单位代码

学 号 39211118

分 类 号 曾斌

****

毕业设计(论文)

一款Windows Phone平台下聚会应用的设计与实现

|  |  |
| --- | --- |
| 院（系）名称 | 软件学院 |
| 专业名称 | 软件工程 |
| 学生姓名 | 曾斌 |
| 指导教师 | 谭火彬 |

2013年 6 月

北京航空航天大学

**本科生毕业设计（论文）任务书**

Ⅰ、毕业设计（论文）题目：

一款Windows Phone平台下聚会应用的设计与实现

Ⅱ、毕业设计（论文）使用的原始资料（数据）及设计技术要求：

Ⅲ、毕业设计（论文）工作内容：

Ⅳ、主要参考资料：

软 件 学院 软件工程 专业 班

学生

毕业设计（论文）时间： 年 月 日至 年 月 日

答辩时间： 年 月 日

成 绩：

指导教师：

兼职教师或答疑教师（并指出所负责部分）：

系（教研室） 主任（签字）：

注：任务书应该附在已完成的毕业设计（论文）的首页。

**本人声明**

我声明，本论文及其研究工作是由本人在导师指导下独立完成的，在完成论文时所利用的一切资料均已在参考文献中列出。

作者：

签字：

时间： 2013 年 6 月

<论文题目>

作 者：

指导教师：

摘 要

关键词：

<论文英文题目>

Author :

Tutor :

Abstract

Keywords:

# 

目 录

1 绪论 1

1.1 课题背景与意义 1

1.2 国内外研究现状 1

1.4 课题研究内容 1

1.5 论文组织结构 1

2 相关原理与技术 3

3 系统概述及需求分析 ………3

4 系统总体设计 4

5 系统详细设计与实现 5

5.1 XXXX…………………………………………………………………………………………….5

5.2 XXXX…………………………………………………………………………………………….5

5.3 XXXX…………………………………………………………………………………………….5

6 系统测试评估 6

6.1 系统测试 6

6.2 系统评估 6

6.3 本章小结 6

总结与展望 7

总结…… 7

展望…… 7

参考文献 8

致谢 9

# **绪论**

## **课题****背景与意义**

鉴于我们已经经过了长期的软件工程理论知识的学习， 我们现在更需要的是将理论应用于实践当中。那么软件工程的理论知识是如何在IT行业前沿得到应用的呢？ 本论文将对如何设计和实现一款Windows Phone平台下的聚会应用进行讨论。在此过程中， 我们将用到大量软件工程、系统分析与设计及设计模式等相关知识， 进而帮助大家学习这些知识。

聚会游戏是指朋友、同事、家人或公司单位等在平时聚会、开派对以及过年过节酒宴上为了活跃气氛、加深彼此感情、提高凝聚力等目的而进行的集体互动游戏。然而，这些游戏一般都需要各种小道具的支持，而更多的时候我们就因缺少这样的工具而扫兴而归。那么我们何不考虑把现在人手一部的智能手机作为我们游戏的工具呢？

作为一个聚会上的手机游戏， 我们希望为用户的聚会创建更良好的气氛，对于一些很有趣但一般只在聚会时才玩的游戏，我们推出我们的联网游戏平台，使用户在平时也可享受这些游戏的乐趣。

## **国内外研究现状**

移动通信和互联网成为当今世界发展最快、市场潜力最大、前景最诱人的两大业务，它们的增长速度都是任何预测家未曾预料到的，所以移动互联网可以预见将会创造经济神话。移动互联网的优势决定其用户数量庞大，截至2012年9月底，全球移动互联网用户已达15亿。

当今， 主要有三大移动平台：

1) 苹果公司的iOS平台

自从2007 iPhone的发布，软件应用呈现爆炸式发展， 之后多款iPhone和iPad的发布，使iOS持续占有着强大的市场份额。 可以说，iOS走到了他的顶峰， 然而， 随着史蒂夫·乔布斯的趋势， iOS还能否保持这么高的市场占有率很让人怀疑。

2) 谷歌公司的Android平台

步苹果之后，谷歌公司推出了Android系统。 由于此系统本身是开源的， 因此深受广大开发者和移动厂商青睐。同时， Android的产品多种多样并实惠， 大部分普通群众都希望能够拥有这么一款设备来进行办公、游戏和交流。

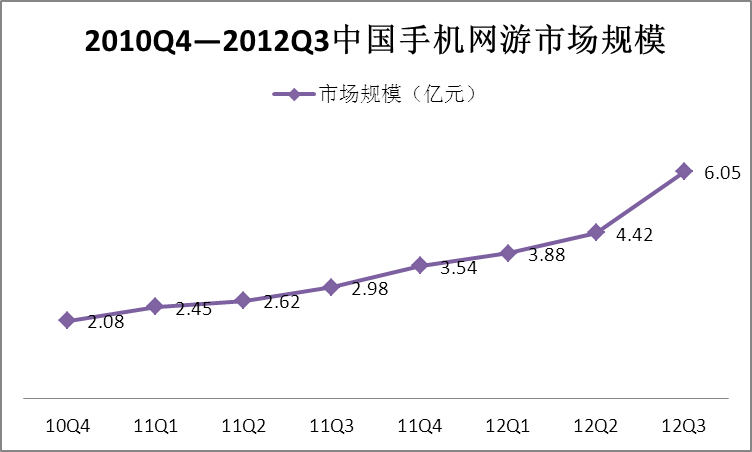
3) 微软公司的Windows Phone平台

Windows Phone作为一个新兴的移动平台，用户基数虽然较少，但却在快速增长中。随着Window Phone 8的发布， 相信会有更多人喜欢这个平台。此平台开发方便、简单，风格独特，功能强大，有着很不错的发展前景。

移动互联网时代的十大模式包括：

* + 移动社交将成客户数字化生存的平台
  + 移动广告将是移动互联网的主要盈利来源
  + 手机游戏将成为娱乐化先锋
  + 手机电视将成为时尚人士新宠
  + 移动电子阅读填补狭缝时间
  + 移动定位服务提供个性化信息
  + 手机搜索将成为移动互联网发展的助推器
  + 手机内容共享服务将成为客户的黏合剂
  + 移动支付蕴藏巨大商机
  + 移动电子商务的春天即将到来

根据《2012中国手机网游市场分析报告》，在手机成为人们生活、娱乐必不可少的工具时，手机游戏得到非常迅速地发展，而手机网络游戏更是飞速发展。



2010Q4-2012Q3中国手机网游市场规模

从上图可以看出，市场规模走势呈现上升的趋势，特别是进入到2012。2012Q3的市场规模达到6.05亿元，环比增长36.8%，同比增长103%。手机网游2012年第三季度的市场规模为6.06亿元，比客户端网游第三季度的116.7亿元相比较而言，相差较大。但是手机网游的发展速度快，增长潜力巨大。

目前，中国手机网游市场的规模目前虽小，但市场规模呈现加速增长趋势，安卓、iPhone、Windows Phone三个手机平台都发展迅速，竞争相对激烈。

在生活中，我们经常会碰到这么一个场景，同学或同事在一起聚会时，他们并不是聊的天花乱坠，而是各自拿着自己的iPhone、Android、Windows Phone在那儿逛社交应用， 这是多可悲的一次聚会啊！我们每个人都有着自己的智能机，却缺乏一个将大家联系起来的平台。在搜索了App Store、Google Play和Windows markrtplace后，我们并不能找到非常符合该需求的应用。

## **课题研究内容**

### 研究目标

本项目意在搭建将参加聚会的朋友们联系起来的平台，包括提供一些本地的聚会热门游戏以及联网的在线游戏。

### 主要任务

深入项目，对项目作进一步的需求分析；

对当前的市场和技术进行调研，完成项目的可行性分析；

完成对该项目的设计；

完成编码工作；

对项目进行测试，保证软件质量；

发布后进行维护和监测。

### 拟采取的技术方案

全程将使用模型驱动开发(MDD, Model-Driven Development)。 采用Visual Paradigm for UML进行UML建模, 终端采用Windows Phone平台API实现， 服务器端采用Node.js语言开发，数据库则采用MySQL数据库实现。

### 预期成果

完成一款优秀的移动手机应用。

## **论文组织结构**

第一章“绪论”，对本论文所阐述的课题进行说明，主要包括课题的背景、国内外研究现状、课题研究内容等。

第二章“需求分析”，详细描述了系统的功能性需求和非功能性需求，同时对界面和运行环境的需求做了明确的解释。

第三章“相关原理与技术，介绍了论文中涉及的关键技术内容。包含服务器开发语言的选择、终端和服务器之间通信协议的选择、数据库管理系统的选择等核心内容的介绍。

第四章“系统总体设计”，从系统架构切入，全面地展现了系统不同组成部分以及他们之间的联系。为后面的详细设计做好充分的准备。

第五章“系统详细与实现”，以系统架构图为标准，对系统的各个子功能模块进行详细深入的分析与设计，主要是对Windows Phone前段布局及自定义控件的实现、使用WebSocket连接Windows Phone终端和服务器、“大话骰”游戏的服务器具体实现以及使用node.js连接数据库等具体内容进行阐述。

第六章“系统的评估测试”，通过对项目进行系统的测试，以保证代码和软件的质量。

# 需求分析

对于普通手机用户而言，他们所希望的游戏应该要是一个交互性强、娱乐性高的手机游戏。考虑到大家在聚会时的场景，选择的游戏要具备很高的互动性。

## 功能性需求

本应用的游戏部分将分为两个部分，一是单机的本地游戏部分，此时游戏的场景应该是一群正在面对面聚会的用户。 他们只需要一台机器，就可以玩上数种聚会游戏。另一部分为联网的在线游戏部分，此时的场景应该是

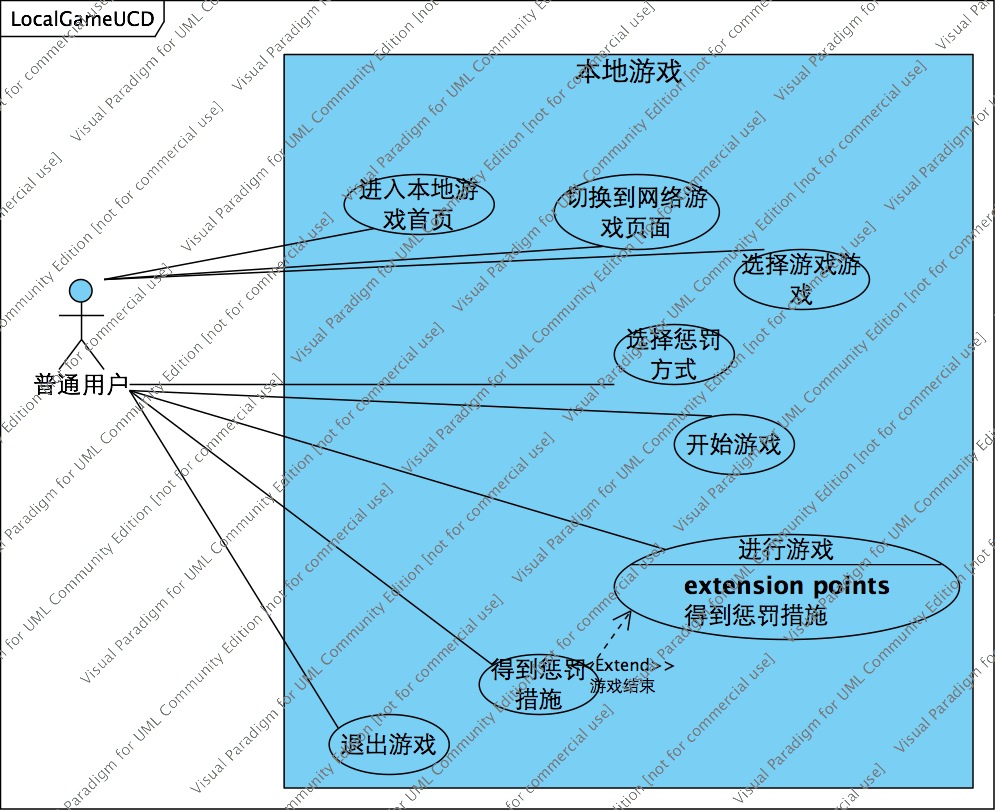


图2.1.1 本地游戏用例图

对于本地游戏， 用户可以完成进入本地游戏首页，切换到网络游戏页面，选择游戏方式，选择惩罚方式，开始游戏，进行游戏，得到推荐的惩罚措施，退出游戏这些活动。

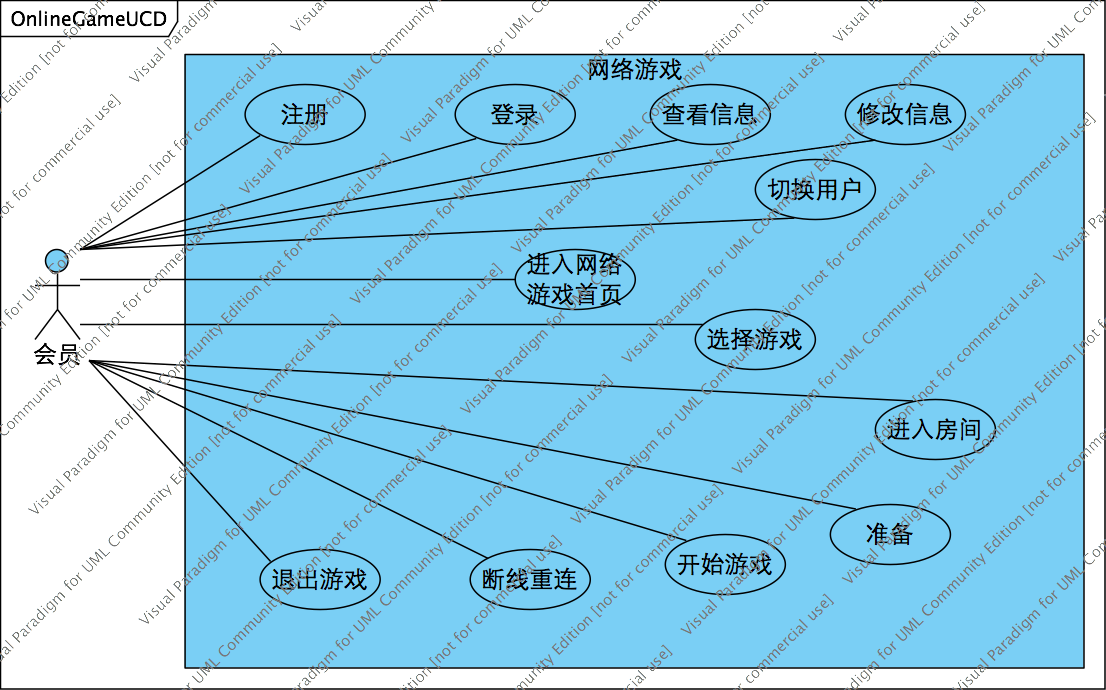


图2.1.2 联网游戏用例图

对于联网游戏方面，用户应该可以完成注册新帐号，登录已有帐号，查看登录的帐号信息，修改信息，切换帐号，进入网络游戏首页，选择游戏方式，创建房间等一系列活动。

## 非功能性需求

### 界面

“聚会太疯狂”应用作为一款Windows Phone平台下的应用，自然要秉承此系统独特的Metro风格界面。

其次，作为游戏应用，背景图片应符合聚会游戏的主题，界面要够绚丽，交互够清晰。

### 系统的可靠性

作为线上产品，只显示系统可以做某些事情是不够的。如果一个系统不能可靠地运行（例如，在加载时，或者在系统故障时，等等），则它就不能满足客户的需要。这里就要对一些问题作好准备：即使硬件出现故障，系统也可以可靠运行吗？复制和故障转移方案是什么？需要手动干预，还是系统可以自动进行故障转移？实现可靠性会对性能造成负面影响吗？实现可靠性的成本有多高？可靠性需要考虑的一些具体的安全性问题：假设攻击者就在外面。如何知道系统用户就是他们所声称的，并只让他们访问经过授权的功能？如何保护我的系统不受攻击？考虑到网络攻击、机器攻击，甚至从您自己的系统内部发起的攻击。

### 系统的扩展性和可维护性

系统的扩展性和可维护性指系统对技术和业务需求变化的支持能力。当技术变化或业务变化时，不可避免将带来系统的改变。移动互联网行业变化之快是难以想象的，任何在当时非常优秀的产品经过一段时间都可能会落后，至少需要一些转变。这时不仅要进行设计实现的修改，甚至要进行产品定义的修改。如何在这种转变过程中花费更小的成本，就需要对系统的扩展性有更高的要求。包括不同用户数量如果增加，如果需要加入语音模式，如果出现添加或转移到别的平台的需求等。

好的软件设计应在系统架构上考虑能以尽量少的代价适应这种变化，常用的技术有面向对象的分析与设计及设计模式。

# 相关原理与技术

## 通信协议

### 关于WebSocket

## 服务器语言

## 数据库

# 系统总体设计

# **系统详细设计与实现**

# **系统测试评估**

## **系统测试**

对于很多同学来说，当他们写完一个程序的所有逻辑代码，点击编译运行，没有发现问题，于是直接将项目上线。可是，上线后没有多久用户在使用该程序的过程中却遇到各式各样的问题，给用户带来了极低的用户体验。这是我们非常不愿意看到的结果，因此，我们必须重视对程序的测试。通过软件测试，我们可以确保软件的质量，为开发提供反馈和保证整个软件开发过程是高质量的。

由于此项目分服务器端和终端两部分的开发，因此测试也分服务器端测试和终端测试

### 服务器端测试

由于服务器端包含了本项目最为核心的逻辑代码，因此测试也应该具体些。包括使用JSLint对代码质量进行检测，用nodeunit对核心代码进行单元测试，和使用cURL对服务器接口进行集成测试。

### 代码质量检测

JavaScript 语言本身是一种弱类型脚本语言，具有相对于 C++ 或 Java 语言更为松散的限制，一切以函数为中心的函数式编程思想也为开发人员提供了更加灵活的语法实现。然而，这种灵活性在带来高效的同时，也成为初学或者经验不足的 JavaScript 开发人员的噩梦。形式各异的代码风格、隐含错误的代码行为，严重影响了整体代码的可读性和稳定性，成为 Web 项目中最为常见问题之一。

### JSLint

因而，我们需要一个有效的 JavaScript 代码质量工具，以便能及时发现并修正 JavaScript 代码中所隐含的问题，保证代码交付质量。JSLint 作为一个灵活有效的 JavaScript 代码质量检测工具，允许使用者指定满足其特定应用开发需求的编码风格约定，使得整个项目的风格统一。

### JSLint的功能

JSLint提供以下功能：

* 检测语法错误：例如大括号“{}”的配对错误。
* 变量定义规范：例如未定义变量的检测。
* 代码格式规范：例如句末分号的缺失。
* 蹩脚语言特性的使用检测：如 eval 和 with 的使用限制。

### 使用JSLint进行测试

1. 安装node-jslint

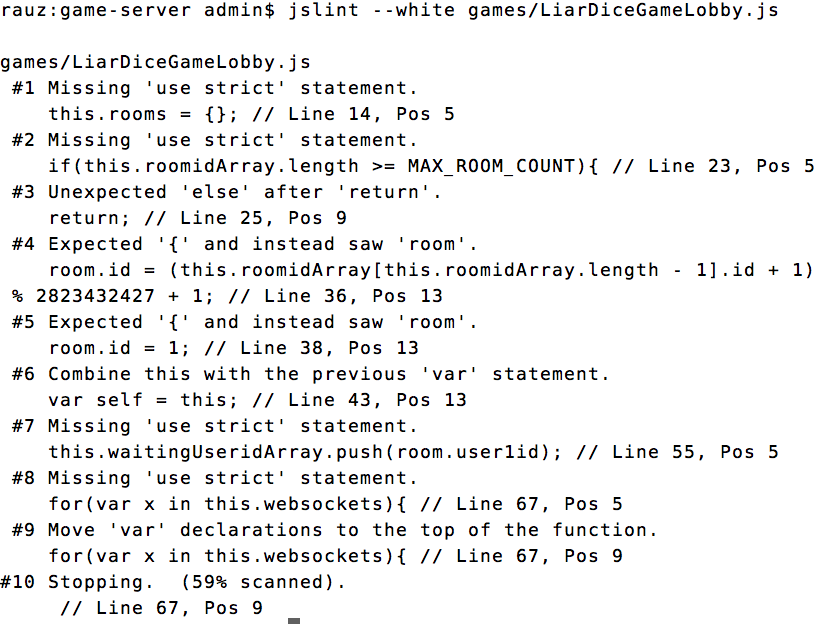


1. 测试

分别在各个目录中运行

$ jslint –white –color \*

得到如下结果：



node-jslint对代码质量要求非常高，我们几乎不可能没有任何警告地通过测试。但是，对于一些无关紧要的警告，我们是可以忽略的。从上图的结果来看，我们并没有非常明显的错误，因此通过。当然，对于有心提高自己代码质量，使其更加美观的同学，可以尽量减少警告条数，但为了一些华而不实的东西花费大量时间却是不可取的。

### 单元测试

上一节我们已经通过了对代码质量的检测，然而要使代码最终顺利正确地得到运行，我们必须对代码地逻辑进行测试。单元测试作为针对程序模块来进行正确性检验的测试工作，自然是不可或缺的步骤。

### nodeunit

虽然node.js提供了自己的assert模块可用于编写一些有用的测试函数，但是真正运行的时候还是非常麻烦，你很难保证所有的测试用例都得到了执行。因此，我们采用nodeunit，它使我们的在编写单元测试代码和运行测试都变得非常简单。

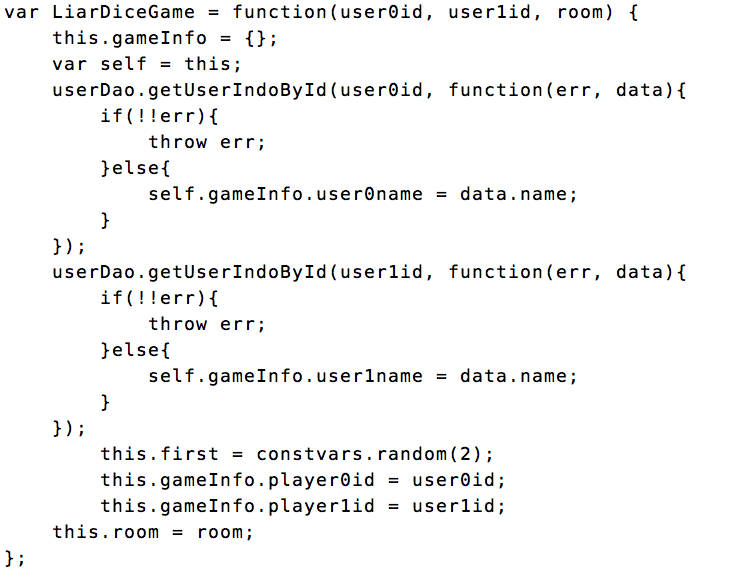
### 使用nodeunit进行单元测试

1. 使用npm安装nodeunit

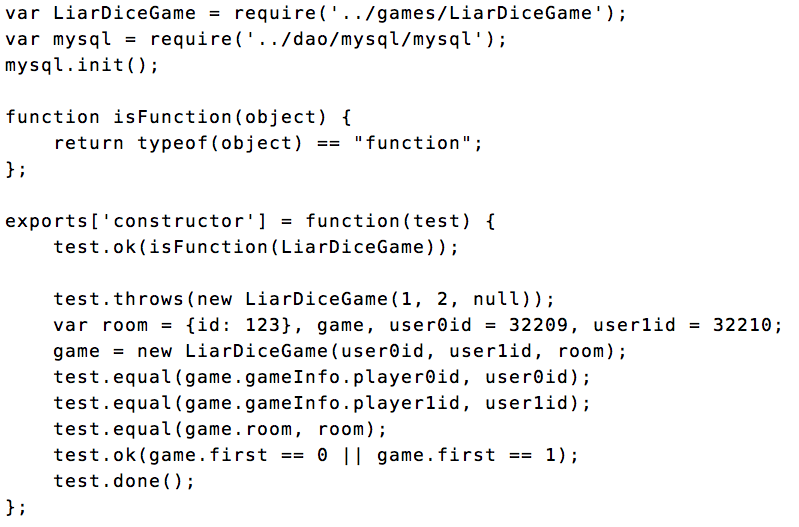


1. 编写单元测试代码

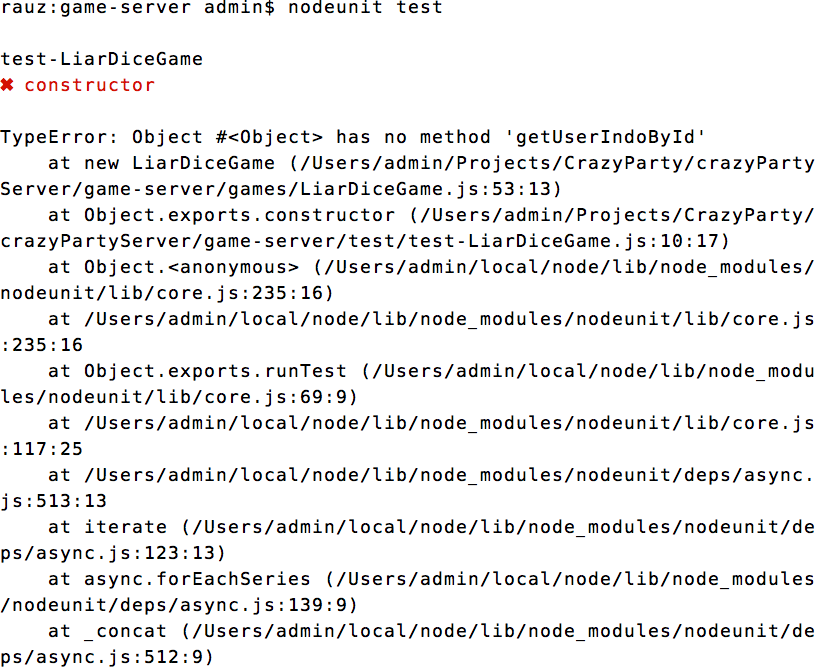
这里，我们只对逻辑稍微复杂的模块和函数编写单元测试。如对大话骰游戏中的LiarDiceGame.js编写, LiarDiceGame的构造函数代码如下：



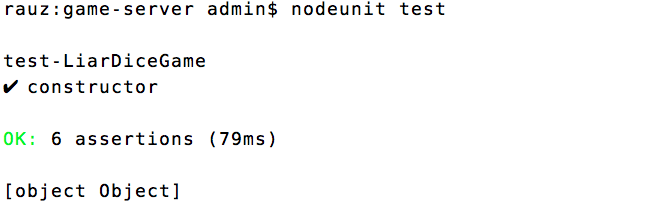
因此我们对此构造函数编写单元测试，代码如下:



运行nodeunit进行测试：



测试结果失败，从输出的内容得知，第53行的“getUserIndoById”有错，改为“getUserInfoById”, 再测， 结果如下：



测试通过！

不过我们好像高兴得太早了，我们并没有添加对LiarDiceGame构造函数中访问数据库部分进行测试，因此这函数并不可靠。事实上，如果我在单元测试中直接添加对game.gameInfo.user0name和game.gameInfo.user1name进行判断，并不会达到预期作用。原因在于，node.js是非阻塞式io调用的， 当数据库调用返回时，测试用例早已经结束。解决方法是在测试中创建一个数据库访问的Mock，这个Mock以参数形式传递到函数内部，并得到运行以替代IO部分，使得函数运行过程中并不是直接访问数据库。这样不仅降低了函数对数据库代码的依赖，也大大提高了单元测试的速度。

### 集成测试

单元测试确保了我们底层代码的正确性，可是当代码组合到一起时，它们还能否正确无误地工作呢？这也就是我们进行集成测试的目的所在。集成测试，将程序模块采用适当的集成策略组装起来，对系统的接口及集成后的功能进行正确性检测的测试工作。其主要目的是检查软件单位之间的接口是否正确，集成测试的对象是已经经过单元测试的模块。

### cURL

CrazyParty的服务器是通过使用http协议为客户机提供服务。对于登陆和注册模块我们可以通过curl进行模拟。

cURL是利用URL语法在命令行方式下工作的文件传输工具。它支持FTP, FTPS, HTTP, HTTPS, GOPHER, TELNET, DICT, FILE 以及 LDAP等协议。

### 使用cURL进行测试

对于用户登陆注册部分，主要有以下测试：

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 序号 | 描述 | 方法 | 参数 | 预期结果 |
| 0001 | 注册新用户 | register | 新的用户名 | 注册成功 |
| 0002 | 注册已有用户 | register | 已有用户名 | 注册失败 |
| 0003 | 使用正确的用户名密码进行登陆 | login | 正确的用户名和密码 | 登陆成功 |
| 0004 | 使用错误的用户名或密码进行登陆 | login | 错误的用户名或密码 | 登陆失败 |

可直接通过curl命令进行测试：



由于服务接收的是POST请求，因此测试时需要加上--data/-d这一参数将数据POST到服务器上。从运行结果中可以看出，

### 终端测试

终端上由于使用VS2010进行开发，因此在代码质量上不会有什么问题。而终端程序中并无太多复杂的逻辑，加上代码中主要是与用户交互的过程，写单元测试也会比较麻烦。因此，终端测试的主要内容是对界面进行测试。

测试用例如下表所示：

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 序号 | 描述 | 页面 | 输入 | 期望结果 |
| 1001 | 测试是否能正常打开应用 | Windows Phone 应用列表页面 | 点击“聚会太疯狂”应用 | 应用能够正常打开， 并进入本地游戏主页 |
| 1011 | 测试游戏在本地游戏页面是否能正常退出 | 本地游戏页面 | 点击Windows Phone的返回按键 | 游戏正常退出 |
| 1012 | 测试是否能选择不同的游戏方式 | 本地游戏首页 | 点击不同的游戏方式 | 选中的游戏方式有一层橙色的边框 |
| 1013 | 测试是否能选择不同的惩罚方式 | 本地游戏首页 | 点击不同的惩罚方式 | 选中的惩罚方式有一层橙色的边框 |
| 1014 | 测试是否能正常开始本地游戏 | 本地游戏首页 | 选择“猜数字”游戏和“真心话大冒险”惩罚方式，点击开始游戏按钮 | 游戏进入“猜数字”游戏页面 |
| 1015 | 测试本地游戏结束后是否自动进入惩罚方式页面 | “猜数字”游戏页面 | 点击显示“结束游戏”按钮 | 页面自动跳转到“真心话大冒险”页面 |
| 1016 | 测试本地游戏页面是否能跳转到联网游戏首页 | 本地游戏页面 | 向左滑动页面，或者点击导航栏的“联网” | 都能正常进入联网游戏首页 |
| 1017 | 测试本地游戏页面是否能跳转到帐号管理首页 | 本地游戏页面 | 向右滑动页面，或者点击导航栏的“帐号” | 都能正常进入帐号管理首页 |
| 1021 | 测试“转瓶子”游戏能否正常进入 | 本地游戏页面 | 选择“转瓶子”游戏， 并点击“开始”按钮 | 页面进入“转瓶子”游戏界面 |
| 1022 | 测试“转瓶子”游戏的人数设置功能 | “转瓶子”游戏界面 | 通过上下滑动数字选择框确定人数， 点击“确定”按钮保存设置 | 数字选择框和“确定”按钮正常工作，桌盘上的分区也随着变化 |
| 1023 | 测试瓶子转动功能 | “转瓶子”游戏页面 | 用手波动瓶子使其转动 | 瓶子能够正常转动，并且瓶子中央正确显示瓶口所指向的数字，瓶子停止转动后自动跳到惩罚方式页面 |
| 1031 | 测试“摇骰子”游戏能否正常进入 | 本地游戏页面 | 选择“摇骰子”游戏， 并点击“开始”按钮 | 页面进入“摇骰子”游戏界面 |
| 1032 | 测试“摇骰子”游戏的骰子数设置功能 | “摇骰子”游戏界面 | 通过上下滑动数字选择框确定骰子数， 点击“确定”按钮保存设置 | 数字选择框和“确定”按钮正常工作，桌盘上的骰子变到相应数目 |
| 1033 | 测试摇动骰子功能 | “摇骰子”游戏页面 | 点击“摇骰子”按钮或者摇动手机 | 骰子的点数在不断发生变化，骰子点数停止变化时自动跳到惩罚方式页面 |
| 1041 | 测试“猜数字”游戏能否正常进入 | 本地游戏页面 | 选择“猜数字”游戏， 并点击“开始”按钮 | 页面进入“猜数字”游戏界面 |
| 1042 | 测试“猜数字”游戏的“显示数字”和“隐藏数字”是否正常工作 | “猜数字”游戏界面 | 先点击“隐藏数字”， 再点击“显示数字” | 点击“隐藏数字”后， 游戏盘上的数字被画布遮挡， “隐藏数字”变成“显示数字”；点击“显示书”，游戏盘上的画布消失， “显示数字”变成“隐藏数字” |
| 1043 | 测试数字选择器能否正常工作 | “猜数字”游戏页面 | 上下滑动数字选择器 | 数字选择器的变化范围时00～99 |
| 1051 | 测试“真心话大冒险”页面是否正常显示 | “真心话大冒险”页面 | 选择“真心话大冒险”作为惩罚方式， 结束游戏 | 页面自动弹出一条真心话或者大冒险， 点击“确定”按钮返回游戏页面。 |
| 1052 | 测试“啤酒转盘”页面是否正常显示 | “啤酒转盘”页面 | 选择“啤酒”作为惩罚方式， 结束游戏 | 页面中显示当前玩家面前的酒杯状态， 点击“确定”按钮返回游戏页面。 |
| 1061 | 测试游戏在帐号管理页面是否能正常退出 | 帐号管理页面 | 点击Windows Phone的返回按键 | 游戏正常退出 |
| 1062 | 测试帐号管理页面是否能跳转到联网游戏首页 | 帐号管理页面 | 向右滑动页面，或者点击导航栏的“联网” | 都能正常进入联网游戏首页 |
| 1063 | 测试帐号管理页面是否能跳转到本地游戏首页 | 帐号管理页面 | 向左滑动页面，或者点击导航栏的“本地” | 都能正常进入本地游戏首页 |
| 1064 | 测试登陆功能 | 帐号管理页面 | 进入帐号管理首页， 输入用户名密码， 点击“登录”按钮 | 当用户名密码正确时， 用户登录成功， 进入用户信息页面；  当用户名密码不正确时，用户登录失败， 提示用户名或密码错误 |
| 1065 | 测试注册功能 | 注册页面 | 进入帐号管理首页， 点击“注册”按钮， 在注册页面中填入用户名和密码， 点击“确定”按钮 | 页面正常显示， 当信息填入正确时，提示注册成功并跳转到用户信息页面； 否则提示注册失败的原因 |
| 1071 | 测试游戏在联网游戏页面是否能正常退出 | 联网游戏首页 | 点击Windows Phone的返回按键 | 游戏正常退出 |
| 1072 | 测试联网游戏首页是否能跳转到帐号管理首页 | 联网游戏页面 | 向右滑动页面，或者点击导航栏的“帐号” | 都能正常进入帐号管理首页 |
| 1073 | 测试联网游戏页面是否能跳转到本地游戏首页 | 联网游戏页面 | 向左滑动页面，或者点击导航栏的“本地” | 都能正常进入本地游戏首页 |
| 1081 | 测试“大话骰”游戏能否正常进入 | 联网游戏页面 | 选择“大话骰”游戏， 并点击“开始”按钮 | 页面进入“大话骰”游戏界面 |
| 1082 | 测试“大话骰”游戏的逻辑是否正确 | “大话骰”游戏界面 | 多次进行游戏 | 游戏逻辑正确 |
| 1083 | 测试摇动骰子功能 | “大话骰”游戏页面 | 点击“摇骰子”按钮或者摇动手机 | 骰子的点数在不断发生变化，骰子点数停止变化时自动跳到惩罚方式页面 |

## **系统评估**

## **本章小结**

# **总结与展望**

## **总结**

## **展望**

# **参考文献**

# **致谢**