[Unit Test Plan]单元测试计划

[1352845 郭雨卿 1352953 朱嘉业]

# 引言

## 目的

本文档为Imangine图片分享网站的单元测试活动提供范围、方法、资源和进度方面的指导

本文档的读者，主要是开发（测试）经历、测试人员和开发人员。

## 测试策略

以类为单元，采用**孤立测试策略**，通过设计相应的驱动和桩的方法来测试类中的方法。

根据本网站的业务逻辑特征，在选择类的被测方法时，我们在每个模块中抽取核心业务相关的类进行测试。大部分方法的规模和复杂度也达到了一定程度（非空且非注释代码行数LOC>10），此外，如果涉及核心功能方法的LOC未达标准，我们同样也要对其进行测试，其他方法不进行单元测试。

执行单元测试的次序是根据《Imangine需求分析规约》中的系统总体流程图，根据其中业务流程相关的类、依赖关系从小到大及其复杂度，整理出如下单元测试次序：

登录注册 -> 上传图片 -> 新建专辑 -> 管理小组

## 范围

单元测试主要包含了计划阶段、设计阶段、实现阶段和执行阶段4个阶段。本单元测试计划是整个软件开发项目中的一部分，起始于详细设计阶段，直到单元测试阶段结束后终止。该计划主要处理与Imanging图片分享网站系统单元测试有关的任务安排、资源需求、人力需求、风险管理、进度安排等内容。

## 参考文献

《Imangine需求规约》

《Imangine需求分析规约》

《Imangine概要设计》

《Imangine详细设计》

## 术语

# 测试项目

根据《Imangine详细设计》中的详细设计内容，单元测试的测试项目如下所示。

## 账户管理模块 - 登录注册

* UserController

| 方法表示符 | 方法名 | 代码行(LOC) |
| --- | --- | --- |
| UT-001-FUN-001 | public standardRes register(String userName, String password,String avatar, String email,String city, Date birthday, String description) | 29 |
| UT-001-FUN-002 | public loginRes login(String userName, String password) | 11 |

* UserService

| 方法表示符 | 方法名 | 代码行(LOC) |
| --- | --- | --- |
| UT-002-FUN-001 | public boolean checkToken(String token) | 3 |
| UT-002-FUN-002 | public String token2id(String token) | 3 |
| UT-002-FUN-003 | public standardRes register(String username, String password, String avatar, String email, String city, Date birthday, String description) | 21 |
| UT-002-FUN-004 | public loginRes login(String username, String password) | 40 |

* UserDAO

| 方法表示符 | 方法名 | 代码行(LOC) |
| --- | --- | --- |
| UT-003-FUN-001 | public interface UserDAO extends CrudRepository<UserEntity,Long> | 3 |

* CryptoService

| 方法表示符 | 方法名 | 代码行(LOC) |
| --- | --- | --- |
| UT-004-FUN-001 | private static byte[] initHmacSHA256Key() | 4 |
| UT-004-FUN-002 | public static String encodeHmacSHA256(byte[] data) | 6 |
| UT-004-FUN-003 | private static String byte2Hex(byte[] src) | 8 |
| UT-004-FUN-004 | public static String encodeSHA256(byte[] data) | 14 |

## 个人主页管理模块

* 上传图片

| 方法表示符 | 方法名 | 代码行(LOC) |
| --- | --- | --- |
|  |  |  |
|  |  |  |
|  |  |  |

* 新建专辑

| 方法表示符 | 方法名 | 代码行(LOC) |
| --- | --- | --- |
|  |  |  |
|  |  |  |
|  |  |  |

* 管理小组

| 方法表示符 | 方法名 | 代码行(LOC) |
| --- | --- | --- |
|  |  |  |
|  |  |  |
|  |  |  |

# 被测函数

根据测试策略中制定的被测方法选取标准，被测函数如下表所示。

## 账户管理模块 - 登录注册

* UserController

| 方法表示符 | 方法名 | 代码行(LOC) |
| --- | --- | --- |
| UT-001-FUN-001 | public standardRes register(String userName, String password,String avatar, String email,String city, Date birthday, String description) | 29 |
| UT-001-FUN-002 | public loginRes login(String userName, String password) | 11 |

* UserService

| 方法表示符 | 方法名 | 代码行(LOC) |
| --- | --- | --- |
| UT-002-FUN-003 | public standardRes register(String username, String password, String avatar, String email, String city, Date birthday, String description) | 21 |
| UT-002-FUN-004 | public loginRes login(String username, String password) | 40 |

其中DAO层代码逻辑较为简单，故不进行单元测试。此外还调用了CryptoService对用户密码进行加密，这一部分的测试重点应当在接口部分，故在集成测试中再行测试。

* 上传图片

| 方法表示符 | 方法名 | 代码行(LOC) |
| --- | --- | --- |
|  |  |  |
|  |  |  |
|  |  |  |

* 新建专辑

| 方法表示符 | 方法名 | 代码行(LOC) |
| --- | --- | --- |
|  |  |  |
|  |  |  |
|  |  |  |

* 管理小组

| 方法表示符 | 方法名 | 代码行(LOC) |
| --- | --- | --- |
|  |  |  |
|  |  |  |
|  |  |  |

# 不被测函数

对于不满足测试策略中选取的被测试方法标准的方法，将不进行单元测试，但是这些方法必须经过严格的代码监视，以保证不会出现低级错误，并且在集成测试阶段统一验证其接口功能的正确性，其列表如下：

| 方法表示符 | 方法名 | 代码行(LOC) | 所在类 |
| --- | --- | --- | --- |
| UT-002-FUN-001 | public boolean checkToken(String token) | 3 | UserController |
| UT-002-FUN-002 | public String token2id(String token) | 3 | UserController |
| UT-003-FUN-001 | public interface UserDAO extends CrudRepository<UserEntity,Long> | 3 | UserDAO |
| UT-004-FUN-001 | private static byte[] initHmacSHA256Key() | 4 | CryptoService |
| UT-004-FUN-002 | public static String encodeHmacSHA256(byte[] data) | 6 | CryptoService |
| UT-004-FUN-003 | private static String byte2Hex(byte[] src) | 8 | CryptoService |

# 测试方法

根据类的规约和操作规约，并且将黑盒测试方法与白盒测试方法结合起来，构造一系列测试用例。其中，用到了等价类划分法、边界值分析法、错误推测法、决策表法等黑盒测试方法；以及逻辑覆盖测试法等白盒测试方法。

对于具有特殊需求的类，辅以下面两种方法设计测试用例：

1. 根据状态转换图构建测试用例，该方法根据被测试的类的对象所处的状态以及状态之间的转移来构造测试用例，对状态之间和状态内部的每一个转换及其可能发生的异常转换、转换的监控条件等进行全面的测试。尽量做到对于逻辑复杂的类，抽象出状态图，测试用例对其状态图的每一个Transaction进行覆盖。

2. 对于该类在实际流程中所牵涉到的一些前置条件和后置条件，都应该进行考虑。列出其前置条件和后置条件，然后尽可能的进行组合，对此设计测试用例。

# 测试通过/失败标准

**测试通过的标准如下：**

1. 所有单元测试的用例都被执行并通过。

2. 所有发现的缺陷都被修整并通过回归测试。

3. 所有被测对象的前置条件和后置条件组合覆盖率达到100%，或能明确给出不需要达到的理由。

4. 单元测试报告被授权人批准。

**测试失败的标准如下：**

1. 严重缺陷密度大于15个/KLOC.

2. 发现软件结构有重大设计问题，其修改会导致20%以上接口、功能、数量的变化，进一步测试相关特性已经无意义。

3. 发现关键功能未被设计，该功能的设计会导致20%以上接口、功能、数量的变化，进一步测试相关特性已经无意义。

**测试结果审批过程：**

开发人员提交单元测试报告 -- 开发或测试经理签字并提交SQA -- SQA对报告进行评审并签字 -- 产品经理签字

# 测试挂起/恢复的条件

**测试挂起的条件有：**

1. 当某个类在单元测试执行过程中，发现有阻塞用例的时候，该类的单元测试被挂起。

2. 当有20%以上的被测类都遇到有阻塞用例的时候，所有类的单元测试被挂起。

3. 当出现新增需求，或者开发人员提出要进行设计变更的时候，相应的类的单元测试被挂起。

**测试恢复的条件有：**

1. 测试被挂起的条件已经被解决。

2. 需要恢复测试的对象达到单元测试入口条件（即被测对象已经通过代码走读和语法检查）。

# 单元测试交付物

1. 单元测试计划

2. 单元测试用例规格及其内容

3. 单元从而是驱动和桩代码

4. 单元测试执行日志

5. 单元测试报告

# 单元测试任务

单元测试任务表如下：

| 任务标识 | 任务描述 | 责任人 | 优先级 | 依赖关系 |
| --- | --- | --- | --- | --- |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |

# 环境需求

## 硬件需求

CPU: Intel Pentium 4 1.7GHz以上

主存：512MB以上

## 软件需求

系统要求：WinXP及其以上、OS X 10.7 Lion及其以上

## 测试工具

Junit4、Mocha.js-2.5.3

# 角色和职责

单元测试角色和职责如下表所示

| 角色 | 职责 |
| --- | --- |
|  |  |
|  |  |
|  |  |

# 单元测试进度

单元测试进度安排表如下所示

| 任务标识 | 任务描述 | 起始日期 | 周期 |
| --- | --- | --- | --- |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

# 风险和应急计划

风险和应急计划安排表如下所示

| 风险ID | 风险描述 | 责任人 | 优先级 | 规避措施 | 应急计划 |
| --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |

# 审批

计划提交人签字： 日期：

开发经理签字： 日期：

产品经理签字： 日期：