1. **Дайте определение понятию «Интернет».**

**Интернет** — всемирная система объединённых компьютерных сетей для хранения, обработки и передачи информации. Часто упоминается как Всемирная сеть и Глобальная сеть, а также просто Сеть.

1. **Дайте определение понятию «Служба Интернет».**

Службы Интернета — это системы, предоставляющие услуги пользователям Интернета. К ним относятся: электронная почта, WWW, телеконференции, списки рассылки, FTP, IRC, а также другие продукты, использующие Интернет как среду передачи информации

1. **Дайте определение понятию «Узел сети Интернет».**

**Узел Интернет:** устройство, имеющее IP-адрес и подключенное сети Интернет (обычно к сети Интернет-провайдера). Каждый узел характеризуется своей программно-аппаратной платформой – аппаратурой и операционной системой.

1. **Дайте определение понятию «клиент-серверное приложение».**

Клиент-серверное приложение = приложение (программа) с клиент-серверной архитектурой: приложение, состоящее из двух компонент – клиента и **сервера**; клиент и сервер взаимодействуют между собой в соответствии с заданными правилами (спецификациями, **протоколами**); для взаимодействия между клиентом и сервером в соответствии с правилами (спецификацией, протоколом) должно быть установлено **соединение**; **инициатором соединения всегда является клиент**.

1. **Дайте определение понятию «сетевой протокол».**

**Сетевой протокол** - это набор правил, позволяющий осуществлять соединение и обмен данными между двумя и более включёнными в сеть компьютерами. Фактически разные протоколы зачастую описывают лишь разные стороны одного типа связи; взятые вместе, они образуют так называемый стек протоколов. Названия "протокол" и "стек протоколов" также указывают и на программное обеспечение, которым реализуется протокол.

1. **Перечислите основные свойства протокола HTTP.**

**HTTP:** основные свойства

* версии HTTP/1.1 – действующий (текстовый), HTTP/2 – черновой (не распространен, бинарный);
* два типа абонентов: клиент и сервер;
* два типа сообщений: request и response;
* от клиента к серверу – request;
* от сервера к клиенту – response;
* на один request всегда один response, иначе ошибка;
* одному response всегда один request, иначе ошибка;
* TCP-порты: 80, 443;
* для адресации используется URI или URN;
* поддерживается W3C, описан в нескольких RFC.

**HTTP** — это [протокол](https://developer.mozilla.org/ru/docs/%D0%A1%D0%BB%D0%BE%D0%B2%D0%B0%D1%80%D1%8C/%D0%BF%D1%80%D0%BE%D1%82%D0%BE%D0%BA%D0%BE%D0%BB), позволяющий получать различные ресурсы, например HTML-документы. Протокол HTTP лежит в основе обмена данными в Интернете. HTTP является протоколом клиент-серверного взаимодействия, что означает инициирование запросов к серверу самим получателем, обычно веб-браузером (web-browser). Полученный итоговый документ будет (может) состоять из различных поддокументов являющихся частью итогового документа: например, из отдельно полученного текста, описания структуры документа, изображений, видео-файлов, скриптов и многого другого.

1. **Перечислите состав информации, пересылаемой в HTTP-запросе.**

**Request**:

* метод;
* URI;
* версия протокола (HTTP/1.1);
* заголовки (пары: имя/заголовок);
* параметры (пары: имя/заголовок);
* расширение.

1. **Перечислите состав информации, пересылаемой в HTTP-ответе.**

**Response:**

* версия протокола (HTTP/1.1);
* код состояния (1xx, 2xx, 3xx, 4xx, 5xx);
* пояснение к коду состояния;
* заголовки (пары: имя/заголовок);
* расширение.

1. **Дайте определение понятию «web-приложение».**

Клиент-серверное приложение, у которого клиент и сервер взаимодействуют по протоколу HTTP называется **web-приложением**.

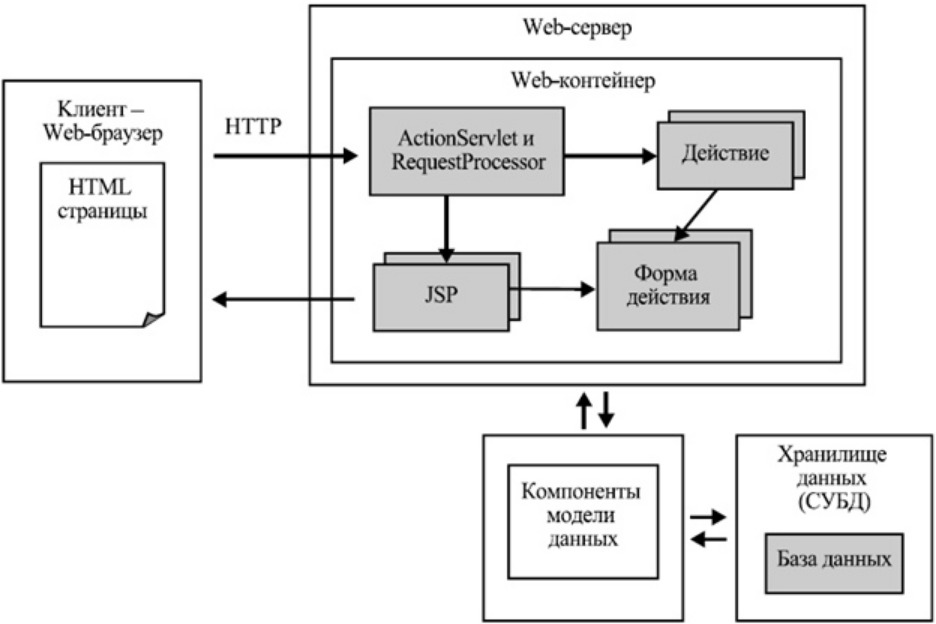
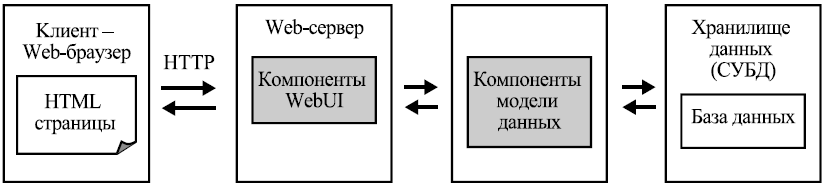
1. **Дайте определение понятиям «frontend» и «backend».**

Фронтенд (англ. Front-end) — клиентская сторона пользовательского интерфейса к программно-аппаратной части сервиса. Бекенд (англ. back-end) — программно-аппаратная часть сервиса.

1. **Дайте определение понятию «кроссплатформенное приложение».**

**Кроссплатформенное приложение**: приложение, способное работать на более чем одной программно-аппаратной (аппаратура + операционная система) платформе. Кроссплатформенность может быть достигнута различными способами: 1) на уровне компилятора (С, С++); 2) на уровне среды (фреймворка) исполнения (Java/JVM, C#/.NET CORE, JS/Node.js/V8).

1. **Изобразите и поясните общую схему web-приложения.**

1. **Назовите основные технологии разработки серверных кроссплатформенных приложений.**

технологии для разработки кроссплатформенных web-серверов:

- PHP/Apache, LAMP;

- Java/JVM/Application Server;

- C#/ASP.NET CORE;

- Python/Django;

- Ruby on Rails;

- JS/Node.js, ….

1. **Поясните понятие «асинхронная операция».**

**Понятие асинхронности**: операция называется асинхронной, если ее выполнение осуществляется в 2 фазы: 1) заявка на исполнение; 2) получение результата; при этом участвуют два механизма: A-механизм, формирующий заявку и потом получающий результат; B-механизм, получающий заявку от A, исполняющий операцию и отправляющий результат A; продолжительность исполнения операции B-механизмом, как правило, непредсказуемо; в то время пока B-механизм исполняет операцию, А-механизм выполняет собственную работу. Применение асинхронности не противоречит применению многопоточности.

Асинхронность позволяет вынести отдельные задачи из основного потока в специальные асинхронные методы или блоки кода.

1. **Поясните принцип выполнения асинхронного запроса с помощью объекта XMLHTTPRequest и Fetch.**



XMLHttpRequest – это встроенный в браузер объект, который даёт возможность делать HTTP-запросы к серверу без перезагрузки страницы.

fetch() позволяет вам делать запросы, схожие с XMLHttpRequest (XHR). Основное отличие заключается в том, что Fetch API использует [Promises (Обещания)](http://habrahabr.ru/post/209662/), которые позволяют использовать более простое и чистое API, избегать катастрофического количества callback'ов и необходимости помнить API для XMLHttpRequest.

Пример с XMLHttpRequest потребует от нас установить два обработчика событий на success и error, а так же вызвать два метода: open() и send(). мы проверяем статус ответа и проверяем, успешно ли выполнился запрос (ожидаем 200 статус). Если всё хорошо, то парсим ответ как JSON.

1. **Поясните основное назначение сервера NODE.JS.**

Чаще всего эту платформу используют для создания веб-сервисов, требующих интенсивного обмена информацией с пользователями, в том числе, для реализации чатов, систем совместной работы, социальных сетей и т.д. Многие программы, созданные на Node.js, состоят из серверной и клиентской частей.

приложение может обратиться с запросом к базе данных сервера, а пока ожидается ответ, обработать какие-то другие запросы. В результате одновременно обрабатываются тысячи соединений, каждое из которых тратит совсем немного времени и ресурсов. Это заметно ускоряет обмен информацией с сервером и, как итог, работу браузерного приложения.

1. **Перечислите основные свойства сервера NODE.JS.**

NODEJS: основные свойства:

* основан на Chrome V8;
* среда (контейнер) исполнения приложений на JavaScript;
* поддерживает механизм асинхронности;
* ориентирован на события;
* однопоточный (код приложения исполняется только в одном потоке, один стек вызовов); обычно в серверах для каждого соединения создается свой поток, в Node.js все соединения обрабатываются в одном JS-потоке;
* не блокирует выполнение кода при вводе/выводе (в файловой системе до 4х одновременно);
* в состав Node.js входят инструменты: npm – пакетный менеджер; gyp - Python-генератор проектов; gtest – Google фреймворк для тестирования С++ приложений;
* использует библиотеки: V8 – библиотека V8 Engine, libuv – библиотека для абстрагирования неблокирующих операций ввода/вывода; http-parser – легковесный парсер http-сообщений (написан на C и не выполняет никаких системных вызовов); c-ares - библиотека для работы с DNS; OpenSSL – библиотека для криптографии; zlib – сжатие и распаковка.