**2.1引言**

2.1.1 编写目的

根据课程设计的要求，结合当下社会流行情况，我们计划利用数据库和unity做一个简易聊天系统，本系统初步目的是针对一些小规模用户提供聊天沟通服务，由于数据库安装在本地，减少了信息泄露的危险，并且还提供了DIY服务，用户可以根据该设计说明书进行功能的添加，创造出一个属于自己的聊天室。

预计读者：具有一定编程能力的在校大学生，该软件的开发人员。

2.1.2 背景

软件系统名称：NEUIM

项目任务提出者：蒙龙，张翰涛

软件系统开发者：张翰涛，蒙龙，罗康，赖煜华，董子圣，曾志强，金鑫一

软件用户：小范围群体沟通用户

运行该软件的计算站：本机

2.1.3 参考资料

1.《Unity 4.X从入门到精通》

2.《Microsoft SQL Server 2008 Internals》

3.《数据结构》

4.《Thingking in java》

5.《软件工程》

2.1.4 人员安排

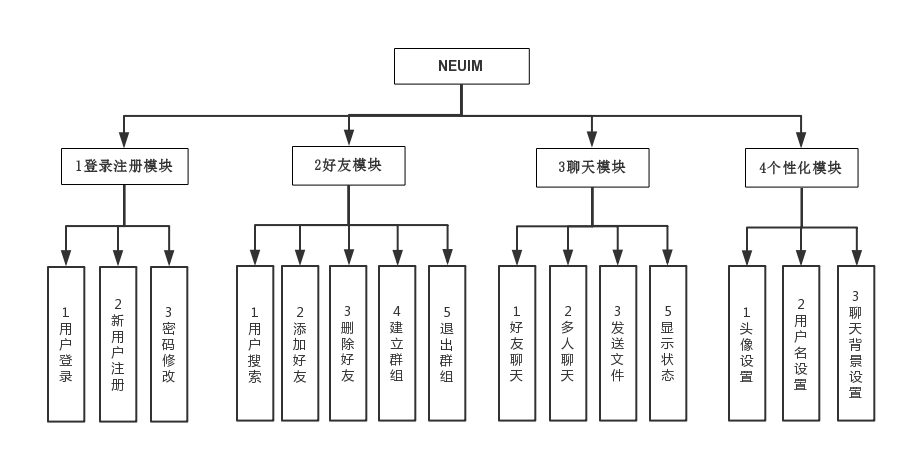
NEUIM设计小组

组长：张翰涛

组员：蒙龙 罗康 赖煜华 董子圣 曾志强 金鑫一

|  |  |  |  |
| --- | --- | --- | --- |
| 工作范围： | 开发环境 | 实现效果 | 人员分配 |
| 后端 | Python  Websocket  sqlight3 | 实现数据结构定义，数据库的连接，与前端沟通 | 张翰韬  董子圣 |
| 服务器 | 云服务器环境 | 租用服务器，部署基本环境 | 赖煜华 |
| 前端 | Unity  C# | UI制作  与服务器端沟通 | 蒙龙 |
| 网络接口部分 | 前后端交换数据约定设计  Json格式编写 | 通过网络协议利用云服务器实现前端和后端信息的交换 | 罗康  曾志强  金鑫一 |

**2.2 系统功能设计**



2.2.1 登录注册模块

注册：

用户在第一次使用“NEUIM”的时候首先需要进行注册，在注册时需要输入唯一邮箱，在客户端中首先完成基本的输入合法性的判断，例如密码，用户名，邮箱格式等输入是否合法。用户输入完成后，客户端将输入打包成Json文件发送给服务器，服务器判断该邮箱是否已经合法进行返回，注册成功后返回账号。在开发工作基本完成且时间的充裕的前提下可以考虑在之后的版本迭代中加入邮箱验证码验证的功能。

登录：

用户在客户端输入账号和密码进行登录，客户端将输入的账号和密码打包成登录请求发送给服务器，服务器通过查找数据库确定用户账号密码是否正确并返回给客户端。若登录成功，则返回包含好友等初始化信息的文件用于初始化应用。

修改密码：

用户可以通过输入注册时的邮箱作为验证，邮箱正确后进行密码重置，可以考虑在之后的版本迭代中添加邮箱验证码的功能。

2.2.2 好友模块

用户搜索：

用户可以通过输入账号来对其他用户获取群聊进行搜索，客户端将搜索内容打包成搜索请求发送给服务器，服务器通过查询数据库返回相关信息，客户端进行显示。

添加好友：

用户在搜索界面搜索到用户和群组之后可以发送添加好友或申请好友的请求。服务器将请求消息发送给对应的用户或群组管理人员，该请求的实现方式可以参考消息转发的实现原理。

删除好友：

用户可以在好友界面删除对应的好友，删除好友之后不能在打开对应的消息窗口。服务器将该好友从信息从数据库中删除，并将删除消息转发给被删除用户。

建立群聊：

用户可以建立群聊，创建者作为群聊的管理者，管理员负责对其他加入群聊申请进行处理。可以考虑在后续的迭代中添加更多的群聊功能，例如禁言，设置多个管理员等。

退出群聊：

用户可以在好友界面退出群聊，删除好友之后不能在打开对应的消息窗口。服务器将该群聊从信息从数据库中删除，并将退出消息转发给群聊管理员。管理人员退出群聊后对应的群聊也将注销。

2.2.3 聊天模块

好友聊天：

用户在好友界面通过双击好友打开对应的聊天界面，可以发送文字和表情，可以考虑在未来的版本迭代中添加文件图片等发送。客户端将相关的信息打包发送给服务器，当目标用户在线时立即进行转发，当用户不在线时进行暂存，当用户上线后立即进行转发。客户端收到消息之后显示为未读消息。

多人聊天：

多人聊天与好友聊天的功能类似。用户在好友界面通过双击群聊打开对应的聊天界面，可以发送文字和表情，可以考虑在未来的版本迭代中添加文件图片等发送。

发送文件：

考虑在开发时间充裕的情况下在未来的版本中添加文件发送的功能，发送的文档可以暂存在服务器中并可以进行多次下载，下载后的文件可以保存在本地。

显示好友状态：

在好友列表中显示好友当前是否在线，可以考虑在未来的迭代版本中添加更多的状态，例如忙碌，请勿打扰等。当用户登录时，服务器返回的消息中包含好友的状态信息，等好友的状态改变时服务器向客户端转发进行状态更新。

2.2.4 个性化模块

头像设置：

用户可以通过点击自己的头像进行头像的设置，可以考虑在前期的版本中让用户在已有的头像中进行选择设置，在未来的版本中进行自定义图片上传。

用户名设置：

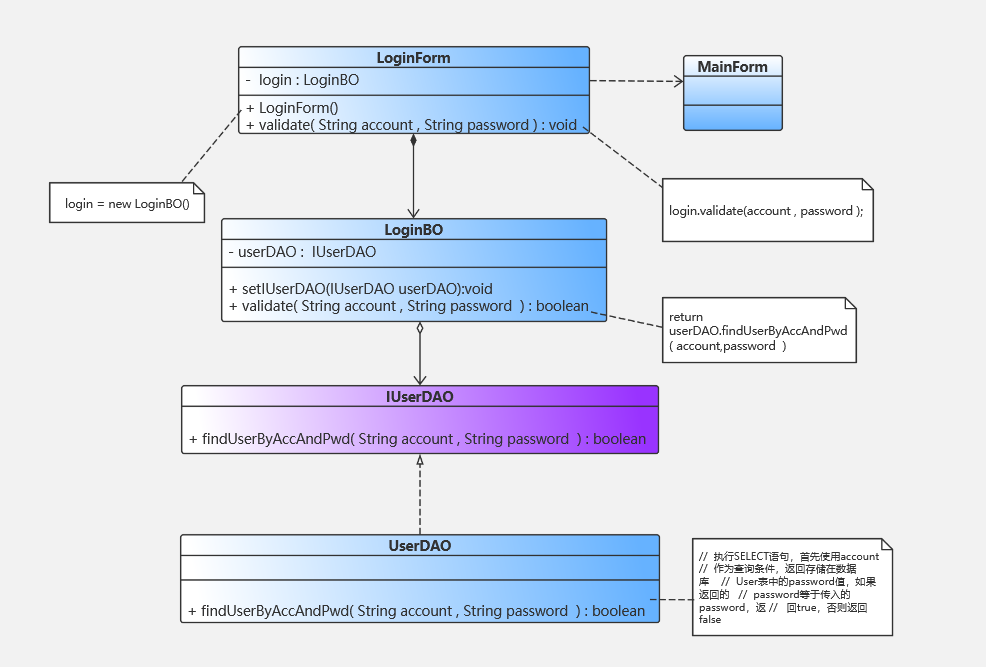
用户可以通过自定义设置用户名，该用户名可以在聊天框，用户搜索中显示等。

聊天背景设置：

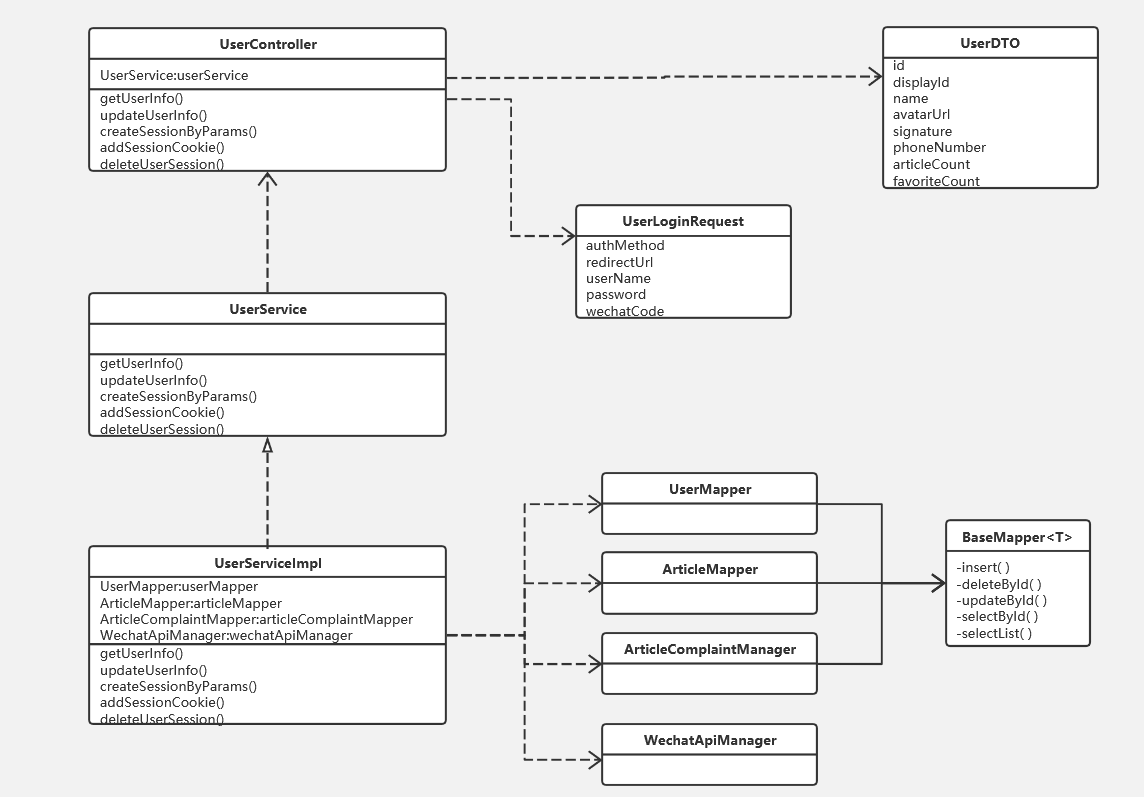
用户可以通过自定义设置聊天背景图片，可以考虑在前期的版本中让用户在已有的背景图中进行选择设置，在未来的版本中进行自定义图片上传。

**2.3 类设计**

2.3.1登录模块类图



2.3.2 发送消息模块



**2.4 数据库设计**

系统中数据库包括如下各表如表2-1所示

表2-1系统数据表

|  |  |  |
| --- | --- | --- |
| 序号 | 表名(英文） | 说明 |
| 1 | User | 记录用户信息 |
| 2 | Friend | 记录用户好友 |
| 3 | Group | 记录用户所在群组 |
| 4 | Message | 记录用户所发信息 |

（1）用户个体信息存储表（User）

用来存储用户基本信息，如表2-2所示

表2-2 用户信息存储表

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 属性(英文） | 类型 | 主键 | 外键 | 空值 | 属性(中文） |
| User\_account | int | √ | √ | 否 | 记录用户账号 |
| User\_pass | int |  |  | 否 | 记录用户密码 |
| User\_image | int |  |  | 否 | 记录用户头像 |
| User\_name | string |  |  | 否 | 记录用户昵称 |
| User\_mail | string |  |  | 否 | 记录用户邮箱 |

（2）用户好友信息存储表(Friend)

用来存储用户好友信息，如表2-3所示

表2-3 用户好友信息存储表

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 属性（英文）） | 类型 | 主键 | 外键 | 空值 | 属性(中文) |
| Friend\_1 | int | √ | √ | 否 | 账号1 |
| Friend\_2 | int | √ | √ | 否 | 账号2 |

（3）用户群组信息存储表(Group）

用来存储用户群组信息，如表2-4所示

表2-4 用户群组信息存储表

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 属性（英文）） | 类型 | 主键 | 外键 | 空值 | 属性(中文) |
| Group\_name | string | √ |  | 否 | 群名称 |
| Group\_user | int |  | √ | 否 | 群成员 |

（4）用户消息存储表(Message）

用来存储用户所发送的消息，如表2-5所示

表2-5 用户消息存储表

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 属性（英文）） | 类型 | 主键 | 外键 | 空值 | 属性(中文) |
| Message\_id | int | √ |  | 否 | 消息id |
| Message\_u1 | int |  | √ | 否 | 消息发送者 |
| Message\_u2 | int |  | √ | 否 | 消息接受者 |
| Message\_group\_mess | bool |  |  | 否 | 是否为群组信息 |
| Message\_groupid | int |  |  | 是 | 群组id |
| Message\_receive | sool |  |  | 否 | 是否接收 |
| Message\_mess | string |  |  | 是 | 消息内容 |
| Message\_time | string |  |  | 否 | 消息发送时间 |

**2.5 接口过程设计**

（1）登录注册模块每个文件的主要功能IPO表如表2-6所示

表2-6 登录注册模块IPO表

|  |  |  |  |
| --- | --- | --- | --- |
| 子模块名称 | I（输入） | P（处理） | O（输出） |
| 注册 | 输出邮箱和密码 | 在数据库创建用户 | 账号创建成功 |
| 登录 | 输入账号和密码 | 在数据库检索是否含有该用户 | 登录 |
| 修改密码 | 输入修改后的密码 | 在数据库中将该用户的密码进行修改 | 密码修改成功 |

（2）好友模块每个文件的主要功能IPO表，如表2-7所示

表2-7 好友模块IPO表

|  |  |  |  |
| --- | --- | --- | --- |
| 子模块名称 | I（输入） | P（处理） | O（输出） |
| 用户搜索 | 用户id | 在数据库中搜索该账号 | 显示出该账号是否存在 |
| 添加好友 | 用户id | 在数据库中搜索该账号，并将其添加到对应账号的好友表中 | 添加好友成功 |
| 删除好友 | 点击删除好友 | 在数据库中搜索该用户，并在其对用用户表中删除好友 | 删除成功 |
| 建立群聊 | 点击创建群聊，输入群聊名称 | 在数据库，创建群聊表 | 创建成功 |
| 退出群聊 | 点击退出群聊 | 在数据库中删除相应的表，并将对应用户中的群聊信息一并删除 | 删除成功 |

（3）聊天模块每个文件的主要功能IPO表，如表2-8所示

表2-8聊天模块IPO表

|  |  |  |  |
| --- | --- | --- | --- |
| 子模块名称 | I（输入） | P（处理） | O（输出） |
| 好友聊天 | 要发送的消息 | 数据库中接收用户接收到一条消息 | 在接收用户中显示 |
| 多人聊天 | 要发送的消息 | 将消息存储在，群中所有用户的消息表上 | 群中所有用户接收到消息 |
| 发送文件 | 要发送的文件 | 数据库中接收用户的对应消息表中存储改文件 | 接收用户接收文件 |
| 显示好友状态 | 单击用户 | 在数据库中读出该用户状态 | 显示该用户的在线情况 |

（4）个性化模块每个文件的主要功能IPO表，如表2-9所示

表2-9个性化模块IPO表

|  |  |  |  |
| --- | --- | --- | --- |
| 子模块名称 | I（输入） | P（处理） | O（输出） |
| 头像设置 | 更换头像和所要更换的照片 | 将叫图片存储到用户头像中 | 主界面显示新换的头像 |
| 用户名设置 | 更改用户名和所要修改的名称 | 将该名称存储到该用户的表中 | 主界面显示该用户的新名称 |
| 聊天背景设置 | 更改聊天背景和所要修改的图片 | 将图片存到该用户聊天表中 | 聊天背景显示为新更换的照片 |

**2.6界面设计**

**（1）**界面转换关系如下 ：

进入登录界面后，如果没有账号可以点击注册按钮进行账号的注册，点击登录后，进入主页可以根据主页的导航进入到各个子模块，每个子模块文件之间的跳转与包含关系如图2-10所示：

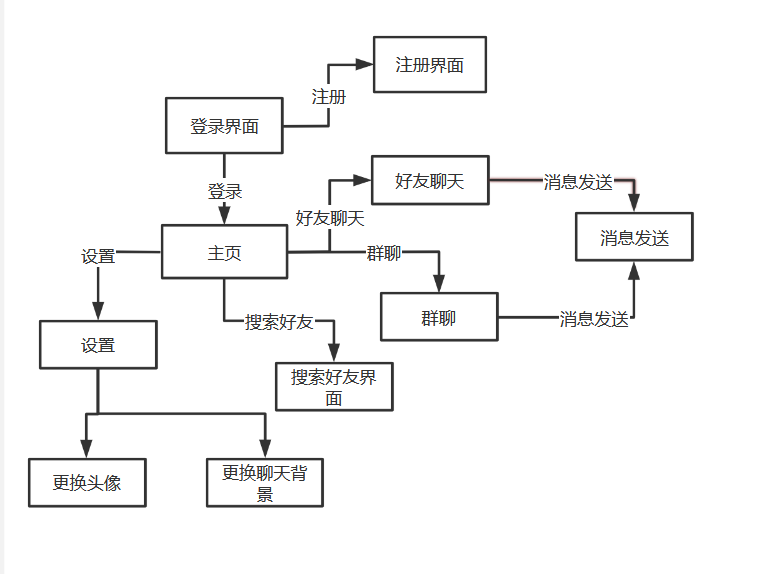
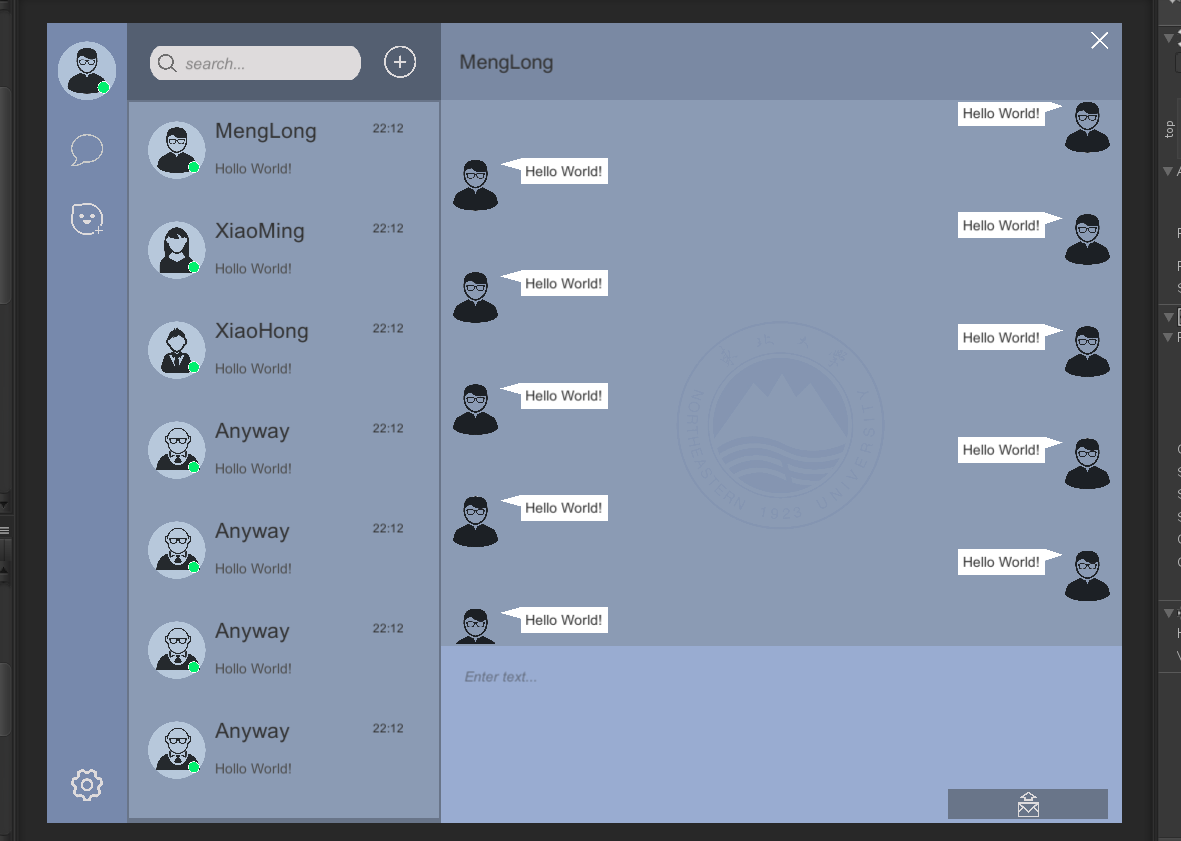


图2-10 页面跳转模块

（2）界面设计结果示例如下：

系统采用多文档窗体结构，如图2-11所示，在主窗体上分为菜单栏、通用工具栏、导航栏、操作工具栏和工作窗口区五个部分。不同喜好风格的用户可以根据自己的使用风格使用这个系统。菜单和通用工具栏同普通系统一样，提供了系统的全部操作功能的接口；导航栏使用了非常直观的树视图结构，整个功能一幕了然，在树视图中包含部门列表和小组列表，查看、维护部门和小组信息非常方便。



登录界面设计

该界面仿照QQ登录时的界面设计，简约大方让人一目了然，同时还有忘记密码和账号注册选项，并配上了具有东大特色的图片



**2.7其他设计**

2.7.1代码设计

“NEUIM”主要包括前端和后端两个部分，其中前端为用户客户端部分，后端为云服务器数据库等部分。客户端中包含软件的用户交互逻辑，发送请求，界面显示，UI设计显示，动画特效等。后端包括云服务器，数据保存，消息转发，响应请求等。

2.7.2 数据结构设计

Json格式的设计，将消息，用户账号，用户密码，接受者账号一并合成一个json字段进行解析。

**2.8设计小结**

这次软件工程大作业采用组的形式开发一个软件，这对于我们中的大部分来说都是第一次进行软件的开发，在开始确定要设计什么软件是就讨论了很久，在接下来的几天里我们讨论编程语言和框架的设计，人员分工。虽然刚开始大家懂的都不是很多，但是我们小组成员经常聚在一起学习，编程，讨论如何去解决这个问题，对于这次大作业不仅考验我们的编程能力，更加考验我们的团队协作能力，在前端和后端的通信之间我们出了一些问题，对于websocket这个东西我们都不是很了解，刚开始都不知道怎么上手，在查阅了好多教程之后我们才知道可以利用http通信，用传输json文件来进行前后端的交流，数据库的设计也是让我们每个人都很头疼，在开发的过程中我们多次因为数据库结构的问题而被迫修改代码，随着越来越多功能的加入，数据库中的表也在不断地扩大，最后也就是在前后端的整合上也是问题频出，可能是交流不是很好的问题，在整合的过程中发现接口对接是无法输出想要的内容，为此也是对代码进行了一顿猛改，好在最后前后端的对接如愿完成了，软件也最终开发完成，在这次开发经历中，可能我们大家都是第一次开发软件，在这次开发过程中也是走了一些弯路，但好在最后都一一解决了，我们也学习到了很多东西，对于软件的设计，前期的沟通很重要，这直接影响了后期的开发速度，因此在对软件进行开发过程中前期的沟通一定要做好。当然我们也很期待进行下一个软件的开发。