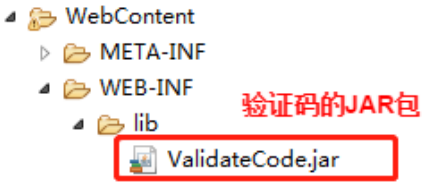
生成验证码技术

实现重点方法：

1、html

```
<div class="col-sm-3">
      <img src="/WebSerTest/codeServlet"/>
</div>
 <div class="col-sm-3">
                                映射路径
   <img src="/WebSerTest/codeServlet"/>
            项目名
 </div>
```

2、web项目导入验证码的jar包



3、核心代码实现

```
//参数1：图片宽度  参数2：图片高度  参数3：字符个数  参数4：干扰线条数
ValidateCode vc = new ValidateCode(150, 30, 4, 9);
//通过响应字节流，把图片响应到浏览器
vc.write(response.getOutputStream());
//获取验证码的数据
String code = vc.getCode();
```

```
//参数1：图片宽度 参数2：图片高度 参数3：字符个数 参数4：干扰线条数
ValidateCode vc = new ValidateCode(150, 30, 4, 9);
//通过响应字节流，把图片响应到浏览器
vc.write(response.getOutputStream());
//获取验证码的数据
String code = vc.getCode();
```

添加点击图片或文字，更换验证码

1、html:

```
<!-- 显示验证码的由codeServlet类处理    给标签一个ID 一个点击的方法
     方法参数指向本IMG标签 -->
    <div class="col-sm-3">
```

```
      <img id="img" src="/WebSerTest/codeServlet" onclick="changeImg(this)"/>
    </div>

<!-- 点击事件的方法 -->
 <script type="text/javascript">
     function changeImg(img){
         img.src = "/WebSerTest/codeServlet?time="+new Date().getTime();
     }
 </script>
```

2、处理类的方法不变

操作请求行相关的方法

```
request.getMethod();    // POST|GET 获得请求方式
request.getRemoteAddr();    // 获得客户机的IP地址
request.getRequestURL();    // 获得请求路径 url
```

http://localhost:8080/day12_03_register/regServlet

```
request.getRequestURI();     // 获得URI  /day12_03_register/regServlet
request.getContextPath());  // 获得项目名称  /day12_03_register

String[] hobbys = request.getParameterValues("hobby");  //为复选框准备的方法返回数组
```

将请求的数据封装到javabean对象中的封装框架

使用工具类:简化数据的封装.BeanUtils是Apache的工具类.

* 使用BeanUtils:
    * commons-beanutils-1.8.3.jar
    * commons-logging-1.1.1.jar

接收表单中的所有的参数. request.getParameterMap(); Servlet自带的方法

使用方法:

1、创建JavaBean对象（bean对象中的成员变量名要与Html表单中的name名保持一致）

```
User user = new User();
```

2、通过工具将请求表单的数据封装到javabean对象中去

```
//注意：不要导错包 org.apache.commons.beanutils.BeanUtils
BeanUtils.populate(user, request.getParameterMap()); //此方法会自动转换4类8种数据类型
```

3、处理Html中日期为String类型存为JavaBean中的定义为Date类型数据的方法

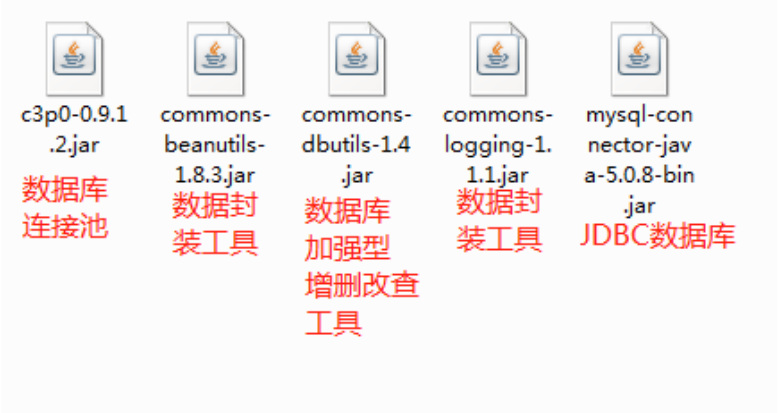
javabean中定义了 一个private Date birthday; 为Date类型

ConvertUtils.register(new DateLocaleConverter(), Date.class); //参数1：转换器　参数2：目标类型(注：要与实体类中的类型一致)

4、处理表单中数据返回是数组类型数据（如表单中的 checkbox会有多个选择这种情况）

```
String arr = Arrays.toString(request.getParameterValues("hobby")); //
{"",""}
arr = arr.substring(1, arr.length()-1);
user.setHobby(arr);
```

5、数据封装到JavaBean中去之后就是向数据库进行存取等操作



重点1、验证码的存

```
// 参数1：图片宽度 参数2：图片高度 参数3：字符个数 参数4：干扰线条数
ValidateCode vc = new ValidateCode(150, 30, 4, 9);
// 获得验证码文本
String code = vc.getCode();
// 得到session对象
HttpSession session = request.getSession();
// 把验证码保存到session域对象中
session.setAttribute("code", code);
// 通过响应字节流，把图片响应到浏览器
vc.write(response.getOutputStream());
```

获取到显示的验证码

获取session对象并将验证码保存到session对象中去

重点2、验证码的去HttpSession session对象保存数据

```
//==========================================================
//获得用户输入的验证码
String code1 = request.getParameter("code");
//从session域对象中获得验证码
session = request.getSession();
String code2 = (String) session.getAttribute("code");
//判断输入的验证码和显示的验证码不同(忽略大小写)
if(!code1.equalsIgnoreCase(code2)){
    //如果验证码输入的不正确  存页面需要提示信息，然后再页面中去取出返回给页面
    request.setAttribute("msg", "验证码错误");
    //请求不能通过 那么只能重新转发回原来的登录页面
    request.getRequestDispatcher("/login.jsp").forward(request, response);
    return;
}
//================以上代码处理验证问题=========================
```

重点3、登录用户名或密码错误的验证处理

```
if (db.upUser(user) == null) {
    // 提示密码或用户名有误 提示错误
    request.setAttribute("msg", "用户名或密码错误");// 因为页面需要提示信息，所以手动设置
    // 请求重新转发到登录页面
    request.getRequestDispatcher("/login.jsp").forward(request, response);
    return;
}
```

重点4、Cookie的会话数据处理

```
// 由于登录页面有记住用户名的复选框 登录成功需要判断用户是否钩选 记住用户名
// 获取是否勾选的状态
String remember = request.getParameter("remember"); // 在页面中的复选框名称
// 将用户输入的用户名保存到Cookie中去
Cookie cookie = new Cookie("remember", user.getUsername());// 创建Cookie对象
// 设置Cookie的保存路径
// cookie.setPath("/day13_01_login"); 方法一
// cookie.setPath(request.getContextPath()); 方法二
// 方法三
cookie.setPath("/");
// 判断这个状态
if (remember != null) {
    cookie.setMaxAge(Integer.MAX_VALUE); // 设置Cookie数据的保存时间
} else {
    cookie.setMaxAge(0); // 删除Cookie
}
response.addCookie(cookie);// 把Cookie对象数据写回到浏览器
```

重点5、转发到其他页面

```
//把user对象保存到session域对象中
session.setAttribute("user", user);
// 一切OK后重定向到首页去
request.getRequestDispatcher("/index.jsp").forward(request, response);
```

重点6、*javascript*技术与JSP技术与java技术及html技术组合　介绍及使用：

```
<script type="text/javascript">
  function changeImg(img){
      img.src = "/D13_JSP/codeServlet?time="+new Date().getTime();
  }
</script>
```

解决验证码点击改变

jsp中的javascript技术

```
<div>
<%
    String msg = (String)request.getAttribute("msg");
    if(msg!=null){
        out.print(msg);
    }
%>
</div>
```

页面中的提示通过请求获取并显示在页面上

```
<%
    //得到Cookie数组
    Cookie[] cookies = request.getCookies();
    String username = "";
    String ck = "";
    for(int i = 0; cookies!=null && i<cookies.length;i++){
        //根据Cookie的name，找到指定的Cookie对象
        if("remember".equals(cookies[i].getName())){
            username = cookies[i].getValue();
            ck = "checked='checked'";
            break;
        }
    }
%>
```

通过请求取出cookies里的所有数据通过Key获取值

```
<div class="col-sm-6">
  <input type="text" class="form-control" name="username"
  value="<%=username %>" id="username" placeholder="请输入用户名">
</div>
```

JSP表单中设置值

```
<label>
  <input type="checkbox" name="remember" <%=ck %> >记住用户名
</label>
```

```
<ol class="list-inline">
<!-- 用<% %>将JAVA代码写JSP的页面里面 -->
<%
    HttpSession session1 = request.getSession();
    User user = (User)session1.getAttribute("user");
    if(user!=null){
%>
        欢迎你：<%=user.getUsername() %>
<%
    }else{
%>
        <li><a href="login.htm">登录</a></li>
        <li><a href="register.htm">注册</a></li>
<%
    }
%>
    <li><a href="cart.htm">购物车</a></li>
</ol>
```

JAVA代码与HTML混合使用

登录功能实现的思想步骤：

1、创建web项目

2、创建用于管理登录表单的servler类

3、创建用户登录的页面和首页（login.jsp　　index.jsp）

4、获取login.jsp页面中的登录请求信息（登录表单请求数据）

4.1、获取采用BeanUtils工具对数据进行封装

4.2、C3P0创建数据库连接池

4.3、Dbutils实现数据库的增删改查功能，通过用户提交的数据去数据库查询，并返回JavaBean对象

4.4、在管理登录表单的servler类中调用相关方法对返回结果判断

如果返回JavaBean对象为空说明查询失败

否则查询成功进入页面的跳转（请求转发到首页页面中去）

5、在进行请求转发前获取页面的相关标记（如记住用户名）将用户名存入Cookie对象，在JSP页面中通过java代码取出并赋值

6、由于登录成功后进入新页面要使用用户的登录信息，所以在请求转发前 将用户请求的信息保存到HttpSession session对象中去

在新的转发的新JSP页面中获取用户信息

Cookie&Session会话技术

➢· 什么是会话　　:用户打开一个浏览器访问页面,访问网站的很多页面,访问完成后将浏览

器全都关闭的过程称为是一次会话.

如同打电话。

➢· 为什么使用会话技术?解决什么问题?

保存用户各自的数据。

* 私有的数据,购物信息数据保存在会话技术中.

➢· 常见的会话技术:

* Cookie :将数据保存到客户端浏览器.

* Session :将数据保存到服务器.

➢· 使用会话技术:

➢· 创建一个Cookie对象:

* Cookie(String name,String value); //cookie只能保存字符串数据。且不能保存中文

➢· 向浏览器保存数据:

HttpServletResponse有一个方法:

* void addCookie(Cookie cookie);

➢· 获得浏览器带过来的所有Cookie:

HttpServletRequest有一个方法:

* Cookie[] getCookies();

➢· Cookie的常用的API:

name: 一但赋值就不能再改了。

value:注意:不能存中文。

maxAge:Cookie缓存的有效时间。

-1:默认。代表Cookie数据存到浏览器关闭(保存在浏览器内存中)。

0:代表删除Cookie.如果要删除Cookie要确保路径一致。

正整数:以秒为单位保存数据有有效时间(把缓存数据保存到磁盘中)

path:表示保存Cookie的路径 默认是当前创建Cookie对象的路径

例:/day16/servlet/loginServlet 此Servlet创建的Cookie

访问:/day16/demoServlet能否访问到Cookie? 不能

/day16/loign.jsp 能不能访问到? 不能

/day16/servlet/aaa/bbb/ServletDemo2能不能访问到? 能

一般把路径设置在根路径下: setPath("/") 表示是当前应用下的所有资源都可以访问Cookie数据

* getName();

* getValue();

* setMaxAge(int maxAge); -- 设置Cookie的有效时间.

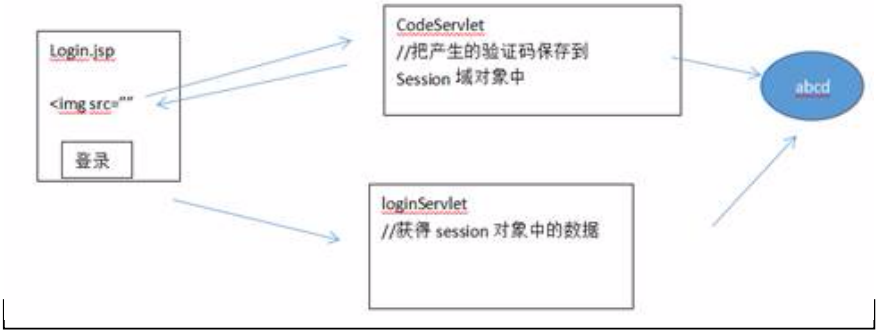
Cookie的数据保存在客户端（不安全）,Session数据保存在服务器端（安全）.
Cookie只能存字符串，Session可以存对象。

session也是一个域对象： ServletContext request
void setAttribute(String key,Object value);
Object getAttribute(String key);
removeAttribute(String key);

应用：购物车、保存用户信息
➢· Session的执行原理：基于Cookie的.
可以被多个资源所共享。一个session对应一个用户会话（浏览器）
➢· 使用Session:
  * request.getSession();//获得session

Session对象的生命周期：
    创建：getSession();
    活着：
        默认30分钟
        setMaxInactiveInterval(int interval) 秒
    销毁：超时。
        服务器非正常关闭。

getSession()原理：（了解）
    1、获得Cookie中的sessionid
    2、没取到sessionid，创建一个session对象同时产生一个新sessionid，并把sessionid写到浏览器
    3、得到这样的sessionid了，
        直接使用对应sessionid的session对象
        如果session销毁了，则执行步骤2



➢· 什么是JSP :Java Server Pages(Java服务器端页面)动态页面

jsp = html + java

➢· JSP的嵌入Java代码:JSP的脚本元素

* <%%> :翻译成类的service方法内部的内容. 定义变量,定义类,直接写代码块.

* <%=%> out.Print();

总结:

ServletContext ：适合共享数据.

* 生命周期:
  * 服务器启动的时候创建.
  * 服务器关闭的时候销毁.
* 作用范围:
  * 整个**web**应用**.**

HttpSession :私有的数据.登录用户的信息.

* 生命周期:
  * 服务器端第一次调用getSession()方法时候.才会创建一个session对象.
  * session销毁三种情况:
    * session过期:默认过期时间30分钟.
    * 非正常关闭服务器:(正常关闭服务器序列化到硬盘)
    * 调用session.invalidate();
* 作用范围:
  * 多次请求(一次会话)

HttpServletRequest :

* 生命周期:
  * 客户端向服务器发送请求的时候,服务器创建一个请求对象.
  * 服务器对这次请求作出响应之后.请求对象就会被销毁.
* 作用范围:
  * 一次请求