React 最佳实践入门

自我介绍

莫涛 妙味课堂联合创始人，10年老前端 目前，在开课吧负责 Web 高级工程师课程

课程说明

本次公开课节选自《WEB 高级工程师》课程，React 部分 课程链接：
https://mp.weixin.qq.com/s/2VMD2kmuVskce-EkBlJUxA

前置基础

- 有一定 JS 基础，熟悉 DOM, BOM, EVENT的基本操作和概念
- 熟悉 ES6 相关新增语法

课程目标

- 快速上手 React 基本操作
- 掌握 React 新增 API hooks 的相关使用
- 基于 Hooks 实现 TodoList

特别说明

本次公开课旨在帮助大家快速了解 react 的上手使用，在讲解具体知识点时，不会特别深入详细，有系统学习 React 需求的同学可以参考我们的大课

据说今天报名会有优惠呀！！！

课程内容

React 是什么？

一个用于构建用户界面的 JavaScript 库 中文手册：https://react.docschina.org/

React 的优势

- 声明式编程可以让代码更加可靠，且方便调试。
- 组件化提升项目的通用性，降低代码的逻辑难度。
- MVVM（Model-View-ViewModel） 的架构方式，节省我们优化 DOM 操作的工作，提升项目性能

命令式编程 和 声明式编程

- 告诉计算机怎么做（How） - 过程
- 告诉计算机我们要什么（What） - 结果

如何使用 React

基于浏览器的模式 -- (学习的过程使用)

- React.js 提供 React.js 核心功能代码，如：虚拟 dom，组件
  - React.createElement(type,props,children);
- ReactDOM 提供了与浏览器交互的 DOM 功能，如：dom 渲染
  - ReactDOM.render(container, element[, callback])
    - element：要渲染的内容
    - container：要渲染的内容存放容器
    - callback：渲染后的回调函数

babel

babel-standalone.js：在浏览器中处理 JSX https://cdn.bootcss.com/babel-standalone/6.26.0/babel.min.js

JSX

JSX 是一个基于 JavaScript + XML 的一个扩展语法 - 它可以作为值使用 - 它并不是字符串 - 它也不是HTML - 它可以配合JavaScript 表达式一起使用

插值表达式

在 JXS 中可以使用 {表达式} 嵌入表达式

表达式：产生值的一组代码的集合

- 变量
- 算术运算
- 函数调用
- ……

注意：分清楚 表达式 与 语句 的区别，if、for、while 这些都是语句，JSX 不支持语句

各种类型内容在插值中的使用

- 注释 {/*注释*/} {/* 多行注释 */}

输出数据类型

- 字符串、数字：原样输出
- 布尔值、空、未定义 会被忽略

列表渲染

- 数组
- 对象 扩展：虚拟 DOM （virtualDOM） 和 diff

条件渲染

- 三元运算符
- 与或运算符

在属性上使用 表达式

JSX 中的表达式也可以使用在属性上，但是使用的时候需要注意

- 当在属性中使用 {} 的时候，不要使用引号包含

JSX 使用注意事项

- 必须有,且只有一个顶层的包含元素
- JSX 不是html，很多属性在编写时不一样
  - className
  - style
- 列表渲染时，必须有 key 值
- 在 jsx 所有标签必须闭合
- 组件的首字母一定大写，标签一定要小写

组件

对具有一定独立功能的数据与方法的封装，对外暴露接口，有利于代码功能的复用，且不用担心冲突问题。

示例

样式

```
.friend-list {
    border: 1px solid #000000;
    width: 200px;
}
.friend-group dt {
    padding: 10px;
    background-color: rgb(64, 158, 255);
    font-weight: bold;
}
.friend-group dd {
    padding: 10px;
    display: none;
}
.friend-group.expanded dd {
    display: block;
}
.friend-group dd.checked {
    background: green;
}
```

创建 FriendList 组件

```
function FriendList() {
    return (
        <div className="friend-list">
```

```
            <dl className="friend-group">
                <dt>家人</dt>
                <dd>爸爸</dd>
                <dd>妈妈</dd>
            </dl>
            <dl className="friend-group">
                <dt>朋友</dt>
                <dd>张三</dd>
                <dd>李四</dd>
                <dd>王五</dd>
            </dl>
            <dl className="friend-group">
                <dt>客户</dt>
                <dd>阿里</dd>
                <dd>腾讯</dd>
                <dd>头条</dd>
            </dl>
        </div>
    );
}
```

组件复用 - 数据抽取

为了提高组件的复用性，通常会把组件中的一些可变数据提取出来

```
let datas = {
    family: {
        title: '家人',
        list: [
            {name: '爸爸'},
            {name: '妈妈'}
        ]
    },
    friend: {
        title: '朋友',
        list: [
            {name: '张三'},
            {name: '李四'},
            {name: '王五'}
        ]
    },
    customer: {
        title: '客户',
        list: [
            {name: '阿里'},
            {name: '腾讯'},
            {name: '头条'}
        ]
    }
};
```

函数式组件

- 函数的名称就是组件的名称
- 函数的返回值就是组件要渲染的内容

React 组件间通信

在 React.js 中，数据是从上自下流动（传递）的，也就是一个父组件可以把它的 state / props 通过 props 传递给它的子组件，但是子组件不能修改 props - React.js 是单向数据流，如果子组件需要修改父组件状态（数据），是通过回调函数方式来完成的。

- 父级向子级通信 把数据添加子组件的属性中，然后子组件中从props属性中，获取父级传递过来的数据
- 子级向父级通信 在父级中定义相关的数据操作方法(或其他回调), 把该方法传递给子级，在子级中调用该方法父级传递消息

React hooks

React hooks 是React 16.8中的新增功能。它们使您无需编写类即可使用状态和其他React功能 参考：https://reactjs.org/docs/hooks-intro.html

React Hooks 优势

- 简化组件逻辑
- 复用状态逻辑
- 无需使用类组件编写

常用 **hook**

- useState
  ```
  const [state, setState] = useState(initialState);
  ```
- useEffect 类组件 componentDidMount、componentDidUpdate 和 componentWillUnmount 需要清除的副作用
- useRef

Hook 使用规则

- 只在最顶层使用 Hook
- 只在 React 函数中调用 Hook
  - React 函数组件中
  - React Hook 中

资料及视频回放获取，明天都会发给顾问小姐姐，找顾问小姐姐获取相应的资料

更多公开课信息，关注顾问小姐姐的微信朋友圈 和 开课吧公众号