复习

JS解释器

开发环境 服务器端 nodejs

浏览器

var $a\_1=2,b=5;

var c; //undefined

const pi=3.14;

学习一门编程语言的基本步骤

(1)了解背景知识：历史、现状、特点、应用场景

(2)搭建开发环境：编写hello world

(3)常量和变量

(4)数据类型

(5)运算符

(6)逻辑结构

(7)通用小程序

(8)函数和对象

(9)第三方框架、库

(10)实用的项目

1.数据类型

数据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

(1)数值型

分为整型和浮点型

整型在内存中占4个字节，浮点型占8个字节

八进制 以0开头

十六进制 以0X开头，a~f 代表10~15 不区分大小写

0XFF -> 255

浮点型分为定点小数和指数型小数

3140 3.14e3 指数

-0.314 3.14e-1

3.14

typeof 检测数据类型

(2)字符串型

数据被引号所包含就是字符串型，不区分单双引号

查看任意一个字符的Unicode码

'然'.charCodeAt() //28982

(3)布尔型

true/false

在程序中表示真或者假

用于一些比较的结果，还有一些是否的结果，例如是否登录，是否注册，是否在售...

(4)未定义型

声明了变量未赋值，结果就是undefined

(5)空——null

只有一个值null，将来会和引用类型一起使用

2.数据类型转换

(1)隐式转换

在运算的过程中，自动的发生转换

1)数字+字符串 数字被转成字符串

2+'3' //'23'

2)数字+布尔型 布尔型转成数字 true->1 false->0

3+true //4

3+false //3

3)布尔型+字符串 布尔型转成字符串

'5'+true //'5true'

JS中加号(+)的作用

执行加法运算

执行字符串间的拼接

减法(-)、乘法(\*)、除法(/)隐式转换

将运算符两端的数据转为数值型，如果转换失败，返回NaN(Not a Number)，不是一个数字，任何值和NaN执行加减乘除都会返回NaN；**自动调用了Number转换成数值型**。

练习：查看以下程序的运行结果

var num1=3,num2=true,num3='tedu';

console.log(num1+num2+num3);//'4tedu'

console.log(num2+num3+num1);'truetedu3'

console.log(num3+num1+num2);'tedu3true'

(2)强制转换

1)强制转为数值型

Number()

|  |
| --- |
| Number('2') //2  Number(true) //1  Number('2a') //NaN  Number(undefined) //NaN  Number(null) //0 |

2)强制转换为整型

parseInt()

常用于将字符串或者数字转为整型，其它的数据返回NaN，如果要转换的字符串开头是非数字，则返回NaN

3)强制转为浮点型

parseFloat()

和parseInt的用法基本一致，只是转换为浮点型数据

'2.5a' 'a2.5'

4)强制将数值型和布尔型转为字符串型

toString()

var num=10;

num.toString(); // '10'

3.运算符

表达式：由运算符连接的操作数据，所组成的形式。

运算符分为算术运算符、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)算术运算符

+ - \* / % ++ --

% 取余

++ 自增，在原来的基础之上加1

-- 自减，在原来的基础之上减1

console.log(a++); 先打印a的值，然后执行自增

console.log(++a); 先执行自增，然后打印a的值

练习：

var num=3;

console.log( num-- + --num );

(2)比较运算符

> < >= <= == != ===(全等于) !==(全不等于)

返回一个布尔型的值

== 比较两个值是否相等

=== 不仅比较值，还会比较类型是否相等

|  |
| --- |
| 如果数据类型不同，会发生数据类型转换 |

3>'10' 字符串转为数值 '10'->10

'3'>'10' 两个字符串比较的是首个字符的Unicode码

3>'10a' //false

3<'10a' //false

3=='10a' //false

NaN==NaN //false

NaN和任何值比较(> < == >= <= ===)，包括自身，都返回false。

(3)逻辑运算符

&& 并且 关联的两个条件都满足，结果是true，否则false

|| 或者 关联的两个条件满足其一，结果是true，否则false

! 非 取反向 !true-> false !false -> true

**逻辑短路**

&& 当第一个条件为false的时候，就不需要再执行第二个条件

|| 当第一个条件为true的时候，就不需要再执行第二个条件

**注意事项：**逻辑短路无需关注整个逻辑运算的结果是true还是false，重点是看是否会执行第二个条件

练习：以下程序是否会保错

|  |
| --- |
| var num=3;  num>5 && console.log(a);  num<1 || console.log(a); |

练习：声明变量保存年龄，如果满18岁，打印'成年人'

(4)位运算符(了解)

在执行运算的时候，计算机会把数据转成二进制进行运算

1 10 11 100 101 110 111 1000 1001 1010

2 4 8

100101=100000+100+1

32 + 4 + 1 = 37

1010110=1000000+10000+100+10

64 + 16 + 4 + 2=86

25=16+8+1

10000+1000+1=11001

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(^) 上下两位不同为1，相同为0

按位右移(>>) 删除二进制的最后一位

按位左移(<<) 在最后一位的后边补0

课后任务

(1)复习今天内容，整理思维导图

(2)练习

声明变量保存一个年份，判断这个年份是否为闰年，结合逻辑短路，如果是闰年，打印闰年

闰年：标准? 4年一闰，100不闰，400再闰

(3)预习if if-else if-else嵌套 switch-case