PinBall-作业报告

Jaeger: **毛奕澄 周启轩 范佳骏**

2023程序设计实习 43组项目

程序功能介绍

- 本程序是对Windows XP系统自带的游戏之一 *三维弹球* 的模仿,除大部分复现了 *三维弹球* 的游戏性之外,还加入了开场动画,地图编辑器,以及自定义游戏材质的功能.

开场动画

- 开场动画模仿 *Star Wars* 系列的开场动画
- 旨在阐述游戏的背景故事,这里提到的许多功能,都在项目中得以实现
- 开场动画在程序开始运行时自动开始播放,鼠标左键单击可以跳过.播放完毕或者跳过后,将会进入游戏,并自动开始播放游戏音乐

游戏界面

游戏界面分为左,右两个半区.左侧半区为游戏,右侧半区展示分数以及功能按键

- 游戏的地图设计主要参考 *三维弹球* ,地图元素包括球,弹力鼓,奖励点,三角形弹力鼓,轨道,四边形弹力鼓,以及玩家可控制的活版
- 游戏过程中,球将会在地图中上下弹跳,根据击中的地图元素不同,将会得到不同的分数,同时,球在得分后一段时间,将会被"激活",并获取双倍分数
- 玩家可以按下 **Z** 按键,来控制左侧活版击球, **M** 按键来控制右侧活版击球(注意:可能需要鼠标点击游戏窗口,让窗口获得焦点后,才能进行键盘控制)
- 若球从两活版中间逃离地图,将视为游戏结束,会弹出游戏结束对话框,提示当前分数,玩家此时可选择再来一场,或者退出游戏

- 右侧半区的上半部分,分别展示当前分数(上),以及最高分数(下)
- 右下角有两个按键,分别为**Edit**和**Menu**
- 点击 **Edit** 按键,将会弹出对话框并暂停游戏,点击 **YES** 切换到地图编辑器,点击**NO**返回游戏并继续
- 点击**Menu**按键,或者在游戏中按下**ESC**,将会弹出暂停菜单并暂停游戏,暂停菜单包括静音背景音乐,静音游戏音效,开始新游戏,返回游戏和退出游戏
- 静音按键为状态按键,右侧图标显示为"X"表示已经静音,显示为"□"表示未静音
- 点击开始新游戏,将会返回游戏界面,并开始一局新游戏
- 点击返回游戏,将会返回游戏界面,并继续正在进行的游戏
- 点击退出游戏,将会弹出对话框,点击**YES**退出程序,点击**NO**返回暂停菜单

地图编辑器

玩家通过在游戏界面,点击**Edit**并确认来进入地图编辑器
地图编辑器同样分为左右两个半区,左半区展示已经加入地图的元素,右半区展示可加入的地图元素,以及功能按键

- 右半区上半部分展示四种可加入到地图中的"动态"地图元素:三角形弹力鼓(右),奖励点,三角形弹力鼓(左),弹力鼓.可以通过按下鼠标左键将它们拖拽到左侧的地图上来将他们加入到地图上.
- 地图元素不允许相互重叠,也不允许放在地图外,同时不允许放置在地图中心发球处.如果放置位置合适,地图元素将显示为半透明,若不合适,将会额外有红色滤镜

- 在允许放置的情况下,松开鼠标左键,元素被放置在地图上,不允许放置的情况下,松开鼠标左键,元素将被回收.
- 已经放置的地图元素,可以再次拖动.若想要移除元素,可以直接拖拽到地图外,也可以在按下左Ctrl的同时,鼠标左键点击来移除

- 右半区中部,展示当前主题(材质)的缩略图与标题,点击缩略图两侧的左右按键,可以切换主题,关于如何安装或制作主题,请见 自定义游戏材质
- 右半区下部有**Play**和**Menu**两个按键
- 按下**Menu**按键,将进入菜单.菜单除了包括静音,返回,退出以外,还包括重力调节,重设地图
- 重力调节为滑动条,可以通过滑动来控制重力大小,允许范围为默认值的50%到300%
- 重设地图为单击按钮,按下重设地图将会将已经放置的元素清空,并将重力还原为100%
- 按下**Play**按键,将弹出对话框,点击**NO**返回地图编辑器,点击**YES**切换到游戏,并在地图编辑器中设计的新地图上开始新游戏.如果此前地图编辑器上没有加入任何"动态"元素,将会在默认地图上开始游戏,然而重力的设置始终有效.

自定义游戏材质

本游戏支持自定义材质包(主题包,Themepack)

- 自定义材质包不需要重新编译源代码,只需要按要求放置后,重新启动游戏,即可在地图编辑器处启用新的材质包
- 作为示例,源代码分支中另外提供由我们自己设计的额外材质包

- 要安装材质包,请将Themepacks文件夹,包括里面的文件,全部放置在编译完成后的**可执行文件**所在的目录下
- 要自己制作材质包,需要自行在Themepacks下建立材质包文件夹,提供材质文件,并按照要求填写theme.json中的所有要求的字段,具体的字段要求,以及它们与地图元素的对应关系,请参照材质包制作指南 来了解更多

设计细节

程序的结构树,请参阅源代码分支,这里介绍主要类的设计细节

模块(Part):视频播放

- SkippableVideoWidget
  - SkippableVideoWidget使用QMediaPlayer控制开场动画的播放
  - 视频和音频分别通过QVideoWidget和QAudioOutput传递给QMediaPlayer
  - 通过QHBoxLayout来控制播放窗口的位置
  - 视频播放完成或者鼠标点击后，释放end信号

模块(Part):UI设计

按钮部分

- MButton类
  - 功能:为本工程中,菜单界面的按钮包装统一的界面风格
  - 继承自:QFrame
  - 部分内部细节:
    - 用QLabel进行文字绘制
    - 使用QGraphicsDropShadowEffect实现外发光
    - 重写父类enterEvent和leaveEvent实现按钮区域高光
  - 暴露给子类:setText(QString const&)用于子类内部更新按钮标题
- MPushButton类
  - 功能:实现QPushButton的替代品
  - 继承自:MButton
  - 部分内部细节:重写父类mousePressEvent,用来释放按钮点击信号
  - 暴露给外界:信号void pushed()
- MCheckButton类
  - 功能:实现QCheckedButton的替代品
  - 继承自:MButton
  - 部分内部细节:
    - 额外添加一个QLabel用于显示图标
    - 重写父类mousePressEvent实现按钮真/假状态控制
    - 重写父类enterEvent和leaveEvent实现图标高光切换
  - 暴露给外界:信号void setChecked(bool)
- MSlider类
  - 功能:包装能拖动,且自动显示滑动条值的QSlider替代品
  - 继承自:MButton
  - 部分内部细节:
    - 额外添加一个QSlider用于拖动更改数值
    - 利用setText()实时更新数值
  - 暴露给外界:
    - void setMinimum(int),void setMaximum(int)设定最大值,最小值

        - bool setDefault(int)设定默认值,若默认值在最大最小值范围外则失败并放回false
        - void number_changed(int)传出变化后的数值

确认框部分

- MConfirmation类
    - 功能:包装外观统一的是/否二元确认框
    - 继承自:QDialog
    - 部分内部细节:
        - 通过QLabel绘制标题,背景
        - 通过两个QPushButton实现是/否选框
        - 重写父类keyPressEvent,实现ESC拒绝,Enter接受
    - 暴露给子类:支持UI细节的微调
    - 暴露给外界:
        - void accepted(),void rejected()
- 各Confirmations
    - 功能:程序实际使用的确认框,对基类UI细节进行调整
    - 继承自:MConfirmation

菜单部分

- MTab类
    - 功能:作为基类,管理菜单界面上按钮的内存,管理基本信号
    - 继承自:QDialog
    - 部分内部细节
        - 通过QLabel展示标题
        - 使用std::vector<MButton*>来管理按钮
        - 重写父类closeEvent,向父界面传递界面关闭信号
        - 重写父类keyPressEvent实现ESC关闭
    - 暴露给子类:按钮的创建与设置
    - 暴露给外界:
        - void closed()关闭信号
        - void exitRequest()退出游戏信号
- PauseMenu类
    - 功能:作为游戏中的暂停界面,提供静音,重新开始,继续,退出游戏的交互
    - 继承自:MTab

    - 部分内部细节:
        - 用MCheckButton提供静音交互
        - 用MPushButton提供剩余上述交互
    - 暴露给外界:
        - void newGameRequest()重新开始信号
        - void setButtons()与其他界面进行静音状态同步

- SettingsMenu类
    - 功能:作为地图编辑器的设置界面,提供静音,设定游戏重力,重设地图,继续,退出游戏的交互
    - 继承自:MTab
    - 部分内部细节:
        - 用MCheckButton提供静音交互
        - 用MSlider提供重力设定交互
        - 用MPushButton提供剩余上述交互
    - 暴露给外界:
        - void resetMapRequest()重设地图信号
        - void gravityChange(int)重力更改信号
        - void setButtons()与其他界面进行静音状态同步

模块(Part):地图编辑器

地图元素部分

- MDraggableShadow类
    - 功能:实现自动检测碰撞箱
    - 继承自:QObject,QGraphicsPixmapItem(由于Q_OBJECT问题,被迫重复继承QObject)
    - 部分内部细节:
        - 用paint绘制图像,并自动设定碰撞箱

- MDraggable类
    - 功能:实现可拖动的地图元素
    - 继承自:QWidget
    - 部分内部细节:
        - 用QPixmap+QLabel绘制图像
        - 重写父类mousePressEvent,mouseMoveEvent,mouseReleaseEvent来检测和跟随鼠标移动

    - 暴露给子类:
        - 图像的设定
    - 暴露给外界(部分):
        - 图像的更新
        - virtual QPoint locatingPoint()const =0图像的定位点
        - void be_moved()从原位移动的信号
        - void be_released_invalidly(),void be_released_validly()放置在非法位置/合法位置的信号
        - void be_removed(MDraggable*)被移除的信号

- 各MDraggables
    - 功能:处理图片,定位点等细节
    - 继承自:MDraggable
    - 部分内部细节:
        - 重写locatingPoint以确定定位点

元素的管理部分

- MDragContainer类
    - 功能:实现地图元素的容器,统一管理地图元素的图片更新
    - 继承自:QObject(图片并没有直接在这一层展示)
    - 部分内部细节:
        - 使用QVector<MDraggable*>管理已放置的地图元素
        - 使用QGraphicsScene管理地图元素的碰撞箱
    - 暴露给外界:
        - QVector encodeMap()将编辑好的地图打包成编码后的地图
        - void clear()重设地图
        - void updateTheme()更新图片材质包

- MapEditor类
    - 主要功能:
        - 承载地图编辑器的图形
        - 向程序中枢暴露地图编辑器的编码结果
        - 向程序中枢和其他部分传导主题(材质)更换的信号
    - 继承自:QWidget
    - 内部细节过于繁琐,且大部分为信号关联,图片设定,UI搭建等,没有特意说明的必要
    - 向外界暴露:

        - void exitRequest()程序退出信号
        - void switchRequest(QVector,int)地图编辑器编码输出信号

模块(Part):程序中枢

- GameWidgetManager类
    - 主要功能:
        - 管理程序的主要窗口:开场动画,游戏,地图编辑器的切换关系
        - 管理程序使用的音乐,音效和高复用次数贴图
    - 继承自:QObject
    - 部分内部细节:
        - 使用QStackedWidget来管理上述窗口
        - 使用QVector<ThemePack*>来统一读取和管理主题包
- ThemePack类
    - 主要功能:动态识别和读取特定形式的主题包,从而对主题切换进行管理
    - 部分内部细节:
        - 使用enum来管理主题包中图片的下标
        - 首先读取.json文件,按照key来获取主题包的标题及各图片文件名
        - 再逐个按照文件名读取图片文件,并加载成QPixmap
- object类是所有地图元素的基类，它的成员变量记录了它是否是加分项、加分多少等信息

模块(Part):游戏部分

- object派生出obline和obcircle,两个派生类分别代表线段型的地图元素和圆弧型的地图元素。它们的成员变量记录了地图元素类的位置信息。它们各自有一个碰撞判断函数（以小球指针为参数），用来判断小球与它们是否发生碰撞。
- obline类的构造函数的参数是线段两个端点的坐标，从而确定其位置。
- obcircle类的构造函数的前三个参数为圆心位置、圆的半径，除此之外还有一个布尔型参量确定其是否是完整的圆，如果不是，则还有四个参量表示圆弧两个端点的坐标。由于地图元素中只会出现劣弧，所以构造函数可以确定圆的位置信息。
- 由这两个类派生出所有的地图元素类：直线墙（stwall）、弹力线（kidney）、圆弧墙（cirwall）、奖励点（award）、弹力鼓（drum），以及小球（ball）。前两个派生自obline，后四个派生自obcircle。它们各自有不同的碰撞函数（以小球指针为参数），依照物理规律描述小球与其发生碰撞后速度的变化。

- 奖励点是一个半径较小的圆，每次小球经过其附近，只要小球与其位置有重合就会加分，不会影响小球运动。
- 直线墙、圆弧墙都有一个浮点型成员变量-弹性系数，小球与其发生碰撞反弹时，法向速度不仅反向，而且要乘上这个系数。
- 弹力线相比直线墙多一个特性：小球碰到它会加特定分数。弹力鼓也是如此。
- map类是整个地图抽象出的类。每当游戏开始，由成员函数构建所有地图元素、在相应位置显示地图元素的图像。
- 加分地图元素弹性系数大于1，不加分地图元素弹性系数不会超过1
- 三角形弹力鼓由一根弹力线、两根直线墙和两个圆弧组成。
- 活板由三根首尾相连的平行等长弹力线组成。靠近墙的端点固定不动，按下Z（M）键，左（右）端的活板开始朝上匀速旋转，旋转角到达最大值时停止旋转并保持。松开Z（M）键，活板朝下匀速旋转，直到恢复原来位置时停止。
- map类中，每经过一个时间步长（由计时器控制），调用onestep成员函数，更新小球和活板的位置以及它们图像的位置，依次对地图元素进行碰撞检测，若发生碰撞，调用地图元素的碰撞函数，如果有加分，还要调用特效函数产生特效。小球掉出活板间的空隙，则游戏结束，调用成员函数清除图像、清除地图元素，并发出游戏结束的信号。小球与加分元素碰撞后会变成另一种球一段时间（由计时器控制），在这段时间内若再与加分元素发生碰撞，加分翻倍。

分工情况

周启轩

- 制作开场视频，并完成控制视频播放的代码
- 设计基础地图，给出具体的坐标
- 绘制背景
- 绘制各基本元素：球（包括激活状态）,弹力鼓，奖励点，三角形弹力鼓，轨道，四边形弹力鼓，以及玩家可控制的活版
- 建立球的碰撞模型，并通过近似方法简化了两运动物体的碰撞

毛奕澄

- 设计和实现游戏主题ui
- 设计和实现地图编辑器
- 提出和实装ThemePack(主题包/材质包)系统

- 实现程序中枢框架

范佳骏

- 编写各个地图元素类，实现球与地图元素的碰撞检测与碰撞模型
- 编写游戏中地图的运行过程，包括地图的构建函数、每个步长的运行函数等等
- 编写游戏中地图图像的显示与位置变化
- 录制碰撞音效

项目总结与反思

1. 项目分工能力有待提高,项目初期的分工设计有所欠缺,导致部分模块内部也存在分歧.经过本次项目,对于项目分工有了更好的理解
2. 沟通同步能力有所欠缺,队员使用Github不够熟练,导致时常漏掉跟进.经过本次项目,对于项目内部沟通成本巨大有了更深刻的认知
3. 代码能力不足,每个队员,在开发过程中,或多或少都写出了低级错误,导致程序无法正常运行,同时,在从网上资料学习的过程中,也常常遇到没能正确解决问题的情况,这时候不得不翻阅文档,以及Qt示例代码,更深入地学习Qt.不论是什么项目,代码能力都是根本,今后也要不断提升代码能力
4. 项目测试缺失,项目开发初期,程序的许多基本逻辑没有得到全面的,跨平台的测试,这导致在后期不断遇到逻辑问题,大大提高了后期开发成本
5. 顶层设计缺失,项目初期仅仅进行了分工,而没有对需求进行明确的分析,没有对各模块讨论进行顶层设计,提高了后期维护成本.工程之始在于分析需求,这是非常宝贵的经验
6. 认识到跨平台的艰辛.三名队员分别在Mac,Intel&Nvidia,AMD设备上进行编程,这导致程序在三个人的设备上常常表现得不同,同时也常常见到不同的错误.程序设计不像OJ作业,要在多种平台上都能正常运作,大大提高了对程序完善性,安全性的要求.在这里,我们不仅看到了对代码能力的要求,更看到了不断地测试的重要性
   总的来说,经过这次大作业,我们对于软件工程和程序设计有了更加深入的了解,代码能力和工具使用能力也取得了提升