# 1.引入

## ⑴类引入

<jsp:directive.page import="zero.space.ch03.BookBean"/> 把所有类包的引入放在一个JSP中，在其他jsp中通include引入那个jsp。这时引入类就必须采用只能在同一个jsp页面中使用，不能跨页调用

<%@ page import="zero.space.ch03.BookBean,com.entity.bean.Department" %> 这样的写法。

## ⑵页面引入

<%@ include file="..."%>速度会快一些, 这个是请求一个页面。引入静态文本(html,jsp),在JSP页面被转化成servlet之前和它融和到一起。静态include用include伪码实现, 不会检查所含文件的变化, 适用于包含静态页面。比如导航栏等元素都相同的时候，试用

<jsp:include page="..." />相对来讲会慢, 这个相当于请求两个页面。引入执行页面或servlet所生成的应答文本。动态include 它总是会检查所含文件中的变化, 适合用于包含动态页面, 并且可以带参数

# 2.JS断点

JS断点要加在点击事件的触发处，但是要加在方法内的函数，而不是加在方法选择器

debugger;

# 3.前端把js的值传到后台

var disagreeChoose= $('#disagreeReason').val();

//alert(disagreeChoose);

if("1"==disagreeChoose){

disagreeReason="立题依据不足";

}else if("2"==disagreeChoose){

disagreeReason="技术路线不可行";

}else if("3"==disagreeChoose){

disagreeReason="低水平重复研究";

}else if("4"==disagreeChoose){

disagreeReason=$("input[name='disagreeText']").val();

}

$("input[name='bean.disagreeReason']").val(disagreeReason);

<%利用隐藏域%>

<input type="hidden" name='bean.disagreeReason' readonly="true" value="<%=bean.getDisagreeReason() %>"/>

后台

String disagreeReason = request.getParameter(“bean.disagreeReason”);

# 4.JS做窗口效果

<a href="javascript:windowClose();" >关 闭</a>

<a href="JavaScript:history.go(-1)">返 回</a>

window.location.reload()//刷新;可以在script中加setTimeout(‘function()’,1000) 每秒刷

window.open("<%=basePath %>readme.zip","微信系统代理配置说明");//下载

//初始化js

$(document).ready(function(){ trace("初始化方法进入"); });

$(function(){ trace("初始化方法进入二"); });

window.onload=function(){ //要初始化的东西 }

jQuery属性attr() 方法设置或返回被选元素的属性值

$('#areaSelect').attr("disabled",true); //设置disabled属性

$('#areaSelect').removeAttr("disabled"); //移除disabled属性

# 5.选择元素（单选，多选）

## ⑴多选 checkbox

var entrustFlag = $("input[name='entrustFlag']:checked").val();//这样只获取name为entrustFlag的选中的input的值

js根据后台传值，将多选框勾选

//权限

**var** rolesId = "<%=result.get(0).getRolesid()%>";

**if**(rolesId.length>0){

**var** roles = document.forms[0].rolesid;

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

**if**(rolesId.indexOf(roles[i].value, 0)>-1){

roles[i].setAttribute("checked","checked");

}

}

}

页面元素

<input type="checkbox" name="rolesid" value="<%=i+1%>"

title="选择/不选择"><%=rolesMap.get(i+1)%></input>

后台接收

String rolesid[] = request.getParameterValues("rolesid");

js判断多选框是否有选择

**var** parts = document.forms[0].partsId;

**var** notCheckFlag = **true**;

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

**if**(parts[i].checked){

notCheckFlag = **false** ;

**break** ;

}

}

**if** (notCheckFlag) {

alert("请至少选择一个角色");

**return false**;

}

页面

<input type=*"checkbox"* name=*"partsId"* value=*"*<%=part.getId()%>*"*

title=*"选择/不选择"*><%=part.getDetail()%></input>

5.1.2 根据页面所有输入框来获取多选框

<script   language="javascript">     
  var   getCK=document.getElementsByTagName('input');     
  var   j=0;         
  for(var   i=0;i<getCK.length;i++)     
  {     
      whichObj=getCK[i];     
      if(whichObj.type=="checkbox")         {     
          whichObj.checked=false;     
          j++;     
      }     
  }     
  document.getElementById('t').innerText=j;     
  </script>

## ⑵单选 select

<select class="selector"></select>

1. 设置value为pxx的项选中

$(".selector").val("pxx");

1. 设置text为pxx的项选中

$(".selector").find("option[text='pxx']").attr("selected",true);

这里有一个中括号的用法，中括号里的等号的前面是属性名称，不用加引号。

1. 获取当前选中项的value

$(".selector").val();

1. 获取当前选中项的text

$(".selector").find("option:selected").text();

## ⑶select的级联

即第二个select的值随着第一个select选中的值变化。$(".selector1").change(function(){

// 先清空第二个

$(".selector2").empty();

// 实际的应用中，这里的option一般都是用循环生成多个了

var option = $("<option>").val(1).text("pxx");

$(".selector2").append(option);

});

## ⑷.select选框宽度和内容宽度

$(function(){

//控制单位下拉框的宽度(有个单位名太长所以要控制一下)

$("select[name='bean.unitId']").attr("style","width:220px");

//如果单位下拉框中的内容超出15个字就省略显示

$("select[name='bean.unitId'] option").each(function(){

var text = $(this).text();

if(text.length>15){

$(this).attr("title",text);

$(this).text(text.substring(0,15)+"...");

}

});

});

# 6.获取元素

## ⑴通过name获取

var a = document.getElementsByName("bean.isGrant")[0];

$("[name=bean.isGrant]");//$ 是jqeury的简写

## ⑵通过id获取

var id1 = document.getElementById(“id1”);

var id2 = $(“id2”);

## ⑶通过classname获取

document.getElementsByClassName()[0];//HTML5 新增的DOM API。IE8以下不支持

$(“.xx”);

1. 交集选择： $(".a.b")--选择同时包含a和b的元素。

2. 并集选择：$(".a, .b")--选择包含a或者包含b的元素。

3. 子级元素选择:$(“.a.b > a”)—选择同时包含class a和b下的所有a标签。

## ⑷通过tagname获取

document.getElementsByTagName('a');// 返回页面上所有超链接节点

(5)追加元素

 jQuery 添加新内容有以下四个方法：

* append() - 在被选元素的结尾插入内容
* prepend() - 在被选元素的开头插入内容
* after() - 在被选元素之后插入内容
* before() - 在被选元素之前插入内容

# 7.function双参数提交

<a href="javascript:checkPayout('2','<%=projectPayout.getKeyId()%>')" class="list\_link">通过</a>

function checkPayout(checkStatusId,payoutId){

if(confirm("确定审核该记录吗？")){

document.getElementsByName("bean.id")[0].value=payoutId;

document.getElementsByName("bean.checkStatusId")[0].value=checkStatusId;

document.projectPayoutForm.actionType.value = "payoutCheck";

document.projectPayoutForm.submit();

}

}

# 8.div定位

父模块要用 style=" position:relative;"

子模块要用 Style=" position:absolute; top:12cm;"

# 9.鼠标移上去，行变色

<style type="text/css">

.table tr:hover,.table tr.hilite

{

background-color:#99FF66;

color:#0000CC;

}

</style>

/\*\*

\*鼠标移动到class为table 的表格上的某行时，进行突出显示

\*/

window.onload=function(){

var rows = document.getElementsByTagName('tr');//取得行

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

{

rows[i].onmouseover = function(){//鼠标移上去,添加一个类'hilite'

this.className += 'hilite';

}

rows[i].onmouseout = function(){//鼠标移开,改变该类的名称

this.className = this.className.replace('hilite','');

}

}

}

# 10.文件下载（网页开一个下载对话框）

<%

String path1 = request.getContextPath();

String basePath2 =

request.getScheme()+"://"+request.getServerName()+":"+request.getServerPort()+path1+"/";

%>

<script type="text/javascript">

function downLoad(){

window.open("<%=basePath2 %>Firefox\_google\_proxy.rar","文件导出");

}

</script>

# 11.对于某个控件加样式

.dropdown select {

border: 0 !important; /\*Removes border\*/

-webkit-appearance: none; /\*Removes default chrome and safari style\*/

-moz-appearance: none; /\* Removes Default Firefox style\*/

background: url('dropdown\_arrow.png') no-repeat; /\*Adds background-image\*/

background-position: 82px 7px; /\*Position of the background-image\*/

width: 100px; /\*Width of select dropdown to give space for arrow image\*/

text-indent: 0.01px; /\* Removes default arrow from firefox\*/

text-overflow: ""; /\*Removes default arrow from firefox\*/

color: #1455a2;

}

<div class="dropdown">

<p>Show: </p>

<select>

<option> 18 per page</option>

<option> 10 per page</option>

<option> 5 per page</option>

</select>

</div> <!-- DropDown -->

# 12.初始化加载js

## ⑴window.onload=function(){ //要初始化的东西 }

## ⑵$(document).ready(function(){ trace("初始化方法进入"); });

## ⑶$(function(){ trace("初始化方法进入二"); });

## ⑷jQuery(function($){ trace("初始化方法进入三"); });

# 13.tr隔行换色

<style type=*"text/css"*>

*.table* **tr***:hover***,***.table* **tr***.hilite*

{

background-color:*#99FF66*;

color:*#0000CC*;

}

</style>

<script type=*"text/javascript"*>

window.onload=**function**(){

**var** rows = document.getElementsByTagName('tr');//取得行

**for**(**var** i=0 ;i<rows.length; i++)

{

rows[i].onmouseover = **function**(){//鼠标移上去,添加一个类'hilite'

**this**.className += 'hilite';

}

rows[i].onmouseout = **function**(){//鼠标移开,改变该类的名称

**this**.className = **this**.className.replace('hilite','');

}

}

}

</script>

# 14.页面时间显示js

<script type=*"text/javascript"*>

**function** getNowFormatDate() {

//获取当前日期

**var** date\_time = **new** Date();

//定义星期

**var** week;

//switch判断

**switch** (date\_time.getDay()){

**case** 1: week="星期一"; **break**;

**case** 2: week="星期二"; **break**;

**case** 3: week="星期三"; **break**;

**case** 4: week="星期四"; **break**;

**case** 5: week="星期五"; **break**;

**case** 6: week="星期六"; **break**;

**default**:week="星期天"; **break**;

}

//年

**var** year = date\_time.getFullYear();

//判断小于10，前面补0

**if**(year<10){

year="0"+year;

}

//月

**var** month = date\_time.getMonth()+1;

//判断小于10，前面补0

**if**(month<10){

month="0"+month;

}

//日

**var** day = date\_time.getDate();

//判断小于10，前面补0

**if**(day<10){

day="0"+day;

}

//时

**var** hours =date\_time.getHours();

//判断小于10，前面补0

**if**(hours<10){

hours="0"+hours;

}

//分

**var** minutes =date\_time.getMinutes();

//判断小于10，前面补0

**if**(minutes<10){

minutes="0"+minutes;

}

//秒

**var** seconds=date\_time.getSeconds();

//判断小于10，前面补0

**if**(seconds<10){

seconds="0"+seconds;

}

//拼接年月日时分秒

**var** date\_str = year+"年"+month+"月"+day+"日 "+hours+":"+minutes+":"+seconds+" "+week;

//显示在id为showtimes的容器里

document.getElementById("showtimes").innerHTML= date\_str;

}

//设置1秒调用一次getNowFormatDate函数

setInterval("getNowFormatDate()",100);

</script>

<div id="showtimes"></div>

# 15.ajax异步判断

## ⑴页面

**function** checkName(){

//项目路径

**var** base = "<%=basePath%>";

**var** username=document.forms[0].username.value;

**var** xmlhttp;

**if** (window.XMLHttpRequest)

{

// IE7+, Firefox, Chrome, Opera, Safari 浏览器执行代码

xmlhttp=**new** XMLHttpRequest();

}**else**{

// IE6, IE5 浏览器执行代码

xmlhttp=**new** ActiveXObject("Microsoft.XMLHTTP");

}

//存储函数（或函数名），每当 readyState 属性改变时，就会调用该函数。

xmlhttp.onreadystatechange=**function**()

{

/\* 0: 请求未初始化 1: 服务器连接已建立 2: 请求已接收 3: 请求处理中 4: 请求已完成，且响应已就绪\*/

// 200: "OK" 404: 未找到页面

**if** (xmlhttp.readyState==4 && xmlhttp.status==200)

{

**var** responseTxt = xmlhttp.responseText;

**if**("此人员已经存在！" == responseTxt){

document.all["username"].focus();

}

document.getElementById("nameMsg").innerHTML=xmlhttp.responseText;

}

}

xmlhttp.open("GET",base+"/UserServlet?methodName=7&username="+username,**true**);

xmlhttp.send();

}

<font color="red"><p id="nameMsg"></p></font>

$.ajax({

type:"POST",

url:"hotTopics/hotTopicsInit.do",

dataType:"json",

success:**function**(data){

**if**(data){

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

**var** date\_time = **new** Date(data[i].createTime.time);

**var** year = date\_time.getFullYear();

//判断小于10，前面补0

**if**(year<10){

year="0"+year;

}

//月

**var** month = date\_time.getMonth()+1;

//判断小于10，前面补0

**if**(month<10){

month="0"+month;

}

//日

**var** day = date\_time.getDate();

//判断小于10，前面补0

**if**(day<10){

day="0"+day;

}

//时

**var** hours =date\_time.getHours();

//判断小于10，前面补0

**if**(hours<10){

hours="0"+hours;

}

//分

**var** minutes =date\_time.getMinutes();

//判断小于10，前面补0

**if**(minutes<10){

minutes="0"+minutes;

}

//秒

**var** seconds=date\_time.getSeconds();

//判断小于10，前面补0

**if**(seconds<10){

seconds="0"+seconds;

}

//拼接年月日时分秒

**var** date\_str = year+"年"+month+"月"+day+"日 "+hours+":"+minutes+":"+seconds+" ";

**if**(data[i].topicTitle.length<=13){

htmlstr2+='<a href="javascript:;" onclick="showContent('+data[i].topicId+',2)"><li class="ind1">'+(i+1)+".&nbsp;"+

'<font size=2>'+data[i].topicTitle+'</font><i style="float:right;">'+date\_str+'</i></a></li>';

}**else**{

htmlstr+='<li class="ind1"><span>'+(i+1)+'.</span>&nbsp;'+

'<a href="javascript:;" onclick="showContent('+data[i].topicId+',2)" title = "'+data[i].topicTitle+'"><font size=2 >'+data[i].topicTitle+'</font><i style="float:right;">'+date\_str+'</i></a></li>'

}

}

}

$('#hotTopics').html(htmlstr2);

}

});

## ⑵后台处理返回

//定义一个数据指针

PrintWriter out =**null**;

**try** {

//响应回复实现，堆变量指向

out = response.getWriter();

} **catch** (IOException e) {

e.printStackTrace();

}

out.write("此人员已经存在！");

}

# 16.js定时执行脚本

**var** globalLastSessionTime = **new** Date();

**function** calcSessionTime(){

**var** now = **new** Date();

**var** past = now - globalLastSessionTime;//ms

**if**(past>600000){

alert("Time out ,please login again.");

window.location.href="Newlogin.jsp";

}

}

**var** t1 = window.setInterval(calcSessionTime,30000);

# 17.判断按钮是否被点击

document.onclick=function(e){

var evt=e||window.event;

var tar=evt.target||evt.srcElement; if( (tar.tagName.toLowerCase()=="input"&&tar.type=="button") || tar.tagName.toLowerCase()=="button"){

alert("你点击的是一个按钮")

} }

# jQuery 鼠标事件

## 1.click事件

$('p').click(function(){

alert('click function is running');

});

## 2.dbclick事件

$('p').dbclick(function(){

alert('dbclick function is running');

});

## 3.mousedown事件

$('p').mousedown(function(){

alert('mousedown function is running');

}); <!--鼠标按下时候的事件-->

## 4.mouseup事件

$('p').mouseup(function(){

alert('mouseup function is running');

}).click(function(){

alert('click function is running');

}); <!--鼠标松开时候的事件,如果在同一元素上松开，那个click事件也会触发-->

## 5.mouseover事件

$('p').mouseover(function(){

alert('mouseover function is running');

});

## 6.mouseout事件

$('p').mouseover(function(){

alert('mouseout function is running');

});

## 7.mouseenter事件

$('p').mouseenter(function(){

alert('mouseenter function is running');

});

## 8.mouseleaver事件

$('p').mouseleaver(function(){

alert('mouseleaver function is running');

});

9. 7、8一般一起使用，在jquery中可以使用hover这个函数代替

$('p').hover(function(){

alert('mouseenter is running');

},function(){

alert('mouseleaver function is running');

});

# css样式

## 1.选择器包括

标记选择器(p{color:red;font-size:20px;}) 类别选择器（.r{} 样式 <... class="r" />） id选择器 (#p{} )

## 2.样式导入

1. <link href="red.css type="text/css" rel="stylesheet">

⑵ <style type="text/css"> @import url(red.css);</style>

优先级： 行内样式>链接式>内嵌式>导入式>链接式（在上面的时候） 后面的样式会覆盖前面的样式，

## 3.盒模型

相框边框--》border

画和相框边框的距离--》padding

相框之间的距离--》margin

用padding设置内边距；

用margin设置外边距。

border-width 边框宽度 border-color 边框颜色

## 6.position位置

relative：相对定位，相对于原本的标准位置偏移指定的距离

absolute：绝对定位，以它的包含框为基准进行偏移

fixed： 固定定位，以浏览器窗口为基准进行定位

z-index（空间位置）：值大的在值小的空间上方；

## 7.display

Inline：把元素变成内联元素

Block： 把元素变成块级元素

## 8.table表格

border：设置表格边框

border-collapse：设置边框分离，合并；默认是separate（分离） collapse 合并；

border-spacing：设置单元格的间距

padding：设置单元格内容和边框之间的距离

table-layout：默认是auto，自动方式，根据单元格的内容自动调整宽度；fixed 固定方式 表格宽度固定

隔行换色是隔行用相同的class属性，然后设置class属性为与邻近的不同颜色

## 9.a标签样式

a:LINK{color：red；text-decoration:none}

a:VISITED{color:green;}

a:HOVER{border-top:1px solid #EEEEEE;border-left:1px solid #EEEEEE;border-bottom:1px solid #717171;border-right:1px solid #717171;}

a:ACTIVE{border-top:1px solid #717171;border-left:1px solid #717171;border-bottom:1px solid #EEEEEE;border-right:1px solid #EEEEEE;}

# JSP

通过scriptlet标签可以在jsp中嵌入java代码

1.<%!%> 可以在里面定义全局变量、方法、类

2.<% %> 可以在里面定义局部变量、编写语句

3.<%= %> 可以在里面输出一个变量或者内容

请求范围

page 只在一个页面保存数据;javax.servlet.jsp.PageContext(抽象类)

Request 只在一个请求保存数据;javax.servlet.http.HttpServletRequest(接口)

Session 在一次会话范围内保存,仅供单个用户使用;javax.servlet.http.httpSession(接口)

Application 在整个服务器上保存,所有用户共享;javax.servlet.ServletContext(接口)

## 遍历map集合

var m = new Map();

m.set(1, "black");

m.set(2, "red");

m.set("colors", 2);

//方法一：

m.forEach(function (item) {

console.log(item.toString());

});

//方法二：

m.forEach(function (value, key, map) {

console.log(value)

})

//方法三：

for (var [key, value] of m) {

console.log(key + ' = ' + value);

}

## 遍历map对象

var map = [{

key : "百度",

value : "李彦宏" },

{

key : "阿里巴巴",

value : "马云" },

];

for (var key in map) {

console.log(map[key]);

}

# EL表达式

使用EL需要在页面引入<%@ taglib uri="http://java.sun.com/jsp/jstl/core" prefix="c" %> //prefix 是指的前缀

1. 取出值

Java后台代码：request.setAttribute(“result”,result1);

Jsp页面取值 ：${result}

1. 判断

<c:if test="${!empty result }"></c:if>

<c:choose>

<c:when test=*"*${user.rolesId == 2}*"*>

被禁用

</c:when>

<c:when test=*"*${user.rolesId == 3}*"*>

管理员

</c:when>

<c:otherwise>

普通用户

</c:otherwise>

</c:choose>

1. 循环

<c:forEach items="${result}" var="user">

</c:forEach>

1. 函数使用（字符串截取等）

页面需要引用两个

<%@ taglib uri=*"http://java.sun.com/jsp/jstl/core"* prefix=*"c"*%>

<%@ taglib uri=*"http://java.sun.com/jsp/jstl/functions"* prefix=*"fn"*%>

<c:if test="${fn:substring(sessionScope.role.noticeRight,0,1) == 1}"></c:if>

 fn:contains(string, substring)

假如参数string中包含参数substring，返回true

 fn:containsIgnoreCase(string, substring)

假如参数string中包含参数substring（忽略大小写），返回true

 fn:endsWith(string, suffix)

假如参数 string 以参数suffix结尾，返回true

 fn:escapeXml(string)

将有非凡意义的XML (和HTML)转换为对应的XML character entity code，并返回

 fn:indexOf(string, substring)

返回参数substring在参数string中第一次出现的位置

 fn:join(array, separator)

将一个给定的数组array用给定的间隔符separator串在一起，组成一个新的字符串并返回。

 fn:length(item)

返回参数item中包含元素的数量。参数Item类型是数组、collection或者String。假如是String类型,返回值是String中的字符数。

 fn:replace(string, before, after)

返回一个String对象。用参数after字符串替换参数string中所有出现参数before字符串的地方，并返回替换后的结果

 fn:split(string, separator)

返回一个数组，以参数separator 为分割符分割参数string，分割后的每一部分就是数组的一个元素

 fn:startsWith(string, prefix)

假如参数string以参数prefix开头，返回true

 fn:substring(string, begin, end)

返回参数string部分字符串, 从参数begin开始到参数end位置，包括end位置的字符

 fn:substringAfter(string, substring)

返回参数substring在参数string中后面的那一部分字符串

 fn:substringBefore(string, substring)

返回参数substring在参数string中前面的那一部分字符串

 fn:toLowerCase(string)

将参数string所有的字符变为小写，并将其返回

 fn:toUpperCase(string)

将参数string所有的字符变为大写，并将其返回

 fn:trim(string)

去除参数string 首尾的空格，并将其返回

1. 遍历

<c:forEach var="item" items="${map2}">

${item.key} > ${item.value} <br>

</c:forEach>

# Tip

## 1.让元素获取焦点

document.all["bean.budgetDate"].focus();

## 2.上传文件

表单input的file域，名字不能有相同的，不然只能传对象上去，具体内容传不到

## HTML中的特殊符号

&it < ; &gt > ; 上标 <sup> ; 下标 <sub> ;

## 图片中的alt属性

设置图片显示错误后出现的文字；title属性来设置图片标题

## 照片堆叠

<[div](https://www.baidu.com/s?wd=div&tn=44039180_cpr&fenlei=mv6quAkxTZn0IZRqIHckPjm4nH00T1Y3rjuBPHPWmHbkPyc3uHT30ZwV5Hcvrjm3rH6sPfKWUMw85HfYnjn4nH6sgvPsT6KdThsqpZwYTjCEQLGCpyw9Uz4Bmy-bIi4WUvYETgN-TLwGUv3EnWf4PjTYPH61) style="width:300px; height:300px; position:relative;">  
 <img src="下面的图片路径" />  
 <div style="width:200px; height:200px; position:absolute; top:10px; left:10px;"><img src="最上面的图片路径" /></div>  
</div>

## div之间的间距控制

{margin:top right bottom left }。

## Div中嵌套table

给table外面加一个div，给这个div设置一个左浮动就行了

## Js传值到后台中文乱码

1. 从get方式换到post方式
2. String str = new String(request.getParameter(“参数名”).getBytes(“iso-8859-1”), “utf-8”);

## 框架

一种常用的网页布局工具，作用是把浏览器分为几个部分，每个部分都可以独立显示不同的网页

1.<frameset > </frameset>

cols属性将窗口分为左右两个部分，数值是百分比

<frameset cols="40%,"><frame><frame> </frameset>

srows属性将窗口分为上中下三个部分

<frameset rows="30%,20%,"> <frame src> <frame><frame> </frameset>

框架之间可以进行嵌套 然后用src属性来向其中嵌入网页

2.嵌入式框架

<iframe> </iframe> 可以内嵌网页或者网址直接显示

## 元素

表格 1.colspan来合并左右单元格 rowspan来合并上下单元格

单选框 type="radio"

复选框 type="checkbox"

下拉框 <select> <option value="">1</option> </select>

JavaScript 弱类型语言，统一用var来定义变量

indexOf()来定位字符串中某一个指定字符首次出现的位置

日期对象 1.getTime()方法 返回从1970.1.1到现在的毫秒数

2.getFullYear() 从Date对象以四位数字返回年份

3.getMonth() 从Date对象返回月份（0-11）

4.getDate() 从Date对象返回一个月中的某一天（1-31）

5.getHours() 返回Date对象的小时(0-23)

6.getMinutes() 返回Date对象的分钟(0-59)

7.getSeconds() 返回Date对象的秒数(0-59)

8.getDay() 从Date对象返回一周中的某一天(0-6)

### 元素隐藏

document.getElementById("p1").style.visibility="hidden";

$('#btn3').prop("style","visibility:hidden;"); //将id为btn3的元素设置不可见。 .prop是为了兼容浏览器

### div定位

父模块要用 style=" position:relative;"

子模块要用 Style=" position:absolute; top:12cm;"

### 取值问题

1. 单选框

var entrustFlag = $("input[name='entrustFlag']:checked").val();

1. 复选框

var obj=document.getElementsByName("embodyTypes"); //选择所有name="'test'"的对象，返回数组

//取到对象数组后，我们来循环检测它是不是被选中

var s="";

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

if(obj[i].checked) s+=obj[i].value+","; //如果选中，将value添加到变量s

}

1. dom元素

document.getElementsByName("bean.isGrant")[0].value

$("[name=bean.isGrant]").val();

$(#要选择元素的id).val();

## Js做url参数隐藏

function submitForm(url, data)

{

　　var eleForm = document.body.appendChild(document.createElement('form'));

　　eleForm.action = url;

　　for (var property in data)

　　{

　　　　var hiddenInput = document.createElement('input');

　　　　hiddenInput.type = 'hidden';

　　　　hiddenInput.name = property;

　　　　hiddenInput.value = data[property];

　　　　eleForm.appendChild(hiddenInput);

　　}

　　this.eleForm = eleForm;

　　if (!submitForm.\_initialized)

　　{

　　　　submitForm.prototype.post = function ()

　　　 {

　　　　　　this.eleForm.method = 'post';

　　　　　　this.eleForm.submit();

　　　　};

　　submitForm.\_initialized = true;

　　}

}

function onAction1()

{

new submitForm('http://192.168.2.35:8080/DQQS/loginWyz.do', { username: 'admin',password:'admin' }).post(); // Url中的参数被隐藏

　　//location.href = "/Activity/ActivityInformation?a\_id=" + "1"; // Url中会带有参数

　　//new submitForm('/Activity/ActivityInformation', { a\_id: "1" }).post(); // Url中的参数被隐藏

}

## jsp页面数据存入json数组

后台

map-》 jsonString

JsonObject json = JsonObject.format();

前台

var data = eval("("+’${data}’+")")

for (var i = 0; i < itemJsonArray.length; i++) {  
var item = itemJsonArray[i];  
var obj = eval('(' + item + ')');  
alert(obj.name);  
  
}

## js对map进行操作

var m = new Map(); // 空Map m.set('Adam', 67); // 添加新的key-value m.set('Bob', 59); m.has('Adam'); // 是否存在key 'Adam': true m.get('Adam'); // 67 m.delete('Adam'); // 删除key 'Adam' m.get('Adam'); // undefined

## js jqery判断checkbox是否选中,全选,取消全选,反选,选择奇数偶数项

// 一,判断选中

　　// js

var ischecked2 = function(){

// this.checked == true

$(document.getElementsByTagName("input")).each(function(i){

if(this.checked == true){

console.log(this.getAttribute('name'));

}

})

}

// jquery

var ischecked1 = function(){

// 方法1,$(this).is(':checked')

$("input[type='checkbox']").each(function(i){

if($(this).is(':checked') == true){

console.log($(this).attr("name"));

}

});

// 方法2.$(this).get(0).checked

$("input[type='checkbox']").each(function(i){

if($(this).get(0).checked == true){

console.log($(this).attr("name"));

}

});

// 方法3.$(this).prop('checked')

$("input[type='checkbox']").each(function(i){

if($(this).prop('checked')){

console.log($(this).attr("name"));

}

});

}

// 二,全选

// jquery

var selectAll1 = function(){

// $("[type='checkbox']").prop("checked",true); // 方法1

$("[type='checkbox']").prop("checked","checked"); // 方法2

}

// js

var selectAll2 = function(){

$(document.getElementsByTagName("input")).each(function(i){

this.checked = true;

})

}

// 三,取消全选

// jquery

var selectNone1 = function(){

// $("[type='checkbox']").prop("checked",false); // 方法1

$("[type='checkbox']").prop("checked",""); // 方法2

}

// js

var selectNone2 = function(){

$(document.getElementsByTagName("input")).each(function(i){

this.checked = false;

})

}

// 四,反选

// jquery

var inverse1 = function(){

$("[type='checkbox']").each(function () {

$(this).prop("checked", !$(this).prop("checked"));

});

}

// js

var inverse2 = function(){

$(document.getElementsByTagName("input")).each(function(i){

this.checked = !this.checked;

})

}

// 五,jquery 选中所有奇数 (index从0开始)

var odd1 = function(){

selectNone1();

$("[type='checkbox']:odd").prop("checked",'true');

}

// 六,jquery 选中所有偶数 (index从0开始)

var even1 = function(){

selectNone1();

$("[type='checkbox']:even").prop("checked",'true');

}

## 用css控制鼠标样式

<span style="cursor:\*">文本或其它页面元素</span>

把 \* 换成如下15个效果的一种：

下面是对这15种效果的解释。移动鼠标到解释上面，看看你的鼠标起了什么变化吧！

hand是手型

pointer也是手型，这里推荐使用这种，因为这可以在多种浏览器下使用。

crosshair是十字型

text是移动到文本上的那种效果

wait是等待的那种效果

default是默认效果

help是问号

e-resize是向右的箭头

ne-resize是向右上的箭头

n-resize是向上的箭头

nw-resize是向左上的箭头

w-resize是向左的箭头

sw-resize是左下的箭头

s-resize是向下的箭头

se-resize是向右下的箭头

auto是由系统自动给出效果