QT 大作业——钢琴模拟器作业报告

一．程序功能介绍

1.多音阶键盘：程序模拟了一个完整的钢琴键盘，包括多个音阶。每个键通过 QSoundEffect 关联一个特定的音效，用户按键时可发出对应的音符声音，可调节不同音阶满足演奏需要。

2.录制与回放：用户可以录制演奏，包括每次按键的具体时间和音符，之后可以回放这些操作。回放功能包括基本的控制播放、停止。

3.节拍器功能：帮助用户练习保持节奏，支持调整节拍数（BPM），并通过视觉和声音反馈指示当前节拍。

4.乐谱查看功能：能够将已存放在文件夹中的乐谱图片调整放入大小合适的窗口，方便演奏时查看。

二．项目各模块与类设计细节

1. Piano 类设计

Piano 类是整个应用的核心，负责钢琴模拟器的大部分功能。该类继承自 QMainWindow，提供了一个完整的钢琴界面和相关的音乐功能。

（1）构造函数：在构造函数中，Piano 类初始化其成员变量，包括多个 QSoundEffect 对象，每个钢琴键对应一个。这些声音效果对象被配置为播放预加载的音频文件，这些文件存储在资源文件中。构造函数还设置了用户界面，连接了各种按钮的点击信号到相应的槽函数，这些槽函数用于处理按键事件、录制和回放等操作。

（2）声音效果：每个钢琴键通过一个 QSoundEffect 对象来管理其声音播放，这使得每次按键都能播放一个特定的声音。这些声音文件是预先录制的钢琴音符，存储为 .wav 文件，并通过资源文件访问。

（3）动态音符调整：通过界面上的 voicehigh 控件，用户可以调整演奏的音高。Piano 类会根据这个控件的值来动态调整音符，通过改变与每个

QSoundEffect 对象关联的资源路径来实现。

（4）钢琴按键：每当用户按下一个钢琴键，相关的槽函数会被触发，该函数从对应的 QSoundEffect 对象播放音符。

（5）录制回放：如果启用了录制功能，按键事件还会将按键信息记录到一个列表中，包括按键的时间戳和音符。在回放模式下，Piano 类会根据录制的信息和时间戳逐一播放录制的音符。使用一个 QTimer 对象来控制音乐的回放。定时器的超时信号连接到一个槽函数，该函数根据录制的数据和当前时间来确定何时播放哪个音符。

（6）乐谱查看：通过一个按钮 library 触发允许用户通过图形界面选择并查看存储在本地的乐谱文件，支持的文件格式包括常见的图片格式如 PNG、JPG 和 JPEG。如果图像文件加载失败（文件损坏或不支持的格式），弹出一个警告对话框，通知用户图像无法加载。

2. MainWindow 类设计

（1）成员变量:Ui::MainWindow *ui: 自动由 Qt Designer 生成的界面类的实例，用于访问和操作所有的用户界面元素。QTimer *m_timer: 用于管理和触发定时事件，主要用于控制节拍器的节拍间隔。int m_beatsPerMeasure: 表示每小节的节拍数，用于配置节拍器的节拍循环。int m_bpm: 表示每分钟的节拍数（BPM），用于设置节拍器的速度。int m_currentBeat: 当前节拍的索引，用于追踪当前的节拍状态。QMediaPlayer *metronome1 和 QMediaPlayer *metronome2: 用于播放节拍声音，支持不同的节拍声效。

（2）构造函数:在构造函数中，MainWindow 初始化其成员变量并设置 UI。它设置了 m_timer 的间隔，基于 m_bpm 计算得出，以及连接 m_timer 的 timeout() 信号到 tick() 槽函数，用于处理每个节拍的逻辑。

（3）功能实现:节拍器逻辑 (tick()): 该函数是节拍器的核心，它根据 m_beatsPerMeasure 的值来决定如何更新 UI 元素（如按钮和箭头的选中状态）以及何时播放相应的节拍声音。此函数通过切换 QMediaPlayer 对象的播放状态来控制音频的播放和停止，以此实现节拍的声音反馈。

（4）音量和节拍控制:on_volume_down_clicked() 和 on_volume_up_clicked(): 这两个槽函数用于调整节拍器的 BPM，通过修改 m_bpm 并重新设置 m_timer 的间隔来实现。on_pushButton_x_x_clicked(bool checked): 这组函数用于处理用户通过界面更改节拍设置的交互，它们更新 m_beatsPerMeasure 并重置节拍器的状态，确保节拍器与用户选择的节拍设置同步。

（5）启动与停止节拍器:on_pushButton_clicked(bool checked): 这个槽函数控制节拍器的启动和停止。当按钮被选中时，它启动节拍器；反之则停止。它通过设置 m_currentBeat 为 0 来重置节拍器的状态，并在启动时立即调用 tick() 来初始化节拍。

三．小组成员分工

徐圣喆：钢琴键，录制及播放功能

熊轲：节拍器

王子昂：乐谱查看，整合功能

四．项目总结与反思

1. 音效管理：项目初期音效加载和管理是一个挑战，因为不同音符的加载和切换需要高效处理。通过预加载所有可能的音效文件并使用 QSoundEffect 进行管理，解决了响应时间慢的问题，实现了即时音效播放。

2.用户界面友好性：初始的用户界面不够直观，逐步优化了界面设计，如调整按钮布局、增加图标和提示信息，使操作更加直观易懂。

3.性能优化：在回放和录制功能中，处理大量数据时遇到性能瓶颈。通过优化数据结构和改进时间管理算法，提高了数据处理的效率和准确性。

4.项目组织与合并：在不同项目的合并中由于使用 QT 版本不同的问题导致出现困难。在项目合作的初期应该考虑并提前计划。分析并做好完整规划再开展具体内容。

项目反思与未来展望

1.功能集成度：虽然项目已经集成了少量功能，但音乐编辑和声音调整功能还有待开发。未来可以考虑添加音频剪辑和混音功能，让用户不仅能播放和记录，还能创作和编辑自己的音乐作品。

2.用户交互体验：项目在用户交互方面仍有改进空间，例如，可以通过增加触摸屏支持和手势操作来提高互动性，可以在平板电脑和触控屏笔记本上使用。