HTML

非表单标签

1. b 粗体　u　下划线　　i 斜体　　del 删除效果
2. a　超链接　href,　target=_blank
3. img　图片
4. frameset(frame) 框架集
5. table 表格　th, tr, td（table data cell）　colspan, rowspan
6. ul　li　or　列表标签
7. embed 用来播放 MP3、视频等等
8. div 虚拟矩形区域　　span 行内元素(不会换行)

表单标签

1. <form>　表单标签　　action 提交表单的地址　　method 表单提交的方式(get,post)
   name（命名方式：regFrm）
2. 表单域(表单域一定要有 name 属性，否则该表单域的值不会被提交！)
   a. 文本域(text 单行文本域, password 密码框,　textarea 多行文本域)
   b. radio 单选按钮(组，name 属性一致即为一组，每一组只能提交一个值)
   c. checkbox 复选框(组，name 属性一致即为一组，每一组可以提交多个值)
   d. select,option 下拉框(可以实现多选一，多选多)
   e. file　文件域 用来实现文件上传(method=post，enctype=”multipart/form-data”)
   f. hidden　　隐藏域　(不需要用户输入，但是服务器又需要的值)
   g. reset　重置
   h. submit　提交按钮
   i. button　　普通按钮(通常用来激活一个 js 函数)

CSS

1. 如何引用 css 信息
   a) 通过 html 元素的：style 属性， class 属性
   b) 通过<style>块
   c) 通过引用外部的 css 文件　　<link>
2. css 选择器
   a) ID 选择器　　#myRed{color:red;}　　我是红色吗?
   b) class 选择器　.myRed{color:red;}　　我是红色吗?
   c) 元素选择器　　b{color:red; font-size:20px;}　　ddddd

3. css 属性
   a) 文本属性
   b) 定位属性(静态定位、绝对定位、相对定位、固定定位、z-index)
   c) 列表属性
   d) 布局属性
   e) 边框属性
   f) …

JAVASCRIPT

js 的基本语法

1. 基本数据类型：undefined，null，number，字符串、boolean，Object
2. 变量的定义：var
3. 控制语句
4. 内置语句：eval(), escape(),unescape()区别?
5. 常用的对象：Date, Math, String, RegExp，Array
6. 数组： var a = []; var b = new Array();
7. 对象的定义。
8. JSON：

```
var  user = {name:'高淇',pwd:'123456',sleep:function(){
       alert("zzzzz");
}};
user.sleep();
```

9. 函数也是对象！
10. 继承是通过 prototype 属性来模拟。 String.prototype.trim = function (){ return this.replace("(^\s+)||(\s+$)","")} " 324324".trim();
11. 对话框：alert 警告框，prompt 询问框，confirm 确认框
12. 浏览器对象
    a) window
       i. open 打开子窗口(google 搜索：子窗口如何操作父窗口。window.opener)
          close 关闭窗口
       ii. 定时操作：setTimeout, clearTimeout. setInterval, clearInterval
    b) navigator
       i. 浏览器信息。 navigator.userAgent.toLowerCase() , ie, firefox
    c) history
       i. 访问历史。 history.go(1)前进一个 history.go(-1)后退一个
    d) location
       i. 地址栏信息

dom 操作

1. 如何获得节点
   a) 直接获得
      i. document.getElementById()
      ii. document.getElementsByName() 返回的是数组
      iii. document.getElementsByTagName() 根据标签名字获取，返回的是数组
   b) 间接获得
      i. 通过父节点获得子节点
         1. childNodes
         2. firstChild, lastChild
      ii. 通过子节点获得父节点
         1. parentNode
      iii. 兄弟节点
         1. nextSibling, preSibling
2. 如何操作节点
   a) 判断节点类型：nodeType： 1 元素节点 3：文本节点
   b) 节点的属性： 通过点操作符即可。
   c) 节点的文本内容：innerHTML
3. 改变 DOM 结构
   a) 增加节点：appendChild, insertBefore
   b) 删除节点：removeChild
   c) 替换节点：replaceChild
4. 表单的操作
   a) 通用属性：value, disabled, 表单对象.elements
   b) 单选按钮和复选框操作的时候，一般采用遍历！
   c) 表单验证：
      i. <submit value=提交 onclick=”return check(this.form);”/>
      ii. <form onsubmit=”return check(this);”>

HTTP 协议

TOMCAT 服务器

1. server.xml
2. web.xml
3. 目录结构的作用：
   a) lib
   b) work（存放 jsp 编译后的 java 文件）
   c) webapps

d) conf

servlet

浏览器发送给服务器的请求内容。 HttpServletReqeust 对象.

getParameter() , getParameterValues()处理复选框

作为作用域：setAttribute, getAttribute, removeAttribute

getRequestURL, getRequestURI, getQueryString, getContextPath

获得客户端 IP 和端口：getRemoteAddr,getRemotePort

request.setCharacerEncoding 设置请求实体的编码。他对 post 有效，对 get 无效。要让对 get 方式提交的数据也有效，可以改动 tomcat 配置文件 server.xml,对 connection 元素增加一个属性：useBodyEncodingForURI=”true”

请求转发：request.getRequestDispatcher(“2.jsp”).forward(request,response);

1. **服务器回送给浏览器的响应内容。 HttpServletResponse 对象**
   setCharacterEncoding， setContentType(“text/html;charset=gbk”)
   getWriter 输出文本 getOutputStream 输出非文本

   重定向：response.sendRedirect(“http://www.baidu.com”);

servlet 类读取 web.xml 中自己的配置信息。 ServletConfig
getInitParameter(“ddd”)

2. **需要读取 web.xml 中的公共信息<context-param>。 ServletContext(项目中只有一个)**
   获得：this.getServletContext()
   作用域：可以被所有的 servlet 共享
   其他方法：getRealPath(“1.jsp”), 获得子目录信息 getResourcePaths

保存以前操作的状态。 客户端保存 Cookie 服务器保存 Session

客户端保存 cookie 的流程：

1. 服务器端写代码： Cookie c = new Cookie(“a’,”aaa”); c.setMaxAge(2000); response.setCookie(c);
2. 响应头中就会有一个：SetCookie：a=aaa
3. 浏览器收到以后，先放到内存中。如果 cookie 设置了有效期，则会写到硬盘的 cookie 文件中。
4. 浏览器以后访问服务器时，就会**自动携带(在请求头中)**有效的 cookie 信息。

服务器端如何保存信息(session 的跟踪机制)：

1. 服务器会 new 一个 session 对象。同时，指定一个 id 给 session 对象。
2. 通过响应头设置：setCookie：sessionId=23748923abcdd23342
3. 浏览器收到 sessionId 这个信息，下次访问一定会携带 sessionId
4. 服务器收到 sessionId 后，根据 sessionId 找到对应的 session 对象。然后，就可以查看对象里面的内容。从而，确定之前我们进行了什么操作。也就是，实现了状态的保存！

serlvet 生命周期：

1. 加载和实例化
   a) 第一个请求的时候。(Servlet 是单例，只有一个 servlet 对象)
   b) 如果配置了<load-on-startup>，启动时加载
2. 初始化
   a) 调用 Init
3. 执行
   a) service　　doGet　　doPost
4. 销毁
   a) destroy

过滤器：

1. 实现 Filter
2. doFilter
3. 配置位于<servlet>之前

监听器：

1. 实现 Listener
2. 配置位于<filter>之后，<servlet>之前。

jsp

1. <% %>　脚本段
2. <%= %>表达式
3. <%!　％＞声明
4. <%-- --%>jsp 注释
5. <%@　include file=”2.jsp” %>　静态导入。两个 jsp 翻译时 java 文件就合并到一起。
6. <jsp:include>　相当于一个类调用另一个类
7. jsp 九个内置对象分别是：
   request,response,out,session.application,page,pageContext,Exception,config

EL 和 JSTL

EL 表达式：操作的是作用域中的属性！
格式：${}

JSTL 标签库：

1. foreach
2. if

3. choose，when，otherwise
4. set
5. import

ajax

技术核心：通过 XHR(XMLHttpRequest)对象发送请求，通过 XHR 对象接收响应。再通过 js 操作 dom 对象，讲接收的新的信息，付给某个元素。从而实现局部刷新效果。

如何使用 XHR 对象：

1. 创建 XHR 对象

```
if(window.XMLHttpRequest){
    req=new XMLHttpRequest();
}else if(window.ActiveXObject){
    req=new ActiveXObject("Msxml2.XMLHTTP");
}
```

2. 创建一个请求

```
req.open("get","DemoServlet?username="+username.value+"&aaa="+new Date());
```

3. 设置监听，处理服务器返回的响应内容

```
req.onreadystatechange=function (){
        //alert(req.readyState);
        if(4==req.readyState){
            if(200==req.status){
                …
            }
        }
};
```

4. 发送请求

```
req.send(null);
```

XML

1. XML 基本语法
2. XML 的解析(DOM 方式、SAX 方式　　JDOM、DOM4J)
3. dtd/schema　定义 XML 数据的语法规则
4. xslt　　定义 xml 数据的显示方式