1.

```
    #!/usr/bin/python3

    # Need to install python3-pyqt5.qtx11extras
    import sys
    from PyQt5.QtGui import  *
    from PyQt5.QtWidgets import *
    from PyQt5.QtCore import *
    from PyQt5.QtX11Extras import *

    #Making the label for the image
    class XMagQLabel(QLabel):
        def setLabel(self, label):
            self.label=label

        def setqimage(self, qimage):
            self.qimage=qimage

        #Used for making the image where ever the mouse is located
        def mouseMoveEvent(self, event):
            # Check bounds
            if event.x()>=self.qimage.width():
                return
            if event.y()>=self.qimage.height():
                return
            if event.x()<0:
                return
            if event.y()<0:
                return
        #Used to find what the pixel color is based on mouse over image
            c=self.qimage.pixelColor(event.pos())
            t="Color [{x:3d},{y:3d}] ({r:2X},{g:2X},{b:2X},{a:2X})".format(x=event.x(),y=event
.y(),r=c.red(),g=c.green(),b=c.blue(),a=c.alpha())
            self.label.setText(t)

    #Set up code for the creation of widget
    #This is the widget specification code
    class XMagQWidget(QWidget):
        replace=False

        def setParent(self,parent):
            self.parent=parent

        def mouseReleaseEvent(self,event):
            if self.replace==False:
                return
            self.replace=False
            self.releaseMouse()
            self.parent.update(event.globalPos())

        def update(self,tf):
            self.replace=tf

    #This is the main class for the actual function of the widget
    class pyqtmag:
        #This is the main function of the widget, where all sub-widgets are created
        def __init__(self):
        #The size and specs of the main large widget
            self.app=QApplication(sys.argv)
            self.w=XMagQWidget()
            self.w.setGeometry(100,100,250,200)
            self.w.setWindowTitle('PyQtMag')
            self.screen=self.w.screen()

            self.layout=QGridLayout()
            self.w.setLayout(self.layout)

        #Push button for quiting out of widget
            self.quit=QPushButton("Quit")
            self.quit.clicked.connect(self.quit_cb)

            self.layout.addWidget(self.quit,1,1,1,1)

        #Replace button to grab a new image
            self.replace=QPushButton("Replace")
            self.replace.clicked.connect(self.replace_cb)

            self.layout.addWidget(self.replace,1,2,1,1)

        #This is a "label" or the image which was taken
        #This image can be configured, such as magnified
            self.label=XMagQLabel()
            self.qimage=QImage()
            self.layout.addWidget(self.label,2,1,1,2)

        #Makes a label at bottom of widget and when mouse is clicked
        #on photo, the pixel color is shown
            self.info=QLabel("Color")
            self.layout.addWidget(self.info,3,1,1,2)

            self.label.setLabel(self.info)
            self.label.setqimage(self.qimage)

        #New button added to save image that has been grabbed
            self.save=QPushButton("Save Image")
            self.save.clicked.connect(self.save_image_cb)

            self.layout.addWidget(self.save,1,3,1,1)

        #A zoom control has been added with a slider.
            self.zoom_control = QSlider(Qt.Vertical)
            self.zoom_control.setMinimum(1)
            self.zoom_control.setMaximum(200)
            self.zoom_control.setSliderPosition(100)
            self.zoom_control.valueChanged.connect(self.zoom)

            self.layout.addWidget(self.zoom_control,1,4,4,1)

            self.rootwindow=QX11Info.appRootWindow()

        #Show the widget
            self.w.show()

            self.window=self.app.allWindows()
            self.w.setParent(self)

        #Function for starting the app
        def start(self):
            self.app.exec_()

        #Function for quiting the application (used for the button)
        def quit_cb(self):
            quit()

        #Function for the save button, saves to the directory as a PNG
        def save_image_cb(self):
            self.qimage.save("/home/iviolette/Pictures/Screen_Grab.png","png")

        #Function for replace button, cursor is used as a box
        #Grabs the location of the image at the mouse location
        def replace_cb(self):
            # Allow mouse to move around to grab a spot
            pm=QPixmap('cursor.png')
            cursor=QCursor(pm,hotX=0,hotY=0)
            self.w.grabMouse(cursor)
            self.w.update(True)

        #This function actually grabs the image from the cursor location
        #and wraps it up into a scaled image for the widget.
        def update(self,pos):
            r=QRect(pos,QSize(50,50))
            ri=self.screen.geometry().intersected(r)
            x=ri.x()
            y=ri.y()
            w=ri.width()
            h=ri.height()
            self.pixmap=self.screen.grabWindow(self.rootwindow,x,y,w,h)
            self.size=QSize(200,200)
            self.pixmap_scaled=self.pixmap.scaled(self.size)
            self.qimage=self.pixmap_scaled.toImage()
            self.label.setPixmap(self.pixmap_scaled)
            self.label.setqimage(self.qimage)

        #Function for zoom, takes the image and scales it, returns it as a label
        def zoom(self, level):
            zoom_quantity = level / 100
            self.label.setScaledContents(True)
            self.label.setFixedSize(self.label.pixmap().size() * zoom_quantity)

    #Start the application
    if __name__=='__main__':
        app=pyqtmag()
        app.start()
```

2.

```
    enscript -b '$n %E %C|$%|Isaac Violette' -T 4 -M Letter -p HW11.ps HW11
```