ОСНОВЫ REACT

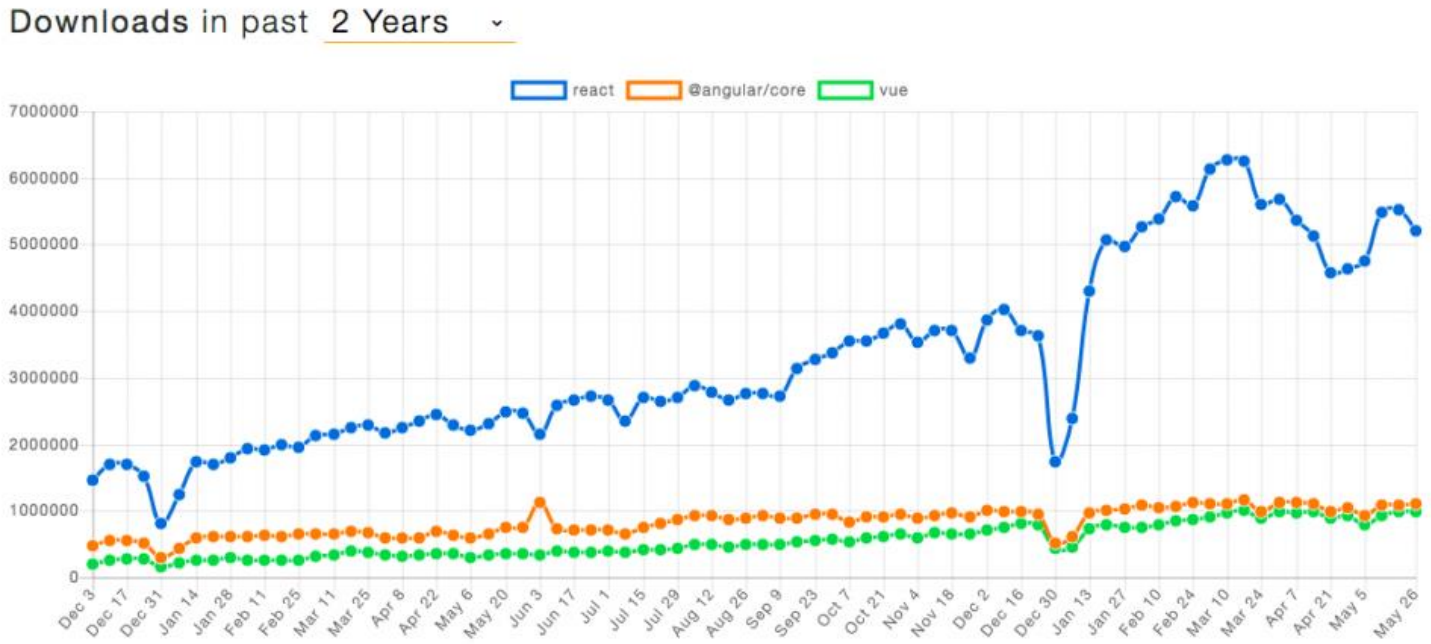
LUXOFT TRAINING

LXFT LISTED NYSE

ПОЧЕМУ REACT?

- 3 САМЫХ ПОПУЛЯРНЫХ ФРЕЙМВОРКА НА РЫНКЕ:

- ANGULAR
- REACT.JS
- VUE.JS



сравнение фреймворков по числу загрузок из NPM

REACT - БИБЛИОТЕКА ИЛИ ФРЕЙМВОРК?

- Что такое фреймворк?
  - Делает приложение более структурированным
  - Дает необходимые инструменты (и ненужные тоже)
  - Следует какому-то архитектурному паттерну (MVC, MVVM, MVP…)
- Что такое библиотека?
  - Дает один инструмент для решения конкретной проблемы
  - Решает конкретную проблему
  - Больше ничего =)
- React.js — это фреймворк или библиотека?

REACT – ЭТО БИБЛИОТЕКА

- ПРОСТО ИНТЕРФЕЙС:

Многие используют React как V в MVC. Так как React никак не соотносится в другими средствами в технологическом стеке, его вполне можно попробовать для небольшой части в существующем проекте.

JSX VS HTML

- Используя React.js, мы отходим от HTML и в то же время приближаемся к JS. Вот что я имею в виду:

Angular

```
<ul>
<li *ngFor="let hero of heroes">
  {{hero.name}}
</li>
</ul>
```

React.JS использует JavaScript:

```
<ul>
    { heroes.map(hero =>
        <li key={hero.id}>{hero.name}</li>
    )}
</ul>
```

React	Ember	Angular	Knockout
ИСПОЛЬЗУЙТЕ JS :)	{{# each}}	*ngFor	data-bind=”foreach”

JSX VS HTML

- В чем преимущество JSX? Используя Angular и Vue, вы выявляете ошибки главным образом во время выполнения, а в случае React.js — во время компиляции, что существенно упрощает жизнь.

```
 + 1 hidden modules

ERROR in ./index.js
Module build failed: SyntaxError: /Users/ptkach/Documents/work/projects/music/index.js: Unterminated JSX contents (31:21)
  29 | }
  30 |
> 31 | ReactDOM.render(<App>, document.getElementById('root'));
     |                     ^
  32 |
    at Parser.pp.raise (/Users/ptkach/Documents/work/projects/music/node_modules/babel-core/node_modules/babylon/index.js:1378:13)
    at Parser.pp.jsxReadToken (/Users/ptkach/Documents/work/projects/music/node_modules/babel-core/node_modules/babylon/index.js:3961:12)
    at Parser.<anonymous> (/Users/ptkach/Documents/work/projects/music/node_modules/babel-core/node_modules/babylon/index.js:4315:21)
    at Parser.readToken (/Users/ptkach/Documents/work/projects/music/node_modules/babel-core/node_modules/babylon/index.js:3632:22)
```

ПРИВЕТ REACT!

- Библиотека, созданная Facebook
- Только для пользовательского интерфейса
- Легко изучить
- Можно использовать не только в браузере
- Высокая скорость
- Большая экосистема
- Огромная популярность (~100 000 звезд на Github начиная с 2013 г.)

ЧТО ТАКОЕ REACT?

- Упрощенно, это библиотека JavaScript, созданная и поддерживаемая компанией Facebook для построения пользовательских интерфейсов (UI).
- ReactJS никак не соотносится с используемым технологическим стеком, и поэтому ReactJS можно использовать для того, чтобы:
  - Добавлять повторно используемые компоненты
  - (верхние, нижние колонтитулы и т. д.)
  - Полностью создавать всю пользовательскую часть интерфейса (как Facebook)

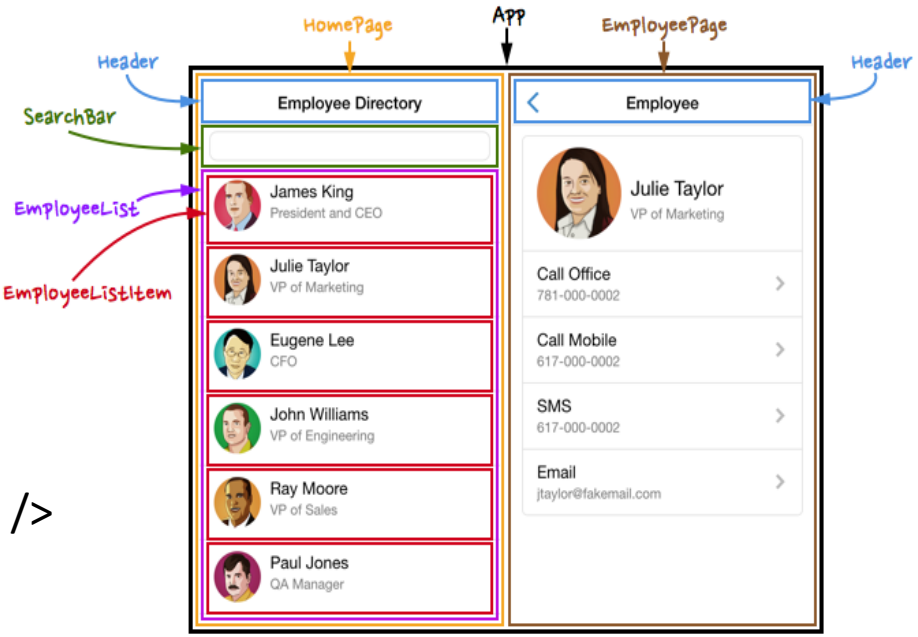
ЧТО ТАКОЕ REACT?

- Существует огромное сообщество пользователей React.js. На данный момент React имеет ~100 000 звезд на Github.
- Вокруг React программисты выстраивают целую вселенную с помощью Redux (самая популярная потоковая реализация), GraphQL, Relay и таких инструментов, как горячая перезагрузка и машина времени.
- Многие крупные компании (Netflix, AirBnb, Uber, Facebook, Instagram, Yahoo и др.) уже используют React и создают множество полезных вещей.
- В React есть все, что нужно для разработки приложений. Существуют десятки библиотек, например: React-bootstrap, react material design, react sliders, scrollers, grids, react animations и т. д.

КОМПОНЕНТ REACT

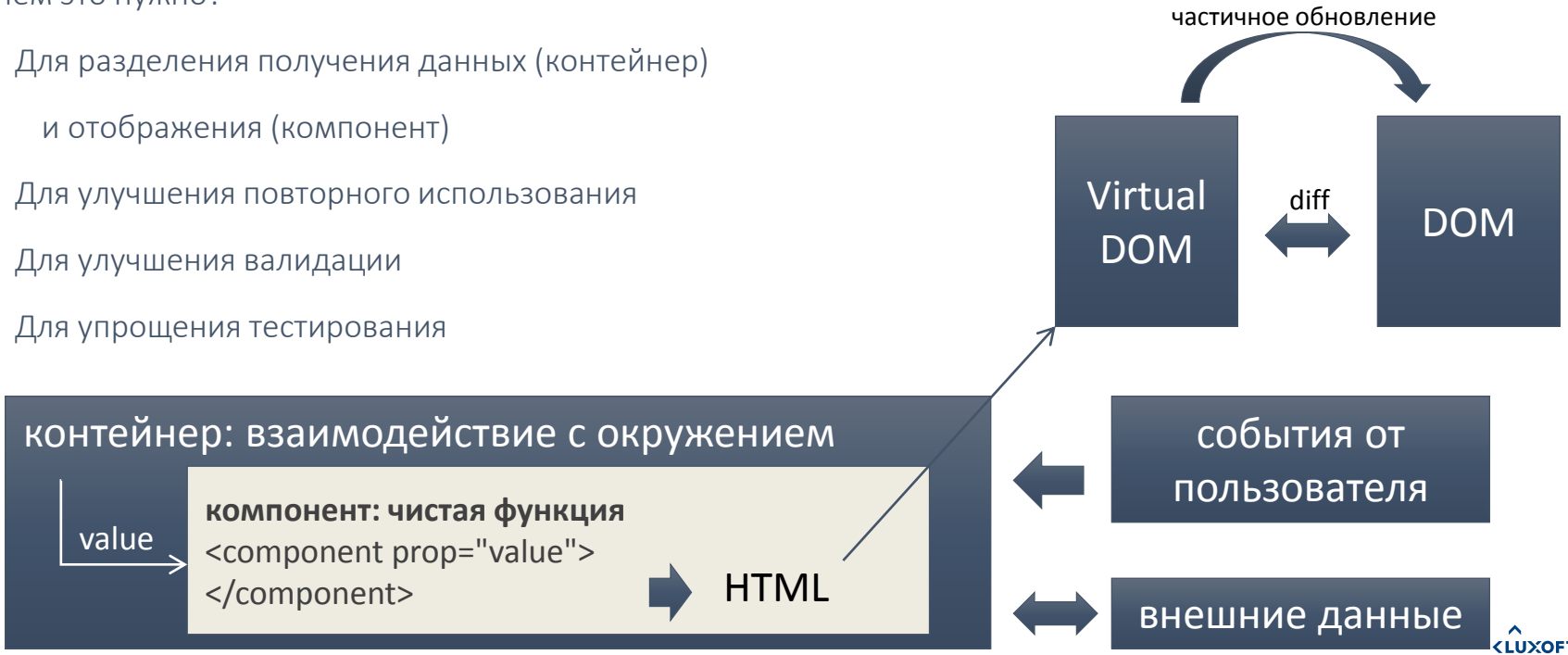
- Любая часть страницы может быть компонентом
- Компонент React объединяет представление и логику

```
<App>
  <HomePage>
    <Header title="Employee Directory"/>
    <SearchBar />
    <EmployeeList employees={employees} />
  </HomePage>
</App>
```



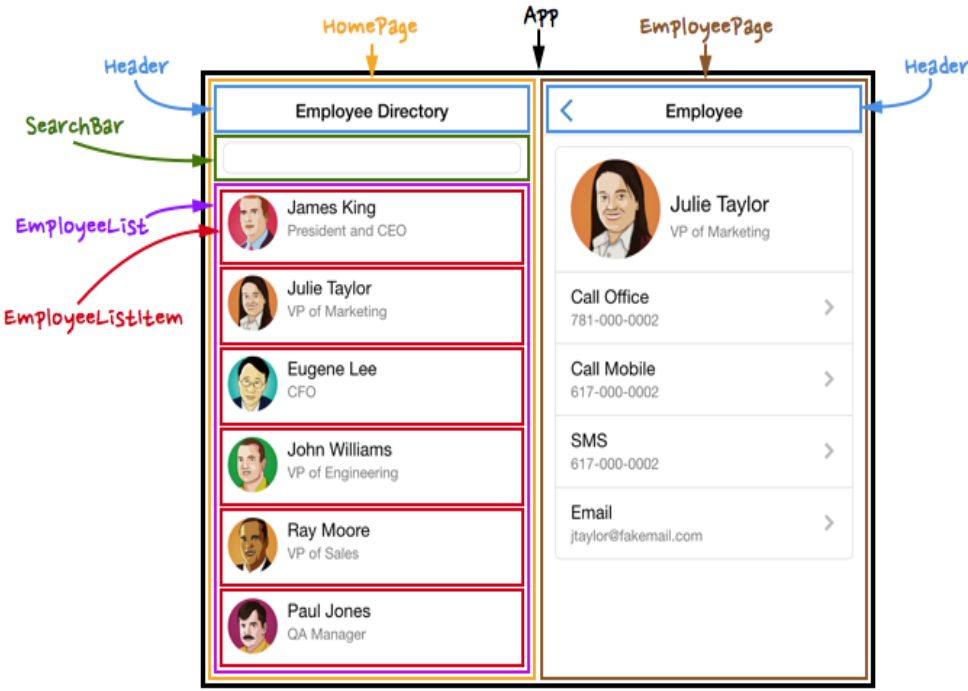
КОНТЕЙНЕРЫ И КОМПОНЕНТЫ

- Мы будем разделять **Контейнеры** (умные компоненты, которые получают данные, содержат логику и предоставляют ее преемникам) и **Компоненты**.
- Зачем это нужно?
  - Для разделения получения данных (контейнер)
  - и отображения (компонент)
  - Для улучшения повторного использования
  - Для улучшения валидации
  - Для упрощения тестирования



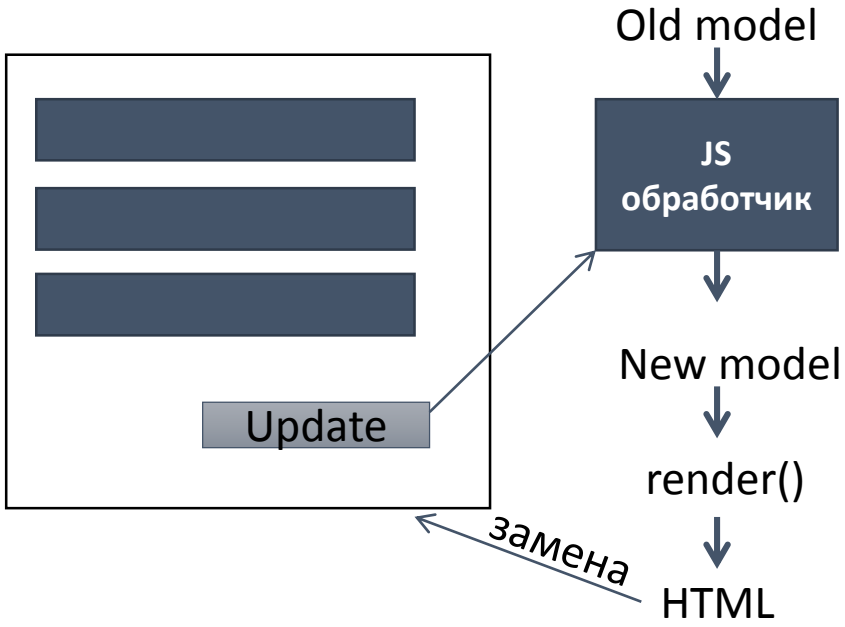
ИЕРАРХИЯ КОМПОНЕНТОВ

- С помощью React приложения собираются из компонентов, имеющих определенную иерархию.
- Самый простой способ структурировать приложение — определить зоны ответственности для каждой части интерфейса и нарисовать рамку вокруг нее, как на иллюстрации.



БЕЗ VIRTUAL DOM

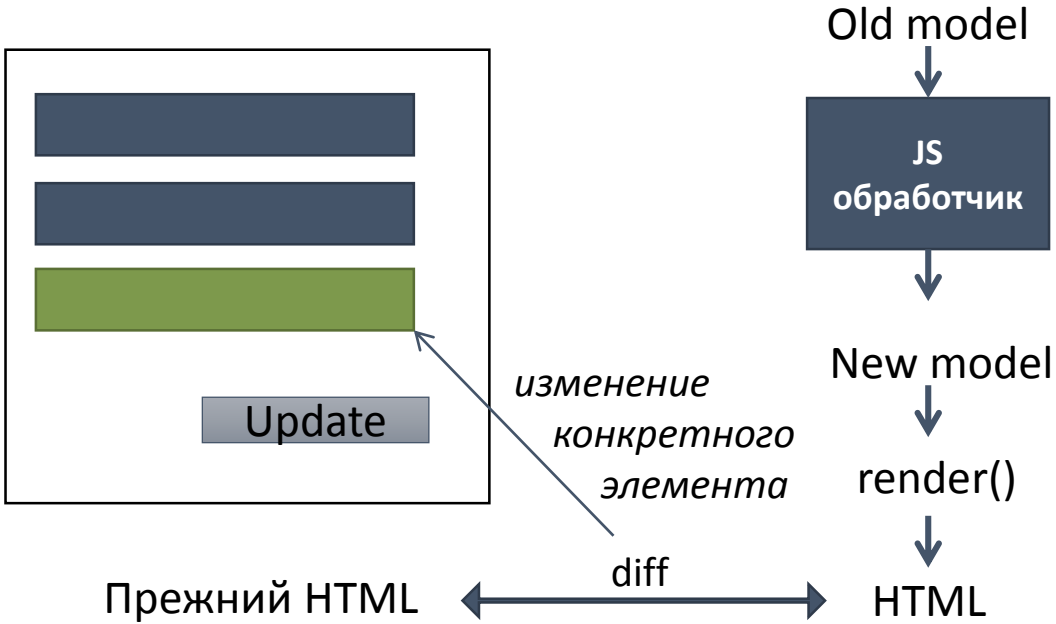
- Сегодня деревья DOM могут быть очень большими, манипуляции с DOM — запутанными, а учет предыдущих состояний DOM — весьма трудоемким.
- Virtual DOM — это абстракция HTML DOM. Эта модель не занимает много места и не связана с зависящими от браузера особенностями реализации.



Постоянная замена **всего HTML** – потеря производительности.

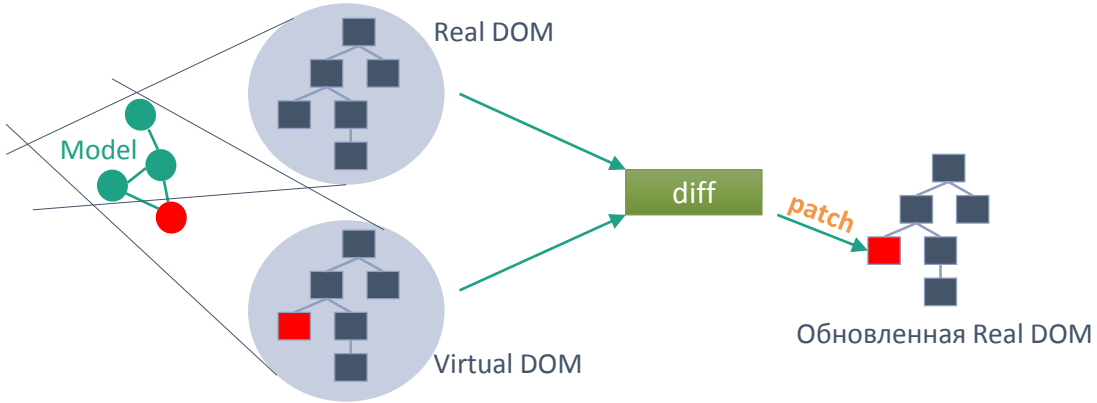
VIRTUAL DOM

- Использование Virtual DOM позволяет React сравнивать старый HTML с новым, находить изменения и применять только их:

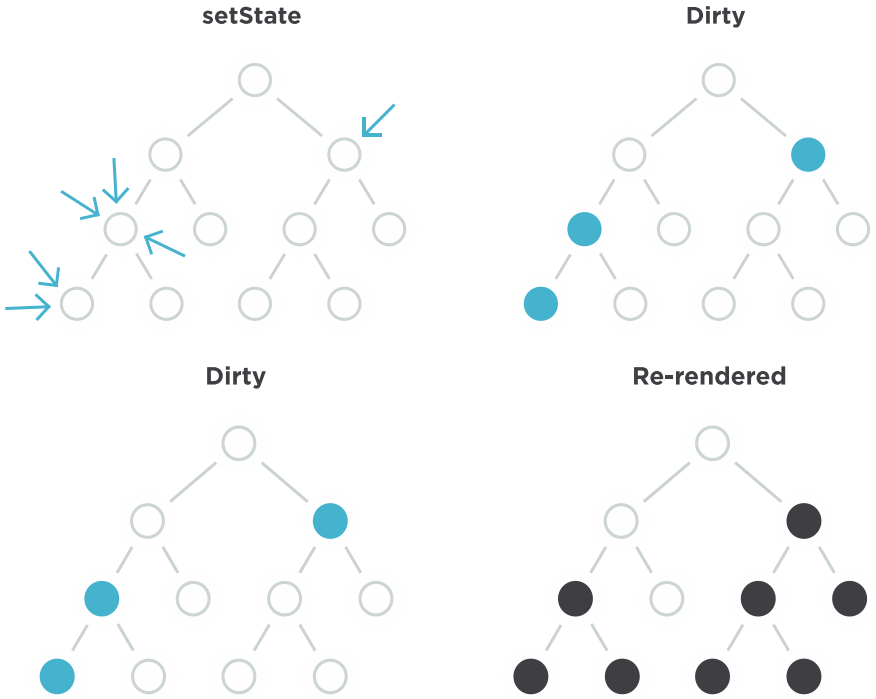


VIRTUAL DOM

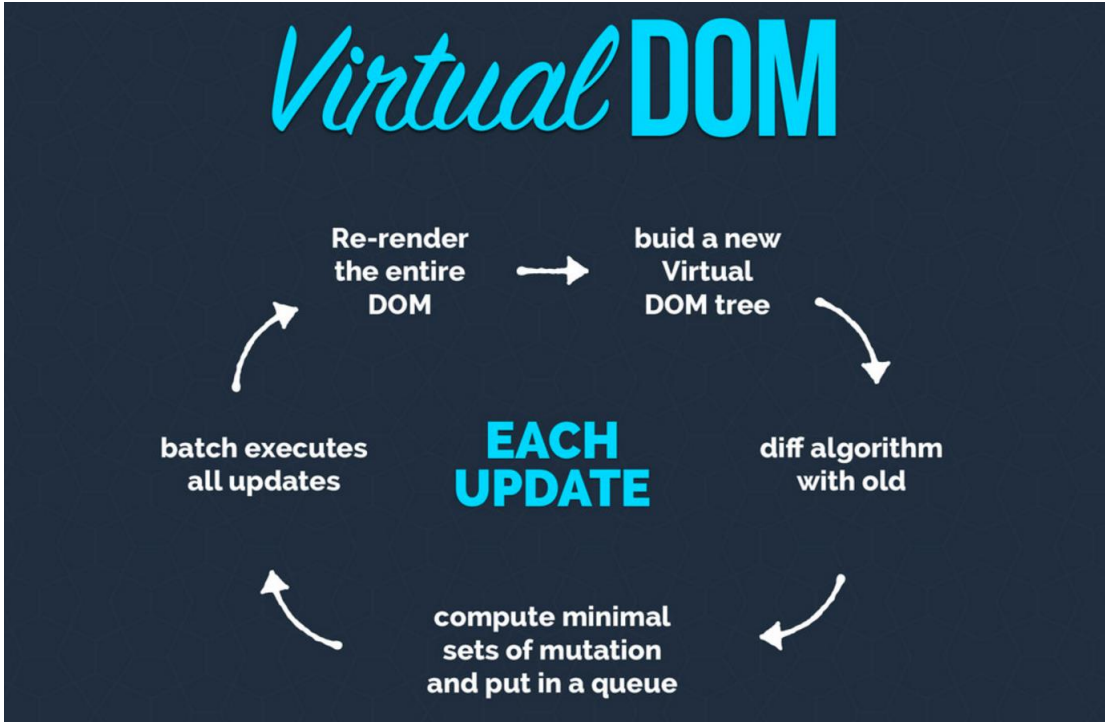
- Использование Virtual DOM позволяет React сравнивать старый HTML с новым, находить изменения и применять только их:



АЛГОРИТМ ОБНАРУЖЕНИЯ РАЗЛИЧИЙ (DIFF)



VIRTUAL DOM

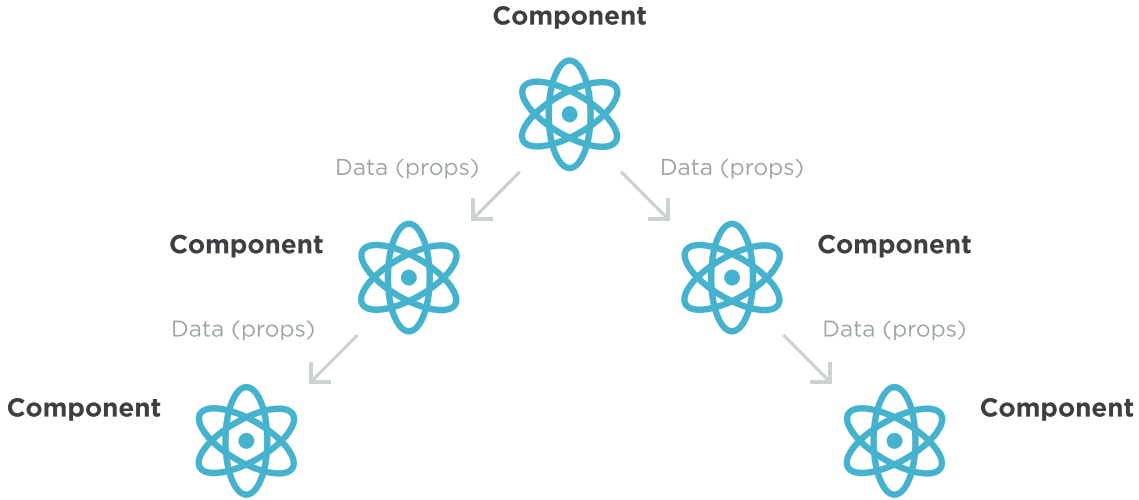

Virtual DOM
Re-render the entire DOM
buid a new Virtual DOM tree
diff algorithm with old
compute minimal sets of mutation and put in a queue
batch executes all updates
EACH UPDATE

ПОТОК ДАННЫХ

ReactJS реализует однонаправленный поток данных. Это означает, что данные передаются сверху вниз через свойства, от верхнего компонента к дочерним и т. д.

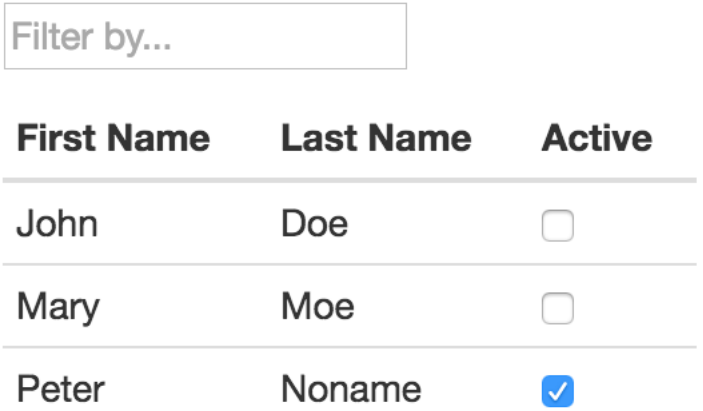
```
...
// Простой компонент передается по свойствам
<Component property1="value" property2=”anotherValue" />
...
```

Вместо коммуникации друг с другом, компоненты React взаимодействуют через родительский элемент.



ПОТОК ДАННЫХ

- Предположим, нам нужно разработать что-то вроде этого:
- Прежде всего разделяем на компоненты:
  - Виджет
  - Таблица
  - Запись таблицы
  - Клетка действия
  - Фильтр



- Иерархия компонентов:
  - Виджет -> Фильтр

    Таблица -> Строчки таблицы -> Ячейки таблицы
- Глядя на эту схему, вы увидите только один разумный способ построения потока данных в виджете.

ПРАКТИКА

Блок 1.
Задание 1.
Основы React.