07.箭头函数和 this

学习要点：

1.箭头函数

2.绑定 this

本节课我们来开始学习 ES6 新增的箭头函数，然后理解一下箭头 this；

一. 箭头函数

1. ES6 新增了一个使用(=>)箭头符号定义函数的语法特性，先来个最简单的；

```
let fn = name => name;
console.log(fn('Mr.Lee'));

//翻译成函数代码为：
let fn = function (name) {
    return name;
};
```

2. 箭头函数也可以传递两个或以上的参数，并实现运算后返回；

```
let fn = (x, y) => x + y;
console.log(fn(10, 20));

//翻译成函数代码为：
let fn = function (x, y) {
    return x + y;
}
```

3. 如果你定义的函数，并不需要传递参数，可以用()括号方式直接返回；

```
let fn = () =>  'Mr.Lee';
console.log(fn());

//翻译成函数代码为：
let fn = function () {
    return 'Mr.Lee';
};
```

4. 如果函数体需要更复杂的操作，可以将箭头符号右边使用传统函数体；

```
let fn = (x, y) => {
    return x + y;
};
console.log(fn(10, 20));
```

5. 如果箭头符号右边是对象，返回的是对象，则需要用圆括号包含着；

```
let fn = name => ({name : name, age : 100});
console.log(fn('Mr.Lee').name);

//翻译成函数代码为：
let fn = function (name) {
    return {
        name : name,
        age : 100
    }
};
```

6. 如果箭头符号左边是对象作为参数，右边是对象的属性运算，也支持；

```
let fn = ({name, age}) => name + ', ' + age;
console.log(fn({name : 'Mr.Lee', age : 100}));
```

7. 自我立即执行函数，也可以使用箭头符号来创建，具体如下：

```
((name) => {
    console.log(name);
})('Mr.Lee');

//翻译成函数代码为：
(function (name) {
    console.log(name);
})('Mr.Lee');
```

二. 绑定 this

1. ES6 之前有一个比较头疼的问题，就是 this 指向的问题，比如下面例子：

```
let obj = {
    name : 'Mr.Lee',
    age  : 100,
    fn : function () {
        setTimeout(function () {
            console.log(this.name + ', ' + this.age);
        }, 500)
    }
};

obj.fn();
```

2. 上面的例子比较典型，this 全局指向 window，在某个对象内部指向当前对象；
3. 当 obj 对象下包含了类似 setTimeout 函数内部，这时 this 指向就出现问题了；
4. Web 环境下，它指向了 Window，而 node 环境下它指向 setTimeout；
5. 所以，我们通俗的做法，就是将 this 在 setTimeout 外部进行赋值保存；

```
let that = this;
setTimeout(function () {
    console.log(that.name);
}, 500);
```

6. 箭头函数的出现，彻底解决了 this 在内部指向的问题，直接指向我们所需要；
7. 因为，箭头函数中的 this 是最外层定义的函数绑定，不受内部影响；

```
setTimeout(() => {
    console.log(this.name + ', ' + this.age);
}, 500)
```