封装 mongodb DB 库之前的一些准备工作　es5、es6 class 类 静态方法 以及单例模式

主讲教师：（大地）
合作网站： http://www.itying.com/
中文文档： http://www.itying.com/koa

目录

一、原生 JS 中的类、静态方法、继承

类、静态方法

```
function Person(name,age) {
    this.name=name;
    this.age=age;
    this.run=function(){
        console.log(`${this.name}---${this.age}`)
    }
}
Person.set=function(){  /*静态方法*/

    alert('静态方法')
}
Person.prototype.work=function(){

    console.log(`${this.name}---${this.age}的工作是程序员`)
}

var p=new Person('张三','20');
p.work();

Person.set();/*调用静态方法*/
```

继承

```
function  Person(name,age) {
    this.name=name;
    this.age=age;

    this.run=function(){
        alert(this.name+'---'+this.age);
    }
}
Person.prototype.work=function(){

}

function Web(name,age){
    Person.call(this,name,age);  /*对象冒充实现继承*/
}

Web.prototype=new Person();    /*原型链继承*/
var w=new Web('李四',20);
w.run();
```

对象冒充实现继承不能继承原型链中方法和属性

原型链继承，实例化子类不能给父类传参

所以需要两者结合实现继承。

二、Es6 中的类、静态方法、继承

类：

```
class Person{
    constructor(name,age) {   /*类的构造函数，实例化的时候执行，new的时候执行*/
        this._name=name;
        this._age=age;
    }
    //定义方法     注意:在es6里面方法之间没有逗号（,）
    getName(){
        alert(this._name);

    }
    setName(name){
        this._name=name
    }
}
var p=new Person('张三1','20');
p.setName('哈哈');
p.getName();
```

Es6 继承

```
class Person{
    constructor(name,age){
        this.name=name;
        this.age=age;
    }
    getInfo(){
        alert(`姓名:${this.name} 年龄:${this.age}`);
    }
    run(){
        alert('run')
    }
}
class Web extends Person{  //继承了Person     extends          super(name,age);
    constructor(name,age,sex){
        super(name,age);   /*实例化子类的时候把子类的数据传给父类*/
        this.sex=sex;
    }
    print(){

        alert(this.sex);
    }
}
var w=new Web('张三','30','男');
w.print();
```

Es6 静态方法

```
class Foo {
  static classMethod() {
    return 'hello';
  }
}
class Bar extends Foo {
}
Bar.classMethod(); // 通过类来调用       'hello'

var b=new Bar();

console.log(b.classMethod()); // 报错
```

三、Es6 中的单例模式

```
class Person {
    static getInstance() {                    实现单例
        if (!Person.instance) {
            Person.instance = new Person();
        }
        return Person.instance;
    }
    constructor(){


        console.log('构造函数里面的方法')
    }
    find(){

        console.log('查找数据库的方法');
    }
}

var p1 = Person.getInstance();
var p2 = Person.getInstance();
```