正课：

1. Date:

2. Error:

3. Function:

1. Date:

API: 3句话:

1. 单位:

FullYear Month Date Day

Hours Minutes Seconds Milliseconds

2. 每个单位上都有一对儿get/set方法

其中: getXXX() 专门获取单位的数值

setxxx() 专门设置单位的数值

比如: date.getDate() date.getFullYear()

date.setDate(20) date.setFullYear(2018)

特例: Day没有set, 不能修改

3. 取值范围:

Month: 0~11 计算机中的month比现实中小1

在取值和修改时，要修正

Date: 1~31

Day: 0~6

Hours: 0~23

Minutes/Seconds: 0~59

计算: 2种:

1. 两个日期对象可相减: 结果为毫秒差

2. 对任意单位做加减:

3步:

1. 获得单位上的当前值: 2018/2/13 getDate() 13

2. 做加减: 13-30 = -17

3. 将计算结果set回去  
 .setDate(-17);

setDate内置日期进制算法。

简化: date.setxxx(.getxxx()+n)

格式化:

.toLocaleString() 转为当地时区的简化版格式

.toLocaleDateString() 转为当地时区的简化版格式，仅保留日期

2. Error:

什么是: 程序执行过程中，导致无法继续正常执行的异常情况

只要程序发生错误，就会中断退出

错误处理: 即使程序发生错误，也保证不会中断退出的机制

如何错误处理:

try{

可能发生错误的正确代码

}catch(err){

一旦发生错误，就自动执行的错误处理代码

}

其中: err是错误对象: 错误发生时，自动创建的封装错误信息的对象。

效率: 放在try catch中的代码，效率略有降低

解决: 用if...else代替try...catch

鄙视: js中错误类型: 6种:

SyntaxError 语法错误

ReferenceError 引用错误 要用的变量没找到

RangeError 范围错误 参数超范围

TypeError 类型错误 错误的使用了(),[]等。

URIError URI错误

EvalError Eval错误

抛出自定义错误:

何时: 协作开发中，程序的作者，向程序的调用者，提示错误的使用程序。

3. Function

什么是: 保存一段可重用的代码段的对象

何时: 只要一段代码可能反复使用时，都要封装为函数，反复调用函数

如何:

创建: 3种:

1. 直接量: function 函数名(参数列表){

函数体;

return 返回值;

}

缺: 会被声明提前, 打乱程序的执行顺序

解决:

2. 赋值: var 函数名=function(参数列表){

函数体;

return 返回值;

}

优: 不会被声明提前

揭示: 函数其实也是一个引用类型的对象

函数名其实只是一个普通的变量

函数名通过地址引用函数对象

function其实是创建一个新函数对象的意思

3. 用new:

var fun=new Function("参数1","参数2",...,"函数体")

以下创建函数正确的是：

function cmp(a,b){return a-b}

var cmp=function(a,b){return a-b}

var cmp=new Function(a,b,"return a-b")

var cmp=new Function("a","b","return a-b")

重载(overload):

什么是: 相同函数名，不同参数列表的多个函数，在调用时，可根据传入的参数不同，自动选择匹配的函数执行。

为什么: 减少API的数量，减轻调用者的负担

何时: 只要一项任务，可能根据传入参数的不同，执行不同的流程时。

如何:

问题:js语法默认不支持重载，因为不允许多个同名函数同时存在！

解决: arguments

什么是: 函数调用时，自动创建的

保存所有传入函数的参数值的

类数组对象(长的像数组的对象)

vs 数组: 相同: 1. 下标; 2. .length; 3. 遍历

不同: 类型不同, API不通用

何时: 重载:

匿名函数:

什么是: 定义函数时，不指定函数名的函数

何时: 一个函数只使用一次时

为什么: 节约内存

如何: 2种:

1. 回调函数: 定义函数后，自己不执行，交给别的函数，由别的函数控制调用和执行过程。

2. 匿名函数自调: 函数定义后，自己调用自己!

为什么: 全局变量，用完，无法自动释放！污染全局！

解决: 今后禁止使用全局变量！

何时: 只有所有功能的代码，必须放在匿名函数自调中

如何: (function(){

//函数体

})();

鄙视:垃圾回收

什么是: 引擎会自动释放不再使用的对象的内存空间

为什么: 内存空间大小是有限的！系统都会释放不再被使用的对象，为后续新对象释放更多空间。

垃圾回收器: 专门统计并回收内存中不再被使用的对象的空间的 小程序

如何:

1. 垃圾回收器伴随主程序启动而启动，在后台伴随主程序运行而运行。

2. 垃圾回收器会记录每个对象被几个变量引用着

3. 只要一个对象不再被任何变量引用，则垃圾回收器自动释放该对象

习惯: 只要一个大对象，不再被使用，应该主动将变量赋值为null