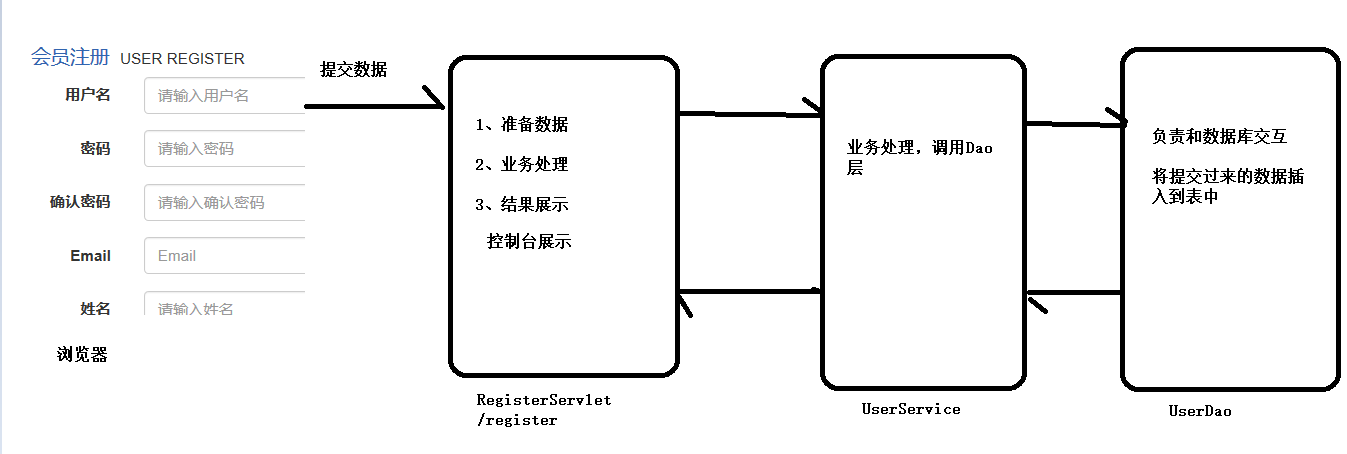
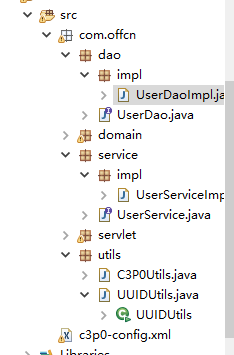
## 案例一：用户注册

### 流程分析



### 代码实现

包结构：

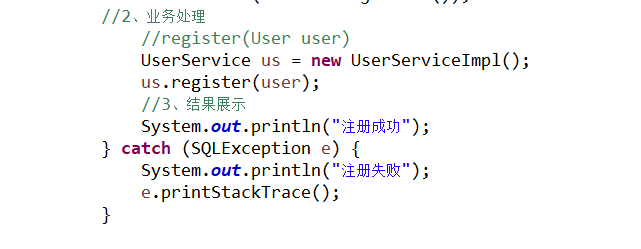


Register.html

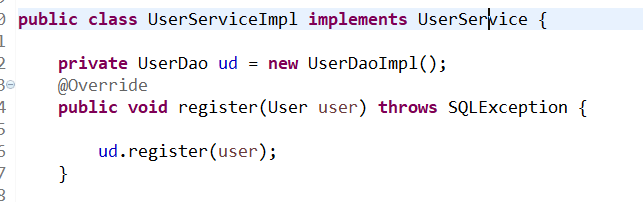


RegisterServlet:

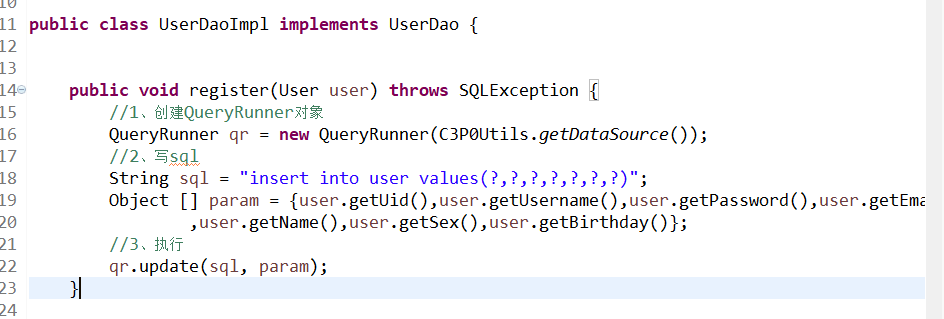




UserService:



UserDao:



### 1.3、代码优化

|  |  |
| --- | --- |
| 方法声明 | 功能描述 |
| String getParameter(String name) | 该方法用于获取某个指定名称的参数值，如果请求消息中没有包含指定名称的参数，getParameter()方法返回null；如果指定名称的参数存在但没有设置值，则返回一个空串；如果请求消息中包含有多个该指定名称的参数，getParameter()方法返回第一个出现的参数值 |
| String[] getParameterValues(String name) | HTTP请求消息中可以有多个相同名称的参数（通常由一个包含有多个同名的字段元素的FORM表单生成），如果要获得HTTP请求消息中的同一个参数名所对应的所有参数值，那么就应该使用getParameterValues()方法，该方法用于返回一个String类型的数组 |
| Enumeration getParameterNames() | getParameterNames()方法用于返回一个包含请求消息中所有参数名的Enumeration对象，在此基础上，可以对请求消息中的所有参数进行遍历处理 |
| Map getParameterMap() | getParameterMap()方法用于将请求消息中的所有参数名和值装入进一个Map对象中返回 |

#### 1.3.1、问题：每一个表单参数就要写一个getParameter()，不利于开发

获取指定属性单个值：getParameter(String name)

获取指定属性多个值：**getParameterValues**(String name)

获取整个参数列表：

返回值类型：Map<String,String[]>

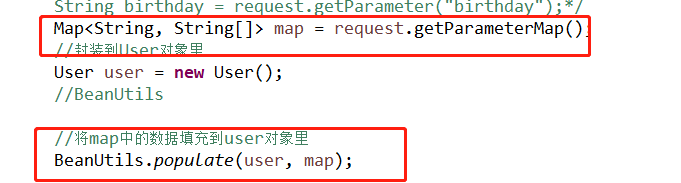
Key: value

Username [lucy]

Hobby [java,mysql,html]

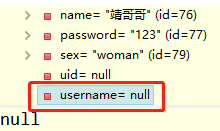


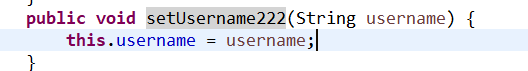
BeanUtils 可以使用map填充javaBean



注意：

1. map中的key，必须和javaBean setXxx对应





2、map中的value 必须和javaBean setXxx中属性的类型对应



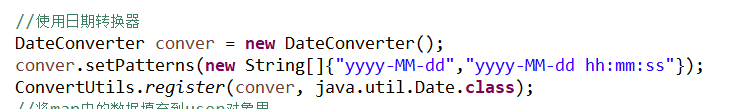


#### 1.3.2、问题：将生日改为日期类型

类型转换：

1. 使用DateConverter里的setPatterns(String[] str)
2. 注册转换器

ConvertUtils.register(转换器,类型的.class)



#### 1.3.3、问题：中文乱码(回顾)

请求：

Get：数据在请求行上，服务器对其固定解码为iso8859-1

New String(“”.getBytes(“iso8859-1”),utf-8s)

改配置文件:server.xml URIEncoding = “utf-8”

Post:

数据在请求体上，服务器对请求体的默认解码为iso8859-1



响应：



### 1.4、讲解：request对象

在Servlet API中，定义了一个HttpServletRequest接口，它继承自ServletRequest接口，专门用来封装HTTP请求消息。由于HTTP请求消息分为请求行、请求消息头和请求消息体三部分，因此，在HttpServletRequest接口中定义了获取请求行、请求头和请求消息体的相关方法，接下来，本节将针对这些方法进行详细地讲解。

#### 1.4.1、request继承树

ServletRequest 请求的顶层接口。 规定了普通请求的操作方法。 无法处理http协议

HttpServletRequest 是ServletRequest的子接口。 可以处理http协议

封装了请求行、头、体

实现类是根据服务器来确定，不同的服务器实现类不同的。

**request对象只能进行行、头、体 的读取数据操作，不能进行写入数据操作**

#### 1.4.2、request操作请求行

|  |  |
| --- | --- |
| **方法声明** | **功能描述** |
| String **getMethod**() | 该方法用于获取HTTP请求消息中的请求方式（如GET、POST等）  忽略大小写判断请求方式 |
| String getRequestURI() | 该方法用于获取请求行中资源名称部分，即位于URL的主机和端口之后、参数部分之前的部分  仅获取资源名，不包含参数列表 |
| String getQueryString() | 该方法用于获取请求行中的参数部分，也就是资源路径后面问号（?）以后的所有内容 |
| String getProtocol() | 该方法用于获取请求行中的协议名和版本，例如，HTTP/1.0或HTTP/1.1 |
| String **getContextPath**() | 该方法用于获取请求URL中属于WEB应用程序的路径，这个路径以“/”开头，表示相对于整个WEB站点的根目录，路径结尾不含“/”。如果请求URL属于WEB站点的根目录，那么返回结果为空字符串（""）  动态获取项目名称。  例如：/day12 |
| String getServletPath() | 该方法用于获取Servlet的名称或Servlet所映射的路径  url-pattern |
| String **getRemoteAddr**() | 该方法用于获取请求客户端的IP地址，其格式类似于“192.168.0.3”  本机：  0:0:0:0:0:0:0:1  localhost  127.0.0.1  统计IP |
| String getRemoteHost() | 该方法用于获取请求客户端的完整主机名，需要注意的是，如果无法解析出客户机的完整主机名，该方法将会返回客户端的IP地址 |
| int getRemotePort() | 该方法用于获取请求客户端网络连接的端口号 |
| String getLocalAddr() | 该方法用于获取Web服务器上接收当前请求网络连接的IP地址 |
| String getLocalName() | 该方法用于获取Web服务器上接收当前网络连接IP所对应的主机名 |
| int getLocalPort() | 该方法用于获取Web服务器上接收当前网络连接的端口号  接收请求的端口号 |
| String getServerName() | 该方法用于获取当前请求所指向的主机名，即HTTP请求消息中Host头字段所对应的主机名部分 |
| int getServerPort() | 该方法用于获取当前请求所连接的服务器端口号，即如果HTTP请求消息中Host头字段所对应的端口号部分  发送响应的端口号 |
| String getScheme() | 该方法用于获取请求的协议名，例如http、https或ftp |
| StringBuffer getRequestURL() | 该方法用于获取客户端发出请求时的完整URL，包括协议、服务器名、端口号、资源路径等信息，但不包括后面的查询参数部分。注意，getRequestURL()方法返回的结果是StringBuffer类型，而不是String类型，这样更便于对结果进行修改  获取到包含http协议的资源路径，不包含参数列表 |



#### 1.4.2、request操作请求头（了解）

|  |  |
| --- | --- |
| **方法声明** | **功能描述** |
| String getHeader(String name) | 该方法用于获取一个指定头字段的值，如果请求消息中没有包含指定的头字段，getHeader()方法返回null；如果请求消息中包含有多个指定名称的头字段，getHeader()方法返回其中第一个头字段的值 |
| Enumeration getHeaders(String name) | 该方法返回一个Enumeration集合对象，该集合对象由请求消息中出现的某个指定名称的所有头字段值组成。在多数情况下，一个头字段名在请求消息中只出现一次，但有时候可能会出现多次 |
| Enumeration getHeaderNames() | 该方法用于获取一个包含所有请求头字段的Enumeration对象 |
| int getIntHeader(String name) | 该方法用于获取指定名称的头字段，并且将其值转为int类型。需要注意的是，如果指定名称的头字段不存在，返回值为-1；如果获取到的头字段的值不能转为int类型，将发生NumberFormatException异常 |
| Long getDateHeader(String name) | 该方法用于获取指定头字段的值，并将其按GMT时间格式转换成一个代表日期/时间的长整数，这个长整数是自1970年1月1日0点0分0秒算起的以毫秒为单位的时间值 |



#### 1.4.4、request操作请求体(重点)

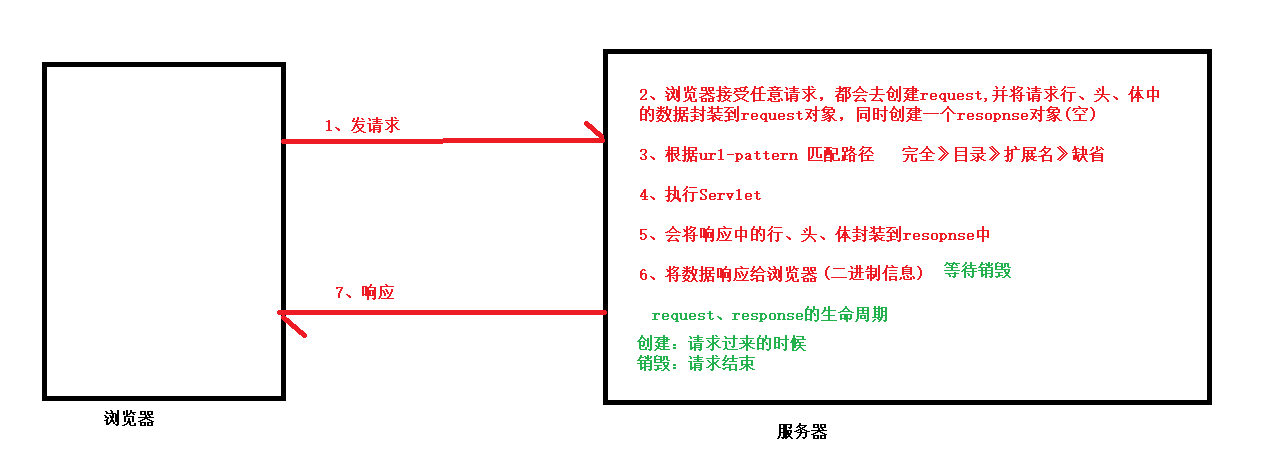
Post：才有请求体

获取指定参数的单个值：getParamater(String name)

获取指定参数的多个值 getParamaterValues(String name)

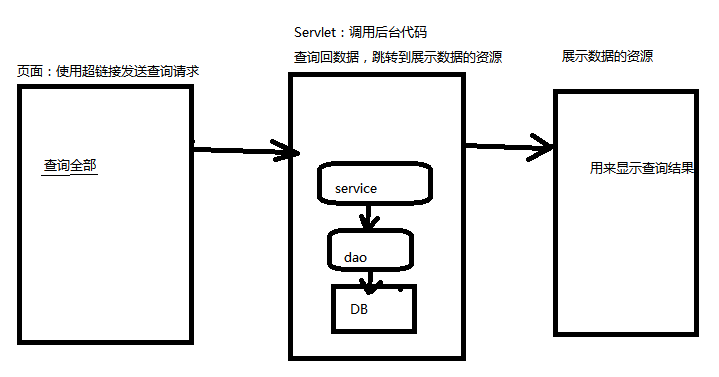
获取整个参数列表 getParamaterMap()

#### 1.4.5、request和response的执行流程(重点)



## 2、案例：查询全部用户信息并在客户端显示出来

### 2.1 流程分析



### 2.2 代码实现

### 2.3 存在问题

#### 2.3.1 查询结果用如何保存（保存在什么作用域？）

使用**ServletContext**作用域

生命周期：从服务器开始到服务器关闭

所有的Servelt都共享这一个域，就会造成数据冲突的问题

解决：

缩小作用范围 域对象

**request 作用域**

request的作用：

1. request可以封装请求行、头、体的信息
2. 也可以当成域对象

域对象的三个方法：

存数据：setAttribute(name,value)

取数据：getAttribute(name)

删除数据：removeAttribute(name)

作用范围：

一次请求有效

#### 2.3.2 如何跳转到展示数据的资源

1 重定向：从request域中取出的数据是空的，因为这时是两次请求。

2 请求转发：



请求转发：不会修改地址栏路径。

因为请求转发时，还未产生响应，request对象还未销毁，所以可以使用request作用域

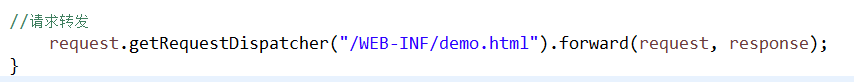
访问受保护目录下的资源

**requestDispatcher:是服务器的资源封装器，可以封装服务器内部所有资源。（包括WEB-INF下资源）**

WEB-INF是受保护目录，不能够通过浏览器直接访问



可以通过请求转发去访问



#### 2.3.3 请求转发和重定向的区别

|  |
| --- |
| 重定向是两次请求两次响应 转发是一次请求一次响应  重定向地址栏发生变化 而转发地址栏不变  重定向可以定向到任何网站 而转发只能本网站内部  request域作用范围是一次请求的访问.(使用重定向 request域中存的值 不能取到 而使用了转发的话 值就可以取到.)  何时使用请求转发？   1. 要求安全，不暴露程序路径 2. 要求高效 3. 使用request作用域 4. 访问受保护目录资源   何时使用重定向？   1. 跳转外网资源 2. 改变地址栏路径   只要不是上述六点，请求转发和重定向是可以相互替代的 |

#### 2.3.4 展示数据资源使用什么？

使用Servlet进行数据展示的缺点？

最大的缺点就是拼接展示数据繁琐。

## JSP入门

### 3.1 JSP是什么

Java Server Pages，SUN公司提供的一种动态网页编程技术。

JSP = HTML+java代码+JSP本身的标签

### 3.2 为什么要有JSP技术

1 Servlet输出页面很繁琐

2 Servlet生成的页面没有预览工具

### 3.3 JSP的运行原理

JSP是动态资源还是静态资源？

是动态资源，放的位置，编写和访问上很像静态资源。

**JSP本质上就是Servlet**

jsp如何运行？

一个jsp在第一次访问时，会被服务器中的jsp引擎（容器中编译jsp的软件模块）

编译成一个Servlet jsp名字\_jsp.java jsp名字\_jsp.class并且放在 Tomcat的work目录下。

JSP和Servlet表现形式上不同

a.编写jsp文件就是编写文本文件一样。编写servlet编写的是java程序。

b.编写jsp以后，直接将jsp拷贝到容器的部署目录就可以了。

c.编写后的servlet要通过web.xml文件加以配置。jsp直接访问就行了。

d.jsp和静态资源放在一起就可以了。-----jsp在使用上更像静态资源。

### 3.4 JSP中如何嵌入java代码

jsp中三种脚本标签：

<%! %> 这个标签可以用来声明变量，方法 ，类

变量和方法是 成员变量和成员方法

类是成员内部类

一般不用该标签

<% %> 在这个标签中编写的java代码

将会出现在 \_jspService()

<%= %> 在这个标签中写的java代码将会出现在

out() 中 向页面输出 该标签中的代码

一定不要加分号

### 3.5 JSP中的注释

<!-- -->

<%-- --%>

### 3.6 修改上面案例使用jsp作为数据展示的资源