

# UserCoinsCoinsProject

## 要求

---

- 使用 java 实现, build 工具使用 maven 或 gradle
- 8080 端口为 api 端口, 8081 端口为 ops 端口, 实现访问 <http://host:8081/ops/jstack> 返回当前 java 进程 jstack
- 数据存储使用 mysql
- 最终交互一个可以独立部署的 microservice 应用. (包含启动脚本, 例如 demo-project.sh [start|stop|restart|status] )
- 框架不限
- 包含单元测试和测试覆盖率统计

## 实现情况

---

- 使用 java 实现, build 工具使用 maven, 详细配置文件见项目中的 pom.xml
- 使用 8080 端口完成访问, java web app 使用 jersey restful api 和 spring 框架实现
- 数据存储使用 mysql, 数据库代码同要求的, 未添加其他。通过 JPA 将运行期的实体对象持久化到数据库中。
- 通过 tomcat 自带的启动 startup.bat 停止 shutdown.bat 等脚本实现独立部署 web 应用。
- 框架使用 Spring 和 JPA
- 单元测试和测试覆盖率统计。

## 使用说明

---

- 将 UserCoinsProject.war 放在 Tomcat 根目录下的 webapp 中
- 使用 Tomcat 根目录下 bin 文件夹总的 startup.bat 开启 Server
  - 在网址中输入 <http://localhost:8080/UserCoinsProject>
  - 进行相应的操作

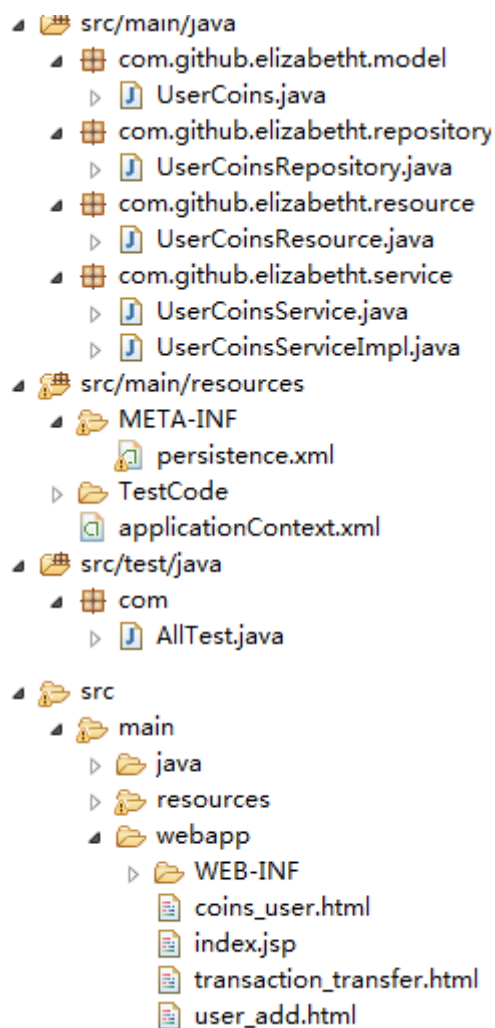
## 项目详解

---

**代码测试:** 测试覆盖率报告在相应文件夹中, 可以具体查询结果。主要技术就是通过 Jersey test framework 中的 WebResource 进行网址传递, 同时利用 form 进行参数传递

**Maven:** 可以通过一小段描述信息来管理项目的构建, 报告和文档的软件项目管理工具。主要实在 pom.xml 中, 其中可以通过它自动下载需要的 jar 夹包并且实现项目的构建和打包。

**Spring 框架:** 在 web.xml 中配置一些必要信息, 包括 listener 和 servlet. 同时在 applicationContext.xml 定义 JPA 和 Hibernate 关系配置信息, 包括使用 bean 配置数据库的连接等。项目主要框架是 html folder 和 java folder. 如下。其中 html folder 是为了一下操作更简单化, 通过表单传递给 restful api。Java folder 中包括 Resource (restful api 中的 get post 操作分析), Service (功能函数), Repository (联系数据库的操作) and Model (对应数据库)。



```
src/main/java
├── com.github.elizabetht.model
│   └── UserCoins.java
├── com.github.elizabetht.repository
│   └── UserCoinsRepository.java
├── com.github.elizabetht.resource
│   └── UserCoinsResource.java
├── com.github.elizabetht.service
│   ├── UserCoinsService.java
│   └── UserCoinsServiceImpl.java
├── src/main/resources
│   ├── META-INF
│   │   └── persistence.xml
│   └── TestCode
│       └── applicationContext.xml
├── src/test/java
│   └── com
│       └── AllTest.java
└── src
    ├── main
    │   ├── java
    │   └── resources
    └── webapp
        ├── WEB-INF
        │   ├── coins_user.html
        │   ├── index.jsp
        │   ├── transaction_transfer.html
        │   └── user_add.html
```

**JPA:** 在 persistence.xml 中定义 persistent unit, 通过 @annotation 在 UserCoins.java 进行数据库 table 的对应

## 具体代码

### 测试

```
@Test
public void testCoinsUserSuccess() {
    WebResource webResource = client().resource("http://localhost:8080/");
    Form form = new Form();
    form.add("userid", "3");

    String res = webResource
        .path("/UserCoinsProject/webapi/UserCoinsResource/coins/user")
        .type(MediaType.APPLICATION_FORM_URLENCODED_TYPE)
        .post(String.class, form);
    assertEquals("user 3 coins is 20", res);
}
```

### restful api

```
@POST
@Path("coins/user")
@Consumes(MediaType.APPLICATION_FORM_URLENCODED)
@Produces(MediaType.TEXT_PLAIN)
public String coinsuser(@FormParam("userid") int userid)
    throws ParseException {
    int coins = userCoinsService.findByUserId(userid);
    if(coins==-1){
        return "user is not existed";
    }
    return "user " + userid + " coins is " + coins;
}
```

### 操作数据库

```
@Repository("UserCoinsRepository")
public interface UserCoinsRepository extends JpaRepository<UserCoins, Long>{
    @Query("select s from UserCoins s where s.user_id = :userid")
    UserCoins findByUserId(@Param("userid") int userid);

    @Modifying
    @Query("update UserCoins u set u.coins = :coins where u.user_id = :userid")
    @Transactional
    void UpdateUserCoins(@Param("userid")int userid, @Param("coins")int coins);
}
```

### 对应数据库

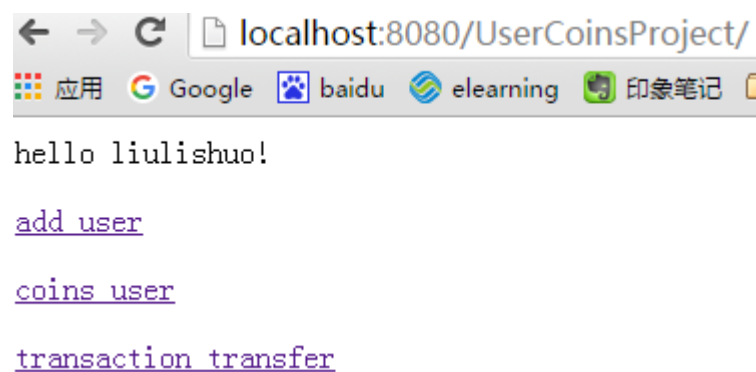
---

```
@Component
@XmlRootElement(name="coins")
@Entity
@Table(name="coins")
public class UserCoins {
    @Id
    @GeneratedValue
    private int id;

    @NotNull
    private int user_id;

    @NotNull
    private int coins;
```

## 运行结果



添加 **user** 和 **coins**

Userid :

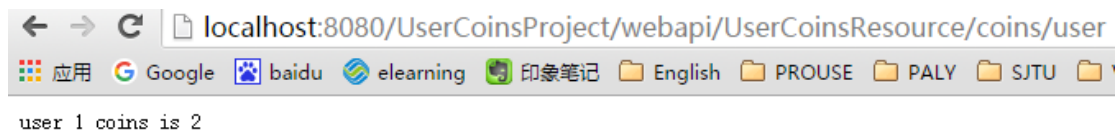
Coins :

---



### 查询 user 的 coins

Userid :



### 转让 coins

FromUserid :

ToUserid :

Coins :



## 参考资料:

EclEmma 测试覆盖率: <http://liangruijun.blog.51cto.com/3061169/803473>

Jersey-test-framework 测试代码: <http://www.hascode.com/2011/09/rest-assured-vs-jersey-test-framework-testing-your-restful-web-services/>

使用 jersey restful api 和 spring 框架实现 java web app:

<http://elizabethht.github.io/blog/2013/12/13/student-enrollment-using-jersey-rest-with-spring/>