学习一门编程语言的基本步骤

* 了解背景知识：历史，现状，特点，应用场景
* 搭建开发环境，编写hello world
* 变量和常量
* 数据类型
* 运算符
* 逻辑结构
* 函数和对象
* 通用小程序
* 第三方的库和框架
* 个人项目

程序员必做50题

* JS概述
* 历史

1995年出现在网景公司的浏览器中

1995年，IE3中也出现了JS，叫做Jscript

1997年，出现了国际标准规范ECMAScript

2009年，出现了CommonJS规范，JS开始往服务器端发展，出现了Nodejs

* 现状

JS既可以运行在浏览器中，也可以运行在服务器端

* 特点

解释型语言，编译一行执行一行

弱类型语言

基于对象

跨平台，运行在不同的操作系统

* 应用场景

开发浏览器端的交互效果，进行服务器端的开发

* 搭建开发环境
* 浏览器

浏览器自带的JS解释器，推荐使用谷歌浏览器

火狐 谷歌 IE safari 欧朋

* 服务器端

安装Nodejs

[www.nodejs.org](http://www.nodejs.org/) 官网

打开命令行窗口

Node –v 回车 查看版本号

（3） 运行JS

创建01.JS和01.html两个文件，把js嵌入到html中

<script src=‘01.js’></script>

Nodejs

Node 空格 拖拽文件 回车

3 JS的语法规范

区分大小写

每行代码结束的分号可以省略

分为单行注释（//...）和多行注释（/\*...\*/）

* 变量

用于保存程序中的数据

* 声明变量

Var x=1;

使用关键字 var 声明变量，x是变量名称，在内存中开辟了一块空间，把数据保存进去。

练习：声明多个变量，分别保存一个员工编号，姓名，性别，生日，工资，并打印出

* 命名规范

变量名称可以由字母，数字，下划线，美元符号组成，不能以数字开头，不能使用关键字，不推荐使用汉字

多个单词组合作为变量名 可以用下划线分辨

userNameOne 驼峰命名法 每个单词首字母大写

（3） 变量赋值

变量声明未赋值则为undeined（未定义），变量可以多次赋值，赋不同类型的值，是弱类型语言的特点。

练习：声明多个变量分别保存一个人的成绩，声明一个变量保存总成绩，最后打印出总成绩。

* 一次声明多个变量

Var a=1,b=2

多个变量之间用逗号隔开

练习：声明多个变量保存商品的单价和数量，声明变量保存总价，打印出总价

课后任务

* 复习今天的内容，整理思维导图
* 重新编写SQL查询语句

练习： 交换两个变量中的数据

a=1 b=2 转换 a=2 b=1

3 预习JS第二天，数据类型，数据类型转换，运算符

1 常量

声明后必须赋值，不允许重新赋值

const userName=‘root’；

练习：声明常量保存圆周率，声明变量保存半径，分别计算园的周长和面积并打印出来

2 数据类型

分为原始类型和引用类型

原始类型分为数值型，字符串型，布尔型，未定义 型，空

引用类型是对象，后边讲

（1）数值型

整数

十进制 1 2 3 4 5 6 7 8 9 10

八进制 1 2 ... 7 10 11 12

十六进制 1 2 ... 8 9 a-f 代表10-15

八进制特点 以0开头的数字

十六进制 以0x开头的数字，a-f代表10-15，不区分大小写

浮点数

3141.5-1

314.15

31.415e1

3.1415e2

tepeof 检测数据类型

（2）字符串型

被引号所包含的数据， 不区分单双引号

查看任意一个字符的Unicode码

‘a'.charCodeAt()

(3) 布尔型

只有两个值，分别是true和false，表示真和假

条件的结果会是布尔型，还可以用于保存状态，例如是否登录，是否注册等等...

（4）未定义型

只有一个值为undefined

声明了变量未赋值为undefinde

（5） 空

只有一个值为null,常和引用类型数据一起使用

3.数据类型转换

（1） 隐式转换

运行过程中自动产生的转换

① 数值+字符串 //数值转为字符串

2+‘3’ //‘23’

（2）数值+布尔型 //布尔型转为数值 true-1 false-2

2+true //3

2+false //4

（3）字符串+布尔型 //布尔型转为字符串

‘4’+true //‘4true'

练习 查看一下程序的运行结果

var a=1,b=true,c='tedu';

console.log(a+b+c);//2tedu

console.log(b+c+a);//truetedu1

console.log(c+a+b);//tedu1true

练习 假设从后端获取到了一个员工的编号，姓名，打印出以下格式，员工编号 姓名

js中加号（+）的作用

加法运算

执行字符串之间的拼接

NaN(Not a Number):不是一个数字，在将数据转化为数值型的时候，没有成功返回的结果。NaN和任何数字执行加减乘除等运算结果还是NaN

结论：隐式转换为数值自动调用Number函数

②强制转

（1）强制转为数值

Number('5a')//NaN

Number('5')//5

Number(true)//1

Number=(false)//0

Number(undefined)//NaN

Number(null）//0

(2)强制转为整型

parseInt(

用于把字符串和小数转为整型,其他转换都是NaN

parseInt(1.9)//1

parseInt(‘3.14')//3

parseInt(‘3.14a’)//3

parseInt(‘a3.14)//NaN

(3)强制转为浮点型

parseFloat()

用于将字符串转换为浮点型

parseFloat('5.18');//5.18

parseFloat('4.7a');//4.7

parseFloat('a4.7');//NaN

parseFloat('3b');//3

（4）强制将数值和布尔型转为字符串

var num=5;

//将转换后得到的字符串保存到变量str中

var str=num.toString();

4.运算符

表达式：由数据或者由运算符所连接的操作数据组成的形式

运算符分为六大类

算术运算符 比较运算符 逻辑运算符 位运算符

赋值运算符 三目运算符

①算数运算符

+ - \* / % ++ --

%（取余）

++（自增） 在原来基础之上加1

--（自减） 在原来基础之上减1

var a=1

var b=a++；//先把a的值赋给b，然后a在执行自增

var c=2

var d=++c，//先让c的值进行自增，然后将自增的结果赋给d

练习 查看以下程序的运行结果

var num=8

var num1=num--;

var num2=--num;

console.log(num)

//面试题

//+ -正负，隐式转换

console.log(5 + + ‘3')；//8

//自增和自减，隐式转换为数值

var a=‘2';

a++；

console.log(a);//3

②比较运算符

> < >= <= ==（等于） !=(不等于) ===（全等于） ！==（不全等于）

== 等于 ，比较值是否相同

=== 全等于，先比较类型，再比较值（推荐）

3>‘10’数字和字符串（布尔型）比较，隐式转换为数值

‘3’>‘10'两个字符串比较，比较的是首个字符的unicode码

‘3’->51 ‘1’->49

3>‘10’//false

3<‘10’//false

3==‘10’//false

NaN==NaN //false

NaN和任何值比较（ > < >= == ===)都是false

③逻辑运算符

&& (逻辑与) 表示并且，关联的两个条件都是true，结果是true，否则是fslse

|| （逻辑或） 表示或者,关联的两个条件至少有一个为true,否则false

! （逻辑非） 非 取反

console.log（！false）//true

短路逻辑

&& 当第一个条件为false，不再执行第二个条件

|| 当第一个条件为true，不再执行第二个条件

短路逻辑无需关注整体上是true还是false，重点是看第二部分有没有执行

练习：查看一下程序是否报错

var num=5;

num>8 && console,log(a);

课后任务

（1） 复习今天内容，整理思维导图

（2） 练习：声明变量保存任意一个年份，使用短路逻辑，如果是闰年，打印‘闰年’

4年一闰，100年不闰，400年再闰

能被4整除，并且不能被100整除，或者能被400整除

（3）预习js day3

④ 位运算符（简单了解）

模拟计算机底层的运算过程，先将值转为二进制然后再运算，运算完再把结果转回成十进制

1 2 3 4 5 6 7 8 9

1 10 11 100 101 110 111 1000 1001

& 按位与，上下两位都是1结果是1，否则是0

| 按位或，上下两位含有1结果是1，否则是0

^ 按位异或，上下两位不同是1，否则是0

3&5 5|7 5&7 9^3

011. 101 101 1001

101. 111 111 0011

001 111 101 1010

1 7 5 10

>> 按位右移 去掉最后的若干位

<< 按位左移 在最后补若干个0

7>>1 让7往右移动一位 符号后数字决定去掉几位

去掉最后一位 -> 11

⑤ 赋值运算符

= += -= \*= /= %= ...

计算赋值 ：先执行计算，在执行赋值

练习：声明变量保存商品的价格，让价格执行打八折，最后打印价格

⑥三目运算符

一目运算符：由一个运算符连接了一个操作数据或者表达式 ++ -- ！

二目运算符：由一个运算符连接了两个操作数据或者表达式

三目运算符：由两个运算符连接了三个操作数据或者表达式 ? :

条件表达式 ? 表达式1 : 表达式2

如果条件表达式为true，执行表达式1，否则执行表达式2

5.浏览器端函数

alert() 弹出警示框

prompt() 弹出提示（输入框），需要使用变量保存用户输入的值，类型是字符串型，如果点击取消返回显示 null

练习：要求弹出两次提示框，分别输入数字，最后要计算两个数字相加的和并打印出来

程序=数据+算法

程序的执行方式分为顺序执行、选择执行、循环执行

6.流程控制（选择执行）

（1）if语句

满30减20

if（条件表达式）{

语句块

}

如果if后语句中的只有一行语句，可以省略大括号。

以下数据作为条件表达式出现，隐式转换为fslse

0 ‘’ 两个引号空字符 NaN undefined null

声明一个变量，保存用户的签名内容，如果签名内容为空，设置签名内容‘这家伙很懒，什么也没留下’，最后打印签名内容

（2）if-else语句

if（条件表达式）{

语句块1

}else{

语句块2

}

（3）if-else 嵌套

if(条件表达式1){

语句块

}else if(条件表达式2）{

语句块2

}else{

语句块n

} //以上所有的条件都是false

（4）switch-case

根据一个表达式的值，选择执行对应的语句块，属于是一种特殊的多项分支语句

switch(表达式){

case 值1：//会将表达式和值1比较

语句块1

break；//结束，跳出switch-case语句

case 值2：

语句块2

break;

default:

语句块n

}

switch后的表达式在case后的值比较的时候只能进行全等于（===），要求值和类型都相同

练习：声明变量保存任意一个城市名称，根据城市的名称打印对应的美食

课后任务：

①复习今天内容，整理思维导图

②练习：弹出两次提示框，分别输入商品的单价和数量，计算出总价，如果总价满1000打九折，假设会员卡内有余额800，如果足以支付，警示框弹出‘pay success’,否则警示框弹出‘pay error’。

③预习JS第四天，循环

1.循环执行

循环是一遍又一遍执行的相同或者相似的内容

循环条件：可以控制循环的次数

循环体：要重复执行的相同或者相似内容

（1）while循环

while（循环条件）{

循环体

}

（2）break

在循环体中执行后，就会跳出循环。

（3）do-while循环

do{

循环体

}while(循环条件）；

练习：循环弹出提示框，声明变量保存用户输入的值，如果输入正确强制结束循环，警示框弹出‘login success’;如果输入的次数超过三次，强制结束循环，提示警示框‘game over’.

(4)for循环

for(初始值；循环条件；增量){

循环体

}

2.break和continue

用于循环体中

break 跳出循环，结束，不再执行任何循环中的内容

continue 跳过某一次循环，不执行某一次中剩余的循环体，还还执行下一次的循环

练习：打印1-100之间所有的奇数，遇到偶数就跳过

3.循环嵌套

在一个循环体的内部，还有其他的循环；任意两个循环之间可以相互嵌套

课后任务：

1.复习今天内容

2.练习

计算1-100之间所有整数的和，一旦总和超过4000，强制结束循环，打印当前和sum+=i

循环打印出2000-2100之间的前10个闰年

使用循环嵌套打印九九乘法表

1.函数 16号

Number()/parselnt()/parseLloat()/alert()/prompt()..

函数：是一个功能体，提供若干个数据，返回处理的结果，为了封装重复执行的代码。

函数分为系统函数和自定义函数

（1）创建普通函数

function 函数名称（）{

函数体——封装的要重复执行的代码

}

调用

函数的名称加（）

每调用一次，就执行一次函数体中的代码

练习：创建函数 getSum,在函数体中封装计算1-100之间所有整数的和并打印出来，调用多次

（2）创建带有参数的函数

function 函数名称（参数列表）{//用于接收传递的数据

函数体

}

调用

函数名称（参数列表）

参数：创建函数是的参数称为形参，调用函数时的参数称为实参；多个参数之间用逗号隔开，实参的数量可以多于或者少于形参的数量，如果形参未赋值则为undefined

练习：创建函数getSun，在函数体中计算1-任意数字之间所有整数的和并打印出来，调用多次

练习：创建函数getSun，计算出任意两个年份之间所有的闰年个数并打印出来，调用多个参数

（3）创建带有返回值的函数

function 函数名称（参数列表）{

函数体

ruturn 值 //返回值，函数调用后得到的结果

}

调用

函数名称（参数列表）

return,返回，用于将函数调用的结果返回出去；ruturn后不加值或者再函数中不写rutron，则返回undefined; return一旦执行，跳出函数，结束函数的调用

练习：创建函数getMax1，传递任意两个数字，返回最大值

练习：创建函数getMax2，传递任意三个数字，返回最大值

练习：创建函数getStatus,传递订单的状态码，返回对应的中文状态。

① 等待付款 ② 等待发货 ③ 运输中

④ 已签收 ⑤已取消

使用swtich-case

对比return和break

return 用于函数中，一旦执行就会跳出函数，结束函数的调用

break 用于 switch-case 语句和循环，一旦执行就会跳出switch-case或者循环，结束相应语句的执行

练习：创建函数isRun，传递任意一个年份，查看是否为闰年，返回一个布尔型的值

2.变量的作用域

作用域： 分为全局作用域和函数作用域，具体指变量和函数的可访问范围

全局变量：在函数外声明的变量，叫全局变量，在任何作用域下都可以访问到

局部变量：在函数内使用var声明的变量，叫局部变量，只能在当前的作用域下访问到

在函数中不加var声明的变量是全局变量，不推荐，后期严格模式下会报错

变量提升：JS程序执行前，会将var声明的变量提升到所在作用域的最前面，只是提升证明，赋值不提升

3.函数的作用域

全局函数：在全局的作用域下创建的函数，可以在任意作用域下访问

局部函数：在函数作用域下创建的函数，只能在所在的作用域下访问

函数提升：

JS程序执行前，会将整个函数提升到所在作用域的最前面，可以先写调用，再写创建

练习：创建一个函数，能否在创建之前调用到

4.递归（提前了解）

在函数内调用自身这个函数，本质是一个死循环

斐波那契数列：第1项和第2项的值为1，从第3项开始，每项的值是前两项相加的值

1 1 2 3 5 8 13 ...

课后任务

（1）复习今天内容，整理思维导图

（2）练习：使用递归循环打印三次

练习：创建函数，传递任意一个数字，返回斐波那契数列中该项的值

使用循环

（3）预习js第六天 匿名函数、对象

1.递归

在函数内调用自身这个函数，本身属于死循环

使用递归，要有跳出条件，结合着return。

斐波那契数列使用递归的时候涉及到太多函数的嵌套，属于CPU密集型算法，JS属于单线程运行（只利用一个内核）JS使用递归会比较慢

2.匿名函数

function(){}

（1）创建函数

function(){

}

函数表达式

var fun=function(){

}

变量fun就是函数名称

调用 fun();

①对比函数名称和函数名称（）的区别

函数名称本质上就是一个变量，保存了一个函数

函数名称（），调用函数，只想函数体中的代码，得到返回结果。

②对比函数声明和函数表达式两种创建函数的区别

函数声明创建的函数存在函数提升，可以先调用再创建

函数表达式创建的函数使用了变量，只存在变量的提升，必须先创建再调用

练习：使用函数表达式创建函数，计算任意两个数字相加的和并返回结果

（2）匿名函数的自调用

全局污染：全局变量的出现，可以在任何的作用域下访问到，对其它的变量造成了污染。

解决方案：把变量都转为局部变量。

（function(){

//函数体封装的代码，都是在函数作用域下，变量都是局部变量

})（）；

（3）回调函数

将一个函数以实参形式传递，需要有形参接收并调用。

function ran(madai){

madai() //接收并调用回调函数

}

function dong();{

}

以下两种传递的都称为回调函数

ran(dong);

ran(function(){})

3.系统函数（全局函数）

Number()/parseInt()/parseFloat()

isNaN() 检测一个值是否为NaN，会自动调用Number转为数值，然后检测。常用于检测用户输入的值是否为数值

是NaN-true 不是NaN-false

isFinite() 检测一个值是否为有限制，是-true， 不是-false;0作为除数得到的结果是无限值Infinity

eval() 行字符串中的表达式

3.对象

对象是由一组属性和方法的集合

我的电脑，包含的属性有品牌，颜色，尺寸...方法（功能）玩游戏，写代码，做设计..门口的一辆汽车，轮子，颜色，型号...方法 代步，拉货...

万物皆对象

（1）JS中的对象

自定义对象，用户创建的对象

内置对象，JS官方所提供的对象

宿主对象，根据不同的环境划分

dom bom

（2）自定义对象的创建

对象字面量

内置构造函数（第三阶段讲到）

（3）使用对象字面量创建对象

使用大括号{}创建空对象

属性名和属性值之间用冒号：隔开，多组属性之间用逗号隔开

属性名的引号可加可不加，如果含有特殊字符必须加

练习：创建一个汽车对象，包含由汽车的品牌型号，颜色，宽度，长度

（4）属性的访问

对象.属性名

对象[‘属性名’]

如果属性不存在返回undefined,如果属性名含有特殊符号只能使用第二种方式。

练习：创建一个图书的对象，包含有图书的编号，名称，价格，作者；修改图书的价格，添加图书的出版社属性，打印整个对象

（5）使用内置构造函数来创建对象

new Object().

创建一个空对象，需要单独添加每个属性

（6）遍历属性

使用循环，依次访问对象中的每个属性

for(var k in 对象){

k 每个属性名

对象[k] 每个属性名对应的属性值

}

课后任务：

（1）复习今天内容，整理思维导图

（2）创建对象（字面量），包含有一组学生的成绩属性，遍历属性，计算出总成绩和平均成绩

（3）预习JS第七天，数组

检测属性是否存在

1.对象.属性名===undefined true--不存在 false--存在

对象.hasOwnProperty('属性名')true-存在 false-不存在

(‘属性名’) in 对象 true -存在 false--不存在

练习：创建一个商品对象，包含有商品的编号，名称，价格；如果价格属性存在打九折，如果产地属性不存在则添加该属性

2.对象中的方法

var ran={

ename:‘然哥’，

paly.function(){

this 指代当前所在的对象

}

}

ran.paly()；// 调用

方法对应的是一个函数，需要调用

练习：创建一个圆对象，包含属性有半径和圆周率，添加两个方法，分别是计算圆的周长和面积，最后调用

练习：创建一个计算机对象，添加两个方法，传递任意两个数字，分别实现计算相加和相减并返回结果

3.数据的存储

原始类型：直接将数据存储在栈内存中

引用类型：在堆内存中保存数据，会生成一个地址，把地址保存在栈内存的变量中。

null：空地址，用于销毁对象，一旦一个引用类型（对象）不被任何的地址所指向，就会自动销毁。

4.数组

数组是数据的集合，每一个数据称为元素

（1）字面量创建

[元素1，元素2]

练习：创建数组，包含有一组商品的名称；

创建数组，包含有一组员工工资。

（2）元素的访问

数组[下标]

下标：JS自动为每个元素添加的编号，从0开始的整数

练习：创建一个数组，包含有一组城市的名称修改下标为二的元素，在数组最后添加若干个新的元素，最后打印数组

（3）数组的长度

数组.length

查看当前数组中元素的数量

在末尾添加一个新的元素 数组[数组.length]=值

练习：创建一个空数组，使用数组的长度添加若干个国家

（4）内置构造函数创建数组

new Array（元素1，元素2，元素3..）

new Array（3） 创建一个数组，初始化长度为3，可以添加多个

练习：分别使用两种方式创建数组

创建数组，包含有一组汽车名称

创建数组，初始化篮球场上的五个位置

（5)数组的分类

索引数组：以0及以上的整数作为下标

关联数组：以字符串作为下标，需要单独添加元素

对象和数组的区别

两者都可以保存有多个数据

对象保存一条数据，有多个属性，例如有一个员工有编号，姓名，性别..

数组保存一组数据，有多个相同的数据，例如一组商品

练习：创建一个数组，包含有一组商品，每一个商品包含的属性有标题，价格

（6）数组的遍历

依次访问数组中的每个元素

for-in

for(var k in 数组）{

k 代表每个元素的下标

数组[k] 每个元素

}

循环

for(var i=0,i<数组.length,i++){

i 代表每个元素的下标

数组[i] 每个元素

}

练习：创建数组包含有一组员工姓名，将姓名为‘然哥’的数据修改为‘ran’

（7）API

API:应用程序编程接口，JS提供的一些函数或者对象中的方法

toString() 将数组转为字符转，元素之间按照逗号分隔

join(‘ ：’)将数组转为字符串，元素之间默认按照逗号分隔，可以按照指定字符分隔

concat(arr2,arr3) 拼接多个数组，arr2,arr3是要拼接的数组

课后任务：

①复习今天内容，整理思维导图

②练习

创建数组保存多个元素，使用遍历数组翻转数组中的元素

['a','b','c','d'] ['b','c','d','a']

创建数组保存一组数字，将数字进行从小到大的排序

冒泡排序——循环嵌套

1.数组

（1）API

如何学习API 作用、需要哪些参数、返回值

reverse() 翻转数组中的元素，返回翻转后的数组

sort() 对数组进行排序，默认是按照Unccode 编码从小到大排序

sort(function(a,b){

return a-b;//数字从小到大

return b-a;//数字从大到小

})

slice(start.end) 截取数组中的元素，start开始的下标，end结束的下标，不包含end本身，如果end为空截取到最后；返回截取到的元素

练习：创建数组包含a-h，每个字母是一个元素，分别截取b-d，f；把两部分拼接起开

练习：创建数组包含a-h，删除d e，替换f为m，在下标1的位置插入z

push() 在数组的末尾添加元素，返回数组长度

pop() 删除数组末尾的一个元素，返回删除的元素

unshift() 在数组的开头添加元素，返回数组长度

shift() 删除数组开头的一个元素，返回删除的元素

（2）二维数组

数组中的元素还是数组

数组中的元素还是数组，用于对于一组数据进行二次分类

[[],[],[]]

访问形式 数组[下标]，[下标]

练习：创建一个数组包含有多个城市，对这组城市进行分类，同一个省份下的城市放到一起

字符串对象

创建对象：是为了将原始类型的数据像引用类型属性据一样，具有属性和方法

JS有三种包装对象：Number对象，String对象，Boolean对象

nuw String() 将数据转为字符串，返回对象类型

String() 将数据转为字符串，返回字符串

练习：创建数组包含一组数据，分别使用数组下的API和String函数将数组转为字符串，对比两种方法的区别

（1）转义字符\

转换字符本身的意义

\'将具有特殊意义的引号转为普通字符

\n a\nb 将n转为了换行符

\t a\tb 将t转为了制表符，多个连续空格，Tab键的效果

（2）API

length 获取字符串的长度

charAr() 获取字符串中下标对应的字符，也可以使用数组的写法 字符串[下标]

练习：声明变量保存字符串javascript，遍历字符串，统计A字符出现的次数

indexOf() 查找字符（字符串）第一次出现的下标，找不到返回1

lastindexOf() 查找字符(字符串)最后一次出现的下标，找不到返回-1

练习：声明变量保存用户输入的邮箱，如果不含有@，打印‘邮箱格式不正确’，否则打印‘格式正确’

toUpperCase()将英文字母转为大写

toLowerCase()讲英文字母转为小写

slice(start.end) 截取字符串，start开始的下标，end结束的下标，end为空截取最后，不包含end本身；如果是负数表示倒数

练习：声明变量保存邮箱，分别截取出邮箱的用户名和域名

substr(syart,count) 截取字符串，start 开始的下标，count截取的长度，count为空截取到最后，如果下标为负数表示倒数

练习：声明变量保存一个人的身份证号，分别截取出出生的年月日和性别

142602199611152011

split() 将字符串按照指定的字符串切割为数组

web.2009.js.jpg

截取出文件的后缀名

匹配模式（了解）

用于字符串的查找和替换

search(/ran/i) 查找满足条件的第一个

match(/ran/ig) 查找满足条件的所有的，格式为数组

replace(/ran/ig,‘xxx’ 查找并替换

/ /

i - ignore 忽略大小写

g - global 全局查找

课后任务：

①复习今天内容，整理思维导图

②练习：

将一位英文中每个英文单词的首字母转为大写，其余小写 （how are you) (how are you )

字符串转数组

③预习JS第九天 Data对象 Math对象

1.Math对象

不需要创建，可以直接使用

PI 圆周率

abs（） 绝对值

ceil（） 向上取整

floor（） 向下取整

round （）四舍五入取整

pow（x，y）计算x的y次方

max（） 获取一组数字最大值

min（） 获取一组数字最小值

2.Date对象

用于对日期时间的存储和计算

(1)创建

new Date('2020/10/22 11:30:50');

new Date(2020,9,22,11,20,30);//月份0-11 1月-12月

new Date();当前操作系统的时间

new Date(100) //储存的是距离计算机元年的毫秒数，对应的日期时间

（2）获取

getFullYear()

getMonth()+1

getDate()

getHours()

getMinutes()

getSeconds()

getDay() 星期范围0-6 对应星期日-星期六

getTime() 距离计算机元年毫秒

练习：创建Date对象，保存当前操作系统的时间，打印出以下格式

‘xxxx年xx月xx日 xx时xx分xx秒 星期四’

练习：计算距离2021年元旦还有x天x小时x分x秒

（3）转为本地字符串格式

用于调试­­­，有兼容性问题，不同的解释器显示可能不同

d.toLocaleString()

toLocaleDateString()

toLocaleTimeString()

（4）修改

sgetFullYear()

setMonth()

setDate()

setHours()

setMinutes()

(5)Number对象

new Number（） 将数据转为数值，返回对象

Number（） 将数据转为数值，返回数值

toFixed（n） 保留小数点后n位

Boolean 对象

new Boolean（） 返回对象

Boolean（） 返回布尔型

强制将数据转为布尔型

错误处理

语法错误（SyntaxError)：不符合语法规范，例如出现中文的符号，出现后不执行任何代码

引用错误（ReferenceError）： 出现了未声明的变量，影响后续的代码

类型错误（TypeError）：使用了不存在对象中的方法或者函数，影响后续的代码

范围错误（RangeError）：提供的值超出了JS规定的范围，影响后续的代码

自定义错误（shrow 错误内容）：影响后续代码执行

try{

尝试执行，可能出现错误，一旦出现错误就会执行catch中的语句

}catch(err){

err, 形参，接收try出现的错误信息。

执行解决错误的代码

}

课后任务：

1.复习今天内容，整理思维导图

2.练习：

双色球

红球：在1-33之间随机取6个，不能重复，按照从小到大的排序

篮球：在1-16之间随机取1个，可以和之前的红球数字重复

把这组数字放入一个新数组中

3.预习Nodejs 第一天

模块

ES6

EcmaScript 6

ES5215-2021

（1）块级作用域

let 声明的变量不存在变量的提升，不允许在同一个作用域下重复声明同一个变量

{ }大括号之间就是块级作用域，包含有if.else.while.for...等大括号下都是块级作用域下，let声明的变量属于局部变量，不能被块级作用域以外访问，防止污染全局

练习：使用for循环，计算1-100之间所有整数的和，使用let声明

（2）函数增强

可以给参数设置默认值

function add(a,b,c){

//如果调用函数的时候，没有传递实参，则会使用参数的默认值

}

（3）箭头函数

简化了匿名函数的写法

()=>{ }

arr.sort( (a,b)=>{

return a-b

});

如果箭头函数的函数体中只有一行代码，并且是return形式，则可以简化未为

sort((a,b)=>a-b)

练习：使用函数表达式创建函数，匿名函数使用箭头函数形式，计算任意两个数字相加的和，并返回结果

（4）模板字符串

`模板字符串 ${js表达式}`

模板字符串之间可以回车换行，解决了字符串的拼接问题

练习：假设从数据库中获取到了一个员工的数据，格式为对象，包含有员工的编号，姓名，性别（1/0）工资，打印出一下格式

编号：

姓名：

性别：男/女

工资：xxx.xx元

Node.js 概述

（1）对比JS

JS运行在客户端浏览器，存在多款浏览器，有兼容性问题；Node.js运行在服务器端，只有一种解释器，没有兼容性问题

两者有共通的内置对象，自定义对象；以及各自的宿主对象

JS用于开发浏览器端的交互效果，Node.js用于服务器端的开发，例如数据库访问，WEB服务器的创建

（2）网址

官网：[www.nodejs.org](http://www.nodejs.org/)

中文翻译 [www.nodejs.cn](http://www.nodejs.cn/)

node -v 查看版本号

node 回车 进入交互模式

两次ctrl+c 退出交互模式

2.全局对象

（1）global

用于查看一个变量和函数是否为全局变量或者全局函数

global.变量 global.函数名称（）

在Node.js脚本文件中不是在全局作用域下，里面的变量和函数都是局部的，可以有效防止全局污染

在JS脚本文件中是在全局作用域下，全局对象叫window,里面的变量和函数都是全局的，会存在全局污染

练习：编写脚本文件03\_global.js,声明变量，创建函数，查看里面的变量和函数是否为全局的

练习：编写文件04\_window.html和04\_window.js把js文件嵌入到html中，在js文件声明变量a和创建函数fn，使用全局对象window访问变量和调用函数

（2）console 对象

log() 打印日志

inof() 打印消息

warn() 打印警告

error() 打印错误

time() 开始计时

timeEnd() 结束计时

开始计时和结束计时提供的参数要保持一致

练习：使用for,while,do-while 循环100000次，查看各自的运行时间

（3）process 对象

进程：计算机上每一个软件启动后都是一个进程

Process.arch 查看当前CPU的架构

process.platform 查看当前的操作系统

process.version 查看当前Node.js的版本号

process.pid 查看当前进程的编号

process.kill(进程编号) 结束指定

（4）Buffer 对象

Buffer：缓冲区，缓冲器，内存中储存数据的区域，常用于储存网络传输时的资源

let buf=Buffer.alloc(5,‘abcde’) //创建Buffer 空间并储存数据，每个英文字符占1个字节，汉字占3个字节

//将buffer转为字符串

buf.toString()

String(buf)

3.模块

Node.js下，每个模块是一个功能体。

分为自定义模块，核心模块，第三方模块

每一个文件就是一个模块

require:是一个函数，用于引入另一个模块

module.exports:导出的对象，默认是一个空对象，如果要导出内容，放入到该对象

课后任务

①复习今天内容，整理思维导图

②练习：

创建主模块（main.js)和功能模块（circle.js)，在功能模块下创建两个函数，传递参数为半径，分别计算出圆的周长和面积。在主模块下引入功能模块后，调用两个函数

③预习Node.js第二天， 文件系统模块 查询字符串模块

1.模块

module:当前模块的对象

module.exports 当前模块的到处对象

require: 是一个函数，用于引入其他的模块，会得到导出的对象

\_\_filename directory 当前模块的绝对路径

\_\_dirname 当前模块的绝对路径+模块名称

模块的分类

以路径开始

2. 目录模块

文件模块

require('./circle.js')  
 常用于引入自定义模块

require('/02\_dong')

首先会到目录下寻找package.json文件中main对应的文件；如果没有会自动找index.js

不以路径开头

文件模块：

require('querystring')

用于引入官方提供的核心模块

requir('ran')

首先会到当前目录下的node\_modules 目录中寻找ran 目录，如果找不到会一直往上一级的node\_modules中寻找ran目录，常用于引入第三方模块

json文件：属于一种数据文件，通常放数组和对象，对象的属性名必须使用双引号，字符串必须使用双引

练习：在03\_1.js文件中引入同一级的目录模块03\_2，目录中含有文件index.js，导出一个函数，计算任意三个数字相加的和

3.包和npm

包：package,就是目录模块，第三方模块的叫法

npm：用于管理包的工具，包含有下载安装、升级、卸载、上传...

npm会在node.js安装的时候附带安装

npm -v 查看npm的版本号

（1）切换命令行的路径

① cd 粘贴路径 回车

如果涉及到盘符的变化，需要添加

盘符名称： 回车

② 在指定目录下的空白区域，按住shift，单机鼠标右键，选择 ‘在此处打开powershell窗口’

(2)下载

网址：[www.npmjs.com](http://www.npmjs.com/)

npm install 包的名称 回车

将包自动放到node\_modules目录中，这个目录不存在自动创建，可能会下载一些其他依赖的包。同时还会生成一个package-lock.json的文件，用于记录包的版本号

4.查询字符串模块

查询字符串：浏览器像WEB服务器发送请求，传递数据的一种方式，位于URL中问号后的部分

keyword=汽车&enc=utf-8

查询字符串模块用于操作查询字符串的工具模块

parse() 将查询字符串解析为对象

练习：获取以下查询字符串商品的名称和价格

pname=apple&price=5199

5.URL模块

URL:统一资源定位，网络上的任何的资源都有对应的URL来访问

<http://www.codeboy.com:9999/products.html>？kw=dell

协议 域名（IP地址） 端口 文件在服务器上的路径 查询字符串

URL模块用于操作URL

parse() 将一个URL解析为对象，可以得到URL中的各个部分

练习：获取以下URL中查询字符串传递的数据。

[https://www.tmooc.cn:443/web/2009.html?cid=3 HYPERLINK "https://www.tmooc.cn/web/2009.html?cid=3&course=nodejs" HYPERLINK "https://www.tmooc.cn/web/2009.html?cid=3 HYPERLINK "https://www.tmooc.cn/web/2009.html?cid=3&course=nodejs"& HYPERLINK "https://www.tmooc.cn/web/2009.html?cid=3&course=nodejs"course=nodejs" HYPERLINK "https://www.tmooc.cn/web/2009.html?cid=3&course=nodejs"& HYPERLINK "https://www.tmooc.cn/web/2009.html?cid=3&course=nodejs" HYPERLINK "https://www.tmooc.cn/web/2009.html?cid=3 HYPERLINK "https://www.tmooc.cn/web/2009.html?cid=3&course=nodejs"& HYPERLINK "https://www.tmooc.cn/web/2009.html?cid=3&course=nodejs"course=nodejs" HYPERLINK "https://www.tmooc.cn/web/2009.html?cid=3&course=nodejs"course=nodejs](https://www.tmooc.cn/web/2009.html?cid=3&course=nodejs)

6.定时器模块（timer)

(1)一次性定时器

开启

let timer=setTimeout(回调函数，间隔时间）

当间隔时间到了，会调用一次回调函数，单位是毫秒

清除

clearTimeout(timer)

(2)周期性定时器

let timer=setInterval(回调函数，间隔时间）

每隔一段时间，调用一次回调函数

清除

clearInterval(timer)

(3)立即执行的定时器

let timer=setImmediate(回调函数）

清除

clearImmediate(timer)

process.nextTick(回调函数）

7.文件系统模块（fs）

文件包括文件形式和目录形式

（1）创建目录

fs.mkdirSync(目录的路径）

（2）移除目录

fs.rmdirSync(目录的路径）

课后任务

（1）复习今天内容，整理思维导图

（2）练习：

开启定时器，3秒后创建1-10，10个目录；创建成功后开启一次性定时器，将创建的10个目录10秒后删除

（3）预习第三天 同步和异步 http 协议

1.文件的系统模块

（1）查看文件的状态

statSync（文件的路径）/stat（文件的路径，回调函1数）

回调函数：异步操作通过回调函数获取结果

err：可能产生的错误

s：具体的文件状态

s.isFile()是否为文件

s.isDirectory()是否为目录

（2)读取目录

readdirSync(文件的路径）/readdir(文件的路径，回调函数）

获取的结果为数组形式

(3)写入文件

writeSync(文件的路径，写入的数据)/writeF(文件的路径，写入的数据，回调函数）

如果文件不存在，则会创建文件然后写入数据

如果文件已经存在，则会在原来的基础上覆盖写入

练习： 使用同步方法去往2.txt中写入数据‘tedu’

（4）追加写入

writeSync(文件的路径，写入的数据)/writeF(文件的路径，写入的数据，回调函数）

如果文件不存在，则会创建文件然后写入数据

如果文件已经存在，则会在文件的末尾追加写入数据

练习：使用异步方法往文件3.txt中写入数据web2009

（5）读取文件

readFileSync(文件的路径）/readFile(文件的路径，回调函数）

读取的结果为buffer数据

（6）删除文件

unlinkSync(文件路径)/unlink(文件的路径，回调函数)

练习：使用异步删除1.txt

（7）检测文件是否存在

existsSync(文件的路径）

true（存在） false-（不存在）

练习：如果文件2.txt存在，则同步删除文件，如果目录mydir2不存在，则同步创建该目录

（8）拷贝问价

copyFileSync(源文件，目标文件）/copyFile(源文件，目标文件,回调函数)

（9）文件流和事件

createReadStream(文件的路径） 创建可写入的流对象

createWriteStream(文件的路径）创建可写入的流对象

事件格式

on（'事件名称’，回调函数）

事件名称是固定用法，一旦出发事件自动执行回调函数

pipe() 管道，可以将读取的流添加到写入流

2.同步和异步

同步：在执行的过程中按照顺序执行，会组织后续代码的执行，如果获取结果通过返回值

异步：在执行的过程中如果有异步操作，会在一个独立的线程执行，不会阻止主程序后续代码的执行；通过回调函数来获取结果

练习：使用异步方法创建目录mydir

//madir

http浏览器和web服务器之间的通信协议

（1）通用的头信息

Request URL:请求的URL,向服务器端请求的内容

Request Method:请求的方法，get/post

Status Code：响应的状态码

1\*\*正在响应，还没结束

2\*\*成功响应

3\*\*响应的重定向，跳转另一个URL

4\*\*客户端请求错误

5\*\*服务器端错误

（2）响应的头信息（Response)

Content-Type:响应的内容类型，html文件对应 text/html

location:要跳转的URL，需要结合着状态码3\*\*使用

（3）请求的头信息（Request)

（4）请求主体（可有可无）

4.http模块

可以创建web服务器，接收浏览器的请求并作出响应

const app=http.createServer();//创建web服务器

app.;listen(8080);//设置端口，用于访问web服务器

//添加事件，通过事件来接收请求并作出响应

app.on('request',(req,res)=>{

req 请求的对象

req.url 获取请求的URL,对应端口号的部分

req.method 获取请求的办法

res 响应的对象

res.write() 设置相应的内容

res.writeHead() 设置响应的状态码和头信息，第二个参数头信息可以为空

res.end() 结束并发送响应

});

课后任务：

（1）复习今天内容，整理思维导图

（2）练习：

根据浏览器的请求，作出对应的响应

/index 响应 '<h2>这是首页</h2>'

/study 跳转 [http://www.tmoo.cn](http://www.tmoo.cn/)

/list 响应文件1.html （提示：先同步读取文件内容，把读取的内容作为要响应的内容）

其他 响应 404 not found

(3)预习第4天 express 第三方包