基于 QT 和 MPLAYER 的视频播放器设计与实现

史雨鑫 白俊鸽
四川大学锦城学院计算机与软件学院 四川 成都 610000

【摘 要】随着互联网行业的发展，网络用户对于视频播放器的需求越来越多样化，个性化。然而当前市面上的视频播放器多具有限制播放视频格式，广告繁多，忽略界面设计等缺陷。本文以基于 QT 的视频播放器为例，介绍如何利用 Mplayer 作为底层支持，QT 作为包装和逻辑构造，以此来实现一个界面美观，功能实用的个人视频播放器。该播放器抛却繁杂广告，提升播放性能，能够给用户还原最便捷流畅的播放体验。

【关键词】QT；Mplayer；视频播放器

引言

近年来随着自媒体行业的飞速发展，生活节奏的加快，让人们疲于阅读文字类信息，倾向于直接观看视频，快捷高效获取信息 [1]。而视频播放器作为必不可少的播放工具，亦应顺应时代需求提供给用户一个更加良好的体验，播放器界面应化繁为简，省却多余广告资讯，播放相关功能应更具多样性与实用性，支持多种视频格式，使用户拥有一个个性化、良好、流畅的观看体验。

1 视频播放内核的选择

在选择使用何种播放内核时，首先考虑了 QT 支持的多媒体模块 QMediaPlayer，但经调研后发现，QMediaPlayer 在视频格式的支持上存在缺陷，它仅支持 rmvb、mpg 和 mp4 格式的视频播放，对于当下时代发展和用户需求来说并不完善，且在功能逻辑控制上存在局限性，故最终选择了 Mplayer 作为播放内核。Mplayer 是一款基于多平台的成熟视频播放器，几乎可以兼容当前常见的所有视频格式，它有两种运行模式：一是正常工作模式，直接响应键盘信息；二是从机模式，Mplayer 在后台为其他程序运行，不再截获键盘事件 [2]。本文中所使用的正是上述的从机模式，将其输出固定于基于 QT 的图形界面窗口，并通过管道重写向其发送控制命令。

2 界面设计

2.1 播放区域设计

播放区域包括一个 QWidget 对象和两个 QSlider 对象。其中 QWidget 对象作为视频输出显示的窗口，Mplayer 的输出和视频播放器具有的所有功能控制都将固定在此窗口。用户可以从该窗口进行视频观赏。而 QSlider 对象则作为视频（音量）播放进度条，用户可以通过此进度条对视频播放进度或音量进行调整。视频进度条通过 border-image 设定了自定义滑块图标，配色搭配整体色调风格，具有简洁美观的特点。

2.2 功能区域设计

功能区域包括了该视频播放器所具有的全部功能，如视频区域截图，快进，快退，加速，减速，播放 / 暂停，单遍循环 / 顺序播放，停止播放，播放上 / 下一个，全屏，音量调节等功能。所有功能均为 QPushButton 对象，将默认按钮替换为风格色调统一的美观图标，并通过 setToolTip 方法为所有图标设置了悬停文本提示，用户可以通过悬浮提示任意切换想要实现的功能与效果。除此之外，功能区域还包括 QLable 对象，用于实时显示当前播放视频的时间、进度等信息。

2.3 播放列表区域设计

播放列表区域包括一个播放列表和两个按键，用户可以通过添加 / 删除两个按钮对播放列表进行视频文件的删减操作，其中列表为 QListWidget 对象，按键为 QPushButton 对象。

3 功能实现

3.1 主要功能描述

本文中所讨论的视频播放器基于 Mplayer[3] 内核和 QT 逻辑设计后拥有许多实用性功能，各功能均围绕播放主旨进行发散，包括双击播放，选中按钮播放，循环播放，快进 / 退，加 / 减速和截图功能等。用户可以通过本视频播放器拥有一个多样化的流畅观看体验。

3.2 主体播放功能

为了实现播放器的基本播放功能，需要先在 QT 主进程中调用 Mplayer 作为后台支持，通过其规定命令行

进行视频播放、暂停等操作，并将 Mplayer 的输出固定于指定窗口，以此进行视频播放。

该主体功能于函数 void play（QString FileName）中实现。其中 FileName 为通过 QT 交互界面获取的用户所选视频文件的路径及名称。在该方法中，为了正确调用并开启 Mplayer 的从模式，需要以命令行形式依次写入 Mplayer 的命令行源码路径，播放文件路径及名称，播放模式设置等相关参数。在参数设置部分，首先要通过 ui.widget->winId() 方法获取指定控件的窗口 ID，再通过 -wid ID 命令将 Mplayer 的输出固定在此指定窗口上；最后通过 -slave 命令启动 Mplayer 的从模式，使其能通过后台命令行进行相应操作，以此来为用户提供一个图形界面接口，实现各种功能。为了播放界面的简洁与美观，还需通过 -quiet 和 -zoom 命令使播放视频居中，并占据指定窗口全屏。

设定完所有播放参数后，需借助 QT 的 QProcess 类启动 Mplayer，该类用于启动外部程序并与它们通信 [4]。process->start（）会以子进程的方式调用外部程序，视频即可在指定窗口进行播放，并且子进程与此时的 QT 主进程共同存在，互不影响，符合视频播放器的主体逻辑设计。

播放 / 暂停的功能通过 on_zanting_clicked 函数实现。在从模式的命令下，可以使用 process->write(“quit\n”) 将视频进行暂停 / 播放处理，用户可以通过此命令的图形界面接口实现对当前或列表中处于选中状态的视频文件的暂停与播放。此处涉及两个逻辑判定，一是判定此时的按钮悬浮文本处于何种状态，二是对列表选中文件和当前播放视频文件名进行匹配。此处选用“播放”状态进行逻辑分析。

当悬浮文本为“播放”时，即代表此时视频处于暂停状态，或此时无子进程运行。则进入第二个逻辑判断，若有列表文件被选中，将该文件名与存储当前播放视频文件名进行匹配比较，若名字一致，则继续当前子进程视频进行播放，若文件名不匹配，则将当前选中文件名作为参数调用 play 函数进行播放。

3.3 附属播放功能

除主体播放功能外，该视频播放器还具有其他播放附属功能，其中多数基于 Mplayer 的从模式命令行实现，在此选择部分功能进行逻辑介绍。

快进 / 回退功能的实现基于从模式命令 -seek，在其后规定相应正负数字即可进行视频进度调整，本工程中通过 QProcess 的 write 方法进行命令写入，并以 10 秒作为快进 / 回退的基本时间单位。

截图功能的实现基于 QPixmap 的 grabWindow 方法，将视频播放窗口的 ID 作为参数传入该方案，即可获得 QPixmap 类型的图片，最后通过 getSaveFileName 方法，使用户可以通过弹窗界面自由选择 png，jpg 和 bmp 三种格式中的一种保存截图。

全屏功能的实现涉及到顶级窗口与非顶级窗口的先后关系。收到全屏信号后，首先需要通过 setWindowFlags 方法将视频播放窗口设置为顶级窗口，再通过 showFullScreen 方法将其设置为全屏模式。showFullScreen 方法只能对 QT 下的顶级窗口生效。退出全屏则需要重写事件捕获函数 eventFilter 来实现。用户可以通过双击鼠标事件来退出全屏模式，此时视频播放窗口正处于顶级窗口模式，可以捕获事件。当捕获到双击鼠标事件时，便通过 setWindowFlags 方法将该窗口重新设置为非顶级窗口，并将其恢复至原本大小，调整至初始界面位置。

用户可以通过播放窗口旁的标签获取到视频播放的实时进度与时间信息，实时反馈信息需要借助 QT 的信号槽机制。信号槽机制 [4] 是 QT 的重要组成部分，其基本逻辑为控件触发信号并发送给信号接收者，后者接收信号后执行槽函数进行相应的逻辑处理。本工程中即使用视频播放子进程 process 作为信号发送者，当新的子进程启动时，即可触发 readyReadStandardOutput 信号，此时的输出管道中已存有输出信息，QT 主进程作为信号接受者，调用 dataRecieve 方法对管道中的输出信息进行分析和处理。当新的子进程开启，进入槽函数处理后，即可通过 get_time_length/time_pos/percent_pos 命令进行新的信号触发，同时获取视频的总长度、当前播放时间和当前播放百分比等信息。以此达到视频播放时可以循环触发信号槽机制的效果。接收相关信息后，通过 process->readLine 方法以行为单位进行读取，并通过 startsWith 方法对信息头进行分析，对三种请求信息分别进行判断后，通过 QString 的 mid 方法对有效信息进行截取和处理。

用户可以对进度条进行拖拽，以此来控制正在播放的视频进度或音量大小。此处选择视频播放进度条进行逻辑介绍。要实现进度条进度随着视频进度实时更新的效果，仍要借助上文提到的信号槽机制。在 dataRecieve 方法中，主进程收到了当前子进程的播放信息，截取其中的视频总播放时间 totalTime 后，通过 QSlider 的 setRange 方法，将整个进度条的总范围设置为 0-totalTime，使之从长度与当前播放视频长度保持一致。随后再截取视频的当前播放进度信息 currentStr，通过 QSlider 的 setValue 方法将进度条的当前位置设置为当前播放时间。所以当前播放时间 / 视频总播放时间，即为当前进度条播放位置 / 进度条总长度，两者呈现出相同的比例关系，通过信号的不断触发，实现了进度条随着

播放进度实时更新的功能。

3.4 界面拖拽功能

在界面的初始化函数中，作为界面设计的一部分将标题栏进行了隐藏，因此整个界面主体失去了原本依附于标题栏的鼠标拖拽功能。为了让用户可以通过鼠标对整个界面进行随意的位置调整，在工程中进行了相应的鼠标事件重写。整体思路为通过获取鼠标位移距离来同步改变界面框体的坐标。

首先通过重写鼠标按下事件 mousePressEvent，用 globalPos 获取鼠标当前坐标。其次重写鼠标移动事件 mouseMoveEvent，实时获取鼠标当前坐标，计算出当前坐标与原坐标的差值，在界面框架的原坐标上加上差值，通过 move 方法对原窗口进行移动。最后重写鼠标释放事件，其与移动事件采用相同逻辑，即可实现窗口主体随着鼠标拖拽进行实时移动的效果。

4 结束语

本文介绍了如何使用 Mplayer 和 QT 制作视频播放器，前者为视频播放提供了友好的底层逻辑和操作便利，后者通过灵活的线程调用和界面设计给用户带来了美观简洁的外观体验。

当然在产品可用度上，该播放器还可作进一步深入和完善，如实现网络查询和播放功能等。总体来说，该产品代码逻辑清晰简单，功能较为完善，适合初学者学习使用。

【参考文献】

[1] 阳晓霞 , 李春来 . 一种 Web 视频播放器的设计与实现 [J]. 信息与电脑 (理论版),2019(15):62-63+66.

[2] 陈将奇 , 陈小平 . 移植 Mplayer 的 ARM9 视频显示系统 [J]. 单片机与嵌入式系统应用 ,2019,19(05):48-50.

[3]MPlayer- 电影播放器 [EB/OL].[2020-06-07].http://www.mplay-erhq.hu.

[4]Qt Project Group.Qt 开发官方参考文档 [EB/OL]. [2020-06-07]. http://qt-project.org/doc/qt-4.8/.