React初识

9.3 比较好的使用场景

1. React是什么?

React 是一个用于构建用户界面的 JAVASCRIPT 库。

React 主要用于构建UI，很多人认为 React 是 MVC 中的 V（视图）。

React 起源于 Facebook 的内部项目，用来架设 Instagram 的网站，并于 2013 年 5 月开源。

React 拥有较高的性能，代码逻辑非常简单，越来越多的人已开始关注和使用它。

---

1. React 特点

1. **声明式设计** —React采用声明范式，可以轻松描述应用。
2. **高效** —React通过对DOM的模拟，最大限度地减少与DOM的交互。
3. **灵活** —React可以与已知的库或框架很好地配合。
4. **JSX** — JSX 是 JavaScript 语法的扩展。React 开发不一定使用 JSX ，但我们建议使用它。
5. **组件** — 通过 React 构建组件，使得代码更加容易得到复用，能够很好的应用在大项目的开发中。
6. **单向响应的数据流** — React 实现了单向响应的数据流，从而减少了重复代码，这也是它为什么比传统数据绑定更简单。

2. 与其他框架的区别

1. vue2和react的区别
2. Vue2与React两个框架的区别和优势对比
3. Vue2对比其他框架
4. Vue3 究竟好在哪里？（和 React Hook 的详细对比）

2. React 安装

1. 通过script引入

```JavaScript
// 引入react核心库
<script
src="https://cdn.staticfile.org/react/16.4.0/umd/react.development.js">
</script>
// 引入react-dom，用于支持react操作DOM
<script src="https://cdn.staticfile.org/react-dom/16.4.0/umd/react-
dom.development.js"></script>
// 引入babel，用于将jsx转为js
```

2. 使用 react脚手架: create-react-app

```Shell
$ npm install -g create-react-app
$ create-react-app my-app
$ cd my-app/
$ npm start
```

使用脚手架创建的项目技术架构为: react + webpack + es6 + eslint

下面是react脚手架项目结构

```Plain Text
public ---- 静态资源文件夹
    favicon.icon ------ 网站页签图标
    index.html -------- 主页面
    logo192.png ------- logo图
    logo512.png ------- logo图
    manifest.json ----- 应用加壳的配置文件
    robots.txt -------- 爬虫协议文件
src ---- 源码文件夹
    App.css -------- App组件的样式
    App.js --------- App组件
    App.test.js ---- 用于给App做测试
    index.css ------ 样式
    index.js ------- 入口文件
    logo.svg ------- logo图
    reportWebVitals.js
      --- 页面性能分析文件(需要web-vitals库的支持)
    setupTests.js
      ---- 组件单元测试的文件(需要jest-dom库的支持)
```

3. React基础

1. VDOM

1.1 创建虚拟DOM的两种方式

1. 纯js方式(一般不用)
2. JSX方式

1.2 虚拟DOM与真实DOM

1. React提供了一些API来创建一些**特别的一般js对象**; **React.createElement(component, props, ...children)**
2. 虚拟DOM比较“轻”，真实DOM比较“重”，因为虚拟DOM是React内部在用，无需真实DOM上那么多的属性。
3. 虚拟DOM最终会被React转化为真实DOM，呈现在页面上
4. 编码时基本只需要操作虚拟DOM的相关数据, React会根据diff算法修改真实DOM, 更新UI

2. JSX

2.1 什么是JSX

1. 全称：JavaScript XML
2. react定义的一种类似于XML的JS扩展语法: JS + XML; 本质是**React.createElement(component, props, ...children)**方法的语法糖
3. 作用: 用来简化创建虚拟DOM

2.2 JSX语法规则

1. 只能有一个根标签
2. 标签名任意: HTML标签或自定义组件
3. 标签属性任意: HTML标签属性或自定义属性
4. 样式的类名指定要用className
5. 内联样式，要用style={{ key: value }}的形式去写
6. 标签中混入JS表达式时要用{}
7. ......

3. 组件

3.1 函数式组件

使用函数创建的**无状态函数式组件**接收带有唯一数据的**props**属性, 它除了可读性更好和编写简单外, 还有以下特点:

1. 组件不会被实例化，整体渲染性能得到提升
2. 组件不能访问this对象
3. 组件无法访问生命周期的方法

4. 无状态组件只能访问输入的props，同样的props会得到同样的渲染结果，不会有副作用

注意: React 16.8新增的Hooks, 可以让你在不编写 class 的情况下使用 state 以及其他的 React 特性

3.2 类式组件

类组件用于复杂组件的编写, 可使用this, 生命周期, state, refs等特性

4. 事件处理

4.1 React事件处理VS原生Dom事件处理

1. React 事件绑定属性的命名采用驼峰式写法（如：onClick,onKeyUp），而不全是小写字母。
2. 如果采用 JSX 的语法你需要传入一个函数作为事件处理函数，而不是一个字符串(DOM 元素的写法)。
3. 另外在React当中，return false不会阻止事件的默认行为，需要调用 e.preventDefault()。

4.2 绑定事件处理函数的this问题

在以类继承的方法定义的组件中，事件处理函数的this指向的并不是当前组件。因为类中的方法默认开启了局部的严格模式，此时this指向undefined。

4.3 绑定事件处理函数的this到当前组件

1. 通过bind方法进行原地绑定，从而改变this指向
2. 通过箭头函数
3. 在constructor中提前对事件进行绑定
4. 将事件的写法改为箭头函数的形式

4.4 向事件处理函数传递参数

1. 通过箭头函数
2. 通过bind

5. state

1. React把每一个有状态的组件都看成是一个状态机，组件内部通过state来维护组件状态的变化。
2. 在事件中触发setState()来修改state数据，state改变后会重新进行render()

6. props

1. React中的数据流是自上而下，从父组件流向子组件。
2. 子组件从父组件提供的props中获取数据，并进行渲染，一般是纯展示的组件。

3. 如果父组件的props更新，则该组件下面所有用到这个属性的子组件，都会重新进行render()
4. **props是只读的**。不要试图修改props，否则会报错。
5. propTypes可以进行类型检查

7. refs

1. 可以用来访问在render()中的Dom节点

8. 表单-受控组件和非受控组件

1. 不是一种新的组件 而是获取表单数据的2种方式
2. 非受控组件 通过ref获取表单数据, 输入框内部的值是用户控制，和React无关
3. 受控组件 将表单value 和 state 里的数据进行关联, 可以通过 onChange 事件控制用户输入，使用正则表达式过滤不合理输入

9. 组件通信方式总结

9.1 组件间的关系

1. 父子组件
2. 兄弟组件（非嵌套组件）
3. 祖孙组件（跨级组件）

9.2 几种通信方式

1. props
2. 消息订阅-发布模式: pubs-sub.js, events.js等
3. 集中式管理: redux, dva等
4. 生产者-消费者模式: conText

9.3 比较好的使用场景

1. 父子组件: props
2. 兄弟组件: 消息订阅-发布模式, 集中式管理
3. 祖孙组件(跨级组件): 消息订阅-发布模式, 集中式管理, context

参考链接:

React官方文档

React 教程 – 菜鸟教程

尚硅谷2021版React技术全家桶全套完整版 – bilibili
React如何处理事件 – 简书
小结React(三)：state、props、Refs – 腾讯云
react的受控组件和非受控组件 – 简书
react–受控和非受控组件 – CSDN
react组件通信方式汇总 – CSDN