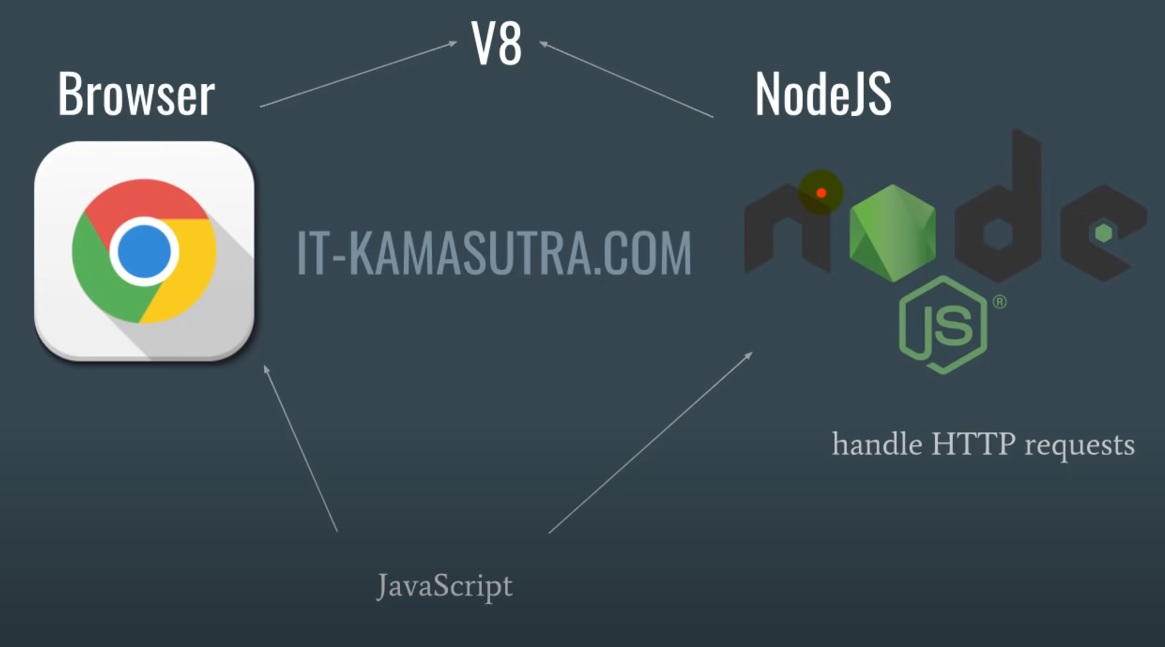
**01. ReactJS - Путь Самурая - Как смотреть данный курс?**

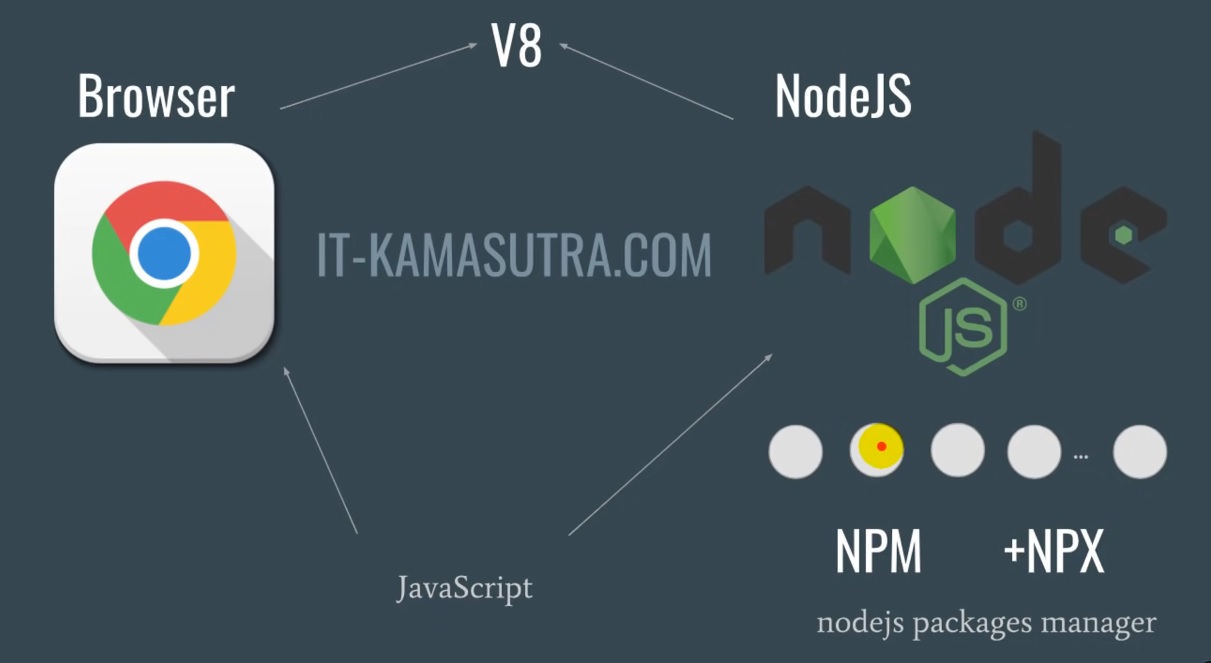
**02. Уроки React JS (Зачем NodeJS на фронте)**

Изначально JS разработан для того, чтобы выполняться в браузере. С развитием JS, он начал работать с базами данных, файловыми системами и т.д., для этого была создана альтернативная браузеру платформа NodeJS.



Браузер работает с html и модифицирует его при помощи JS. NodeJS работает с файлами, базам данных и т.д. при помощи JS. И браузер и NodeJS внутри используют движок V8, только каждая из этих платформ расширяет его по своему.

NodeJS называют серверным JS, потому, что одна из программ, которую можно написать под него сможет принимать http запросы от браузера, обрабатывать и возвращать ответы – програмный http сервер.



Современная разработка – это сложный процесс, поэтому Frontend разработчик использует NodeJS для получения и работы с модулями, компилирующими, минифицирующими, склеивающими JS, запускающими сервер и т.д. Модули разработаны под NodeJS и призваны упростить разработку. Node Package Manager(npm) – менеджер пакетов, устанавливается вместе с NodeJS, занимается, в основном, каталогизацией, установкой пакетов/модулей, поддержанием актуальных версий. npx позволяет не только устанавливать пакеты, но и запускать/исполнять.

**03. Уроки React JS (Установка NodeJS)**

NodeJS как и браузер является мультиплатформенным. Аналогично сайту, запущенному в браузере и работающему на любой ОС и платформе, программы NodeJS будут работать внутри NodeJS независимо от ОС и платформы.

После установки NodeJS запускаем не Node.js, а Node.js command prompt.

**04. Уроки React JS (create-react-app)**

Создаём новое React приложение согласно документации:

<https://github.com/facebook/create-react-app>

<https://create-react-app.dev/>

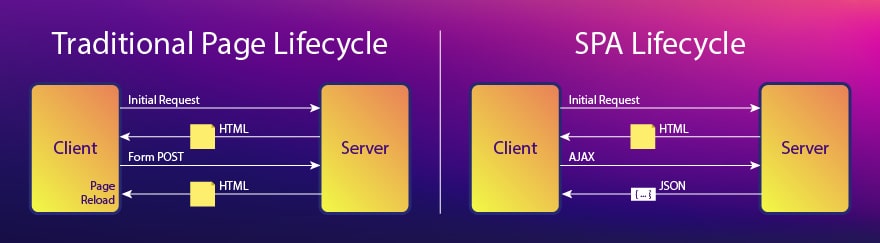
npx create-react-app my-app

cd my-app

npm start

Рекомендуется называть папку в стиле kebab-case, например 01-first-project. Проект/пакет/модуль, в том числе и React приложение характеризуется наличием файла package.json, в котором хранится информация об этом пакете.

**05. Уроки React JS (SPA: single page application)**

****

В случае традиционной страницы сервер генерирует множество html и возвращает его в браузер. Следом за html приходит малое количество JS. Браузер/клиент отображает этот html, а чтобы проверить наличие изменений на сервере необходимо обновить страницу, клиент посылает на сервер новый запрос, сервер возвращает html страницу целиком. Если изменений не произошло, новая html страница аналогична предыдущей. При таком подходе, для каждой страницы на многостраничном сайте существует отдельный адрес, после запроса на которой загружается отдельная html страница.

SPA – маленькая полупустая html страница, в которую помимо html приходит много JS генерирующего динамическую разметку. SPA страницы/приложения загружают страницу единожды. Html приходит мало, страница практически пустая, контент и готовая разметка не приходит, но приходит больше количество JS. JS посылает AJAX запрос на сервер, сервер возвращает JSON файл с данными, на основании которых JS генерирует html. Страница не перезагружается, кроме того JS периодически отправляет на сервер AJAX запросы проверяя наличие изменений. Если изменения есть, генерируется небольшой html соответствующий этим изменениям. Все операции происходят с такой скоростью, что пользователь этого не замечает. В SPA всегда одна страница (не landing page!), при переходе на другую страницу многостраничного сайта, URL адрес меняется, но клиент не отправляет на сервер запрос, чтобы получить новую html страницу, вместо этого JS перехватывает изменение URL и делает AJAX запрос на сервер, на основе полученного в ответ JSON файл, JS стирает ненужное содержимое предыдущей страницы и рендерит новый контент, страница не перезагружается и не перерисовывается целиком.

Плюсы SPA – экономия трафика, уменьшение нагрузки на сервер, увеличение скорости работы страницы. Серверу проще отдать 100 человекам пустую html страницу + JS, а затем легковесный JSON практически не требующий генерации, нежели взять данные, сгенерировать html страницу с этими данными и вернуть клиенту. Сервер выступает в роли API (программный интерфейс приложения, application programming interface), и становится универсальным, он единожды отдаёт html и далее работает с AJAX запросами. При разработке мобильных приложений, html не используется, но нужны JSON данные, для их получения достаточно сделать запрос на тот же URL, что и браузер, но далее данные обрабатываются и отрисовываются по-другому. Технически сервер пригоден для работы с мобильным приложением, это существенная экономия средств.

React, Angular и другие фреймворки работают по принципу SPA.