课堂笔记

> 表达式： 由运算符组成，如 100+200 , true && 100 > 50

1. 数据类型转换

1.1 JavaScript 基本数据类型总结

- 数值 number
- 字符串 string
- 布尔值 boolean
- 空 null
- 未定义 undefined

1.2 显示类型转换（强制类型转换）

> 使用转换函数来进行转换 Number()、String()、Boolean()

① 其他类型转为 number

- 使用 Number() 函数

```
string -> number:
    纯数字字符串，转为对应的数字。  "123"->123, "123.29"->123.29
    纯数字字符串两端如果是空格的话，忽略空格转为对应的数字。  " 123"->123, " 123   "->123
    空字符和纯空格组成的字符串，转为0。  ""->0, "   "->0
    其他情况，转为NaN。  "123abc"->NaN, "12.34.101"->NaN

boolean -> number:
    true -> 1
    false -> 0

null -> number:
    转为 0

undefined -> number:
    转为 NaN
```

- parseInt()、parseFloat() 用于字符串转为数字（截取字符串里面的数字）

```
parseInt() 把纯数字字符串或者纯数字开头的字符串，转为对应的整数，数字开头的字符串会截取;忽略小数，转为整数。 其他情况都是NaN
parseFloat() 把纯数字字符串或者纯数字开头的字符串，转为对应的数字，数字开头的字符串会截取;保留小数。 其他情况都是NaN
```

> 注意：纯数字两端是空格，会忽略空格，当做纯数字字符串

② 其他类型转为 string

- String() 函数

```
根据原来的值，创建一个新字符串返回。
```

- 数据调用 toString() 方法

```
根据原来的值，创建一个新字符串返回。
null 和 undefined 是无法调用 toString() 方法
```

③ 其他类型转为 boolean

- Boolean() 函数

```
0、NaN、空字符、null、undefined 转为 false
其他情况都转为 true
```

1.3 隐式类型转换（自动类型转换）

> 看数据处在怎样的运算环境中。
>
> 隐式类型转为的规则同 Number()、String()、Boolean() 相同

① number 的运算环境

```
如果运算符是 +、-、*、/、%、** ,运算符两端的操作数自动转为 number 类型
但是，+ 如果两边的操作数有一个是字符串，那么 + 就是字符串连接符了，意味着就不是数字的运算环境了。
```

② string 的运算环境

```
+ 运算符，操作数里面只有有一个字符串，就是字符串的运算环境，另一个操作自动转为字符串
```

③ boolean 的运算环境

```
if 语句的条件判断  if (100){}  if(undefined){}   if (100>20){},  if (100-20) {}
while 条件
逻辑运算符组成的表达式，在计算的过程中会把其他数据转为boolean
```

2. 逻辑运算符 && 和 ||

2.1 逻辑与 &&

逻辑与 and 并且
要求运算符两侧的条件都成立，最终结果才成立。
运算符两侧都是 true 最终结果才是 true

2.2 逻辑或 ||

逻辑或 or 或者
要求运算符两侧的条件只要成立一个，最终结果就成立。
运算符两侧都是只有一个是true，最终结果就是true。

2.3 逻辑运算符组成的表达式的取值

运算的时候，逻辑运算符两边的操作数转为布尔值，但是最终返回的是原值。

-- 逻辑与取值规则：
---- 如果第一个值是true(或者可以转为true)，则返回第二个操作数值作为表达式的值。
---- 如果第一个值是false(或者可以转为false)，则返回第一个操作数的值作为表达式的值。

-- 逻辑或取值规则：
---- 如果第一个值是true(或者可以转为true)，返回第一个操作数的值作为表达式的值。
---- 如果第一个值是false(或者可以转为false)，返回第二个操作数的值作为表达式的值。

3. 运算符优先级

https://developer.mozilla.org/zh-CN/docs/Web/JavaScript/Reference/Operators/Operator_Precedence

加括号优先级最高
一元运算符优先级比较高 如：++、--、！ 等
逻辑运算符&&和||优先级比较低
优先级相同，从左到右