### 正则表达式：

/\*是否带有小数\*/

function isDecimal(strValue ) {

var objRegExp= /^\d+\.\d+$/;

return objRegExp.test(strValue);

}

/\*校验是否中文名称组成 \*/

function ischina(str) {

var reg=/^[\u4E00-\u9FA5]{2,4}$/; /\*定义验证表达式\*/

return reg.test(str); /\*进行验证\*/

}

/\*校验是否全由8位数字组成 \*/

function isStudentNo(str) {

var reg=/^[0-9]{8}$/; /\*定义验证表达式\*/

return reg.test(str); /\*进行验证\*/

}

/\*校验电话码格式 \*/

function isTelCode(str) {

var reg= /^((0\d{2,3}-\d{7,8})|(1[3584]\d{9}))$/;

return reg.test(str);

}

/\*校验邮件地址是否合法 \*/

function IsEmail(str) {

var reg=/^\w+@[a-zA-Z0-9]{2,10}(?:\.[a-z]{2,4}){1,3}$/;

return reg.test(str);

}

//关闭所有combotree子页面

$("#sss").combotree('tree').tree("collapseAll");

//按照自定义的顺序排序

order by decode(ss.provincecode,

'40101005110000001',

1,

'41311016110101101',

2,

'41211011840000001',

3,

'41311016140100001',

4,

'41311046430000001',

5,

'41311026210102900',

6

)

### 导出表格

exportExcelPlusFunc: function(){

console.log("2::::::::::::::::::::::::");

console.log(arrTemp);

console.log(chooseParaSearch);

chooseParaSearch.sheetname = '系统使用情况统计表';

chooseParaSearch.excelParam = 'true'

chooseParaSearch.titlefile = JSON.stringify(arrTemp)

console.log(chooseParaSearch.titlefile);

commonAction.getRestData(chooseParaSearch, function(value) {

var data =JSON.parse(value);

var filename = data.filename;

var backname = '系统使用情况统计表'

var filetype = filename.substring(filename.lastIndexOf("."));

var isscore = 'yes';

url = encodeURI("../../../pis\_lcc\_webcore/rest/webcore/downloadFile?filename=" + filename+ "&backname=" + backname + filetype+"&isscore="+isscore);

//pis\_10kvmanage\_core/rest/webcore/downloadFile

url = encodeURI(url, 'UTF-8');

document.location = url;

});

},

//动态表头的拼接(从数据库获取表头)

var getMoudleData = function() {

var requestPara = {

restUrl : "lcc/lccModelTongJi/getModuleInfo"

};

commonAction.getRestData(requestPara,function(data) {

var temp = $.parseJSON(data);//表头

function clickCell(deptcodeParam, id, fileid, value) {

if (fileid) {

return '<a href="javascript:void(0)" onclick="searchMXFunc(\''

+ deptcodeParam

+ '\',\''

+ id

+ '\')">' + value + '</a>';

} else {

return value;

}

}

;

// //////////////////////新加的//////////// （此类为为表格加的点击事件获取表格信息）

function clickMCell(deptcodeParam, id, fileid, value) {

if (fileid) {

return '<a href="javascript:void(0)" onclick="searchMMXFunc(\''

+ deptcodeParam

+ '\',\''

+ id

+ '\')">' + value + '</a>';

} else {

return value;

}

};

function clickLogCell(deptcodeParam, id, fileid, value) {

if (fileid) {

return '<a href="javascript:void(0)" onclick="searchLogMXFunc(\''

+ deptcodeParam

+ '\',\''

+ id

+ '\')">' + value + '</a>';

} else {

return value;

}

};

function clickLogNumCell(deptcodeParam, id, fileid, value) {

if (fileid) {

return '<a href="javascript:void(0)" onclick="searchLogNumMXFunc(\''

+ deptcodeParam

+ '\',\''

+ id

+ '\')">' + value + '</a>';

} else {

return value;

}

};

// ================开始拼接表头 二级表头

var arrTemp = [ [], [] ];//定义列数组

// 一部分为静态表头

arrTemp[0].push({

field : 'ROWNUM',

title : '排序',

width : 64,

sortable : true,

align : 'center',

rowspan : 2,

colspan : 1,

frozen : true,

});

arrTemp[0].push({

field : 'PROVINCENAME',

title : '单位',

width : 150,

align : 'center',

rowspan : 2,

colspan : 1,

frozen : true

});

arrTemp[0].push({

field : 'LOADPERSON',

title : '登录用户数',

width : 100,

sortable : true,

align : 'center',

rowspan : 2,

colspan : 1,

frozen : true,

formatter : function(value, row, index) {

return clickLogCell(row.PROVINCECODE, "", true,

value);

},

});

arrTemp[0].push({

field : 'TOTAL',

title : '总访问人次',

width : 100,

sortable : true,

sortable : true,

align : 'center',

rowspan : 2,

colspan : 1,

order:'asc',

formatter : function(value, row, index) {

return clickCell(row.PROVINCECODE, "", true,

value);

},

frozen : true,

});

for ( var i = 0; i < temp.length; i++) {

//一级表头

var objP = {

field : "" + (temp[i]["MODULE\_ID"] + 2),

title : temp[i]["MODULE\_NAME"],

align : 'center',

colspan : 2,

rowspan:1,

width : 200,

};

//二级表头

var objCL = {

field : "" + (temp[i]["MODULE\_ID"]),

title : "访问总人次",

colspan : 1,

rowspan:1,

sortable : true,

align : 'center',

width : 100,

formatter : function(value, row, index) {

// console.log(tempCLR)

return clickCell(row.PROVINCECODE,

this.field, true, value);

},

};

var objCR = {

field : "" + (temp[i]["MODULE\_ID"] + 1),

title : "访问总人数",

colspan : 1,

rowspan:1,

align : 'center',

sortable : true,

width : 100,

formatter : function(value, row, index) {

return clickMCell(row.PROVINCECODE,

this.field - 1, true, value);

},

};

arrTemp[0].push(objP);

arrTemp[1].push(objCL);

arrTemp[1].push(objCR);

}

//厉害

### 控制台引入jquery或者别的CDN

var script=document.createElement("script");

script.type="text/javascript";

script.src="https://code.jquery.com/jquery-3.4.1.min.js";

document.getElementsByTagName('head')[0].appendChild(script);

var popper=document.createElement("script");

popper.type="text/javascript";

popper.src="https://unpkg.com/@popperjs/core@2";

$(“head”)[0].append(popper)

### 闭包的例子：

function init() {

var name = "Mozilla"; // name 是一个被 init 创建的局部变量

function displayName() { // displayName() 是内部函数,一个闭包

alert(name); // 使用了父函数中声明的变量

}

displayName();}

init();

function makeFunc() {

var name = "Mozilla";

function displayName() {

alert(name);

}

return displayName;}

var myFunc = makeFunc();

myFunc();

function makeAdder(x) {

return function(y) {

return x + y;

};}

var add5 = makeAdder(5);var add10 = makeAdder(10);

console.log(add5(2)); // 7

console.log(add10(2)); // 12

Window 对象  
  
众多例子

1、

var age = 22;

delete window.age //false

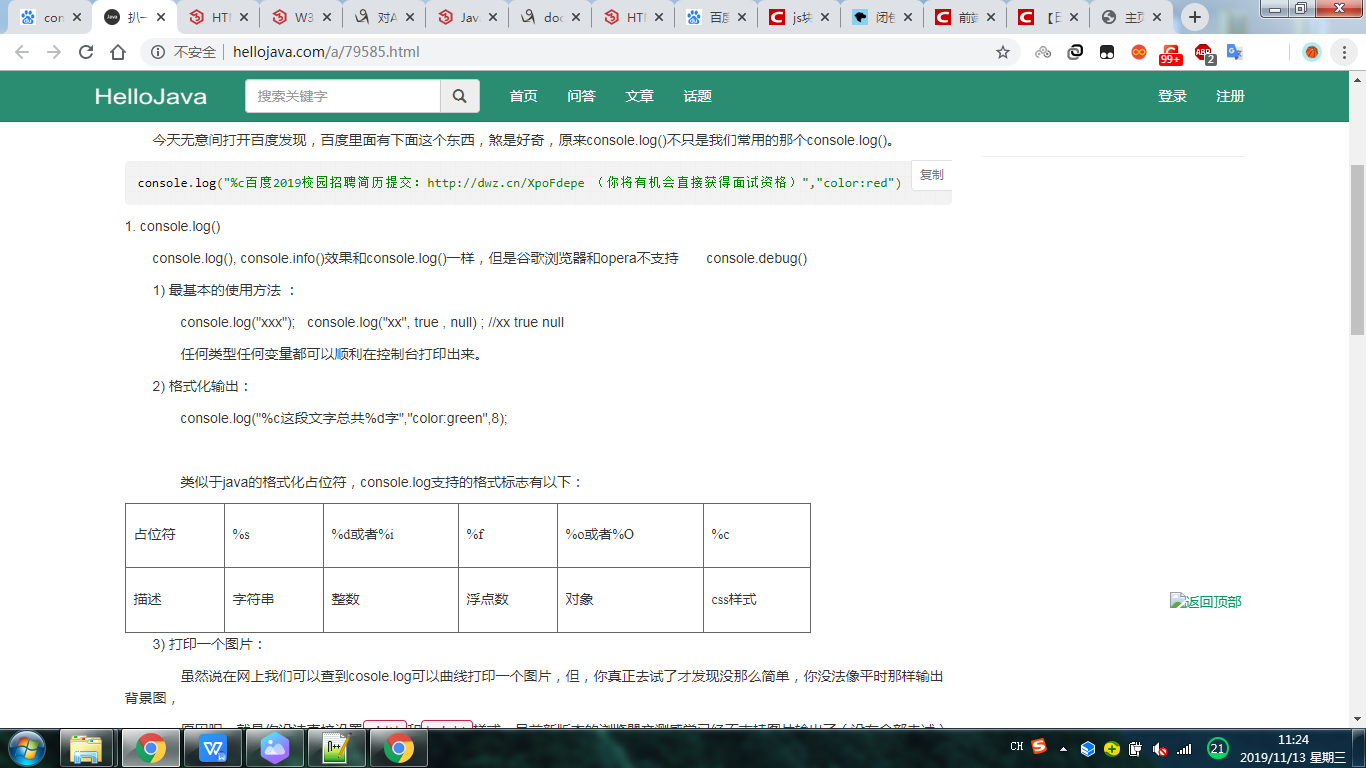
2、

var a=b

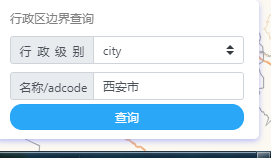
//报错：VM1568:1 Uncaught ReferenceError: **b is not defined**

var a=window.b

//不报错 返回undefined



### 如下效果的文本框



原来：span变为块级元素 增加内外边距 边框 以及背景颜色

Html：

<div class="input-card">

<label style='color:grey'>行政区边界查询</label>

<div class="input-item">

<div class="input-item-prepend">

<span class="input-item-text" >行政级别</span>

</div>

<select id="level">

<option value="district">district</option>

<option value="city">city</option>

<option value="province">province</option>

</select>

</div>

<div class="input-item">

<div class="input-item-prepend">

<span class="input-item-text" >名称/adcode</span>

</div>

<input id='district' type="text" value='朝阳区'>

</div>

<input id="draw" type="button" class="btn" value="查询" />

</div>

Css：

.input-item {

position: relative;

display: -ms-flexbox;

display: flex;

-ms-flex-wrap: wrap;

flex-wrap: wrap;

-ms-flex-align: center;

align-items: center;

width: 100%;

height: 3rem;

}

.input-item:last-child {

margin-bottom: 0;

}

.input-item>select, .input-item>input[type=text], .input-item>input[type=date] {

position: relative;

-ms-flex: 1 1 auto;

flex: 1 1 auto;

width: 1%;

margin-bottom: 0;

}

.input-item>select:not(:last-child), .input-item>input[type=text]:not(:last-child), .input-item>input[type=date]:not(:last-child) {

border-top-right-radius: 0;

border-bottom-right-radius: 0

}

.input-item>select:not(:first-child), .input-item>input[type=text]:not(:first-child), .input-item>input[type=date]:not(:first-child) {

border-top-left-radius: 0;

border-bottom-left-radius: 0

}

.input-item-prepend {

margin-right: -1px;

}

.input-item-text, input[type=text],input[type=date], select {

height: calc(2.2rem + 2px);

}

.input-item-text {

width: 6rem;

text-align: justify;

**padding: 0.4rem 0.7rem;**

**display: inline-block;**

text-justify: distribute-all-lines;

/\*ie6-8\*/

text-align-last: justify;

/\* ie9\*/

-moz-text-align-last: justify;

/\*ff\*/

-webkit-text-align-last: justify;

/\*chrome 20+\*/

-ms-flex-align: center;

align-items: center;

margin-bottom: 0;

font-size: 1rem;

font-weight: 400;

line-height: 1.5;

color: #495057;

text-align: center;

white-space: nowrap;

**background-color: #e9ecef;**

**border: 1px solid #ced4da;**

**border-radius: .25rem;**

**border-bottom-right-radius: 0;**

**border-top-right-radius: 0;**

}

.input-item-text input[type=checkbox], .input-item-text input[type=radio] {

margin-top: 0

}

.input-card {

display: flex;

flex-direction: column;

min-width: 0;

word-wrap: break-word;

background-color: #fff;

background-clip: border-box;

border-radius: .25rem;

width: 22rem;

border-width: 0;

border-radius: 0.4rem;

box-shadow: 0 2px 6px 0 rgba(114, 124, 245, .5);

position: fixed;

bottom: 1rem;

right: 1rem;

-ms-flex: 1 1 auto;

flex: 1 1 auto;

padding: 0.75rem 1.25rem;

}

.input-text {

line-height: 2rem;

margin-right: 2rem;

}

### Js运算 解决精度问题

Math.floor(Math.random()\*(m-n+1))+n n到m随机数

加法运算

function accAdd(arg1,arg2){

var r1,r2,m;

try{r1=arg1.toString().split(".")[1].length}catch(e){r1=0}

try{r2=arg2.toString().split(".")[1].length}catch(e){r2=0}

m=Math.pow(10,Math.max(r1,r2))

return +((arg1\*m+arg2\*m)/m).toFixed(2);

}

减法运算

function Subtr(arg1,arg2){

var r1,r2,m,n;

try{r1=arg1.toString().split(".")[1].length}catch(e){r1=0}

try{r2=arg2.toString().split(".")[1].length}catch(e){r2=0}

m=Math.pow(10,Math.max(r1,r2));

//last modify by deeka

//动态控制精度长度

n=(r1>=r2)?r1:r2;

return +((arg1\*m-arg2\*m)/m).toFixed(2);

}

除法运算

function accDiv(arg1,arg2){

var t1=0,t2=0,r1,r2;

try{t1=arg1.toString().split(".")[1].length}catch(e){}

try{t2=arg2.toString().split(".")[1].length}catch(e){}

with(Math){

r1=Number(arg1.toString().replace(".",""))

r2=Number(arg2.toString().replace(".",""))

return +((r1/r2)\*pow(10,t2-t1)).toFixed(2);

}

}

乘法运算

function accMul(arg1,arg2)

{

var m=0,s1=arg1.toString(),s2=arg2.toString();

try{m+=s1.split(".")[1].length}catch(e){}

try{m+=s2.split(".")[1].length}catch(e){}

return Number(s1.replace(".",""))\*Number(s2.replace(".",""))/Math.pow(10,m)

}

### 垂直上下居中的方法

第一种：知道元素的宽和高

div.box{

weight:200px;

height:400px;

position:absolute;

left:50%;

top:50%;

margin-left:-100px;

margin-top:-200px;

}

第二种：不知道元素的宽和高

div.box{

weight:200px;

height:400px;

position:absolute;

left:50%;

top:50%;

transform:translate(-50%,-50%);

}

第三种：flex布局

父级元素：{

display:flex;

flex-direction:row;

justify-content:center;

align-items:center;

}

子级元素：{

flex:1

}

==========================

### $.extend() 深拷贝与浅拷贝

==========================

let d1=[0,1,[2,3],4],

d2=$.extend(false,[],d1);

d1[0]=1;

d1[2][0]=1;

console.log(d1,d2);

// false 浅拷贝 d2的值会随着 d1 值得改变而改变

0: 1

1: 1

2: (2) [1, 3]

3: 4

=======

0: 0

1: 1

2: (2) [1, 3]

3: 4

======

let d1=[0,1,[2,3],4],

d2=$.extend(true,[],d1);

d1[0]=1;

d1[2][0]=1;

console.log(d1,d2);

//true d2值不会改变d1同上 d2下方所示 还是拷贝后的值不随着d1改变

0: 0

1: 1

2: (2) [2, 3]

3: 4

### Promise 的用法

解决回调地狱

1. 新建一个promise

let pp = new Promise(function(resolve, reject){

setTimeout(function(){

var num = Math.ceil(Math.random()\*20); //生成1-10的随机数

console.log('随机数生成的值：',num)

if(num<=10){

resolve(num);

}else{

reject('数字大于10了即将执行失败回调');

}

}, 2000);

})

pp.then( //这是将成功和失败两个返回都放在then函数中 then两个参数

data=>{

console.log('resolved成功回调');

console.log('成功回调接受的值：',data);

},

data=>{

console.log('rejected失败回调');

console.log('失败执行回调抛出失败原因：',data);

}

)

第二种

pp.then( //这是一个then函数 一个catch函数捕获失败 可以避免then成功写错代码不执行

data=>{

console.log('resolved成功回调');

console.log('成功回调接受的值：',data);

}

)

.catch(

reason=>{

console.log('catch到失败回调')

console.log('catch失败执行回调抛出失败原因：',reason);

})

1. 函数执行后如果生成的值小于10 其状态会变为resolve 输出data会返回成功值
2. 如果大于10 状态变为rejected

以上是对promise的resolve用法进行了解释，相当于**resolve**是对promise成功时候的回调，它把promise的状态修改为fullfiled，那么，**reject**就是失败的时候的回调，他把promise的状态修改为rejected，这样我们在then中就能捕捉到，然后执行“失败”情况的回调。

All函数 .all()

Promise.all来执行，all接收一个数组参数，这组参数为需要执行异步操作的所有方法，里面的值最终都算返回Promise对象。这样，三个异步操作的并行执行的，等到它们都执行完后才会进到then里面。那么，三个异步操作返回的数据哪里去了呢？都在then里面，all会把所有异步操作的结果放进一个数组中传给then，然后再执行then方法的成功回调将结果接收，结果如下：（分别执行得到结果，all统一执行完三个函数并将值存在一个数组里面返回给then进行回调输出）

.race()

all是等所有的异步操作都执行完了再执行then方法，那么race方法就是相反的，谁先执行完成就先执行回调。先执行完的不管是进行了race的成功回调还是失败回调，其余的将不会再进入race的任何回调

### 选取两个数之间的10个数 存入数组并排序

let iArray=[];

function getRandom(istart,iend){

let iChoice=iend-istart+1,

res=Math.floor(Math.random()\*iChoice+istart);

return res;

}

for(let i=0;i<10;i++){

iArray.push(getRandom(10,100));

}

iArray.sort((a,b)=>a>b);

console.log(iArray);

### 统计字符串中出现次数最多的字符和次数

let str = "asddfssssaasswef";

let obj = {};

for (let i = 0; i < str.length; i++) { //统计次数存入obj

if (!obj[str.charAt(i)]) {

obj[str.charAt(i)] = 1;

} else {

obj[str.charAt(i)]++;

}

}

console.log(obj);

let max = 0;

let charmax;

for (let key in obj) { //找到最大值的key

if (obj[key] > max) {

max = obj[key];

charmax = key;

}

}

console.log(`出现最多的字符是${charmax},出现了${max}次`);

### Popper 的使用

Npm 引入或者cdn

new Popper(reference,popper,options);

其中：

Popper构造函数接收三个参数：

reference是要定位到的HTMLElement节点，或者是一个提供定位参数的Object（不支持IE10+），所以还是使用dom节点吧 父

popper是要被定位的HTMLElement节点 子

reference和popper元素都是构造函数必须的，本文后面如不做解释则默认就是指这两个对象

默认的定位方式可能就是把popper元素定位到reference元素下方附近。

Popper.js提供其他定位方式和方法，使用options对象参数来配置

参考：<https://blog.csdn.net/jhzhahuaiyu/article/details/90213582>

var box = document.querySelector('.box'),popper = document.querySelector('.popper'),

Popper.createPopper(aaa, bbb, {

placement: 'right',

});

Var SignName = Popper.createPopper(aaa, bbb, {

placement: 'right',

});

### Js原生 触发点击事件

// IE

if(document.all) {

document.getElementById("su").click();

}

// 其它浏览器

else {

var e = document.createEvent("MouseEvents");

e.initEvent("click", true, true);

document.getElementById("su").dispatchEvent(e);

}

C:\Users\Administrator\AppData\Local\Google\Chrome\User Data\Default\Extensions

谷歌插件安装地址目录

(function(){alert(2)})() 立即执行函数

（function（）{}）（）

LHH-XS5VJ-6Y