以下是javascript原型和原型链一些学习总结。

原型，我们每创建一个新函数都有为这个函数创建一个原型对象，

而我们的新函数也会自动的有一个prototype属性，这个属性就指向这个函数的原型对象，函数的原型对象自己也有一个constructor属性，这个属性指向原函数（就是有prototype的属性的那个函数）。

如果说我们刚刚创建的新函数是个构造函数，那么，用这个构造函数创建出来的每一个实例的prototype属性都会指向构造函数的原型对象.

代码示例：

Function Person(){

Name:’test’,

Age:23

//指向此函数的原型对象

Prototype(此条属性每个函数都有)

}

Var person1 = new person();

Var person2 = new person();

接下来说说原型链，还是刚刚这个图，当我们调用person1的name属性的时候，javascript引擎会现在person1本身找一下，看有没有这个属性，如果有，返回此属性，如果没有，就通过prototype属性去他的原型对象上找，如果有，返回，如果没有，再继续上面的步骤，从继续往上找，直到顶端，返回undefinde.

这个顺序也和重要，比如说Person.prototype上有name这个属性，而我们的实力person1想改变name的值，我们只需要在person1里面再写一遍name属性就可以了，因为我们访问属性的时候是现在自身找的，在自身找到之后会直接返回，就不会往上继续找了。

下面写一些可能会常用到的api

delete删除属性 比如说我们想删除person1的name属性，

delete person1.name

hasOwnProperty() 用于检测属性是否在实例中，这个方法是object.prototype中的一个方法

比如说， alert(person1.age) //23

Person1.hasOwnProperty(“age”); //false

这样就可以判断出，person1的age属性是从原型得到的

Function Person{

Name:’test’,

Age:23,

Prototype

}

Person的原型对象

Person prototype

Constructor

Name:’test’,

Age:23

Person2{

prototype

}

Person1{

prototype

}