<项目名称>

测试报告

版本 <4.0>

修订历史记录

|  |  |  |  |
| --- | --- | --- | --- |
| **日期** | **版本** | **说明** | **作者** |
| <28/08/2018> | <1.0> | 编写了文档的最初版本 | 汪喆昊 |
| <06/09/2018> | <2.0> | 补全了一些遗漏的信息 | 汪喆昊 |
| <08/09/2018> | <3.0> | 增加了后端测试的信息 | 李琥 |
| <09/09/2018> | <4.0> | 审核了文档 | 同睿哲 |

目录

1. 简介 4

1.1 目的 4

1.2 范围 4

1.3 定义、首字母缩写词和缩略语 4

1.4 参考资料 4

1.5 概述 4

2. 测试概要 4

3. 测试环境 5

4. 测试结果及分析 5

4.1 需求覆盖率及缺陷分布 5

4.2 缺陷严重程度 5

5. 缺陷清单 6

5.1 功能性缺陷 6

5.2 非功能性缺陷 6

6. 测试结论与建议 7

测试报告

# 简介

## 目的

该测试报告目的是记录XPlanner项目的各方面测试环境、条件、数据、结果，并且对测试结果进行分析，并对项目进行改进。

## 范围

该测试报告影响到XPlaner项目前后端代码、测试用例文档、项目总结报告。

## 定义、首字母缩写词和缩略语

### XPlanner

XPlanner为面向在校大学生的日程管理工具。在文中，根据上下文不同，XPlanner会指代本项目或本项目的前后端代码。

### Planner

逻辑上类似于插件的功能。逻辑代码直接编写在XPlanner中。通过合理安排架构使得增加扩展功能变得比较方便。

### Planner Store

管理Planner的中心。用户可以根据需要在Planner Store中启用、禁用Planner。

### Keeper

记录用户饮食信息，分析并生成日程的Planner。

### Spider

爬取一些常用的消息发布网站的通知并生成日程，由用户选择添加的Planner。

### Reader

接受用户输入的文本，生成日程并导入日程表的Planner。

## 参考资料

《Xplanner软件需求规约》，《测试用例》。

## 概述

测试概要包含了测试的时间、地点、人员、测试方法、测试内容的简要说明。

测试环境描述了测试使用的软硬件环境及其配置。

测试结果及分析描述了测试的结果，并给出一些分析。

缺陷清单包含所有功能性缺陷和非功能性缺陷相关的信息。

测试结论与建议包含了对于项目测试的总结和对项目的改进建议。

# 测试概要

测试进行于9月5、6、7号，测试地点为虚拟地点（组员屏幕共享）。人员为所有组员，包括同睿哲、李琥、贾兴国、汪喆昊。测试前期主要采用白盒测试、错误猜测、界面测试、控制流测试、基本路径测试，以确保操作逻辑、数据流的正确性和软件的可靠性；测试中期采用黑盒测试、变异测试、界面测试以及探索性测试，目的是增加软件可用性和；测试后期采用集成测试，随机测试和潜在用户的即兴测试，以测试软件兼容性及其整体效果。

# 测试环境

实机测试使用了iphone6s plus，64位，苹果A9+M9协处理器。

电脑测试分别使用

1. windows10，64位，i7 8750H处理器。
2. windows7 64位，i7 6500U处理器。
3. windows10 64位，i5 6200U处理器。
4. osx high sierra 64位，i7处理器。

# 测试结果及分析

## 需求覆盖率及缺陷分布

**表1 需求覆盖率及缺陷分布**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **测试内容** | | **测试**  **用例数** | **需求**  **覆盖率** | **缺陷数** | **缺陷率** | **备注** |
| **功能项** | **管理账号（包括绑定jaccount、登录）** | 5 | 100% | 0 | 0% | 无 |
| **管理日程** | 9 | 100% | 0 | 0% | 无 |
| **使用Reader管理食物信息和日程** | 2 | 100% | 0 | 0% | 无 |
| **使用Keeper管理运动信息和日程** | 3 | 100% | 0 | 0% | 无 |
| **使用Spider管理通知信息和日程** | 3 | 100% | 0 | 0% | 无 |
| **管理Planner** | 1 | 100% | 0 | 0% | 无 |
| **功能项小计** | 23 | 100% | 0 | 0% | 无 |
| **非功能项** | **性能** | 3 | 100% | 1 | 33% | 无 |
| **可靠性** | 1 | 100% | 0 | 0% | 无 |
| **易用性** | 6 | 100% | 1 | 17% | 无 |
| **可测试性** | 1 | 100% | 0 | 0% | 无 |
| **可扩展性** | 1 | 100% | 0 | 0% | 无 |
| **可维护性** | 1 | 100% | 0 | 0% | 无 |
|  | **非功能项小计** | *13* | 100% | *2* | 15% | 无 |
|  | **总计** | *36* | 100% | *2* | 5% | 无 |

## 缺陷严重程度

**表2 缺陷严重程度**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **严重程度** | **致命** | **严重** | **一般** | **细微** | **总数** |
| **缺陷个数** | 0 | 0 | 0 | 2 | 2 |
| **占缺陷百分比** | 0% | 0% | 0% | 100% | 100% |

# 缺陷清单

## 功能性缺陷

**表3 功能性缺陷列表**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **序**  **号** | **缺陷**  **编号** | **严重程度** | **功能模块** | **缺陷标题** | **缺陷描述** | **测试用例编号** |
|  |  |  |  |  |  |  |

表中相关项说明：

1. 缺陷编号：为每个缺陷赋予一个唯一的编号，可以通过此编号对缺陷进行跟踪。例如：Bug001。
2. 缺陷严重程度：缺陷可以根据严重程度分为以下几种情况。
3. 致命（fatal）：致命的错误，测试执行直接导致系统死机、蓝屏、挂起、或是程序非法退出；系统的主要功能或需求没有实现。
4. 严重（critical）：严重错误，系统的次要功能点或需求点没有实现；数据丢失或损坏。执行软件主要功能的测试用例导致系统出错，程序无法正常继续执行；程序执行过于缓慢或是占用过大的系统资源。
5. 一般（major）：不太严重的错误，这样的缺陷虽然不影响系统的基本使用，但没有很好地实现功能，没有达到预期的效果。如次要功能丧失，界面错误，打印内容、格式错误，提示信息不太正确，或用户界面太差，简单的输入限制未放在前台进行控制，删除操作未给出提示，操作时间长等。
6. 细微（minor）：一些小问题，对功能几乎没有影响，产品及属性仍可使用。如软件的实际执行过程与需求有较小的差异；程序的提示信息描述容易使用户产生混淆；有个别错别字、 文字排列不整齐；辅助说明描述不清楚，显示格式不规范，长时间操作未给用户进度提示，提示窗口文字未采用行业术语，可输入区域和只读区域没有明显的区分标志，系统处理未优化等。
7. 功能模块：所测试并出现该缺陷的功能模块名称。
8. 缺陷标题：描述缺陷的标题。
9. 缺陷描述：对缺陷的详细描述。描述缺陷的重现步骤和问题所在，描述需清晰明了，使研发人员可以按照描述将缺陷重现出来。必要时可以放上相应的截图，以便于研发人员理解问题所在。

## 非功能性缺陷

**表4 非功能性缺陷列表**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **序**  **号** | **缺陷**  **编号** | **严重程度** | **测试类型** | **缺陷标题** | **缺陷描述** | **测试用例编号** |
| 1 | 01 | 细微 | 易用性 | 月历设计没有考虑折叠 | 月历设计没有设计成可折叠的样式。 | （由于不可抗因素没有在测试用例中写明） |
| 2 | 02 | 细微 | 性能 | 用户登录时加载用户数据速度不够快 | 用户登录（绑定jaccount）后，加载所有用户数据所需时间比较长。 | XPlanner-testcase-044 |

# 测试结论与建议

在本软件的开发周期中，软件测试占据较大的比例。前期采用白盒测试等一系列测试保证了软件操作逻辑和数据流的正确，很好地避免了严重和致命错误的发生；中期的测试更具有探索性，一方面探索更高效的需求开发，另一方面以测试驱动开发，在原本的代码上添加了必要的补充和功能的完善，大大增强了软件可用性；后期测试则更为随机，由潜在用户所进行的测试帮助本小组对客户需求有了更实际的理解，给小组的反思和进步带来了大量经验。

目前，项目本身基本没有检测出功能性缺陷，不过有一定的非功能性缺陷。

对于非功能性缺陷01“月历设计没有考虑折叠”这个缺陷，我们认为这是对于用户体验考虑不够周全而导致的。日后在设计界面、功能的时候，要充分地考虑用户体验。

对于非功能性缺陷02“登录时加载用户信息不够快”这个缺陷，我们认为这是一开始设计时没有对性能的足够要求导致的。其实本缺陷并不是很明显，列为缺陷只是为了提醒读者日后注意此问题。

综上，我们在项目中尽可能多地进行了测试并且改进项目。我们会尝试着在以后的学习、实践中活用从本项目测试工作中获取的经验。