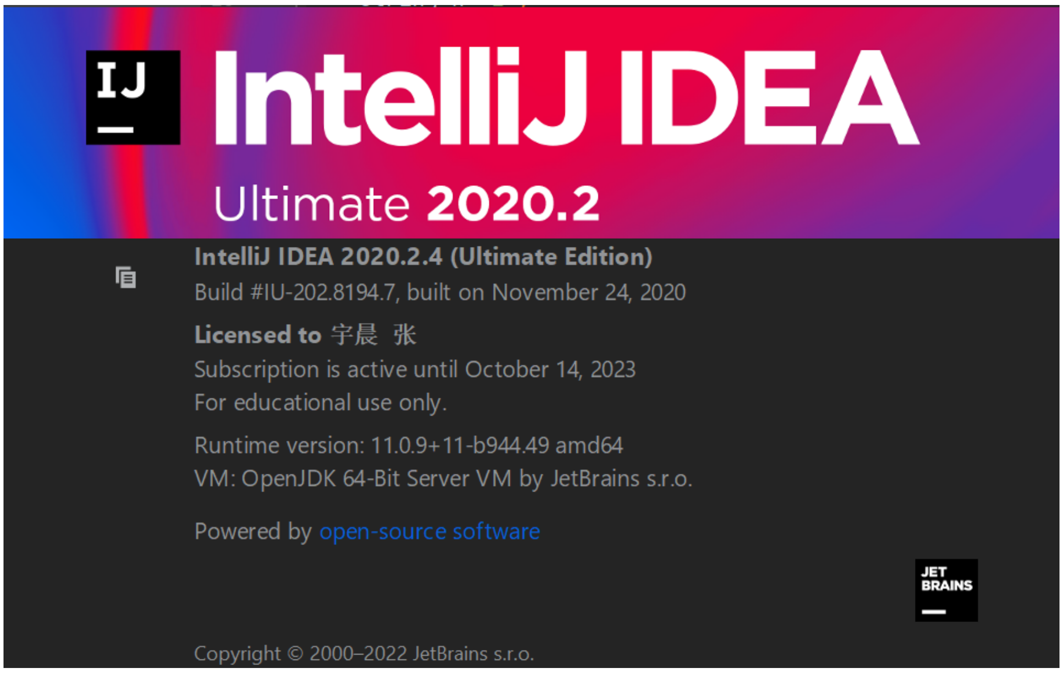
Java EE 第二次实习 Servlet技术

课程	姓名	学号	班级
Java EE	张宇晨	2020012249	软工2001班

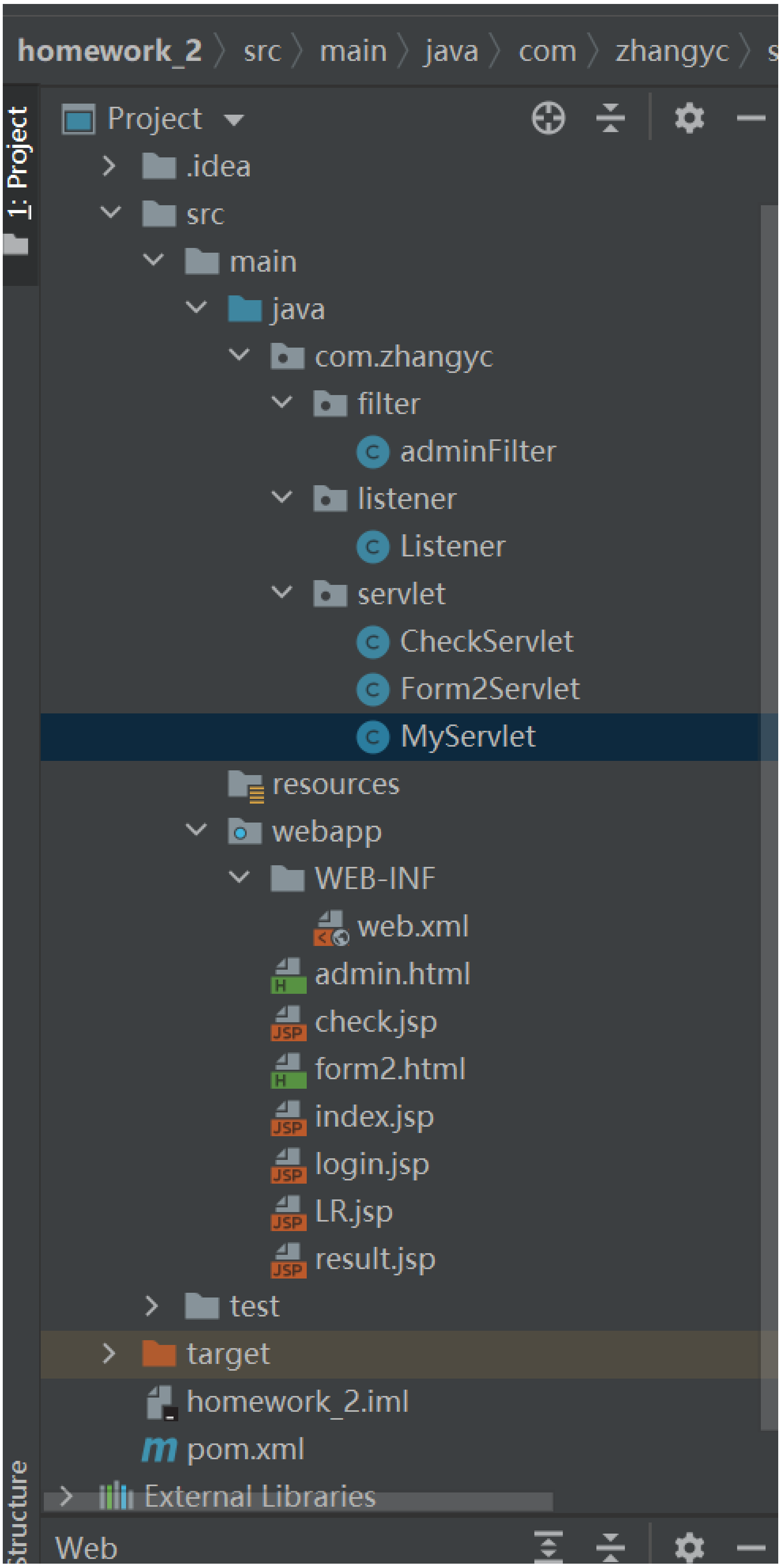
开发环境与工程结构

Intellij IDEA 2020.2



IntelliJ IDEA 2020.2.4 (Ultimate Edition)
Build #IU-202.8194.7, built on November 24, 2020
Licensed to 宇晨 张
Subscription is active until October 14, 2023
For educational use only.
Runtime version: 11.0.9+11-b944.49 amd64
VM: OpenJDK 64-Bit Server VM by JetBrains s.r.o.
Powered by open-source software
Copyright © 2000–2022 JetBrains s.r.o.

工程结构



```
.idea
src
  main
    java
      com.zhangyc
        filter
          adminFilter
        listener
          Listener
        servlet
          CheckServlet
          Form2Servlet
          MyServlet
    resources
    webapp
      WEB-INF
        web.xml
      admin.html
      check.jsp
      form2.html
      index.jsp
      login.jsp
      LR.jsp
      result.jsp
  test
target
homework_2.iml
pom.xml
External Libraries
```

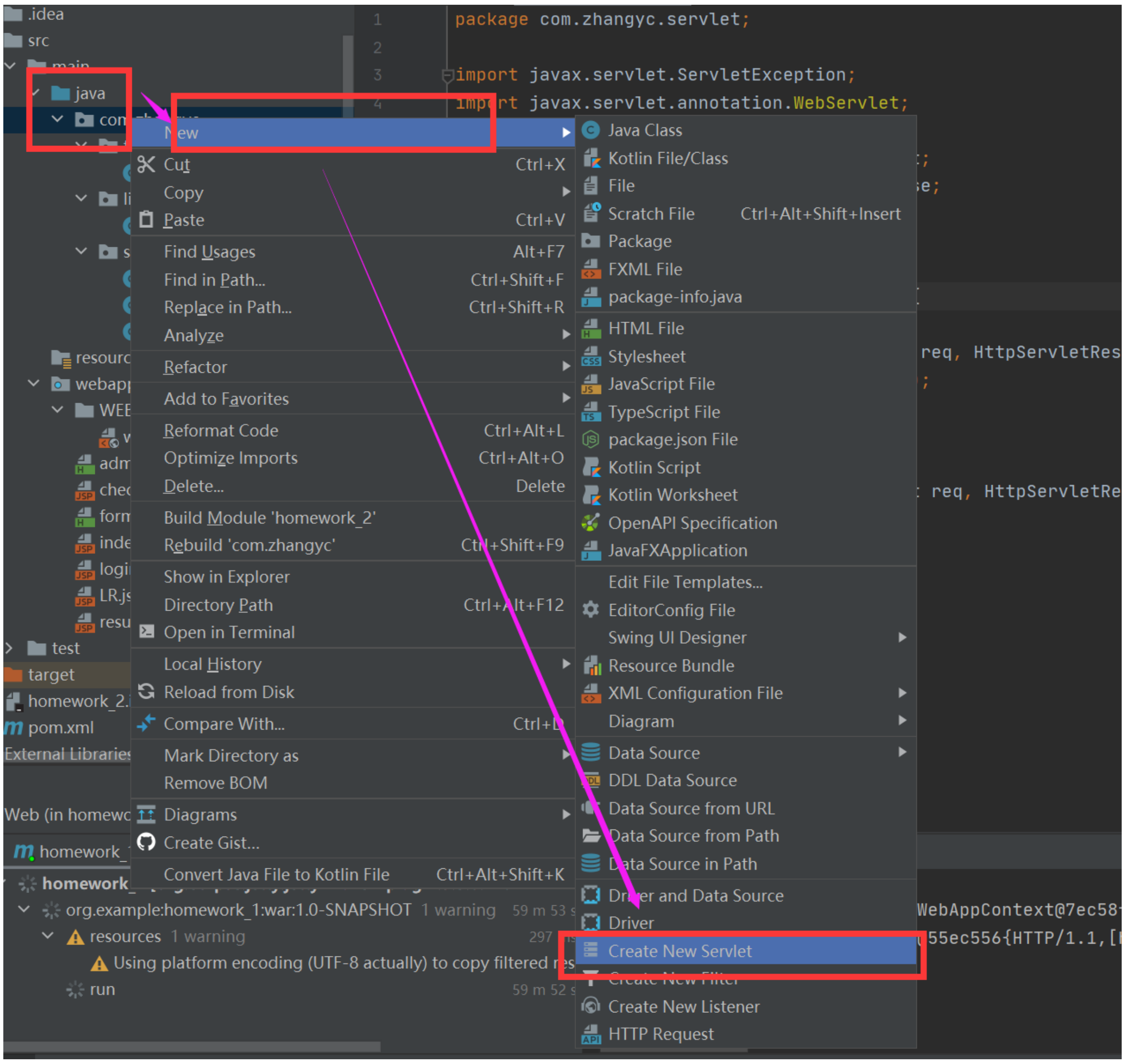
实现内容

1. Servlet

(1)第一个Servlet

a. 实习流程

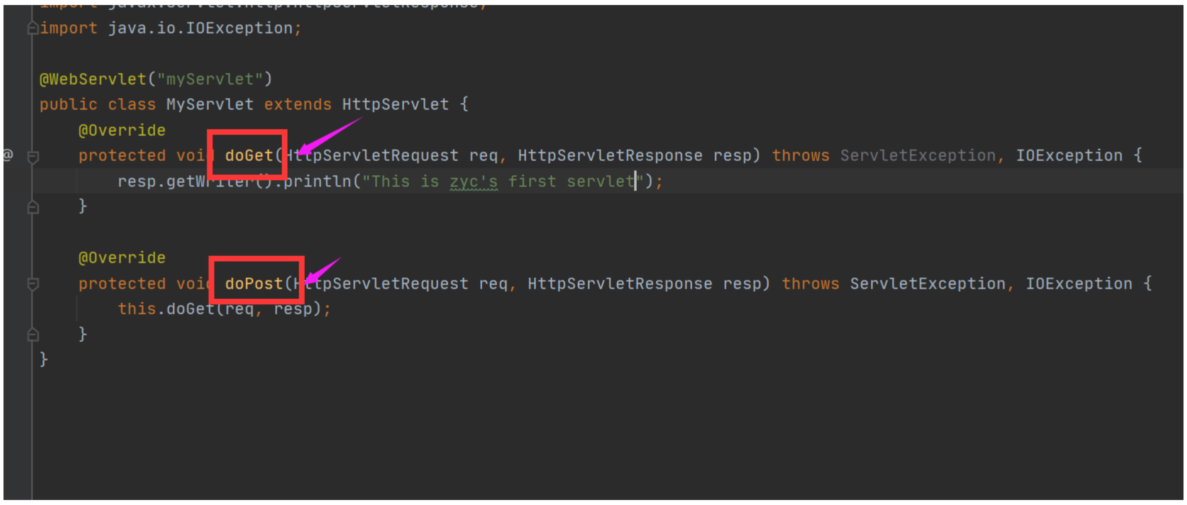
创建一个Servlet



```
package com.zhangyc.servlet;

import javax.servlet.ServletException;
import javax.servlet.annotation.WebServlet;
```

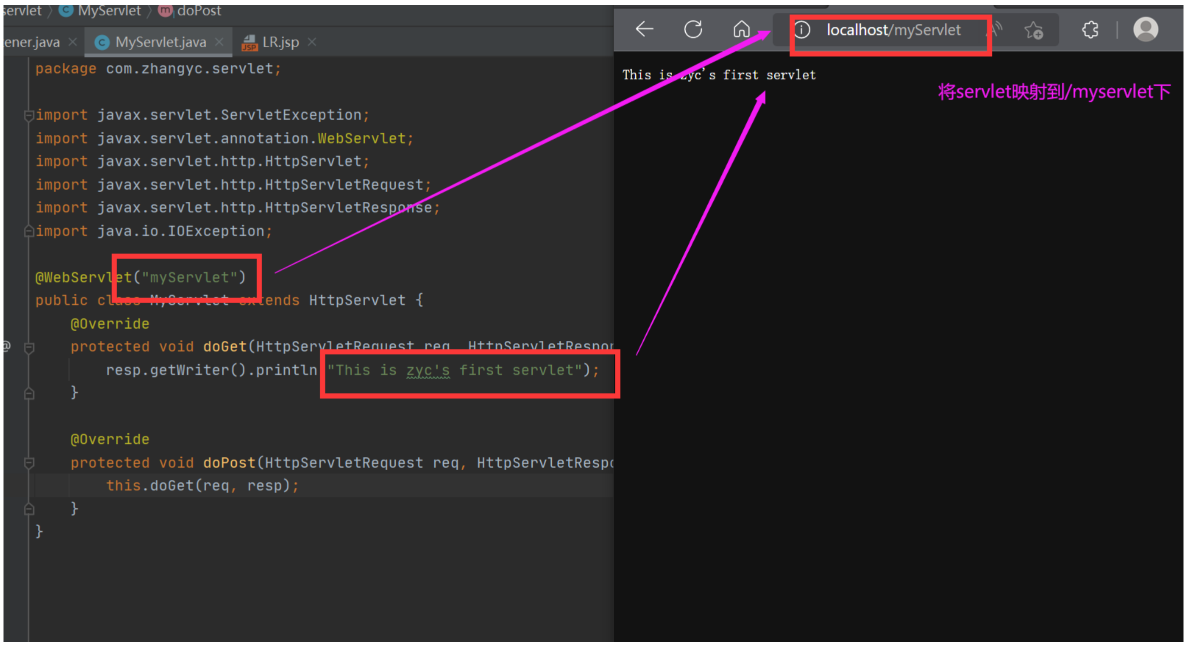
```
org.example:homework_1:war:1.0-SNAPSHOT  1 warning
  resources  1 warning
    Using platform encoding (UTF-8 actually) to copy filtered res
  run
```

重写doGet()和doPost()方法



```
import java.io.IOException;

@WebServlet("myServlet")
public class MyServlet extends HttpServlet {
    @Override
    protected void doGet(HttpServletRequest req, HttpServletResponse resp) throws ServletException, IOException {
        resp.getWriter().println("This is zyc's first servlet");
    }

    @Override
    protected void doPost(HttpServletRequest req, HttpServletResponse resp) throws ServletException, IOException {
        this.doGet(req, resp);
    }
}
```

使用注解方式映射servlet的路径


ervlet 〉 MyServlet 〉 doPost
ener.java × MyServlet.java × LR.jsp ×
localhost/myServlet
将servlet映射到/myservlet下

```
package com.zhangyc.servlet;

import javax.servlet.ServletException;
import javax.servlet.annotation.WebServlet;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import java.io.IOException;

@WebServlet("myServlet")
public class MyServlet extends HttpServlet {
    @Override
    protected void doGet(HttpServletRequest req, HttpServletRespon
        resp.getWriter().println("This is zyc's first servlet");
    }

    @Override
    protected void doPost(HttpServletRequest req, HttpServletRespo
        this.doGet(req, resp);
    }
}
```

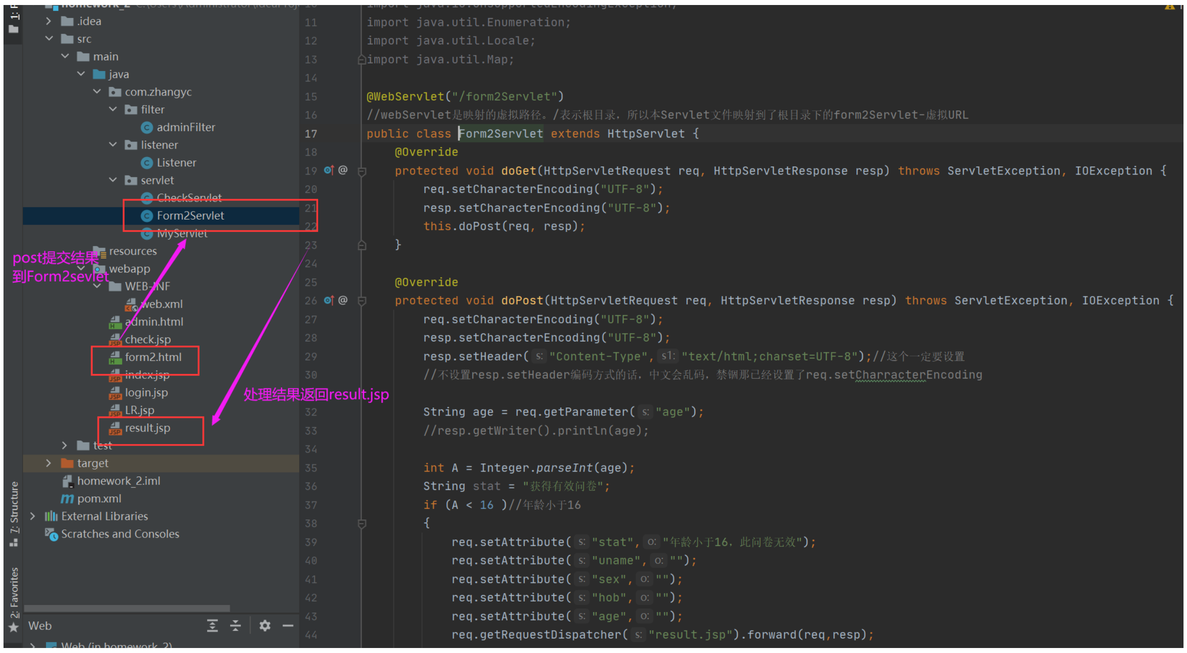
```
This is zyc's first servlet
```

b. 代码展示

```
package com.zhangyc.servlet;

import javax.servlet.ServletException;
import javax.servlet.annotation.WebServlet;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import java.io.IOException;

@WebServlet("myServlet")
public class MyServlet extends HttpServlet {
    @Override
    protected void doGet(HttpServletRequest req, HttpServletResponse resp)
throws ServletException, IOException {
        resp.getWriter().println("This is zyc's first servlet");
    }

    @Override
    protected void doPost(HttpServletRequest req, HttpServletResponse resp)
throws ServletException, IOException {
        this.doGet(req, resp);
    }
}
```

(2)表单处理

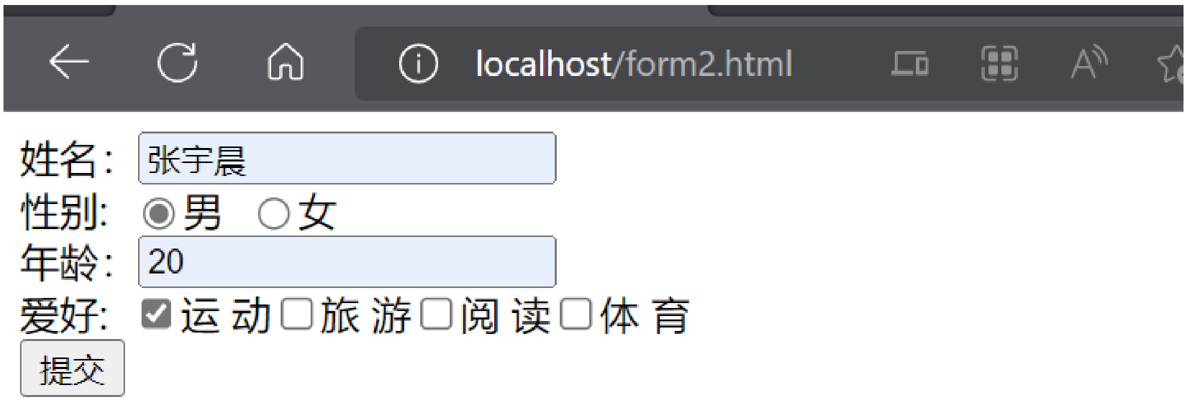
a. 实习流程



```
.idea
src
  main
    java
      com.zhangyc
        filter
          adminFilter
        listener
          Listener
        servlet
          CheckServlet
          Form2Servlet
          MyServlet
    resources
    webapp
      WEB-INF
        web.xml
      admin.html
      check.jsp
      form2.html
      index.jsp
      login.jsp
      LR.jsp
      result.jsp
  test
target
homework_2.iml
pom.xml
External Libraries
Scratches and Consoles
```

```java
import java.util.Enumeration;
import java.util.Locale;
import java.util.Map;

@WebServlet("/form2Servlet")
//webServlet是映射的虚拟路径。/表示根目录，所以本Servlet文件映射到了根目录下的form2Servlet-虚拟URL
public class Form2Servlet extends HttpServlet {
    @Override
    protected void doGet(HttpServletRequest req, HttpServletResponse resp) throws ServletException, IOException {
        req.setCharacterEncoding("UTF-8");
        resp.setCharacterEncoding("UTF-8");
        this.doPost(req, resp);
    }

    @Override
    protected void doPost(HttpServletRequest req, HttpServletResponse resp) throws ServletException, IOException {
        req.setCharacterEncoding("UTF-8");
        resp.setCharacterEncoding("UTF-8");
        resp.setHeader("Content-Type", "text/html;charset=UTF-8");//这个一定要设置
        //不设置resp.setHeader编码方式的话，中文会乱码，禁锢那已经设置了req.setCharracterEncoding

        String age = req.getParameter("age");
        //resp.getWriter().println(age);

        int A = Integer.parseInt(age);
        String stat = "获得有效问卷";
        if (A < 16 )//年龄小于16
        {
            req.setAttribute("stat", "年龄小于16，此问卷无效");
            req.setAttribute("uname", "");
            req.setAttribute("sex", "");
            req.setAttribute("hob", "");
            req.setAttribute("age", "");
            req.getRequestDispatcher("result.jsp").forward(req,resp);
```

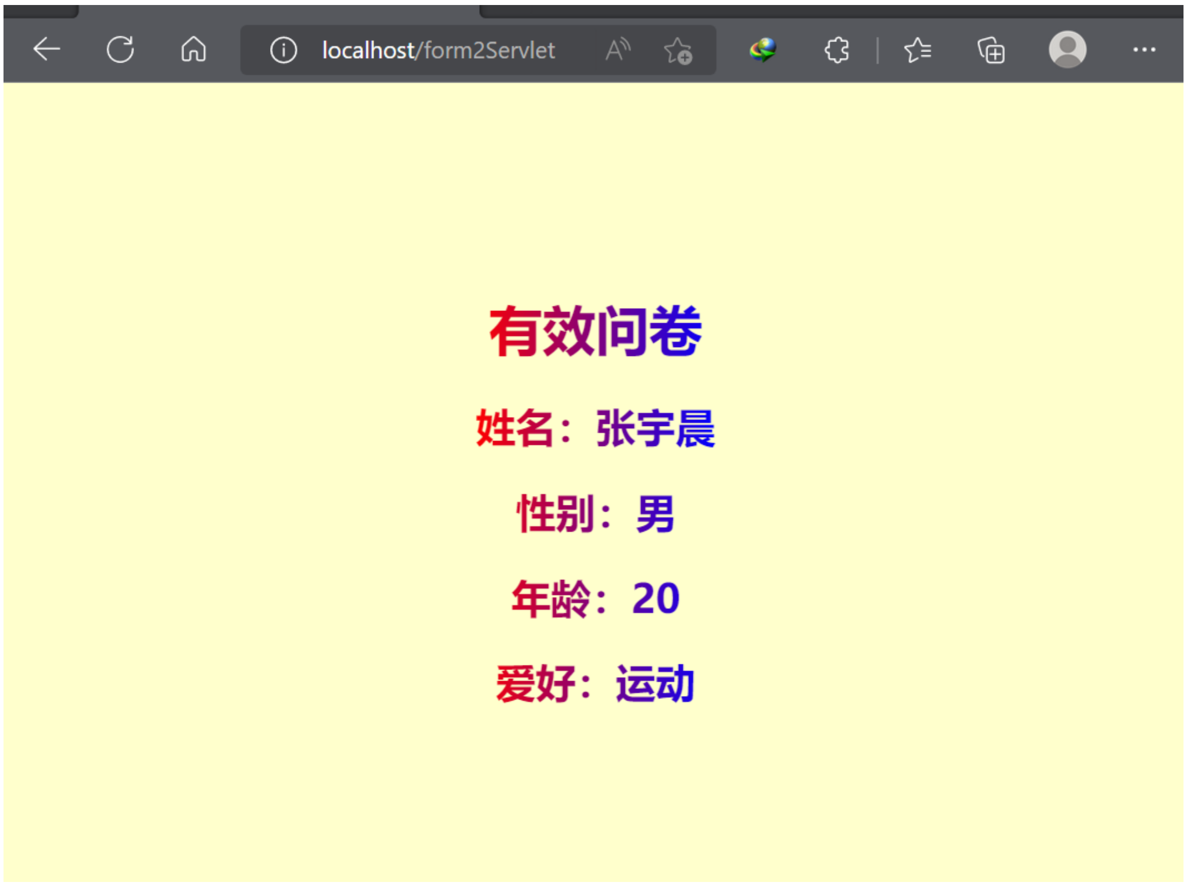
form2页面



姓名：	张宇晨
性别:	[✓ 男] 女
年龄：	20
爱好:	[✓ 运 动] 旅 游 阅 读 体 育

提交

处理结果


localhost/form2Servlet

有效问卷

姓名：张宇晨

性别：男

年龄：20

爱好：运动

b. 代码展示

Form2.html

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>Form_Test</title>
</head>

<body>
<form action="/form2Servlet" method="post" onsubmit="return check()">
    <div></div>
    <div class="labl_r" style="display: flex; flex-direction: row">
        <div>姓名：</div>
        <input type="text" name="name" >
    </div>
    <div class="labl_r" style="display: flex; flex-direction: row">
        <div>性别：&nbsp;</div>
        <input type="radio" name="sex" value="男">男&nbsp;&nbsp;
        <input type="radio" name="sex" value="女">女&nbsp;&nbsp;
    </div>
    <div class="labl_r" style="display: flex; flex-direction: row">
        <div>年龄：</div>
        <input type="text" name="age" id="idage">
    </div>
    <div class="labl_r" style="display: flex; flex-direction: row">
        <div>爱好：&nbsp;</div>
        <input type="checkbox" name="hobs" value="运动">运 动
        <input type="checkbox" name="hobs" value="旅游">旅 游
        <input type="checkbox" name="hobs" value="阅读">阅 读
        <input type="checkbox" name="hobs" value="体育">体 育
    </div>
    <button  type="submit" >提交</button>
    <!--        <input id="tbutton" type="button" onclick="cal()" value="提交">--
>
</form>
<script>
    console.log("666666");
    function check()
    {

        console.log("6666666");
        var age = document.getElementById("idage").value;
        console.log(age);
        if (age=="" || age == null)
        {
            alert("请填写年龄");
            return false;
        }
        return true;
    };
</script>
```

Form2Servlet

```
package com.zhangyc.servlet;

import javax.servlet.*;
import javax.servlet.annotation.WebServlet;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import java.io.BufferedReader;
import java.io.IOException;
import java.io.UnsupportedEncodingException;
import java.util.Enumeration;
import java.util.Locale;
import java.util.Map;

@WebServlet("/form2Servlet")
//webServlet是映射的虚拟路径。/表示根目录，所以本Servlet文件映射到了根目录下的
form2Servlet-虚拟URL
public class Form2Servlet extends HttpServlet {
    @Override
    protected void doGet(HttpServletRequest req, HttpServletResponse resp)
throws ServletException, IOException {
        req.setCharacterEncoding("UTF-8");
        resp.setCharacterEncoding("UTF-8");
        this.doPost(req, resp);
    }

    @Override
    protected void doPost(HttpServletRequest req, HttpServletResponse resp)
throws ServletException, IOException {
        req.setCharacterEncoding("UTF-8");
        resp.setCharacterEncoding("UTF-8");
        resp.setHeader("Content-Type","text/html;charset=UTF-8");//这个一定要设置
        //不设置resp.setHeader编码方式的话，中文会乱码，禁锢那已经设置了
req.setCharracterEncoding

        String age = req.getParameter("age");
        //resp.getWriter().println(age);

        int A = Integer.parseInt(age);
        String stat = "获得有效问卷";
        if (A < 16 )//年龄小于16
        {
            req.setAttribute("stat","年龄小于16，此问卷无效");
            req.setAttribute("uname","");
            req.setAttribute("sex","");
            req.setAttribute("hob","");
            req.setAttribute("age","");
            req.getRequestDispatcher("result.jsp").forward(req,resp);
            return;
        }
        String name = req.getParameter("name");//获取表单中的用户名
        String sex = req.getParameter("sex");//获取表单中的性别
        String[] hobs = req.getParameterValues("hobs");
        String hob = "";
        if (hobs == null) {
            hob = "该用户没有选择爱好";

        } else {
            for (int i = 0; i < hobs.length; ++i) {
                hob += hobs[i];
            }
        }
        if (name == null || name == "") {
            name = "该用户没有填写姓名";
        }
        if (sex == null || sex == "") {
            sex = "该用户没有填写性别";
        }
        //resp.getWriter().println(name);
        req.setAttribute("uname",name);
        req.setAttribute("sex",sex);
        req.setAttribute("hob",hob);
        req.setAttribute("stat","有效问卷");
        req.setAttribute("age",age);
        req.getRequestDispatcher("result.jsp").forward(req,resp);
        return;
    }
}
```

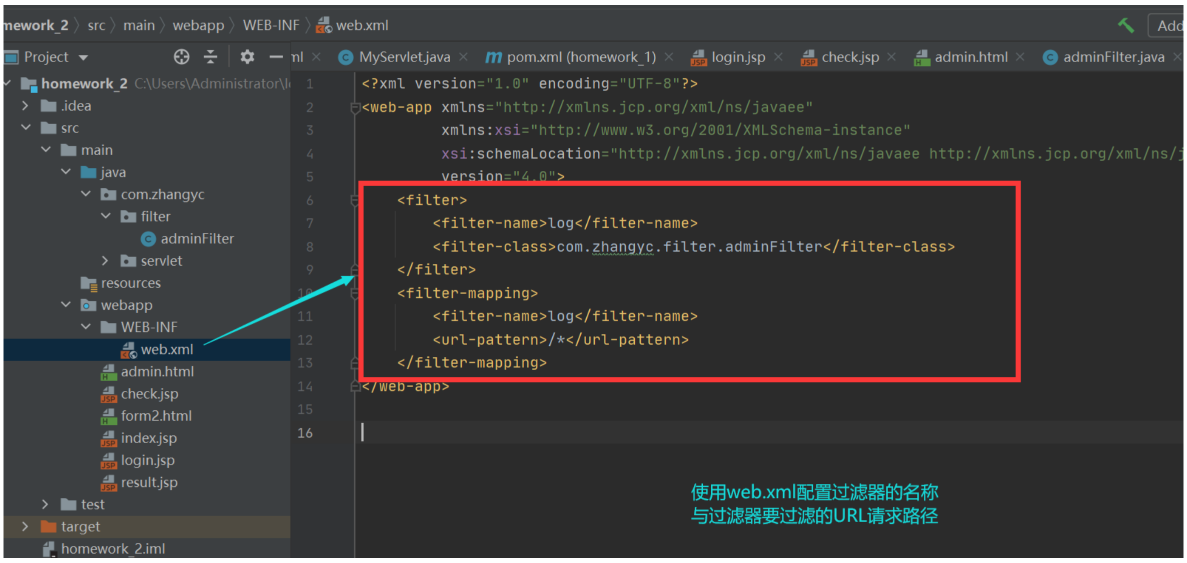
result.jsp

```
<%@ page language="java" contentType="text/html; charset=UTF-8"
pageEncoding="UTF-8"%>
<html>
<head>
    <meta http-equiv="Content-Type" content="text/html; charset=utf-8" />
    <title>The first Java EE practice</title>

    <style>
```

```
        .colortxt{

            background: linear-gradient(to right, red, blue);
            -webkit-background-clip: text;
            color: transparent;

        }
        body{
            background: #ffffcc;
        }
    </style>
</head>
<body>
<%
    request.setCharacterEncoding("UTF-8");
    response.setCharacterEncoding("UTF-8");
    String stat = request.getAttribute("stat").toString();
    String name = request.getAttribute("uname").toString();
    String sex = request.getAttribute("sex").toString();
    String age = request.getAttribute("age").toString();
    String hob = request.getAttribute("hob").toString();

%>
<div style="
    position: absolute;
    left: 50%;top: 50%;
    transform: translate(-50%,-50%); text-align: center;">
    <h1 class="colortxt"><%=stat%></h1>
    <h2 class="colortxt">姓名：<%=name%></h2>
    <h2 class="colortxt">性别：<%=sex%></h2>
    <h2 class="colortxt">年龄：<%=age%></h2>
    <h2 class="colortxt">爱好：<%=hob%></h2>


</div>

</body>
</html>
```

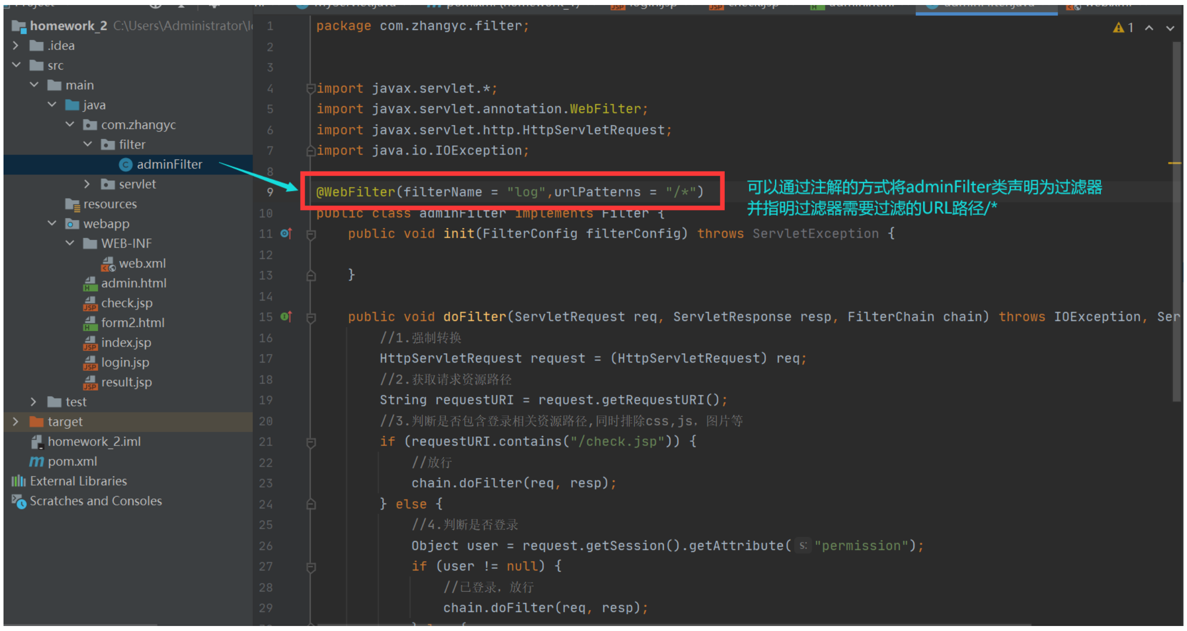
2. 过滤器

（1）使用过滤器重写实习一中的2.3（登录验证）



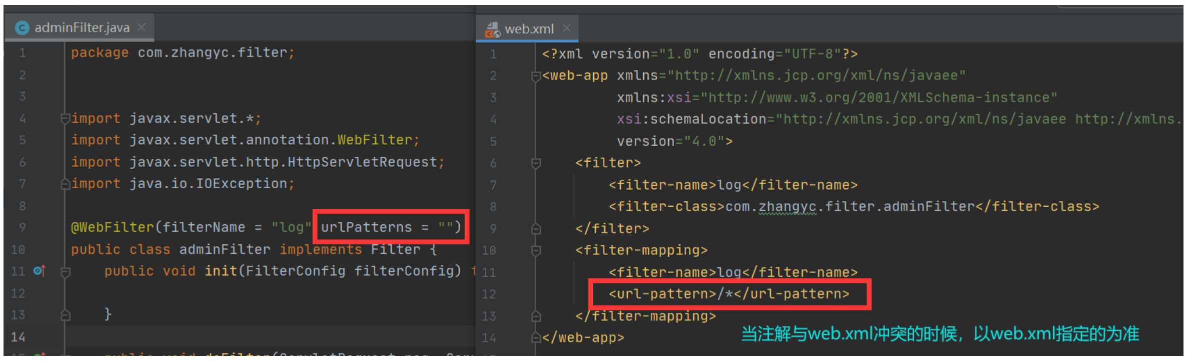
```xml
<?xml version="1.0" encoding="UTF-8"?>
<web-app xmlns="http://xmlns.jcp.org/xml/ns/javaee"
         xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
         xsi:schemaLocation="http://xmlns.jcp.org/xml/ns/javaee http://xmlns.jcp.org/xml/ns/j
         version="4.0">
    <filter>
        <filter-name>log</filter-name>
        <filter-class>com.zhangyc.filter.adminFilter</filter-class>
    </filter>
    <filter-mapping>
        <filter-name>log</filter-name>
        <url-pattern>/*</url-pattern>
    </filter-mapping>
</web-app>
```

使用web.xml配置过滤器的名称
与过滤器要过滤的URL请求路径



```java
package com.zhangyc.filter;


import javax.servlet.*;
import javax.servlet.annotation.WebFilter;
import javax.servlet.http.HttpServletRequest;
import java.io.IOException;

@WebFilter(filterName = "log",urlPatterns = "/*")
public class adminFilter implements Filter {
    public void init(FilterConfig filterConfig) throws ServletException {

    }

    public void doFilter(ServletRequest req, ServletResponse resp, FilterChain chain) throws IOException, Ser
        //1.强制转换
        HttpServletRequest request = (HttpServletRequest) req;
        //2.获取请求资源路径
        String requestURI = request.getRequestURI();
        //3.判断是否包含登录相关资源路径,同时排除css,js，图片等
        if (requestURI.contains("/check.jsp")) {
            //放行
            chain.doFilter(req, resp);
        } else {
            //4.判断是否登录
            Object user = request.getSession().getAttribute( s: "permission");
            if (user != null) {
                //已登录，放行
                chain.doFilter(req, resp);
```

可以通过注解的方式将adminFilter类声明为过滤器
并指明过滤器需要过滤的URL路径/*



adminFilter.java

```java
package com.zhangyc.filter;


import javax.servlet.*;
import javax.servlet.annotation.WebFilter;
import javax.servlet.http.HttpServletRequest;
import java.io.IOException;

@WebFilter(filterName = "log" urlPatterns = "")
public class adminFilter implements Filter {
    public void init(FilterConfig filterConfig) t

    }
```

web.xml

```xml
<?xml version="1.0" encoding="UTF-8"?>
<web-app xmlns="http://xmlns.jcp.org/xml/ns/javaee"
         xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
         xsi:schemaLocation="http://xmlns.jcp.org/xml/ns/javaee http://xmlns.
         version="4.0">
    <filter>
        <filter-name>log</filter-name>
        <filter-class>com.zhangyc.filter.adminFilter</filter-class>
    </filter>
    <filter-mapping>
        <filter-name>log</filter-name>
        <url-pattern>/*</url-pattern>
    </filter-mapping>
</web-app>
```

当注解与web.xml冲突的时候，以web.xml指定的为准

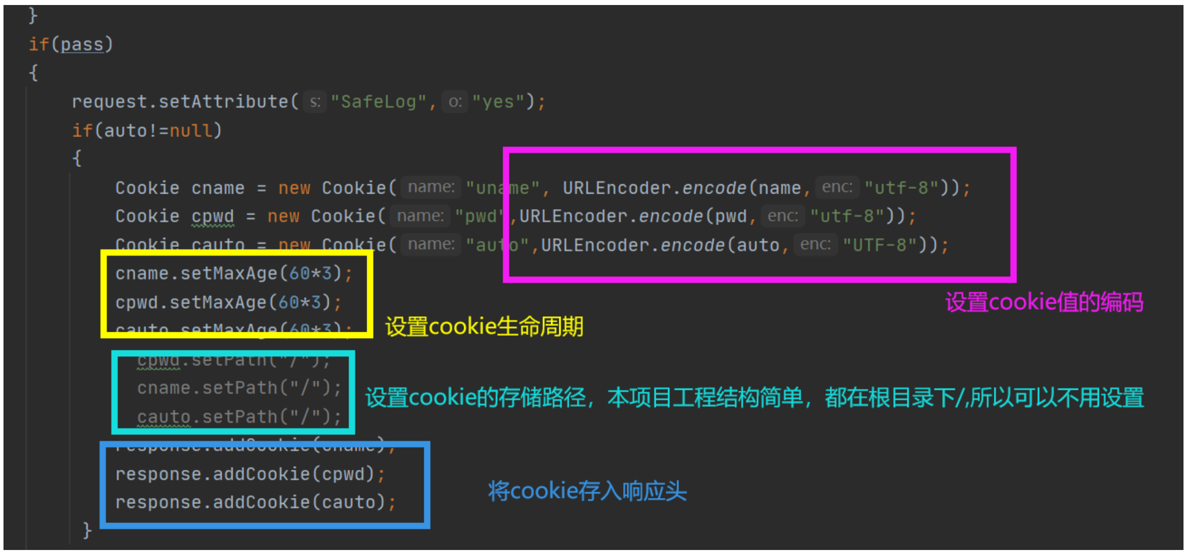
实际上，一般不会将过滤路径声明为/*,这样会对本应用下所有的页面访问过滤而产生异常。在本实习中要对admin.html管理员页面进行过滤保护，则只需要设置为/admin.html即可。

当需要对一组页面进行过滤的时候，也可以通过其他形式，如下

```
具体资源路径拦截:    "/index.jsp"     //这是指访问index.jsp的时候会经过过滤器

具体目录拦截:       "/user/*"         //这是指访问user目录下的所有资源时会经过过滤器

具体后缀名拦截:      "*.jsp"          //这时指访问后缀名为.jsp的资源时会经过过滤器

所有资源拦截:       "/*"             //这是指访问所有资源的时候都会经过过滤器
```

（2）使用Cookie为（1）中的登录验证实现自动登录功能。（在登录页面添加“自动登录”的复选框，若选择下次访问该应用时使用cookie存储的用户名和口令完成字典登录）

设置cookie



```
}
if(pass)
{
    request.setAttribute( s: "SafeLog", o: "yes");
    if(auto!=null)
    {
        Cookie cname = new Cookie( name: "uname", URLEncoder.encode(name, enc: "utf-8"));
        Cookie cpwd = new Cookie( name: "pwd",URLEncoder.encode(pwd, enc: "utf-8"));
        Cookie cauto = new Cookie( name: "auto",URLEncoder.encode(auto, enc: "UTF-8"));
        cname.setMaxAge(60*3);
        cpwd.setMaxAge(60*3);
        cauto.setMaxAge(60*3);
          cpwd.setPath("/");
          cname.setPath("/");
          cauto.setPath("/");
        response.addCookie(cname);
        response.addCookie(cpwd);
        response.addCookie(cauto);
    }
```

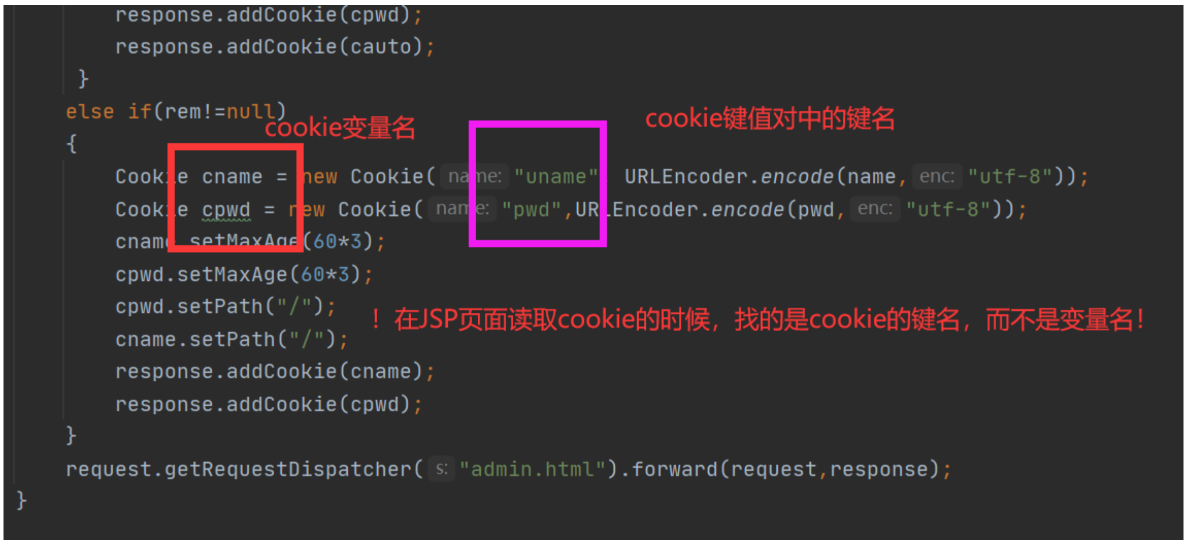
设置cookie值的编码

设置cookie生命周期

设置cookie的存储路径，本项目工程结构简单，都在根目录下/,所以可以不用设置

将cookie存入响应头

读取cookie



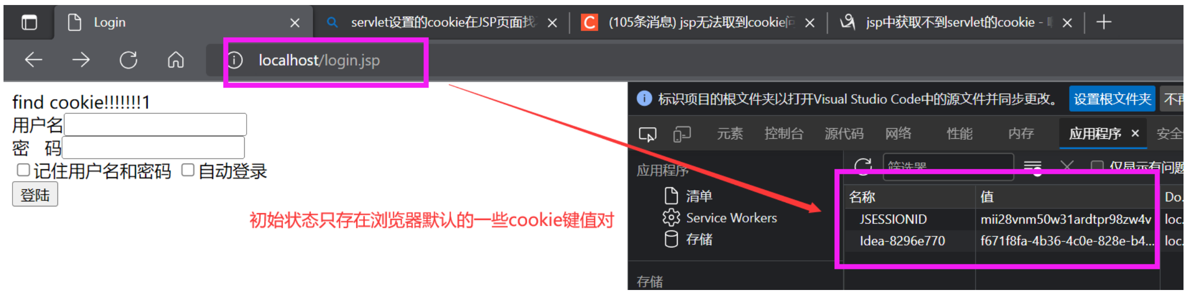
```
        response.addCookie(cpwd);
        response.addCookie(cauto);
    }
    else if(rem!=null)
    {
        Cookie cname = new Cookie( name: "uname", URLEncoder.encode(name, enc: "utf-8"));
        Cookie cpwd = new Cookie( name: "pwd",URLEncoder.encode(pwd, enc: "utf-8"));
        cname.setMaxAge(60*3);
        cpwd.setMaxAge(60*3);
        cpwd.setPath("/");
        cname.setPath("/");
        response.addCookie(cname);
        response.addCookie(cpwd);
    }
    request.getRequestDispatcher( s: "admin.html").forward(request,response);
}
```

cookie变量名

cookie键值对中的键名

！在JSP页面读取cookie的时候，找的是cookie的键名，而不是变量名！

结果测试：



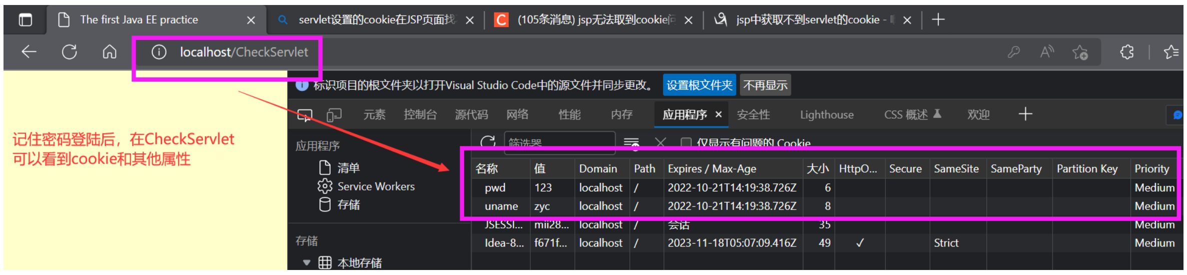
Login
servlet设置的cookie在JSP页面找
(105条消息) jsp无法取到cookie
jsp中获取不到servlet的cookie -
localhost/login.jsp

find cookie!!!!!!!1
用户名
密　码
☐记住用户名和密码 ☐自动登录
登陆

初始状态只存在浏览器默认的一些cookie键值对

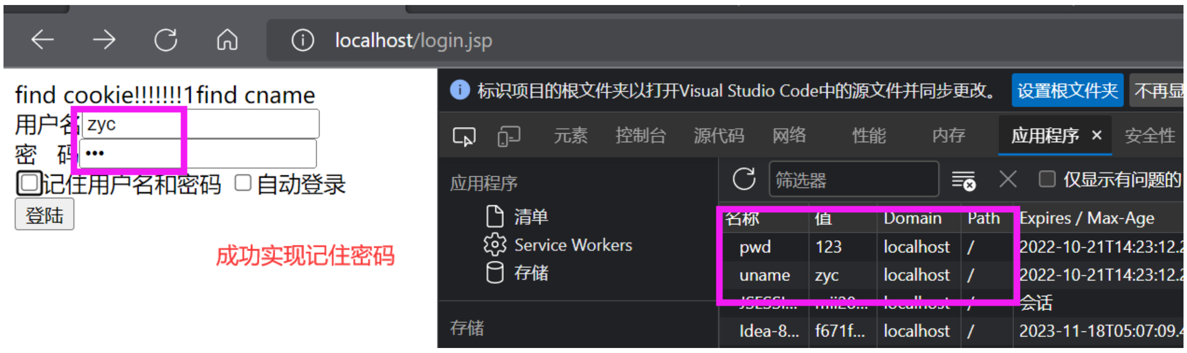
标识项目的根文件夹以打开Visual Studio Code中的源文件并同步更改。
设置根文件夹
元素
控制台
源代码
网络
性能
内存
应用程序
应用程序
清单
Service Workers
存储
存储

名称	值	Do
JSESSIONID	mii28vnm50w31ardtpr98zw4v	loc
Idea-8296e770	f671f8fa-4b36-4c0e-828e-b4...	loc


localhost/CheckServlet

记住密码登陆后，在CheckServlet
可以看到cookie和其他属性

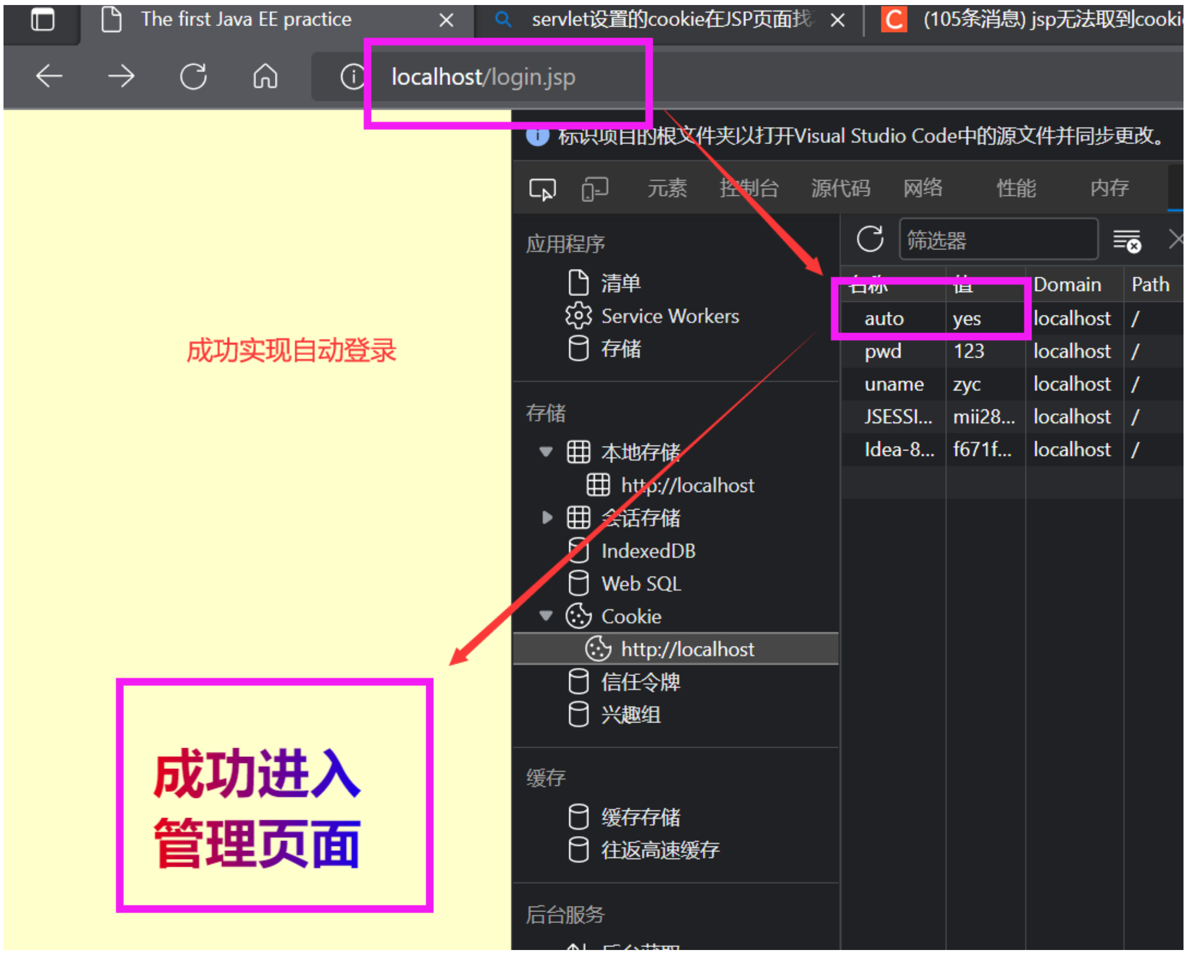
名称	值	Domain	Path	Expires / Max-Age	大小	HttpO...	Secure	SameSite	SameParty	Partition Key	Priority
pwd	123	localhost	/	2022-10-21T14:19:38.726Z	6						Medium
uname	zyc	localhost	/	2022-10-21T14:19:38.726Z	8						Medium
JSESSI...	mii28...	localhost	/	会话	35						Medium
Idea-8...	f671f...	localhost	/	2023-11-18T05:07:09.416Z	49	✓		Strict			Medium


localhost/login.jsp

find cookie!!!!!!!1find cname
用户名 zyc
密　码 •••
☐记住用户名和密码 ☐自动登录
登陆

成功实现记住密码

名称	值	Domain	Path	Expires / Max-Age
pwd	123	localhost	/	2022-10-21T14:23:12.2
uname	zyc	localhost	/	2022-10-21T14:23:12.2
JSESSI...	mii28...	localhost	/	会话
Idea-8...	f671f...	localhost	/	2023-11-18T05:07:09.4


localhost/login.jsp

成功实现自动登录

名称	值	Domain	Path
auto	yes	localhost	/
pwd	123	localhost	/
uname	zyc	localhost	/
JSESSI...	mii28...	localhost	/
Idea-8...	f671f...	localhost	/

成功进入
管理页面

代码展示

login.jsp

```
<%--
  Created by IntelliJ IDEA.
  User: Administrator
  Date: 2022/10/15
  Time: 19:55
  To change this template use File | Settings | File Templates.
--%>
```

```
<%@ page contentType="text/html;charset=UTF-8" language="java" %>
<html>
<head>
    <title>Login</title>
</head>
<body>
<%
    Cookie[] cookies=request.getCookies();
    String n="";
    String p="";
    if(cookies != null)
    {
//        out.print("find cookie!!!!!!!1");
        for (int i = 0; i < cookies.length; ++i)
        {
            if (cookies[i].getName().equals("uname"))
            {
                n = cookies[i].getValue();
//                  out.print("find cname");
            }
            if (cookies[i].getName().equals("pwd"))
            {
                p = cookies[i].getValue();
            }
            if (cookies[i].getName().equals("auto"))
            {
                request.setAttribute("LogSafe","yes");

 request.getRequestDispatcher("/admin.html").forward(request,response);
            }
        }
    }

%>

<form action="/CheckServlet" method="post">
    <div style="display: flex; flex-direction: row">
        用户名
        <input type="text" name="uname" value="<%=n%>">
    </div>
    <div style="display: flex; flex-direction: row">
        密 &nbsp;&nbsp;码
        <input type="password" name="pwd" value="<%=p%>">
    </div>
    <input type="checkbox" name="rem" value="yes">记住用户名和密码
    <input type="checkbox" name="auto" value="yes">自动登录
    <div></div>
    <div><button type="submit">登陆</button></div>
</form>
<%
    if(request.getAttribute("success")!=null)
    {
%>
<%=request.getAttribute("success")%>
<%}%>
</body>
</html>
```

CheckServlet

```
package com.zhangyc.servlet;

import javax.servlet.ServletException;
import javax.servlet.ServletRequest;
import javax.servlet.annotation.WebServlet;
import javax.servlet.http.*;
import java.io.IOException;
import java.net.URLEncoder;

@WebServlet("/CheckServlet")
public class CheckServlet extends HttpServlet {
    protected void doPost(HttpServletRequest request, HttpServletResponse
response) throws ServletException, IOException {
        doGet(request,response);
    }

    protected void doGet(HttpServletRequest request, HttpServletResponse
response) throws ServletException, IOException {
        //ServletRequest req = (HttpServletRequest) request;
        request.setCharacterEncoding("UTF-8");
        response.setCharacterEncoding("UTF-8");
        //HttpSession session = request.getSession();
        String name = request.getParameter("uname");
        String pwd = request.getParameter("pwd");
        String rem = request.getParameter("rem");
        String auto = request.getParameter("auto");
        boolean pass = false;
        if(name.equals("zyc") && pwd.equals("123")) pass=true;
        else
        {
            request.setAttribute("success","用户名或密码错误");
            request.getRequestDispatcher("login.jsp").forward(request,response);
        }
        if(pass)
        {
            request.setAttribute("SafeLog","yes");
            if(auto!=null)
            {
                Cookie cname = new Cookie("uname", URLEncoder.encode(name,"utf-
8"));
                Cookie cpwd = new Cookie("pwd",URLEncoder.encode(pwd,"utf-8"));
                Cookie cauto = new Cookie("auto",URLEncoder.encode(auto,"UTF-
8"));
                cname.setMaxAge(60*3);
                cpwd.setMaxAge(60*3);
                cauto.setMaxAge(60*3);
//                cpwd.setPath("/");
//                cname.setPath("/");
//                cauto.setPath("/");
                response.addCookie(cname);
                response.addCookie(cpwd);
                response.addCookie(cauto);
             }
            else if(rem!=null)
            {
                Cookie cname = new Cookie("uname", URLEncoder.encode(name,"utf-
8"));
                Cookie cpwd = new Cookie("pwd",URLEncoder.encode(pwd,"utf-8"));
                cname.setMaxAge(60*3);
                cpwd.setMaxAge(60*3);
//                cpwd.setPath("/");
//                cname.setPath("/");设置cookie的存储路径，在本项目中可设可不设
                response.addCookie(cname);
                response.addCookie(cpwd);
            }

 request.getRequestDispatcher("admin.html").forward(request,response);
        }

    }
}
```

adminFilter

```
package com.zhangyc.filter;


import javax.servlet.*;
import javax.servlet.annotation.WebFilter;
import javax.servlet.http.HttpServletRequest;
import java.io.IOException;

//@WebFilter(filterName = "log",urlPatterns = "")
public class adminFilter implements Filter {
    public void init(FilterConfig filterConfig) throws ServletException {

    }

    public void doFilter(ServletRequest req, ServletResponse resp, FilterChain
chain) throws IOException, ServletException {
        //1.强制转换
        HttpServletRequest request = (HttpServletRequest) req;
        //2.获取请求资源路径
        String requestURI = request.getRequestURI();
        //3.判断是否包含check.jsp页面
        //if (requestURI.contains("/check.jsp") )
        if (requestURI.contains("/CheckServlet") )
        {
            //放行
            chain.doFilter(req, resp);
        }
        else
        {
            //4.判断是否登录
            Object permission = request.getAttribute("LogSafe");
            if (permission != null)
            {
                //已登录，放行
                chain.doFilter(req, resp);
            }else
            {
                //未登录，跳转登陆页面
                request.setAttribute("login_msg","您未登录");

 request.getRequestDispatcher("/login.jsp").forward(request,resp);
            }
        }

    }

    public void destroy() {

    }
}
```

admin.html

```
<html>
<head>
    <meta http-equiv="Content-Type" content="text/html; charset=utf-8" />
    <title>The first Java EE practice</title>

    <style>
        .colortxt{

            background: linear-gradient(to right, red, blue);
            -webkit-background-clip: text;
            color: transparent;

        }
        body{
            background: #ffffcc;
        }
    </style>
</head>
<body>

<div style="
    position: absolute;
    left: 50%;top: 50%;
    transform: translate(-50%,-50%); text-align: center;">
    <h1 class="colortxt">成功进入管理页面</h1>
</div>

</body>
</html>
```

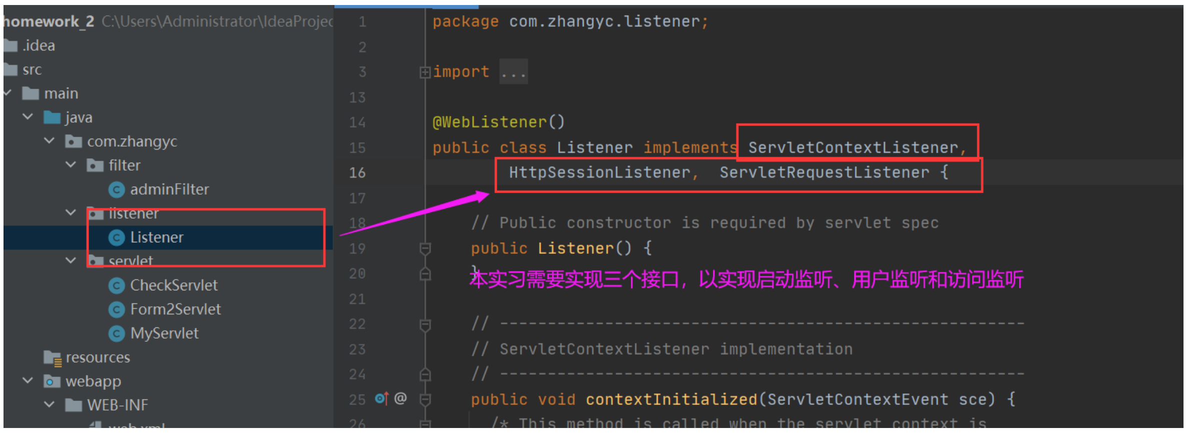
web.xml部分

```
<?xml version="1.0" encoding="UTF-8"?>
<web-app xmlns="http://xmlns.jcp.org/xml/ns/javaee"
         xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
         xsi:schemaLocation="http://xmlns.jcp.org/xml/ns/javaee
http://xmlns.jcp.org/xml/ns/javaee/web-app_4_0.xsd"
         version="4.0">
    <filter>
        <filter-name>log</filter-name>
        <filter-class>com.zhangyc.filter.adminFilter</filter-class>
    </filter>
    <filter-mapping>
        <filter-name>log</filter-name>
        <url-pattern>/admin.html</url-pattern>
    </filter-mapping>
</web-app>
```

3.Servlet生命周期事件

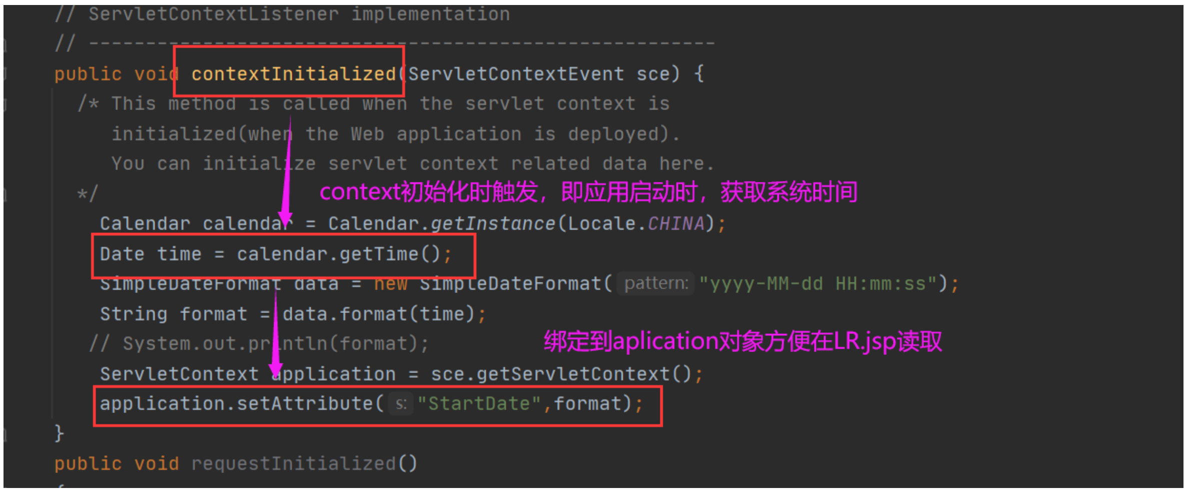
使用生命周期事件统计当前应用的在线人数、启动时间、请求数。

a. 实习流程



```
package com.zhangyc.listener;

import ...

@WebListener()
public class Listener implements ServletContextListener,
        HttpSessionListener,  ServletRequestListener {

    // Public constructor is required by servlet spec
    public Listener() {
    }

    // -------------------------------------------------------
    // ServletContextListener implementation
    // -------------------------------------------------------
    public void contextInitialized(ServletContextEvent sce) {
      /* This method is called when the servlet context is
```

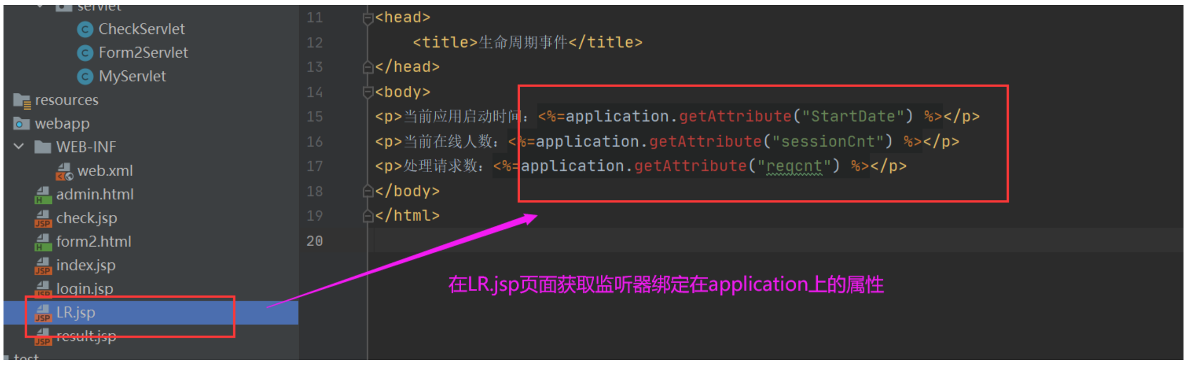
抽象函数实现（其他抽象函数的实现类似，这里只给出了Contex的监听函数解释）



```
    // ServletContextListener implementation
    // -------------------------------------------------------
    public void contextInitialized(ServletContextEvent sce) {
      /* This method is called when the servlet context is
         initialized(when the Web application is deployed).
         You can initialize servlet context related data here.
      */
        Calendar calendar = Calendar.getInstance(Locale.CHINA);
        Date time = calendar.getTime();
        SimpleDateFormat data = new SimpleDateFormat( pattern: "yyyy-MM-dd HH:mm:ss");
        String format = data.format(time);
       // System.out.println(format);
        ServletContext application = sce.getServletContext();
        application.setAttribute( s: "StartDate",format);
    }
    public void requestInitialized()
```

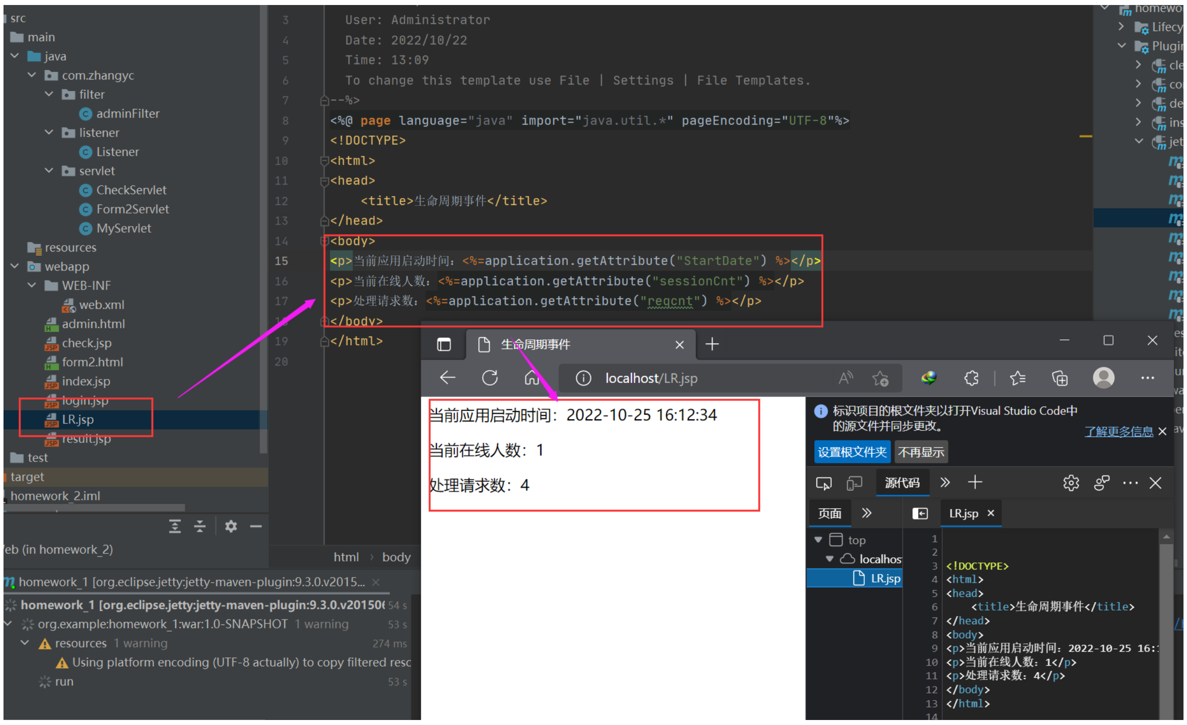
context初始化时触发，即应用启动时，获取系统时间
绑定到application对象方便在LR.jsp读取

LR.jsp页面



```
<head>
    <title>生命周期事件</title>
</head>
<body>
<p>当前应用启动时间：<%=application.getAttribute("StartDate") %></p>
<p>当前在线人数：<%=application.getAttribute("sessionCnt") %></p>
<p>处理请求数：<%=application.getAttribute("reqcnt") %></p>
</body>
</html>
```

测试



```
  User: Administrator
  Date: 2022/10/22
  Time: 13:09
  To change this template use File | Settings | File Templates.
--%>
<%@ page language="java" import="java.util.*" pageEncoding="UTF-8"%>
<!DOCTYPE>
<html>
<head>
    <title>生命周期事件</title>
</head>
<body>
<p>当前应用启动时间：<%=application.getAttribute("StartDate") %></p>
<p>当前在线人数：<%=application.getAttribute("sessionCnt") %></p>
<p>处理请求数：<%=application.getAttribute("reqcnt") %></p>
</body>
</html>
```

```
当前应用启动时间：2022-10-25 16:12:34

当前在线人数：1

处理请求数：4
```

```
<!DOCTYPE>
<html>
<head>
    <title>生命周期事件</title>
</head>
<body>
<p>当前应用启动时间：2022-10-25 16:1
<p>当前在线人数：1</p>
<p>处理请求数：4</p>
</body>
</html>
```

```
homework_1 [org.eclipse.jetty:jetty-maven-plugin:9.3.0.v2015...
homework_1 [org.eclipse.jetty:jetty-maven-plugin:9.3.0.v201506  54 s
  org.example:homework_1:war:1.0-SNAPSHOT  1 warning  53 s
    resources  1 warning  274 ms
      Using platform encoding (UTF-8 actually) to copy filtered resc
  run  53 s
```

b. 代码展示

Listener

```java
package com.zhangyc.listener;

import javax.servlet.*;
import javax.servlet.annotation.WebListener;
import javax.servlet.http.HttpSessionAttributeListener;
import javax.servlet.http.HttpSessionEvent;
import javax.servlet.http.HttpSessionListener;
import javax.servlet.http.HttpSessionBindingEvent;
import java.text.SimpleDateFormat;
import java.util.Calendar;
import java.util.Date;
import java.util.Locale;

@WebListener()
public class Listener implements ServletContextListener,
        HttpSessionListener, HttpSessionAttributeListener,
ServletRequestListener {

    // Public constructor is required by servlet spec
    public Listener() {
    }

    // -------------------------------------------------------
    // ServletContextListener implementation
    // -------------------------------------------------------
    public void contextInitialized(ServletContextEvent sce) {
      /* This method is called when the servlet context is
         initialized(when the Web application is deployed).
         You can initialize servlet context related data here.
      */
        Calendar calendar = Calendar.getInstance(Locale.CHINA);
        Date time = calendar.getTime();
        SimpleDateFormat data = new SimpleDateFormat("yyyy-MM-dd HH:mm:ss");
        String format = data.format(time);
       // System.out.println(format);
        ServletContext application = sce.getServletContext();
        application.setAttribute("StartDate",format);
    }
    public void requestInitialized()
    {

    }
    public void contextDestroyed(ServletContextEvent sce) {
      /* This method is invoked when the Servlet Context
         (the Web application) is undeployed or
         Application Server shuts down.
      */
    }

    // -------------------------------------------------------
    // HttpSessionListener implementation
    // -------------------------------------------------------
    public void sessionCreated(HttpSessionEvent se) {
        /* Session is created. */
        ServletContext application = se.getSession().getServletContext();
        Object cnt = application.getAttribute("sessionCnt");
        if(cnt==null)
        {
            application.setAttribute("sessionCnt","1");
        }
        else {
            int a = Integer.parseInt(cnt.toString());
            a++;
            application.setAttribute("sessionCnt",String.valueOf(a));
        }
    }

    public void sessionDestroyed(HttpSessionEvent se) {
        /* Session is destroyed. */
        ServletContext application = se.getSession().getServletContext();
        Object cnt = application.getAttribute("sessionCnt");
        int a = Integer.parseInt(cnt.toString());
        a--;
        application.setAttribute("sessionCnt",String.valueOf(a));
    }

    // -------------------------------------------------------
    // HttpSessionAttributeListener implementation
    // -------------------------------------------------------

    public void attributeAdded(HttpSessionBindingEvent sbe) {
      /* This method is called when an attribute
         is added to a session.
      */
    }

    public void attributeRemoved(HttpSessionBindingEvent sbe) {
      /* This method is called when an attribute
         is removed from a session.
      */
    }
    public void attributeReplaced(HttpSessionBindingEvent sbe) {
      /* This method is invoked when an attribute
         is replaced in a session.
      */
    }

    public void requestDestroyed(ServletRequestEvent sre) {

    }

    public void requestInitialized(ServletRequestEvent sre) {
        ServletContext application = sre.getServletContext();
        Object cnt = application.getAttribute("reqcnt");
        if(cnt==null)
        {
            application.setAttribute("reqcnt","1");
        }
        else {
            int a = Integer.parseInt(cnt.toString());
            a++;
            application.setAttribute("reqcnt",String.valueOf(a));
        }

    }
}
```

LR.jsp

```
<%--
  Created by IntelliJ IDEA.
  User: Administrator
  Date: 2022/10/22
  Time: 13:09
  To change this template use File | Settings | File Templates.
--%>
<%@ page language="java" import="java.util.*" pageEncoding="UTF-8"%>
<!DOCTYPE>
<html>
<head>
    <title>生命周期事件</title>
</head>
<body>
<p>当前应用启动时间：<%=application.getAttribute("StartDate") %></p>
<p>当前在线人数：<%=application.getAttribute("sessionCnt") %></p>
<p>处理请求数：<%=application.getAttribute("reqcnt") %></p>
</body>
</html>
```

总结

问:

```
@WebServlet 注解 和 web.xml 的优缺点
```

答:

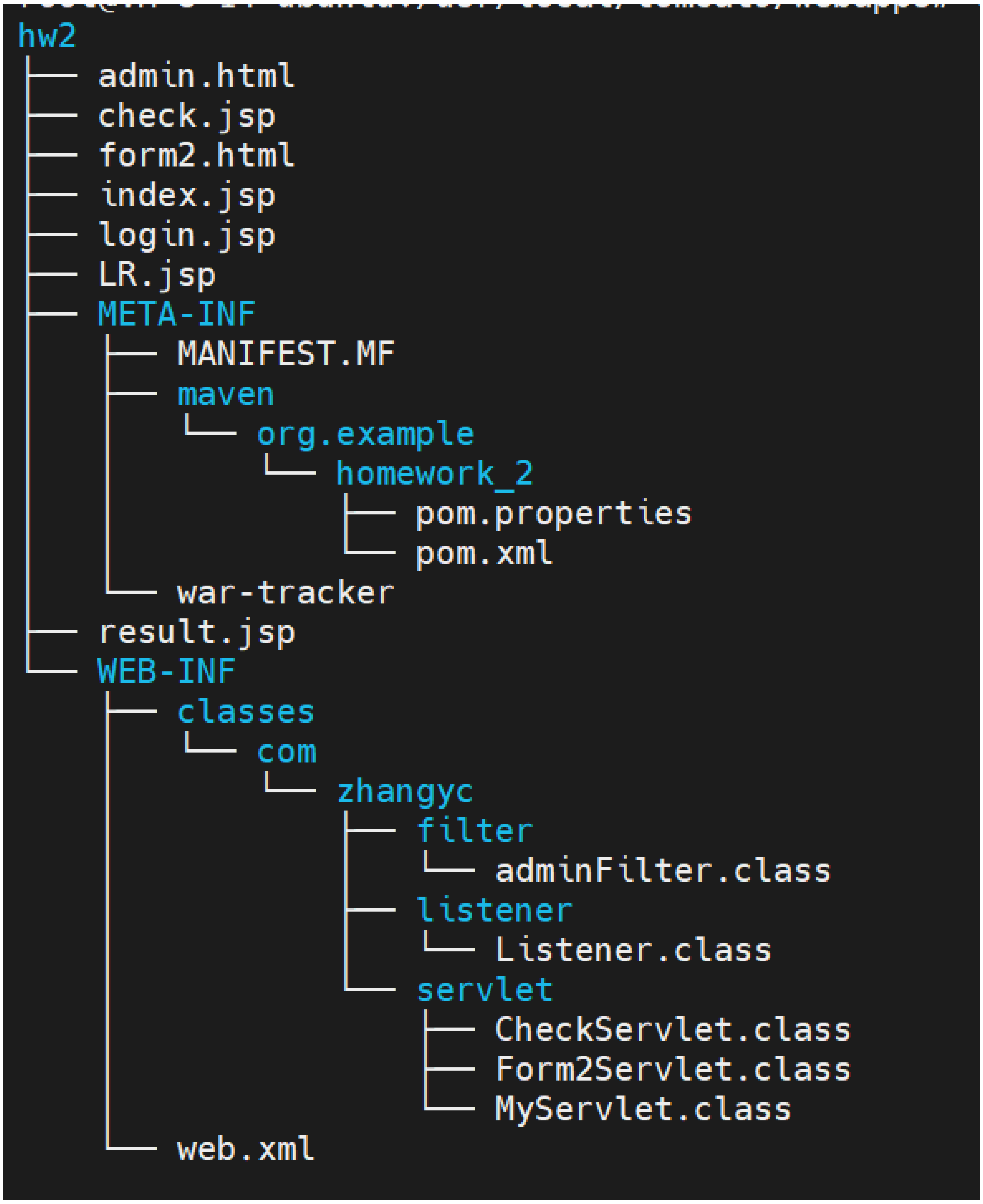
```
使用 web.xml 或 @WebServlet 注解都可以配置 Servlet， 两者各有优缺点。

@WebServlet 注解配置 Servlet：
优点：@WebServlet 直接在 Servlet 类中使用，代码量少，配置简单。每个类只关注自身业务逻辑，与其他 Servlet 类互不干扰，适合多人同时开发。
缺点：Servlet 较多时，每个 Servlet 的配置分布在各自的类中，不便于查找和修改。

web.xml 配置文件配置 Servlet：
优点：集中管理 Servlet 的配置，便于查找和修改。
缺点：代码较繁琐，可读性不强，不易于理解。
```

补充

项目已部署在个人网站，欢迎老师访问评阅

（http://www.zhangyuchen.cn/javaEE_wk/）点击访问

war包结构如下



```
hw2
├── admin.html
├── check.jsp
├── form2.html
├── index.jsp
├── login.jsp
├── LR.jsp
├── META-INF
│   ├── MANIFEST.MF
│   ├── maven
│   │   └── org.example
│   │       └── homework_2
│   │           ├── pom.properties
│   │           └── pom.xml
│   └── war-tracker
├── result.jsp
└── WEB-INF
    ├── classes
    │   └── com
    │       └── zhangyc
    │           ├── filter
    │           │   └── adminFilter.class
    │           ├── listener
    │           │   └── Listener.class
    │           └── servlet
    │               ├── CheckServlet.class
    │               ├── Form2Servlet.class
    │               └── MyServlet.class
    └── web.xml
```