高级语言程序设计

实验报告

南开大学 计算机大类

姓名 史文天

学号 2110688

班级

2022年5月7日

目录

[高级语言程序设计大作业实验报告 4](#_Toc102846209)

[**一、** **作业题目** 4](#_Toc102846210)

[**二、** **开发软件** 4](#_Toc102846211)

[**三、** **课题要求** 4](#_Toc102846212)

[**四、** **主要流程** 4](#_Toc102846213)

[**1． 整体流程** 4](#_Toc102846214)

[**2． 主窗口包含成员** 5](#_Toc102846215)

[**3． 封装类介绍** 5](#_Toc102846218)

[**4． 单元测试** 6](#_Toc102846220)

[**五、** **单元测试** 6](#_Toc102846221)

[**六、** **收获** 6](#_Toc102846222)

[**1． 高级for循环** 6](#_Toc102846223)

[**2． 简单AI实现** 7](#_Toc102846224)

[**3． 游戏优化** 7](#_Toc102846229)

[**4． 重力效果模拟** 7](#_Toc102846234)

高级语言程序设计大作业实验报告

1. **作业题目**

制作一个2D横版休闲游戏，实现用炸弹炸鱼、与NPC玩球的玩法。

1. **开发软件**

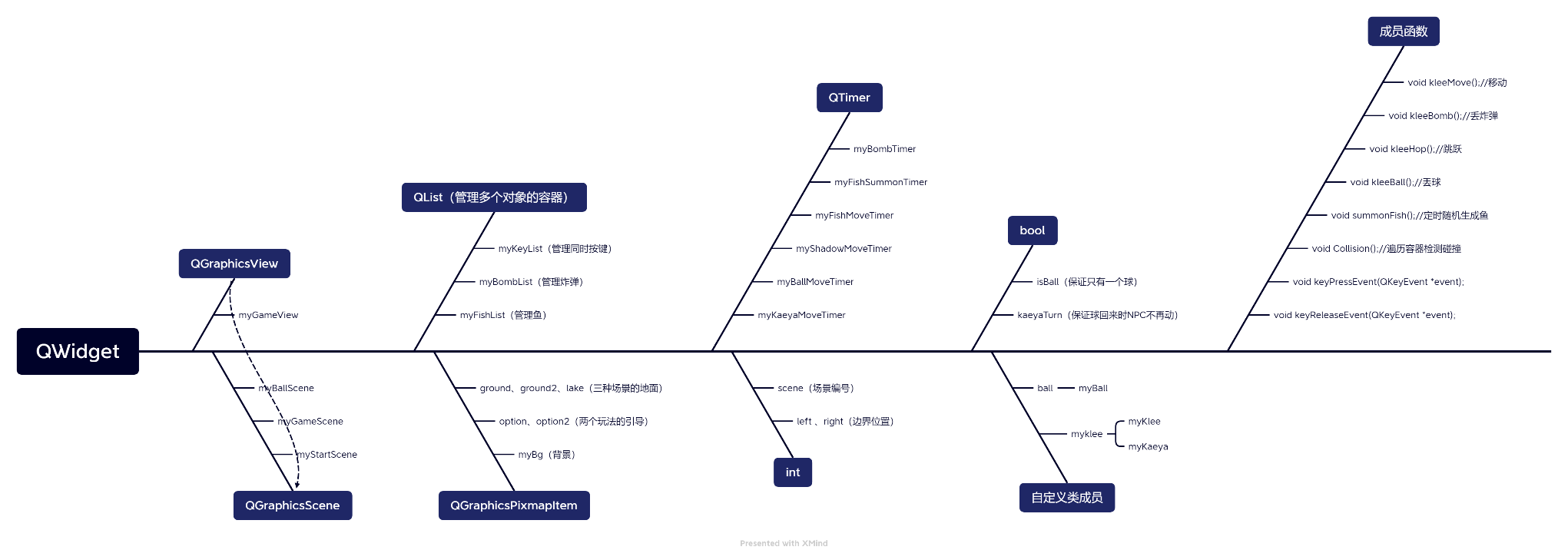
Qt Creator 4.3.0

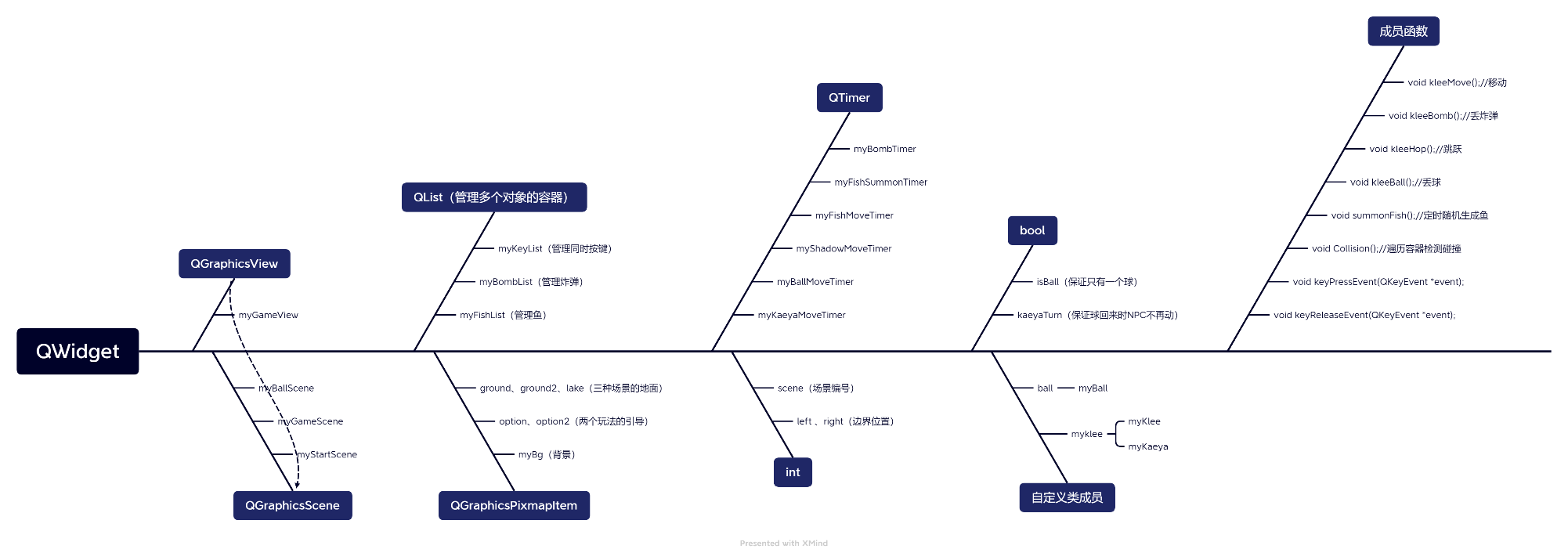
1. **课题要求**
2. 面向对象。
3. 单元测试。
4. 模型部分
5. 验证
6. **主要流程**
   1. **整体流程**

实现思路：

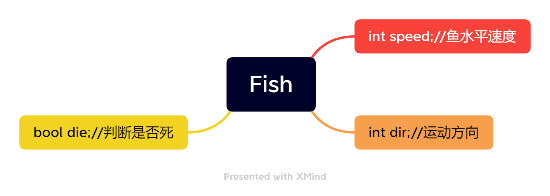
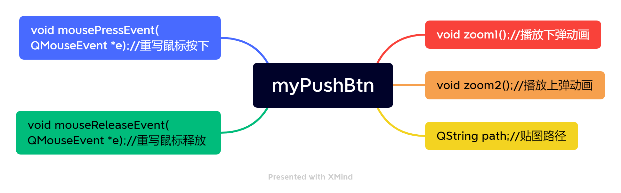
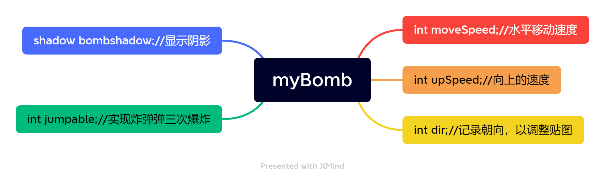
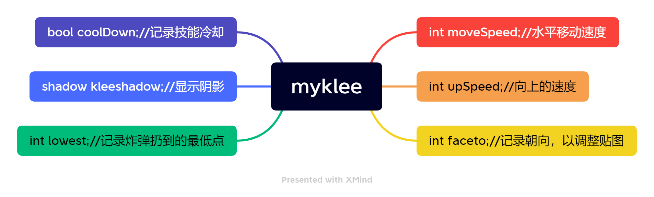
以QGraphics体系为基础，对游戏内包含的对象用QGraphicsPixmapItem类进行封装，并在主窗口QWidget内通过函数逻辑实现对象间的交互。

* 1. **主窗口包含成员**

****

****

* 1. **封装类介绍**

****

* 1. **单元测试**

通过多次试玩，保证游戏不出现贴图错误、内存溢出、成员丢失、AI错误等问题。

1. **单元测试**

测试：

表 1：测试案例

|  |  |  |
| --- | --- | --- |
| **操作** | **输出** | **目的** |
| 点击按钮实现场景切换 | 正常运行 | 场景切换后不应有功能失效 |
| 键盘控制移动 | 正常运行 | 人物应在地面移动，而且不应越过边界，阴影应跟随人物移动 |
| 控制一边移动一边跳跃 | 正常运行 | 跳跃功能应正常进行，并可以同时检测多个按键 |
| 边移动，边跳跃，边丢炸弹或球 | 正常运行 | 炸弹和球应在正确位置生成，并正确表现物理效果 |
| 炸鱼功能试玩 | 正常运行 | 炸弹应能与鱼交互，鱼死后应该在播放效果后移出场景并释放。 |
| 打球功能试玩 | 正常运行 | 球应可以左右弹跳，NPC应有基本的AI |

具体试玩过程在视频中有展示。

1. **收获**
   1. **高级for循环**

for(auto Fish:myFishList){

if(scene==2)

Fish->show();

else {

Fish->hide();

}

if(!Fish->die)

Fish->move();

}

可以用以上代码方便地遍历链表类QList内所有成员

* 1. **简单AI实现**

if((myBall.x()<myKaeya.x()&&myBall.x()>700)&&kaeyaturn&&myKaeya.x()-myBall.x()>300){

myKaeya.moveBy(-10,0);

}

if((myBall.x()<myKaeya.x()&&myBall.x()>700)&&kaeyaturn&&myKaeya.x()-myBall.x()<=300){

myKaeya.moveBy(-5,0);

}

if((myBall.x()>myKaeya.x()&&myBall.x()>700)&&kaeyaturn){

myKaeya.moveBy(10,0);

}

if(myKaeya.y()<725)

{

myKaeya.jump();

}

if(myBall.x()==myKaeya.x()&&myKaeya.y()==725){

myKaeya.upSpeed=20;

myKaeya.moveBy(0,-myKaeya.upSpeed);

}

通过检测角色与球的位置关系控制角色的移动模式

* 1. **游戏优化**

myKlee.setCacheMode(QGraphicsItem::DeviceCoordinateCache);

myKlee.kleeshadow.setCacheMode(QGraphicsItem::ItemCoordinateCache);

connect(teleportBall,&MyPushBtn::clicked,[=](){

//传送锚点传送去打球

myFishMoveTimer->stop();

myFishSummonTimer->stop();

myBombTimer->stop();

}

通过调整类的刷新模式，以及切换场景后停止定时器等方式最大程度保证了画面的高刷新率。

* 1. **重力效果模拟**

void myklee::jump(){

this->moveBy(0,-upSpeed);

upSpeed--;

}

在定时器中调用该函数，可以实现Y方向上模拟重力加速度的效果。