Урок 1
Начало работы с React

План урока

1. Реактивные фреймворки JavaScript.
2. Понятие SPA.
3. Основные черты React.
4. Устанавливаем и настраиваем React.

Реактивные фреймворки JavaScript

По историческим причинам все браузеры на всех платформах понимают лишь 3 технологии. На базовом уровне вы с ними уже знакомы:

- HTML — язык разметки, отвечает за структуру страницы;
- CSS — язык стилей, отвечает за стили страницы, ее оформление;
- JavaScript — язык программирования, отвечает за клиентское взаимодействие со страницей.

На текущий момент последнюю версию языка JavaScript описывает спецификация ECMAScript 2020. Несмотря на множество полезных нововведений, начиная с версии ES6, “чистый” JavaScript все еще считается не очень эффективным способом разработать большое и поддерживаемое приложение с грамотной архитектурой. Главная причина этого — необходимость писать много boilerplate[1] кода, а также необходимость проектирования, тонкой настройки, а затем и тестирования множества составляющих приложения:

- управление состоянием;
- роутинг;
- шаблонизация;
- разделение на компоненты.

[1] https://en.wikipedia.org/wiki/Boilerplate_code

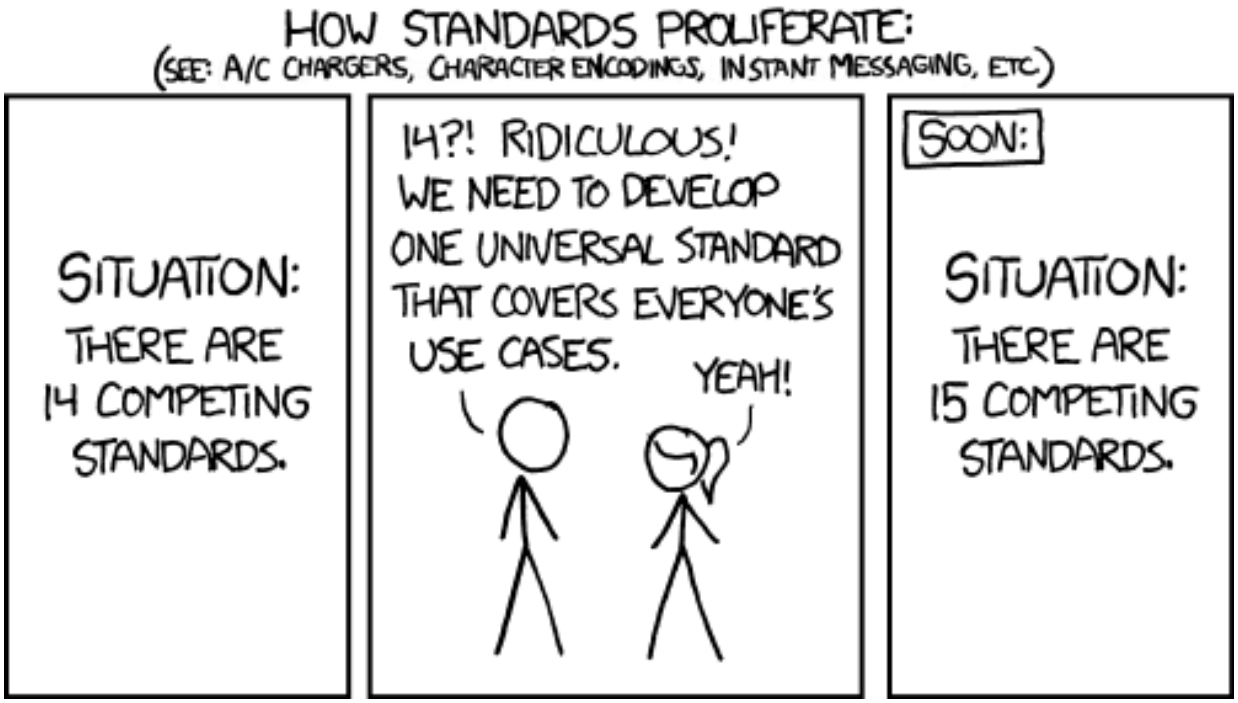
Использование современных библиотек или фреймворков, таких как React, Vue и Angular значительно упрощает процесс разработки, так как они предоставляют готовую реализацию для большинства повседневных задач.

Первым крупным шагом на пути упрощения JavaScript-кода стала библиотека jQuery, которая позволила лаконично выполнять операции над DOM[2]-элементами, работать с анимациями и AJAX[3]. jQuery стала настолько популярной, что только в 2021 году, согласно данным StackOverflow[4], React смог обогнать jQuery по распространенности.

С появлением тысяч новых библиотек, а также css-фреймворков (вспомним Bootstrap) разрозненность технологического стека в вебе стала серьезной проблемой. Нужно не только выучить десяток технологий, используемых на текущем проекте, но и

[2] DOM — Document Object Model
[3] AJAX — Asynchronous JavaScript And XML
[4] Результаты опроса

правильно интегрировать их в одну работающую систему. Конечно же, процесс разработки нужно было как-то стандартизировать.



В 2009 году, с релизом NodeJS, JavaScript переживал бурный рост, и следующей эволюционной ступенью языка стали реактивные фреймворки. Их развитие началось с простой идеи: если большая часть фронтенд-приложений управляет только ограниченным набором шаблонов, можно отделить слой представления данных от слоя данных. Поясним на примере.

Представим абстрактный сайт-фотохостинг, где есть только одна страница — список всех фото из базы. Бэкенд данного сайта занимается только хранением контента и передачей его клиенту. На клиенте поставим фреймворк, который будет получать данные и специальный набор инструкций для отрисовки данных — шаблон. Данный фреймворк будет ответственен за:

- подстановку данных в шаблон
- реализацию бизнес-логики (логика манипулирования страницей)
- применение стилей

Слово “реактивные” в названии несет особый смысл: это значит, что интерфейс приложения автоматически изменяется при изменении данных. Например, если на нашем фотохостинге есть заголовок с количеством фото, то при получении новой порции данных

с сервера значение текста в заголовке автоматически увеличится без участия программиста.

Так как фронтенд теперь зависит только от приходящих данных, любое веб-приложение может быть представлено как:

- бэкенд на любом языке отдает данные по запросу в формате JSON
- фронтенд на каком-либо реактивном фреймворке получает данные и осуществляет рендеринг

Понятие SPA

Познакомившись с понятием реактивного фреймворка, введем также концепции, которые часто неразрывно связаны с этими фреймворками.

SPA (Single Page Application, или “одностраничное приложение”) — это веб-приложение, размещенное на одной HTML-странице, которое в процессе взаимодействия с пользователем динамически переписывает текущую страницу с помощью JavaScript. Важно отметить, что SPA загружает весь необходимый код вместе с загрузкой самой страницы.

Вспомним, как писались веб-приложения раньше. Обычно это было несколько HTML-страниц, которые либо лежали на сервере в виде статичных файлов, либо генерировались на сервере с помощью шаблонизатора (например, с помощью PHP или Django). Если пользователь переходит на страницу профиля, то сервер генерирует для него отдельную HTML-страницу и отправляет по сети.

Объясним разницу между этими подходами ниже.

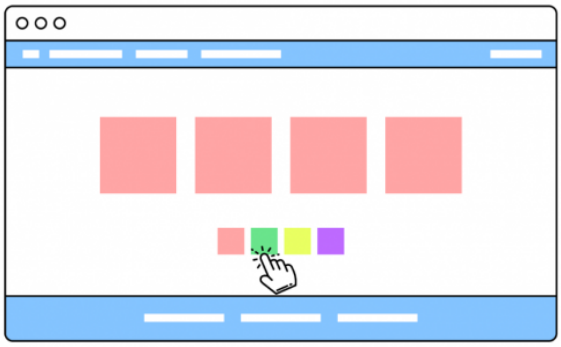
Источник картинок: журнал “КОД”.

Слева — обычный сайт, справа — SPA-приложение.

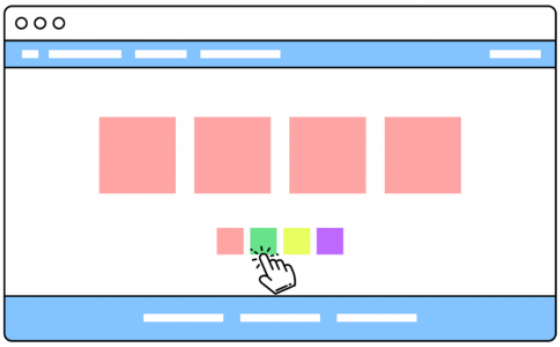
Сайт

Одностраничное приложение

Кнопки снизу меняют цвета всех квадратов на выбранный цвет. Нажмем на зеленую кнопку.

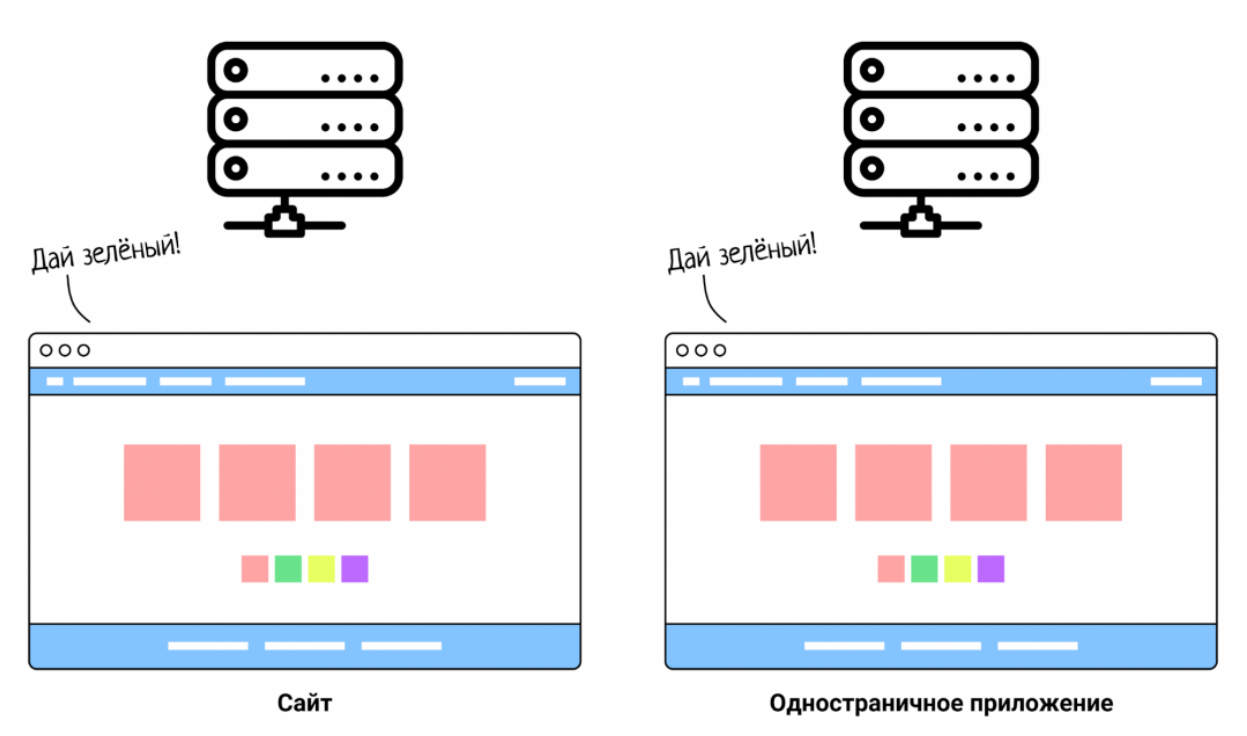


Сайт



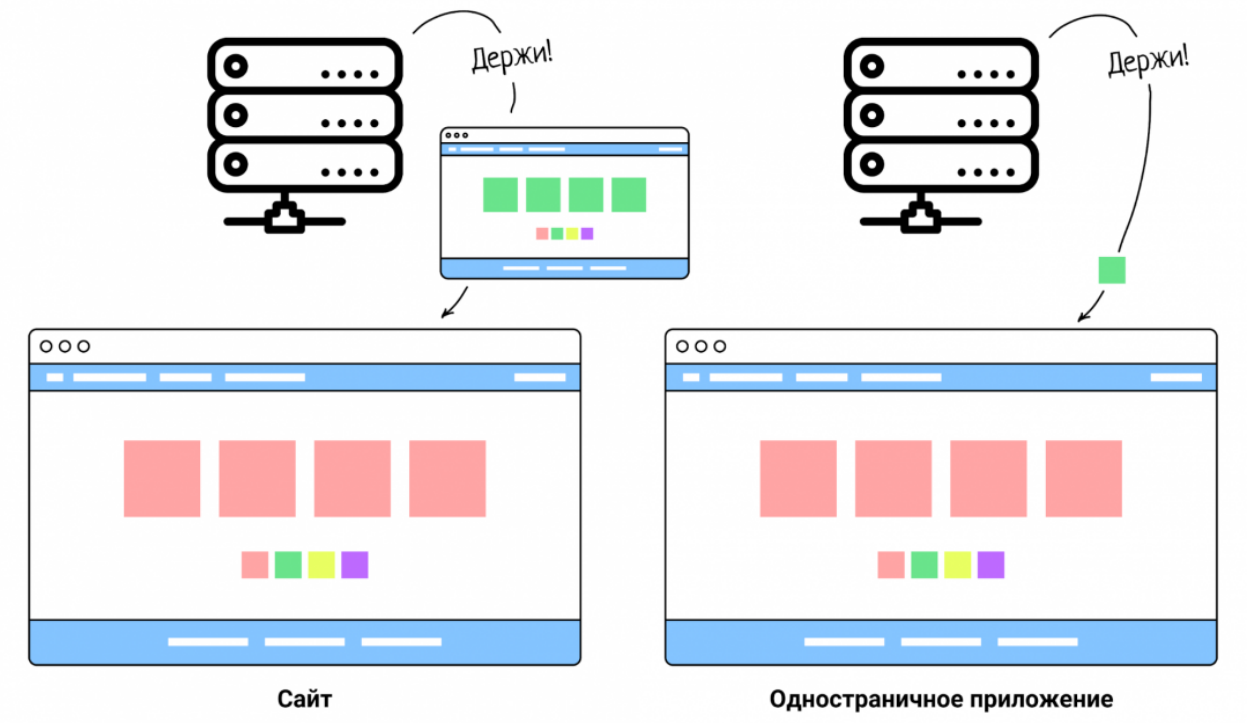
Одностраничное приложение

Разница в поведении проявляется во взаимодействии с сервером:



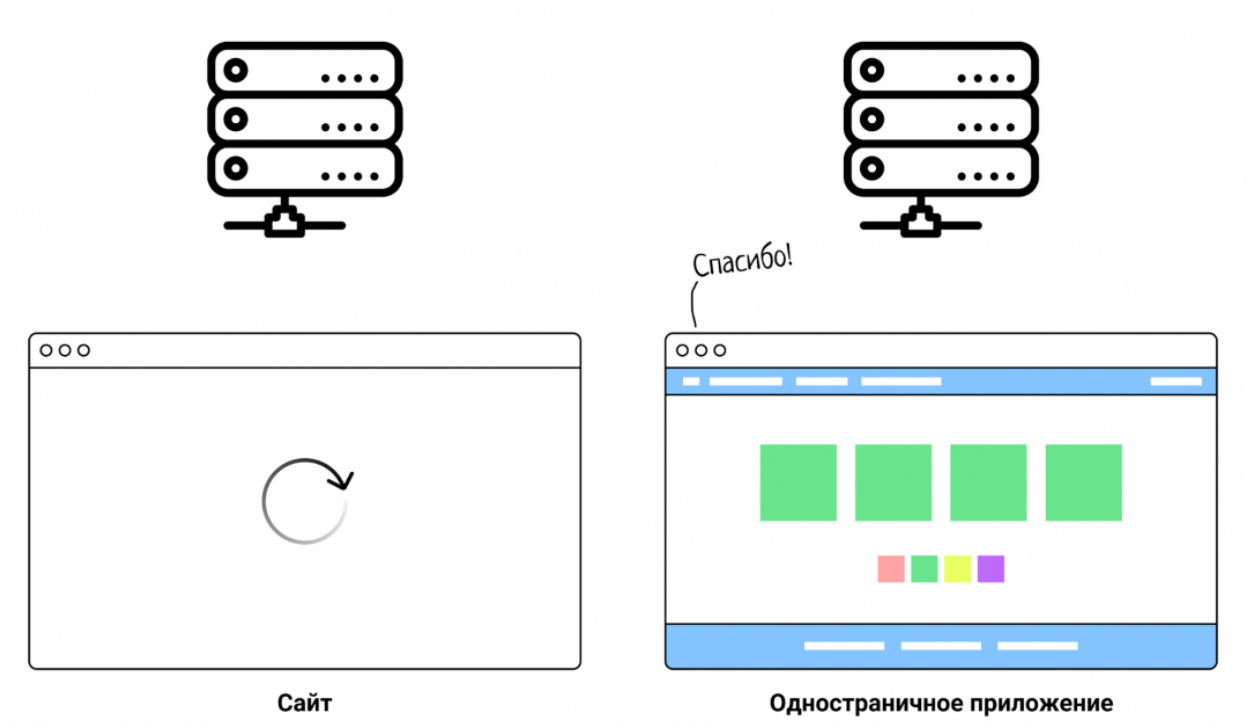
Сайт

Одностраничное приложение



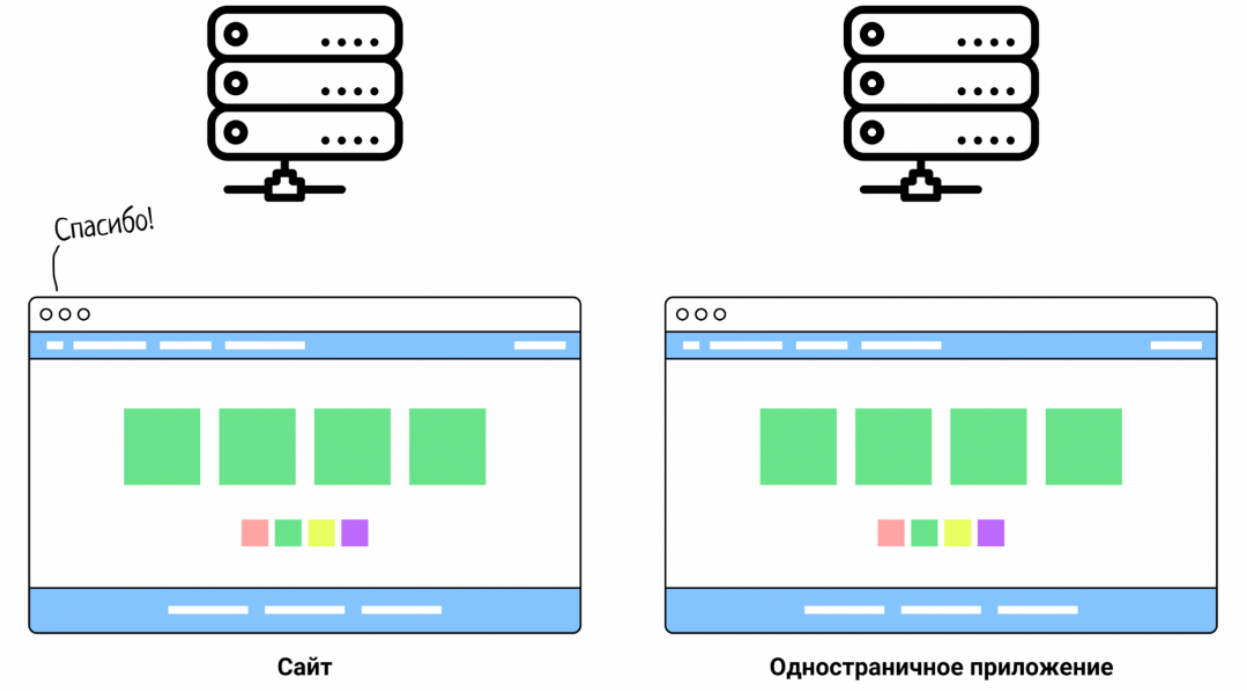
Сайт

Одностраничное приложение



Сайт

Одностраничное приложение



Сайт

Одностраничное приложение

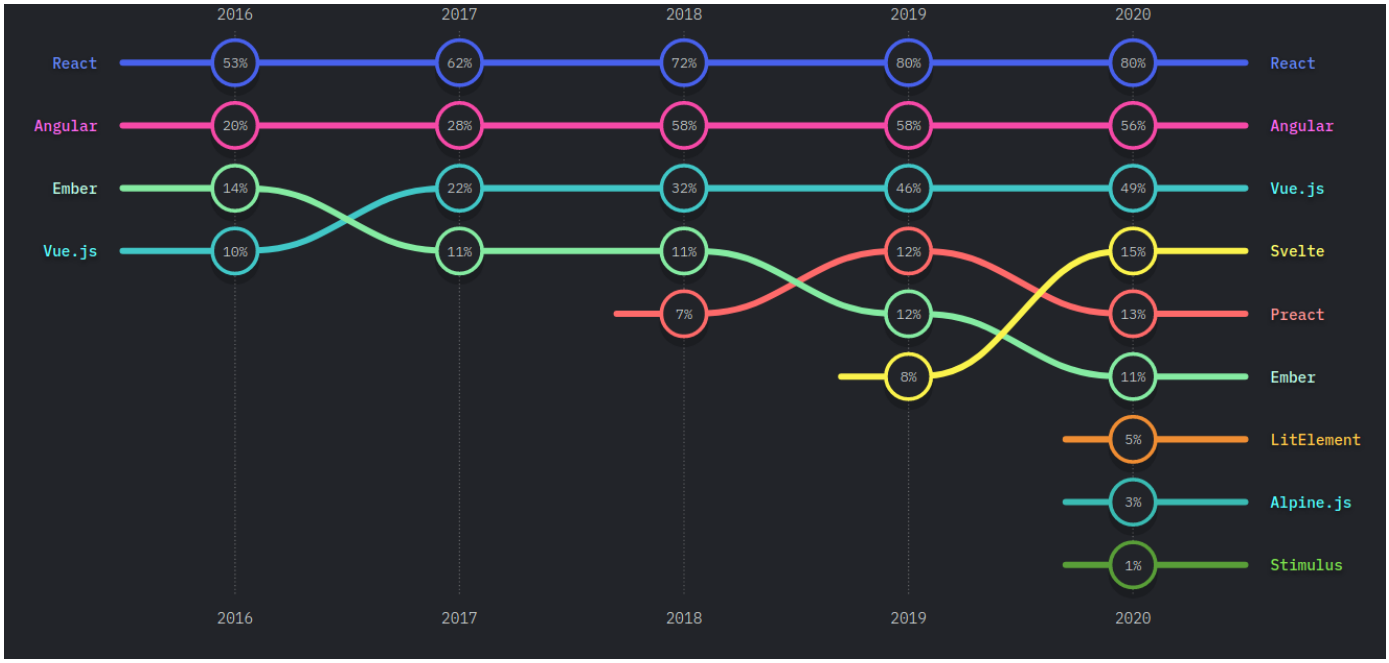
Страница обычного сайта обновится после получения ответа от сервера, так как он отдает целиком новый HTML-файл. В SPA-приложении фреймворк сам динамически перерисовывает содержимое в зависимости от ответа сервера.

Сформулируем основные преимущества SPA:

- не требуется server-side рендеринг;
- сайт — это набор статических файлов (html, js, css, картинки, шрифты), которые лежат на CDN;
- нет затрат на масштабирование фронтенда;
- код фронтенда локализован в одном проекте (раньше часть кода лежала на сервере в виде логики шаблонизации, а остальная часть на клиенте);
- уменьшение затрат на разработку бэкенда (ведь нужно разработать только API);
- уменьшение нагрузки на бэкенд (ведь мы посылаем только API запросы);

Основные черты React

На данный момент рынок делят 3 технологии для создания SPA: React, Vue и Angular (Svelte набирает популярность, но еще не входит в “большую тройку”). В последнем исследовании State of JS представлен их рейтинг по распространенности в проектах:



Мы не будем их сравнивать, но отметим основные сильные стороны React, которые сделали его самой популярной технологией для создания SPA-приложений.

React — это декларативная, эффективная и гибкая JavaScript библиотека для создания пользовательских интерфейсов. Она позволяет вам собирать сложный UI из маленьких изолированных кусочков кода, называемых “компонентами” (именно так написано в документации). Ее разработала компания Facebook в 2013 году.

Может возникнуть вопрос, почему же он назван библиотекой, если до этого, говоря об SPA, мы постоянно использовали термин “фреймворк”. На самом деле, это действительно так и называть React фреймворком — это ошибка. Это объясняется тем, что React ответственен только за представление (отрисовку) данных. Если вы знакомы с аббревиатурами MVC или MVVM, то React — это лишь буква V в них. Если же нет, то достаточно понять следующее: React выполняет только одну задачу — отрисовать компоненты UI, которые оперируют данными вашего приложения. Он не обязывает разработчика соблюдать четкую структуру проекта и не предоставляет каркас проекта, как все фреймворки.

Немного о разнице между библиотекой и фреймворком: ru, eng.

Однако на практике в речи React часто называют фреймворком, так как подразумевают еще и его экосистему, например, связку React+react-router-dom+Redux.

У React есть несколько важных преимуществ:

1. Компонентный подход. Компонент в React — это основной строительный блок для создания фрагментов HTML-кода, подходящих для повторного использования.
2. Предоставляет удобный слой абстракции, избавляя от необходимости работать с DOM напрямую.
3. Универсальность: React можно использовать на сервере и при разработке мобильных приложений.
4. Очень большое сообщество и поддержка крупной компании Facebook.
5. Как следствие пункта 4, большой набор готовых библиотек на все случаи жизни.

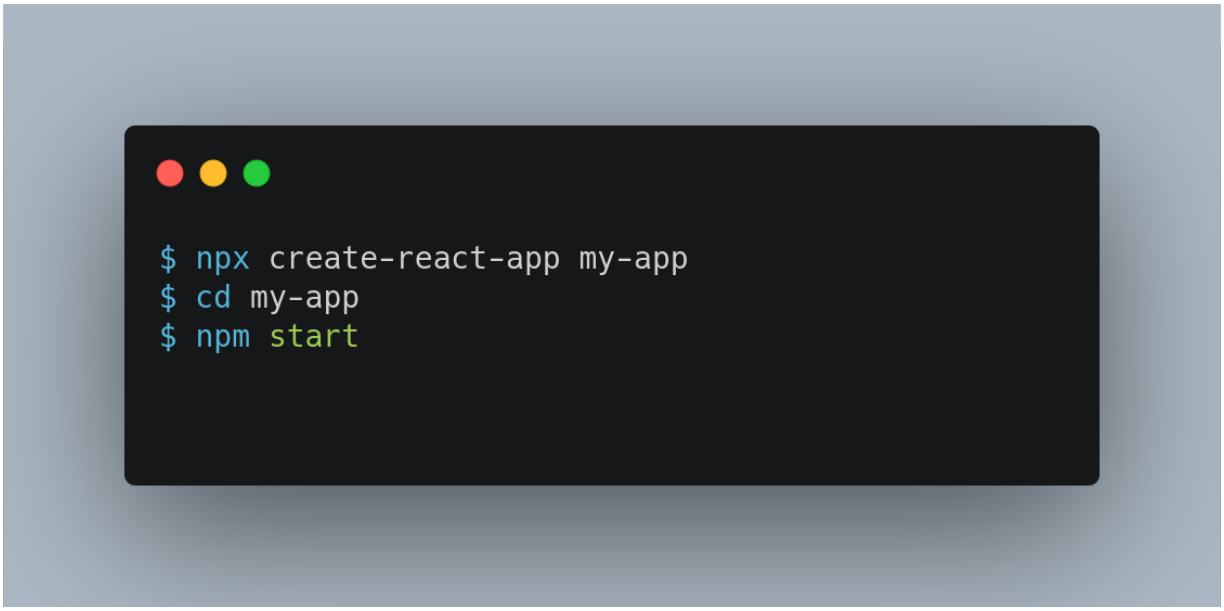
Перейдем к рассмотрению React на практике.

Устанавливаем и настраиваем React

Чтобы начать создавать приложения на React, необходимо установить NodeJS и npm. Помимо этого, необходимо быть знакомым с понятиями HTML, CSS, JavaScript и знать

некоторые особенности нового синтаксиса JavaScript ES6+ (памятка по ним в отдельном конспекте).

Для новичков готовый каркас React-приложения помогает создать утилита create-react-app. Все, что нужно сделать, чтобы создать новый проект, — это написать в терминале



Приложение будет доступно по адресу http://localhost:3000.

Create React App не обрабатывает бэкенд логику или базы данных, он только предоставляет команды для сборки фронтенда, поэтому вы можете использовать его с любым бэкэндом. “Под капотом” используются Babel и webpack, но вам не нужно ничего знать о них.

С этого момента и начнем постигать React на практике.

Полезные ссылки урока:

ECMAScript vs JavaScript

Документация React

React vs Angular vs Vue