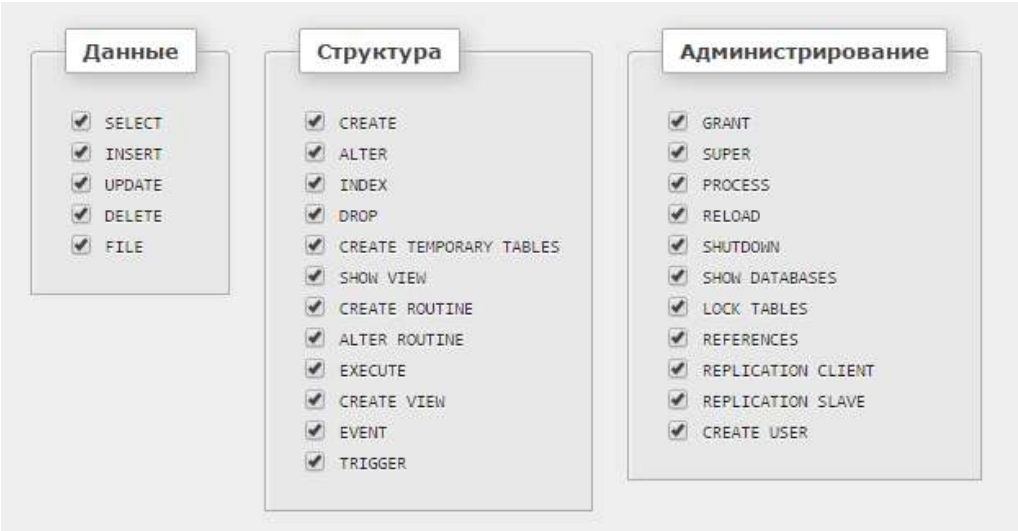


Рисунок 2.64 Настройка учетной записи

После заполнения и отметки всех полей нажать на кнопку «Добавить пользователя» и, если все сделано без ошибок, можно увидеть всплывающее сообщение «Был добавлен новый пользователь».

Далее необходимо возвратиться на страницу конфигурации базы данных сайта и указать недостающие данные: имя пользователя, пароль и

имя базы данных, а префикс таблицы оставить неизменным. Нажать «Далее» и перейти к обзору настроек сайта. Через несколько секунд появится сообщение «Поздравляем, вы установили Joomla!»

Прежде чем приступить к работе с сайтом и панелью администратора, обязательно нужно удалить все установочные файлы, включая папку «INSTALLATION». Для этого достаточно нажать по оранжевой кнопке «Удалить директорию installation» и дождаться сообщения о том, что данная директория успешно удалена.

Важно полностью удалить директории INSTALLATION, так как без этого этапа установка Joomla! не будет завершена.

3. Размещение сайтов в сети Интернет

Современный мир невозможно представить без сети Интернет, которая стала неотъемлемой частью жизни каждого человека. В настоящее время широчайшие возможности web-ресурсов уже не ограничиваются исключительно текстовой информацией и электронной почтой, а включают в себя социальные сети, чаты, форумы, блоги, файлообменники, онлайн-магазины и так далее. Все это оказывает существенную помощь

каждому человеку в его повседневной деятельности, позволяя быть в курсе последних новостей, а также способствуя поиску актуальной информации.

Важно заметить, что схема создания любого сайта в целом не является слишком сложной, поскольку многочисленные шаблоны в значительной степени облегчают разработку web-страницы. В то же время для создания действительно эффективного сайта, который будет приносить прибыль, следует внимательно изучить многие web-инструменты, а также понять схему их расположения на web-ресурсе.

3.1 Выбор хостинг-провайдера

Сегодня все чаще встречаются новые термины: «хостинг-провайдер», «хостер», «хостинг», «виртуальный хостинг», «физический хостинг». Это говорит о том, что Интернет прочно вошел в обыденное сознание людей. В XXI веке информационное пространство человека значительно расширилось за счет ресурсов Интернета. Представление о хостинге становится таким же важным для современного человека, как понятие телевидения, радио, газет и журналов.

Хостинг (англ. hosting) - услуга по предоставлению вычислительных мощностей для физического размещения информации на сервере, постоянно находящемся в сети (обычно Интернет). Под понятием услуги хостинга подразумевают как минимум услугу размещения файлов сайта на сервере, на котором запущено ПО, необходимое для обработки запросов к этим файлам (web-сервер). Как правило, в услугу хостинга уже входит предоставление места для почтовой корреспонденции, баз данных, DNS, файлового хранилища и т.п., а также поддержка функционирования соответствующих сервисов.

Проще говоря, хостинг - это место на сервере со специальным программным обеспечением. Причем сервер всегда должен находиться в сети, чтобы необходимый сайт работал круглосуточно.

Выбрать хороший хостинг - непростая задача. Множество компаний предлагают услуги хостинга по разным ценам, существует даже условно-бесплатный хостинг.

Большие интернет-проекты, которые посещают сотни тысяч пользователей ежедневно, размещают свои сайты на отдельных серверах и платят за это немалые деньги. Владельцу обычного коммерческого или личного сайта разместить его на отдельном сервере не по карману, да и ни к

чему. Поэтому подавляющее большинство сайтов в Интернете пользуется виртуальным хостингом.

Виртуальным он называется потому, что на диске одного сервера размещено несколько сайтов. Это немного напоминает общежитие. Однако владельцам сайтов такое «соседство» увидеть невозможно. Каждый владелец сайта (на качественных хостингах) имеет собственную панель управления, так называемую панель администратора, с помощью которой можно управлять всеми аспектами размещенного сайта.

Сайты, размещенные на виртуальном хостинге, совершенно автономны и не могут повлиять друг на друга или причинить друг другу вред.

Хостинг нужно проверить перед тем, как разместить на нем сайт.

Существует три самых важных показателя серверов хостинг-провайдеров, по которым можно осуществлять выбор:

1. Uptime - время бесперебойной работы сервера в течение установленного промежутка времени. Он измеряется в процентах (%) и должен быть близок к 100%.

Подавляющее большинство хостингов уверяют, что Uptime их серверов равен 99,99%. Если Uptime измеряется каким-нибудь специальным сервисом, то величина более 99,75% может считаться вполне приемлемой. Разместить интернет-проект на хостинге с Uptime равным 100% просто невозможно, так как любой хостинг должен останавливаться на профилактику и обслуживание.

Найти в Интернете сервис, проверяющий доступность любого сервера не проблема. Но так как для измерения этого показателя требуется длительное время, самому измерить Uptime выбираемого хостинга очень проблематично. И в большинстве случаев приходится доверять данным его владельцев. Если на странице сайта хостинга размещена информация с реальным Uptime, то это показатель его высокой надежности.

2. Скорость загрузки страниц.

После размещения сайта в Интернете необходимо, чтобы его страницы достаточно быстро загружались при просмотре их посетителями. Если загрузка идет долго, посетителю это просто надоест, и он уйдет, так и не дождавшись открытия страницы. Скорость загрузки страниц измеряется в килобайтах в секунду. (КБ/сек.)

Прежде чем разместить сайт на хостинге, этот показатель нужно проверить. В отличие от Uptime, скорость загрузки страниц легко проверить. Это можно сделать, например, с помощью сервиса host-tracker.com. Но нельзя принимать решение о размещении сайта на хостинге на основе показателей только одного выбранного хостинга. В первую очередь обязательно нужно произвести измерение скорости загрузки страниц нескольких хостинг-провайдеров.

Нужно обратить внимание: сервис host-tracker.com производит замер скорости загрузки более чем из 100 точек по всему миру и выдает данные измерений как по различным городам разных стран, так и усредненный результат. Именно этим, результирующим, показателем и нужно

пользоваться. Дело в том, что измеренная скорость зависит не только от работы серверов, но и от внешних каналов связи. Для разных городов условия связи сильно различаются и могут со временем изменяться. На решение разместить сайт у конкретного хостинг-провайдера должна оказать влияние именно результирующая цифра.

3. Время ответа сервера - это время, которое проходит с момента, когда браузеру дана команда начать загрузку страницы (нажатием клавиши или кнопки) до момента старта загрузки.

Учитывать нужно именно результирующие значения. Окончательный выбор хостинга для сайта следует делать после тщательного измерения и сравнения всех перечисленных показателей.

Из двух показателей, которые реально можно измерить, конечно же, важнее скорость загрузки страниц. Так как для каждого посетителя того или иного сайта пауза перед началом загрузки страницы привычна. Психологически оценивать сайт посетитель начинает одновременно с началом загрузки страницы. Задержка загрузки, превышающая 1,5 - 2 секунды, раздражает посетителей, а меньшее значение задержки большинство из них не замечает.

3.2 Хостинг

Хостинг – это место на дисковом пространстве жесткого диска серверной машины, которое обеспечивает круглосуточный доступ к информации, находящейся на нем. Хостинг предназначен для обеспечения работы информационных ресурсов, имеющих базу данных, а также интерфейс для взаимодействия с этими базами данных. Неотъемлемой частью работы хостинга по определению местоположения того или иного ресурса является доменное имя.

Выделяют следующие виды категорий хостинга: классический – в него входит «Виртуальный хостинг» и «VPS». Для обычных web-сайтов, таких как сайт-визитка, корпоративный сайт, landing page, которые несут в себе презентационных характер, используют в основном «Виртуальный хостинг». Для сложных систем, в которых используются особые процессы резервного копирования, где необходима своя среда работы с определенными компонентами, применяют «VPS».

Основные хостинг-провайдеры обрабатывают заказы в автоматическом режиме. Для приобретения хостинга на сайте хостинг-провайдера нужно выбрать необходимый вид, заполнить поля с общими регистрационными данными. После этого система автоматически выставит счет, который можно в онлайн-режиме оплатить удобным способом. Как только счет будет оплачен, биллинговая система автоматически вышлет на адрес электронной почты доступы к купленному хостингу для произведения работ по установке сайта на сервер.

Перед размещением сайта в сети Интернет следует провести его **тестирование**, т.е. убедиться в том, что он правильно отображается разными браузерами:

– тексты хорошо читаются на выбранном фоне,
– рисунки расположены на своих местах,
– гиперссылки обеспечивают правильные переходы и т. д.

Для функционирования web-сайта необходимы специальные условия для работы компонентов связки базы данных и файлов системы управления. Условно процесс установки сайта на сервер можно разделить на 5 этапов:

1. Загрузка файлов на сервер.
2. Создание базы данных и пользователей базы данных.
3. Подключение и настройка установочного пакета.
4. Установка.
5. Настройка параметров сайта.

1. Перейти в панель управления хостингом, ввести в соответствующие поля логин и пароль, который предоставил в письме хостинг-провайдер. Рисунок 3.1

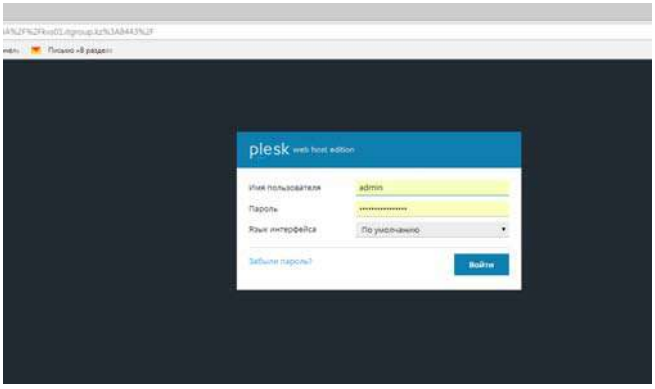


Рисунок 3.1 Панель управления хостингом

2. После авторизации откроется основная панель хостинга, в которой доступен широкий спектр настроек. Для работы необходимы только «Менеджер файлов», «Базы данных» и «Настройки хостинга». Рисунок 3.2

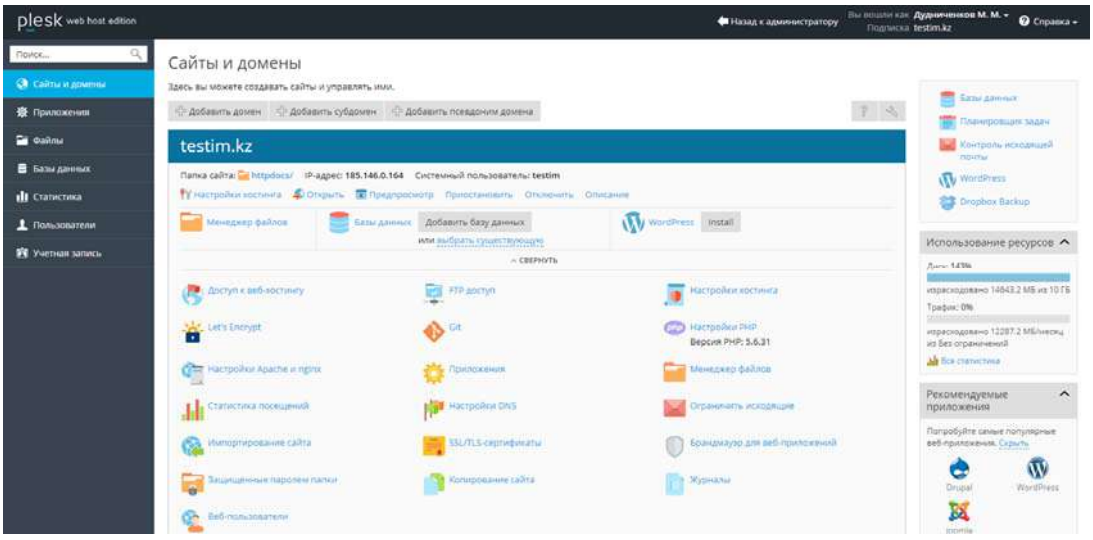


Рисунок 3.2 Основная панель хостинга

3. В открывшейся панели выбрать папку «Менеджер файлов». Рисунок 3.3

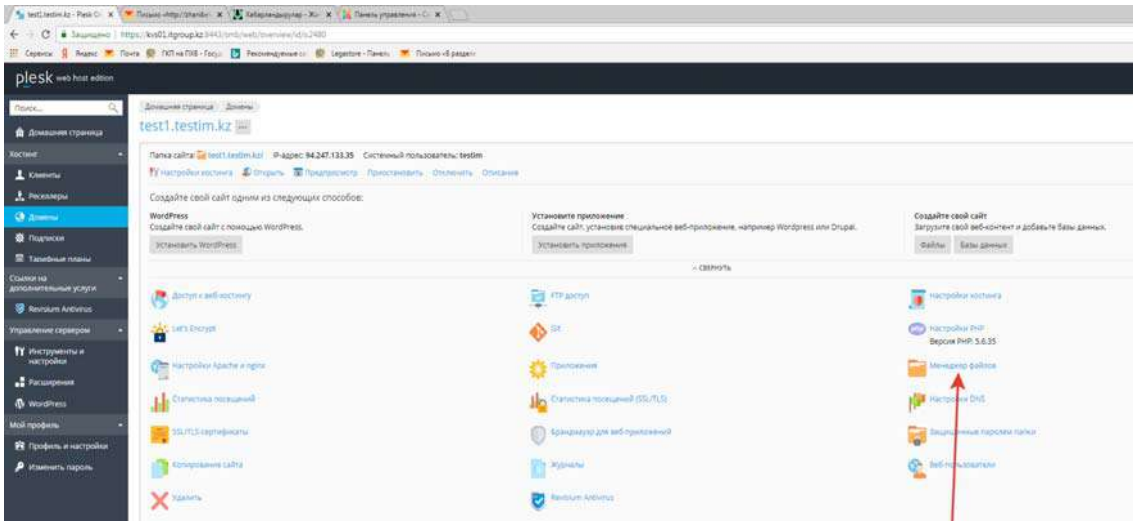


Рисунок 3.3 Основная панель хостинга. Выбор папки Менеджер файлов

4. После того, как открылся «Менеджер файлов», в корневой каталог необходимо загрузить исходные файлы сайта: нажать на кнопку «Загрузить». Рисунок 3.4

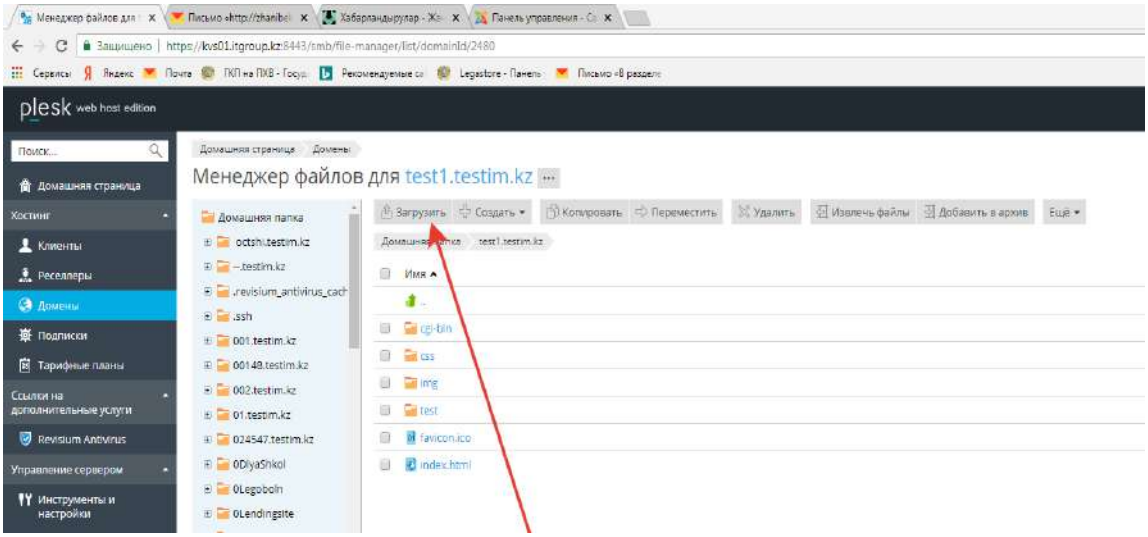


Рисунок 3.4 Загрузка исходных файлов

5. После нажатия загрузки откроется окно для выбора архива сайта с компьютера. Рисунок 3.5

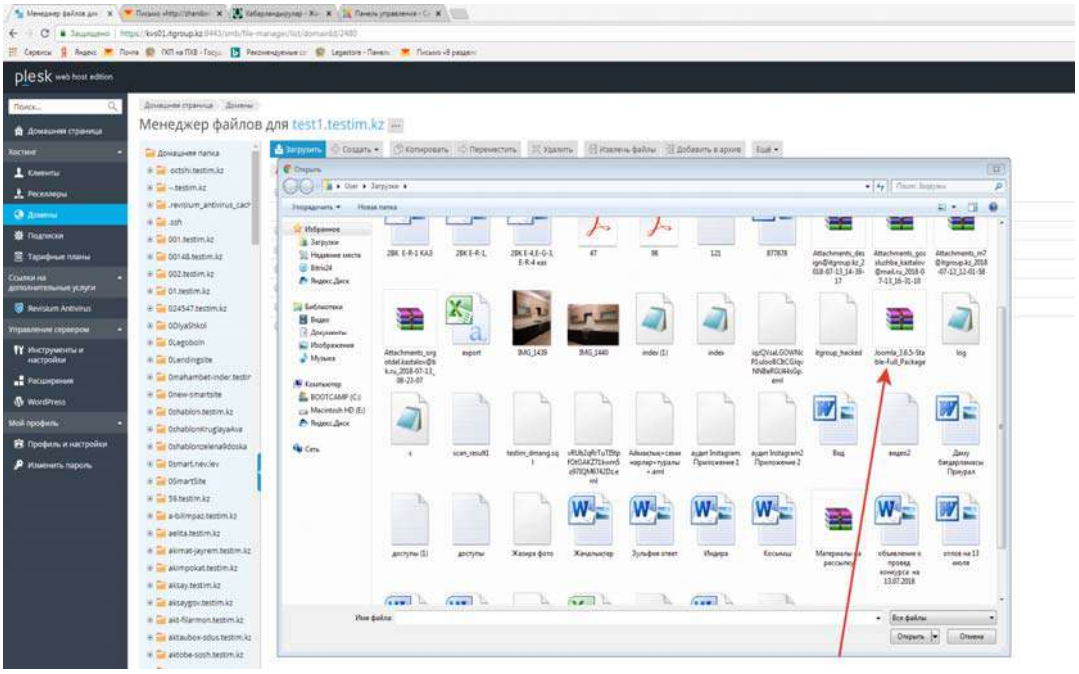


Рисунок 3.5 Выбор архива сайта с компьютера

6. Загрузившийся файл выделить и извлечь, нажав на кнопку «Извлечь файлы». Рисунок 3.6

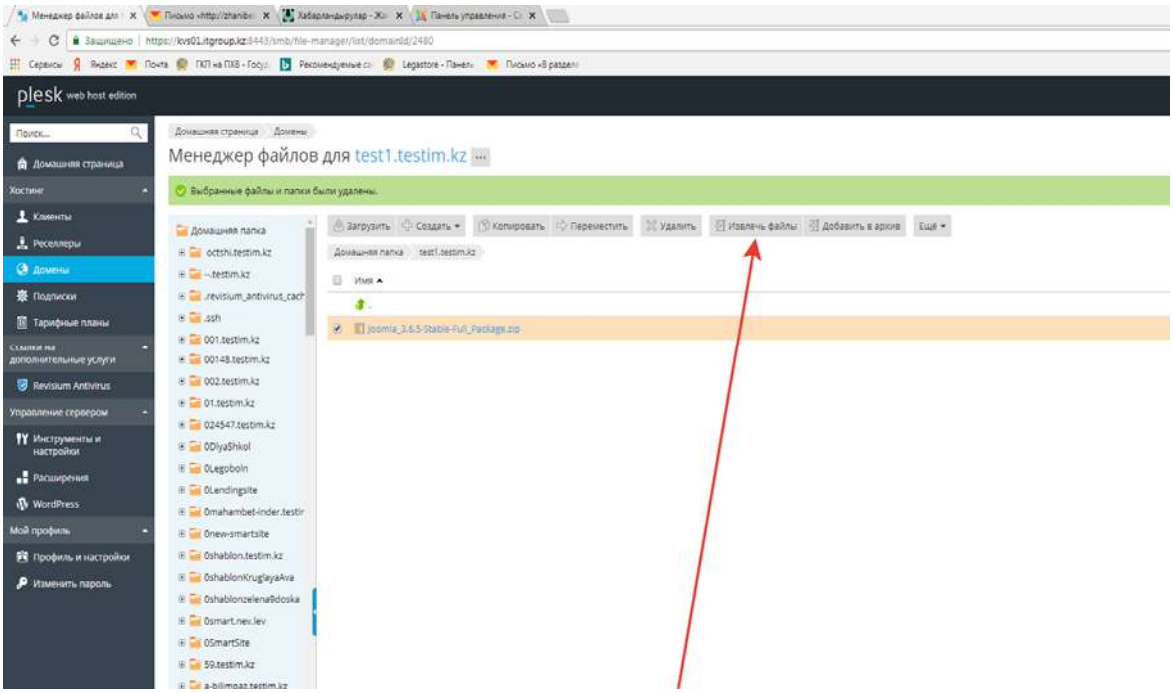


Рисунок 3.6 Извлечение файлов из загруженного архива

7. Далее необходимо создать базу данных и пользователя: нажать на вкладку «Базы данных». Рисунок 3.7

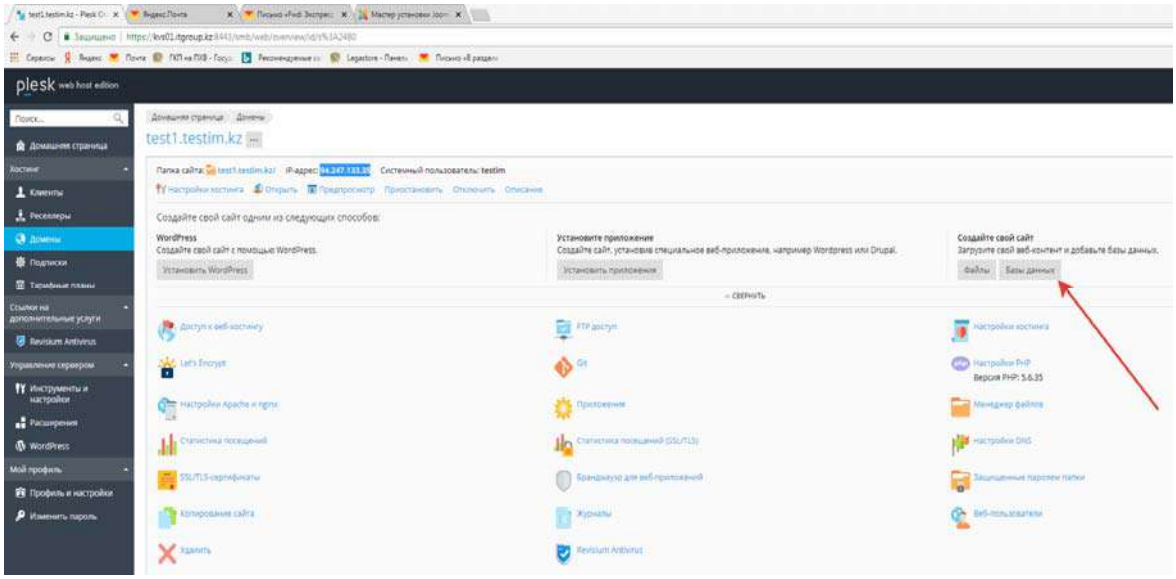


Рисунок 3.7 Основная панель хостинга. Вкладка «Базы данных»

8. Откроется список существующих баз данных (не обязательно). Для создания новой выбрать «Добавить базу данных». Рисунок 3.8

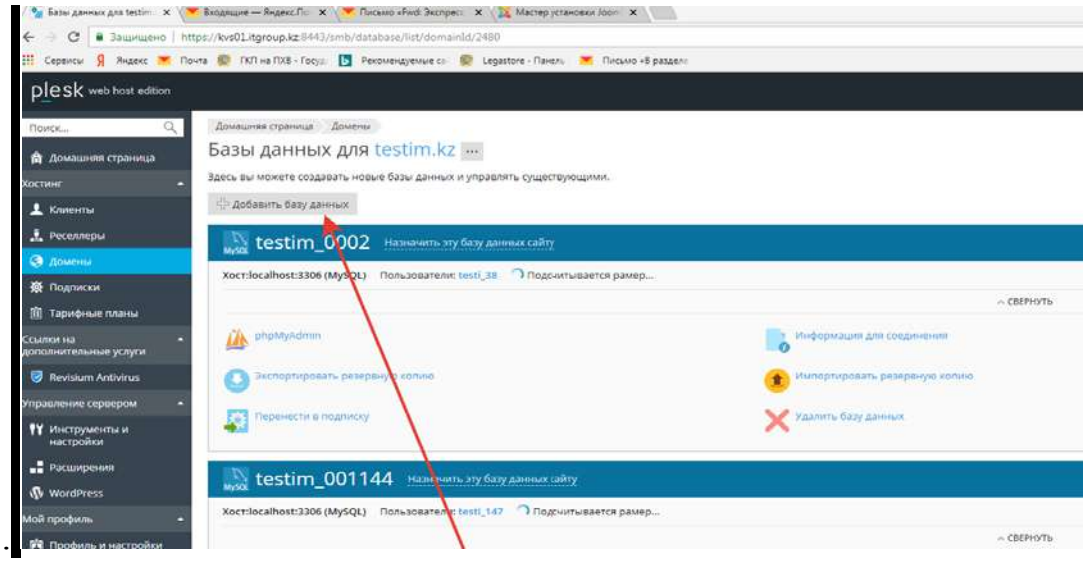


Рисунок 3.8 Добавление новой базы данных

9. После добавления базы данных откроется следующее окно с полями. Необходимо создать имя базы данных, имя пользователя базы данных и пароль. Все параметры следует запомнить или сохранить, записав в Блокнот. Рисунок 3.9-3.11

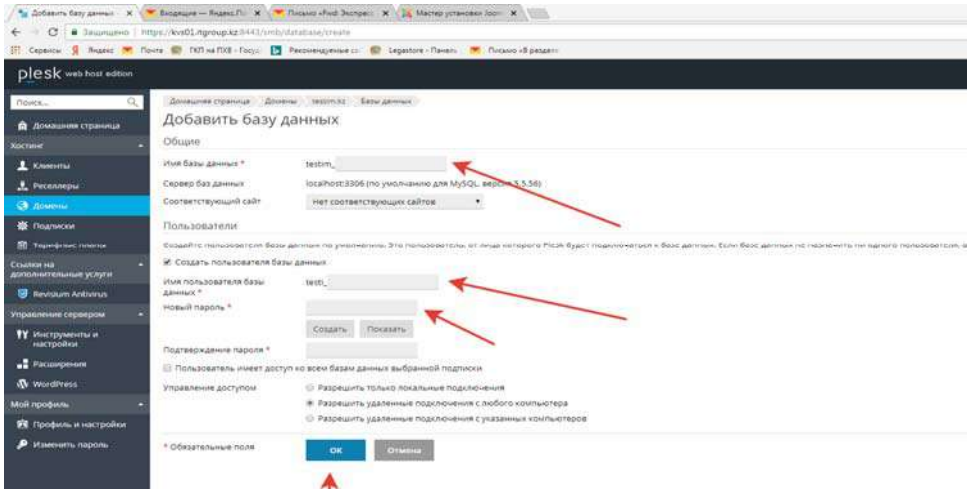


Рисунок 3.9 Заполнение обязательных полей во вкладке «Добавить базу данных»

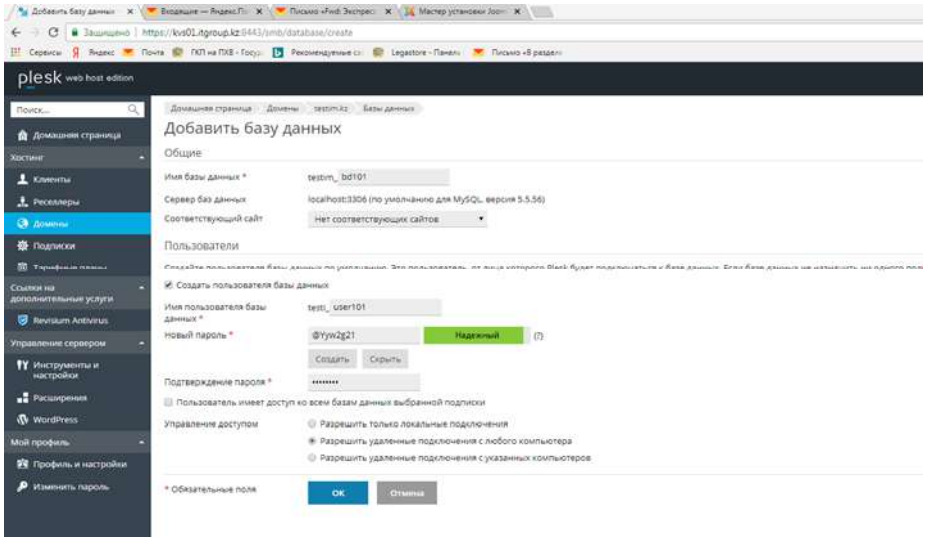


Рисунок 3.10 Заполнение обязательных полей во вкладке «Добавить базу данных»

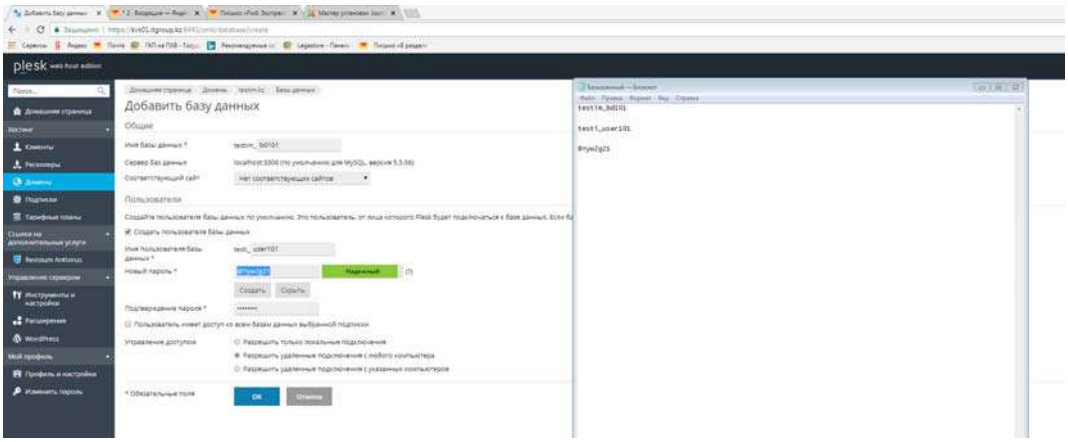


Рисунок 3.11 Сохранение Имени пользователя базы данных и пароля в Блокнот

10. Открыть новую вкладку в браузере и в строке набрать адрес, созданного домена. Домен запустит процесс инсталляции системы управления сайтом. Рисунок 3.12

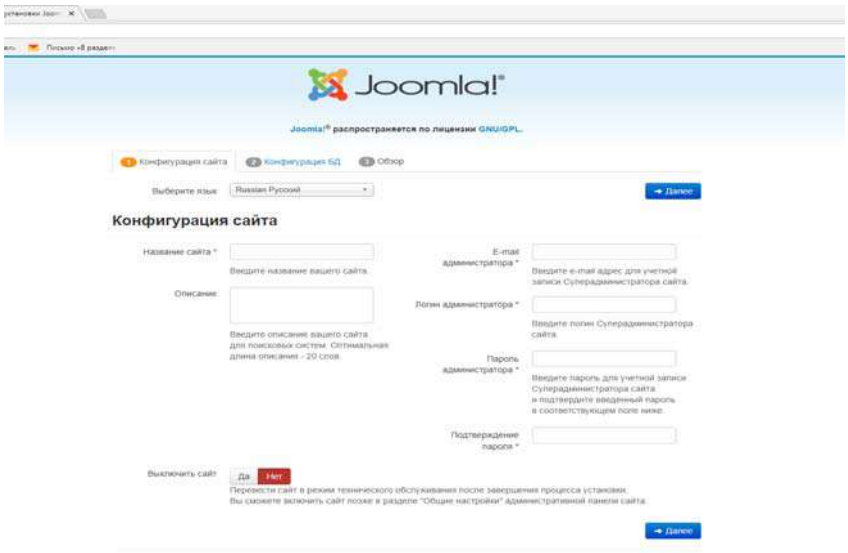


Рисунок 3.12 Инсталляция системы управления сайтом

11. В открывшемся окне заполнить поля: название сайта, описание, e-mail, пароль и логин для входа. Рисунок 3.13

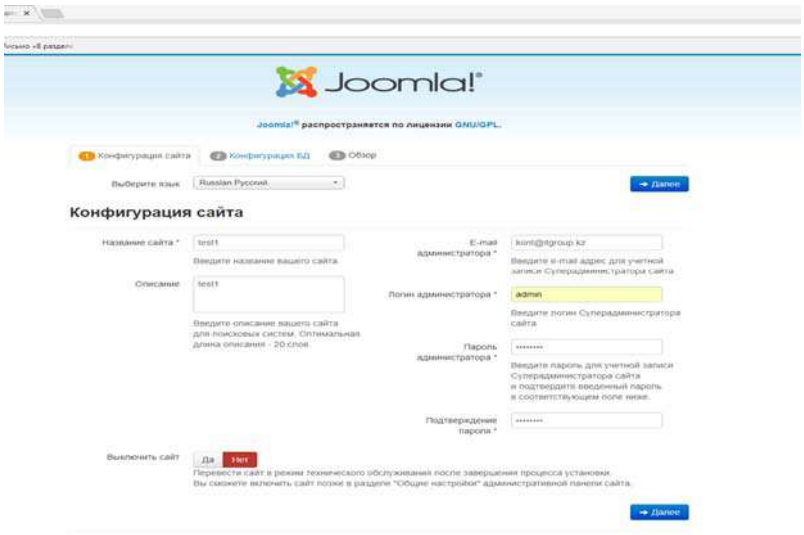


Рисунок 3.13 Заполнение полей в системе управления сайтом

12. После завершения конфигурации сайта нажать кнопку «Далее». Откроется конфигурация БД. Здесь указать параметры созданной ранее базы данных. Рисунок 3.14-3.15

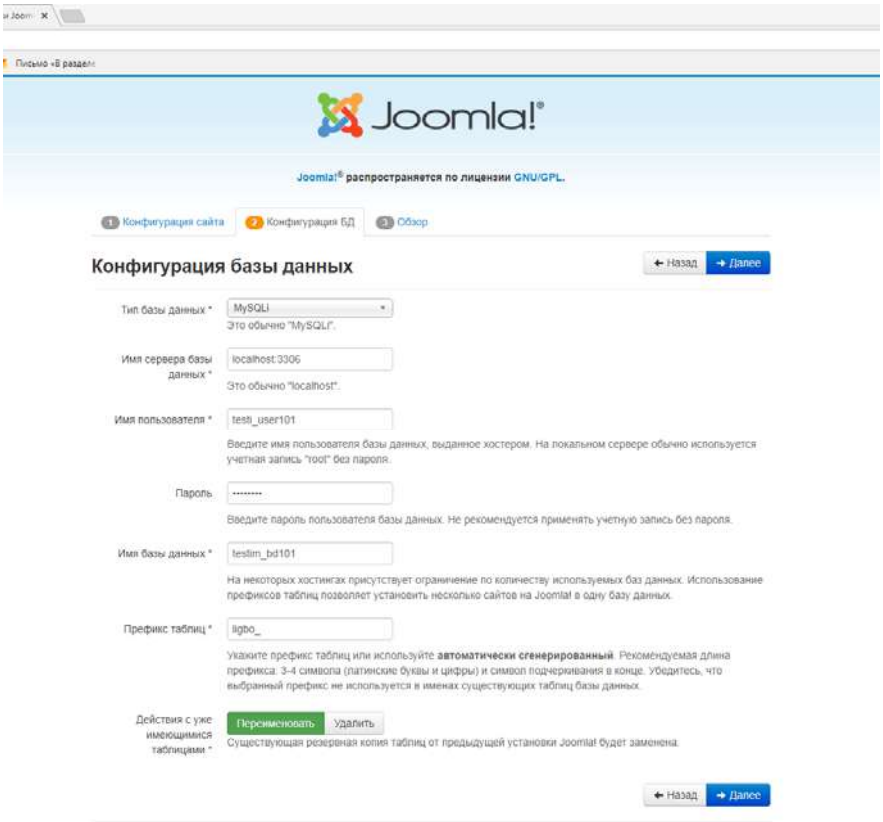


Рисунок 3.14 Конфигурация базы данных

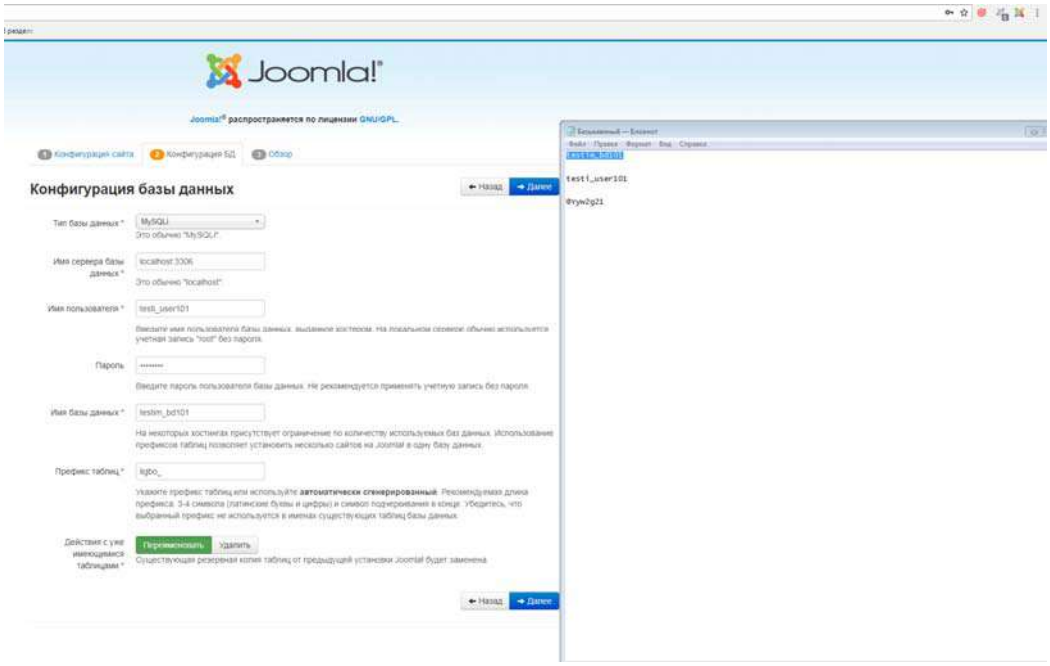


Рисунок 3.15 Конфигурация базы данных

13. При верном заполнении полей нажать кнопку «Далее»: откроется следующий этап «ОБЗОР». Рисунок 3.16

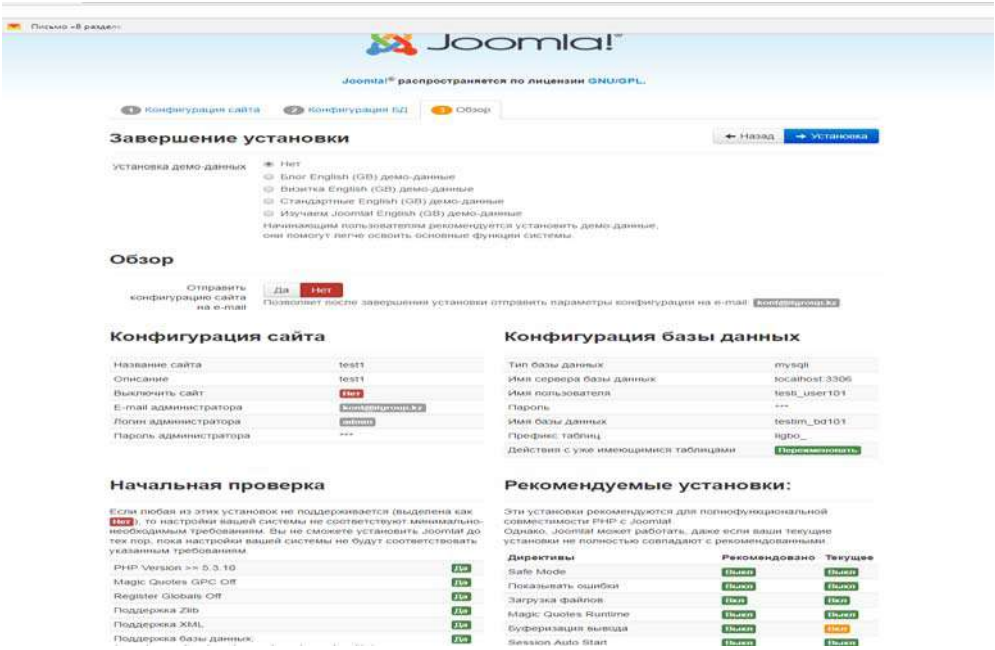


Рисунок 3.16 Обзор

Альтернативный способ установки готового сайта на хостинге

Повторить пункты с 10 по 12 предыдущего этапа.

1. Найти свою базу данных и импортировать **Backup** базы данных. Рисунок 3.17

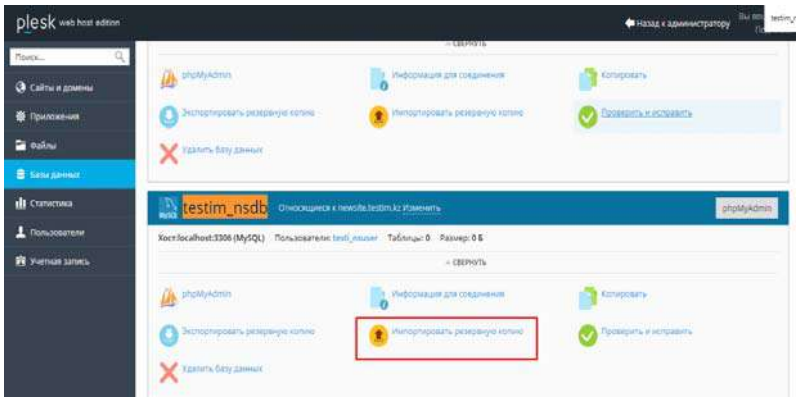


Рисунок 3.17 Альтернативный способ установки

2. Выбрать **Backup** базы данных на компьютере и нажать на кнопку «ОК» Рисунок 3.18

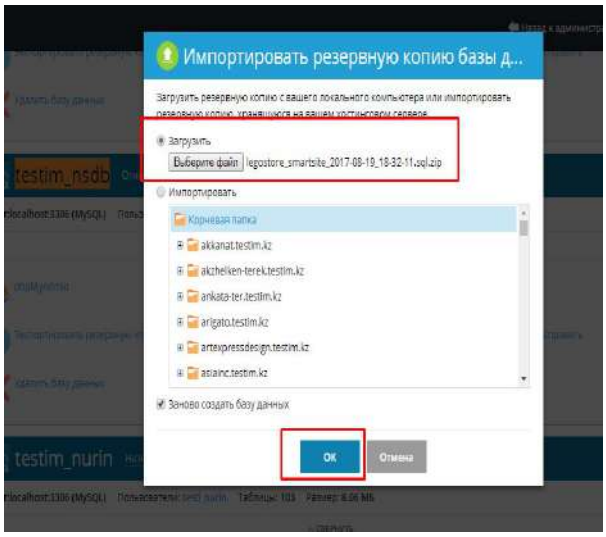


Рисунок 3.18 Backup базы данных

3. Должно появиться окошко с таким содержанием: Рисунок 3.19

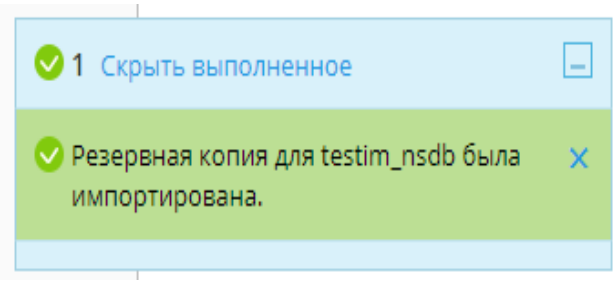


Рисунок 3.19 Окно подтверждения импорта базы данных

4. Вернуться в первую вкладку, где извлекался архив с файлами.
5. Найти файл configuration.php и указать права для записи. Рисунок 3.20

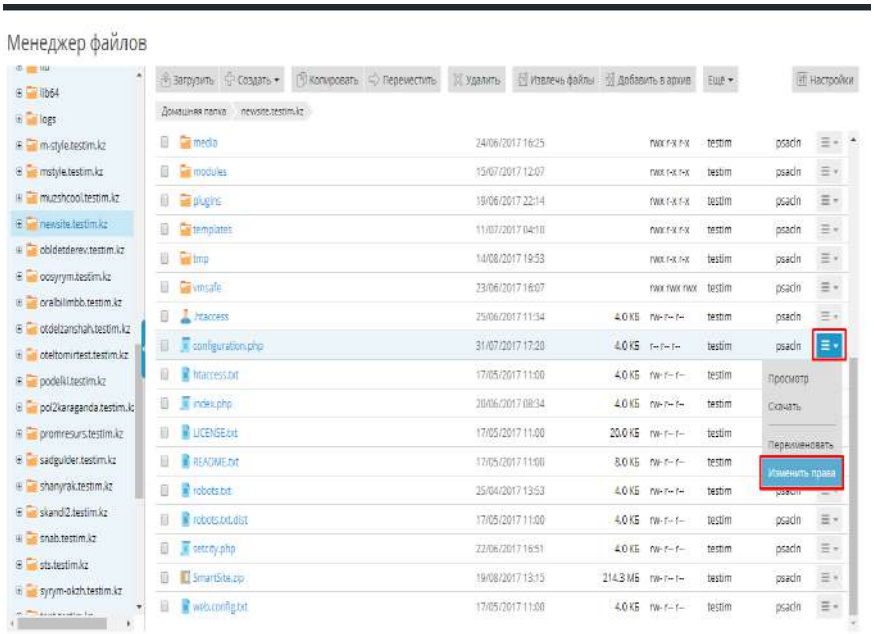


Рисунок 3.20 Указание прав для записи в файле configuration.php

6. В появившимся окне поставить необходимые галочки, после чего нажать на кнопку «ОК» Рисунок 3.21

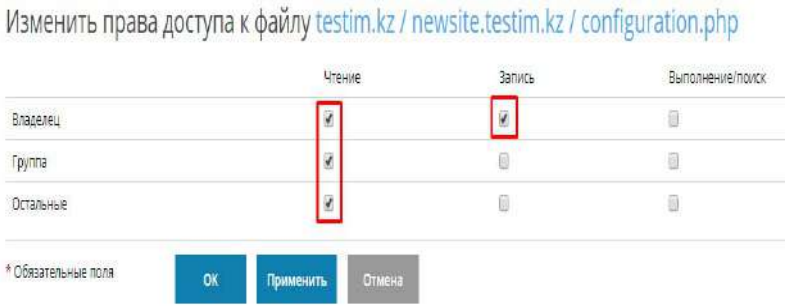


Рисунок 3.21 Изменение права доступа к файлу configuration.php

7. Далее перейти к редактированию файла configuration.php Рисунок 3.22

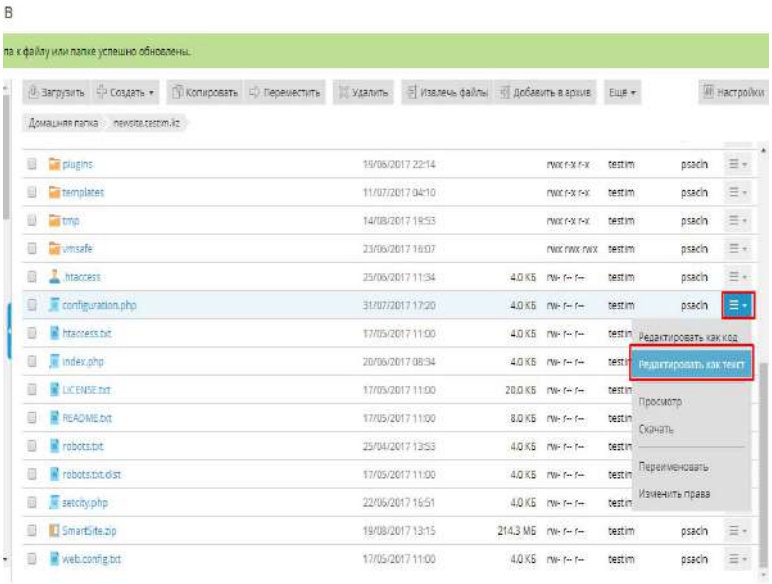


Рисунок 3.22 Переход к редактированию файла configuration.php

8. В появившемся окне найти необходимые строки и изменить их в соответствии с параметрами базы данных, которая создана ранее. Рисунок 3.23

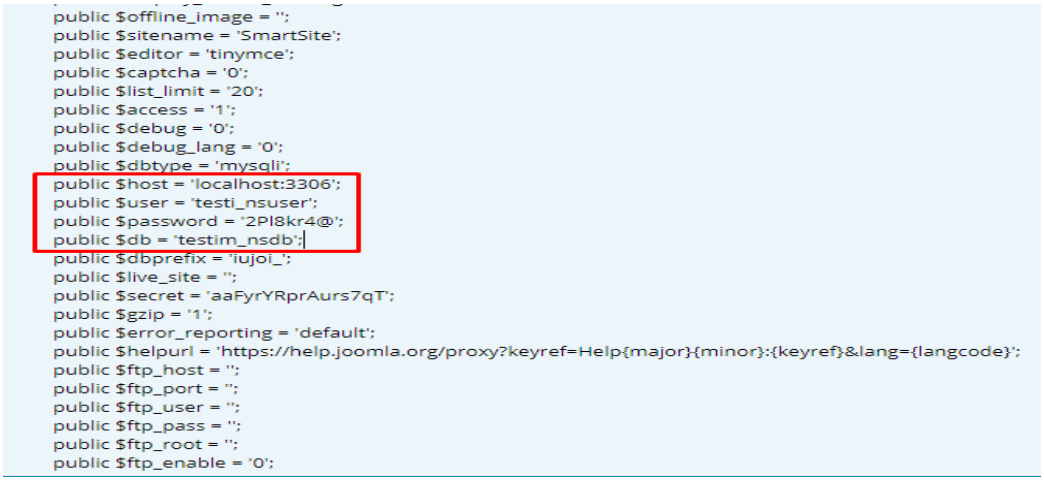


Рисунок 3.23 Необходимые строки для изменения в файле configuration.php

public $host = 'localhost:3306'; - это адрес сервера где расположена база данных,

public $user = 'testi_nsuser'; - это название юзера для подключения к базе данных,

public $password = '2Pl8kr4@'; - это пароль для подключения к базе данных,

public $db = 'testim_nsdb'; - это название самой базы данных.

9. Необходимо прописать пути для логирования и загрузки файлов. Для этого нужно пролистнуть страничку и найти выделенные строчки: Рисунок 3.24

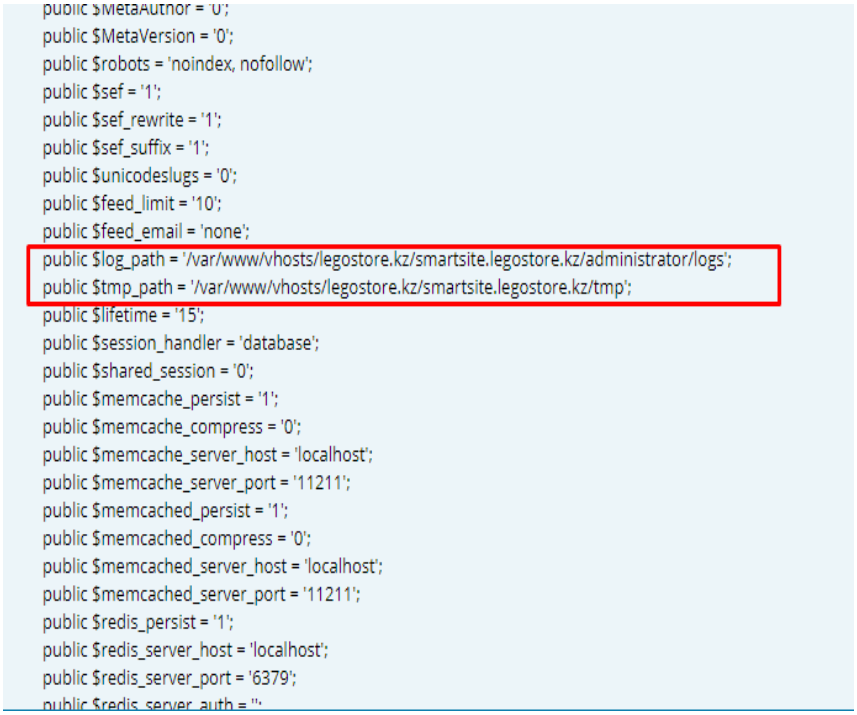


Рисунок 3.24 Строки в файле configuration.php для указания пути для логирования и загрузки файлов

10. Изменить строки на свои пути.

11. Далее нажать «ОК».

На этом установка на сервере завершена, можно переходить к наполнению сайта контентом.

3.3 Основы безопасности web-сайтов

Web-сайты, а также все сети, к которым подключены web-серверы, подвержены угрозам безопасности. Web-серверы по дизайну открывают окно между пользовательской сетью и миром. Забота об обслуживании сервера, обновлениях web-приложений и кодировании web-сайта будет определять размер этого окна, ограничивать информацию, которая может пройти через него, и тем самым устанавливать степень безопасности в Интернете. Цель безопасности web-сайта - предотвратить любые виды атак. Более формальное определение безопасности web-сайта - **это действие/практика защиты web-сайтов от несанкционированного доступа, использования, модификации, уничтожения или сбоев.**

«Web-безопасность» имеет две составляющие: внутренний компонент и общедоступный компонент. Относительная безопасность сайта высока, если сеть настроена со строгими разрешениями, web-сервер обновлен до последней версии со всеми выполненными настройками, все приложения на web-сервере исправлены и обновлены, а код web-сайта сделан с соблюдением высоких стандартов. web-безопасность относительно ниже, если серверы, приложения и код сайта являются сложными или старыми. Хорошо известно, что некачественно написанное программное обеспечение формирует проблемы безопасности. Количество ошибок, которые могут создавать проблемы с web-безопасностью, прямо пропорционально размеру и сложности web-приложений и web-сервера. Фактически все сложные программы имеют либо ошибки, либо слабые стороны. Кроме того, web-серверы являются по своей сути сложными программами. web-сайты сами по себе сложны и преднамеренно приглашают к более активному взаимодействию с общественностью. По этой причине существует большая вероятность взлома сайтов.

Самый безопасный web-сервер в мире - это тот, который отключен. Простые web-серверы, имеющие несколько открытых портов и сервисов на этих портах, являются лучшими. Для запуска сложных сайтов требуются мощные и гибкие приложения, которые, естественно, более подвержены проблемам web-безопасности.

Большинство нарушений безопасности web-сайта - это не кража данных или беспорядок с макетом web-сайта, а попытка использовать сервер в качестве ретранслятора электронной почты для спама или для создания временного web-сервера, как правило, для работы с файлами незаконного характера. Другие очень распространенные способы злоупотребления компрометируемыми машинами включают использование серверов в качестве части ботнета или для биткойнов. Взлом регулярно выполняется

автоматическими сценариями в попытке использовать известные проблемы безопасности web-сайта в программном обеспечении.

Следующие рекомендации помогут сайту безопасно работать в Интернете:

1. Обновление программного обеспечения до актуального состояния. Обеспечение постоянного обновления программного обеспечения имеет крайне важное значение для безопасности сайта. Это относится как к серверной операционной системе, так и к любому программному обеспечению, которое может быть запущено на web-сайте, например, в CMS или на форуме. Когда в программном обеспечении обнаружены слабые места в области безопасности web-сайта, хакеры незамедлительно пытаются злоупотребить ими.

Используя управляемый хостинг-решение, не нужно беспокоиться о применении обновлений безопасности для операционной системы, поскольку хостинговая компания должна позаботиться об этом.

Используя стороннее программное обеспечение на своем web-сайте, такое как CMS или форум, необходимо убедиться, что любые исправления безопасности быстро применяются. У большинства поставщиков есть список рассылки или RSS-канал, в котором подробно описаны проблемы безопасности web-сайта.

2. Защита от атак XSS. XSS - это термин, используемый для описания класса атак, которые позволяют злоумышленнику внедрять клиентские сценарии через web-сайт в браузеры других пользователей. Атаки с использованием межсайтового скриптинга (XSS) вводят вредоносный код JavaScript на web-страницы, который затем запускается в браузерах пользователей, а также может изменять содержимое страницы или украсть информацию для отправки злоумышленнику.

Лучшая защита от уязвимостей XSS - это удаление или отключение любой разметки, которая может содержать инструкции для запуска кода. Для HTML это включает в себя такие элементы, как <script>, <object>, <embed>, и <link>. Многие web-фреймворки автоматически деактивируют ввод данных из форм HTML по умолчанию.

3. Сообщения об ошибках. Необходимо быть осторожными при выдаче информации в сообщениях об ошибках. Предоставлять следует только минимальные ошибки для пользователей web-сайта, чтобы убедиться, что на сервере нет утечки секретов. Не рекомендуется предоставлять подробные сведения об исключении, так как они могут сделать сложные атаки, такие как SQL-инъекция, намного проще. Хранить подробные ошибки нужно в журналах сервера и показывать пользователям только необходимую им информацию.

4. Проверка с обеих сторон. Проверка всегда должна выполняться как на стороне клиента, так и на стороне сервера. При проверке на стороне сервера информация отправляется на сервер и проверяется на одном из его языков. Если проверка не пройдена, ответ отправляется клиенту, страница с web-формой обновляется и отображается обратная связь. Проверка на

стороне сервера достаточна для успешной и безопасной проверки формы. Однако для лучшего взаимодействия с пользователем необходимо рассмотреть возможность использования проверки на стороне клиента. Этот тип проверки выполняется с использованием языков сценариев, таких как JavaScript. При этом пользовательский ввод может быть проверен по мере ввода. Существует несколько различных типов проверки: обязательные поля, правильный формат и поля подтверждения.

5. Проверка паролей. Важно использовать надежные пароли. Хакеры часто применяют сложное программное обеспечение для взлома паролей. Для защиты от взлома пароли должны быть сложными, включать в себя заглавные и строчные буквы, цифры и специальные символы. Несмотря на то, что пользователям это может не нравиться, обеспечение соблюдения требований пароля поможет защитить их информацию в долгосрочной перспективе.

Пароли всегда должны храниться в виде зашифрованных значений, предпочтительно с использованием одностороннего алгоритма хэширования, такого как SHA. Использование этого метода означает, что при аутентификации пользователей всегда сравниваются зашифрованные значения. Для дополнительной безопасности web-сайта рекомендуется «солить» пароли, используя новую «соль» за пароль. (**«Соль»** (также модификатор) — строка данных, которая передает хеш-функции вместе с паролем).

Пример создания хеша с солью на PHP:

```
$password = 'password'; // Сам пароль
$hash1 = md5($password); // Хешируем первоначальный пароль
$salt = 'sflprt49fhi2'; // Соль
$saltedHash = md5($hash1 . $salt); // Складываем старый хеш с солью и пропускаем через функцию md5()
```

При использовании «соленых» паролей процесс их взлома в большом количестве становится медленнее, так как каждое предположение должно быть хэшировано отдельно для каждой модификации + пароль.

6. Загрузка файлов. Разрешение пользователям загружать файлы на сайт может быть большой угрозой безопасности, даже если это обычная замена аватара. Риск состоит в том, что любой загруженный файл может содержать скрипт, который при запуске на сервере полностью открывает сайт.

Если на сайте есть форма для загрузки файлов, то необходимо обрабатывать все файлы с большим подозрением. Разрешая пользователям загружать изображения, нельзя полагаться на расширение файла или тип, чтобы убедиться, что файл является изображением, поскольку их можно легко подделать. Даже открытие файла и чтение заголовка или использование функций для проверки размера изображения не являются надежными.

Любые файлы, загруженные на сайт, хранятся в папке вне web-корня или в базе данных в виде капли. Если файлы не доступны напрямую, то нужно создать сценарий для извлечения файлов из частной папки и доставить

их в браузер. Теги изображений поддерживают атрибут src, который не является прямым URL-адресом для изображения. Поэтому атрибут src может указывать на скрипт доставки файлов, позволяющий установить правильный тип содержимого в заголовке HTTP. Например:

```
<img src="/imageDelivery.php?id=1234" />
 <?php
 // imageDelivery.php
 // Fetch image filename from database based on $_GET["id"]
 ...
 // Deliver image to browser
 Header('Content-Type: image/gif');
 readfile('images/'.$fileName); ?>
```

7. Настройка брандмауэра. Большинство хостинг-провайдеров имеют дело с конфигурацией сервера для сайта. Если размещать свой web-сайт на собственном сервере, есть несколько пунктов, которые нужно проверить:

– Убедиться, что установлена настройка брандмауэра, и блокируются все несущественные порты. Если возможно, создать демилитаризованную зону (DMZ), обеспечивающую доступ только к портам 80 и 443.

– Разрешая загрузку файлов из Интернета, использовать только безопасные методы транспорта на сервер, такие как SFTP или SSH.

– Ограничить физический доступ к серверу.

8. Использование HTTPS. HTTPS (HyperText Transfer Protocol Secure) - это протокол web-передачи (HTTP), который добавляет уровень безопасности для данных, проходящих через безопасный уровень сокета (SSL) или соединение протокола безопасности транспортного уровня (TLS). HTTPS обеспечивает зашифрованную связь и защищенное соединение между удаленным пользователем и основным web-сервером. HTTPS предназначен, в первую очередь, для обеспечения повышенного уровня безопасности по протоколу HTTP для конфиденциальных данных и транзакций, таких как данные фактуры, транзакции с кредитными картами, пользовательский логин и т.д. HTTPS шифрует каждый пакет данных с переходом с использованием технологии шифрования SSL или TLS.

HTTPS настроен и поддерживается по умолчанию в большинстве web-браузеров и автоматически инициирует безопасное соединение, если его запрашивают web-серверы. HTTPS работает в сотрудничестве с органами сертификации, которые оценивают сертификат безопасности на доступном web-сайте.

Форма входа часто задает файл cookie. Например, файл cookie отправляется с каждым другим запросом на сайт, который делает вход в систему и используется для аутентификации этих запросов. Взломщик, похищающий эти данные, сможет идеально подражать пользователю и выполнить свой сеанс входа в систему. Чтобы избежать подобных атак, необходимо всегда использовать HTTPS для своего сайта.

9. Инструменты безопасности сайта. Самый эффективный способ проверить безопасность сайта - использовать некоторые инструменты, которые часто называют проверкой на проникновение или ручным тестированием.

Есть много коммерческих и бесплатных продуктов, которые помогут проверить сайт на безопасность. Примеры бесплатных инструментов, на которые стоит обратить внимание:

– Netsparker (доступно бесплатное издание сообщества и пробная версия). Хорошо подходит для тестирования SQL-инъекций и XSS.

– OpenVAS утверждает, что является самым продвинутым сканером безопасности с открытым исходным кодом. Хорошо проверяет известные уязвимости, в настоящее время просматривает более 25 000.

– SecurityHeaders.io (бесплатная онлайн-проверка). Инструмент для быстрого уведомления о том, какие заголовки безопасности включены и правильно настроены для домена.

– Xenotix XSS Exploit Framework Инструмент из OWASP (Open Web Application Security Project), который включает в себя огромный выбор примеров атаки XSS, которые можно запустить, чтобы быстро проверить, являются ли входы сайта уязвимыми в Chrome, Firefox и IE.

Контрольные вопросы:

1. Дайте определение безопасности web-сайта.
2. Назовите общие меры безопасности в сети Интернет и для web-сайтов.
3. Какие угрозы для безопасности web-сайта могут возникнуть, если не обновлять программное обеспечение?

Заключение

Учебное пособие «Создание web-страниц, сайтов с применением web-технологий» можно использовать для организации и целенаправленного управления деятельностью обучаемого по изучению курса «Web-программирование и интернет-технологии», стимулирования деятельности в рамках отдельного занятия, рационального сочетания различные видов учебной деятельности с учетом дидактических особенностей.

Цель учебного пособия − комплексное рассмотрение современных клиентских и серверных технологий web-разработки, применяемых для создания web-сайтов, и практическое изучение данных технологий на конкретных примерах с использованием языков программирования HTML/XHTML, JavaScript, Bootstrap верстки, современных систем управления сайтом WordPress, Typo3, Joomla!. Пособие содержит теоретические сведения по представлению и передаче информации в сети Интернет, основным технологиям разработки web-страниц, а также практические задания по использованию возможностей web-приложения для управления базами данных на сервере Php MyAdmin.

Данное учебное пособие содержит задания для самостоятельного выполнения с целью более глубокого понимания материала студентами. В результате освоения предлагаемого курса обучающиеся познакомятся с функциями сетевых служб и протоколов сети Интернет, изучат основы MySQL, смогут разрабатывать высококачественные web-сайты и приложения, познакомятся с методами программирования языков HTML/XHTML, JavaScript и создания зрелищных web-сайтов с использованием Bootstrap верстки, смогут создавать сайты с помощью систем управления WordPress, Typo3, Joomla!.

Глоссарий

AppleTalk - стек протоколов, разработанных Apple Computer для компьютерной сети.

Bootstrap – это система управления сайтом (фреймворк), которая содержит готовые CSS, HTML и JavaScript компоненты.

CSS (Cascading Style Sheets) «каскадные таблицы стилей» – формальный язык описания внешнего вида web-страниц.

Datagram Delivery Protocol - протокол доставки дейтаграмм, его основная задача заключается в доставке дейтаграмм сокета в порт через сеть AppleTalk.

DHCP (Dynamic Host Configuration Protocol) — сетевой протокол, позволяющий сетевым устройствам автоматически получать IP-адрес и другие параметры, необходимые для работы в сети.

DLC (Data Link Control)- высокоуровневый протокол канала передачи данных в иерархии информационной модели OSI, немаршрутизируемый протокол.

DNS (Domain Name System) - система, основным назначением которой является преобразование доменных имен устройств в IP-адреса, либо IP-адресов в доменные имена.

Firewall - технологический барьер, предназначенный для предотвращения несанкционированного или нежелательного сообщения между компьютерными сетями или хостами.

GIF (*GraphicsInterchangeFormat*) —растровый формат графических изображений.

HTML (HyperTextMarkupLanguage) - язык гипертекстовой разметки.

HTTPS (HyperText Transfer Protocol Secure) - вариант стандартного протокола web-передачи (HTTP), который добавляет уровень защиты данных, передаваемых через соединение с протоколом безопасного уровня сокетов (SSL) или протокола безопасности транспортного уровня (TLS).

IP (Internet Protocol — IP) - межсетевой протокол, который определяет адресацию сетевого уровня, а также задает способ, время и место перенаправления пакетов.

IPX/SPX (internetwork packet exchange/sequenced packet exchange) - протокол IPX работает на сетевом уровне модели OSI, обеспечивает доставку пакетов. Протокол SPX работает на транспортном и сеансовом уровнях, обеспечивает поддержание сеанса связи и гарантированную доставку данных.

JavaScript – объектно-ориентированный язык для создания интерактивных web- страниц и web-приложений.

JPEG (Joint Photographic Experts Group)- графический формат при оформлении web-страниц.

LAN (Local Area Network) - локальная компьютерная сеть, покрывающая обычно относительно небольшую территорию или небольшую группу зданий (дом, офис, фирму, институт).

MAC (Media Access Control) - физический адрес, присваиваемый каждой единице активного оборудования или некоторым их интерфейсам в компьютерных сетях Ethernet.

Metacomputer - устройство большой мощности.

MySQL – система управления базами данных с открытым исходным кодом для web-приложений.

Master/slave - модель взаимодействия в вычислительных комплексах, телекоммуникационных и информационных системах, в которой одно главное устройство (ведущее устройство) или процесс осуществляет однонаправленное управление подчиненным (ведомым) устройством или процессом, или их группой.

Network - компьютерная сеть.

NetBEUI (NetBIOS Extended User Interface) - расширенный пользовательский интерфейс дейтаграммной передачи NetBIOS.

OSI (Open Systems Interconnection) - семиуровневая модель стандартной сетевой архитектуры. Включает физический, канальный, сетевой, транспортный, сеансовый, представления данных, прикладной уровни.

PHP (Hypertext Preprocessor) - это язык программирования, специально разработанный для написания web-приложений (сценариев), исполняющихся на web-сервере.

PHPMyAdmin — web-приложения для управления базами данных СУБД MySQL на сервере, написанные на языке web-программирования PHP.

Peer-to-peer (P2P) - одноранговая, децентрализованная компьютерная сеть, основанная на равноправии участников.

PNG *(P*ortableNetworkGraphics) — растровый формат хранения графической информации.

RDAP (Registration Data Access Protocol) - протокол регистрации доступа к данным, позволяет пользователям получать доступ к текущей информации о регистрации, был создан в качестве возможной замены для WHOIS протокола.

RDP (Remote Desktop Protocol) - протокол удаленного рабочего стола, использующийся для обеспечения удаленной работы пользователя с сервером, на котором запущен сервис терминальных подключений.

RRAS (Routing and Remote Access Service) - служба маршрутизации и дистанционного доступа, позволяет удаленным клиентам проходить физические границы сетевого окружения, чтобы подсоединяться к сети и использовать ее ресурсы.

SQL-инъекция (SQL injection) - один из распространенных способов взлома сайтов и программ, работающих с базами данных, основанный на внедрении в запрос произвольного SQL-кода.

SSH (Secure Shell) - сетевой протокол прикладного уровня, позволяющий производить удаленное управление операционной системой и туннелирование TCP-соединений (например, для передачи файлов).

TCP (Transmission Control Protocol) - обеспечивает сквозную доставку данных между прикладными процессами, запущенными на узлах, взаимодействующих по сети.

TYPO3 — система управления сайтом (CMS/CMF*) с открытым исходным кодом и свободной лицензией, написана на языке PHP.

UDDI (Universal Description Discovery & Integration) - кроссплатформенное программное обеспечение, основанное на XML, который позволяет организациям публиковать описания web-сервисов (WSDL) для последующего их поиска другими организациями и интеграции в свои системы, а также определять, как сервисы или приложения взаимодействуют через Интернет.

Uptime - время непрерывной работы (доступности) системы, время от начала работы сервера (хостинга), где размещается сайт, и до его остановки, которая не зависит от ряда причин (например, перезагрузка, выключение, прекращение работы приложения и др).

URL-адрес - (Uniform Resource Locator — указатель размещения сайта в Интернете. URL-адрес содержит доменное имя и указание пути к странице, включая название файла этой страницы.

VoIP (Voice over Internet Protocol), IP-телефония — это технология для передачи голоса через Интернет по протоколу IP.

WHOIS (who is) — сетевой протокол прикладного уровня, базирующийся на протоколе TCP, основное применение — получение регистрационных данных о владельцах доменных имен, IP-адресов и автономных систем.

WAN (Wide Area Network) - глобальная компьютерная сеть, охватывающая большие территории и включающая большое число узлов.

Web-сервер — сервер, принимающий запросы от браузера и возвращающий ответ в виде HTML разметки.

Web- страница – текстовый документ, содержащий специальные команды.

WordPress – это многофункциональная платформа с открытым исходным кодом, ориентированная на создание новостных блогов и различных онлайн-публикаций.

XML (eXtensible Markup Language) - расширяемый язык разметки, разработанный для решения задач крупномасштабных электронных публикаций.

XSS (Cross-Site Scripting) - тип атаки на web-системы, заключающийся во внедрении в выдаваемую web-системой страницу вредоносного кода (который будет выполнен на компьютере пользователя при открытии им этой страницы) и взаимодействии этого кода с web-сервером злоумышленника.

Антивирус - программа, обнаруживающая и удаляющая вирусы, «троянских коней», «червей» и т.д.

Атрибуты — это специальные команды, которые расширяют действия тега.

База данных – это совокупность структурированных данных.

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

Гиперссылка (Hyperlink) – объект, с помощью которого при нажатии (по щелчку) правой кнопки мыши осуществляется переход к заданному месту.

Динамический сайт (активный) - это тот сайт, который может генерировать и возвращать содержание сайта на основе конкретного URL-адреса - запроса и данных.

Идентификатор – уникальное название элемента web-страницы, который присваивается с помощью атрибута id.

Клиентские программы – программы, написанные на языках, таких как JavaScript и VBScript, которые обрабатываются на клиентском компьютере.

Локальный сервер - специальная программа, позволяющая web-разработчикам разрабатывать сайт на локальном (домашнем) компьютере, без необходимости выхода в Интернет.

Переменная - именованная часть памяти, в которую могут помещаться разные значения переменной.

Прокси-сервер - промежуточный сервер (комплекс программ) в компьютерных сетях, выполняющий роль посредника между пользователем и целевым сервером (при этом о посредничестве могут как знать, так и не знать обе стороны), позволяющий клиентам как выполнять косвенные запросы (принимая и передавая их через прокси-сервер) к другим сетевым службам, так и получать ответы.

Селекторы (от англ. select - выбирать) – это элементы каскадной таблицы стилей CSS, которые указывают на тот элемент на web-странице, к которому должны будут применяться стили.

Серверные языки программирования – программы, работающие на сервере, взаимодействуя с базами данных и поддерживая связь между пользователем и сервером.

Сетевой протокол – это набор правил и методов взаимодействия объектов вычислительной сети, охватывающий основные процедуры, алгоритмы и форматы преобразования и передачи данных в сети.

Система управления контентом (содержимым) или CMS (англ. Content management system) - это набор программ, позволяющий добавлять, редактировать, удалять содержимое web-ресурсов и сайтов и управлять им.

Система управления контентом Joomla – это набор скриптов и модулей, которые дают возможность быстро создать сайт.

Событие — это реакция программы на действие пользователя (щелчок мышью по кнопке, уменьшение мышкой окна браузера, ввод текста с клавиатуры и т.д.).

Соль (также модификатор) — строка данных, которая передает хеш-функции вместе с паролем.

Статический сайт (пассивный) – это тот сайт, который возвращается с тем же кодированным содержанием с сервера всякий раз, когда

запрашивается конкретный ресурс. Тип переменной — определяет множество значений, которые могут быть ей присвоены и операции, которые могут быть с нею произведены.

Тэг или тег (от англ. tag) – специальное зарезервированное слово языка HTML, представляющее из себя текст, заключенный в угловые скобки <>.

Файл стилей (style.css) – это файл, в котором заданы значения параметров отображения информации, соответствующие выбранному шаблону.

Фреймворк – программный структурированный каркас, включающий программное обеспечение для создания web-сайтов и облегчающий их разработку.

Хостинг (hosting) - услуга по предоставлению вычислительных мощностей для физического размещения информации на сервере, постоянно находящемся в сети.

Языки web-программирования - это специализированные языки, которые предназначены для создания программ с использованием интернет-технологий и обработки текстовых массивов данных.

Список литературы и интернет-источников

1. David Sawyer McFarland, CSS: The Missing Manual, 2015
2. http://bourabai.ru/php/joomla.htm
3. http://bourabai.kz/php/
4. http://labs.org.ru/567-2/javascript/
5. http://ru.wikipedia.org/wiki/Typo3
6. http://stepkinblog.ru/bootstrap/sozdanie-prostogo-sajta-na-bootstrap-3-praktika-po-projdennomu-materialu.html/
7. http://topgorod.com/hi-tech/open-source/7364-typo3.html
8. http://uzeron.com/view_article.php?id=62
9. http://www.htmlandcssbook.com/press/
10. http://www.internet-technologies.ru/articles/osnovy-sozdaniya-baz-dannyh-mysql.html
11. http://www.webnav.ru/books/dreamweaver/server_programing_intro/
12. http://webcache.googleusercontent.com/search?q=cache:http://kamenskiy-koho911.narod.ru/Data/Raspred.doc
13. https://10kilogramm.ru/podrobnaya-instrukciya-po-ustanovke-joomla-na-lokalnyj-kompyuter.php
14. https://blogwork.ru/chto-takoe-bootstrap/
15. https://beonmax.com/ru/courses/bootstrap/getting-started/
16. https://cisco.com/c/ru_ru/products/security/firewalls/what-is-a-firewall.html
17. https://coba.tools/wordpress
18. https://goldbusinessnet.com/domen-i-xosting/kak-skachat-wordpress-ustanovit-na-xosting/
19. https://itchief.ru/lessons/bootstrap-3/website-creation-business-cards-(part-1)
20. https://ru.wikihow.com/
21. https://sites.google.com/site/webkss2kurs/verstka/adaptivnyj-veb-dizajn
22. https://searchnetworking.techtarget.com/definition/client-server
23. https://www.creativebloq.com/web-design/website-security-tips-protect-your-site-7122853
24. https://www.seostop.ru/sozdanie-saita/wordpress/shablon.html
25. https://www.youtube.com/watch?v=ZLh8gqHewrM
26. https://www.intuit.ru/studies/courses/42/42/lecture/27177
27. https://webformula.pro/article/sovremennye-cms/
28. https://webref.ru/layout/learn-html-css
29. Jon Duckett, HTML & CSS, 2011 by John Wiley & Sons, Inc., Indianapolis, Indiana ISBN: 978-1-118-00818-8
30. Shay Howe, HTML & CSS Develop & Style Websites, ISBN 13: 978-0-321-94052-0 ISBN 10: 978-0-321-94052-0
31. Andy Harris. HTML, XHTML, & CSS All-in-One For Dummies®, 2nd Edition, ISBN: 978-0-470-53755-8
32. Вьюшкова Е.А., Параскун Н.В. Информатика: Учеб.для 11 кл.- Изд. «Арман- ПВ», 2015

33. Борисенко А.А. Web-дизайн. Просто как дважды два. - М.: Эксмо, 2008.- 320 с.
34. Дженнифер Роббинс. HTML5, CSS3 и JavaScript. Исчерпывающее руководство; [пер. с англ. М. А. Райтман]. — 4-е издание. — М. : Эксмо, 2014. — 528 с
35. Морето Сильвио. «Bootstrap в примерах», ДМК-Пресс, 2017 г.-314 с.
36. Орлов Л. В. Web-сайт без секретов. / Л. В. Орлов. - 2-е изд. - М.: Бук-пресс, 2006. - 512 с.
37. Основы программирования в PHP (практические примеры). Алматинский государственный политехнический колледж, Алматы 2016-69 стр.
38. Пол Макфедрис. Создание Web-страниц. М.: АСТ Астрель, 2010 - 415, 456
39. Росс В. С. Создание сайтов: HTML, CSS, PHP, MySQL. Учебное пособие, ч. 1 — МГДД (Ю)Т, М.:2010 – 107 с.
40. Соболева М.Л, Алфимова А.С. Информационные технологии. Лабораторный практикум, М:Прометей, 2012 г-49 стр.
41. Создание Web-страниц и Web-сайтов. Самоучитель : [учеб. пособие] / под ред. В. Н. Печникова. - М.: Изд-во Триумф, 2006.- 464 с.
42. Фримен Эрик, Фримен Элизабет Изучаем HTML, XHTML и CSS, 2012 Питер
43. Цимбал, *А.А.* Технологии создания распределенных систем / А. А. Цимбал, М.Л. Аншина – СПб. : Питер, 2003.
44. Чекалов А. Прагматический подход к разработке приложений Web баз данных. – Электр. дан. – Режим доступа: Web-сервер Citforum http://www.citforum.ru/internet/webdbapp/ index.shtml
45. Э. Крамер. HTML - Наглядный курс Web-дизайна. М.-Спб.-Киев: Диалектика, 2011 - 153 с.
46. Якушев, Л. В. Начинаем работать в Интернет. Краткое руководство. - М.: Издательский дом «Вильяме», 2006. -128 с
47. www.techopedia.com/definition/25315/point-to-point-protocol-ppp
48. https://www.techopedia.com/definition/5361/hypertext-transport-protocol-secure-https
49. https://searchnetworking.techtarget.com/definition/Address-Resolution-Protocol-ARP
50. https://www.streetdirectory.com/travel_guide/2273/computers_and_the_internet/internet_security_basics_101.html
51. https://studopedia.ru/20_94820_organizatsiya-raspredelennoy-obrabotki.html.
52. http://sp.cmc.msu.ru/courses/sdpi/mdwrinet.pdf.
53. http://www.xnets.ru/plugins/content/content.php?content.103
54. https://pengstud.com/blog/kak-obezopasit-svoj-sajt/.
55. https://habr.com./sandbox/44286/

56. Проектирование информационных систем: методические указания к выполнению практического задания №9 для студентов специальности 071900 «Информационные системы и технологии» сост. Г.К. Конопелько, Д.Г. Конопелько – Хабаровск: Изд-во Хабар гос.техн. унт-та, 2005 – 27 с
57. http://perldoc.narod.ru/DBI-DBD-MySQL-spec1.pdf.
58. https://www.tutorialspoint.com/mysql/mysql-installation.htm
59. https://github.com/shtormnick/SQLbook/blob/master/2_Installation/Installation.md
60. https://1st-network.ru/prog/ramki-css
61. http://www.uzeron.com/view_article.php?id=62
62. http://dreamtutor.ru/index.php/page/get_lesson/5/3/1
63. http://labs.org.ru/javascript-2/
64. https://blogwork.ru/chto-takoe-bootstrap/.
65. https://www.bestfree.ru/article/webdesign/mysql.php.
66. http://cmsplugin.ru/page/typo3-korporativnaja-sistema-upravlenija-sajtom.

РАМАЗАНОВА Л.Е. АУБАКИРОВА К.М.
ТОЙШИБЕКОВА А.Б. ТОКАРЕВ А.Н.

СОЗДАНИЕ WEB-СТРАНИЦ, САЙТОВ С ПРИМЕНЕНИЕМ WEB-ТЕХНОЛОГИЙ

Подписано в печать 10.12.2018 г.
Формат 60*84 1/8
Печать цифровая
Усл. печ. л. 16,5. Тираж 32 экз.

Отпечатано компания «Профи Полиграф»