正课:

1. ES6:

1. ES6:

面向对象:

封装: 2种:

1. 直接量:

var obj={

属性名:变量名, //如果属性名等于变量名，只写一个

方法名(){ ... }

}

2. 构造函数:

用class 类型名{ }包裹构造函数和原型对象方法

构造函数名提升为类型名，构造函数一律更名为constructor

直接定义在class中的方法，默认就保存进原型对象

继承: 两种类型间正式的继承:

1. 用extends 父类型，代替Object.setPrototypeOf()

2. 用super() 代替call

访问器属性:

从Object.defineProperty中取出，成为独立的原型对象方法

get 访问器属性名(){ ... }

set 访问器属性名(){ ... }

静态方法:

class 类型名{

...

static 方法名(){

}

}

Promise: 代替所有异步回调:

异步回调: 在不影响主程序情况下，等待一段时间后，调用回调函数或触发事件

问题: callback hell

callback hell: 因为提前传入回调函数参数，导致嵌套层级很深。

解决: Promise

Promise: 封装一段任务，并提供一个向后继续执行的开关