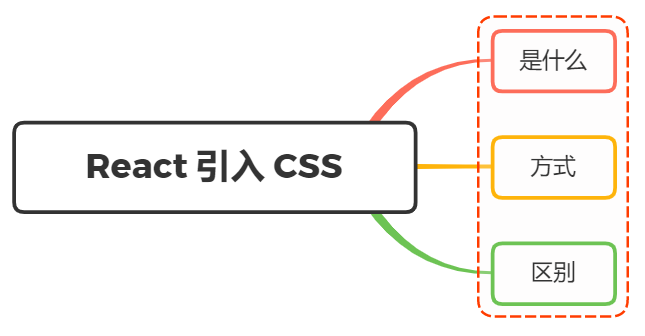
# 面试官：说说react中引入css的方式有哪几种？区别？



## 一、是什么

组件式开发选择合适的css解决方案尤为重要

通常会遵循以下规则：

* 可以编写局部css，不会随意污染其他组件内的原生；
* 可以编写动态的css，可以获取当前组件的一些状态，根据状态的变化生成不同的css样式；
* 支持所有的css特性：伪类、动画、媒体查询等；
* 编写起来简洁方便、最好符合一贯的css风格特点

在这一方面，vue使用css起来更为简洁：

* 通过 style 标签编写样式
* scoped 属性决定编写的样式是否局部有效
* lang 属性设置预处理器
* 内联样式风格的方式来根据最新状态设置和改变css

而在react中，引入CSS就不如Vue方便简洁，其引入css的方式有很多种，各有利弊

## 二、方式

常见的CSS引入方式有以下：

* 在组件内直接使用
* 组件中引入 .css 文件
* 组件中引入 .module.css 文件
* CSS in JS

### 在组件内直接使用

直接在组件中书写css样式，通过style属性直接引入，如下：

import React, { Component } from "react";  
  
const div1 = {  
 width: "300px",  
 margin: "30px auto",  
 backgroundColor: "#44014C", //驼峰法  
 minHeight: "200px",  
 boxSizing: "border-box"  
};  
  
class Test extends Component {  
 constructor(props, context) {  
 super(props);  
 }  
   
 render() {  
 return (  
 <div>  
 <div style={div1}>123</div>  
 <div style={{backgroundColor:"red"}}>  
 </div>  
 );  
 }  
}  
  
export default Test;

上面可以看到，css属性需要转换成驼峰写法

这种方式优点：

* 内联样式, 样式之间不会有冲突
* 可以动态获取当前state中的状态

缺点：

* 写法上都需要使用驼峰标识
* 某些样式没有提示
* 大量的样式, 代码混乱
* 某些样式无法编写(比如伪类/伪元素)

### 组件中引入css文件

将css单独写在一个css文件中，然后在组件中直接引入

App.css文件：

.title {  
 color: red;  
 font-size: 20px;  
}  
  
.desc {  
 color: green;  
 text-decoration: underline;  
}

组件中引入：

import React, { PureComponent } from 'react';  
  
import Home from './Home';  
  
import './App.css';  
  
export default class App extends PureComponent {  
 render() {  
 return (  
 <div className="app">  
 <h2 className="title">我是App的标题</h2>  
 <p className="desc">我是App中的一段文字描述</p >  
 <Home/>  
 </div>  
 )  
 }  
}

这种方式存在不好的地方在于样式是全局生效，样式之间会互相影响

### 组件中引入 .module.css 文件

将css文件作为一个模块引入，这个模块中的所有css，只作用于当前组件。不会影响当前组件的后代组件

这种方式是webpack特工的方案，只需要配置webpack配置文件中modules:true即可

import React, { PureComponent } from 'react';  
  
import Home from './Home';  
  
import './App.module.css';  
  
export default class App extends PureComponent {  
 render() {  
 return (  
 <div className="app">  
 <h2 className="title">我是App的标题</h2>  
 <p className="desc">我是App中的一段文字描述</p >  
 <Home/>  
 </div>  
 )  
 }  
}

这种方式能够解决局部作用域问题，但也有一定的缺陷：

* 引用的类名，不能使用连接符(.xxx-xx)，在 JavaScript 中是不识别的
* 所有的 className 都必须使用 {style.className} 的形式来编写
* 不方便动态来修改某些样式，依然需要使用内联样式的方式；

### CSS in JS

CSS-in-JS， 是指一种模式，其中CSS由 JavaScript生成而不是在外部文件中定义

此功能并不是 React 的一部分，而是由第三方库提供，例如：

* styled-components
* emotion
* glamorous

下面主要看看styled-components的基本使用

本质是通过函数的调用，最终创建出一个组件：

* 这个组件会被自动添加上一个不重复的class
* styled-components会给该class添加相关的样式

基本使用如下：

创建一个style.js文件用于存放样式组件：

export const SelfLink = styled.div`  
 height: 50px;  
 border: 1px solid red;  
 color: yellow;  
`;  
  
export const SelfButton = styled.div`  
 height: 150px;  
 width: 150px;  
 color: ${props => props.color};  
 background-image: url(${props => props.src});  
 background-size: 150px 150px;  
`;

引入样式组件也很简单：

import React, { Component } from "react";  
  
import { SelfLink, SelfButton } from "./style";  
  
class Test extends Component {  
 constructor(props, context) {  
 super(props);  
 }   
   
 render() {  
 return (  
 <div>  
 <SelfLink title="People's Republic of China">app.js</SelfLink>  
 <SelfButton color="palevioletred" style={{ color: "pink" }} src={fist}>  
 SelfButton  
 </SelfButton>  
 </div>  
 );  
 }  
}  
  
export default Test;

## 三、区别

通过上面四种样式的引入，可以看到：

* 在组件内直接使用css该方式编写方便，容易能够根据状态修改样式属性，但是大量的演示编写容易导致代码混乱
* 组件中引入 .css 文件符合我们日常的编写习惯，但是作用域是全局的，样式之间会层叠
* 引入.module.css 文件能够解决局部作用域问题，但是不方便动态修改样式，需要使用内联的方式进行样式的编写
* 通过css in js 这种方法，可以满足大部分场景的应用，可以类似于预处理器一样样式嵌套、定义、修改状态等

至于使用react用哪种方案引入css，并没有一个绝对的答案，可以根据各自情况选择合适的方案

## 参考文献

* https://zh-hans.reactjs.org/docs/faq-styling.html#gatsby-focus-wrapper
* https://mp.weixin.qq.com/s/oywTpNKEikMXn8QTBgITow