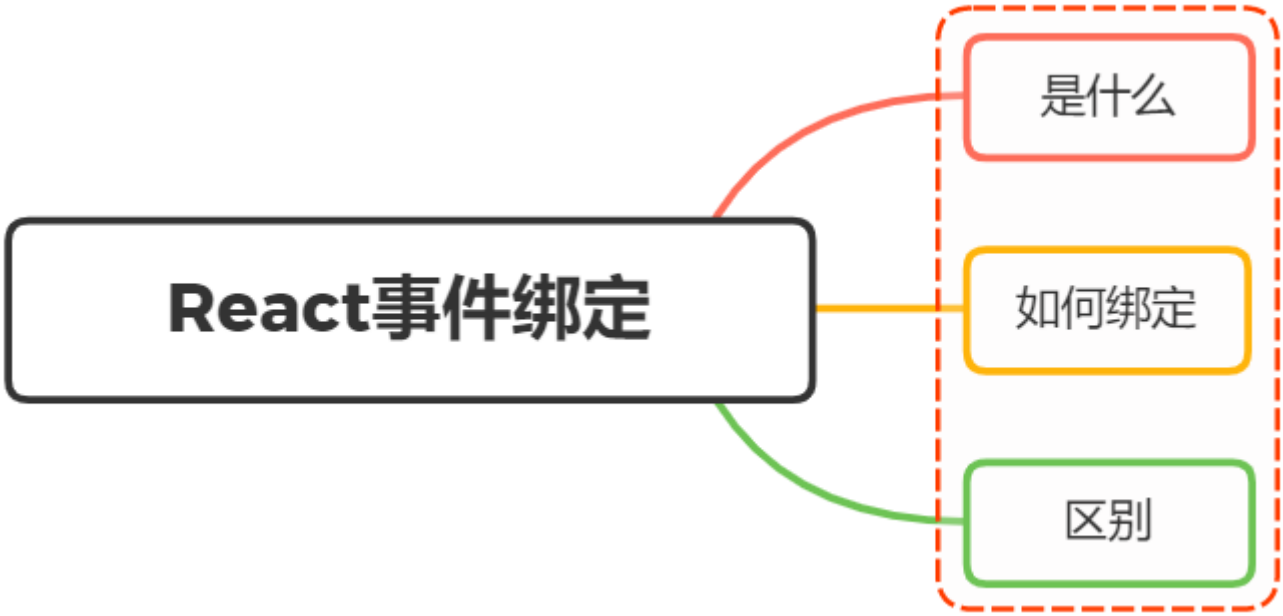
面试官：React事件绑定的方式有哪些？区别？



一、是什么

在react应用中，事件名都是用小驼峰格式进行书写，例如onclick要改写成onClick

最简单的事件绑定如下：

```
class ShowAlert extends React.Component {
  showAlert() {
    console.log("Hi");
  }

  render() {
    return <button onClick={this.showAlert}>show</button>;
  }
}
```

从上面可以看到，事件绑定的方法需要使用{}包住

上述的代码看似没有问题，但是当将处理函数输出代码换成console.log(this)的时候，点击按钮，则会发现控制台输出undefined

二、如何绑定

为了解决上面正确输出this的问题，常见的绑定方式有如下：

- render方法中使用bind
- render方法中使用箭头函数
- constructor中bind

- 定义阶段使用箭头函数绑定

render方法中使用bind

如果使用一个类组件，在其中给某个组件/元素一个`onClick`属性，它现在并会自定绑定其`this`到当前组件，解决这个问题的方法是在事件函数后使用`.bind(this)`将`this`绑定到当前组件中

```
class App extends React.Component {
  handleClick() {
    console.log('this > ', this);
  }
  render() {
    return (
      <div onClick={this.handleClick.bind(this)}>test</div>
    )
  }
}
```

这种方式在组件每次`render`渲染的时候，都会重新进行`bind`的操作，影响性能

render方法中使用箭头函数

通过`ES6`的上下文来将`this`的指向绑定给当前组件，同样再每一次`render`的时候都会生成新的方法，影响性能

```
class App extends React.Component {
  handleClick() {
    console.log('this > ', this);
  }
  render() {
    return (
      <div onClick={e => this.handleClick(e)}>test</div>
    )
  }
}
```

constructor中bind

在`constructor`中预先`bind`当前组件，可以避免在`render`操作中重复绑定

```
class App extends React.Component {
  constructor(props) {
    super(props);
    this.handleClick = this.handleClick.bind(this);
  }
  handleClick() {
    console.log('this > ', this);
  }
  render() {
```

```
    return (
      <div onClick={this.handleClick}>test</div>
    )
  }
}
```

定义阶段使用箭头函数绑定

跟上述方式三一样，能够避免在render操作中重复绑定，实现也非常的简单，如下：

```
class App extends React.Component {
  constructor(props) {
    super(props);
  }
  handleClick = () => {
    console.log('this > ', this);
  }
  render() {
    return (
      <div onClick={this.handleClick}>test</div>
    )
  }
}
```

三、区别

上述四种方法的方式，区别主要如下：

- 编写方面：方式一、方式二写法简单，方式三的编写过于冗杂
- 性能方面：方式一和方式二在每次组件render的时候都会生成新的方法实例，性能问题欠缺。若该函数作为属性值传给子组件的时候，都会导致额外的渲染。而方式三、方式四只会生成一个方法实例

综合上述，方式四是最优的事件绑定方式

参考文献

- https://segmentfault.com/a/1190000011317515
- https://vue3js.cn/interview/