第三天

- 函数可以像对象一样通过打点或者其他方式 添加一个属性名和属性值
- 并且函数也可以通过打点等方式拿到这个属性值
- 每个函数都有一个自带的属性叫name 匿名函数的name属性的值是空字符串

创建函数的两种方式

- 1.创建函数 第一种方式创建函数 `function name(val) {}`
- 2.第二种方式创建函数 `var variable = function () {}`

函数的执行 参数（形参，实参）

- `name()`
  - 形参是个变量 代表传过去的值
  - 实参 真实传到函数里面的数据 是一个具体值
  - 函数执行的时候，如果实参是一个变量或者表达式，会先获得变量代表的值，或先执行表达式获取结果，再把结果传给形参
- 函数执行的时候，如果实参是一个变量或者是表达式,会先获取变量代表的值，或者先执行表达式获取最终的结果。再传给函数

arguments

- `arguments` 类数组 像一个数组可以通过 `[]` 加数字的方式拿到里面的值,但是没有数组有的方法,就是实参的集合,这里面可以不需要形参。
  - `arguments` 不管函数有没有形参， `arguments` 都是完整的 所有的 实参的 集合

匿名函数 自执行函数

- 创建方式
  - `var fn4 = function () {`
    `创建变量，把函数赋值给变量`
    `}`
  - `(function () {})()`
    - 第一步两个（）（）
    - 第二步在第一个小括号里放入 `function () {}`

        - 大括号里面写需要的代码
        - 第二个（ ）里面是实参
        - 切记这样的自执行函数前面的表达式必须有分号结束
        - ctrl+f 当前文件搜索
    - 第三种 !或~
        - 如果前面的代码块没有分号结束，可以用各种符号开头 例如：`% ^ &`
        - 如果前面代码已经用分号结束，就只能用`！`或`~`
- 自己执行自己
- 正常函数，我们需要一个`fnName()`函数名加小括号的方式去执行函数

switch case

- `switch case`是三个等号的绝对比较
  `switch (判断的值) {`
  `case 对比的值 :`
  `alert('比较成功后执行的代码')`
  `break;`
  `default:`
  `console.log()`
  `}`
    - 相当于`if else if else`里面的else`
    - 如果所有对比都不成立，就会执行default里面的代码块
    - 如果前面有对比成功的 就不会走到这里了

for循环

- 语法：`for (var i = 0; i < arr.length; i++) {`
  `console.log(arr[i])`
  `}`
  `
    - 1. var i = 0; 创建变量i 默认值设为0
    - 1. i < arr.length; (判断)
    - 1. 如果第二步是true 运行代码块 如果为false就结束循环
    - 1. i++ 意思就是i自增 i = i + 1

```
var arr = ['xiaobai', 'dabai', 'yuanxiao']
// console.log(arr[0], arr[1], arr[2])

for (var i = 0; i < arr.length; i++) {
    // var i = 0; 0 < 3; 代码执行; i++
```

```
    // i 等于 1;  1 < 3; 代码执行; i++
    // i 等于 2;  2 < 3; 代码执行; i++
    // i 等于 3;  3 < 3; 结束循环
    console.log(arr[i], i)
}
```

- 例子：一次循环 在偶数的时候打印出a 奇数的时候打印出b
  - `var arr1 = [1, 2, 3, 4]`
    `var flag = true`
    `for (var i = 0; i < arr1.length; i++) {`
    `if (flag) {`
    `console.log('a')`
    `flag = false`
    `} else {`
    `console.log('b')`
    `flag = true`
    `}`
    `}`

Break 和 Continue 语句

- break 语句用于跳出循环。
  - break 语句跳出循环后，会继续执行该循环之后的代码（如果有的话）：
  - break 语句（不带标签引用），只能用在循环或 switch 中。
- continue 用于跳过循环中的一个迭代。
  - continue 语句中断循环中的迭代，如果出现了指定的条件，然后继续循环中的下一个迭代。
  - continue 语句（带有或不带标签引用）只能用在循环中。

`Math.max`是一个函数`Math.max()`是函数执行，函数的结果是获取一个最大值

js算法

- `Math.random()` 随机数
- `Math.max(7.25,7.30)` 比较两个数的最大值
- `Math.E` 常数
- `Math.PI` 圆周率
- `Math.SQRT2` 2 的平方根

- `Math.SQRT1_2` 1/2 的平方根
- `Math.LN2` 2 的自然对数
- `Math.LN10` 10 的自然对数
- `Math.LOG2E` 以 2 为底的 e 的对数
- `Math.LOG10E` 以 10 为底的 e 的对数

- `Math.round(4.7)` 四舍五入
-