[**一些JavaScript Quiz**](http://www.cnblogs.com/jenry/archive/2010/12/08/1900401.html)

**a.x = a = { }, 深入理解赋值表达式**

var o = {x : 1};  
var a = o;  
  
a.x = a = {name:100};  
  
console.log(a.x); // undefined  
console.log(o.x); // {name:100}  
  
// a.x = a = {name:100};  
// 等价于 a.x = (a = {name:100});  
// 首先计算a.x的引用，然后计算(a = {name:100})的返回值

**if 语句的简写**

var condition = true, numb = 0;  
if(condition) {  
 alert('rain-man')  
}  
if(condition) {  
 numb = 1 + 2;  
}

等同于

var condition = true, numb = 0;  
condition && alert('rain-man');  
condition && (numb = 1 + 2);

**&& 和 || 的计算取值**

(true && 222); // 222  
!!(true && 222); // true  
(false && 222 ); // false

(false || 222); // 222  
!!(false || 222); // true

!!variable 会返回和原值相等的boolean值

**Object的构造**

function Object() { [native code] }  
Object.prototype = {  
 constructor: function Object() { [native code] },  
 hasOwnProperty: function hasOwnProperty() { [native code] },  
 isPrototypeOf: function isPrototypeOf() { [native code] },  
 propertyIsEnumerable: function propertyIsEnumerable() { [native code] },  
 toLocaleString: function toLocaleString() { [native code] },  
 toString: function toString() { [native code] },  
 valueOf: function valueOf() { [native code] }  
};  
  
Object.prototype.constructor === Object; // true

**prototype中的一些细节**

var A = function(){  
 this.name = 'rain-man';  
};  
A.prototype = {  
 name : 'cnblogs'  
};  
var o = new A();  
console.log(o.name); // 'rain-man'

**创建对象，并保持原型链**

var O = function(obj) {  
 function T() {}  
 T.prototype = obj;  
 return new T();  
};  
  
var obj = {name: 'obj', age: 0 },   
 obj1 = O(obj),   
 obj2 = O(obj1);  
  
// 更改原型链的一处，所有原型链都会更改  
obj.name = 'superclass';   
console.log(obj1.name); // 'superclass'  
console.log(obj2.name); // 'superclass'  
  
// 每一层可单独处理  
obj1.name = 100;  
console.log(obj1.name); //100  
delete obj1.name; //暴漏原型链  
console.log(obj1.name); // 'superclass'