**3D**仿真钓鱼游戏

实

现

分

析

报

告

****

课 程： 软件工程导论

题 目： 3**D仿真钓鱼**游戏**实现**

专 业： 软件工程

班 级： **软工1701及1702**

小组成员：**郑鸿棣31701298(组长)**

**徐余浩 31701297**

**陈瑜安31701279**

目录

1. 引言 3

1.1编写目的 3

1.2项目背景 3

1.3参考资料 3

二．钓鱼游戏设计说明 4

2.1程序清单 4

2.2小组代码规范 4

2.3储存分配 5

2.4注释设计 5

三．项目计划展示 5

四．实现展示 6

五．软件测试 10

5.1测试计划 10

5.2单元测试用例 12

5.3 集成测试 13

5.4确认测试 13

5.5白盒测试 14

# 引言

## 1.1编写目的

为那些想体验一下钓鱼或者偶尔钓一下鱼的人，提供一款游戏来缓解念想。

## 1.2项目背景

钓鱼，作为一种悠闲自得的运动项目，在现在快节奏的生活下，便变成了难以实现的奢望。因此，我们希望为那些想体验一下钓鱼或者偶尔钓一下鱼的人，提供一款3D仿真钓鱼游戏，让他们在移动设备上体验钓鱼的乐趣。

## 1.3参考资料

1 张海藩，牟永敏. 软件工程导论. 北京：清华大学出版社，2013

2 刘国柱. Unity3D/2D 游戏开发从0到1. 北京：电子工艺出版社，2018

# 二．钓鱼游戏设计说明

## 2.1程序清单

你做的一个完整的工程之后，所有你编写的程序代码全部列写出来，包括主函数，调用函数，子函数等。

## 2.2小组代码规范

1.明确方法功能，精确（而不是近似）地实现方法设计。如果一个功能将在多处实现，即使只有两行代码，也应该编写方法实现。

说明：

虽然为仅用一两行就可完成的功能去编方法好象没有必要，但用方法可使功能明确化，增加程序可读性，亦可方便维护、测试。

2.应明确规定对接口方法参数的合法性检查应由方法的调用者负责还是由接口方法本身负责，缺省是由方法调用者负责。

说明：

对于模块间接口方法的参数的合法性检查这一问题，往往有两个极端现象，即：要么是调用者和被调用者对参数均不作合法性检查，结果就遗漏了合法性检查这一必要的处理过程，造成问题隐患；要么就是调用者和被调用者均对参数进行合法性检查，这种情况虽不会造成问题，但产生了冗余代码，降低了效率。

3.明确类的功能，精确（而不是近似）地实现类的设计。一个类仅实现一组相近的功能。说明：划分类的时候，应该尽量把逻辑处理、数据和显示分离，实现类功能的单一性。

示例：

数据类不能包含数据处理的逻辑。通信类不能包含显示处理的逻辑。

4.所有的数据类必须重载toString() 方法，返回该类有意义的内容。说明：父类如果实现了比较合理的toString() ， 子类可以继承不必再重写。

5.数据库操作、IO操作等需要使用结束close()的对象必须在try -catch-finally 的finally中close()。

6.异常捕获后，如果不对该异常进行处理，则应该记录日志（针对后台）。

说明：若有特殊原因必须用注释加以说明。

7.自己抛出的异常必须要填写详细的描述信息。

说明：便于问题定位。

8. 在程序中使用异常处理还是使用错误返回码处理，根据是否有利于程序结构来确定，并且异常和错误码不应该混合使用，推荐使用异常。说明：一个系统或者模块应该统一规划异常类型和返回码的含义。但是不能用异常来做一般流程处理的方式，不要过多地使用异常，异常的处理效率比条件分支低，而且异常的跳转流程难以预测。

9.避免使用不易理解的数字，用有意义的标识来替代。涉及物理状态或者含有物理意义的常量，不应直接使用数字，必须用有意义的静态变量来代替。

10.数组声明的时候使用 int[] index ，而不要使用 int index[] 。

 11.异常捕获尽量不要直接 catch (Exception ex) ，应该把异常细分处理。

12.不要使用难懂的技巧性很高的语句，除非很有必要时。说明：高技巧语句不等于高效率的程序，实际上程序的效率关键在于算法。

## 2.3储存分配

对于钓到的鱼的质量以及使用的渔具和鱼饵储存在用户的手机上，若有排行榜功能，我们则会建立一个服务器和数据库，使得数据储存在上面。

## 2.4注释设计

地图：用unity3d的terrain组件，点击paint texture按钮，给“白模”的地形系统绘制特定的地形贴图纹理，再通过抬高地形，往里面注入水体，来达到制作成河流、海水、湖泊等地形。最后用trees按钮，添加一些树木以及花草在上面。用碰撞体来创建一个“空气墙”，不让用户走出我们设定的范围。

人物：导入characeters角色包，选择第一人称，把主摄像机当作角色的眼睛。

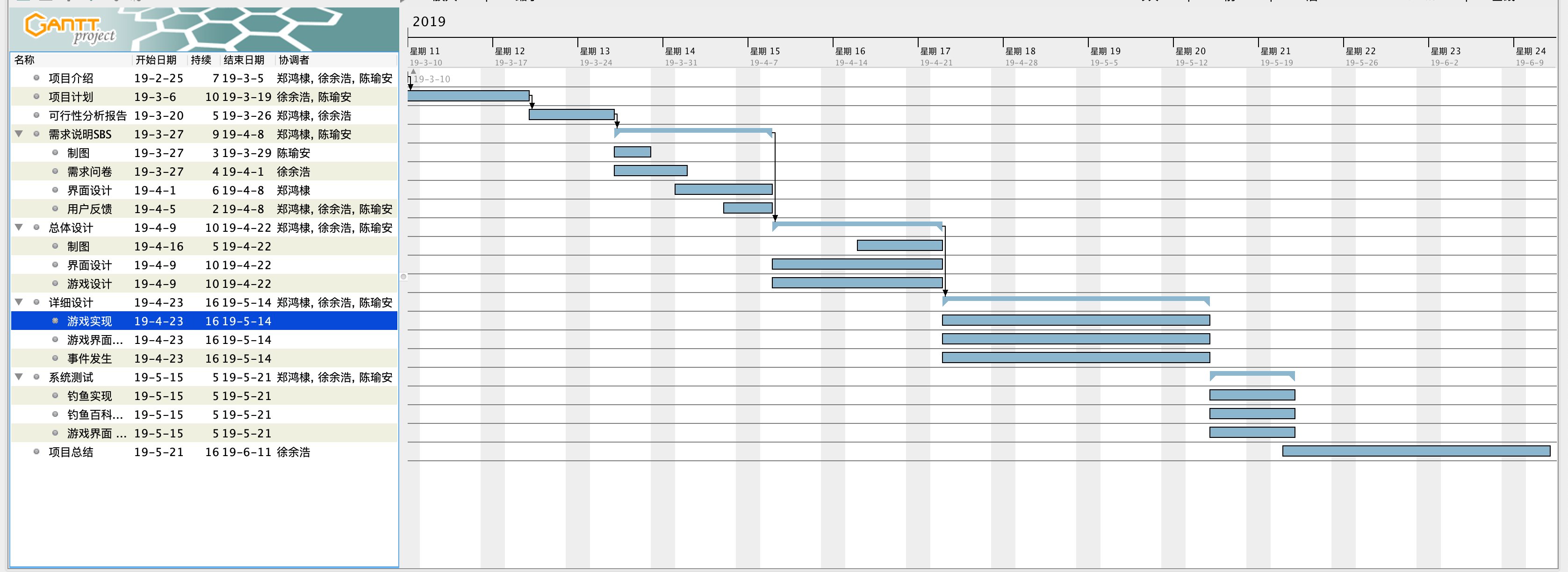
鱼竿、鱼饵、鱼类：在网上搜索各自的包，将他们导入，再自己建模来补充一些

钓鱼功能：用手机的平衡模拟器模拟甩杆，用点击来控制力度使其在最佳范围内。

UI界面：创建UGUI界面。再用canvas创建画布，用eventsystem（事件管理）、panel（整体移动）、text（文字）、image（图像）、button（按钮，使其动起来）控件完善

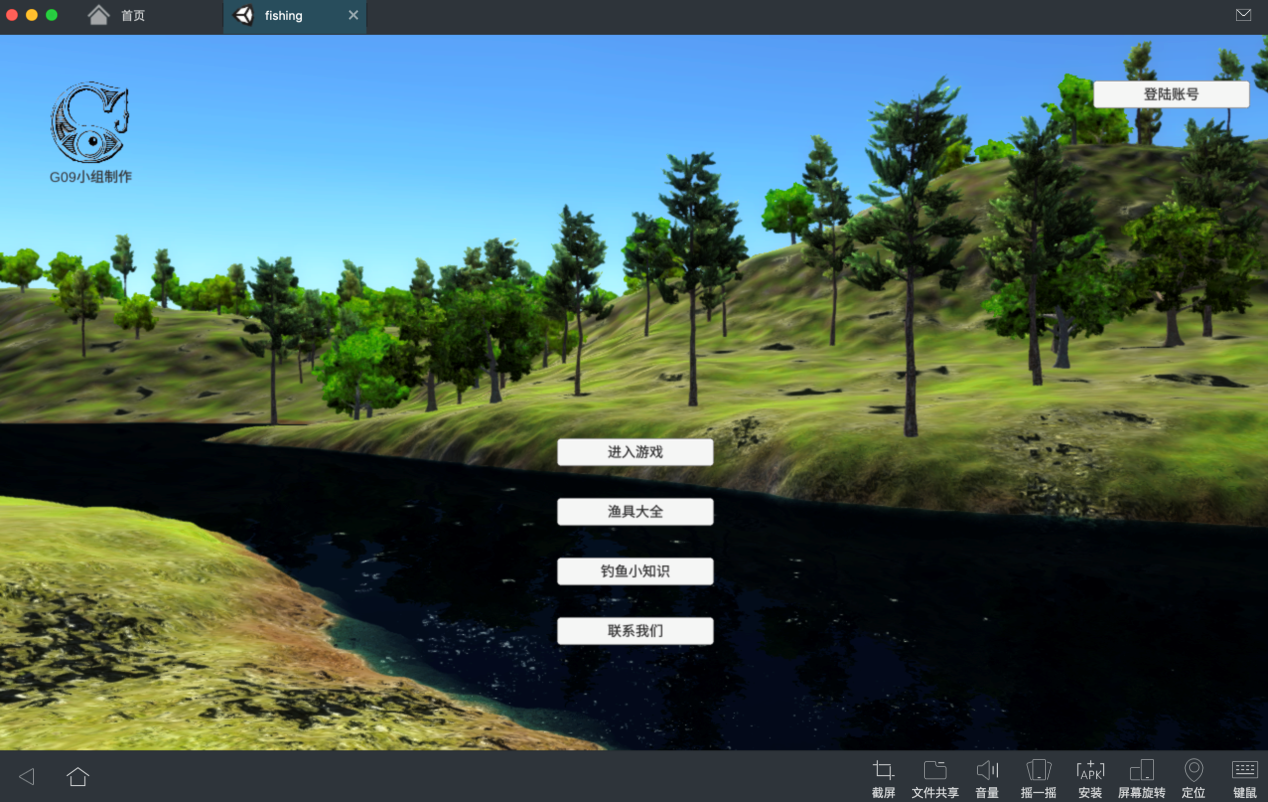
# 三．项目计划展示

预计3个月完成该项目

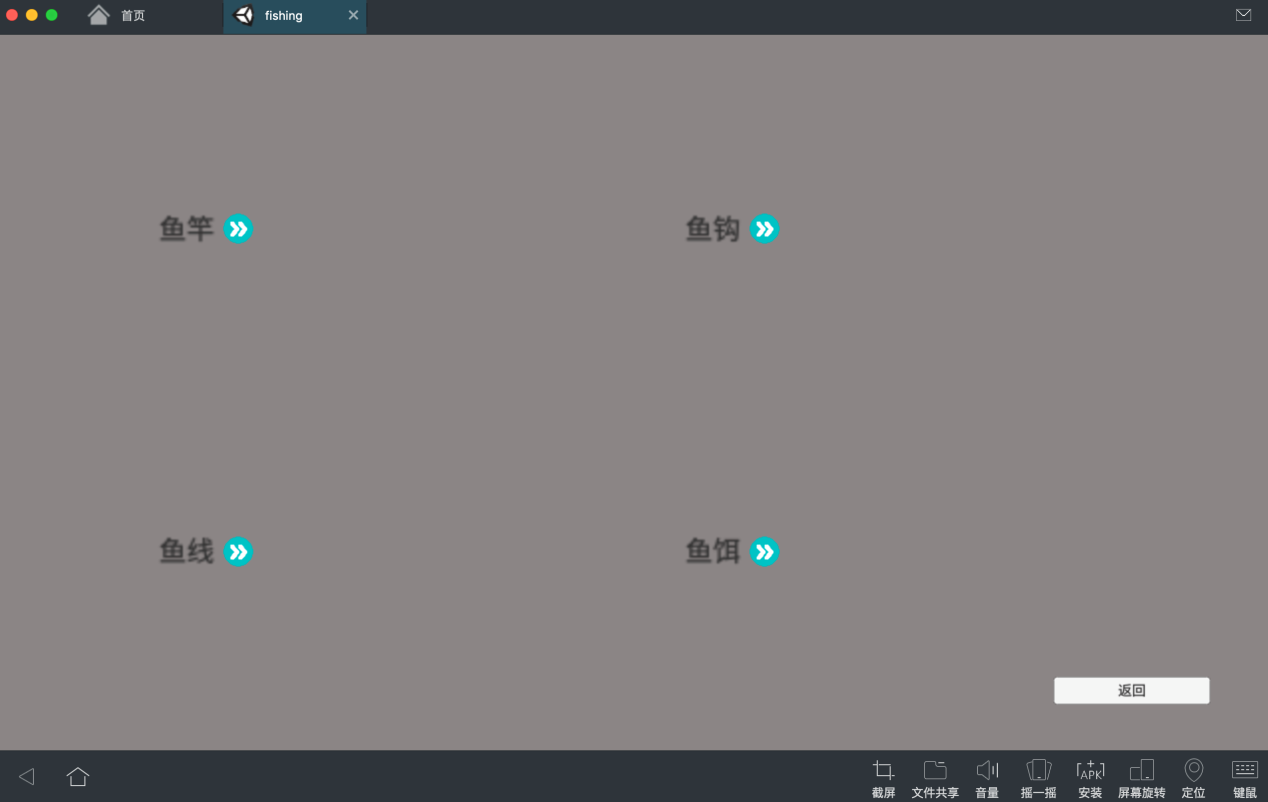


# 四．实现展示

初始界面



购买渔具

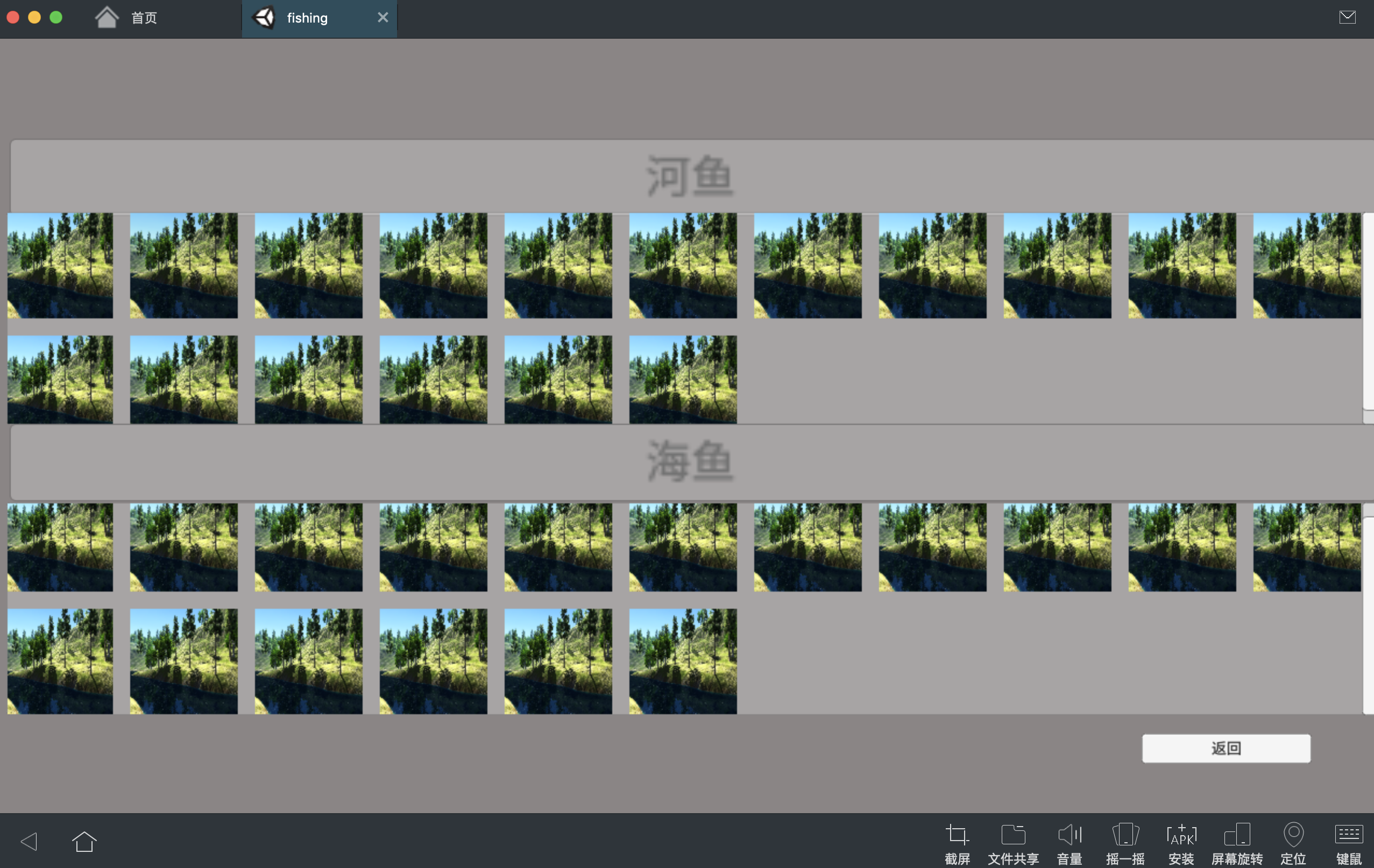




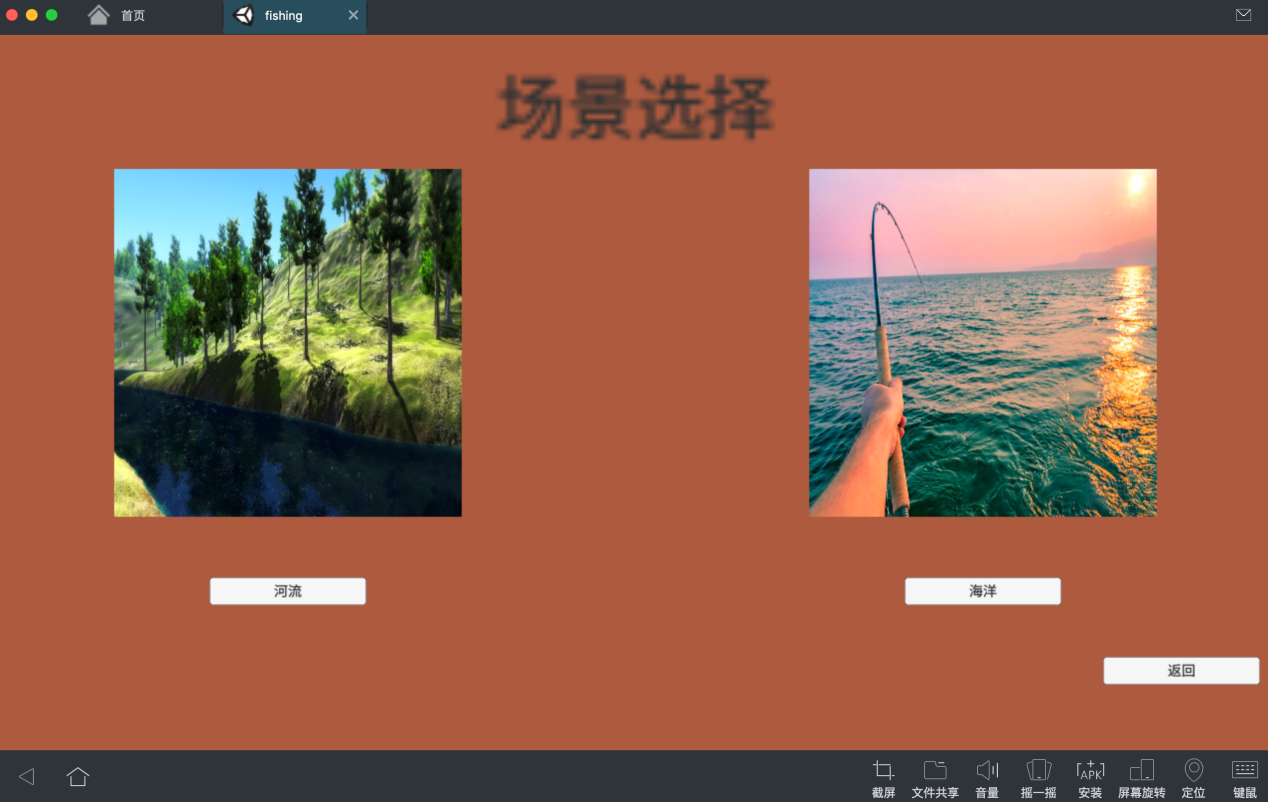
反馈



鱼类百科



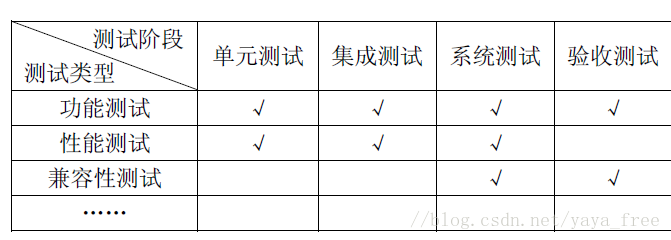
钓鱼场所（初始/进阶）

****



# 五．软件测试

## 5.1测试计划



#### 5.1.1 测试策略

1. 总体策略

软件系统在进行单元、集成、确认、系统、安装、验收测试时，发现一级错误（大于等于1）、二级错误（大于等于2）暂停测试返回开发。软件系统经过单元、集成、确认、系统、安装、验收测试，分别达到单元、集成、确认、系统、安装、验收测试停止标准。软件系统通过验收测试，并已得出验收测试结论。软件项目需暂停以进行调整时，测试应随之暂停，并备份暂停点数据。软件项目在其开发生命周期内出现重大估算，进度偏差，需暂停或终止时，测试应随之暂停或终止，并备份暂停或终止点数据

1. 测试范围

1. 响应时间

我把“响应时间”的概念确定为“对请求作出响应所需要的时间”，把响应时间作`为用户视角的软件性能的主要体现。响应时间划分为“呈现时间”和“系统响应时间”两个部分。

2. 并发用户数

我把“并发用户数”与“同时在线数”进行区别对待，我的“并发用户数”的标准是：并发用户数取决于测试对象的目标业务场景，因此，在确定这个“并发用户数”前，必须（必要）先对用户的业务进行分解、分析出典型的业务场景（也就是用户最常使用、最关注的业务操作），然后基于场景采用某些方法（有多种计算并发用户数的数学模型与公式）获得“并发用户数”。

这样做的原因是：假设一个应用系统、最高峰有500人同时在线、但这500人却不是并发用户数、因为假设在一个时间点上、有50%的人在填写复杂的表格（填写表格动作对服务器没有任何负担、只有在“提交”动作的时候才会对服务器系统构成压力）、有40%的人在不停的从一个页面跳转到另外一个页面（不停发出请求与回应、产生服务器压力）、还有10%的人挂在线上，没有任何操作在发呆：）（没有对服务器构成压力的动作）。因此只有那40%的人真正对服务器产生了压力，从这里例子可以看出、并发用户数关心的是不但是业务并发用户数、还取决于业务逻辑、业务场景。因此我们需要本文第六部分性能测试文档4、5、6。

3. 吞吐量

我把吞吐量定义为“单位时间内系统处理的客户请求的数量”，直接体现软件系统的性能承载能力，对于交互式应用系统来说、吞吐量反映的是服务器承受的压力、在容量规划的测试中、吞吐量是一个重要指标、它不但反映在中间件、数据库上、更加体现在硬件上。我们在以下方面利用这个指标：

（1）用来协助设计性能测试场景，衡量性能测试是否达到了预计的设计目标、比如J2EE应用系统的连接池、数据库事务发生频率、事务发生次数。

（2） 用来协助分析性能瓶颈、参照本文第二部分总的RBI方法。

4. 性能计数器

性能计数器式描述服务器或操作系统性能的一些数据指标、例如对WINDOW来说使用内存数、CPU使用率、进程时间等都是常见的计数器。

对于性能计数器这个指标来说、需要考虑到的不但有硬件计数器、web服务器计数器、Weblogic服务器计数器、Servlet性能计数器、EJB2的性能计数器、JSF性能计数器、JMS性能计数器。找到这些指标是使用性能计数器的第一步、关键是找到性能瓶颈、确定系统阀值、提供优化建议才是性能计数器使用的关键。性能计数器复杂而繁多、与代码上下文环境、系统配置情况、系统架构、开发方式、使用到的规范实现、工具、类库版本都有紧密的联系、在此不作赘述。

5. 思考时间

我把思考时间确定为“休眠时间”。从业务系统的角度来说，这个时间指的是用户在惊醒操作时、每个请求之间的时间间隔、从自动化测试的角度来说、要真实的测试模拟用户操作、就必须在测试脚本中让各个操作之间等待一段时间、体现在脚本上就是在操作之间放置一个Think的函数，体现为脚本中两个请求语句之间的间隔时间、不同的测试工具提供了不同的函数或方法来实现思考时间、比如HP LoadRuner和IBM Rational Performance Tester的方式就完全不同。

1. 风险分析

存在风险：由于测试组成员之前都没有过软件测试的经验，只有一些基础的理论知识。所以测试准备做得不是很充分。可能会有部分测试用时过长，或者某个人的测试工作不能按时完成。会造成对整体时间以及测试进度的影响。

风险处理：必要的简化测试内容，尽量简化的达到测试目的。完成任务的人员给予尚未解决问题的组员以帮助，尽量短时间完成各自的任务。

## 5.2单元测试用例



|  |
| --- |
| 测试模块：鱼类百科 |
| 标识符：1 |
| 测试项：用户在点击鱼类百科中的按钮，跳出该鱼类的栖息地等等信息。 |
| 测试环境：网易MUMU 安卓手机 |
| 前置条件：APP相关数据库等其他运行环境正常 |
| 操作步骤：  1.点击鱼类百科中不同鱼的按钮 |
| 测试结果：没有出现异常。 |

|  |
| --- |
| 测试模块：钓鱼游戏运行 |
| 测试项：用户在进入游戏之后的人物移动，钓鱼操作等 |
| 测试环境：网易MUMU 安卓手机 |
| 前置条件：APP相关数据库等其他运行环境正常 |
| 操作步骤：  1.进入APP点击开始游戏，在场景里进行移动，钓鱼等操作。 |
| 测试结果：没有出现异常。 |

|  |
| --- |
| 测试模块：背包功能 |
| 标识符：1 |
| 测试项：用户在点击游戏界面的背包按钮，看是否会出现道具以及一些之前捕捉到的鱼类。 |
| 测试环境：网易MUMU 安卓手机 |
| 前置条件：APP相关数据库等其他运行环境正常 |
| 操作步骤：  1.点击背包中各种渔具 |
| 测试结果：没有出现异常。 |

## 5.3 集成测试

1.先对主控制模块进行测试，将存根程序代替所有直接附属与主控制模块的模块。

2.每次用一个实际模块代换一个存根程序。

3.在结合进一个模块的同时进行测试

4.在不断进行回归测试

5.重复以上步骤。

未发现各模块借口间存在的问题

## 5.4确认测试

Alpha测试

时间：2018.5.25

地点：寝室604

人员：陈云

测试结果：在三位开发人员的“指导下”，该同学对3D仿真钓鱼APP的已开发功能用了一遍，同时仔细审核了一下自己原先的需求定义。认为该App与自己理想需求存在一定的差异。有如下差异：1、界面设计不够理想 2、玩法比较简单 3、仍然存在部分功能模块问题

在操作方面，该同学认为还算比较简单，难度不大。

本软件已交付给陈云初步使用，他对基本功能的实现表示认同，但也提出了一些建议，比如进一步的细化钓鱼的具体玩法，带来更好的游戏体验。这些功能有待进一步的完善与开发。

Beta测试

## 5.5白盒测试

一：安装、卸载测试

1、直接安装：启动和功能点

可以

2、覆盖安装：新版本是否能继承老版本用户数据（比如用户资料、收藏、浏览历史等）

因为还没有新版本，此项待测

3、卸载安装：是否能正常卸载，卸载成功后是否可以正常安装使用

可以

二：升级测试

1、升级覆盖安装

2、下载后手动覆盖安装

3、跨版本升级、升级后可以正常使用。

未升级，此阶段待测

三：功能测试

1、功能点

可以正常观看鱼类百科，进行钓鱼等功能。

四：各种网络状态下进行的测试

还未开启云服务，暂时不需要网络，即任何情况都可使用

五： 中断性测试

如突然来电、短信弹出、闹钟、QQ聊天信息、微信、低电量等是app能否正常使用

可以