正课:

1. \*\*\*\*自定义继承

ES5

1. 保护对象:

对象的属性: 数据属性,访问器属性

防篡改

2. Object.create

1. \*\*\*\*自定义继承:

何时: 如果觉得父对象不是想要的，可以换！

为什么: 一个对象希望使用另一个对象中的成员

如何:

1. 只修改一个对象的父对象:

obj.\_ \_proto\_ \_=father

问题: \_ \_proto\_ \_是内部属性,可能禁止使用

解决: Object.setPrototypeOf(child,father);

2. 同时修改多个子对象的父对象:

只要修改构造函数的原型对象

构造函数.prototype=father

时机: 必须在开始创建第一个子对象之前，就要更换

3. 两种类型间的继承:

问题: 两种类型间包含部分相同的属性结构和方法定义。

解决: 抽象父类型

如何:

1. 定义抽象父类型

相同部分的属性结构定义在父类型构造函数中

相同部分的原型对象方法定义在父类型的原型对象中

2. 让子类型继承抽象父类型

让子类型的原型对象继承父类型的原型对象

在子类型的构造函数中，借用父类型的构造函数

问题: 如果一个函数，不用new也不用.调用，则其中的this默认指window

解决: call

call: 2件事:

1. 调用一个函数

2. 将函数中的this替换为任意想要的对象

何时: 只要函数中的this不是想要的，都可用call在调用时更换

如何: fun.call(obj,参数1,....)

调用fun，并替换fun中的this指向obj对象, 再向fun中传入后续参数1,...

ES5

1. \*\*\*\*保护对象:

为什么: 普通js中的对象，可随时添加/删除属性，可任意修改属性值

何时: 今后，只要创建了对象，都要对对象提供保护。

如何: 2个层面

1. 保护对象的属性: 分为

命名属性: 可用.直接访问到的属性

又分为2类:

数据属性: 实际存储属性值得属性

每个数据属性，不再是简单的变量

而是一个包含四大特性的小对象

四大特性:{

value: 值, //实际存储属性值

writable: true/fales, //控制是否可修改

enumerable: true/false, //控制是否可被遍历

//并不是完全隐藏，用.依然可直接访问

configurable: true/false

//控制: 1. 能否修改前两个特性

//2. 能否删除当前属性

//一旦但改为false, 不可逆！

}

获取四大特性:

var attrs=Object.getOwnPropertyDescriptor(

obj,"属性名"

);

默认值: 以直接量方式创建的对象，四大特性默认值都为true！

修改四大特性:

Object.defineProperty(obj,"属性名",{

要修改的特性:值,

... : ...

});

强调: 通常修改其他特性时，都会伴随修改configurable为fales

问题: defineProperty每次只能修改一个属性的四大特性

解决:

Object.defineProperties(eric,{

属性名:{

四大特性:值,

... : ...,

},

... : {

... : ...,

}

})

问题: 数据属性的四大特性只能用固定的三种规则保护属性。无法使用自定义规则保护属性。

解决:

访问器属性: 不直接存储属性值，而是仅专门保护其他数据属性的特殊属性。

为什么: 为了用任意自定义规则保护属性

何时: 只要希望用自定义规则保护属性

如何:

添加访问器属性:

不能用直接量添加访问器属性

只能用defineProperty或defineProperties添加

Object.defineProperty(eric,"属性名",{

get(){ return 另一个受保护的数据属性}

set(val){

只有验证val的值符合要求

才将val保存到受保护的数据属性中

否则

就不赋值，而且报错！

}

enumerable:true,

configurable:false

})

强调: 被访问器属性保护的数据属性，通常都要隐藏(enumerable:false)！

强调: 凡是用defineProperty添加的新属性，四大特性默认反而为false！

内部属性: 不能用.直接访问到的属性

比如: \_ \_proto\_ \_ , class

2. 防篡改(结构): 3个级别

1. 防扩展: 禁止向对象中添加新属性

Object.preventExtensions(obj)

对象中有一个内部属性extensible,表示是否可添加新属性。默认为true

本质: preventExtensions就是将extensible改为false!

2. 密封: 兼具防扩展的基础上，同时禁止删除任何属性

Object.seal(obj)

本质: 2件事:

1. 改extensible为false

2. 将所有属性的configurable自动都改为false

3. 冻结: 即禁止扩展，又禁止删除，同时还禁止修改一切属性值

Object.freeze(obj)

何时: 只要一个对象，连属性值都不允许修改时

本质: 3件事:

1. 改extensible为false

2. 将所有属性的configurable自动都改为false

3. 将所有属性的writable都改为false

2. Object.create:

创建一个新对象，继承指定的父对象，同时扩展新属性

何时: 如果只有父对象，也想创建子对象时

如何: var child=Object.create(father);

如果需要为子对象扩展自有属性时

var child=Object.create(father,{

新属性:{

四大特性:值,

},

... : {

四大特性:值,

}

})