# 面试题

1. noscript标签是用来定义脚本未被执行时的替代内容。
2. undefined和null与任何有意义的值比较返回的都是false，但是null与undefined之间互相比较返回的是true。
3. undefined值是派生自null值的，因此规定对它们的相等性测试null == undefined要返回true；但全等性测试null === undefined会返回false，因为它们是不同类型的值。
4. void 作为运算符后面接的是表达式，void expression。而void(0)也是被当做void 0。如果直接void()，则是把void当做函数使用了，但是此时void并没有定义。
5. NOSCRIPT标签用来定义在脚本未被执行时的替代内容。也可以用在检测浏览器是否支持脚本，若不支持脚本则可以显示NOSCRIPT标签里的innerText
6. 短路或  如果第一项是true 则结果为true 如果第一项为false 结果 为第二项（第二项不做计算直接返回）
   1. A||B:当A为假时候，结果等于B，当A为真，结果直接为A，不执行B。
   2. A&&B:当A为假时候，直接为false，当A为真，结果直接为B。
7. isNaN（）在接受一个值后，会尝试将这个值转换为数值。某些不是数值的值会直接转换为数值，例如字符串“10”或Boolean值。而任何不能被转换为数值的值都会导致这个函数返回true。
   1. isNaN(false) //false
   2. isNaN('123') //false
8. 关于Javascript中数字的部分知识总结：
   1. Javascript中，由于其变量内容不同，变量被分为基本数据类型变量和引用数据类型变量。
   2. 基本类型变量用**8字节**内存，存储基本数据类型(数值、布尔值、null和未定义)的值，引用类型变量则只保存对对象、数组和函数等引用类型的值的引用(即**内存地址**)。
   3. JS中的数字是不分类型的，也就是没有byte/int/float/double等的差异。
9. 保留字 （字母排序）

abstract

boolean break byte

case catch char class const continue

debugger default delete do double

else enum export extends

false final finally float for function

goto

if implements import in instanceof int interface

long

native new null

package private protected public

return

short static super switch synchronized

this throw throws transient true try typeof

var void volatile

while with

1. 一个promise可能有三种状态：等待（pending）、已完成（fulfilled）、已拒绝（rejected）。  
   promise必须实现then方法（可以说，then就是promise的核心），而且then必须返回一个promise，同一个promise的then可以调用多次，并且回调的执行顺序跟它们被定义时的顺序一致  
   then可以接受另一个promise传入，也接受一个“类then”的对象或方法，即thenable对象。
2. 在jsx中直接写行内样式时不能采用引号，而是style={{color:'red'}}的方式。
3. jsx中class是js关键字，要用className。
4. typeof
   1. typeof Symbol()    //"symbol"
   2. typeof Number()    //"number"
   3. typeof String()    //"string"
   4. typeof Function()  //"function"
   5. typeof Object()    //"object"
   6. typeof Boolean()   //"boolean"
   7. typeof null    //"object"
   8. typeof undefined   //"undefined"
5. Flash提供了ExternalInterface接口与JavaScript通信，ExternalInterface有两个方法，call和addCallback，call的作用是让Flash调用js里的方法，addCallback是用来注册flash函数让js调用。
6. 位运算 按位异或运算
   1. eg:

1<<4   左移相当于1\*2^4=16

a^=16-1=15

a=a^15=10^15

^ 异或运算：

10的二进制00001010，

15的二进制00001111

=========>00000101  转成十进制：5

（按位异或运算，同为1或同为0取0，不同取1）

1. Number转换转为0

Number() Number(0) Number('') Number('0') Number(false) Number(null) Number([]) Number([0])

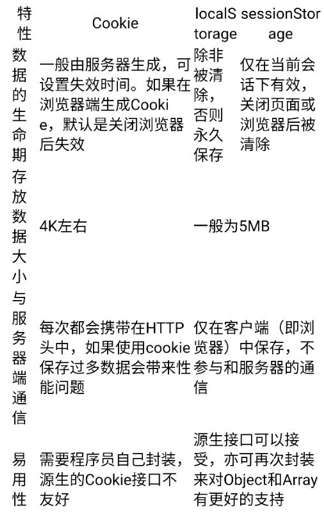
1. console.log(Number("")); //0 console.log(Number(null)); //0 console.log(Number(undefined)); //NaN console.log(parseInt("")); //NaN console.log(parseInt(null)); //NaN console.log(parseInt(undefined)); //NaN
2. Number()可以用于任何数据类型的转换。转换规则如下：   
   1. 如果是Boolean值，true和false将分别被转换为1或0.   
   2. 如果是null值，返回0；   
   3. 如果是undefined，返回NaN;   
   4. 如果是字符串则遵循以下规则：   
    a. 字符串只把含数字（包含前面带正号或负号的情况），则将其转换为 十进制数值，前导零会被忽略。（例如“011”会变成11）；   
    b.如果字符串中包含有效的浮点格式，则将其转换为相应的浮点数值,前 导零会被忽略；   
    c.如果字符串中包含十六进制格式，则将其转换为相同大小的十进制整数；//number("01f")=31   
    d.如果字符串是空的，则将其转换为0；//number(" ")=0;   
    e.如果字符串中包含除以上格式之外的字符，则将其转换为NaN. eg://number("helloworld")=NaN  
   5.如果是数值，不用说当然是传啥返啥；   
   6.如果是对象，则调用对象的valueOf()方法，然后依照前面的规则转换返回的值。如果转换的结果是NaN，则调用对象的toString()方法，然后再次按照前面的规则转换返回的字符串值。
3. var f = function g() {return 23; }; typeof g();

如果是typeof f，结果是function

如果是typeof f()，结果是number

如果是typeof g,结果是undefined.

如果是typeof g(),结果是ReferenceError，g is not defined

1. console.log(1+ +"2"+"2");第一个+"2"中的加号是一元加操作符，+"2"会变成数值2，因此1+ +"2"相当于1+2=3.然后和后面的字符串“2”相合并，变成了字符串"32".
2. 
3. 在==的隐式转化中，是对两边进行Number()
4. 6 种基本数据类型是指：Undefined、Null、Boolean、Number、String 与 Symbol.
5. 
6. 严格模式:
   1. 严格模式禁止使用with语句，因为with语句无法在编译时就确定属性到底归属哪个对象。
   2. 严格模式下禁止删除变量，只有configurable设置为true的对象属性，才能够被删除。
   3. 严格模式下arguments不在追踪参数的变化
7. '',false,0,[]都能转换为数字类型0, NaN做任何运算返回都是false
8. setTimeout中所执行函数中的this指向window
9. SVG和Canvas区别

SVG

* 1. 不依赖分辨率
  2. 支持事件绑定
  3. 大型渲染区域的程序
  4. 不能用来实现网页游戏

Canvas

1. 依赖分辨率
2. 不支持事件绑定
3. 最适合网页游戏
4. 能够以".jpg"或“.png”格式保存结果图像
5. typeof输出都用双引号引起来
6. 在ES6中提供了ArrayBuffer、TypeArray、DataView
   1. ArrayBuffer:连续的内存缓冲区，用于实际存储各种类型的数组数据
   2. Typed Array View类： 比如Int32Array、Unit8Array、Float32Array等，表示一个特定类型的数组
   3. DataView:工具类，提供getUint8、setFloat32等工具方法修改ArrayBuffer不同位置的数据值。
7. HTML5新增的布局标签

menu, hgroup, figure, header, footer, aside, nav, article, section

figure这个标签经常是在主文中引用的图片，插图，表格，代码段等等

HTML <hgroup>元素表示文档的一个部分的多级标题。它将一组<h1> - <h6>元素分组。

1. <menu>标签用于上下文菜单，工具栏和列表表单控件和命令。
2. 严格模式是浏览器根据web规范去显示页面，是一种要求比较严格的DTD，不允许使用任何表现层的语法，混杂模式是一种以向后兼容的方式去显示，可以实现IE5.5以下版本浏览器的渲染模式。
3. 严格模式主要有以下限制：
   1. 变量必须先声明后使用
   2. 函数的参数不能有同名属性，否则报错
   3. 不能使用with语句
   4. 不能对只读属性赋值，否则报错
   5. 不能使用前缀为0表示八进制，否则报错
   6. 不能删除不可删除的属性，否则报错
   7. 不能删除变量delete prop，会报错，只能删除属性delete global[prop]
   8. eval和arguments不能被重新赋值，
   9. arguments不会自动反映函数参数的变化
   10. 不能使用arguments.callee
   11. 不能使用arguments.caller
   12. 禁止this指向全局对象
   13. 不能使用fn.caller和fn.arguments获取函数调用的堆栈
   14. 增加保留字（比如protected，static，inerface）
4. null和undefined区别：
   1. null时对象，undefined是一个数据类型
   2. 转成数值时：null为0，undefined为NaN
   3. 变量声明还未赋值时，变量的值为undefined
   4. null用来表示尚未存在的对象，常用来表示函数企图返回一个不存在的对象。
   5. undefined表示“缺少值”，就是尚未定义。  
      典型的用法是：
      1. 变量声明了，但尚未定义
      2. 调用函数时，应该提供的参数没有提供
      3. 对象没有赋值的属性，默认返回undefined
      4. 函数没有返回值，默认返回undefined
   6. null表示“没有对象”，即此处不应该有值。  
      典型的用法是：
      1. 作为函数的参数，表示该函数的参数不是对象
      2. 作为对象原型链的终点。
5. background-position默认对应css盒子模型的padding的左上角，可以通过background-origin修改它的默认值。backround-origin:border-box;
6. sass是什么？有哪些好处和坏处

css预处理器

好处：变量，mixin、函数、规则嵌套、颜色处理

坏处：css的文件体积和复杂度不可控，调试难度增加、成本

1. javaScript包含核心（ECMAScript）、文档对象模型(DOM)、浏览器对象模型(BOM),ECMAScript是JavaScript的规范，JavaScript是ECMAScript的实现。
2. 哪些操作会造成内存泄漏

反复重写一个属性会造成内存大量占用（但是关闭IE后内存会被释放）

给DOM对象用attachEvent绑定事件

从外到内执行appendChild，这时即使调用removeChild也无法释放。

闭包，意外的全局变量，没有清理的DOM元素，被遗忘的定时器，或者回调函数，子元素存在引用。

1. 内存泄露指任何对象在不在拥有或者需要它之后仍然存在。
2. MVC一种使用MVC(Model View Controller 模型 视图 控制器)设计创建Web应用程序的模式
   1. Model（模型）表示应用程序核心（比如数据库记录列表）
   2. View(视图)显示数据
   3. Controller(控制器)处理输入（写入数据库记录）

优点：

MVC分层有助于管理复杂的应用程序

MVC分层同时也简化了分组开发

MVC三个部分职责明确，相互分离，大大提高了应用的灵活性和重要性

1. 数组去重 Array.from(new Set(arr))