什么时候不能使用箭头函数

1. 箭头函数有什么缺点

- 没有 `arguments` 对象;
- 没有自己的 this;
- 无法通过 `apply`, `call`, `bind` 改变 `this`。
- 不能作为构造函数（用 new 调用会报错）；

```
const fn1 = () => {
    console.log("arguments", arguments);
};
fn1(100, 200);
// ReferenceError: arguments is not defined

function fn2() {
    console.log("arguments", arguments);
}
fn2(100, 200);  // Arguments {0: 100, 1: 200, length: 2}
```

箭头函数中的 this 是父作用域中的 this。

```
const fn = () => {
    console.log("this", this);
}
// Window
// 箭头函数中的 this 和父作用域中的 this 一样，相当于在函数外获取 this，就是 window 对象
```

以 call 方法为例，无法通过箭头函数改变 this 指向。普通函数可以。

```
const fn1 = () => {
    console.log("this", this);
}
fn1.call({a: 10});
// this Window
```

```
const fn2 = function() {
    console.log("this", this);
}
fn2.call({a: 10});
// 或 fn2.apply({a: 10});
```

```
// 或 const f2 = fn2.bind({a: 10}); f2();
// this {a: 10}
```

作为构造函数会报错。

```
function Fn3() {
    console.log('a');
};
const f3 = new Fn3();  // a

const Fn4 = () => {
    console.log('a');
};
const f4 = new Fn4(); // ReferenceError: Fn4 is not defined
```

2. 什么时候不能使用箭头函数

- 对象方法
- 原型方法
- 构造函数
- 动态上下文中的回调函数
- Vue 生命周期函数和 method 使用箭头函数，函数里面获取不到想要的 this, 因为生命周期函数和 method 方法在一个对象中，Vue 组件本质上是对象，里面的的方法是对象方法。
- React 可以，React 组件本质上是 class。

1. 对象方法不能使用箭头函数来获取 this。

```
// 普通函数可以获取到 this
const obj1 = {
    name: "zhangsan",
    getName: function() {
        console.log("name", this.name)
    }
}
obj1.getName();  // name zhangsan

// 箭头函数中获取不到 this
const obj2 = {
    name: "zhangsan",
    getName: () => {
        console.log("name", this.name)
    }
}
obj2.getName();  // name
// this.name 是空的，什么都没有，因为 this 指向父作用域的 window，window 中没有定义 name 变量
```

2. 原型方法不能使用箭头函数来获取 this。

```
const obj1 = {
    name: "zhangsan"
};
obj1.__proto__.getName = () => {
    console.log(this.name);
}
obj1.getName();  //
// 上面语句执行什么都不打印
```

3. 在构造函数中不能使用箭头函数，因为箭头函数没有自己的 `this` 上下文

```
const Fn = (name, age) => {
    this.name = name;
    this.age = age;
}
const fn = new Fn("zhangsan", "shanghai");  // TypeError: Fn is not a constructor
```

4. 动态上下文中的回调函数不能使用箭头函数，否则会出现 this 指向不明确。

```
const btn = document.getElementById("btn");
btn.addEventListener("click", () => {
    // console.log(this === window);  this 不是指向 btn 这个按钮
    this.innerHTML = "clicked";
})
```

5. 需要在函数中使用 `arguments` 对象：在某些情况下，需要使用 `arguments` 对象来获取函数参数。在这种情况下，不能使用箭头函数，因为它们没有自己的 `arguments` 对象。

```
const fn1 = () => {
    console.log("arguments", arguments);
};
fn1(100, 200);
// ReferenceError: arguments is not defined
```

6. 需要在函数中使用 `super` 关键字：在某些情况下，需要使用 `super` 关键字来调用父类的方法。在这种情况下，不能使用箭头函数，因为它们没有自己的 `super` 对象。