ОНЛАЙН-ОБРАЗОВАНИЕ

```
let lesson = {
  id:       '11'
  themes:   ['Router', 'Optimization'],
  date:     '05.04.2018',
  teacher: {
    name:       'Юрий Дворжецкий',
    position: 'Lead Developer’
  }
};
```

Как меня слышно && видно?

Если нет – напишите, если слышите – смайлик в чат.

Скажите пару слов

1. Вопросы?
2. Проблемы?
3. Пожелания?
4. Или что всё хорошо

OTUS

О вебинаре:

Что сможем делать после вебинара?

- Знать и разбираться в AJAX и SPA
- Разрабатывать SPA приложения на React с помощью React Router
- Оптимизировать приложения правдами и неправдами

План

- Немного о классике, модерне и современности
- Много о React Router
- И расскажу про оптимизацию

OTUS

HTML Pages,
AJAX
SPA

Classic HTML

- Страницы получаются GET запросом /index
- Сервер генерирует HTML разметку с данными
- В результирующей HTML ссылки на CSS и JS
- CSS и JS хранятся на сервере и отдаются
- JS анимирует страницы

OTUS

Classic HTML

+ Это просто

+ Генерировать HTML на сервер просто (PHP)

- Пользователь при переходе/действии загружает новую страницу целиком

- Видит «пустую страницу» даже при небольших действиях

OTUS

Упражнение

Найти классический HTML-сайт.

Отправить его в чат.

Полазить по нему с открытой консолью Network (+prserve logs)

Ужаснуться.

OTUS

AJAX (Asynchronous Javascript and XML)

- Страницы получаются GET запросом /index
- Сервер генерирует HTML разметку с минимумом данных
- JS получает и передаёт данные посредтсвом $.ajax | Fetch | XMLHttpRequest, отображает их
- Переход на страницу – это переход на страницу

OTUS

AJAX

\+ Меньше передаётся данных

\+ При действиях не перезагружается страница

\- Тяжело писать что-то сложнее заполнения данными таблицу

\- Видит «пустую страницу» только при переходе

OTUS

Упражнение

Найти страницы с AJAX архитектурой.

Отправить его в чат.

Полазить по нему с открытой консолью Network (+prserve logs)

Увидеть как передаются данные.

SPA (Single Page Application)

- Страница получаются GET запросом /index
- Сервер отдаёт статическую HTML
- JS получает и передаёт данные посредтсвом $.ajax | Fetch | XMLHttpRequest, и рендерит на основе их DOM
- Переход на страницу – это перерендеринг DOM посредством JS (но URL менется)

OTUS

SPA

\+ Передаётся абсолютный минимум данных

\+ При действиях и перехдах не перезагружается страница

\- SEO – сайт прозрачен для поисковиков

\- Хорошее SPA сложно сделать

OTUS

Упражнение

Найти SPA-сайты

Отправить его в чат.

Полазить по нему с открытой консолью Network (+prserve logs)

Увидеть как передпются.

OTUS

React Router v3
React Router v4

Router

- Роутер крайне полезная утлита, помогает в построении больших приложений и, так называемых, SPA – Single Page Application
- React не предоставляет собственных механизмов роутинга, а вот коммьюнити React выпустило react-router
- React роутер синхронизирует состояние Вашего приложения с URL в заивисимости от настроек
- В SPA теперь вместо страниц (т.к. она одна) становятся роуты.

Router

- Путь роута – это строка, за исключением некоторых специальных символов. И может содержать следуюшие обозначения:
  - :paramName – фрагмент пути роута после /, ?, or #. Соответствующая строчка называется параметром
  - () – опциональная часть пути роута
  - * – любые символы до следующего символа паттерна или пути
  - ** - любые символы до /, ?, # в пути роута

Router paths

```
<Route path="/hello">           /hello
<Route path="/hello/:name">     /hello/michael and /hello/ryan
<Route path="/hello(/:name)">   /hello, /hello/michael
<Route path="/files/*.*">       /files/hello.jpg and /files/hello.html
<Route path="/**/*.jpg">        /files/hello.jpg and /files/path/to/file.jpg
```

Компонент Router-a

- Как ни странно, но роутер, это компонет.
- В данном примере три пути (страницы)
- В данном случае в зависимости от URL страницы (hash)будет рендерится один из путей
- Управляет этим hashHistory

```
render((
  <Router history={hashHistory}>
    <Route path="/" component={Home}/>
    {/* add the routes here */}
    <Route path="/grid" component={Grid}/>
    <Route path="/form" component={Form}/>
  </Router>
), document.getElementById('app'));
```

OTUS

Упражнение

Написать роуты для сайта погоды (Ваш функционал + новости и прочая общесайтовая информация).

Link

- В общем случае, если пользователь откроет ссылку то ему отобразится страница по соответствующему пути, но как не перерисовывая всю страницу можно менять роуты?
- Для этого есть специальный компонент Link, который принимает путь к роуту

```
<ul role="nav">
<li>
  <Link to="/grid">Grid</Link>
</li>
<li>
  <Link to="/form">Form</Link>
</li>
</ul>
```

Оптимизация

- Общие элементы будут рендериться на каждом роуте
- Это можно избежать представив приложение в виде вложенных роутов
- Собственно, перериовка компонентов App будет осуществляться только один раз при переходе на корневой роут

```
render((
<Router history={hashHistory}>
  <Route path="/" component={App}>
  {/* make them children of `App` */}
  <Route path="/grid" component={Repos}/>
  <Route path="/form" component={About}/>
  </Route>
</Router>
), document.getElementById('app'));
```

react-router@4.x

```
import {HashRouter, Route, Switch, Link} from 'react-router-dom';

<HashRouter>
  <App>
    <Switch>
      <Route exact path='/' component={Home}/>
      <Route exact path='/schedule' render={Schedule}/>
      <Route path='/schedule/edit' compent={Edit}/>
    </Switch>
  </App>
</HashRouter>

<Link replace to='/schedule/edit'/>
```

Стили и параметры в link

```
<Link to="/grid" activeStyle={{ color: 'red' }}>Grid</Link>
<Link to="/grid" activeClassName="activeLink">Grid</Link>

<Router history={hashHistory}>
  <Route path="/" component={App}>
    <Route path="/grid" component={Repos}/>
    <Route path="/grid/:id" component={Repos}/>
  </Route>
</Router>
</div>

//Grid component
render() {
    return <h2>{this.props.params.id}</h2>
}
```

Browser History

- React Router позволяет не использовать хэши, а полноценные URL.
- Это называется browserHistory
- URL-ы страниц выглядят прекрасно, но это требует серьёзной конфигурации сервера, который обрабатывает запросы на страницу.

```
render((
<Router history={browserHistory}>
    {/* ... */}
</Router>
), document.getElementById('app'))

// handle every other route with index.html, which will contain
// a script tag to your application's JavaScript file(s).
app.get('*', function (request, response){
  response.sendFile(path.resolve(__dirname, 'public', 'index.html'))
})
```

Управление жизненным циклом

Страницу (роут) создаёт сам роутер. Чтобы управлять жизненным циклом, можно использовать следующие методы:

- onEnter – при открытии роута
- onChange – роут тот же, но поменялись парметры
- onLeave – при уходе с роута

```
<Route
  path="/users/:userId/teams"
  onEnter={userIsInATeam} />
<Route
  path="/users/:userId/teams"
  onChange={onChange} />
<Route
  path="/users/:userId/teams"
  onLeave={onLeave} />
```

Что почитать

- Hooks, позволяющие управлять роутингом
- Динамический роутинг (не на компонентах)
- Интеграция с redux

Вопросы?

Упраженение

Опишите Ваше приложение, какие и что у Вас будет в роутах, какие компоненты будут меняться.

OTUS

Оптимизация

Упраженение

У Вас есть React-приложение.

1. С чего начнёте оптимизацию?
2. Как Вы будете его оптимизировать?

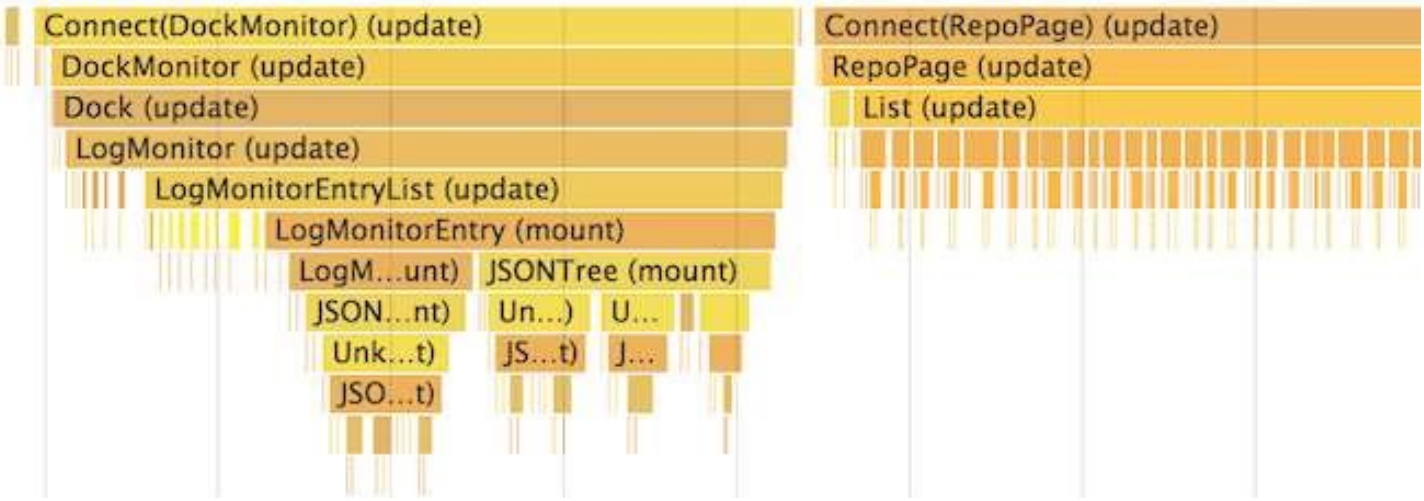
OTUS

Упраженение

Оптимизация начинается с выявления проблемы.

1. Профилируйте
2. Анализируйте
3. Засекайте
4. React tools
5. ?react_perf

?react_perf


Connect(DockMonitor) (update)
DockMonitor (update)
Dock (update)
LogMonitor (update)
LogMonitorEntryList (update)
LogMonitorEntry (mount)
LogM…unt)
JSONTree (mount)
JSON…nt)
Un…)
U…
Unk…t)
JS…t)
J…
JSO…t)
Connect(RepoPage) (update)
RepoPage (update)
List (update)

Оптимизация

- Очень честная
- Честная
- Нечестная

Оптимизация

- Очень честная – архитектураня (SPA, Ajax)
- Честная – техническая (React, zip-ы)
- Нечестная - видимая

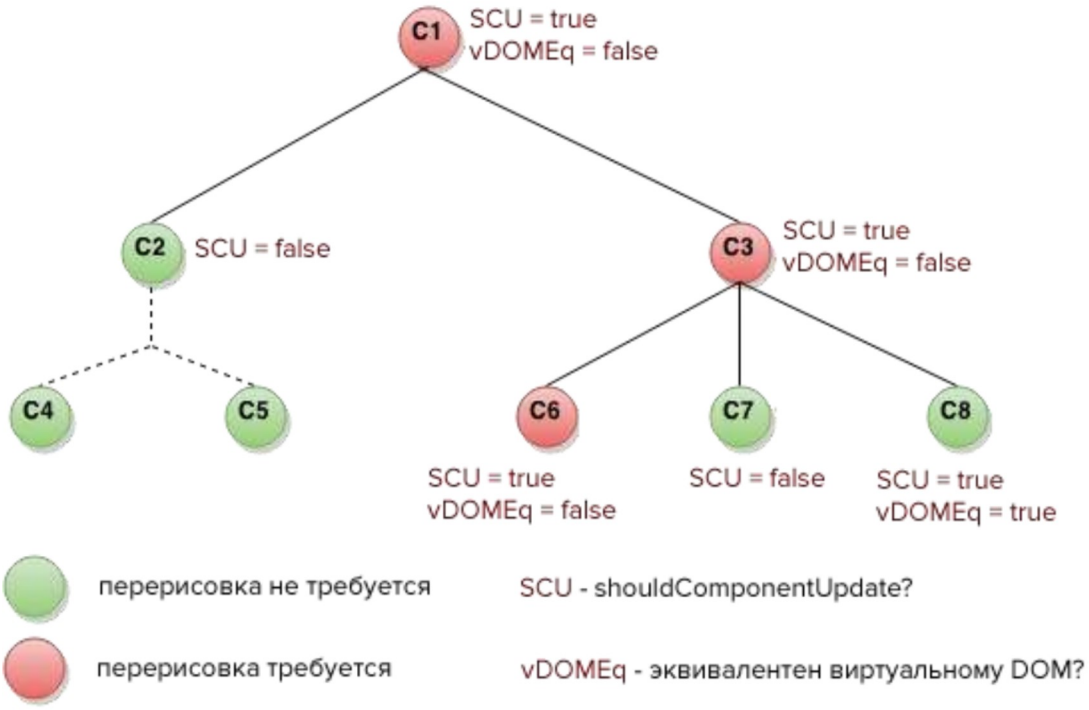
shuouldComponentUpdate

- С помощью этого метода можно сказать React-у, необходимо ли перерисовывать компонент.

```
class MyComponent extends Component {

    shouldComponentUpdate(
        nextState, nextProps
    ) {
        // TODO: add code
        return true;
    }
}
```

Дерево обновления



PureComponent

- Содержит реализацию shouldUpdate
- Props only!

```
class MyComponent extends PureComponent {

}
```

recompose pure

```
// DatagridBody.js
import React, { Children } from 'react';
import pure from 'recompose/pure';

const DatagridBody = ({ resource, ids, data, children }) => (
    <tbody>
        {ids.map(id => (
            <tr key={id}>
                {Children.map(children, (field, index) =>
                    <DatagridCell
                        record={data[id]}
                        key={`${id}-${index}`}
                        field={field}
                        resource={resource}
                    />
                )}
            </tr>
        ))}
    </tbody>
);

export default pure(DatagridBody);
```

recompose shouldUpdate

```
// DatagridBody.js
import React, { Children } from 'react';
import shouldUpdate from 'recompose/shouldUpdate';

const DatagridBody = ({ resource, ids, data, children }) => (
    ...
);

const checkPropsChange = (props, nextProps) =>
 (nextProps.ids !== props.ids ||
  nextProps.data !== props.data);

export default shouldUpdate(checkPropsChange)(DatagridBody);
```

recompose shouldUpdate

```
// List.js
import React from 'react';
import { connect } from 'react-redux';

const List = (props) => ...
const mapStateToProps = (state, props) => {
    const resourceState = state.admin[props.resource];
    return {
        ids: resourceState.list.ids,
        data: Object.keys(resourceState.data)
            .filter(id => resourceState.list.ids.includes(id))
            .map(id => resourceState.data[id])
            .reduce((data, record) => {
                data[record.id] = record;
                return data;
            }, {}),
    };
};

export default connect(mapStateToProps)(List);
```

Что почитать

- https://habrahabr.ru/post/319536/
- https://habrahabr.ru/post/327364/

Вопросы?

Домашнее задание

На весь блок React:
Приложение для самостоятельной работы в блоке React - веб-приложение погоды. На странице приложения должна быть возможность добавлять города в список избранных. По каждому городу показывается информация о температуре, ветре, другие параметры.

ДЗ сегодня: Добавить страницу погоды по конкретному городу. При переходе на нее должен меняться url, показываться информация на несколько дней вперед. (Router)

Дополнительно: browserHistory

Спасибо за внимание!

SPA Вам!