

Introduction to React

НАШИ ПРАВИЛА




Включенная камера




Вопросы по поднятой руке



Не перебиваем друг друга



Все вопросы, не связанные с тематикой курса (орг-вопросы и т. д.), должны быть направлены куратору



Подготовьте свое рабочее окружение для возможной демонстрации экрана (закройте лишние соцсети и прочие приложения)

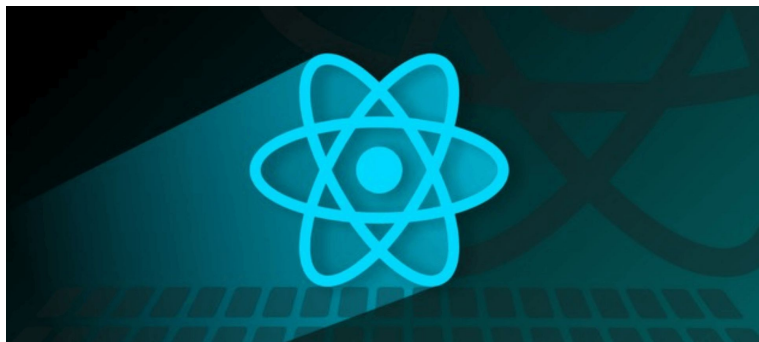
ЦЕЛЬ

Ознакомиться с основными понятиями react. Установить проект

ПЛАН ЗАНЯТИЯ

- 1. Создание проекта
- 2. Определение React

Создание React проекта



Создание репозитория. Работа с GIT

клонировать репозиторий с сервера на свой компьютер

```
git clone [url]
```

скачать с сервера последние изменения

```
git pull
```

Создать и ответить в новую ветку (делать такое от главной ветки, предварительно подтянув изменения с помощью git pull)

```
git checkout -b branch-name
```

загрузить свои обновления на удаленный репозиторий (сделать коммит)

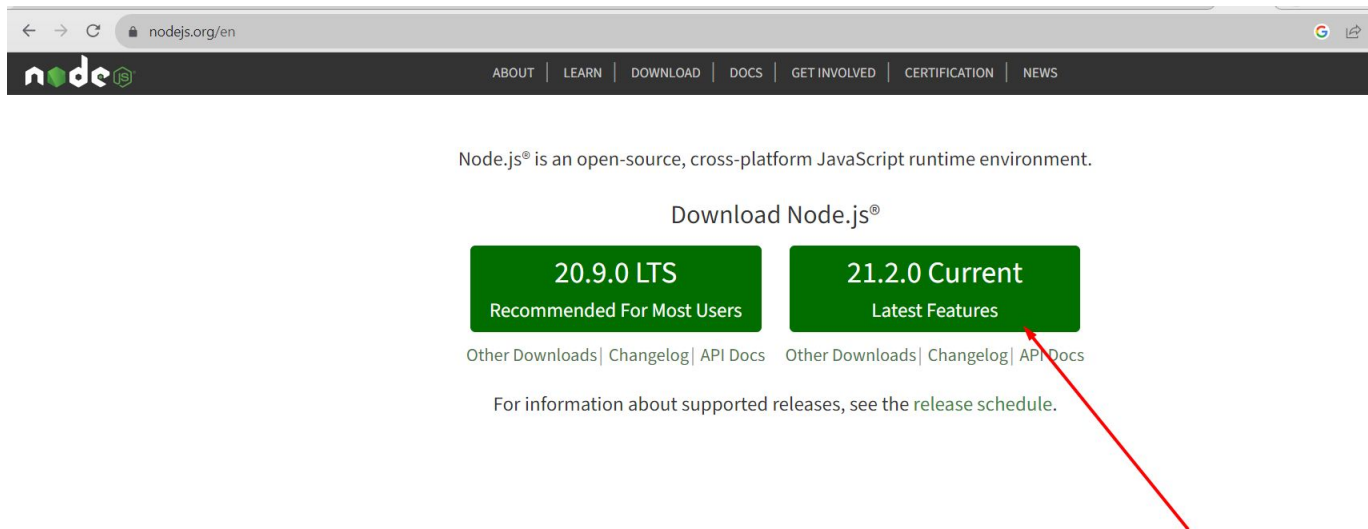
```
git add .
```

```
git commit -m "message"
```

```
git push
```

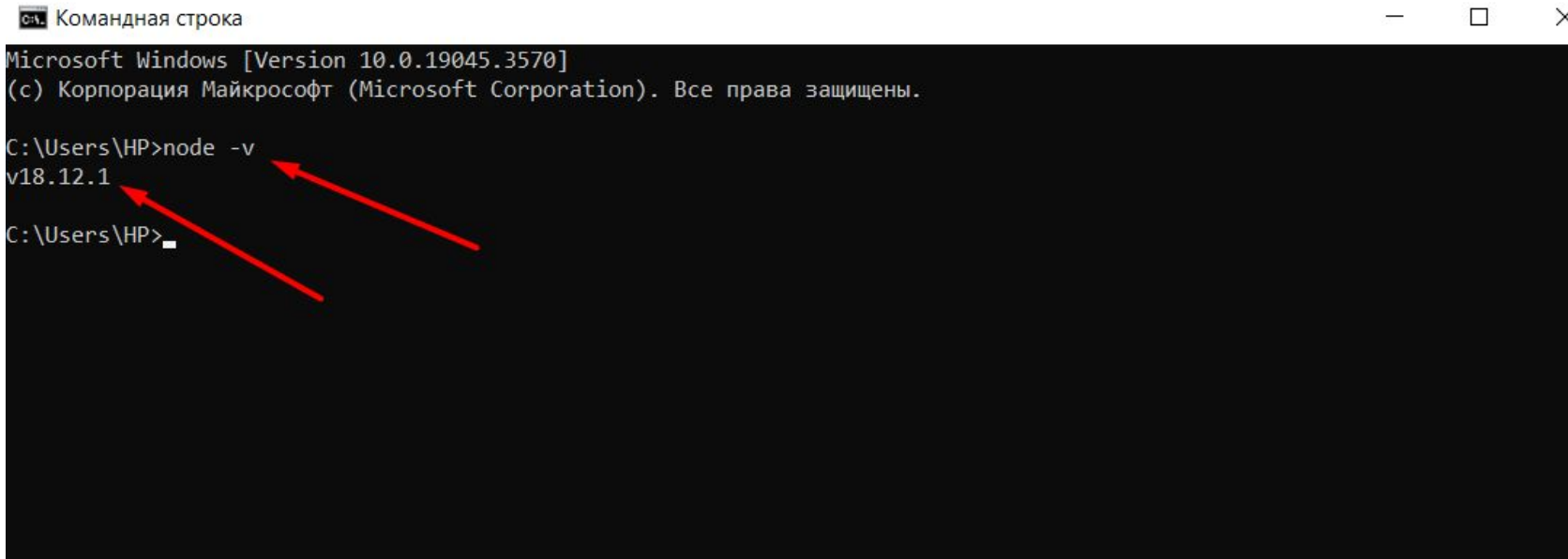
Установка NodeJS

1. Перейдите на сайт <https://nodejs.org/en>
2. Скачайте и установите последнюю версию Node.js



Проверка версии NodeJS

1. Перейдите в командную строку или терминал
2. Введите команду `node -v`



The screenshot shows a Windows Command Prompt window titled "Командная строка". The window displays the following text:

```
Microsoft Windows [Version 10.0.19045.3570]  
(c) Корпорация Майкрософт (Microsoft Corporation). Все права защищены.  
  
C:\Users\HP>node -v  
v18.12.1  
  
C:\Users\HP>
```

Two red arrows point to the command `node -v` and its output `v18.12.1`.

Создание проекта

1. Откройте VSCode, перейдите в папку, в которой будет лежать ваш проект и в терминале введите следующую команду - **`npx create-vite@latest . --force --template react`**
2. Выполните команду **`npm install`**
3. Запустите проект - **`npm run dev`**

Структура проекта

└─ node_modules/	# Папка с установленными npm-зависимостями (создается после `npm install`)
└─ public/	# Папка для статических файлов (например, изображений, шрифтов и т.д.)
└─ vite.svg	# Логотип Vite (пример статического ресурса)
└─ src/	# Исходный код приложения
└─ App.css	# Стили для компонента App
└─ App.jsx	# Главный React-компонент приложения
└─ index.css	# Глобальные стили для всего приложения
└─ main.jsx	# Точка входа в приложение (рендеринг React)
└─ assets/	# Папка для хранения ваших медиафайлов, таких как изображения (по умолчанию может быть пустой)
└─ .gitignore	# Файлы и папки, игнорируемые Git
└─ index.html	# Главный HTML-файл, который является базой для приложения
└─ package.json	# Файл конфигурации npm с информацией о проекте и зависимостях
└─ vite.config.js	# Конфигурационный файл для Vite
└─ README.md	# Файл с описанием проекта (по умолчанию может быть пустым)

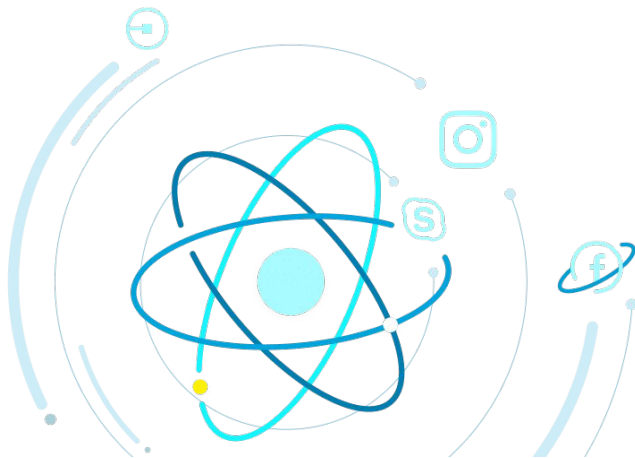
Менеджер пакетов - npm

npm (Node Package Manager) — это инструмент для управления пакетами (библиотеками и модулями) в экосистеме Node.js. Он позволяет устанавливать, обновлять, удалять и управлять зависимостями в вашем проекте.

Примечание: для запуска команд npm не забудьте перейти в директорию вашего проекта. Команда **cd my_app**

React — это JavaScript-библиотека для разработки пользовательского интерфейса.

React используют для создания одностраничных и многостраничных приложений, разработки крупных сайтов.



Основные преимущества React JS:

1. Компонентный подход:

React.js основан на концепции создания множества маленьких, переиспользуемых компонентов. Эти компоненты затем объединяются в более крупные приложения.

2. Виртуальный DOM:

Это одна из главных особенностей React.js. Это означает, что React не обновляет все элементы на странице, а только те, которые действительно изменились, что делает приложение меньше и быстрее.

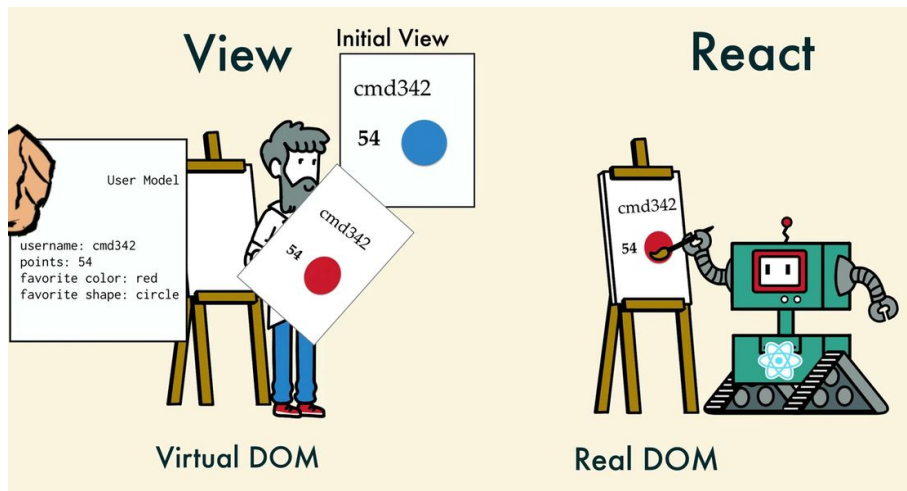
3. Однонаправленный поток данных:

Философия React.js заключается в том, что данные должны передаваться только в одном направлении: от родительского компонента к дочернему.

4. Использование синтаксиса JSX

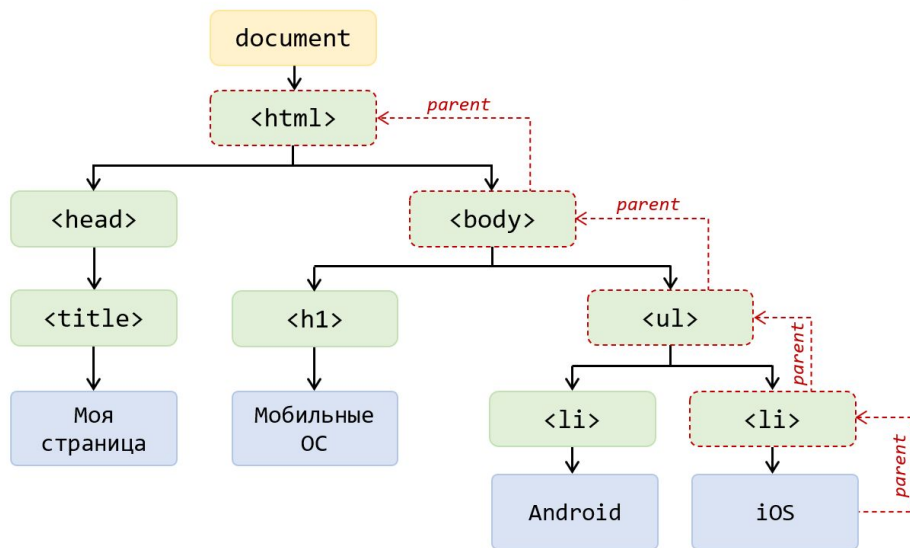


Virtual Dom

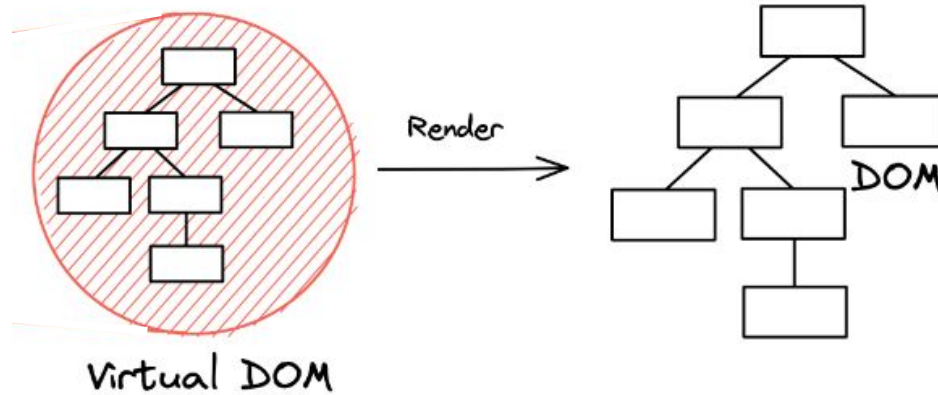


Что делает манипуляции с DOM медленными?

Когда происходят изменения в DOM, например, при добавлении, удалении или изменении элемента, происходят обновления DOM. Браузер перестраивает DOM-дерево и обновляет отображение веб-страницы в соответствии с этими изменениями.



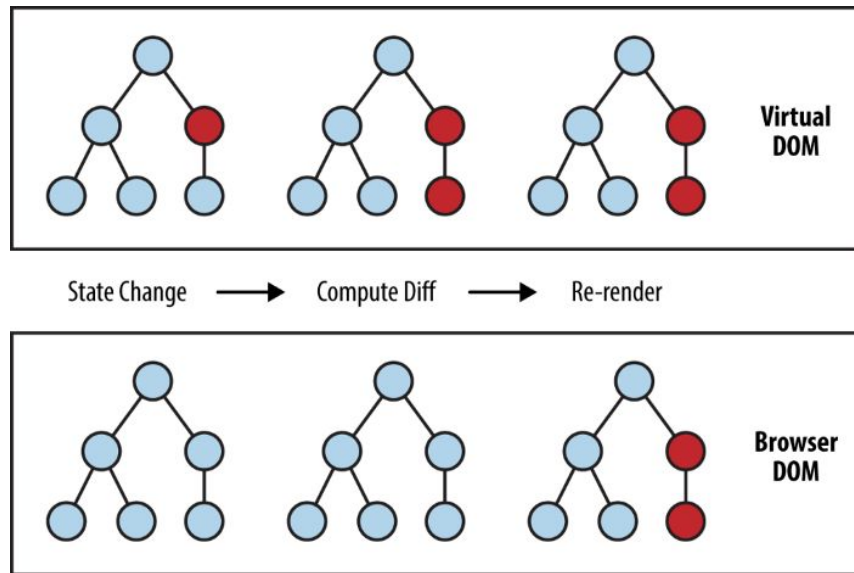
Virtual DOM (виртуальный DOM) - это концепция программирования, в которой «виртуальное» представление пользовательского интерфейса хранится в памяти и синхронизируется с «настоящим» DOM




Принцип работы React и VirtualDOM

Когда в UI добавляются новые элементы, создается Virtual DOM в виде дерева. Каждый элемент является узлом этого дерева. При изменении состояния любого элемента, создается новое дерево. Затем это новое дерево сравнивается (diffed) со старым.

После этого вычисляется наиболее эффективный метод внесения изменений в DOM. Цель данных вычислений состоит в минимизации количества операций, совершаемых с DOM. Тем самым, уменьшаются накладные расходы, связанные с обновлением DOM.





Ваша новая IT-профессия – Ваш новый уровень жизни

Программирование с нуля в
немецкой школе AIT TR GmbH