Alpha1S\_Qt模块说明

-何柳丝

## 一、通用模块说明

Qt的帮助文档是学习Qt最佳的参考文档。Qt编辑器QtCreator左侧“帮助”按钮可直接打开帮助文档，或者打开Qt安装路径中Assistant程序。QtCreator左侧“欢迎”按钮界面中的“示例”和“教程”选项中，有大量可参考的程序代码，也是学习Qt的最佳途径。

### Qt 资源（RCC）

* Qt资源系统：通过Qt帮助文档（即Assistant）-索引-搜索：The Qt Resource System，即可得到Qt资源系统的详细介绍。
* Qt资源加载注意事项：对于静态库和动态库，Qt资源会做不同的处理，静态库不会将资源编译到库文件中，因此在处理静态库资源文件时，需要采用外部资源的方式来引用资源文件。

静态库引用资源示例：

先通过终端命令rcc -binary myresource.qrc -o myresource.rcc生成rcc文件

再由代码中添加QResource::registerResource("/path/to/myresource.rcc");进行注册

### Qt多语言

要使用Qt多语言机制，带翻译的文本在代码中需要以tr()包含。

* 1. 创建多语言：在项目.pro文件中添加

TRANSLATIONS += ../languages/mp3proj\_zh\_CN.ts \

../languages/mp3proj\_en.ts \

* 1. 工具 -> 外部 -> Qt语言家 -> 更新翻译，会在对应目录生成xxxxx.ts文件
  2. 启动Linguist程序，打开xxxx.ts文件，将翻译文本写入“译文”
  3. 选择 文件->发布 生成xxxx.qm文件
  4. 通过QCoreApplication::installTranslator(QTranslator \* translationFile)注册翻译文件

更多详细信息，可以通过搜索Qt帮助文档：Qt Linguist Manual

### Qt样式表Qt Style Sheet

整个工程界面是基于Qt样式表的方式实现，除qss资源文件外，程序中也有部分Widget通过直接设置stylesheet来完成。

* Qt样式表：兼容CSS样式设置，要做出更酷炫、简洁的效果，需要对CSS有一定的了解
* Qt帮助文档中搜索：Qt Style Sheets（Qt样式表的基本介绍），Qt Style Sheets Examples（众多的样式表示例代码），Qt Style Sheets Reference（详细的控件样式表属性说明）

### Qt编译工具QMake

Qt帮助文档中搜索：qmake进入qmake Manual可详细的了解QMake的编写和使用。

### Qt应用程序发布

Qt帮助文档中搜索：Deploying Qt Applications可详细了解Qt应用程序发布事项。

## 二、软件模块说明

### UBXRobot3DView、LibFBX模块说明

* 功能说明：

LibFBX负责加载和绘制Alpha1S.FBX 3D模型以及根据Alpha1s.xml创建舵机节点信息。

UBXRobot3DView负责提供显示，控制，以及更新3D模型。

* 设计说明

3D模型的显示和操作整体沿袭Windows版本的框架设计，由于Qt本身封装了对OpenGL的支持，因此实现上要做修改。

* + 1. LibFBX去除所有Glut相关函数的调用（必须去除，确保在Qt框架下能编译通过），Qt OpenGL支持源生OpenGL接口
    2. **UBXRobot3DWidget类：**继承QGLWidget类，实现IUBXRobotControl接口，用于显示3D模型，并提供控制3D模型接口
    3. **UBXRobot3DDisplay类：**继承QObject，内部封装QBasicTimer定时器对象，定时刷新3D模型。UBXRobot3DDisplay对象通过moveToThread移到线程去处理。负责与LibFBX通信
    4. **UBXRobot3DControlWidget类：**继承QWidget，实现zxObserverBase接口。提供控制界面UI，并负责响应UBXRobotController模块与实体机器人通信返回的数据，以更新界面显示。
    5. **UBXRobot3DContainer类：**继承QWidget，实现IUBXRobot3DView，

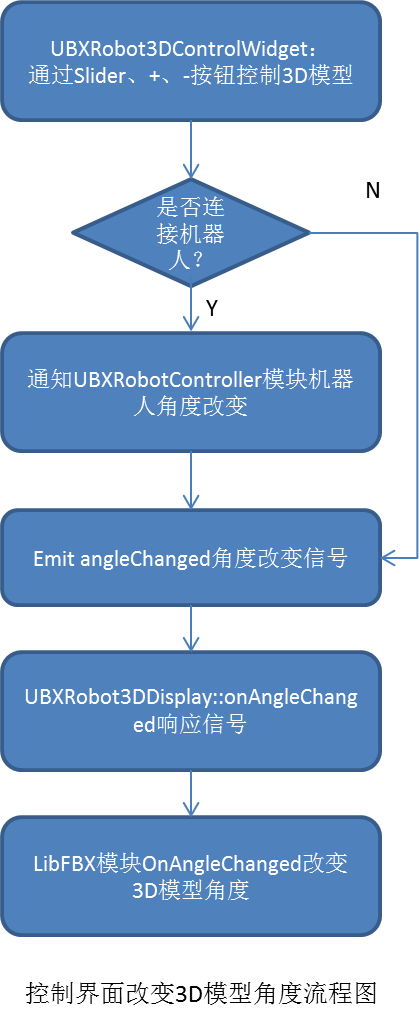
IUBXRobotViewControlNotify接口。作为UBXRobot3DWidget、

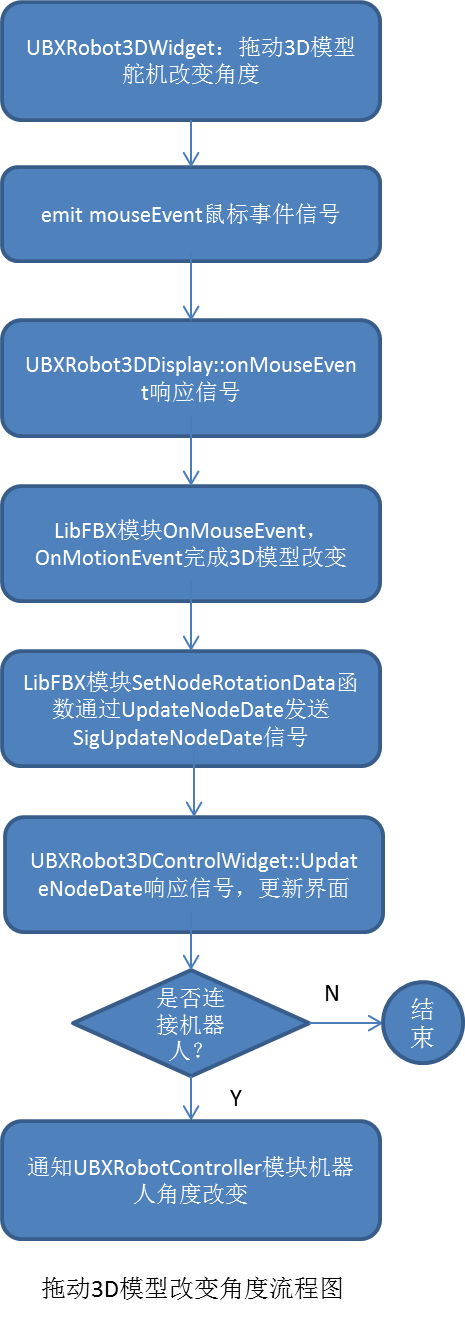
UBXRobot3DControlWidget对象的主窗体，同时转发外部接口调用。

* + 1. **注意事项：**
       - Qt多线程编程，尽量使用信号和槽进行接口设计，尤其是与UI的通信，否则会引起程序崩溃。
       - Qt OpenGL多线程编程，stackoverflow上有人提供了基本思路

<http://stackoverflow.com/questions/18345356/qglwidget-in-another-thread-what-is-the-documentation-referring-to>

* + - * Qt帮助文档中：搜索QGLWidget，请详细阅读对于此类的说明，如果能完全理解并按照说明实现，Qt绘制OpenGL对象就能掌握。
    1. 程序关键流程图





### mp3proj模块说明

* 环境配置

对mp3波形的解析需要借助第三方库，编译之前请确保安装以下库。

* + 1. mpg123：官网<http://www.mpg123.org>，

sourceforge地址<https://sourceforge.net/projects/mpg123/>

* + 1. sniffle ：wav文件的读写需要依赖sniffle库

libsndfile安装教程<http://macappstore.org/libsndfile/>

需要提前安装brew

http://webmedia.blog.163.com/blog/static/416695020123261226695/

* 功能说明

mp3proj模块负责音乐波形的解析和绘制，并提供接口与UBXMotionEditor进行通信。

* 设计说明

基本接口保持与原windows下一致，但绘制和事件通知也一并在mp3proj模块中实现。

* + 1. **Wave类：**mp3波形数据类，负责解析mp3文件为波形数据。
    2. **WaveLayer类：**波形绘制类，内部封装绘制算法。
    3. **MP3Widget类：**继承QWidget，提供绘制波形的窗体，提供更新数据接口（响应UBXMotionEditor动作数据的改变）。
    4. **mp3Container类：**继承QWidget，内部整合MP3Widget对象和QScroolBar滚动条对象。
    5. **MainWidget类：**继承QWidget，实现ILayerOptionHandler接口，是Mp3模块的主窗体类，并负责与UBXMotionEditor接口进行通信
    6. **MP3Player类：**负责mp3波形文件的播放
    7. **注意事项：**
       - MP3文件的播放直接使用QMediaplayer，是异步进行的。期间遇到mp3选定位置开始播放总是不成功，解决办法和原因已经记录在MP3Player实现类中，后续如果再有类似问题，可能需要考虑其他的解决方案。如找一个支持播放音视频的第三方库，自己实现mp3音频数据的播放。
       - 一开始的版本直接播放mp3文件发现需要等待很久才有声音播放，猜测可能是系统内部还是需要将mp3文件解码为wav文件才进行播放导致。后来新增libsndfile库，直接保存wav文件，并让MP3Player类直接播放wav文件，速度略有提升。

### UBXActionLib模块说明

UBXActionLib模块中原先提供的http类有问题，因此另外实现了HttpTransfer和HttpDownload类完成Http请求。

* 功能说明

UBXActionLib模块负责获取后台服务器动作列表项，并提供显示、预览、下载等相关功能。

* 设计说明

此模块涉及到网络数据获取，因此一开始就设计为多线程程序结构，界面各类对象与线程对象都通过信号和槽的方式进行通信。

* + 1. **Httptransfer类**：http传输类，内部封装线程对象，主要提供动作列表相关的http请求。此类完成动作列表json数据的接收，以及动作预览图的下载。
    2. **HttpDownload类：**动作文件下载类，内部封装线程对象，负责动作文件的下载。
    3. **ActionLibWidget类：**动作库首页，CoverFlow显示宣传图片，显示热门动作。
    4. **PictureFlow类：**第三方的CorverFlow控件，用于显示首页最上面的几张宣传图。主要有可能需要修改的接口PictureFlowState::reposition()用于控制每个图片slider的角度等。具体的使用示例，可直接参看ActionLibWidget中对此类的使用。
    5. **ActionListsPage类：**动作列表页面，分页，分类型展示动作列表项。
    6. **FavoritesPage类：**我的收藏页面，分页，分类显示我的收藏。由于服务器不支持分页和分类查询我的收藏，因此这部分界面的显示，需要两次查询服务器，第一次是查询我的收藏列表，服务器返回我的收藏信息（json格式，但不适合于在PC上显示完整信息），再查询对应动作ID的详细信息，构成PC可完整显示的ActItemData，再通知界面显示，后续如果服务器支持我的收藏根据PC端返回与动作列表查询一致的信息时，这块需要改为与动作列表页面相同的处理方式。
    7. **ActionDetailWidget类：**动作详情页，显示动作详细信息，QWebView在Mac OS下无法播放视频，暂时未找到解决办法。
    8. **ActionListItemPreview类：**动作项列表控件（一竖列），主要供动作列表页使用。
    9. **HomePageItemPreview类**：动作列表控件，主要供首页显示动作项，注意与ActionListItemPreview的区别，主要第一列是否显示checkbox，其他的功能和接口基本一致。
    10. **DownLoadItemDelegate类：**继承QStyledItemDelegate，提供绘制TableWidget的代理，自定义绘制对象，本身并不复杂，可直接参看源码。
    11. **CommentItemDelegate类：**继承QStyledItemDelegate，提供给动作详情页，评论回复控件。但由于服务器暂时不提供回复相关功能，PC端暂时不实现评论。若将来服务器支持了，此类即可用上。
    12. **CActUpdateMangr类：**管理类，全局唯一，负责管理httptransfer、httpdownload类对象，并提供相关接口给外部监听信号，更新数据。
    13. **注意事项：**
        - HttpTransfer、HttpDownload都是要通过moveToThread移到线程中执行的对象，因此其提供的接口都涉及到多线程编程，建议都以Qt信号和槽的方式进行通信，注意编译窗口的提示，若有相关线程的警告或者输出

“Cannot create children for a parent that is in a different thread”类似这样的提示，注意要深刻理解Qt的线程模型。

* + 1. 程序关键流程图

动作列表查询，动作下载，我的收藏基本流程比较相似，因此只列表查询动作列表流程图，其他的流程可类推。

