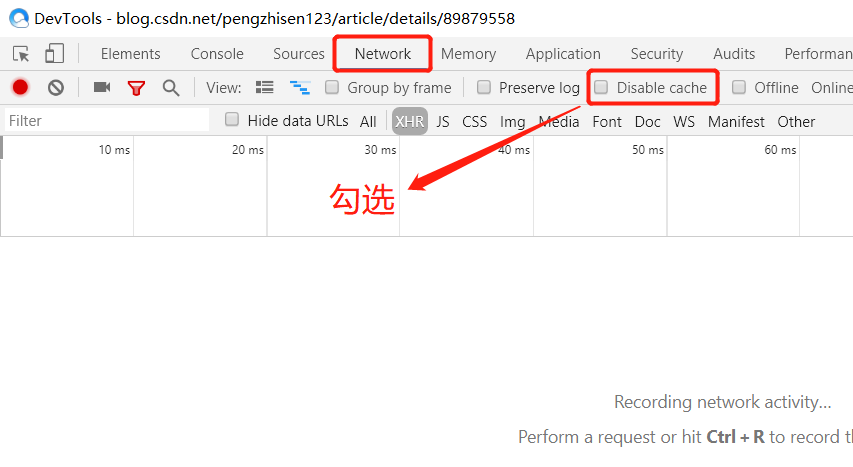
# 页面禁用缓存

拿QQ浏览器举例。

点击F12，勾选Disable cache然后刷新页面即可。



# 字符串操作

## 1.去除字符串最后一个字符

var comCode = ‘abcsd/’;

comCode = comCode.substring(0,comCode.length-1);

## 2.去除字符串第一个字符

var comCode = ‘abcsd/’;

comCode = comCode.substr(1);//返回从索引1到结尾的字符串

## 3. 字符串的反转(借用数组)

const str = "www.mzh.ren";  
str.split('').reverse().join('');//返回反转后的字符串

## 4. 判断一个字符串是否是数字

!isNaN(parseInt(len))

或者

Number.isNaN(parseInt(len))

二者的区别是：isNaN(par)会尝试通过Number(par)将par转换为数字，若失败返回NaN。

而Number.isNaN(par)会直接比较par是否===NaN，不会转换par，所以使用Number.isNaN时，得先使用parseInt(par)尝试转换参数为数字，parseInt(par)失败时返回NaN。

## 5.数值型的字符串和数字比较大小

数值型的字符串会转化为数字再和另一个数字比较大小

# select标签的操作

## 1.获取当前选中的option的下标(selzyh为select的id)

$("#selzyh").prop('selectedIndex');

## 2.根据option的下标来选中option(selzyh为select的id)

$("#selzyh").get(0).selectedIndex = 0;

## 3.获取option的最大下标值(selzyh为select的name)

$("select[name='selzyh']").find("option:last").index();

## 4.获取当前选中option的value(selzyh为select的id)

$("#selzyh option:selected").val();

## 5.获取当前选中的option中的文本

$("#selzyh option:selected").text();

## 6.设置当前option被选中

$(this).prop("selected",true); //$(this)即为当前option对象

## 7.回显layui样式的select

正常的让option被选中之后，还得重新渲染form才能回显。

layui.form.render();

且满足格式如下：

<form id="addform" class="layui-form">

......

<select lay-verify="required" lay-search="">

<option></option>

...

</select>

</form>

## 8.select的回显

var all\_select=$("#edit\_technicalGrade > option");

1. for(var i=0;i<all\_select.length;i++){
2. var svalue=all\_select[i].value;
3. if(pdata.technicalGrade==svalue){  //取select中所有的option的值与其进行对比，相等则令这个option添加上selected属性
4. $("#edit\_technicalGrade option[value='"+svalue+"']").attr("selected","selected");
5. break;
6. }
7. }

# 父页面调用iframe子页面的方法

1.父页面得有iframe子页面，如：

<div class="span10" style="height:100%;">  
 <iframe id="childRightFrame" style="height:650px;width: 103%;margin-left:-40px;" src="${path}/work/toWorkRight?userSysId=${requestScope["userSysId"]}" scrolling="no" frameborder="0" name="childRightFrame"></iframe>  
</div>

这样父页面加载时，会去请求controller中的方法toWorkRight()并传递参数，再由该方法使用视图解析器返回一个jsp页面。

2.在父页面中使用js来调用子jsp页面中的js方法。如：

$("#childRightFrame")[0].contentWindow. showPatData();

其中，childRightFrame是父页面中<iframe>的id，$("#childRightFrame")[0].contentWindow是

拿到子页面的页面对象，showPatData()则是子页面中的js方法。

3.要注意的问题是页面的加载顺序。

4.可以通过attr()给子页面传递数据，如：

$("#subChildRightFrame").attr("src",rootUrl + "/work/toWorkRightFrom?patInHosId=" +currentPatInHosId);

5.父页面拿到子页面的数据

var patInHosId = $("#indexPage").contents().find("#selzyh option:selected").val();

其中，indexPage是iframe标签的id。本语句拿的是子页面中id为selzyh的select标签的当

前选中的option的值。需要注意的是find方法根据id取元素时不再需要$。

# 数组的操作

## 1.数组的创建

var array = new Array();

注意：如果要创建多维数组，必须一级一级的创建，不能一次性创建。

继续创建二维数组：

array[0] = new Array();

## 2.数组的清空

var patInHosIds = (“1”,”2”,”3”);

patInHosIds.splice(0,patInHosIds.length);

## 3.遍历一个数组中是否有相同的元素

var code = {“ab”,”cd”,”ab”,”gt”};

for(var i = 0;i < code.length;i++){  
 for(var j = i+1;j < code.length;j++){  
 if(code[j] === code[i]){//连类型也判断  
 对比之后的操作

}  
 }  
}

# 操作单选按钮

## 1.设置单选按钮可以取消和选中

$(function(){

    var old = null; //用来保存原来的对象

    $(".radioCheck").each(function(){//循环绑定事件

        if(this.checked){

            old = this; //如果当前对象选中，保存该对象

        }

        this.onclick = function(){

            if(this == old){//如果点击的对象原来是选中的，取消选中

                this.checked = false;

                old = null;

            } else{

                old = this;

            }

        }

    });

});

其中radioCheck是该单选框的class值

## 2.使用单选按钮控制多选框的全选与全取消

var flag = true;  
$(document).on("click","#radioCheck",function () {//设置全选按钮的状态  
 for(var i=0;i<$("[name='patient']").length;i++){  
 $("[name='patient']")[i].checked = flag;  
 }  
 flag =! flag;  
});

其中：1. radioCheck是单选框的id值。2. patient是多选框的name值。

# js插件

## 1.My97时间插件的使用

1.需要引入该js文件

WdatePicker.js

2.页面书写该js时间标签

<span>开始时间</span>  
<input id="select-start-time" class="Wdate" />  
<span>结束时间</span>  
<input id="select-end-time" class="Wdate" />

3.设置该标签的点击事件，如：结束时间不能早于开始时间

$(document). on("click","#select-start-time",function () {//开始时间  
 WdatePicker({  
 lang: 'auto',  
 dateFmt: 'yyyy-MM-dd',  
 maxDate: '#F{$dp.$D(\'select-end-time\')||\'%y-%M-%d\'}',  
 readOnly: true  
 })  
}).on("click","#select-end-time",function () {//结束时间  
 WdatePicker({  
 lang: 'auto',  
 dateFmt: 'yyyy-MM-dd',  
 minDate: '#F{$dp.$D(\'select-start-time\')}',  
 maxDate: '%y-%M-%d',  
 readOnly: true  
 })  
});

4.直接调用My97

## 2.插件select2.js

1.作用

给select加一个input框，根据option的内容筛选option

2.引入

需要引入select2.css、jquery.js、select2.js

3.使用

(1)在前端页面生成一个select。如：

<select id="sel" style="width: 150px;">

    <option value="1">王者荣耀</option>

    <option value="2">部落冲突</option>

    <option value="3">枪战王者</option>

    <option value="4">永恒纪元</option>

    <option value="5">英魂之刃</option>

    <option value="6">皇室战争</option>

</select>

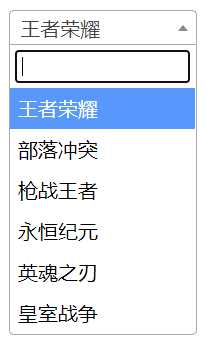
(2)在js里调用select2()方法

$(function(){

    $("#sel").select2();//不加参数的最基本用法

});

效果图如下：



此时，可以输入汉字来筛选option。

4.但有时要求用拼音也能筛选option，所以在前端操作时需要引入pinyin.js

注意：运用前端的pinyin.js插件消耗cpu性能多一些，最好是将汉字的拼音形式当作实体类的一个属性存进数据库，用的时候直接查询拼接成select即可。

(1)pinyin.js可以将汉字转化为拼音

var chars = pinyin.getFullChars(“王者荣耀”).replace(/[^a-zA-Z0-9]/g,'');

chars值为WangZheRongYao

var char = chars.replace(/[a-z]/g,'');

char值为WZRY

(2)运用以上方法可以将select拼成如下的形式

<select id="sel2" style="width: 150px;">

    <option value="1">王者荣耀|wangzherongyao|wzry</option>

    <option value="2">部落冲突|buluochongtu|blct</option>

    <option value="3">枪战王者|qiangzhanjiyuan|qzjy</option>

    <option value="4">永恒纪元|yonghengjiyuan|yhjy</option>

    <option value="5">英魂之刃|yinghunzhiren|yhzr</option>

    <option value="6">皇室战争|huangshizhanzheng|hszz</option>

</select>

(3)调用带参的select2()

$("#sel2").select2({//加参数的用法

    templateResult :function(item){

        var str = item.text;

        var idx = str.indexOf('|');

        return str.substring(0,idx);

    },

    templateSelection:function(item){

        var str = item.text;

        var idx = str.indexOf('|');

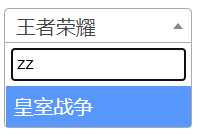
        return str.substring(0,idx);

    },

    placeholder:"输入字母或汉字"

});

(4)效果图如下



## 3.右键菜单插件contextMenu.js

# 复选框(checked)的操作

## 1.根据name属性拿到所有选中的复选框的对象

$('input[name="patient"]:checked');

## 2.判断复选框是否被选中

if ($("#isGather").is(":checked")){}

## 3.使id为isAppend的复选框被选中

$("#isAppend").prop('checked', true);

## 4.获取所有name为formId的checkbox

$(":checkbox[name='formId']")

## 5.单个checkbox设置多个checkbox的全选与取消全选

var $selectAll = $("#selectAll");  
$selectAll.click(function() {  
 if ($selectAll.prop("checked") == true) {  
 $(":checkbox[name='formId']").prop("checked", true);  
 } else {  
 $(":checkbox[name='formId']").prop("checked", false);  
 }  
});

# input框操作

## 1.监测input框中值的改变，可以使用jquery的keyup事件。如：

$(document).on("keyup","#search",function () {

alert("input框中的值每次改变都会触发该事件！");

});

其中，search是该input框的id值。

## 2.可以使用js原生Date对象来赋值input框的值，如：

//页面初始化时开始时间和结束时间默认为当前时间  
function defaultTime() {  
 //设置结束日期为当前日期  
 var date = new Date();  
 var seperator1 = "-";  
 // var seperator2 = ":";  
 var month = date.getMonth() + 1;  
 var strDate = date.getDate();  
 if(month >= 1 && month <= 9) {  
 month = "0" + month;  
 }  
 if(strDate >= 0 && strDate <= 9) {  
 strDate = "0" + strDate;  
 }  
 // var time = date.getFullYear() + seperator1 + month + seperator1 + strDate +" " + date.getHours() + seperator2 + date.getMinutes() +  
 //seperator2 + date.getSeconds();  
 var time = date.getFullYear() + seperator1 + month + seperator1 + strDate;  
 $("#select-start-time").val(time);  
 $("#select-end-time").val(time);  
}

我把拼接时分秒的给注释了，只到天。

补充：select-start-time和select-end-time是input框的id，该方法可以和My97时间插件配合使用。

## 3.设置input只能输入数字字符

<input type="text" onkeyup="this.value=this.value.replace(/\D/g,'')">

## 4.监测input框中的Enter事件

$(“#table”).on(“keypress”,”tr td input”,function(event){

if(event.keyCode == “13”){

event.preventDefault();//阻止默认事件的发生

alert(“触发了Enter事件”);

}

});

## 5.检查是否有某个属性

例如：检查标签中是否有user\_id属性。

var type = $(“input”).attr(“user\_id);

if(typeOf(type) != undefined){

//该标签存在此user\_id属性

}

## 6. maxlength属性

其值为input框键入的最大字符数

## 7.判断聚焦

$(input).is(“:focus”)：值为true或者false

## 8.添加金钱的逗号分隔符

thisThousandSeperator.value=missing.replace(/\B(?=(\d{3})+(?!\d))/g, ',');

例如：

<input type="text" onblur="this.value = this.value.replace(/\B(?=(\d{3})+(?!\d))/g, ',')" onfocus="this.value=this.value.replace(/,/g,'')"/>

# 阻止浏览器默认右击事件

## 1. 在div中阻止浏览器默认右键点击事件

$("div").bind("contextmenu", function(){

return false;

});

## 2.整个页面屏蔽浏览器默认右击事件

document.oncontextmenu = function() {

return false;

}

# 取消默认事件

event.preventDefault();

# 在页面某区域右击出现菜单

配合步骤：1.在html页面写上一个菜单，如：

<div id="rightClick" class="rightClick">  
 <ul style="list-style: none;">  
 <li>菜单项一</li>  
 <li>菜单项二</li>  
 <li>菜单项三</li>  
 <li>菜单项四</li>  
 </ul>  
</div>

2.让菜单在页面初始化时隐藏并在该区域禁用掉浏览器默认的右击菜单，可以在js里写。如：

$(function(){

//页面加载时隐藏右击的菜单

$("#rightClick").hide();

//在table上阻止浏览器默认右击事件  
$(".layui-tab-content").bind("contextmenu",function () {  
 return false;  
});

});

3.设置右击出现自己的菜单，如：

$(document).on ("mousedown",".layui-table-body",function (e) { //为table的tr绑定右击事件  
 if(3 == e.which){ //e.which是1时为左击，是2时为中击(滚轮)，是3时为右击  
 //获取右键点击的坐标  
 var x = e.clientX;  
 var y = e.clientY;  
 //设置菜单出现的位置  
 $("#rightClick").css({

position: "absolute",  
 left:x,  
 top:y  
 });  
 //需要在右击出现的菜单上也禁止浏览器默认的右击事件  
 $("#rightClick").bind("contextmenu",function () {  
 return false;  
 });  
 //出现菜单  
 $("#rightClick").show();  
 }  
});

4. 当单击页面任意位置时需要隐藏table中右击出现的菜单。

$(document).click(function () {

//当单击页面任意位置时需要隐藏table中右击出现的菜单  
 $("#rightClick").hide();  
});

# table操作

## 1.点击tr时，获取该tr对象

$(document).on(“click”,”table tr”,function(){

$(this);// $(this)即为该tr对象，你可以根据它拿到它的所有的td的值

});

补充：可以随意修改事件类型，双击事件、右击事件等都行。

## 2.点击td时，获取td、tr的下标

$('table tr td').click(  
  function() {  
    alert( $(this).parent().parent().find("tr").index($(this).parent()[0]));//第几行  
   alert($(this).index());//第几列  
  });

注意：如果是点击td中的button而触发的点击事件，需要多写parent()保证得到正确的对象。

## 3.使td中的内容可编辑

给<td>加上contentEditable="true" 属性即可

<td contentEditable="true"> </td>

## 4.得到tr中td的数量

## 5.点击tr，触发tr中checkbox的点击事件

方式1

(1)<tr onclick=”autoCheck(this.sectionRowIndex);”> // this.sectionRowIndex是该tr的下标

<td><input type="checkbox" name="formChecked" onclick="sumFormCheck();"></td>

</tr>

(2)

function autoCheck(index) {//打印预览弹框单击tr，触发tr中的复选框单击事件  
 $("#formCheckTable tbody tr").eq(index).find("input[name='formChecked']")

.trigger("click");  
}

function sumFormCheck() {

event.stopPropagation();//阻止触发父元素的点击事件

}

方式2

(1)<tr onclick=”autoCheck(this);”>

<td><input type="checkbox" name="formChecked" onclick="sumFormCheck();"></td>

</tr>

(2)

function autoCheck(self) {//打印预览弹框单击tr，触发tr中的复选框单击事件  
 $(self).find("input[name='formChecked']").click();  
}

function sumFormCheck() {

event.stopPropagation();//阻止触发父元素的点击事件

}

# json与string类型数据的相互转换

需要引入json2.js

**浏览器支持的转换方式**(Firefox，chrome，opera，safari，ie9，ie8)等浏览器：

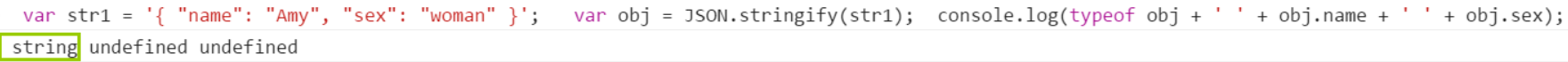
## 1.string转json

使用JSON.parse(jsonstr); //可以将json形式的字符串转换成json对象



## 2.json转string

使用JSON.stringify(jsonstr);//可以将json对象转换成json形式的字符串



# **弹框事件**

## 1.浏览器自带的离开弹框

window.onbeforeunload = function (event) {

return "数据有变动，请点击保存后再离开！";  
};

## 2.浏览器自带的确认弹框

//用户点击确认时，返回的值(变量value)就是true；点击取消，变量value就是false

var value = confirm(“数据有变动，需要保存吗？”);

# js语法

## 1.对象是key:value的形式

var a = {

    b : {

        c : function(){

            alert("haha");

        }

    },

    d : {

        e : function(){

        }

    }

}

对象里是key：value的格式，每对key：value之间用逗号隔开。如果要调用a对象中的方法，

如函数c。可以这么写：a.b.c();

## 2.||的使用

$("#field\_group>option").each(function (){

// 如果||左边是true，其右边不执行；如果||左边是false，右边会执行  
 $(this).val() == data.itemGroupCode || $(this).attr("selected",true);  
});

## 3.&&的使用(与||作用相反)

$("#field\_group>option").each(function (){

// 如果||左边是true，其右边执行；如果||左边是false，右边不会执行  
 $(this).val() == data.itemGroupCode && $(this).attr("selected",true);  
});

## 4.便历json数组的方式

for(var index in arr){  
 //arr是json数组，index是索引

}

# 元素隐藏/显示

## 1.判断是否隐藏

if(node.is(':hidden')){}

## 2.判断是否显示

if(node.is(':visible')){}

## 3.设置隐藏/显示

$(“#id”).hide();//隐藏

$(“#id”).show();//显示

# 正则表达式

1.test与RegExp

let hd = “houd”;

let a = “u”;// let a = “\\w”, \w表示字母、数字、下划线。

let reg = new RegExp(a,”g”);

console.log(reg.test(hd));

test方法用于检测某字符串是否匹配某种模式，若匹配返回true，否则返回false。

new RegExp(a,”g”)的第一个参数可以是字符串、变量、表达式，第二个参数是匹配的范围，”g”是指全部匹配。

2.

# layer组件

引入layui.all.js且使用layer的方法时，可以先执行以下语句加载模块

layui.use('layer', function(){  
 var layer = layui.layer;  
});

## 1.官网

地址：<https://www.layui.com/>

## 2.静态table

## 3.图标

文档：https://www.layui.com/doc/element/icon.html

## 4.按钮

文档：<https://www.layui.com/doc/element/button.html>

## 5.弹框

文档：https://layer.layui.com/

## 6.加载遮罩

var loading = layer.msg('正在保存，请稍等。。。', {icon: 16, shade: 0.3, time: 0});

ajax代码。。。

parent.parent.layer.close(loading);

# window.open()

window.open()是js打开新窗口的一种方式

基本语法：

window.open(pageURL,name,parameters);

例子：

window.open('page.html', 'newwindow', 'height=100, width=400, top=0, left=0, toolbar=no, menubar=no, scrollbars=no, resizable=no, location=no, status=no');

# html

## 1.标签公有属性title

<span title=”鼠标停留显示！”></span>

当鼠标停留在该<span>标签上时，便会显示属性title的值。

## 2.overflow属性

一般用在div中，当子元素的高度超出父元素的高度时：

overflow:hidden;表示隐藏掉超出的部分

overflow:auto;表示出现滚动条，通过拖动滚动条来显示子元素的内容

## 3.z-index

当使用position:absolute的元素被其他元素遮挡的时候，可以使用z-index提高该元素的显示级别。如：z-index:99999; //99999是最高的显示级别。

## 4.设置div可编辑/不可编辑

<div contenteditable = “true”></div>

<div contenteditable = “false”></div>

# each循环

## 1.跳出/跳过each循环

return false;/return true;

## 2.判断each是否循环到了最后一个元素

例：

//1.先拿到需要遍历的元素的个数

var len = $ (":button").length;

$ (":button").each (function (i, dom){

//2.判断是否是最后一个元素

if (i === len - 1){

//3.是最后一个元素时的操作

}

});

# jquery选择器

注意：jquery对象返回的永远是一个数组对象，若未找到指定元素，就会返回一个空的数组对象。比如判断jquery对象是否存在：

if($(“#id”).length > 0){//存在jquery对象}

(1) $(‘\*[name]’) //选择带有name属性的元素

(2) $(‘tr[name]’) //选择带有name属性的tr元素

(3) $("#" + id + "").val(value); //动态id选择，id是一个变量

(4) $("div[class]") 选择所有含有class属性的div元素   
(5) $("input[name='keyl']") 选择所有的name属性等于'keyl'的input元素   
(6) $("input[name!='keyl']") 选择所有的name属性不等于'keyl'的input元素   
(7) $("input[name^='keyl']") 选择所有的name属性以'keyl'开头的input元素   
(8) $("input[name$='keyl']") 选择所有的name属性以'keyl'结尾的input元素   
(9) $("input[name\*='keyl']") 选择所有的name属性包含'keyl'的input元素   
(10) $("input[id][name$='keyl']") 可以使用多个属性进行联合选择，该选择器是得到所有的含有id属性并且那么属性以keyl结尾的元素

(11) $(“ul li:first”)：获取ul下第一个li元素(jQuery对象)，其中ul和li可以换成任意元素，只要元素之间有从属关系。比如：$(“a b:first”)：获取a下第一个b元素，元素也可以用id或者元素自身的属性代替。

(12) $(“:focus”) 获取页面中得到焦点的元素，需要做长度的判断。如：

if($(“:focus”).length > 0){}

# 时间操作

## 1.两个时间的大小判断

var date1 = new Date(“2020-01-18”);

var date2 = new Date(“2020-01-19”);

if(date1.getTime() < date2.getTime()){//IE11及以下不支持带‘-’的时间格式，得转化

//IE下可使用.replace(/-/g, '/')来转化日期字符串，将’-’改为‘/’

//操作

}

## 2.校验日期格式

(1)HH:mm格式

var reg =/^(0?[0-9]|1[0-9]|2[0-3])[:](0?[0-9]|[1-5][0-9])$/;

var time = “12:00”;

if(reg.test(time)){

//格式是HH:mm后的操作

}

(2)

## 3.字符串格式的时间转换为Date对象

var currentRecordTime = new Date(currentRecordTime.replace(/-/g, '/'));

补充：currentRecordTime是字符串格式的时间，IE11下需要用到replace方法来转换。

## 4.计算两个时间之间的间隔

let startTime = new Date('2018-1-1'); // 开始时间

let endTime = new Date(); // 结束时间

console.log(endTime - startTime); // 毫秒数

console.log(Math.floor((endTime - startTime) / 1000)); // 秒数

console.log(Math.floor((endTime - startTime) / 1000 / 60)); // 分钟

console.log(Math.floor((endTime - startTime) / 1000 / 60 / 60)); // 小时

console.log(Math.floor((endTime - startTime) / 1000 / 60 / 60 / 24)); // 天数

# 阻止 click 事件冒泡到父元素

event.stopPropagation();//阻止触发父元素的点击事件

# js中获取session域的值

var name= "${sessioScope.变量名}";

var name= "${变量名}";

# 获取元素的背景色

$(element).css("backgroundColor");

element是dom对象，如果是jquery对象，直接调用css方法即可。

# 按百分比缩小浏览器页面

document.body.style.zoom = window.screen.width / 1920;

# 清除页面的缓存

1.在引入js时，可以通过添加随机数的方式来解决每次都需要清缓存的问题。如：

<script>  
 //引入私有的js  
 document.write("<script type='text/javascript' src='/nursing/static/js/record/continueRecord.js?v="+Math.random()+"'></script>");  
</script>

2.html页面添加标签属性，也可以自动清除缓存，如添加以下标签：

<meta http-equiv="Pragma" content="no-cache">

<meta http-equiv="Cache-Control" content="no-cache">

<meta http-equiv="Expires" content="0">

# getUrlParam方法

该方法可以通过传递参数名称的方式拿到url中参数的值。

例：  
url: <http://localhost:8880/nursing/workIndex.html?userSysId=2718&wardId=191&patInHosId=28884>

function getUrlParam(name){  
 var reg = new RegExp("(^|&)" + name + "=([^&]\*)(&|$)"); // 构造一个含有目标参数的正则表达式对象  
 var r = window.location.search.substr(1).match(reg); // 匹配目标参数  
 if (r != null)  
 return unescape(r[2]);  
 return null; // 返回参数值  
}

var value = getUrlParam(wardId);//获取url中wardId参数的值

# new Date().getTime()在IE下的兼容问题

const date = '2018-01-30 09:07:15';

const timestamp = new Date(date).getTime(); //这里在ie下返回NaN,具体场景使用的是IE11.0版本。

//将-替换为/，就可以在IE下正确转换为时间对象。

const date = '2018-01-30 09:07:15'.replace(/-/g, '/');

const timestamp = new Date(date).getTime();

# 将base64位的字符串转化为图片

前提是这个字符串是由正确的图片转化来的！！！

例：

<img alt="star" src="data:image/gif;base64,R0lGODlhEAAQAMQAAORHHOVSKudfOulrSOp3WOyDZu6QdvCchPGolfO0o/XBs/fNwfjZ0frl3/zy7////wAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAACH5BAkAABAALAAAAAAQABAAAAVVICSOZGlCQAosJ6mu7fiyZeKqNKToQGDsM8hBADgUXoGAiqhSvp5QAnQKGIgUhwFUYLCVDFCrKUE1lBavAViFIDlTImbKC5Gm2hB0SlBCBMQiB0UjIQA7" />

# 获取页面的滚动高度

var height = $(document).scrollTop();

# 输出jquery的版本号

alert(jQuery.fn.jquery);

# js的几种for循环

1.

2.

3.

4.

# !!的使用

1. !可将变量转换成boolean类型，null、undefined和空字符串取反都为true，其余都为false。
2. !!可作为类型判断

var a;

而if(a!=null&&typeof(a)!=undefined&&a!=’’){} 相当于 if(!!a){}

# pop()和shift()

1. pop()：移除并返回列表中最后一个元素
2. shift()：移除并返回列表中第一个元素

# js的replace()

replace(参数1，参数2)：

1.参数1可以是字符串、规则表达式。

2.该方法默认只替换匹配的第一个元素。

3.如果要匹配所有，需要加上\g。例：str = str.replace(/</g,"&lt;");

# js判断数据是否为字符串

function isString(str){  
 return(typeof str=='string')&&str.constructor==String;  
}

# js的定时器

## 1.window.setTimeout()

使一段代码在指定时间后运行。

使用方式：window.setTimeout(code,millisec);

其中，code可以是用引号括起来的一段代码，也可以是一个函数名。而millisec是毫秒数。

var time = setTimeout(function(){

console.log(“500毫秒后运行”);

},500);

//清除定时器

window.clearTimeout(time);

//另一种使用方式

function t(){console.log(“哈哈”);}

setTimeout(t,500);

## 2.window.setInterval()

使一段代码每过指定时间就运行一次。

使用方式：window.setInterval(code,millisec);

其中，code可以是用引号括起来的一段代码，也可以是一个函数名。而millisec是毫秒数。

var time = setInterval(function(){

console.log(“每过500毫秒后运行一次”);

},500);

//清除定时器

window. clearInterval(time);

# 转义符号

&apos; ---- ‘

&quot; ----- "  
&lt; ---- <  
&gt; ----- >  
&nbsp; ----- 不断开空格

# js遍历对象的几种方法

const obj = {id:1,name:’ zhangsan’,age:18 }

1.for...in

for(let key in obj){

console.log(key + ‘------’ + obj[key]);

}

2.Object对象的方法

Object.keys(obj)//返回key组成的数组

Object.values(obj)//返回value组成的数组

3.使用Object.getOwnPropertyNames(obj)方法

Object.getOwnPropertyNames(obj).forEach(function (key)){

console.log(key + ‘------’ + obj[key]);

}

# null和字符串相加

console.log(null+"");

console.log(typeof (null+""));

null和字符串相加结果是字符串。