Создание страницы ошибки

№ урока: 9 **Курс:** React Essential

Средства обучения: Браузер Chrome, редактор кода VS Code или любой другой.

Обзор, цель и назначение урока

Узнать, что такое Предохранители, React.Fragment, Строгий Режим в React. Научиться создавать страницу ошибки.

Изучив материал данного занятия, учащийся сможет

- Создавать ErrorBoundary компоненты.
- Использовать React.Fragment.
- Использовать Строгий Режим.

Содержание урока

1. Предохранители
2. React.Fragment
3. Строгий Режим

Резюме

- **Предохранители** — это компоненты React, которые отлавливают ошибки JavaScript в любом месте деревьев их дочерних компонентов, сохраняют их в журнале ошибок и выводят запасной UI, вместо рухнувшего дерева компонентов. Предохранители отлавливают ошибки при рендеринге, в методах жизненного цикла и конструкторах деревьев компонентов, расположенных под ними.
- **Предохранители не поймают ошибки в**:
    - обработчиках событий (подробнее);
    - асинхронном коде (например колбэках из setTimeout или requestAnimationFrame);
    - серверном рендеринге (Server-side rendering);
    - самом предохранителе (а не в его дочерних компонентах).
- **ErrorBoundary Component** - классовый компонент является предохранителем, если он включает хотя бы один из следующих методов жизненного цикла:

  static getDerivedStateFromError() или componentDidCatch().

  Используйте static getDerivedStateFromError() при рендеринге запасного UI, в случае отлова ошибки. Используйте componentDidCatch() при написании кода для журналирования информации об отловленной ошибке.
- Предохранители работают как JavaScript-блоки catch {}, но только для компонентов. Только классовые компоненты могут выступать в роли предохранителей. На практике чаще всего целесообразным будет один раз описать предохранитель и дальше использовать его по всему приложению.

  Обратите внимание, что **предохранители отлавливают ошибки исключительно в своих дочерних компонентах**. Предохранитель не сможет отловить ошибку внутри

CyberBionic Systematics ® 2020
19 Eugene Sverstyuk Str., 5 floor
Kyiv, Ukraine

Tel. 0 800 750 312
E-mail: edu@cbsystematics.com
Site: www.edu.cbsystematics.com | itvdn.com

самого себя. Если предохранителю не удаётся отрендерить сообщение об ошибке, то ошибка всплывает до ближайшего предохранителя, расположенного над ним в дереве компонентов. Этот аспект их поведения тоже напоминает работу блоков catch {} в JavaScript.

- **Где размещать предохранители** - степень охвата кода предохранителями остаётся на ваше усмотрение. Например, вы можете защитить им навигационные (route) компонента верхнего уровня, чтобы выводить пользователю сообщение «Что-то пошло не так», как это часто делают при обработке ошибок серверные фреймворки. Или вы можете охватить индивидуальными предохранителями отдельные виджеты, чтобы помешать им уронить всё приложение.
- Начиная с React 16, **ошибки, не отловленные** ни одним из предохранителей, будут приводить к размонтированию всего дерева компонентов React. Хотя принятие этого решения и вызвало споры, судя по нашему опыту, бóльшим злом будет вывести некорректный UI, чем удалить его целиком. К примеру, в приложении типа Messenger, вывод поломанного UI может привести к тому, что пользователь отправит сообщение не тому адресату.
- **Стек вызовов компонентов -** в режиме разработки React 16 выводит на консоль сообщения обо всех ошибках, возникших при рендеринге, даже если они никак не сказались на работе приложения. Помимо сообщения об ошибке и стека JavaScript, React 16 также выводит и стек вызовов компонентов. Кроме этого, в стеке вызовов компонентов выводятся имена файлов и номера строк. Такое поведение по умолчанию настроено в проектах, созданных при помощи Create React App.

- Предохранители не отлавливают ошибки, произошедшие в обработчиках событий. React не нуждается в предохранителях, чтобы корректно обработать ошибки в обработчиках событий. В отличие от метода render и методов жизненного цикла, обработчики событий не выполняются во время рендеринга. Таким образом, даже если они сгенерируют ошибку, React всё равно знает, что нужно выводить на экран. Чтобы отловить ошибку в обработчике событий, пользуйтесь обычной JavaScript-конструкцией try / catch.

- **Фрагменты (React.Fragment)** — позволяют формировать список дочерних элементов, не создавая лишних узлов в DOM. Существует сокращённая запись объявления фрагментов. Она выглядит как пустые теги. Фрагменты, объявленные с помощью <React.Fragment>, могут иметь ключи. Например, их можно использовать при создании списка определений, преобразовав коллекцию в массив фрагментов. key — это единственный атрибут, допустимый у Fragment.
- **StrictMode** — инструмент для обнаружения потенциальных проблем в приложении. Также, как и Fragment, StrictMode не рендерит видимого UI. Строгий режим активирует дополнительные проверки и предупреждения для своих потомков. Проверки строгого режима работают только в режиме разработки, они не оказывают никакого эффекта в продакшен-сборке.
- **На данный момент StrictMode помогает в**:
    - Обнаружении небезопасных методов жизненного цикла.
    - Предупреждении об использовании устаревшего API строковых реф.
    - Предупреждении об использовании устаревшего метода findDOMNode.
    - Обнаружении неожиданных побочных эффектов.
    - Обнаружении устаревшего API контекста.

CyberBionic systematics

CyberBionic Systematics ® 2020
19 Eugene Sverstyuk Str., 5 floor
Kyiv, Ukraine

Tel. 0 800 750 312
E-mail: edu@cbsystematics.com
Site: www.edu.cbsystematics.com | itvdn.com

Закрепление материала

- Где размещать предохранители?
- Зачем нужен React.Fragment?
- В какие 2 этапа работает React?

Дополнительное задание

Задание
Прочитать документацию React, а именно - раздел «Строгий режим». Узнать в какие 2 этапа работает React (render phase, commit phase).

Самостоятельная деятельность учащегося

Задание 1
Выучите основные понятия, рассмотренные на уроке.

Задание 2
Создайте ErrorBoundary компонент, который будет отображать надпись «something went wrong», если в дочернем компоненте произошла ошибка.

Задание 3
Создайте Error Page (Component) - который будет оборачивать все роуты приложения.

Рекомендуемые ресурсы

Предохранители
Фрагменты
Строгий режим

CyberBionic systematics
CyberBionic Systematics ® 2020
19 Eugene Sverstyuk Str., 5 floor
Kyiv, Ukraine
Tel. 0 800 750 312
E-mail: edu@cbsystematics.com
Site: www.edu.cbsystematics.com | itvdn.com