В этом уроке мы с вами будет рендерить страницу товаров. Наши екшены и редьюсеры уже готовы и нам осталось подключить store к компоненту.

Теперь давайте подключим эти данные к нашей компоненте.

Для этого создадим mapStateToProps и добавим его в connect.

import React, {Component} from 'react'

import {connect} from 'react-redux'

import {fetchPhones} from 'actions'

class Phones extends Component {

componentDidMount () {

this.props.fetchPhones()

}

render () {

return (

<div>Phones</div>

)

}

}

const mapStateToProps = state => ({

})

const mapDispatchToProps = {

fetchPhones

}

export default connect(mapStateToProps, mapDispatchToProps)(Phones)

Теперь мы хотим найти все телефоны для этой страницы по массиву ids и телефонам. Но делать мы это будем не прямо в mapStateToProps, а вынесем поиск елементов в файл selectors.

Тогда это будет будет намного легче поддерживать и тестировать.

Давайте напишем что мы получим phones из функции getPhones и передадим в нее state.

import React, {Component} from 'react'

import {connect} from 'react-redux'

import {fetchPhones} from 'actions'

import {getPhones} from 'selectors'

class Phones extends Component {

componentDidMount () {

this.props.fetchPhones()

}

render () {

return (

<div>Phones</div>

)

}

}

const mapStateToProps = state => ({

phones: getPhones(state)

})

const mapDispatchToProps = {

fetchPhones

}

export default connect(mapStateToProps, mapDispatchToProps)(Phones)

Теперь давайте в корне создадим файл selectors и опишем эту функцию.

export const getPhones = state => {

}

Давайте создадим вспомогательную функцию getPhoneById, которая будет получать телефон по id из state

import R from 'ramda'

const getPhoneById = (state, id) => R.prop(id, state.phones)

export const getPhones = state => {

}

Теперь мы можем промапить наши ids, чтобы найти все нужные телефоны

import R from 'ramda'

const getPhoneById = (state, id) => R.prop(id, state.phones)

export const getPhones = state => {

const phones = R.map(id => getPhoneById(state, id), state.phonesPage.ids)

return phones

}

Теперь эти данные должны быть доступны в нашей компоненте внутри props.phones. Нам осталость только написать разметку.

Давайте добавим стили к render функции

import React, {Component} from 'react'

import {connect} from 'react-redux'

import {fetchPhones} from 'actions'

import {getPhones} from 'selectors'

class Phones extends Component {

componentDidMount () {

this.props.fetchPhones()

}

render () {

const {phones} = this.props

return (

<div>

<div className='books row'>

{phones.map((phone, index) => this.renderPhone(phone, index))}

</div>

</div>

)

}

}

const mapStateToProps = state => ({

phones: getPhones(state)

})

const mapDispatchToProps = {

fetchPhones

}

export default connect(mapStateToProps, mapDispatchToProps)(Phones)

Также мы проходимся мапом по phones и вызываем для каждого метод renderPhones.

Теперь давайте его опишем. Для того, чтобы показывать только первые 60 символов из описания, мы можем использовать функцию Ramda take. Также где нужно проставляем ссылки Link из react-router, на страницу отдельного телефона, которую мы добавим в дальнейшем.

renderPhone (phone, index) {

const shortDescription = `${R.take(60, phone.description)}...`

return (

<div className='col-sm-4 col-lg-4 col-md-4 book-list' key={index}>

<div className='thumbnail'>

<img

className='img-thumbnail'

src={phone.image}

alt={phone.name}

/>

<div className='caption'>

<h4 className='pull-right'>${phone.price}</h4>

<h4>

<Link to={`/phones/${phone.id}`}>

{phone.name}

</Link>

</h4>

<p>{shortDescription}</p>

<p className='itemButton'>

<button

className='btn btn-primary'

>

Buy Now!

</button>

<Link

to={`/phones/${phone.id}`}

className='btn btn-default'

>

More info

</Link>

</p>

</div>

</div>

</div>

)

}

Если мы посмотрим в браузер, то у нас отрендерилась наша страница с товарами.

Если у вас что-то не получается, либо возникли вопросы и комментарии, пишите их прямо под этим видео и я обязательно на них отвечу.