React

Что такое React?

- Это библиотека на Javascript, которая дает возможность что-то удобно и быстро нарисовать
- В React полно фишек, которые облегчают жизнь разработчика:
- Удобная обработка событий пользователя
- Простая перерисовка
- Virtual dom
- Реакт «оброс» большим количеством библиотек под разные задачи: роутинг, формочки на 50 полей, анимации, и т. д.

Как запустить / установить?

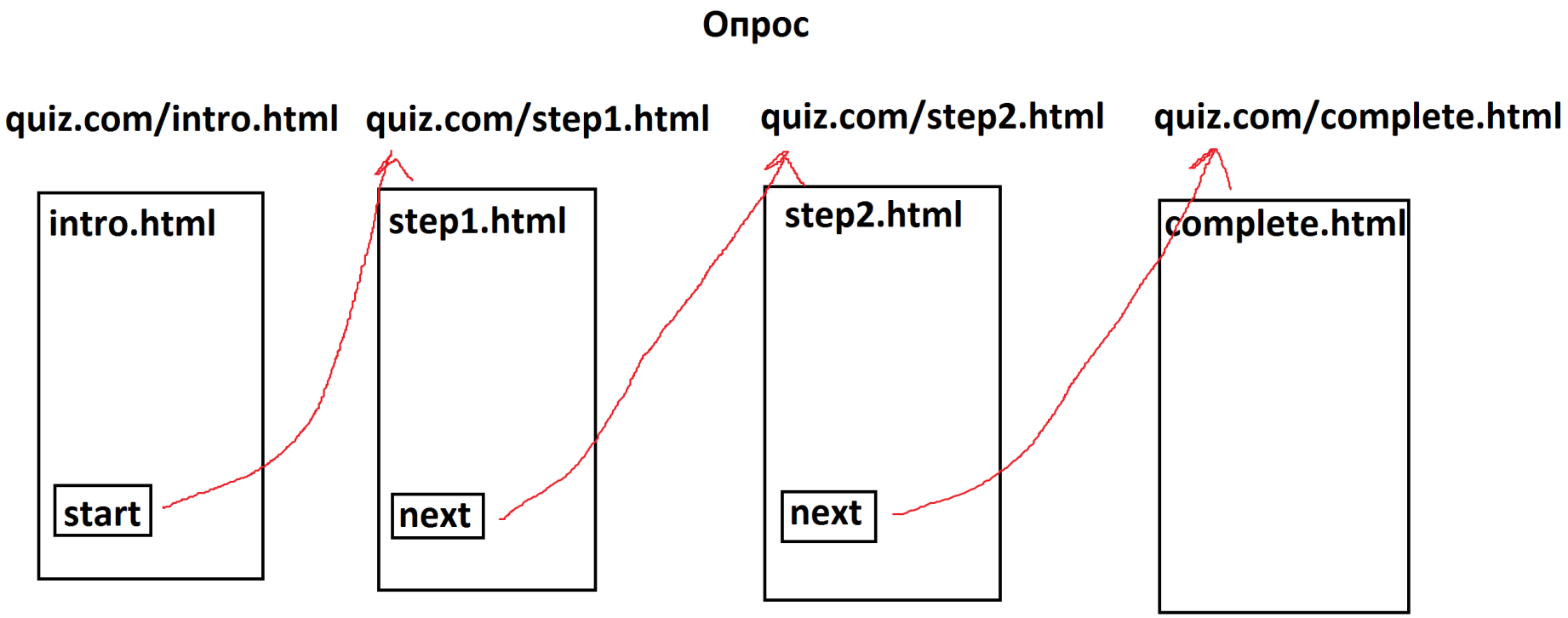
- Из html
- CreateReactApp
- https://codesandbox.io/ или https://jsfiddle.net/boilerplate/react-jsx

Стрелочные функции

```
const doSomething = ( ) => {
    / * do something * /
};
```

```
const summ = (a, b) => {
    return a + b;
};
```

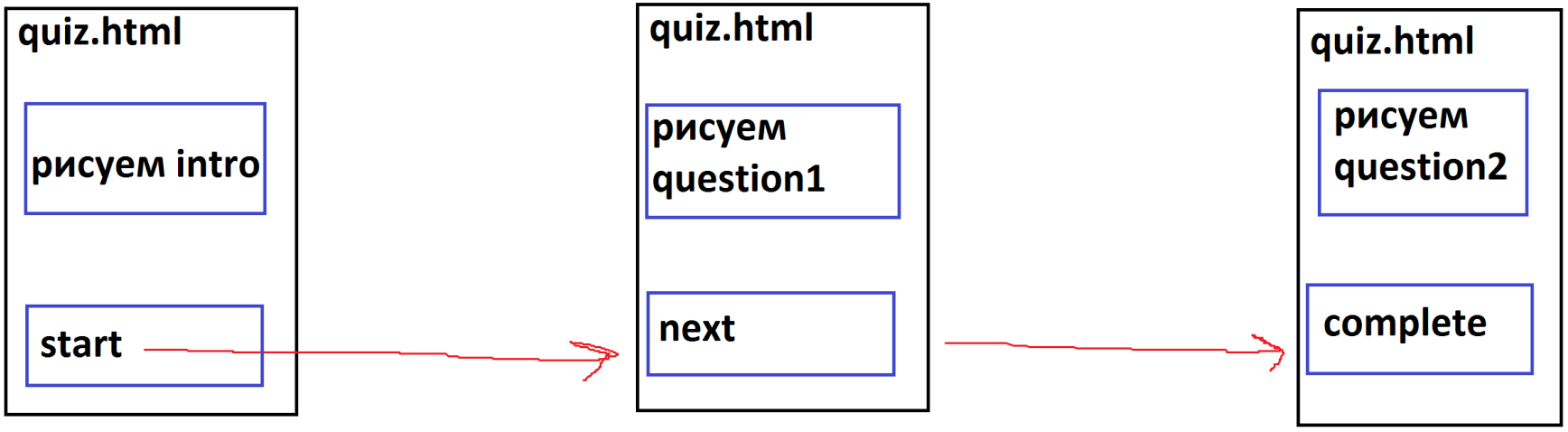
Как раньше делали страницы



SPA

Quiz

quiz.com



когда нажимаем start - остаемся на той же страницы и перерисовыем контент с помощью js

когда нажимаем start - остаемся на той же страницы и перерисовыем контент с помощью js

Как нарисовать кнопку ?

```
class Button extends React.Component {
    render(){
        return(
            <button>Click me</button>
        )
    }
}

ReactDOM.render(<Button />, document.getElementById('root'));
```

Как нарисовать кнопку ?

```
class Button extends React.Component {
    render(){
        return(
            <button />
        )
    }
}

ReactDOM.render(<Button />, document.getElementById('root'));
```

Свойства

```
class Button extends React.Component {
    render(){
        return(
            <button>Click me!</button>
        )
    }
}

ReactDOM.render(<Button />, document.getElementById('root'));
```

Свойства - строки

```
class Link extends React.Component {
    render(){
        return(
            <a target="_blank" href="https://www.google.com/">Ссылка на гугл</a>
        )
    }
}

ReactDOM.render(<Link />, document.getElementById('root'));
```

Свойства - числа

```
class Input extends React.Component {
    render(){
        return(
            <input type="number" value={20} />
        )
    }
}

ReactDOM.render(<Input />, document.getElementById('root'));
```

Свойства функции

```
let count = 0;

function writeClicks() {
    count += 1;

    alert('you clicked times ' + count);
}

class Button extends React.Component {
    render(){
        return(
            <button onClick={writeClicks}>Click me!</button>
        )
    }
}

ReactDOM.render(<Button />, document.getElementById('root'));
```

Еще пример для сравнения

```
class Accordion extends React.Component {
  state = { opened: true };

  toggleOpen = () => {
    let opened = this.state.opened;

    this.setState({ opened: !opened });
  }

  render() {
    let text = "";
    let opened = this.state.opened

    if (opened) {
      text = "Закрыть";
    } else {
      text = "Открыть";
    }

    return (
      <div>
        <button onClick={this.toggleOpen}>{text}</button>
        {opened && <div>А этот текст спрятанный</div>}
        <div>А этот текст всегда отображается</div>
        <div>Еще какой-то текст</div>
        <a target="_blank" href="https://www.google.com/">Ссылка на гугл</a>
      </div>
    );
  }
}

ReactDOM.render(<Accordion />, document.getElementById('root'));
```

Как нарисовать несколько элементов

```
class HelloWorld extends React.Component {
  render() {
    return (
      <div>
        <h1>Hello world!</h1>
        <a href="https://reactjs.org/">React documentation</a>
        <img src="https://icatcare.org/app/uploads/2018/07/Helping-your-new-cat
      </div>
    );
  }
}

ReactDOM.render(<HelloWorld />, document.getElementById('root'));
```

Стили

```
<style>
  .blackSquare {
    background: ■black;
    height: 20px;
    width: 600px;
  }
</style>
```

```
class StylesExample extends React.Component {
  render() {
    return (
      <div style={{ background: 'green', width: 500, height: 200 }}>
        <span style={{ fontSize: 30, color: 'red' }}>Стилизованный текст</span>
        <div style={{ background: 'yellow', border: '3px dashed', marginTop: 20 }}>Еще текст</div>
        <div className="blackSquare" />
      </div>
    );
  }
}

ReactDOM.render(<StylesExample />, document.getElementById('root'));
```

Работа с состоянием

```
class StateExample extends React.Component {
  state = { opened: false };

  toggleOpen = () => {
      const opened = this.state.opened;
      this.setState({ opened: !opened });
  }

  render() {
      const opened = this.state.opened;

      let text = '';

      if (opened) {
          text = 'Спрятать кота';
      } else {
          text = 'Показать кота';
      }

      return (
          <div>
              <button onClick={this.toggleOpen}>{text}</button><br />
              {opened === true && <img src="https://icatcare.org/app/uploads/2018/07/Helping-your-new-cat
          </div>
      );
  }
}

ReactDOM.render(<StateExample />, document.getElementById('root'));
```

Работа с состоянием

- Вся динамика приложения (ввод пользователя, какое-то состояние приложение) хранятся в state
- Когда мы вызываем setState – мы кладем в state новое значение и компоненте вызывается render( ) – все перерисовывается
- Если ваши данные как-то будут меняться, скорее всего вам стоит положить их в state

Работа с состоянием

- Если вы кладете в setState объект, или массив, например setState({ students: […] }), этот массив должен быть создан заново.
- Нельзя вот так:
- const prevStudents = this.state.students;
- prevStudents.push(“Вася”);
- this.setState({ students: prevStudents });

Работа с состоянием

- Придется клонировать массив (создать новый, но с теми же элементами:
- https://www.freecodecamp.org/news/how-to-clone-an-array-in-javascript-1d3183468f6a/

```
numbers = [1, 2, 3];
numbersCopy = [];

for (i = 0; i < numbers.length; i++) {
  numbersCopy[i] = numbers[i];
}
```

Один компонент рисует другой

```
class AwesomeButton extends React.Component {
  render() {
      return (
          <button
              style={{
                  background: 'yellow', border: '1px dashed',
                  cursor: 'pointer', fontSize: 20, padding: 20
              }}
          >
              Нажми меня
          </button>
      );
  }
}

class Example extends React.Component {
  render() {
      return (
          <div>
              <AwesomeButton />
          </div>
      );
  }
}

ReactDOM.render(<Example />, document.getElementById('root'));
```

Общение между компонентами – пропсы

```
class Circle extends React.Component {
    render() {
        const clicks = this.props.clicks;
        const width = clicks * 10;
        const height = width;

        return (
            <div
                style={{
                    background: 'red',
                    borderRadius: '50%',
                    height: height,
                    width: width
                }}
            />
        );
    }
}
```

```
class Clicker extends React.Component {
    state = { clicks: 0 }

    handleClick = () => {
        const previousClicks = this.state.clicks;
        this.setState({ clicks: previousClicks + 1 });
    }

    render(){
        const clicks = this.state.clicks;

        return(
            <div>
                <button onClick={this.handleClick}>Click me!</button>
                <Circle clicks={clicks} />
            </div>
        )
    }
}

ReactDOM.render(<Clicker />, document.getElementById('root'));
```

Общение между компонентами – коллбэки

```
class Cat extends React.Component {
    render() {
        const name = this.props.name;
        const onSleep = this.props.onSleep;

        return (
            <div>
                <h2>{name}</h2>
                <button onClick={onSleep}>Идти спать</button>
            </div>
        );
    }
}
```

```
class Cats extends React.Component {
    state = { sleepyCats: 0 }

    handleSleep = () => {
        const previousSleepyCats = this.state.sleepyCats;
        this.setState({ sleepyCats: previousSleepyCats + 1 });
    }

    render(){
        const sleepyCats = this.state.sleepyCats;

        return(
            <div>
                <Cat name="Пушок" onSleep={this.handleSleep} />
                <Cat name="Рыжик" onSleep={this.handleSleep} />
                <Cat name="Белыш" onSleep={this.handleSleep} />
                <div>
                    Сейчас спят котов: {sleepyCats}
                </div>
            </div>
        )
    }
}
```

Переиспользуемые компоненты

```
class Cats extends React.Component {
    render() {
        return (
            <div>
                {/*------cat1-------*/}
                <Cat
                    name="Белыш"
                    color="Черный"
                    age={3}
                    hobby="Рвать обои"
                    avatar="https://icatcare.org/app/uploads/2018/07/Helping-your-new-cat
                />
                {/*------cat2-------*/}
                <Cat
                    name="Пушок"
                    color="Белый"
                    age={5}
                    hobby="Бегать и врезаться в стены"
                    avatar="https://www.petmd.com/sites/default/files/styles/article_imag
                />
                {/*------cat3-------*/}
                <Cat
                    name="Рыжик"
                    color="Рыжий"
                    age={7}
                    hobby="Смотреть телевизор"
                    avatar="https://valevets.com/wp-content/uploads/2014/08/kitten1.jpg"
                />
            </div>
        );
    }
}
```

Работа с инпутами

```
class Example extends React.Component {
    state = { value: 'Hello world' };

    onInputChange = (event) => {
        const value = event.target.value;

        this.setState({ value: value });
    }

    render(){
        const value = this.state.value;

        return(
            <div>
                <input onChange={this.onInputChange} value={value} />
                Вы ввели: {value}
            </div>
        )
    }
}
```

```
ReactDOM.render(<Example />, document.getElementById('root'));
```

Как нарисовать массив

```
const students = ['Andrey', 'Nikolay', 'Daria', 'Anna'];
class Example extends React.Component {
    render(){
        return(
            <div>
                {students.map(function (student) {
                    return (
                        <div><h3>{student}</h3></div>
                    );
                })}
            </div>
        )
    }
}

ReactDOM.render(<Example />, document.getElementById('root'));
```

Пример с фильтрами

```
class Example extends React.Component {
    state = {
        students: ['Andrey', 'Nikolay', 'Daria', 'Anna'],
        filter: '',
    };

    onInputChange = (event) => {
        const newValue = event.target.value;

        this.setState({ filter: newValue });
    }

    render(){
        const students = this.state.students;
        const value = this.state.value;
        const filter = this.state.filter;

        const filteredStudents = students.filter(function (student) {
            return student !== filter;
        });

        return(
            <div>
                <div>
                    {filteredStudents.map(function (student) {
                        return (
                            <div><h3>{student}</h3></div>
                        );
                    })}
                </div>
                <input
                    type="text"
                    value={value}
                    onChange={this.onInputChange}
                />
            </div>
        )
    }
```

TODO – LIST

Вспомогательные библиотеки

- Роутинг – React Router
- Формы – React Final Form
- Логика приложения – Redux

Что почитать

- https://ru.reactjs.org/
- https://www.freecodecamp.org/learn/front-end-development-libraries/