JavaScript重点难点摘记

1.引入css文件用link标签(元素);用href属性

<link rel="stylesheet" href="css文件路径">

2.webstorm内容自动换行

在行数上右键，选择use soft wraps

3.重新安装Chrome浏览器

windows找不到Chrome浏览器的时候,重新下载安装;

4.引入js文件用script标签,用src属性

<script src="js文件路径"></script>

5.document.write()的方法

1. 概念: 常用来网页向文档中输出内容

例如: document.write('我爱学习-我喜欢学习,我很高兴和大家分享学习经验与技巧... ...')

2. 以变量的方式输出

例如: var str = 'hello world';//声明一个变量str

document.write(**str**);//把变量str输出

3. 输出变量和字符串的组合

备注:字符串用加号"+";

字符串用双引号""括起来

例如: var str = "hello world";//声明一个变量str

document.write(str + " " + "My name is beibeixue");//在浏览器的可视窗口输出hello world My name is beibeixue

4.输出html标签

document.write("我爱学习") =>网页中展示的是粗体"我爱学习"

6.判断一个变量属于什么数据类型

console.log(typeof(n));//n是变量 变量不能加引号

7.全局变量和私有变量的概念

1)全局变量: 在全局作用域下声明(预解释的时候)的变量,是全局变量;

2)在[私有作用域中声明的变量]和[形参赋值]的变量,都是私有变量;

8.判断是全局变量还是私有变量

3)在私有变量中,但凡有[形参赋值]或者[私有作用域中声明的变量],那么这个私有作用域中的变量都是私有变量;

4)如果私有作用域下没有形参赋值或者提前声明,那么这里的变量是全局作用域下的变量,如果这个变量进行了运算,那么运算的结果会影响全局作用域中变量的结果;

9.函数执行fn()过程

1)形成一个私有作用域

2)如果有形参,先给形参赋值;

3)进行私有作用域中的预解释;

4)私有作用域中的代码从上到下执行;

10.闭包

概念: 函数形成了一个私有作用域保护了里面的私有变量不受外界干扰(外面修改不了私有的,私有的也修改不了外面的);=>这种机制叫"闭包";

11.全局变量的细节问题: 带var和不带var的区别?

区别一:

带var: 进行预解释,在[赋值]的[前面]执行不报错;

不带var: 不进行预解释,在[赋值]的[前面]执行会报错;

区别二:

var num = 12; =>全局变量,赋值12;首先相当于给window增加了一个全局变量;同时还给window增加了一个属性名window.num并赋值12;=>window.num的值是12;

num2 = 10; =>window的属性,值是10;注意,这里是num2=10;并不是var num2=10;相当于给window增加了一个num2的属性名,属性值是10;(window的变量,也是window的属性;)

1)首先看num2是不是全局变量(在前面有没声明var过);

2)如果num2不是全局变量,但num2是window的属性;相当于输出的是window.num2的属性值;

细节三:

1)全局作用域声明了变量并赋值,私有作用域也声明了变量并赋值;在私有作用域赋值前输出,值是undefined;在全局作

用域下声明赋值后输出,值是当时所赋的值;

2)私有作用域中出现一个变量不是私有的,则往上级作用域进行查找,上级没有则继续向上查找,一直找到window为止,如果window下没有;1.获取值的结果:会报错total is not defined at fn;2.设置值total=100;=>相当于给window增加了一个属性名total,属性值是100;

JS中如果在不进行任何**特殊处理**的情况下,上面的代码报错,下面的代码都不再执行了;有一种机制可以处理 =>trycatch 可以查阅了解下!

12.预解释细节

1)预解释的时候不管条件是否成立,都要把带var的进行提前声明;

2)预解释的时候,只解释等号左边的,等号右边的值不参与预解释;

3)函数体内的return[下面]的代码虽然不再执行,在需要进行预解释;return[后面]跟着的是我们返回的值;所以不进行预解释;

4.在预解释的时候,如果名字已经声明过,不需要重新声明;但需要重新赋值;

在JS中如果变量的名字和函数的名字重复了,也算冲突;(只能保留一个fn,只不过曾经fn存的是一个数字,后来存的是函数值而已);

自执行函数在全局作用域下不预解释; 也不赋值;

5.window预解释

声明加定义 fn=xxxfff000;

声明 var fn;(不需要重新声明)

声明(不重复进行)加定义 fn=xxxfff111;

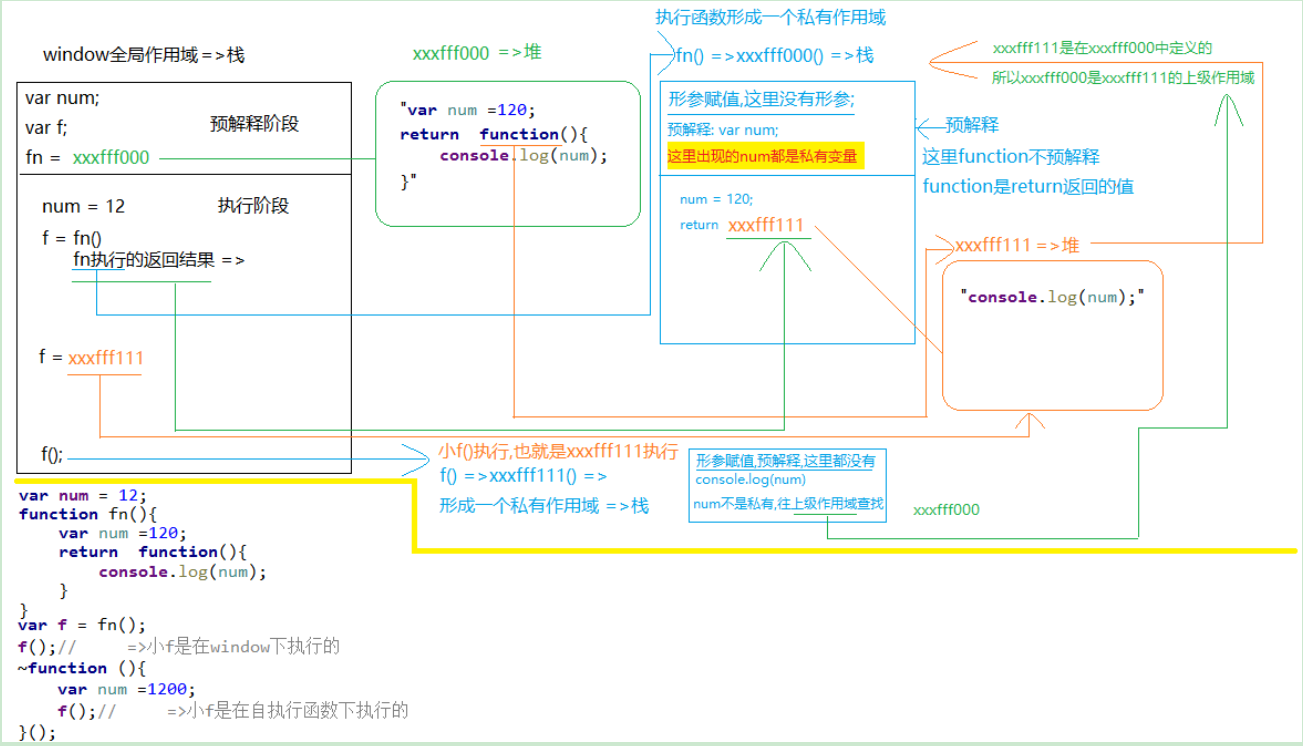
=>fn=xxxfff111

只输出两次,第三次就报错; 以上是解析;以下是案例:

```
fn();//2
function fn(){console.log(1)};
fn();//2
var fn = 10;
fn();//报错: fn is not a function ?????  报错的这行:fn已经变成了10;10不是函数所以fn()执行就报错
function fn(){console.log(2)};
fn();
```

13.作用域链

概念: 在私有作用域中,代码执行会遇到变量;首先要确认这个变量是否为私有变量,如果是私有变量,那么和外面的没有任何关系;如果不是私有的,则往当前作用域的上级作用域进行查找,如果上级作用域也没有则继续查找,一直找到window为止;=>这就叫作用域链;



14.如何查找当前作用域的上一级作用域???

概念: 看当前函数是在哪个作用域下定义的,那么它的上级作用域就是谁;=>和函数在哪儿执行[没有任何的关系];

15.

1.堆内存

引用数据类型在赋值的时候,会开一个堆内存;

16.检查本地代码写的状态github

git在本地有三个状态

1)写完代码之后没执行任何操作=>状态还没加入至工作区

2)git add -A之后=>添加至工作区的状态

3)git commit -m""之后=>创建代码版本的状态

4)可以push到服务器

git status=>查看本地所处的哪种状态的命令..

exit=>关闭命令窗口的命令(不用再用鼠标去点窗口的叉叉)