**Myfox聊天软件测试报告**

**（UT测试、FT测试）**

**一、UT测试**

**1.UT测试计划**

**1.1测试目标**

首先在随着编写各个模块的类代码的时候，对一些比较重要或者复杂的类单独抽出来进行测试，对一些比较关键的static方法和数据也进行测试，在整个编码的过程中保证软件的关键部分的正确性，随着编码的进度构造相应的测试用例和结果验证数据。

**1.2测试方法**

Myfox聊天软件分为客户端和服务器端，使用java语言，jdk版本为jdk8u171-linux-64,编码和运行环境为ubuntu16.04 64bit,服务器端的代码在本地编辑，使用git远程传输到服务器。项目的客户端代码和服务器端代码使用javac和java命令编写的bash脚本编译和执行，实现自动化快速编译和执行，针对一些小模块的数据和方法，编写了UTTest.java文件用来作为测试的入口文件：

UTtest.java:

import java.util.\*;

import java.io.\*;

import java.math.\*;

public class UTTest {

/\*\*

\* @param args the command line arguments

\*/

public static void main(String[] args) {

// TODO code application logic here

Scanner sc=new Scanner(System.in);

/\*

\*1.基本数据类型的测例

\*从命令行接收多组测试数据

\*/

while(sc.hasNext())

{

int data0=sc.nextInt();

String pwd = sc.nextLine();

String result = ManDB.checkaccount(data0,String pwd);

System.out.println("for input : "+data0 + " " + pwd + " result : " + result);

}

/\*

\*2.类类型的测试数据

\*通过类的构造方法构造类的多组实例

\*/

AccountInfo acc1 = new AccountInfo(1,"name","pwd");

AccountInfo acc2 = new AccountInfo(2,"name","pwd");

MessageProbuf.Message mess1 = ManMess.buildmess(sid1,"no","no","no","queryfriend",-1,"server","no");

MessageProbuf.Message mess2 = ManMess.buildmess(sid2,"no","no","no","addfriend",-1,"server","no");

System.out.println("mess1 : " + mess1);

System.out.println("mess1 : " + mess2);

/\*

\*2.测试类的静态方法

\*通过类的构造方法构造类的多组实例

\*/

if(true == ManMess.sendmess(mess1)){

System.out.println("mess1 send ok");

}

else{

System.out.println("mess1 send error");

}

//....................

}

}

在编写软件的主要模块代码的时候，只需要将需要测试的相关函数或类对象抽取出来放到测试类文件夹里，然后从命令行接收测试数据或直接在程序中构造测试对象，通过查看输出的结果与预先设定好的结果进行对比得出测试结论。

此外，在服务器端，为了测试mysql数据库的连接状况，查看自定义数据库的表格的数据信息是否符合预期设计的状态，编写了以下java文件特定用于服务器数据库的测试：

JDtest.java:

import java.sql.\*;

public class JDtest {

// JDBC driver name and database URL

static final String JDBC\_DRIVER = "com.mysql.cj.jdbc.Driver";

static final String DB\_URL = "jdbc:mysql://localhost/MyfoxDB";

// Database credentials -- 数据库名和密码

static final String USER = "root";

static final String PASS = "\*\*\*\*\*\*\*";

public static void main(String[] args) {

Connection conn = null;

Statement stmt = null;

try{

// Register JDBC driver

Class.forName("com.mysql.cj.jdbc.Driver");

// Open a connection

System.out.println("Connecting to database...");

conn = DriverManager.getConnection(DB\_URL,USER,PASS);

// Execute some queries

System.out.println("Creating statement...");

stmt = conn.createStatement();

String sql;

sql = "SELECT \* FROM accountinfo";

ResultSet rs = stmt.executeQuery(sql);

// Extract data from result set

while(rs.next()){

//Retrieve by column name

int accnum = rs.getInt("accnum");

String accname = rs.getString("accname");

//Display values

System.out.println("accnum: " + accnum);

System.out.println(", accname: " + accname);

}

// Clean-up environment

rs.close();

stmt.close();

conn.close();

}catch(SQLException se){

se.printStackTrace();

}catch(Exception e){

e.printStackTrace();

}finally{

try{

if(stmt!=null)

stmt.close();

}catch(SQLException se2){

}

try{

if(conn!=null)

conn.close();

}catch(SQLException se){

se.printStackTrace();

}

}

}

}

**1.3测试环境和工具**

单元测试运行操作系统：ubuntu1604 64bit上，由于java是跨平台语言，所有的测试理论上可以在windows上运行；

测试使用的pc配置：内存4G，Intel i5处理器，

机械磁盘500G,AMD1G独立显卡

服务器信息：单核，内存1G ， 操作系统ubuntu1604 64bit，固态SSD 40G

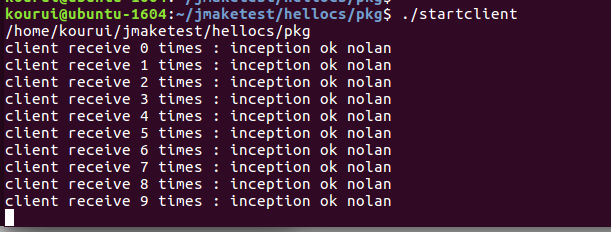
jdk: jdk8u171-linux-64

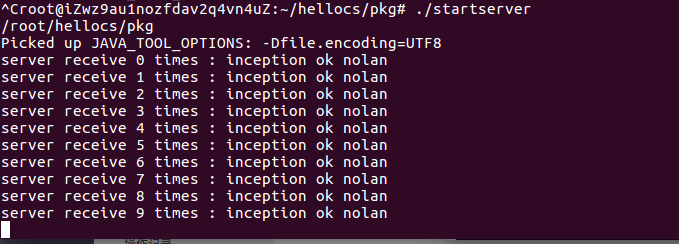
**2.UT测试过程**

**2.1测试ManWeb模块的connect()函数是否能连通上服务器**

|  |  |  |  |
| --- | --- | --- | --- |
| 测试名称 | CS端网络连接测试 | 测试编号 | 001 |
| 测试对象 | ManWeb.connect() | 测试时间 | 20180615 |
| 测试说明 | 测试ManWeb模块的connect()函数是否能连通上服务器  结果在服务器的terminal会有连通提示 | | |
| 前置条件 | 导入netty的jar包到编译和运行脚本中 | | |
| 输入设备 | terminal | 输出设备 | terminal |

|  |  |  |  |
| --- | --- | --- | --- |
| 输入 | 预期输出 | 实际输出 | 结论 |
| ./startclient  ./startserver | CS端各有10条消息，无遗漏 | CS端各有10条消息，无遗漏 | 通过 |





**2.2测试ManMess类的static方法buildmess是否能构造正确的消息对象**

|  |  |  |  |
| --- | --- | --- | --- |
| 测试名称 | 自定义消息格式MessageProbuf | 测试编号 | 002 |
| 测试对象 | ManMess.buildmess(...) | 测试时间 | 20180615 |
| 测试说明 | 测试ManMess类的static方法buildmess是否能构造正确的消息对象 | | |
| 前置条件 | 导入protobuf的jar路径到编译和运行的脚本文件中 | | |
| 输入设备 | terminal | 输出设备 | terminal |

|  |  |  |  |
| --- | --- | --- | --- |
| 输入 | 预期输出 | 实际输出 | 结论 |
| 源程序中直接构造Message对象 | 正确格式的消息 | 正确格式的消息 | 通过 |

Message.proto文件：

message Message {

required int32 senderid = 1;

required string sendername = 2;

required string senderip = 3;

required string senderpwd = 4;

required string sendtext = 5;

required int32 receiverid = 6;

required string receivername = 7;

required string receiverip = 8;

}

ManMess.buildmess(...)函数：

public static MessageProbuf.Message buildmess(int sid,String sn,String sip,String spwd,String stext,int rid,String rn,String rip){

MessageProbuf.Message.Builder builder = MessageProbuf.Message.newBuilder();

builder.setSenderid(sid);

builder.setSendername(sn);

builder.setSenderip(sip);

builder.setSenderpwd(spwd);

builder.setSendtext(stext);

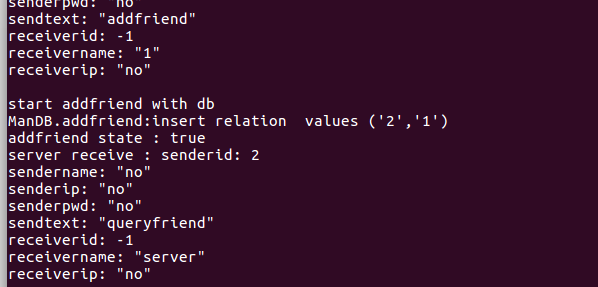
builder.setReceiverid(rid);

builder.setReceivername(rn);

builder.setReceiverip(rip);

return builder.build();

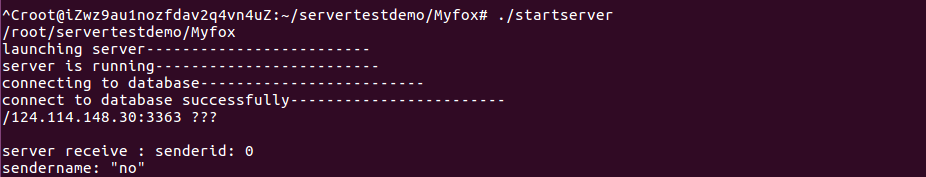
}



**2.3服务器端mysql数据库连通性测试**

|  |  |  |  |
| --- | --- | --- | --- |
| 测试名称 | 连接mysql | 测试编号 | 003 |
| 测试对象 | ManDB.java | 测试时间 | 20180616 |
| 测试说明 | 服务器端设计MyfoxDB数据库，然后尝试连通性，查看表格信息 | | |
| 前置条件 | 写好并执行到数据库的sql脚本文件 | | |
| 输入设备 | terminal | 输出设备 | terminal |

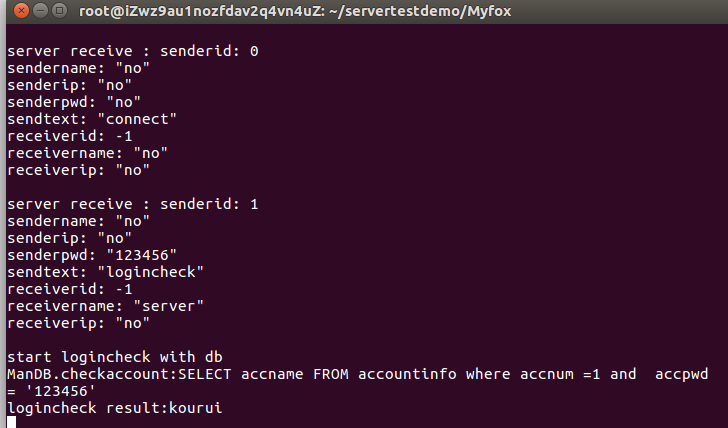
|  |  |  |  |
| --- | --- | --- | --- |
| 输入 | 预期输出 | 实际输出 | 结论 |
| 无 | connect to database successfully | connect to database successfully | 通过 |



**2.4账号登录服务器模块的测试**

|  |  |  |  |
| --- | --- | --- | --- |
| 测试名称 | 服务器验证账号密码 | 测试编号 | 004 |
| 测试对象 | Serverhandler的checkaccount模块 | 测试时间 | 2018/6/16 |
| 测试说明 | 服务器验证账号密码，成功则发送logincheckok消息，失败则发送logincheckerror消息，客户端酌情处理 | | |
| 前置条件 | Message自定义格式正确，构造函数正确 | | |
| 输入设备 | terminal | 输出设备 | terminal |

|  |  |  |  |
| --- | --- | --- | --- |
| 输入 | 预期输出 | 实际输出 | 结论 |
| logincheck消息 | 来自服务器的有效验证信息 | 来自服务器的有效验证信息 | 通过 |

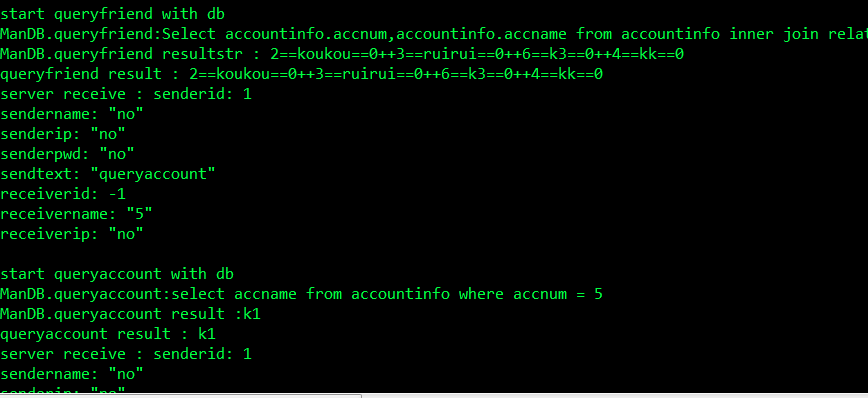


**2.5添加好友的数据库查询以及结果返回**

|  |  |  |  |
| --- | --- | --- | --- |
| 测试名称 | 服务器查询数据库 | 测试编号 | 005 |
| 测试对象 | ManDB.queryfriend() | 测试时间 | 20180616 |
| 测试说明 | 添加好友的数据库查询操作以及结果返回 | | |
| 前置条件 | 数据库信息正确无误，连接数据库正确 | | |
| 输入设备 | termina | 输出设备 | terminal |

|  |  |  |  |
| --- | --- | --- | --- |
| 输入 | 预期输出 | 实际输出 | 结论 |
| 个人的account id | 所有的好友 | 所有的好友 | 通过 |

**服务器日志文件查询数据库部分：**



**二、FT测试**

**3.1 FT测试计划**

**3.1测试目标**

通过图形用户界面，模拟用户的各种操作，以此来测试软件的功能，包括注册、登录，设置个人信息，查询陌生人，添加好友，选择好友进行对话聊天，切换账号，退出登录等各种操作，查看业务逻辑和UI逻辑是否存在矛盾，并及时进行调整。

**3.2测试环境和工具**

两台pc机一台服务器：

（1）ubuntu1604 64bit：

内存4G，Intel i5处理器，

机械磁盘500G,AMD1G独立显卡

（2）windows7/10 64bit ：

内存4G，Intel i5处理器，

机械磁盘500G,AMD1G独立显卡

（3）服务器信息：

单核，内存1G ， 操作系统ubuntu1604 64bit，固态SSD 40G

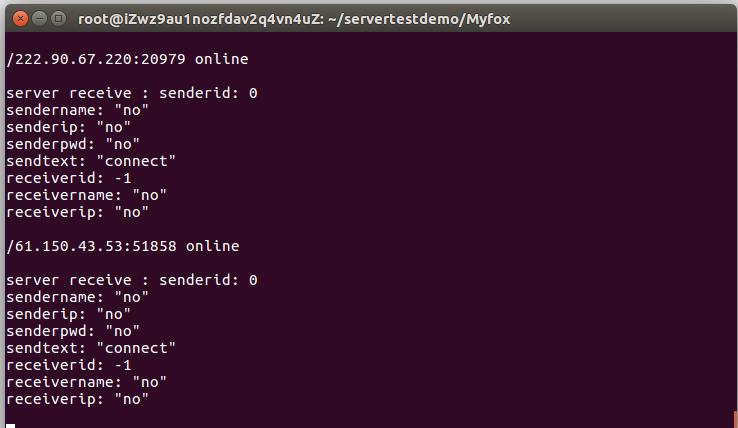
jdk: jdk8u171-linux-64

**3.3FT测试过程**

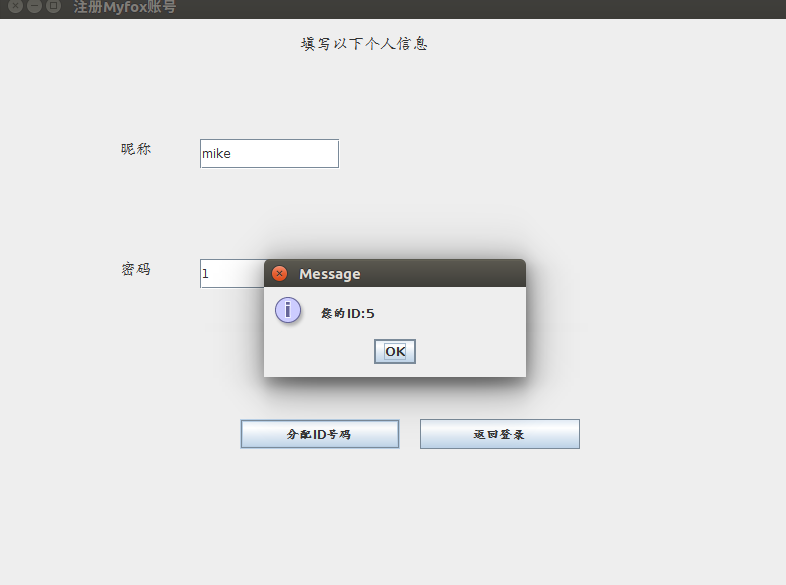
1.欢迎界面：当有一个客户端打开时，会启动一个UI线程和联网线程，如果由于网络原因无法连接网络，欢迎界面的下方会提示“网络连接异常，请检查网络设置”，正常情况下没有这个消息。



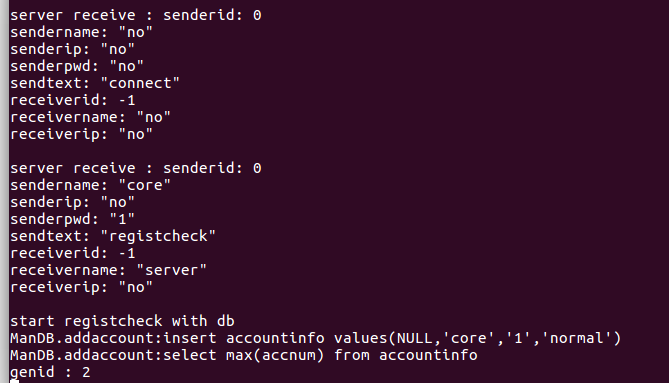
此时服务器端的日志输出如下：说明这两个客户端成功上线



2.注册界面：用户输入自定义的昵称和密码，然后点击分配ID号码按钮，UI界面展示系统分配的唯一的ID号码：

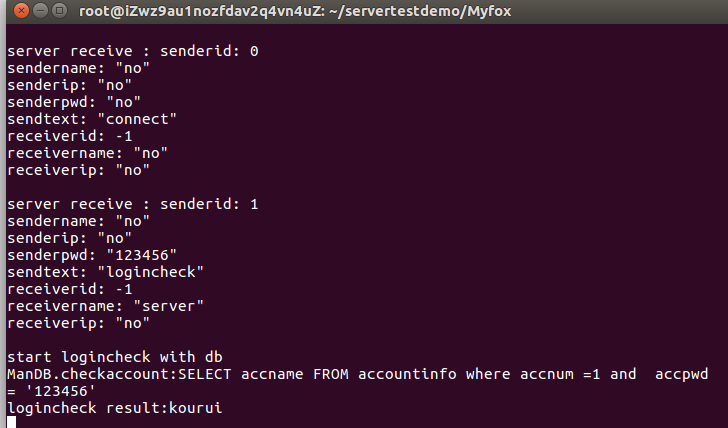


此时服务器端日志输出为客户端发出的registcheck类型的消息,并且服务器生成唯一ID通过registcheckok的消息发送给客户端：



3.登录

服务器端收到来自客户端的checkaccount的消息，进行账号信息的验证：



4.功能选择界面：分为好友管理，聊天室，设置个人信息，切换账号



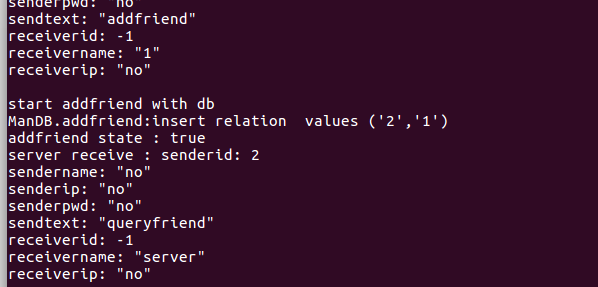
5.好友管理：

左侧列表显示好友信息，包括id,昵称，状态

右侧功能可以查询陌生人，并添加它为好友，添加过程在服务器端通过操纵数据库执行。



服务器端添加好友的日志信息：



6.聊天室选择在线的好友进入聊天界面：



7.聊天对话：

ubuntu聊天界面：



windows聊天界面：

