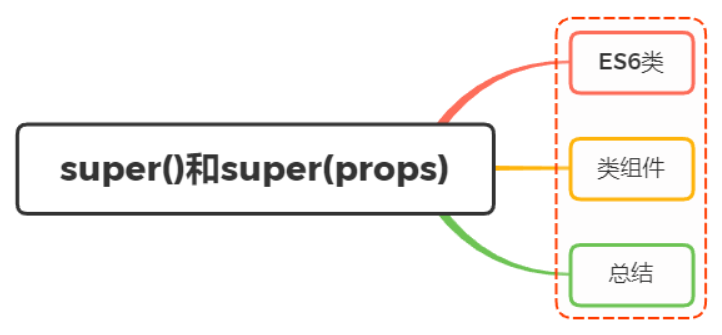
# 面试官：super() 和 super(props) 有什么区别？



## 一、ES6 类

在 ES6 中，通过 extends 关键字实现类的继承，方式如下：

class sup {  
 constructor(name) {  
 this.name = name;  
 }  
  
 printName() {  
 console.log(this.name);  
 }  
}  
  
class sub extends sup {  
 constructor(name, age) {  
 super(name); // super代表的事父类的构造函数  
 this.age = age;  
 }  
  
 printAge() {  
 console.log(this.age);  
 }  
}  
  
let jack = new sub("jack", 20);  
jack.printName(); //输出 : jack  
jack.printAge(); //输出 : 20

在上面的例子中，可以看到通过 super 关键字实现调用父类，super 代替的是父类的构建函数，使用 super(name) 相当于调用 sup.prototype.constructor.call(this,name)

如果在子类中不使用 super，关键字，则会引发报错，如下：



报错的原因是 子类是没有自己的 this 对象的，它只能继承父类的 this 对象，然后对其进行加工

而 super() 就是将父类中的 this 对象继承给子类的，没有 super() 子类就得不到 this 对象

如果先调用 this，再初始化 super()，同样是禁止的行为

class sub extends sup {  
 constructor(name, age) {  
 this.age = age;  
 super(name); // super代表的事父类的构造函数  
 }  
}

所以在子类 constructor 中，必须先代用 super 才能引用 this

## 二、类组件

在 React 中，类组件是基于 ES6 的规范实现的，继承 React.Component，因此如果用到 constructor 就必须写 super() 才初始化 this

这时候，在调用 super() 的时候，我们一般都需要传入 props 作为参数，如果不传进去，React 内部也会将其定义在组件实例中

// React 内部  
const instance = new YourComponent(props);  
instance.props = props;

所以无论有没有 constructor，在 render 中 this.props 都是可以使用的，这是 React 自动附带的，是可以不写的：

class HelloMessage extends React.Component {  
 render() {  
 return <div>nice to meet you! {this.props.name}</div>;  
 }  
}

但是也不建议使用 super() 代替 super(props)

因为在 React 会在类组件构造函数生成实例后再给 this.props 赋值，所以在不传递 props 在 super 的情况下，调用 this.props 为 undefined，如下情况：

class Button extends React.Component {  
 constructor(props) {  
 super(); // 没传入 props  
 console.log(props); // {}  
 console.log(this.props); // undefined  
 // ...  
 }  
}

而传入 props 的则都能正常访问，确保了 this.props 在构造函数执行完毕之前已被赋值，更符合逻辑，如下：

class Button extends React.Component {  
 constructor(props) {  
 super(props); // 没传入 props  
 console.log(props); // {}  
 console.log(this.props); // {}  
 // ...  
 }  
}

## 三、总结

在 React 中，类组件基于 ES6，所以在 constructor 中必须使用 super

在调用 super 过程，无论是否传入 props，React 内部都会将 porps 赋值给组件实例 porps 属性中

如果只调用了 super()，那么 this.props 在 super() 和构造函数结束之间仍是 undefined

## 参考文献

* <https://overreacted.io/zh-hans/why-do-we-write-super-props/>
* <https://segmentfault.com/q/1010000008340434>