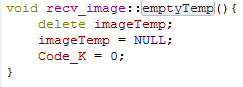


接收图片窗口是节点窗口的私有成员（即一个节点窗口只有一个接收图片窗口。目前界面应该一个节点在同一时间只能就收一个节点发来的图片），当该节点收到图片头数据或者图片data数据，就会自动show出这个窗口。

Receive again：调用槽函数，删除

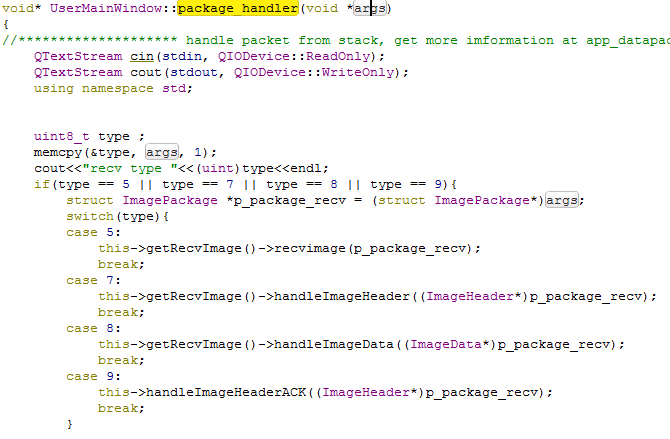


节点窗口的处理：根据数据包的类型，调用接收窗口不同的函数来处理。

一 9是收到ACK，停止节点窗口的超时ACK计时器。

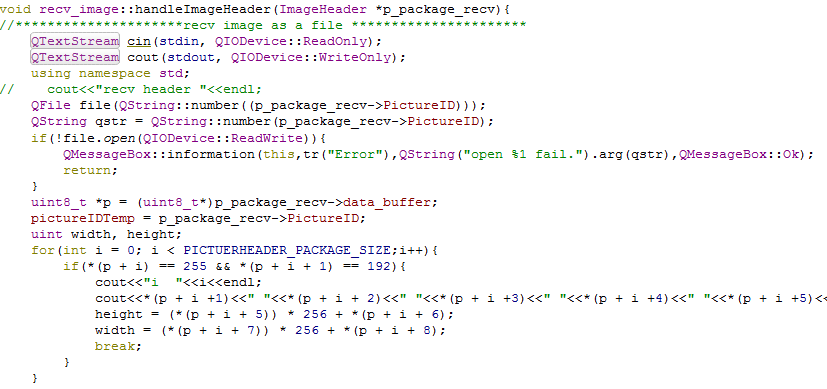
二 578交给接收图片窗口处理。（5应该是还没区分图片头部和data部分时写的）

三 只要节点启动，无论受到图片头部还是data，都是show出接收窗口。（接收窗口是节点窗口的成员）

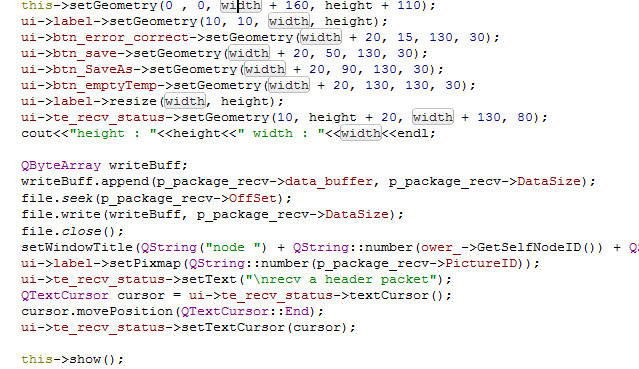


**节点收到图片头：（jpg头，7）**

处理图片头部：先生成图片文件，获得图片大小（这里通过找到头部的SOFmarker，0xFFC0，这个字段中有图片高和宽）。



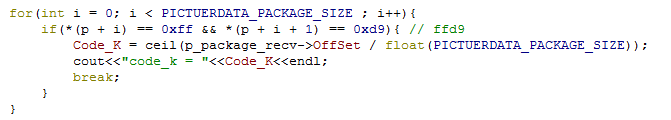
然后根据图片的大小，设置界面的控件位置。把新的数据加到文件中，显示文件内容。都是先存到这个图片文件中（文件名是pictureID，二进制格式），然后再用pixmap来显示。



**节点收到图片数据：（jpg头，8）**

写到文件里面，然后再重新显示一次这个文件。这里可以找到EOI字段，不过只是cout出来了。（感觉PixMap应该封装好了图片显示的一些功能，比如没有EOI的时候就自动添加，EOI之后还有数据的话就忽略。因此可能，虽然没有改变文件本身的内容，显示就没有问题。）

这里code\_k应该是收到了多少个数据包（偏移除以数据包大小，因为无论图片头部还是数据包，整个包的大小是一样的。）



这个函数是类型为5时调用的函数，应该是一开始没有把头和数据分开来发时调用的。



**Error Correct：**rs纠错，直接调用rs\_decode\_image.py函数解码，生成文件是pictureID.re,然后显示这个文件。

