**Shell程序简要开发文档**

1. 、开发环境及工具

Manjaro定制系统

Qtcreater6.7

1. 、功能（需求）描述

2.1 shell 程序能够提供命令的输入，执行并显示执行结果的功能。

2.2 shell 程序能够提供可 shell 编程的功能，能够执行简单的 shell 脚本。

2.3 shell 程序能够提供 I/O 重定向和管道的功能。

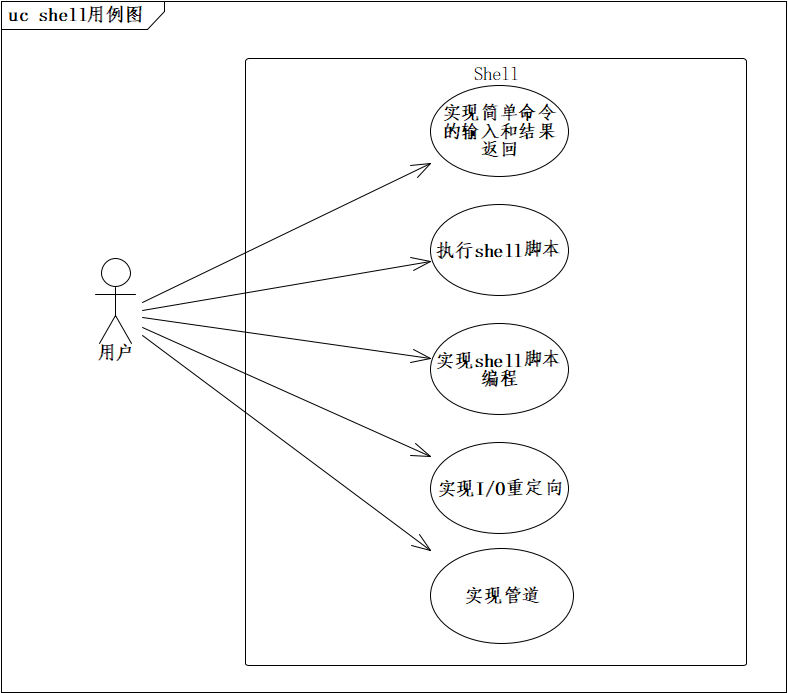


图1.1 Shell程序用例图

1. 编写目的

用户最好具有基本的计算机科学知识，了解基本的编程概念和命令行操作。并且熟悉一些Shell脚本的基础语法和命令。用户最好有一定的Linux或Unix系统的使用经验，能够理解命令行环境下的工作方式，熟悉常见的Shell命令和管道操作。

1. 项目背景
2. 项目名称：shell程序。
3. 项目负责人：杨检
4. 系统设计人员：杨检、马秋玲、王新龙
5. 具体的功能实现

5.1 简单命令的执行操作

表5.1 “简单命令的执行操作”用例文档

|  |  |
| --- | --- |
| **用例名** | 简单命令的执行操作 |
| **简要描述** | 用户通过该用例进行简单命令的操作 |
| **参与者** | 用户，shell系统 |
| **前置条件** | Shell启动 |
| **后置条件** | 无 |
| 基本事件流  （1）该用例起始于shell启动  （2）输入简单指令  （3）接受指令检验是否合法  （4）弹出提示  （5）弹出结果 | |
| **备选事件流**   1. \* 用户在操作提交之前，随时都能够结束子用例   A-1 操作失败   1. 平台给提示操作执行失败 2. 用户可以选择再次执行，也可以结束该用例 | |
| **补充约束-数据需求**  无 | |
| 相关图 | |

5.1 编译脚本

表5.2 “编译脚本”用例文档

|  |  |
| --- | --- |
| **用例名** | 编译脚本 |
| **简要描述** | 用户通过该用例进行编译脚本 |
| **参与者** | 用户，shell系统 |
| **前置条件** | Shell启动 |
| **后置条件** | 无 |
| 基本事件流  （1）该用例起始于shell启动  （2）输入vim命令  （3）进入编辑界面  （4）编写代码  （5）保存退出 | |
| **备选事件流**   1. \* 用户在操作提交之前，随时都能够结束子用例   A-1 操作失败   1. 平台给提示操作执行失败 2. 用户可以选择再次执行，也可以结束该用例 | |
| **补充约束-数据需求**  无 | |
| 相关图 | |

表5.3 “执行脚本”用例文档

|  |  |
| --- | --- |
| **用例名** | 执行脚本 |
| **简要描述** | 用户通过该用例进行执行脚本 |
| **参与者** | 用户，shell系统 |
| **前置条件** | Shell启动 |
| **后置条件** | 无 |
| 基本事件流  （1）该用例起始于shell启动  （2）输入执行shell命令  （3）接受命令  （4）执行命令  （5）得到结果 | |
| **备选事件流**   1. \* 用户在操作提交之前，随时都能够结束子用例   A-1 操作失败   1. 平台给提示操作执行失败 2. 用户可以选择再次执行，也可以结束该用例 | |
| **补充约束-数据需求**  无 | |
| 相关图 | |

表5.4 “实现I/O重定向”用例文档

|  |  |
| --- | --- |
| **用例名** | 实现I/O重定向 |
| **简要描述** | 用户通过该用例进行实现I/O重定向 |
| **参与者** | 用户，shell系统 |
| **前置条件** | Shell启动 |
| **后置条件** | 无 |
| 基本事件流  （1）该用例起始于shell启动  （2）输入重定向命令  （3）接受命令  （4）执行命令  （5）输入查询命令  （6） 得到结果 | |
| **备选事件流**   1. \* 用户在操作提交之前，随时都能够结束子用例   A-1 操作失败   1. 平台给提示操作执行失败 2. 用户可以选择再次执行，也可以结束该用例 | |
| **补充约束-数据需求**  无 | |
| 相关图 | |

表5.5 “实现管道”用例文档

|  |  |
| --- | --- |
| **用例名** | 实现管道 |
| **简要描述** | 用户通过该用例进行实现管道 |
| **参与者** | 用户，shell系统 |
| **前置条件** | Shell启动 |
| **后置条件** | 无 |
| 基本事件流  （1）该用例起始于shell启动  （2）输入管道命令  （3）接受命令  （4）创建管道  （5）创建子进程  （6） 进行命令  （7） 得到结果 | |
| **备选事件流**   1. \* 用户在操作提交之前，随时都能够结束子用例   A-1 操作失败   1. 平台给提示操作执行失败 2. 用户可以选择再次执行，也可以结束该用例 | |
| **补充约束-数据需求**  无 | |
| 相关图  IMG_256 | |

1. 核心技术

Split函数：的功能是将用户输入的字符串按空格分割为单词，并返回一个字符串向量。这是整个程序中处理用户输入命令的一个重要组成部分。

execute\_command函数：负责执行用户输入的命令。它涵盖了创建子进程、执行命令、I/O 重定向和管道操作等关键部分，是整个程序实现命令执行功能的核心。

execute\_script 函数：允许用户从脚本文件中读取并执行命令。它打开脚本文件，逐行读取其中的命令并调用 execute\_command 函数执行，是实现脚本执行功能的核心部分。

main函数：中的循环结构负责提供一个交互式的命令行界面，让用户可以输入命令并与Shell交互。它包含了对用户输入的处理逻辑，判断是否执行脚本或普通命令，以及退出程序的逻辑。

1. 参考资料

参考书籍：

《Unix-Linux 编程实践教程》

《经典Shell脚本编程指南》