**<!-- 2016/8/4 -->**

**1.请写出JQuery EasyUI和extJS常用组件**

**Jquery:**

**1.**[X-editable](http://vitalets.github.com/x-editable/index.html" \t "D:/%E6%A1%86%E6%9E%B6%E4%B8%AD%E7%A7%91%E8%BD%AF%E9%80%9A/%E5%91%A8%E5%90%89%E6%98%93/%E9%9D%A2%E8%AF%95%E9%A2%98/_blank)

这个插件能够让你在页面上创建可编辑的元素。它能够使用任何引擎（bootstrap、jquery-ui、jquery），并且包含弹出式和内联模式。

**2.**[Garlic.js](http://garlicjs.org/" \t "D:/%E6%A1%86%E6%9E%B6%E4%B8%AD%E7%A7%91%E8%BD%AF%E9%80%9A/%E5%91%A8%E5%90%89%E6%98%93/%E9%9D%A2%E8%AF%95%E9%A2%98/_blank)

Garlic.js 自动在本地保存表单文本字段的值，直到表单被提交。这样，你的用户如果不小心关闭选项卡或浏览器，也不会失去任何宝贵的数据。

**3.**[Pickadate.js](http://amsul.github.com/pickadate.js/" \t "D:/%E6%A1%86%E6%9E%B6%E4%B8%AD%E7%A7%91%E8%BD%AF%E9%80%9A/%E5%91%A8%E5%90%89%E6%98%93/%E9%9D%A2%E8%AF%95%E9%A2%98/_blank)

完全响应式和轻量级的 jQuery 日期输入选择器。

**4.**[jHERE](http://jhere.net/" \t "D:/%E6%A1%86%E6%9E%B6%E4%B8%AD%E7%A7%91%E8%BD%AF%E9%80%9A/%E5%91%A8%E5%90%89%E6%98%93/%E9%9D%A2%E8%AF%95%E9%A2%98/_blank)

有了 jHERE，你可以很容易地添加互动地图到您的网站中。拥有一个强大的地图 API，高度可定制的标记，事件处理和信息气泡。

**5.**[Responsive Img](http://responsiveimg.com/" \t "D:/%E6%A1%86%E6%9E%B6%E4%B8%AD%E7%A7%91%E8%BD%AF%E9%80%9A/%E5%91%A8%E5%90%89%E6%98%93/%E9%9D%A2%E8%AF%95%E9%A2%98/_blank)

Responsive Img 是一个 jQuery 插件，它能够根据页面的宽度改变一张图片的 src 属性。

**2.JSON是一种轻量级交互方式,用JSON语法表示如下Employee对象的数组:**

**name:张三,age:29,org:咨询二部**

**name:李四,age:24,org:增值业务部**

**[{"name":"张三","age":29,"org":"咨询二部"},{"name":"李四","age":24,"org":"增值业务部"}]**

**3.描述B/S中的session机制**

**HttpSession由服务器创建，是web容器提供的一种**

**安全性很高的会话跟踪机制，通过request.getSession()**

**创建新的HttpSession(中文称之为会话)，支持中文**

**通过setAttribute(String,Object)封装数据，通过**

**getAttribute(String)来拿取数据。HttpSession在创建后**

**开始进行倒计时，默认最大不活动周期为1800s（可以更改）**

**，在这期间如果用户再次提交请求，则重新倒计时。**

**HttpSession是一段时间概念，我们无法从本地找到一个具体的**

**HttpSession实体**

**HttpSession销毁的几种途径**

**1：超过最大不活动周期，在最大不活动周期到期后**

**依然没有任何请求提交，则HttpSession销毁**

**默认1800s**

**2：直接调用session.invalidate()**

**立刻销毁当前已经存在的session**

**3：关闭浏览器，注意关闭浏览器并不是严格意义上的**

**销毁当前session，而是当我们重新打开浏览器时**

**会再次分配给一个新的session，原来的老session无法**

**获取了**

**4.请写出IE和非IE下,采用Ajax与后台通讯的过程时xmlhttp对象的创建方式**

**var xmlhttp;**

**function create(){**

**if(window.XMLHttpRequest){**

**//非ie**

**xmlhttp = new XMLHttpRequest();**

**}else{**

**//ie**

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

**}**

**}**

**5.请说明FTP与SFTP协议的主要区别,并且用FTP命令写出下载192.168.0.100主机**

**interface下的a.txt文件至本地D盘根目录下的过程**

**FTP是文件传输协议。在网站上，如果你想把文件和人共享，最便捷的方式莫过于把文件上传到FTP服务器上，其他人通过FTP客户端程序来下载所需要的文件。**

**FTP进行文件传输需要通过端口进行。一般所需端口为：**

**1. 控制链路—TCP端口21。控制器端。用于发送指令给服务器以及等待服务器响应。**

**2. 数据链路---TCP端口20。数据传输端口。用来建立数据传输通道的。主要用来从客户向服务器发送一个文件、从服务器向客户发送一个文件、从服务器向客户发送文件或目录列表。**

**SFTP是Secure File Transfer Protocol的缩写，是安全文件传送协议。可以为传输文件提供一种安全的加密方法。跟ftp几乎语法功能一样。**

**SFTP是SSH的一部分，是一种传输档案至Blogger伺服器的安全方式。它本身没有单独的守护进程，必须使用sshd守护进程来完成相应的连接操作，所以从某种意义上来说，SFTP并不像一个服务器程序，而更像是一个客户端程序。SFTP同样是使用加密传输认证信息和传输的数据，所以使用SFTP是十分安全的。但由于这种传输方式使用了加密/解密技术，所以传输效率比普通的FTP要低得多。在对网络安全性要求更高时，代替FTP使用。**

**ftp 192.168.0.100**

**get /interface/a.txt**

**6.Tomcat作为常用的web服务器之一,请写出后台启动tomcat的过程以及实时监视cata..控件的命令**

**初始化部分1：**

**0. setCatalinaHome,setCatalinaBase，存入系统属性**

**1. 装载配置文件(catalina.properties)，写入System.setProperty**

**2. 创建classloader(common,server,shared),默认创建common classloader,**

**三者都使用StandardClassLoader(URL[])创建。**

**common的parent classloader为null，且它将作为其他两个的父classloader。**

**3. 向MBeanServer注册classloader，交给MBeanServer管理**

**4. setContextClassLoader(catalinaLoader),catalinaLoader即为server**

**classloader。**

**5. 如果可以获取到securitymanager，还要显示的对一些安全类进行预加载。**

**6**

**启动org.apache.catalina.startup.Catalina，并调用他的setParentClassLoader(sharedLoader)方法。**

**初始化部分2：**

**0.**

**解析启动参数指令，并将相应的参数赋值到catalina类中，同时写入System.setProperty，事实上，catalina是一个启动配置相关的脚本类。**

**1. 初始化目录，实际上就是catalina\_home,catalina\_base之类。**

**2.**

**初始化命名系统（initNaming），即向System.setProperty注入java.naming.factory.initial之类，用于上下文查询的工厂类是org.apache.naming.java.javaURLContextFactory**

**3.**

**解析server.xml配置文件——使用digester，解析配置文件中的层次结构，并将关系注入到对应的类中。创建connector时创建protocolhandler**

**4. 初始化standardserver，初始化阶段会触发fireLifecycleEvent，如果在外围注入相应的listner就可以监听事件**

**5. 注册standardserver到MBeanServer**

**6. 初始化service服务,**

**7. 注册standardservice到MBeanServer**

**8. 初始化connector**

**9. 注册connector到MBeanServer**

**10. 初始化protocolHandler**

**11. 初始化endpoint(底层socket初始化等)**

**开启部分：**

**0. 启动standardserver启动阶段会触发fireLifecycleEvent，如果在外围注入相应的listner就可以监听事件**

**1. 启动service服务,启动阶段会触发fireLifecycleEvent，如果在外围注入相应的listner就可以监听事件**

**2.**

**启动（也包含了init）container（这里指engine）,顺便启动（也包含了init）Realm,然后一次初始化（也包含了init）child加**

**container（这里主要是host）**

**接下来启动StandardPipeline，其主要工作是将pipeline中的value注册到MBeanServer中，默认只有一个basic**

**value，它在调用standardengine构造函数时被设置。**

**最后，启动container的后台线程ContainerBackgroundProcessor，它所做的工作在ContainerBackgroundProcessor类中**

**3. 启动executors**

**4.**

**启动connector，将protocolHandler注册到MBeanServer。初始化mapperListener并将它注册到MBeanServer。**

**初始化MBeanServer主要做以下工作：registerEngine，registerHost，registerContext，registerWrapper等和具体的web应用模块相关的问题。**

**5. 启动protocolHandler将endpoint和GlobalRequestProcessor注册到MBeanServer。**

**6. 启动endpoint，启动worker，acceptor线程**

**7.渲染css时，link和import的区别？**

页面中使用CSS的方式主要有3种：行内添加定义style属性值，页面头部内嵌调用和外面链接调用，其中外面引用有两种：link和@import。外部引用CSS两种方式link和@import的方式分别是：

XML/HTML代码  
<link rel="stylesheet" rev="stylesheet" href="CSS文件" type="text/css" media="all" />    
XML/HTML代码  
<style type="text/css" media="screen">    
@import url("CSS文件");    
</style>

两者都是外部引用CSS的方式，但是存在一定的区别：

　　区别1：link是XHTML标签，除了加载CSS外，还可以定义RSS等其他事务；@import属于CSS范畴，只能加载CSS。

　　区别2：link引用CSS时，在页面载入时同时加载；@import需要页面网页完全载入以后加载。

　区别3：link是XHTML标签，无兼容问题；@import是在CSS2.1提出的，低版本的浏览器不支持。

     区别4：ink支持使用Javascript控制DOM去改变样式；而@import不支持。

**8.ajax跨域？**

**解决ajax跨域问题方法1：**

**在服务器端直接设置header内容：Access-Control-Allow-Origin：\*    或者   Access-Control-Allow-Origin：允许访问的url**

**这样就可以直接请求到任何网站或者允许访问的 header('Access-Control-Allow-Origin：允许访问的url ');**

**9.绝对定位怎么实现居中？**

**.box{**

**position:absolute;**

**width:400px;**

**left:50%;**

**margin-left:-200px;/\* 此处负值是宽度的一半 \*/**

**}**

**10.页面如何实现延迟加载**

**1：延迟加载js代码**

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

**function demo(){}**

**$(function(){**

**setTimeout('demo()',1000); //延迟1s**

**})**

**</script>**

**2：将js代码放置在body底部最后加载**

**11.平时上传用的什么？前端的上传框架用过什么？**

**SmartUpload**

**ajaxFileUpload**

**12.css样式表根据所在网页的位置，可分为那几种？**

**1：外链式**

**<link type="text/css" href="" rel="stylesheet" />**

**2：内嵌式**

**书写在head标签内**

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

**3：行内式**

**书写在标签内**

**style=""**

**13.css选择符有哪些？哪些属性可以继承？优先级算法如何计算？内联和important哪一个优先级高？**

**Css选择器：**

**标记选择器：直接使用标签名作为选择的依据**

**id选择器：使用#id名作为选择的依据**

**类别选择器：使用.class名作为选择的依据**

**交集选择器：标记后面紧跟类别或者id必须同时满足**

**两个条件才可以成功选取**

**并集选择器：用逗号将多个选择器隔开，则满足其中任意**

**一个就可以成功选取**

**后代选择器：根据左祖先右后代原则，可以精确选取到**

**子元素**

**全选选择器：使用\*表示选择全部元素**

**不可继承的：display、margin、border、padding、background、height、min-height、max-height、width、min-width、max-width、overflow、position、left、right、top、bottom、z-index、float、clear、table-layout、vertical-align、page-break-after、page-bread-before和unicode-bidi。**

**所有元素可继承：visibility和cursor。**

**内联元素可继承：letter-spacing、word-spacing、white-space、line-height、color、font、font-family、font-size、font-style、font-variant、font-weight、text-decoration、text-transform、direction。**

**终端块状元素可继承：text-indent和text-align。**

**列表元素可继承：list-style、list-style-type、list-style-position、list-style-image。**

**原则一： 继承不如指定**

**原则二： #id > .class > 标签选择符**

**原则三：越具体越强大**

**原则四：标签#id >#id ; 标签.class > .class**

**CSS优先级权重计算法**

**CSS优先级包含四个级别（标签内选择符，ID选择符，Class选择符，元素选择符）以及各级别出现的次数!**

**根据这四个级别出现的次数计算得到CSS的优先级。**

**CSS优先级的计算规则如下：**

**\* 元素标签中定义的样式（Style属性）,加1,0,0,0**

**\* 每个ID选择符(如 #id),加0,1,0,0**

**\* 每个Class选择符(如 .class)、每个属性选择符(如 [attribute=])、每个伪类(如 :hover)加0,0,1,0**

**\* 每个元素选择符（如p）或伪元素选择符(如 :firstchild)等，加0,0,0,1**

**然后，将这四个数字分别累加，就得到每个CSS定义的优先级的值，**

**然后从左到右逐位比较大小，数字大的CSS样式的优先级就高。**

**important优先级高**

**14.javascript中split()和join（）的区别？**

**join() 方法用于把数组中的所有元素放入一个字符串。**

**元素是通过指定的分隔符进行分隔的。**

**指定分隔符方法join("#");其中#可以是任意**

**与之相反的是split()方法：用于把一个字符串分割成字符串数组.**

**stringObject.split(a,b)这是它的语法**

**a是必须的决定个从a这分割**

**b不是必须的，可选。该参数可指定返回的数组的最大长度** **。如果设置          了该参数，返回的子串不会多于这个参数指定的数组。如果没有设置该参数，整个字符串都会被分割，不考虑它的长度。**

**15.什么事ajax？ajax请求的时候get和post的区别？**

**AJAX即“Asynchronous Javascript And XML”（异步JavaScript和XML），是指一种创建交互式网页应用的网页开发技术。**

**AJAX = 异步 JavaScript和XML（标准通用标记语言的子集）。**

**AJAX 是一种用于创建快速动态网页的技术。**

**通过在后台与服务器进行少量数据交换，AJAX 可以使网页实现异步更新。这意味着可以在不重新加载整个网页的情况下，对网页的某部分进行更新。**

**传统的网页（不使用 AJAX）如果需要更新内容，必须重载整个网页页面**

区别:

1.get请求需注意缓存问题,post请求不需担心这个问题

2.post请求必须设置Content-Type值为application/x-form-www-urlencoded

3.发送请求时,因为get请求的参数都在url里,所以send函数发送的参数为null,而post请求在使用send方法时,却需赋予其参数

**16.请给出异步加载js的方案，不少于两种。**

**$.script()**

**$.ajax({**

**dataType:"script",**

**})**

**17.请写出jquery绑定事件的方法，不少于两种。**

**$(selector).bind("事件",function(){**

**})**

**$(selector).click(function(){**

**})**

**18.设置一个已知id为“div1”的div的html内容为aabbcc，字体颜色设置为黑色（不能使用第三方框架）。**

**jquery:**

**$("#div1").html("aabbcc").css("color","black");**

**js:**

**document.getElementById("divl").innerHTML="aabbcc";**

**document.getElementById("divl").style.color="black";**

**19.当一个dom节点被点击的时候，我们希望能够执行一个函数，应该怎么做？**

**$(dom).click(function(){**

**})**

**20.Js中String是基本数据类型么？基本数据类型有哪些？在内存中各占有多少个地址？**

**是**

**String number null boolean undefined**

以上5种类型在内存中占有固定大小的空间，它们的值保存在栈内存中。

**<！-- 2016/7/29 -->**

**1. 前端会多少？css和HTML结合能做个页面么？**

**可使用JspDay4\_frame课件页面**

**2. 假如说有个需求，在页面上有三个Tab，如何实现？用javaScript和HTML和css**

**html:**

**<div id="tabs0">**

**<ul class="menu0" id="menu0">**

**<li onclick="setTab(0,0)" class="hover">新闻</li>**

**<li onclick="setTab(0,1)">评论</li>**

**<li onclick="setTab(0,2)">技术</li>**

**<li onclick="setTab(0,3)">点评</li>**

**</ul>**

**<div class="main" id="main0">**

**<ul class="block"><li>新闻列表</li></ul>**

**<ul><li>评论列表</li></ul>**

**<ul><li>技术列表</li></ul>**

**<ul><li>点评列表</li></ul>**

**</div>**

**</div>**

**Css:**

**body,div,ul,li{**

**16.**

**17. margin:0 auto;**

**18.**

**19. padding:0;**

**20.**

**21.}**

**22.**

**23.body{**

**24.**

**25. font:12px "宋体";**

**26.**

**27. text-align:center;**

**28.**

**29.}**

**30.**

**31.a:link{**

**32.**

**33. color:#00F;**

**34.**

**35. text-decoration:none;**

**36.**

**37.}**

**38.**

**39.a:visited {**

**40.**

**41. color: #00F;**

**42.**

**43. text-decoration:none;**

**44.**

**45.}**

**46.**

**47.a:hover {**

**48.**

**49. color: #c00;**

**50.**

**51. text-decoration:underline;**

**52.**

**53.}**

**54.**

**55.ul{**

**56.**

**57. list-style:none;**

**58.**

**59.}**

**60.**

**61..main{**

**62.**

**63. clear:both;**

**64.**

**65. padding:8px;**

**66.**

**67. text-align:center;**

**68.**

**69.}**

**70.**

**71.**

**72.**

**73.#tabs0 {**

**74.**

**75. height: 200px;**

**76.**

**77. width: 400px;**

**78.**

**79. border: 1px solid #cbcbcb;**

**80.**

**81. background-color: #f2f6fb;**

**82.**

**83.}**

**84.**

**85..menu0{**

**86.**

**87. width: 400px;**

**88.**

**89.}**

**90.**

**91..menu0 li{**

**92.**

**93. display:block;**

**94.**

**95. float: left;**

**96.**

**97. padding: 4px 0;**

**98.**

**99. width:100px;**

**100.**

**101. text-align: center;**

**102.**

**103. cursor:pointer;**

**104.**

**105. background: #FFFFff;**

**106.**

**107.}**

**108.**

**109..menu0 li.hover{**

**110.**

**111. background: #f2f6fb;**

**112.**

**113.}**

**114.**

**115.#main0 ul{**

**116.**

**117. display: none;**

**118.**

**119.}**

**120.**

**121.#main0 ul.block{**

**122.**

**123. display: block;**

**124.**

**125.}**

**Javascript:**

**function setTab(m,n){**

**var tli=document.getElementById("menu"+m).getElementsByTagName("li");** 

**var mli=document.getElementById("main"+m).getElementsByTagName("ul");**

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

**tli[i].className=i==n?"hover":"";**

**mli[i].style.display=i==n?"block":"none";**

**}**

**}**

**4.<div id="testid" name="textname">,请写出至少2种隐藏此div的jquery语句**

**$("div#testid").hide();**

**$("div#testid").css("display","none");**

**$("div#testid").get(0).style.display = none;**

**5.如何将list，map数据放入一个json中，请写出格式。**

**分别放入**

**JSONArray ja = JSONArray.fromObject(list);**

**JSONObject jo = JSONObject.formObject(map);**

**放入一个JSON中**

**JSONObject jo = new JSONObject();**

**jo.put("key1",list);**

**jo.put("key2",map);**

**<！-- 2016/7/26 -->**

**1.cookie与session 的区别是什么？**

**两种会话跟踪机制**

**Cookie和HttpSession**

**1:Cookie由服务器创建，是浏览器提供的一种会话**

**跟踪机制，所有浏览器都支持cookie会话跟踪机制**

**cookie存在构造方法，直接可以在服务器端new出来**

**传递一个键值对，数据类型必须都为String，默认不**

**支持中文，一帮我们将可以分辨权限的数据放置在**

**cookie中用来对用户的权限等进行解析，当服务器**

**返回响应时，cookie一并返回客户端，并最终保存在**

**用户本地，我们从浏览器的设置中可以找到本地的**

**cookie实体文件**

**本地~~用户**

**服务器~~饭店**

**cookie~~vip会员卡**

**2:HttpSession由服务器创建，是web容器提供的一种**

**安全性很高的会话跟踪机制，通过request.getSession()**

**创建新的HttpSession(中文称之为会话)，支持中文**

**通过setAttribute(String,Object)封装数据，通过**

**getAttribute(String)来拿取数据。HttpSession在创建后**

**开始进行倒计时，默认最大不活动周期为1800s（可以更改）**

**，在这期间如果用户再次提交请求，则重新倒计时。**

**HttpSession是一段时间概念，我们无法从本地找到一个具体的**

**HttpSession实体**

**HttpSession销毁的几种途径**

**1：超过最大不活动周期，在最大不活动周期到期后**

**依然没有任何请求提交，则HttpSession销毁**

**默认1800s**

**2：直接调用session.invalidate()**

**立刻销毁当前已经存在的session**

**3：关闭浏览器，注意关闭浏览器并不是严格意义上的**

**销毁当前session，而是当我们重新打开浏览器时**

**会再次分配给一个新的session，原来的老session无法**

**获取了**

**HttpSession和Cookie有什么关系**

**HttpSession和cookie是两种会话跟踪机制，**

**表面上两者没有任何联系，HttpSession的应用场合**

**一般较高，两者没有任何关系吗?答案是否定的，**

**HttpSession并不能独立从一个页面传递到另外一个**

**页面，一般情况下它是通过浏览器提供的cookie会话**

**跟踪机制进行跳转的，**

**Cookie cookie = new Cookie("jsessionid",\*\*\*)**

**key为jsesssionid**

**value为session的id数**

**当开启的session会话跟踪机制后，cookie会自动**

**帮助session进行传递，我们不需要手动进行设置**

**这也是为什么在禁用cookie后，HttpSession会话跟踪**

**机制也会失效**

**禁用Cookie就无法维持Session会话跟踪机制了吗?**

**String path = "/ServletDay4\_cookie\_session/servlet/TestSession2";**

**//重写url**

**String finalPath = response.encodeURL(path);**

**response.sendRedirect(finalPath);**

**通过response.encodeURL(路径)**

**将sessionid封装进提交的url路径**

**则sessionid可以传递到新的页面，从而web容器**

**根据id提供的会话，这个会话就是在跳转前创建好的**

**（因为id一致），从而继续维持会话跟踪机制**

**<！-- 2016/7/20 -->**

**1.jQuery的选择器、怎么用**

**$("标签名")**

**直接使用标签名作为选择元素的依据**

**$(".class名")**

**直接使用.class属性名作为选择元素的依据**

**$("#id名")**

**直接使用#id属性名作为选择元素的依据**

**$("标签#id名")**

**$("标签.class名")**

**必须同时满足标签带有id或者相应class**

**的元素**

**$("选择器,选择器,选择器")**

**只要符合其中任意一个选择器就可以成功选取**

**$("选择器 选择器 选择器")**

**根据左祖先右后代的原则精确选取子元素**

**$("\*")**

**选择全部元素**

**$("选择器 > 选择器")**

**仅仅选取字辈元素**

**$("选择器 + 选择器")**

**选取的条件**

**1：从上向下选取**

**2：必须紧邻**

**3：必须互为兄弟**

**$("选择器 ~ 选择器")**

**向下选取所有的兄弟元素，不需要紧邻**

**$("选择器:first")**

**拿取指定元素序列中第一个**

**$("选择器:last")**

**拿取指定元素中最后一个**

**注意以上两个选择器选择都是**

**全文中的第一个和最后一个，**

**如果要选择每个ul中的第一个或者最后一个**

**请使用**

**:first-child**

**:last-child**

**$("选择器:eq(index)")**

**根据索引值拿取元素**

**$("选择器:gt(index)")**

**拿取索引值大于index的指定元素**

**&gt;**

**$("选择器:lt(index)")**

**&lt;**

**拿取索引值小于index的指定元素**

**$("选择器:contains('文本内容')")**

**拿取含有特定内容的元素**

**$("选择器:has(选择器)")**

**拿取含有特定内容得元素**

**$("选择器:hidden")**

**拿取被隐藏的特定元素**

**注意如果条件符合也包含**

**<input type="hidden" />**

**.text()拿取已经获得元素中的文本**

**$("选择器:visible")**

**拿取所有可见元素**

**$("选择器[属性名='属性值']")**

**拿取属性名为属性值的特定元素**

**$("选择器[属性名!='属性值']")**

**拿取属性名不为属性值的特定元素**

**html() 效果类似于 javascript中 innerHTML**

**$("选择器[属性名\*='部分属性值']")**

**拿取属性名部分匹配的特定元素**

**$("选择器[属性名^='属性的开头']")**

**拿取以属性名为\*\*\*开头的特定元素**

**$("选择器[属性名$='属性的结尾']")**

**拿取以属性名为\*\*\*结尾的特定元素**

**$(选择器:even)**

**拿取索引值为偶数和0的特定元素**

**$(选择器:odd)**

**索取索引值为奇数的特定元素**

**$(":header")**

**拿取h1~h6的标题**

**$(":input")**

**拿取所有的input表单元素**

**注意包含<textarea> <button> <select>**

**$(":text")**

**拿取单行文本输入框**

**$(":password")**

**拿取单行文本密码框**

**$(":radio")**

**拿取单选框**

**$(":checkbox")**

**拿取多选框**

**$(":image")**

**拿取input中type="image"的元素**

**注意此选择器不能拿取<img>标签**

**$(":submit")**

**拿取所有的提交按钮**

**val()拿取所获得元素的value属性**

**$(":reset")**

**拿取所有重置按钮**

**$(":button")**

**拿取button按钮**

**此选择器能够选择：**

**<input type="button" />和<button>**

**$(":checked")**

**拿取已经选中的元素**

**$(":selected")**

**下拉框被选中的元素**

**<！-- 2016/7/18 -->**

**1、css的布局怎么写**

**一，流体浮动布局**

**描述：当视图变化时，跟着变化，设置百分比大小，采用浮动来实现**

**二，流体定位布局**

**描述：当视图变化时，跟着变化，设置百分比大小，用定位方式实现**

**三，固定浮动布局**

**描述：固定大小，不能随着视图改变而改变，设置固定大小，用浮动来实现**

**四，固定定位布局**

**描述：固定大小，不能随着视图改变而改变，设置固定大小，用定位实现**

**浮动：float:浮动的方向;**

**定位：分四种**

**1：相对定位**

**相对于元素原先所在位置的左上角，**

**定位之后元素依然占据一行，原先位置被占用**

**2：绝对定位**

**相对于距离元素最近的定位过的父元素的左上角**

**进行定位，定位改变元素属性，不再占据一行，原先位置**

**不再占据**

**3：fixed定位**

**元素永远按照body来定位**

**4：static定位**

**不进行任何定位，呈现标准文本流形式**

**2、选择器都有哪些**

**Css选择器：**

**标记选择器：直接使用标签名作为选择的依据**

**id选择器：使用#id名作为选择的依据**

**类别选择器：使用.class名作为选择的依据**

**交集选择器：标记后面紧跟类别或者id必须同时满足**

**两个条件才可以成功选取**

**并集选择器：用逗号将多个选择器隔开，则满足其中任意**

**一个就可以成功选取**

**后代选择器：根据左祖先右后代原则，可以精确选取到**

**子元素**

**全选选择器：使用\*表示选择全部元素**

**3.9个内置对象**

**JSP九种内置对象：request, response, session, application, page, pageContext, out, exception, config**

**1.request对象**

**客户端的请求信息被封装在request对象中，通过它才能了解到客户的需求，然后做出响应。它是HttpServletRequest类的实例**

**2.response对象**

**response对象包含了响应客户请求的有关信息，但在JSP中很少直接用到它。它是HttpServletResponse类的实例。**

**3.session对象**

**session对象指的是客户端与服务器的一次会话，从客户连到服务器的一个WebApplication开始，直到客户端与服务器断开连接为止。它是HttpSession类的实例.**

**4.out对象**

**out对象是JspWriter类的实例,是向客户端输出内容常用的对象**

**5.page对象**

**page对象就是指向当前JSP页面本身，有点象类中的this指针，它是java.lang.Object类的实例**

**6.application对象**

**application对象实现了用户间数据的共享，可存放全局变量。它开始于服务器的启动，直到服务器的关闭，在此期间，此对象将一直存在；这样在用户 的前后连接或不同用户之间的连接中，可以对此对象的同一属性进行操作；在任何地方对此对象属性的操作，都将影响到其他用户对此的访问。服务器的启动和关闭 决定了application对象的生命。它是ServletContext类的实例。**

**7.exception对象**

**exception对象是一个异常对象，当一个页面在运行过程中发生了异常，就产生这个对象。如果一个JSP页面要应用此对象，就必须把isErrorPage设为true，否则无法编译。他实际上是java.lang.Throwable的对象**

**8.pageContext对象**

**pageContext对象提供了对JSP页面内所有的对象及名字空间的访问，也就是说他可以访问到本页所在的SESSION，也可以取本页面所在的application的某一属性值，他相当于页面中所有功能的集大成者，它的本 类名也叫pageContext。**

**9.config对象**

**config对象是在一个Servlet初始化时，JSP引擎向它传递信息用的，此信息包括Servlet初始化时所要用到的参数（通过属性名和属性值构成）以及服务器的有关信息（通过传递一个ServletContext对象）**

**4.重定向怎么写**

**response.sendRedirect("/工程名/servlet/类名");**

**response.sendRedirect("/工程名/\*\*\*.jsp");**

**5.标签有哪些**

**Html标签**

**JSP内建动作**

**JSTL（java standard tag lib）jstl标签**

**Struts1标签**

**Struts2标签**

**<！-- 2016/7/12 -->**

1. **jsp标签用过吗  
   使用过 JSP内建动作 JSTL标签**
2. **Jquery选择器**

**分为基本选择器，过滤选择器，表单选择器**

**等，详情参见课件JqueryDays 中的 selector1.jsp~**

**selector3.jsp 共五十余种选择器**

**<！-- 2016/7/11 -->**

**1: 前台页面上怎么处理数据量比较大的问题**

**适当加入分页，筛选机制**

**2: Jquery怎么发送请求？**

**$.get(url,data,callback)**

**$.post(url,data,callback)**

**$.getJSON(url,data,callback)**

**url为发送请求目的地，data为传输数据，callback为回调函数**

**3: Jquery中你用过哪些它独有的函数？**

**($.each()、$.ajax()、$.get()、$.post()我想到的就这些)**

**4: 纯js怎么实现a页面跳转到b页面？跳转过去后，点击浏览器上的返回键返回不了怎么办？**

**跳转：window.location = b;**

**或者getElementByTagName("a")[0].href=b;**

**返回：history.go(-1);**

**5: easyui怎么把一个表单分成三列？怎么设置各占50%？**

**6: Jquery中的ajax请求？**

**$.ajax({**

**//要提交到目的地**

**url:"servlet/Page",**

**//1.2新特性，设置为false则不缓存此页面**

**cache:false,**

**//默认true，为异步请求，更改为false则变为同步请求，浏览器被锁住**

**async:true,**

**//异步请求的类型**

**type:"post",**

**//要传递的值 ,常使用serialize()方法序列化表单**

**data:"key="+value,**

**+"&key2="+value2,**

**//返回的值的类型支持 json html text script 和 xml**

**dataType:"json",**

**success:function(data){**

**//服务器无误是回调函数**

**}，**

**error:function(){**

**//服务器出现异常时回调函数**

**}**

**7: EasyUI  中的treagird  select怎么用？怎么添加值，具体怎么用？如果是javascript怎么用？**

**8: 页面怎么发送一个List？（问完之后又问）不用ajax呢？**

**使用ajax发送json给后台。**

**$.ajax({**

**type:'POST',**

**url : xxx.action",**

**data : {"list" : '把list拼成json数据'},**

**dataType : "json",**

**success : function(data){**

**//成功处理**

**}**

**});**

**不使用ajax可以将其封装在范围中，一般常用**

**request.setAttribute("key",list);**

**或者**

**<c:set var="key" value="<%=list %>" scope="范围" />**

**9: mvc的controller怎么处理分页的**

**接受前台传递过来的当前页默认值以及每页记录数，调用**

**数据拿取总记录数，通过计算取出总页数，根据分页查询公式**

**拿取分页查询的集合，将总记录数 总页数 集合封装进json中**

**发送给页面，页面获取后拿取数据**

**<！-- 2016/7/10 -->**

**1.ajax如何实现同步？同步异步的区别？**

**request.open(method,url,async,username,password);**

**async:true为异步**

**false为同步**

**异步方式客户端不需要等待服务器的响应。同步方式客户端必须等待服务器响应之后才可以继续执行后续操作。要根据实际情况选择方式，异步效率高但是对服务器造成更高的负载，当某些页面可能会发出多个请求，甚至是有组织的有计划的有队形的高强度的request时，后一个是会覆盖前一个的，则应该使用同步**

**<！-- 2016/7/7 -->**

**1.用没用过 jqueryui**

**用过**

**2.动态表单怎么弄**

<html >

< head>

     < title>动态添加表单元素</title>

< /head>

< script language="javascript">

function AddElement(mytype){

var mytype,TemO=document.getElementById("add");

var newInput = document.createElement("input");

newInput.type=mytype;

newInput.name="input1";

TemO.appendChild(newInput);

var newline= document.createElement("br");

TemO.appendChild(newline);

}

< /script>

< body>

< form action="" method="get" name="frm">

< div id="add">

           <input type="text" name="textfield">

< /div>

< /form>

< input name="" type="button" value="新建文本框" onClick="AddElement('text')" />

< input name="" type="button" value="新建复选框" onClick="AddElement('checkbox')" />

< input name="" type="button" value="新建单选框" onClick="AddElement('radio')" />

< input name="" type="button" value="新建文件域" onClick="AddElement('file')" />

< input name="" type="button" value="新建密码框" onClick="AddElement('password')" />

< input name="" type="button" value="新建提交按钮" onClick="AddElement('submit')" />

< input name="" type="button" value="新建恢复按钮" onClick="AddElement('reset')" />

< /body>

< /html>

**3.怎么解析动态表单生成的文件(貌似是这么问的)**

**4.配置过拦截器吗，如果配置了多个拦截器，怎么控制它对请求的拦截顺序**

**在struts.xml文件中配置<interceptor-ref name="">节点从上到下顺序拦截**

**5.get  post   区别 怎么用**

**get:默认不支持中文，通过浏览器url进行传递值**

**?key=value&key=value最多传递256个字符，速度快**

**安全性低下，一般链接肯定是get请求**

**post:支持中文，通过消息体进行值得传递，无法从地址栏**

**查询传递的值，没有大小限制，速度慢，但是安全性较高**

**一般上传操作必须使用post**

**6.表单提交 ajax**

**$.ajax({**

**//要提交到目的地**

**url:"servlet/Page",**

**//1.2新特性，设置为false则不缓存此页面**

**cache:false,**

**//默认true，为异步请求，更改为false则变为同步请求，浏览器被锁住**

**async:true,**

**//异步请求的类型**

**type:"post",**

**//要传递的值 ,常使用serialize()方法序列化表单**

**data:$("form").serialize(),**

**+"&key2="+value2,**

**//返回的值的类型支持 json html text script 和 xml**

**dataType:"json",**

**success:function(data){**

**//服务器无误是回调函数**

**}，**

**error:function(){**

**//服务器出现异常时回调函数**

**}**

**7.tomcat端口号**

**默认为8080端口，可以在tomcat的conf文件夹中的server.xml**

**中更改**

**<！-- 2016/7/6 -->**

**1.重定向与请求转发**

**请求转发：**

**是同一次请求，跳转之后浏览器地址栏依然是**

**跳转之前的地址，跳转之前是do\*\*\*方法**

**跳转之后依然是do\*\*\*方法，因为是同一次请求，**

**所以request范围有效。** **请求转发代码执行后后面的代码依然会执行；浏览器----》A----》B**

**重定向**

**不是同一次请求，跳转之后浏览器地址栏是最终的目的地**

**跳转之后肯定是doGet(),request范围会失效。**

**重定向代码执行后后面的代码依然会执行**

**浏览器----》A----》浏览器----》B**

**2.前端比如修改easyui既有样式可不可以**

**可以部分控件可以直接通过参数修改**

**3.get和post的区别？**

**method:表单提交的方式**

**get：速度快安全性较低，不支持中文**

**通过浏览器url进行值得传递，用户可以从**

**浏览器地址栏直接查看值传递的情况**

**base1.html?key1=value1&key2=value2&key3=value3\*\*\***

**最多传递256个字符**

**post：速度慢安全性高，通过消息体传递值**

**无法查看，没有大小限制支持中文**

**4.css有哪几种方式？（当时没明白他什么意思，后来才知道他想问的是css的三种使用方式）**

**1:内嵌式**

**将css代码直接书写在head标签的**

**style标签内**

**type:表示MIME格式**

**这种格式用来给浏览器解析让浏览器**

**知道接下来是什么技术**

**内嵌式将页面结构和样式初步的解耦，但是**

**代码重用性较低**

**2:外链式**

**引入一个外部的css独立文件到本页面**

**rel:stylesheet表示引入的是样式表**

**type:MIME格式文件**

**href:引入外部独立css文件的路径**

**应用最为广泛，代码重用性强，将页面结构和样式解耦**

**3:行内式**

**直接将css代码书写在标签内**

**这种方式将页面结构和样式严重耦合在一起**

**三种引入css方式的优先级**

**行内式> 外联式和内嵌式谁放在后面会覆盖之前出现冲突的**

**css渲染效果**

**5.怎么让a标签中的字体变成红色，修改所有的a标签中的字体样式（怎么强制指定css）**

**<a style="color:red;font-family:字体;">\*\*\*</a>**

**a{**

**font-family:字体;**

**}**

**6.Js选择器由哪些？**

**getElementById()**

**根据id拿取一个元素**

**getElementsByName()**

**根据name属性拿取多个元素**

**getElementsByTagName()**

**根据标签名拿取多个元素**

**剩下的如果面试官说还有，可以立刻怼它那些在api文档中不被承认，也查询不到，不正规的方法是不应该广泛使用的，更多选择器请不要使用这种老掉牙的古董，应该使用jquery选择器**

**7.两个div的id属性相同时，使用js选择器选中该id元素会取到什么？**

**会取到第一个元素，不推荐这样做，getElementById只能拿到第一个元素**

**8.如何使用js操作html（不让用html（），说html（）不常用，使用起来很麻烦）**

**修改元素html结构：**

**document.getElementById(id).innerHTML=new HTML**

**修改元素属性：**

**document.getElementById(id).attribute=new value**

**9.选中一个div，怎么向上选取它的父元素向下选取它的子级元素**

**$("div").parent();**

**$("div").children();**

**10.Parent和parents有什么区别**

**parent:就是当前元素的父元素**

**parents:则是当前元素的祖先元素**

**<div id='div1'>**

**<div id='div2'><p></p></div>**

**<div id='div3' class='a'><p></p></div>**

**<div id='div4'><p></p></div>**

**</div>**

**$('p').parent()取到的是div2,div3,div4**

**$('p').parent('.a')取到的是div3**

**$('p').parent().parent()取到的是div1**

**$('p').parents()取到的是div1,div2,div3,div4**

**$('p').parents('.a')取到的是div3**

**11.怎么使用js控制样式**

**用javascript获取和设置style**

**DOM标准引入了覆盖样式表的概念**

**document.getElementById( id ).style.属性名 = 值;**

**12.Ajax的请求有哪几种方式（jquery中ajax自己的方式）**

**$.ajax()**

**$.get()**

**$.post()**

**13.Ajax中常用的参数有哪几个，返回值默认是什么类型的，不用dataType指定怎么传json类型的数据**

**$.ajax({**

**//要提交到目的地**

**url:"servlet/Page",**

**//1.2新特性，设置为false则不缓存此页面**

**cache:false,**

**//默认true，为异步请求，更改为false则变为同步请求，浏览器被锁住**

**async:true,**

**//异步请求的类型**

**type:"post",**

**//要传递的值 ,常使用serialize()方法序列化表单**

**data:"key="+value**

**+"&key2="+value2,**

**//返回的值的类型支持 json html text script 和 xml**

**dataType:"json",**

**success:function(data){**

**//服务器无误是回调函数**

**}，**

**error:function(){**

**//服务器出现异常时回调函数**

**}**

**不指定dataType为json可以指定为text，则返回普通字符串**

**通过json转换器进行转换**

**JSON.parse(返回的字符串);**

**使用这种方式则放弃jquery的自动转换功能**

**采用ajax原始的转换json数据方式，key值转换为属性名，**

**value值转换为属性值**

**14.纯js怎么实现页面跳转**

**A：使用 window.location = "要跳转的页面"**

**B：获得a元素通过.href的方式修改href属性值**

**15.如果页面A中有个定时器,10秒之后触发一个方法,该怎么实现**

**在javascritp中，有两个关于定时器的专用函数，分别为：**

**1.倒计定时器：timename=setTimeout("function();",delaytime);**

**2.循环定时器：timename=setInterval("function();",delaytime);**

**第一个参数“function()”是定时器触发时要执行的动作，可以是一个函数，也可以是几个函数，函数间用“；”隔开即可。比如要弹出两个警告窗口，便可将“function();”换成**

**“alert('第一个警告窗口!');alert('第二个警告窗口!');”；而第二个参数“delaytime”则是间隔的时间，以毫秒为单位，即填写“5000”，就表示5秒钟。**

**倒计时定时器是在指定时间到达后触发事件，而循环定时器就是在间隔时间到来时反复触发事件，两者的区别在于：前者只是作用一次，而后者则不停地作用。**

**16.div怎么隐藏,然后通过点击事件再让他出来**

**$("div").hide();**

**$("button").click(function(){**

**$("div").show();**

**})**

**<！-- 2016/7/5 -->**

**1.css用的熟练吗，举个例子，我想给页面加一个渐变色，问用css怎么加？**

**对应不同浏览器效果会有偏差**

**body{**

**background-image: -moz-linear-gradient(top, #8fa1ff, #3757fa); /\* Firefox \*/**

**background-image: -webkit-gradient(linear, left top, left bottom, color-stop(0, #ff4f02), color-stop(1, #8f2c00)); /\* Saf4+, Chrome \*/**

**filter: progid:DXImageTransform.Microsoft.gradient(startColorstr='#c6ff00', endColorstr='#538300', GradientType='0'); /\* IE\*/**

**}**

**-moz-linear-gradient有三个参数。第一个参数表示线性渐变的方向，top是从上到下、left是从左到右，如果定义成left top，那就是从左上角到右下角。第二个和第三个参数分别是起点颜色和终点颜色。你还可以在它们之间插入更多的参数，表示多种颜色的渐变。**

**-webkit-gradient是webkit引擎对渐变的实现参数，一共有五个。**

**第一个参数表示渐变类型（type），可以是linear（线性渐变）或者radial（径向渐变）。**

**第二个参数和第三个参数，都是一对值，分别表示渐变起点和终点。这对值可以用坐标形式表示，也可以用关键值表示，比如 left top（左上角）和left bottom（左下角）。**

**第四个和第五个参数，分别是两个color-stop函数。color-stop函数接受两个参数，第一个表示渐变的位置，0为起点，0.5为中点，1为结束点；第二个表示该点的颜色。**

**IE依靠滤镜实现渐变。startColorstr表示起点的颜色，endColorstr表示终点颜色。GradientType表示渐变类型，0为缺省值，表示垂直渐变，1表示水平渐变。**

**2.如果我要往一个空的div里面加别的标签该怎么实现，如果往一个非空的div加标签怎么实现，不用html怎么实现？**

**jquery:**

**$("div").html(添加其他的标签)**

**$("div").append(添加的内容不影响原来的内容)**

**3.如果两个div的ID一样，我在JS里面选中ID，它会选中几个div**

**会选中第一个，但是不推荐这样做因为在同一个页面中**

**id建议唯一，因为document.getElementById("")仅仅拿取一个元素**

**4.问如果两个class一样，我选中class，然后它会选中页面上几个div？**

**两个全部拿取，class可以不唯一**

**5.jquery怎么发送ajax，口头描述，ajax你传值常用的类型是什么？**

**$.ajax({**

**})**

**$.get()**

**$.post()**

**传值可以使用 html text json script xml**

**6.如果我同时发送多个ajax，多个异步或是多个同步，会出现什么问题。**

**发送多个同步请求效率会慢，必须等待上一个结束才能执行下一个**

**发送多个同步请求则可能会导致返回的结果混乱**

**7.怎么样在一个方法里面这两个ajax，其中第二个ajax，需要用到第一个的传值，可不可以两个挨着写**

**不能，即便挨着写也是两个不同的异步请求，很可能第二个ajax在发出请求的时候第一个ajax的值还没有传回来 同步两个请求**

**8.同步ajax跟异步ajax有什么不同，那个好用？**

**request.open(method,url,async,username,password);**

**async:true为异步**

**false为同步**

**异步方式客户端不需要等待服务器的响应。同步方式客户端必须等待服务器响应之后才可以继续执行后续操作。要根据实际情况选择方式，异步效率高但是对服务器造成更高的负载，当某些页面可能会发出多个请求，甚至是有组织的有计划的有队形的高强度的request时，后一个是会覆盖前一个的，则应该使用同步**

**9.同步ajax，用么技术实现？**

**原生ajax中**

**request.open(method,url,async,username,password);**

**async:true为异步**

**false为同步**

**在jquery的$.ajax()方法中设置async参数为false也可以发送同步请求**

**10.前端框架用过什么？**

**Jquery EasyUI**

**11.tomcat如何改端口号？如果有多个Tomcat的时候，怎么处理的？**

**修改Server.xml文件，把8080端口改成80就可以了。**

**8080是Tomcat服务器的默认的端口号。我们可以通过修改Tomcat服务器的conf目录下的主配置文件server.xml来更改.用记事本打开server.xml文件，找到如下部分:**

**<Connector port="8080" maxHttpHeaderSize="8192"**

**maxThreads="150" minSpareThreads="25" maxSpareThreads="75"**

**enableLookups="false" redirectPort="8443" acceptCount="100"**

**connectionTimeout="20000" disableUploadTimeout="true" />**

**将其中的port="8080"更改为新的端口号即可**

**其中涉及到的很多其它port也应该做修改才能保证多个tomcat同时启动**

**12.了解html5,css3？**

**html5：**

**为 HTML5 建立的一些规则：**

* **新特性应该基于 HTML、CSS、DOM 以及 JavaScript。**
* **减少对外部插件的需求（比如 Flash）**
* **更优秀的错误处理**
* **更多取代脚本的标记**
* **HTML5 应该独立于设备**
* **开发进程应对公众透明**

**HTML5 中的一些有趣的新特性：**

* **用于绘画的 canvas 元素**
* **用于媒介回放的 video 和 audio 元素**
* **对本地离线存储的更好的支持**
* **新的特殊内容元素，比如 article、footer、header、nav、section**
* **新的表单控件，比如 calendar、date、time、email、url、search**

**可适当介绍因为html5推行时因为禁用flash而导致的纷争引起的推广障碍以及微软ie浏览器**

**兼容性差，最好将课上段子引入**

**CSS3**

**CSS即层叠样式表（Cascading StyleSheet）。 在网页制作时采用层叠样式表技术，可以有效地对页面的布局、字体、颜色、背景和其它效果实现更加精确的控制。 只要对相应的代码做一些简单的修改，就可以改变同一页面的不同部分，或者页数不同的网页的外观和格式。CSS3是CSS技术的升级版本，CSS3语言开发是朝着模块化发展的。以前的规范作为一个模块实在是太庞大而且比较复杂，所以，把它分解为一些小的模块，更多新的模块也被加入进来。这些模块包括： 盒子模型、列表模块、超链接方式 、语言模块 、背景和边框 、文字特效 、多栏布局等。**

**13.tomcat的原理？**

**当浏览器访问一个由tomcat搭建的web应用程序时，tomcat会把相应的文件，转换成浏览器能够显示的文件。**

**浏览器访问的是一个jsp文件时，tomcat会把jsp文件编译成一个servlet，servlet再给浏览器发送相应的数据。只有第一次访问的时候才会编译成servlet以后访问就直接调用对应的servlet。**

**当浏览器访问的是servlet和html时，tomcat就会直接传递给浏览器。**

**14.用没用过tree?**

**15.controller层数据怎么返回到页面?**

**servlet：通过四范围传递值或者通过？传递值**

**struts1：通过out.print();**

**ajax：通过out.print();**

**16.如何取得当前时间?JS**

**var myDate = new Date();  
myDate.getYear();        //获取当前年份(2位)  
myDate.getFullYear();    //获取完整的年份(4位,1970-????)  
myDate.getMonth();       //获取当前月份(0-11,0代表1月)  
myDate.getDate();        //获取当前日(1-31)  
myDate.getDay();         //获取当前星期X(0-6,0代表星期天)  
myDate.getTime();        //获取当前时间(从1970.1.1开始的毫秒数)  
myDate.getHours();       //获取当前小时数(0-23)  
myDate.getMinutes();     //获取当前分钟数(0-59)  
myDate.getSeconds();     //获取当前秒数(0-59)  
myDate.getMilliseconds();    //获取当前毫秒数(0-999)  
myDate.toLocaleDateString();     //获取当前日期  
var mytime=myDate.toLocaleTimeString();     //获取当前时间**

**myDate.toLocaleString( );        //获取日期与时间**

**17.文件上传下载?**

**使用SmartUpload**

**//使用SmartUpload插件进行文件的上传**

**//1：引入SmartUpload**

**SmartUpload su = new SmartUpload();**

**//2：设置允许上传文件的后缀名**

**su.setAllowedFilesList("jpg,jpeg,gif,bmp");**

**//3：设置允许上传文件的总大小 这里设置的是6m**

**su.setTotalMaxFileSize(6\*1024\*1024);**

**//4：初始化**

**su.initialize(this.getServletConfig(),**

**request,response);**

**//5:上传**

**su.upload();**

**//设置一个路径，将文件最终保存在此路径下**

**//可以使用绝对路径，也可以使用相对路径**

**//因为目前的定位就在服务器端，所以使用相对路径直接在服务器选取**

**//拿取所有上传的文件**

**Files files = su.getFiles();**

**//拿取唯一上传文件**

**File file = files.getFile(0);**

**//拿取上传文件的全名**

**String fileName = file.getFileName();**

**//组合一个要上传的路径**

**String path = "/image/"+fileName;**

**//另存到此路径**

**file.saveAs(path);**

**使用SmartUpload下载：**

**SmartUpload su = new SmartUpload();**

**//设置保存的地点，注意填写null由浏览器提示保存地址**

**su.setContentDisposition(null);**

**su.initialize(this.getServletConfig(), request,response);**

**su.downloadFile(路径);**

**<！-- 2016/7/4 -->**

**01: 4范围及区别**

**PageContext pageContext  页面跳转立刻失效**

**HttpRequest request      重定向失效，因为不是同一次请求了**

**HttpSession session      session销毁后失效（默认最大不活动周期1800s）**

**ServletContext application tomcat开启情况下不会失效**

**所谓失效有效是指 在setAttribute(String,Object)后是否**

**可以通过getAttribute(String)取出**

**02: J2EE的规范**

**J2EE存在如下13种技术规范**

**1、JDBC（java Database Connectivity）：**

**JDBC API为访问不同的数据库提供了一种统一的途径，就像ODBC一样，JDBC对开发者屏蔽了一些细节问题，同时，JDBC对数据库的访问也具有平台无关性。**

**2、JNDI(Java Name and Directory Interface):**

**JNDI API 被用于执行名字和目录服务。它提供了一致的模型用来存取和操作企业级的资源如DNS和LDAP，本地文件系统，或应用服务器中的对象。**

**3、EJB（Enterprise JavaBean）：**

**J2ee技术之所以赢得全体广泛重视的原因之一就是EJB，他们提供了一个框架开发和实施分布式商务逻辑，由此很显著简化了具有可伸缩性和高度复杂的企业级应用开发。EJB规范定义了EJB组件何时如何与他们的容器继续拧交互作用。容器负责提供公用的服务，例如目录服务、事务管理、安全性、资源缓冲池以及容错性。但是注意的是，EJB并不是J2EE的唯一途径。正是由于EJB的开放性，使得有的厂商能够以一种和EJB平行的方式来达到同样的目的。**

**4、RMI（RemoteMethod Invoke）：remote（遥远的） invoke（调用）：**

**正如其名字所表示的那样，RMI协议调用远程对象上方法。它使用了序列化方式在客户端和服务器端传递数据。RMI是一种被EJB使用的更底层的协议。**

**5、Java IDL（接口定义语言）/CORBA：公共对象请求代理结构（Common Object Request Breaker Architecture）：**

**在java IDL的支持下，开发人员可以将Java和CORBA集成在一起。他们可以创建Java对象并使之可以在CORBA ORB中展开，或者他们还可以创建Java类并做为和其他ORB一起展开的CORBA对象客户。后一种方法提供了另外一种途径，通过它可以被用于你的新的应用和旧系统相集成。**

**6、JSP(Java Server Pages):**

**Jsp页面由html代码和嵌入其中的Java新代码所组成。服务器在页面被客户端所请求以后对这些java代码进行处理，然后将生成的html页面返回给客户端的浏览器。**

**7、Java Servlet：**

**servlet是一种小型的java程序，它扩展了web服务器的功能。作为一种服务器端的应用，当被请求时开始执行，这和CGI Perl脚本很相似。Servlet提供的功能大多和jsp类似，不过实现方式不同。JSP通过大多数的html代码中嵌入少量的java代码，而servlet全部由java写成并生成相应的html。**

**8、XML（Extensible Markup Language）：**

**XML是一种可以用来定义其他标记语言的语言。它被用来在不同的商务过程中共享数据。XML的发展和Java是互相独立的，但是，它和java具有相同目标正是平台独立。通过java和xml的组合，我们可以得到一个完美的具有平台独立性的解决方案。**

**9、JMS（Java Message Service）：**

**Ms是用于和面向消息的中间件相互通信的应用程序接口（API）。它既支持点对点的域，有支持发布/订阅类型的域，并且提供对下列类型的支持：经认可的消息传递，事务性消息传递，一致性消息和具有持久性的订阅者的支持。JMS还提供了另一种方式对您的应用与旧的后台系统相集成。**

**10、JTA（Java Transaction Architecture）：**

**JTA定义了一种标准API，应用系统由此可以访问各种事务监控。**

**11、JTS（Java Transaction Service）:**

**JTS是CORBA OTS事务监控的基本实现。JTS规定了事务管理器的实现方式。该事务管理器是在高层支持Java Transaction API（JTA）规范，并且在较底层实现OMG OTS specification 的java映像。JTS事务管理器为应用服务器、资源管理器、独立的应用以及通信资源管理器提供了事务服务。**

**12、JavaMail：**

**JavaMail是用于存取邮件服务的API，它提供了一套邮件服务器的抽象类。不仅支持SMTP服务器，也支持IMAP服务器。**

**13、JAF（JavaBeans Activation Framework）：**

**JavaMail利用JAF来处理MIME编码的邮件附件。MIME的字节流可以被转换成java对象，或者转换自Java对象。大多数应用都可以不需要直接使用JAF**

**03: ajax的error**

**$.ajax({**

**url:"",**

**success: function (data) {**

**/\* 服务器无误 \*/**

**alert(data);**

**},**

**error: function (jqXHR, textStatus, errorThrown) {**

**/\*错误信息处理\*/**

**/\*弹出jqXHR对象的信息\*/**

**alert(jqXHR.responseText);**

**alert(jqXHR.status);**

**alert(jqXHR.readyState);**

**alert(jqXHR.statusText);**

**/\*弹出其他两个参数的信息\*/**

**alert(textStatus);**

**alert(errorThrown);**

**}**

**});**

**error函数中有三个参数：**

**第一个参数 jqXHR jqXHR：这里的jqXHR是一个jqXHR对象，在Jquery1.4和1.4版本之前返回的是XMLHttpRequest对象，1.5版本以后则开始使用jqXHR对象，该对象是一个超集，就是该对象不仅包括XMLHttpRequest对象，还包含其他更多的详细属性和信息。**

**这里主要有4个属性：**

**•readyState :当前状态,0-未初始化，1-正在载入，2-已经载入，3-数据进行交互，4-完成。**

**•status  ：返回的HTTP状态码，比如常见的404,500等错误代码。**

**•statusText ：对应状态码的错误信息，比如404错误信息是not found,500是Internal Server Error。**

**•responseText ：服务器响应返回的文本信息**

**第二个参数 String textStatus：返回的是字符串类型，表示返回的状态，根据服务器不同的错误可能返回下面这些信息："timeout"（超时）, "error"（错误）, "abort"(中止), "parsererror"（解析错误），还有可能返回空值。**

**第三个参数 String errorThrown：也是字符串类型，表示服务器抛出返回的错误信息，如果产生的是HTTP错误，那么返回的信息就是HTTP状态码对应的错误信息，比如404的Not Found,500错误的Internal Server Error。**

**04: eclipse和MyEclipse的区别**

**Eclipse 是一个IDE（Integrated Developing Environment），而这个IDE是允许安装第三方开发的插件来使自身的功能得到扩展和增强的，而Myeclipse就是其中的一种有名的插件集之一，MyEclipse将开发者常用到的一些有用的插件都集合起来，但它是收费的。**

**05: $.each()**

**Jquery遍历方法**

**$.each(循环体,function(index,element){**

**})**

**index:表示索引值**

**element:表示每一个被遍历的元素**

**06: 页面如何传值**

**1：通过链接传递值**

**<a href="demo.html?key=value&key2=value2">**

**2：通过表单传递值**

**以上接受都是使用request.getParameter("key");**

**在页面也可是使用四范围传递值**

**PageContext**

**跳转失效**

**HttpServletRequest**

**重定向失效，一般用于另一个页面一次使用**

**也就是一锤子买卖，不得使用重定向**

**A ---> B**

**HttpSession**

**通过session传递值，如果HttpSession销毁**

**则失效，不受跳转方式干扰，一般用户权限的控制，一般适用于多个页面使用**

**A ---> B ---> C**

**例如题目：10个页面都需要这个值放置在哪个范围中较好**

**答案为session**

**ServletContext**

**全文使用，整个工程中只有一份，类似全局变量，只要tomcat不关闭则不会失效，但是注意如果进行修改后续用户会受到影响，一般谨慎使用**

**以上四个范围通过setAttribute(String,Object)**

**传递值，通过getAttribute(String)接受值**

**07: 标签遍历用的是什么**

**jstl标签**

**<%**

**List<Boy> list = new ArrayList<Boy>();**

**list.add(new Boy("赵明轩",true,27));**

**list.add(new Boy("李民",false,21));**

**request.setAttribute("mylist",list);**

**%>**

**<%--**

**items:要被循环的循环体**

**var:表示要被遍历的集合的中每一个元素的key值，**

**注意这个key值由标签添加，范围默认就是pageContext**

**begin:起始索引值**

**end:结束索引值**

**step:步长**

**--%>**

**<c:forEach items="${requestScope.mylist}"**

**var="etoak" begin="2" end="7" step="3">**

**<tr>**

**<td><c:out value="${pageScope.etoak.name}"></c:out></td>**

**<td><c:out value="${etoak.married}"></c:out></td>**

**<td><c:out value="${etoak.age}"></c:out></td>**

**</tr>**

**</c:forEach>**

**08: JS和JQuery的区别**

**what？**

**<！-- 2016/7/3 -->**

**01：cookie和session有什么区别**

**Cookie会话跟踪机制：**

**由服务器创建，最终保存在本地，是浏览器提供的一种会话**

**跟踪机制，安全性较低，一般适用于网站的登录,权限识别等，**

**cookie默认不支持中文，直接通过构造方法new出来，存在**

**两个参数，都必须是String，cookie存在一个生命周期，**

**设置正数，则开始倒数，设置负数则仅仅存在浏览器缓存中**

**设置0则立刻销毁当前cookie，cookie存在实体，最终保存在**

**用户本地。通过浏览器可以找到具体的实体文件**

**HttpSession会话跟踪机制**

**由服务器创建，是web容器提供的一种安全性较cookie高的会话**

**跟踪机制，通过request.getSession()来拿取已经存在的**

**HttpSession或者创建一个新的HttpSession(),（在使用jsp时通过pageContext.getSession()创建session）session中**

**可以存储任意类型的数据，因为value值是Object数据类型（**

**支持中文）。**

**HttpSession存在一个最大不活动周期，默认是1800s，可以**

**修改，当创建后如果没有任何请求提交则开始倒计时，如果**

**存在请求提交，则时间重置，理论上HttpSession可以一直维持**

**当超过最大不活动周期HttpSession失效，里面的信息无法**

**拿取。HttpSession没有实体，无法找到具体文件**

**Cookie和HttpSession的关系**

**Cookie和HttpSession是两种会话跟踪机制，**

**一般使用的场合不同，但是Cookie的传递是通过**

**浏览器的支持将cookie从一个位置传输到另外一个位置，**

**注意：即使我们没有使用cookie机制，但只要使用了浏览器**

**cookie会话跟踪机制就开启了**

**HttpSession并不能自己从一个位置跳转到另外一个位置**

**完全是借着Cookie来进行传递，在cookie中使用**

**key：jsessionid value：32位随机码来进行传输**

**也就是说HttpSession搭了cookie一个顺风车**

**这也就是为什么当cookie禁用时HttpSession也失效的原因**

**如果Cookie被禁用那么HttpSession就无法使用了**

**吗？还可以使用重写URL的方式来进行维持**

**response.encodeURL(要被重新编码的路径)**

**可以通过路径来传输jsessionid**

**02：jquery中$.ajax中参数**

**$.ajax({**

**//要提交到目的地**

**url:"servlet/Page",**

**//1.2新特性，设置为false则不缓存此页面**

**cache:false,**

**//默认true，为异步请求，更改为false则变为同步请求，浏览器被锁住**

**async:true,**

**//异步请求的类型**

**type:"post",**

**//要传递的值 ,常使用serialize()方法序列化表单**

**data:"key="+value**

**+"&key2="+value2,**

**//返回的值的类型支持 json html text script 和 xml**

**dataType:"json",**

**success:function(data){**

**//服务器无误是回调函数**

**}，**

**error:function(){**

**//服务器出现异常时回调函数**

**}**

**03：用到的前台技术**

javascript ajax jquery EasyUI

**04：jquery怎么拿取另一个页面的数据**

**//文本**

**$("div#show").load("data/data.txt");**

**//html**

**$("div#show").load("data/data.html",function(){**

**alert("载入完毕！");**

**});**

**/\*仅仅适用于js文件\*/**

**$.getScript("data/data.js",function(){**

**alert("js载入完毕！");**

**});**

**05：页面展示数据 分页  如何展示出来 如何用table到话 前端该怎么写**

**分别设置三个函数**

               1：check()此函数用来负责数据的查询

            将当前页和每页记录数发送给服务器端由服务器端封装为json发回

             在本地解析遍历后封装进table中

               2：**changeCurrentPage(id)**设置根据id不同更改当前页的函数

               3：为页面上按钮绑定激发事件从而调用第二个函数对当前页进行修改

**function check(){**

**$.ajax({**

**//要提交到目的地**

**url:"servlet/Page",**

**//异步请求的类型**

**type:"post",**

**//要传递的值，注意getpost都这样传递值**

**data:"currentPage="+currentPage**

**+"&pageSize="+pageSize,**

**//返回的值的类型支持 json html text script 和 xml**

**dataType:"json",**

**success:function(data){**

**//alert(data);**

**totalCount = data.totalCount;**

**totalPage = data.totalPage;**

**// [{"id":1,"title":"冰岛队夺得欧洲杯冠军","pid":1},{"id":2,"title":"山东鲁能惨遭降级","pid":1},]**

**var list = data.list;**

**var table = "<table border='1' style='width:500px;'><tr><td>id</td><td>pid</td><td>标题</td></tr>";**

**/\***

**$.each(循环体,function(索引值,每一个对象){**

**});**

**\*/**

**$.each(list,function(i,news){**

**table +="<tr><td>"+news.id+"</td><td>"**

**+news.pid+"</td><td>"+news.title+"</td></tr>";**

**});**

**table += "</table>";**

**$("div#show").html(table);**

**$("label#msg").html("当前是第"+currentPage+"页,共有"**

**+totalPage+"页,共有"+totalCount+"条记录");**

**//调用一个函数，这个函数给按钮绑定了事件**

**addEventForButton();**

**},**

**error:function(err){**

**alert(err.status);**

**}**

**});**

**}**

**/\***

**根据按钮id的不同来确定**

**当前页的变化**

**\*/**

**function changeCurrentPage(id){**

**if(id=="first"){**

**currentPage=1;**

**}else if(id=="pre"){**

**currentPage--;**

**}else if(id=="next"){**

**currentPage++;**

**}else if(id=="end"){**

**currentPage=totalPage;**

**}**

**//修改当前页后再次进行查询**

**check();**

**}**

**function addEventForButton(){**

**//为已经绑定事件的元素解绑一切操作**

**$("button").unbind();**

**if(totalPage>1){**

**if(currentPage==1){**

**$("#next,#end").click(function(){**

**changeCurrentPage($(this).attr("id"));**

**});**

**}else if(currentPage==totalPage){**

**$("#first,#pre").click(function(){**

**changeCurrentPage($(this).attr("id"));**

**});**

**}else{**

**$("button").click(function(){**

**changeCurrentPage($(this).attr("id"));**

**});**

**}**

**}**

**}**

**<！-- 2016/7/1 -->**

**1: 在js中怎么创建一个数组**

    var arr = new Array();

    var arr1 = [];

    var arr2 = ['a',3,'b'];

    var arr3 = new Object[];

**2: 什么情况下使用ajax?**

    异步请求

    当不需要刷新整个页面 ; 需要进行局部刷新时

    数据传输效率更高 不需要HTML结构<html><head>

    JSON {key:value}

    同步请求

    XMLHttpRequest.open(1,2,false)

    $.ajax({async:false})

    发送请求之后,浏览器会被锁死,用户后续的操作需要等待响应的返回

**3: jquery循环数组**

    $.each( 数组, function(下标值 , obj){});

**4: 下拉列表的级联如何实现**

    <select> <select>

    页面加载时 , 发送ajax请求检索第一个下俩列表需要的数据集合

    将数据集合封装json返回给客户端 遍历添加到第一个下拉列表中

    为第一个下拉列表提供change(),当选择了某一个option选项触发第二次ajax请求,

    同时将选中的option的value值发送给服务器

    在遍历添加第二个下拉列表数据时,清除之前添加的option元素

    while(option.length>1)

        option.remove(option.attr[1]);

    $("").empty()

    $("option[value!=0]").remove()

**5: jquery的选择器(怎么给一个id为XXX的按钮加点击事件)?**

    基本选择器

        # . element :first :last :eq :lt ;gt

    层级选择器

        > " " + ~

    属性选择器

        attr[param]  attr[param=value]

        attr[param!=value]  attr[param^=value]

        attr[param$=value]

    表单选择器

        input submit password file text checkbox radio

        checked selected

    $("#XXX").click(function(){});

**6: document load 和 document ready的区别?**

    <div><img src='abc.img'/></div>

    load  当整个页面全部加载完毕之后触发

    ready  当整个页面的结构全部加载完毕之后触发

**7：jquery的四个核心函数是?**

    $(expression) 根据提供的表达式在页面中查找匹配的元素

        $("#tt")  $("div")

    $(html)  创建一个使用原始HTML标记字符串匹配的jquery对象

        html  表示页面中不存在的一组html元素

        需要将这组html追加到当前页面

    $("div")  $("<div>aaaaa</div>").appendTo(body);

    $(element)   $(document).ready()

            $(document.body.img)

    $(function(){}) == $(document).ready(function(){})

    都是为了创建一个jquery对象 - 使用这个jquery对象表示页面中的一组元素

**8: jsp常用指令有哪些**

    page include taglib

**9: 批量上传数据是怎么操作的**

    smartupload  commons-fileupload

    表单元素如何设置

        多个文件域  添加数组

        myfile[0] [1] [2]

    服务器如何解析

        struts2   File[] myfile

        spring-mvc   MultipartFile[] myfile;

**10: 对mvc的理解?**

    m model 模型  -  javaBean

    v view  视图 - JSP

    c controller 控制器 - servlet

    分工开发    解耦