Перечень вопросов к экзамену

по дисциплине "Разработка web-приложений"

**1. Понятие Web-приложения. История web-разработки.**

Web-приложение - это программное обеспечение, которое доступно через браузер интернета и не требует установки на устройстве пользователя. Они работают на удаленных серверах и обеспечивают взаимодействие пользователя с сервером через интернет.

История web-разработки началась в начале 1990-х годов, когда был создан World Wide Web Тимом Бернерс-Ли. Первые веб-страницы были статическими и представляли собой простой текст и изображения. Затем были разработаны CGI (Common Gateway Interface) скрипты, позволяющие создавать динамические веб-страницы.

С появлением технологии AJAX (Asynchronous JavaScript and XML) в начале 2000-х годов, стало возможным создавать интерактивные и динамические web-приложения, которые могут обмениваться данными с сервером без перезагрузки всей страницы.

С развитием технологий HTML5, CSS3 и JavaScript, возможности web-разработки значительно расширились. Современные web-приложения могут включать в себя анимацию, видео, аудио, геолокацию, работу в реальном времени и другие функции, которые ранее были доступны только в десктопных приложениях.

В настоящее время web-приложения широко используются во многих сферах, таких как электронная коммерция, социальные сети, облачные сервисы, онлайн образование и другие. Они обеспечивают удобство доступа и использования для пользователей с любого устройства с подключением к интернету.

**2. Функционал, задачи и особенности создания Web-приложения.**

Web-приложения - это программное обеспечение, которое доступно через интернет и работает веб-браузере пользователя. Они позволяют пользователям взаимодействовать с различными сервисами, ресурсами или другими пользователями через интернет.

Основные задачи создания Web-приложения:

* Построение удобного и интуитивно понятного интерфейса, который обеспечивает удобство использования приложения для пользователя.
* Разработка функционала, который позволяет решать конкретные задачи пользователей, удовлетворяя их потребности.
* Создание безопасной архитектуры приложения, защищающей данные пользователей и предотвращающей возможные угрозы.
* Поддержка масштабируемости приложения, чтобы оно могло успешно работать при увеличении количества пользователей и объемов данных.
* Обеспечение надежной работы приложения в условиях различных сетевых и окружающих условий.

Особенности создания Web-приложения:

* Использование технологий веб-разработки, таких как HTML, CSS, JavaScript, для создания интерфейса и реализации функционала приложения.
* Разработка серверной части приложения, которая обеспечивает взаимодействие с базой данных, управление данными и обработку запросов от клиентской части.
* Тестирование приложения на различных устройствах и браузерах для обеспечения его корректной работы.
* Оптимизация производительности приложения для быстрой загрузки и отзывчивости интерфейса.
* Поддержка и обновление приложения для добавления нового функционала, исправления ошибок и обеспечения безопасности данных пользователей

**3. Этапы разработки Web-приложения.**

1. Планирование: определение целей и требований к приложению, составление плана разработки, определение бюджета и сроков.

2. Дизайн: разработка дизайна пользовательского интерфейса, создание макетов и прототипов, определение структуры приложения.

3. Разработка: написание кода, создание функционала приложения, тестирование и отладка.

4. Тестирование: проведение тестов на соответствие требованиям, исправление ошибок и устранение недочетов.

5. Релиз: запуск приложения на сервере, проверка его работоспособности, предоставление доступа пользователям.

6. Сопровождение и обновление: поддержка приложения, решение проблемных ситуаций, выпуск обновлений и улучшений.

**4. Взаимосвязь web-приложений c сетями WWW и Интернет.**

Web-приложения являются программными приложениями, которые доступны через браузер с помощью интернета. Они используют сети WWW (World Wide Web) и Интернет для своей работы.

Web-приложения взаимодействуют с серверами, которые хранят данные и обрабатывают запросы от пользователей. Пользователи могут использовать web-приложения для доступа к различным сервисам и функционалу, таким как онлайн-банкинг, социальные сети, электронная почта, онлайн-магазины и многое другое.

Для работы web-приложений необходим доступ к сетям WWW и Интернету, так как все данные передаются через сеть и серверы. Без подключения к сети Интернет web-приложения не смогут функционировать и обработать запросы пользователей.

Таким образом, взаимосвязь web-приложений с сетями WWW и Интернет очень тесная и обеспечивает функционирование многих онлайн-сервисов и ресурсов.

**5. Технология "клиент-сервер". Понятие, структура технологии.**

Технология "клиент-сервер" - это архитектурная модель, которая описывает взаимодействие между клиентом (конечным пользователем или приложением) и сервером (удаленным компьютером или сервисом), которые обмениваются данными и ресурсами.

Структура технологии "клиент-сервер" состоит из двух основных компонентов:

1. Сервер - это центральный компьютер или программа, которая предоставляет определенные сервисы или ресурсы клиентам. Сервер обычно имеет более мощные вычислительные ресурсы и обеспечивает обработку запросов от клиентов, хранит и управляет данными.

2. Клиент - это компьютер или приложение, которое обращается к серверу для получения требуемых данных или сервисов. Клиент отправляет запросы на сервер, получает ответы и обрабатывает полученные данные.

Клиент и сервер соединены посредством сети, такой как интернет, локальная сеть или другое средство связи. Соединение между клиентом и сервером обеспечивает передачу данных и управление ресурсами.

**6. Протоколы передачи данных.**

Протоколы передачи данных - это набор правил и стандартов, определяющих способы обмена информацией между устройствами в компьютерных сетях. Эти протоколы определяют формат и структуру данных, методы кодирования, передачу и обработку ошибок, а также другие аспекты передачи данных.

Некоторые из наиболее распространенных протоколов передачи данных включают в себя:

1. Протокол передачи гипертекста HTTP - используется для передачи веб-страниц и других данных в интернете.

2. Протокол передачи почты SMTP - используется для отправки и получения электронной почты.

3. Протокол передачи файлов FTP - используется для передачи файлов между компьютерами.

4. Протокол передачи голоса VoIP - используется для передачи голосовой информации через интернет.

5. Протокол передачи данных TCP/IP - широко используемый протокол передачи данных в компьютерных сетях.

Эти и другие протоколы играют важную роль в обеспечении эффективной и надежной передачи данных в компьютерных сетях.

**7. Системы доменных имен (DNS). Понятие, примеры.**

Системы доменных имен (DNS) - это распределенная база данных, которая используется для связывания доменных имен с IP-адресами. Она позволяет пользователям использовать удобные для запоминания доменные имена для доступа к веб-сайтам, почтовым серверам и другим ресурсам в Интернете, вместо запоминания IP-адресов каждого отдельного ресурса.

Примеры доменных имен включают в себя:

1. google.com - доменное имя поисковой системы Google.

2. facebook.com - доменное имя популярной социальной сети Facebook.

3. youtube.com - доменное имя популярного видеохостинга YouTube.

4. wikipedia.org - доменное имя онлайн-энциклопедии Википедия.

С помощью системы DNS устройства в сети могут использовать эти доменные имена для поиска соответствующих IP-адресов и установления соединения с нужным ресурсом.

**8. Web-сеть. Основные понятия, адресация ресурсов.**

Web-сеть - это сеть веб-страниц и других ресурсов, соединенных между собой гиперссылками и доступных через Интернет. В основе этой сети лежит протокол HTTP, который обеспечивает передачу данных между клиентом и сервером.

Основные понятия, связанные с Web-сетью:

1. Веб-страница - это HTML-документ, который отображается в веб-браузере. Он содержит текст, изображения, видео и другие элементы, которые могут быть интерактивными.

2. URL (Uniform Resource Locator) - это адрес ресурса в Интернете. Он состоит из протокола (например, http://), доменного имени (например, www.example.com) и пути к ресурсу на сервере.

3. Веб-сервер - это компьютер, который хранит веб-ресурсы и отвечает на запросы клиентов, отправляя им запрашиваемую информацию.

4. Веб-браузер - это программа, которая позволяет просматривать веб-страницы и взаимодействовать с веб-ресурсами.

5. Гиперссылка - это элемент веб-страницы, который позволяет перейти к другой странице или ресурсу, кликнув на него.

Адресация ресурсов в Web-сети осуществляется с помощью URL. Когда пользователь вводит URL в адресную строку браузера или щелкает на гиперссылке, браузер отправляет запросе на сервер, и если ресурс существует, сервер отправляет его обратно на клиентскую сторону для отображения.

**9. Методы запроса. Понятие, назначение основных методов.**

Методы запроса - это специальные команды, которые клиент отправляет серверу для выполнения определенных действий над ресурсами. Они используются в протоколе HTTP и определяют, как сервер должен обрабатывать запросы от клиентов.

Основные методы запроса в HTTP:

1. GET - используется для получения данных с сервера. Этот метод не должен изменять состояние сервера и не должен иметь побочных эффектов.

2. POST - используется для отправки данных на сервер для обработки. Этот метод может изменить состояние сервера или создать новый ресурс.

3. PUT - используется для обновления уже существующего ресурса на сервере. Если ресурса не существует, то он будет создан.

4. DELETE - используется для удаления ресурса с сервера.

5. PATCH - используется для частичного обновления уже существующего ресурса.

6. OPTIONS - используется для запроса информации о возможностях сервера или параметрах ресурса.

7. HEAD - аналог метода GET, но не возвращает тело ответа, а только заголовки.

Каждый из этих методов имеет свои особенности и назначение, и используется в зависимости от конкретной задачи, которую необходимо выполнить на сервере.

**10.Протокол HTTP.**

HTTP (HyperText Transfer Protocol) - это протокол передачи данных, который используется для обмена информацией в Интернете. Он определяет правила, по которым веб-сервер и веб-браузер взаимодействуют друг с другом, передавая запросы и ответы.

HTTP работает по модели «клиент-сервер», где клиент (например, веб-браузер) отправляет запрос на сервер, а сервер обрабатывает этот запрос и отправляет обратно ответ. Обычно запросы отправляются на порт 80 для HTTP или порт 443 для HTTPS.

Протокол HTTP состоит из методов запроса, статусных кодов ответа, заголовков запроса и ответа. Методы запроса могут быть, например, GET (получить данные), POST (отправить данные), PUT (обновить данные) и DELETE (удалить данные). Статусные коды ответа указывают на успешность выполнения запроса (например, 200 - успешно, 404 - страница не найдена).

HTTP поддерживает различные версии, текущей является HTTP/2. Он также может быть использован в сочетании с шифрованием SSL/TLS для обеспечения безопасной передачи данных (HTTPS).

**11.Структуры HTTP-сообщений.**

**12.Коды состояния. Категории кодов состояния.**

**13.Язык гипертекстовой разметки (HTML). Понятие, назначение языка,**

**синтаксис языка.**

**14.Структура HTML-документа.**

**15.Элементы управления HTML-формы.**

**16.Способы внедрения сценария JavaScript в документ HTML.**

**17.Каскадные таблицы стилей. Понятие, назначение.**

**18.CSS-правило. Структура.**

**19.Оформление HTML-документов с использованием каскадных таблиц**

**стилей. Способы записи стилей для элементов.**

**20.Понятие JavaScript. Приведите конкретные способы и среды создания**

**JavaScript-сценариев.**

**21.Структура и основные конструкции JavaScript кода.**

**22.Типы данных и переменные в JavaScript.**

**23.Массивы и объекты в JavaScript. Создание, доступ к элементам,**

**хранение информации.**

**24.Условные конструкции и циклы в JavaScript.**

**25.Общие подходы к дизайну web-приложения. Разработка макета**

**страницы.**

**26.HTML5: обзор возможностей, достоинства в сравнении с**

**предыдущими версиями.**

**27.Понятие SEO-оптимизации.**

**28.Объектно-ориентированное программирование. Основные принципы**

**ООП.**

**29.Системы управления контентом. Понятие, основные функции.**

**30.Основные типы CMS. Популярные системы управления контентом, их**

**особенности.**