QT大作业之Eisaea

队伍成员

队长：刘晨宇

队员：孙艺芷

功能介绍

我们实现了一个4K音游，即有四条音轨，每条音轨会落下一些音符，在对应的时间按下键盘上相应的键即可获得相应的分数。

我们实现了调节音符流速、调节判定延迟、暂停游戏、重试游戏等功能，并且使用python脚本导入了丰富的曲目，提供了精美的曲绘。

构建方式

采用 Qt 5.15.2 MinGW 64-bit 编译

值得注意的是我们使用了QMediaPlayer类，此类不提供解码器，故需单独安装音频解码器。我们提供了windows下的音频解码器安装文件（LAVFilters-0.77.2-Installer.exe）。不安装会造成无法播放歌曲。

类设计细节

utils.h

用于存放一些工具类和用extern声明的全局变量

class QLabel_C

此类是QLabel的派生类，重载了mousePressEvent函数，使Label具有按钮的功能

- void clicked() 一个signal空函数
- virtual void mousePressEvent(QMouseEvent* event) 被点击时emit一个clicked信号

由于时间紧迫以及交互界面十分庞大、按钮数量比较多，没有加入播放音效等功能。

class Song

用于存放一首歌曲的信息，包括难度、曲绘路径、谱面路径、名称、作者、bpm

全局变量

- extern Song songs[song_size] 存放42首歌曲
- extern QString font_path[font_size] 存放字体文件路径
- extern double FALL_SPEED 控制下落速度
- extern int WAIT_TIME 控制游戏延迟
- extern string pack_name[9] 曲包名称
- extern string help_text[help_tot] 帮助文本

游戏界面

class Button

此类不是qt派生类，表示一个音符。存放每个音符的类型（点按或长按）、起始时间、结束时间、判定时间序列、音符位置、以及一个QPixmap存放音符的图片

- void Button::fall(int ntime) 通过当前时间ntime更新音符的位置

  根据公式 $\text{position} = \text{checklineposition} - \text{speed}*(\text{ntime - checktime})$ 来确定位置

class Track

此类不是qt派生类，表示一条音轨。音轨中存放这条轨道上所有的音符，并控制它们下落、绘制窗口内的音符、响应用户输入并向游戏界面发出信号。

- void Track::drawtrack(QPainter &painter) 绘制窗口内的音符
- void Track::onpressed(int ntime) 响应用户按下按钮
- void Track::onreleased(int ntime) 响应用户释放按钮
- void Track::uptateStatus(int ntime) 更新所有音符的信息

class Gamescene

此类是QWidget的派生类，表示游戏的窗口。存放四条音轨，连击数、每个判定的数量等等信息，并检测用户输入，传递给对应的音轨。使用QLabel来存放按钮、界面素材、需要绘制的文字等等。

- void Gamescene::paintEvent(QPaintEvent *event) 重载paintEvent绘制当前界面
- void Gamescene::updateStatus(int ntime) 更新每条音轨信息
- void Gamescene::keyPressEvent(QKeyEvent *event) 重载KeyEvent来响应用户输入
- void Gamescene::keyReleaseEvent(QKeyEvent *event) 同上
- void Gamescene::pause() 暂停游戏并显示返回、重试、退出按钮

游戏机制

我们使用了多个QTimer和connect函数来实现游戏的多线程机制。

- QTimer m_timer 以一定的帧率刷新当前界面
- QTimer bgTimer 控制背景运动
- connect(&m_track[i],&Track::checksignal,this,&gamescene::checkslot);

  当一条音轨发出判定信号时，checkslot函数执行统计连击数、更新分数并重新绘制当前界面
- QTimer startcount 控制开始的倒计时

每个音符的位置变化由Track负责，而游戏界面只负责刷新并显示它们。音符的判定流程如下：

1. 每条Track里有两个 `vector<Button>`，表示待加入当前游戏的音符`nex_but`和已经加入当前游戏的音符`ontrack_but`
2. Updatestatus检测到当前时间大于第一个`nex_but`的判定开始时间减去运动所需的时间，把这个音符加入`ontrack_but`
3. Track检测到mousepressed信号，如果当前第一个`ontrack_but`是点按，则根据时间差确定判定等级并emit相应的信号。若大于lost的时间差则不执行任何操作。
4. 长条判定和arcaea的蛇十分类似，Track用一个变量记录当前音轨是否被按下，并在upstateStatus里对长条的判定序列进行判定
5. 若超过了lost的判定时间还没有检测到mousepressed信号，则在upstateStatus发出lost信号

谱面文件是导入的malody的json文件，具体方式是使用QFile存放文件，读入为QByteArray，再转换为QJsonDocument，使用QJsonObject和QJsonArray的成员函数转换为数据。读入流程是：

1. 对于有变速的谱面遍历bpm序列，计算变速的时间点
2. 对于每个note，通过节拍数、当前的bpm、上一次变更bpm的时间计算当前音符的开始时间和结束时间(ms)
3. 对于长条，适当地在中间加入一些判定。

其他界面

每个界面都派生于基类basescene，basescene中包含一些共有的元素和方法，每个派生类又包含若干个QLabel_C来设置自己的界面元素。

界面间的交互方式是直接new一个新界面并close当前界面，需要使用`setAttribute(Qt::WA_DeleteOnClose);`来释放内存。界面布局复刻了创新3d立体音游arcaea，总体美观度较高，内存也能正确释放。

class basescene

- 各种 draw、drawtext、drawpix函数：在不同的位置绘制元素（以及阴影）的函数
- 各种 qtimer：实现多线程的动画效果
- QGraphicsOpacityEffect：调节透明度
- QLabel_C setbox：设定功能

class mainscene

主界面，右侧的游戏伙伴——光 是贯穿整个游戏的角色，在多个界面中都有显示，点击帮助按钮可以从她那里获得帮助。

class songselect

歌曲选择界面，点击对应曲目会发射信号，对应槽函数改变显示内容

class chapterselect

章节选择界面，每个章节（World里共有8个章节）包含6首左右曲目

class resultscene

结束界面，展示游玩结果，交互功能较少

项目总结与反思

本次项目使用了很多程设课上学到的c++特性，如继承、多态等，并设计了较为良好的交互方式和较为美观的界面。使用qtimer制作的动画效果也尚可，
值得一提的是对带变速的json谱面设计读取的算法花费了一些功夫。

不足的地方有未使用Qt自带的ui文件，窗口跳转的方式还可以改进等