Redux practice. Part 2

|  |  |  |  |
| --- | --- | --- | --- |
| № урока: | 4 | Курс: | React Advanced |
|  | | |  |
| Средства обучения: | | | Текстовый редактор или IDE, браузер, Node.js, терминал |

# Обзор, цель и назначение урока

В этом уроке мы продолжаем знакомство с Redux на практике. На этот раз мы начнем разрабатывать полноценный чат и в процессе узнаем о принципах Smart & Dumb компонентов в Redux, подключим ESLint в наш проект, поймем принцип работы HMR, подключим Redux DevTools и сделаем много других полезных мелочей и улучшений.

**Изучив материал данного занятия, учащийся:**

* Узнает о принципах построения полноценных Redux приложений
* Узнает принципы умных и глупых компонентов в Redux
* Закрепит знания касаемые Redux
* Узнает о различных инструментах, улучшающих качество разработки

# Содержание урока

1. Ретроспектива полученных знаний
2. Настройка ESLint
3. Подключение шаблона чата из codepen.
4. Smart & Dumb components.
5. Настройка алиасов в Webpack.
6. Настройка Hot Module Replacement.
7. Redux DevTool

# Резюме

ESLint – это инструмент, позволяющий проверять и валидировать JS код на этапе разработки, упрощая при этом нашу разработку.

ESLint, как и Babel, конфигурируется при помощи отдельного файлика в

корне проекта - .eslintrc.

В нем вы можете указать различные настройки, такие как парсер, различные плагины, параметры окружения и т.д. и непосредственно сами правила линтинга.

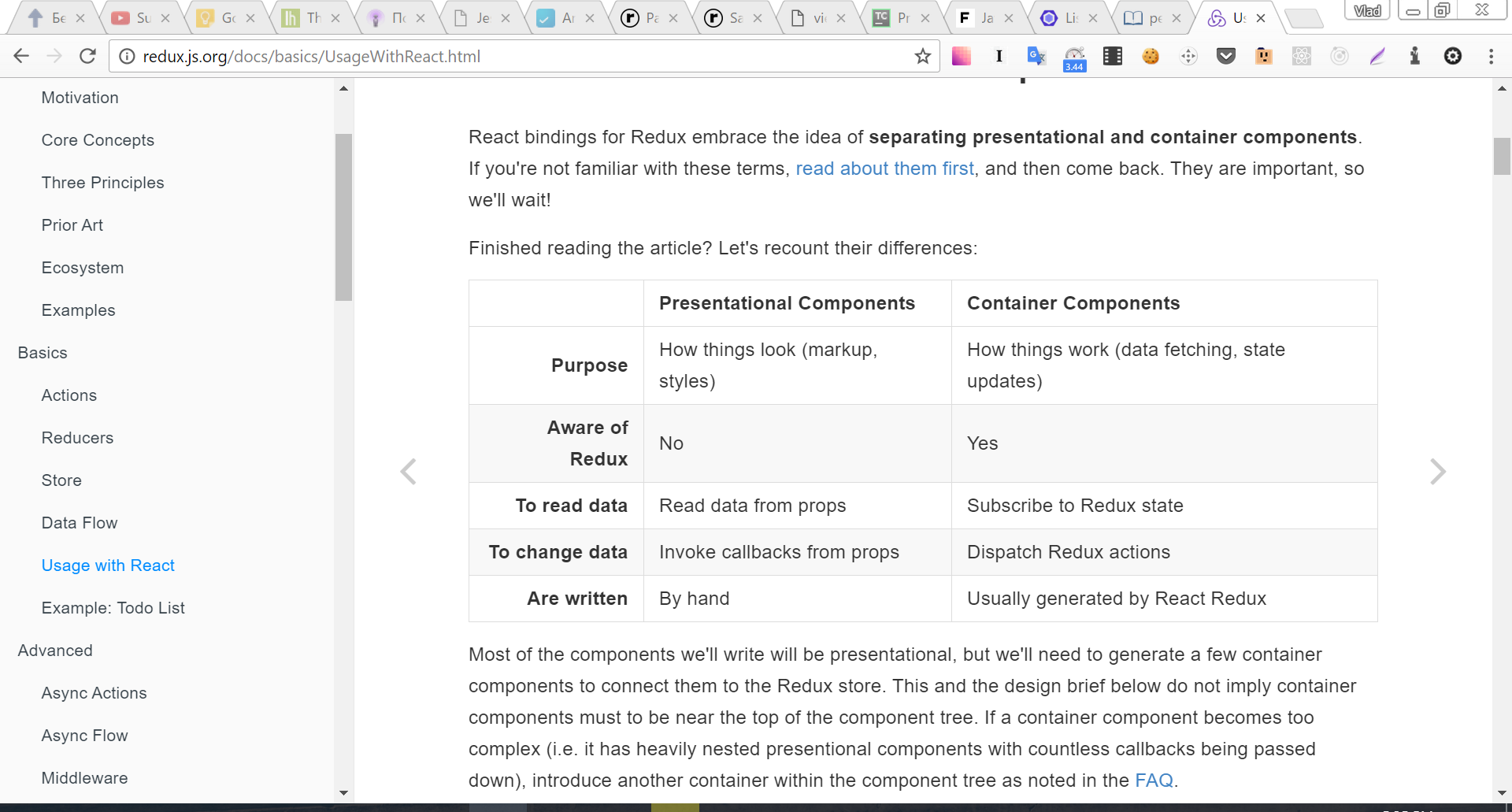
Вы можете использовать за основу рекомендуемые правила ESLint и дальше перезаписывать какие-то определенные, делается это при помощи значения "eslint:recommended" в параметре “extends”.

У каждого правила есть статус код (0 – nothing, 1 – warning, 2 – error), который сообщает как реагировать на это правило ESLint и также у каждого правила могут быть какие-либо опциональные параметры.

Правил для ESLint очень много, все вы их можете найти здесь - <http://eslint.org/docs/rules/>, плюс ко всему их кол-во может быть увеличено при помощи сторонних плагинов.

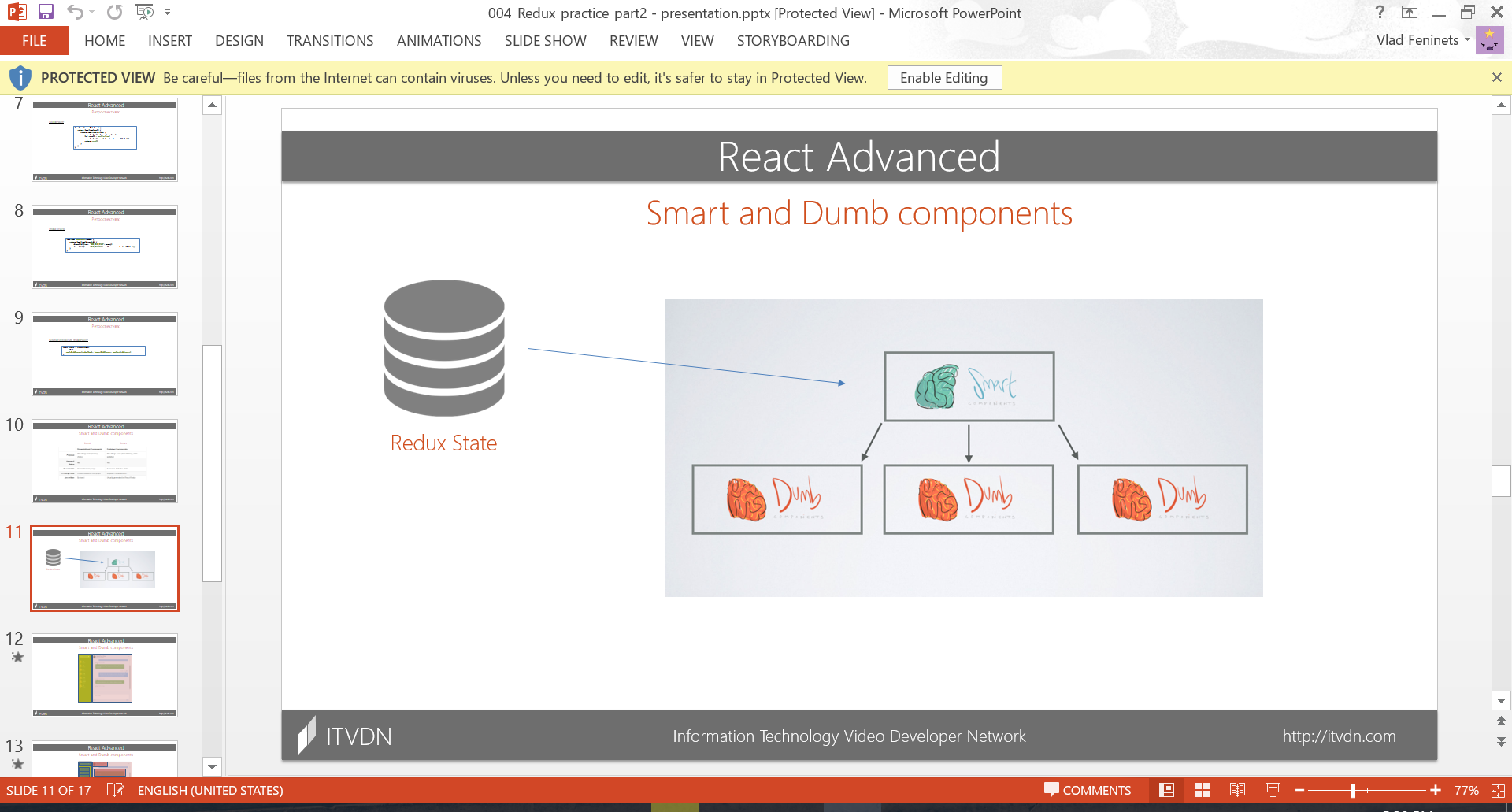
Поэтому комбинация правил сугубо ваш выбор. Мой совет – наследуйтесь от «рекомендуемых» правил и уже в процессе пробуйте их переопределять.

Вы уже наверняка знакомы с такими терминами как «Умные» и «Глупые» компоненты, в Redux они также есть, и вот в чем их отличия:



Резюмируя - все «умные» компоненты содержат минимум разметки и прокидывают данные из Redux State своим «детям», а в свою очередь эти «дети» не подозревая о существовании Redux просто получают props и callbacks.

И визуально это выглядит примерно так:



Для того, чтобы в JSX вместо «className» писать привычный «class» можно установить присет для babel под названием babel-plugin-react-html-attrs

(ссылка на гитхаб: <https://github.com/insin/babel-plugin-react-html-attrs>)

Hot Module Replacement в приложениях с использованием React/Redux позволяют забыть о клавише перезагрузки на вашей клавиатуре, HMR настраивается с помощью пакета react-hot-loader, автором которого является Ден Абрамов – создатель самого Redux.

На гитхаб-странице этого пакета есть отличная документация, включая Webpack v2.

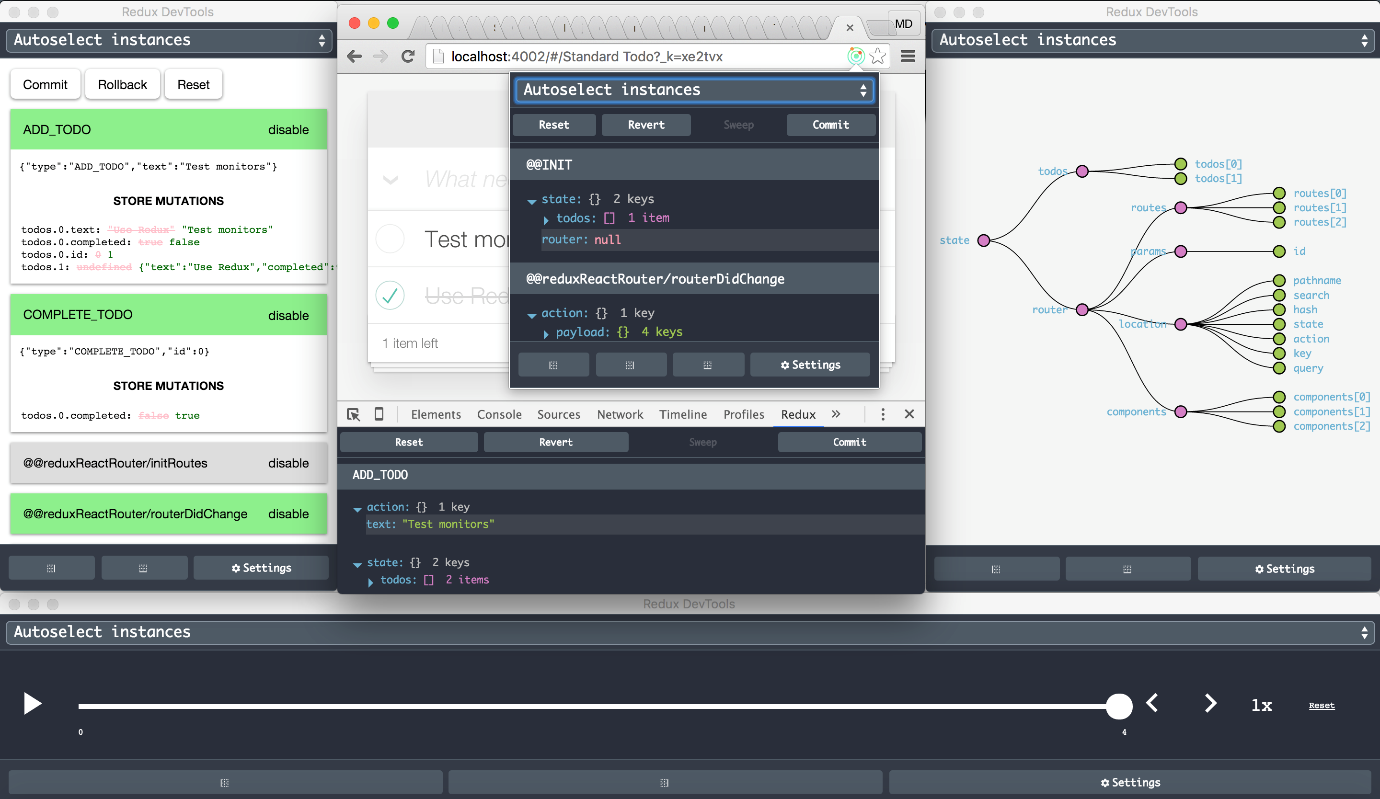
Если вкратце, то наш ReactDOM.render нужно вынести в отдельную функцию и вызывать ее как после инициализации, так и при каких-либо изменениях.

Делается это при помощи проверки hot.module.replacement, в теле которого мы вызываем метод hot.module.accept и в кач-ве аргументов передаем ему отслеживаемый компонент и callback, который будет вызван при его изменении.

Еще одним инструментом существенно упрощающим нашу с вами разработку является Redux DevTools (ссылка на репо - <https://github.com/zalmoxisus/redux-devtools-extension>)

Его можно подключать как в виде отдельного компонента, отдельного приложения на Electron и масса других вариантов, так и скачав расширение для конкретно вашего браузера. В репозитории есть README, в котором написано как подключить DevTool в каждом конкретном случае.

Визуально это вылядит вот так:



И этот полезный инструмент позволяет вам просматривать State вашего приложения, вызывать прям оттуда Actions, просматривать логи, «путешествовать во времени», использовать экспорт/импорт для последующего репродьюса и многое-многое другое.

Рекомендую вам поиграться с ним, на практике экономит очень много времени и сил.

# Закрепление материала

* Как настроить ESLint?
* Как сконфигурировать правила в ESLint?
* В чем различия Smart от Dumb компонента
* Почему мы удалили react-hmre плагин для Babel и для чего он нужен был?
* С помощью какого пакета можно настроить HMR в React/Redux?
* Каким функционалом обладает Redux DevTools?

# Рекомендуемые ресурсы

Репозиторий с примерами

<https://github.com/fnnzzz/react-advanced-itvdn>

Официальная документация Redux

<http://redux.js.org/>

ESLint

<http://eslint.org>

Присет для Babel, позволяющий использовать “class” в JSX

<https://github.com/insin/babel-plugin-react-html-attrs>

React Hot Loader for HMR

<https://github.com/gaearon/react-hot-loader>

Redux DevTools

<https://github.com/gaearon/redux-devtools>