LAB 10: Cassandra Cabrera and Mike Menendez **Due:** March 2, 2020

Summary:

This assignment was of great interest to us both as being able to create a quick GUI for our personal one-off scripts is extremely handy. Granted the hex color portion wasn't the main take away for us whereas learning the core concepts behind drawing a GUI that hooks into the the native system was fun to learn as it not something that is part of the core cirriculum at CSUMB.

The main pain points that we experienced were around knowing which portions of the Qt framework to import rather than loading in the entire framework which is inefficient. The other pain point for us was the bonus with the color wheel due to it requiring us to think generically. After playing with this idea for awhile, we then decided to look into the docs in more depth and discovered the page for the QColorDialog which natively implemented the behavior and functionality that we were looking for.

Code:

Task 1 & 2

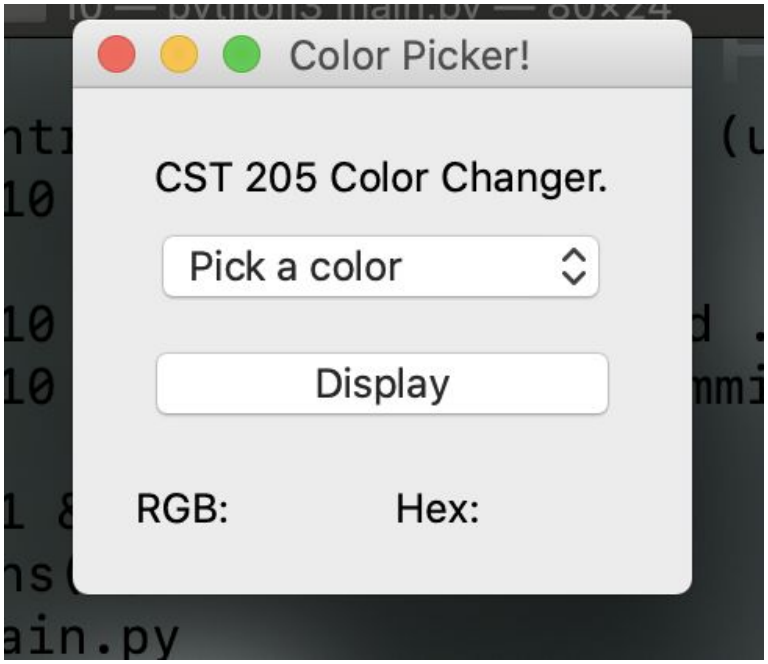
```
'''
Authors: Cassandra Cabrera & Mike Menendez
Date: March 2, 2020
Purpose: Task 1 & 2 of Lab 10
    - Create a dropdown of colors
    - create button that when clicked will display chosen color
'''

import sys
from PyQt5.QtWidgets import (QApplication, QWidget, QLabel, QPushButton,
                             QLineEdit, QHBoxLayout, QVBoxLayout, QComboBox)
from PyQt5.QtCore import pyqtSlot, Qt
from PyQt5.QtGui import QPainter, QColor, QPen
from PyQt5.QtGui import QIcon

#defines my rgb values
my_rgb_dict = {"Pick a color" : "", "red" : (255,0,0), "orange" : (255,127,0),
"yellow" : (255,255,0), "green" : (0,255,0), "blue" : (0,0,255),
"purple" : (75,0,130), "lilac" : (143,0,255)}

#defines my hex values
my_hex_dict = {"Pick a color" : "", "red" : "#FF0000", "orange" : "#FF7F00",
"yellow" : "#FFFF00", "green" : "#FFFF00", "blue" : "#0000FF",
"purple" : "#4B0082)", "lilac" : "#8F00FF"}

#sets up the window for the selected color to be displayed
class Color(QWidget):
    def __init__(self, color):
        super().__init__()
        self.title = 'Your Color Selection!'
        self.left = 10
        self.top = 10
        self.width = 440
        self.height = 280
        self.color = color
        self.initUI()

    def initUI(self):
        super().__init__()
        self.setWindowTitle(self.title)
        self.setGeometry(self.left, self.top, self.width, self.height)

        # Set window background color
        self.setAutoFillBackground(True)
        p = self.palette()
        p.setColor(self.backgroundRole(),
            QColor(self.color[0],self.color[1],self.color[2]))
        self.setPalette(p)

        self.show()
```

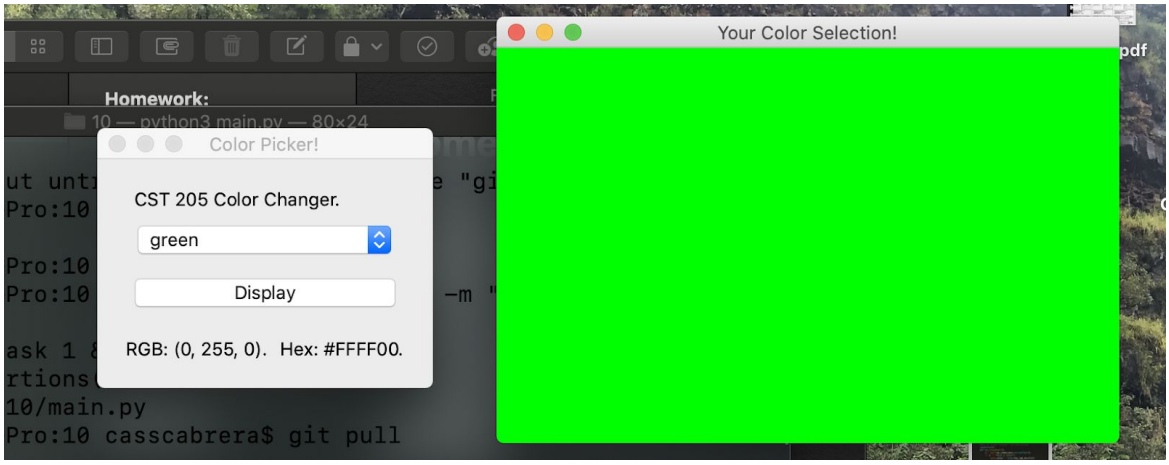
Line 47, Column 13

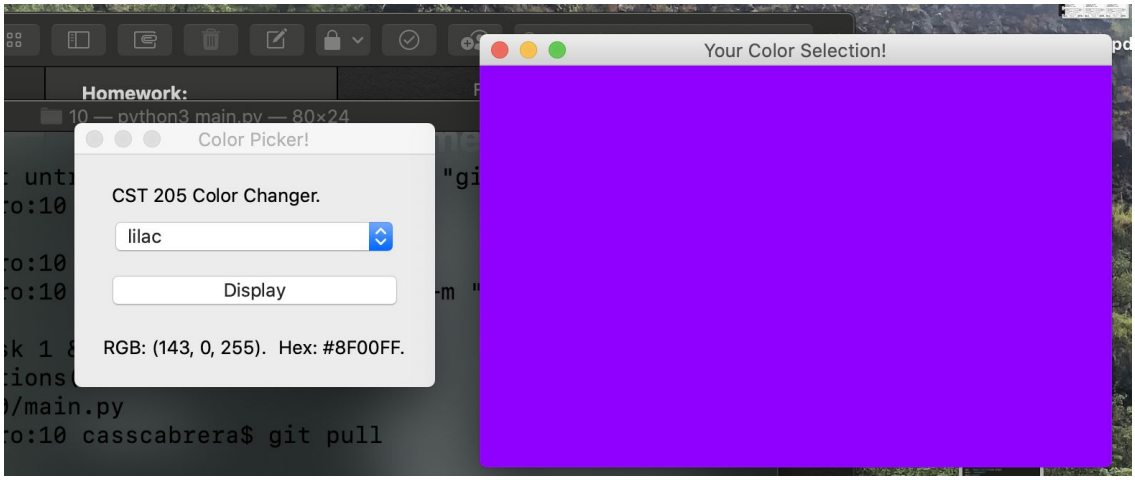
```
        self.show()

#sets up the main window
class Window(QWidget):
    def __init__(self):
        super().__init__()

        #creates the dropdown list
        self.my_combo_box = QComboBox()
        self.my_combo_box.addItems(my_rgb_dict)
        self.my_title = QLabel("CST 205 Color Changer.")
        self.my_rgb = QLabel("RGB: ")
        self.my_hex = QLabel("Hex: ")
        self.my_btn = QPushButton("Display",self)

        self.my_btn.clicked.connect(self.on_click)

        tbox = QVBoxLayout()
        hbox = QHBoxLayout()
        vbox = QVBoxLayout()

        hbox.addWidget(self.my_rgb)
        hbox.addWidget(self.my_hex)

        vbox.addWidget(self.my_title)
        vbox.addWidget(self.my_combo_box)
        vbox.addWidget(self.my_btn)

        tbox.addLayout(vbox)
        tbox.addLayout(hbox)

        self.setLayout(tbox)
        self.my_combo_box.currentIndexChanged.connect(self.update_ui)
        self.setWindowTitle("Color Picker!")

    #sets the text based on color
    @pyqtSlot()
    def update_ui(self):
        my_text = self.my_combo_box.currentText()
        my_index = self.my_combo_box.currentIndex()
        self.my_rgb.setText(f'RGB: {my_rgb_dict[my_text]}.')
        self.my_hex.setText(f'Hex: {my_hex_dict[my_text]}.')

    #the on click listener for the button
    @pyqtSlot()
    def on_click(self):
        i = self.my_combo_box.currentText()
        if i != "Pick a color":
            self.color = Color(my_rgb_dict[i])
```

```
    #the on click listener for the button
    @pyqtSlot()
    def on_click(self):
        i = self.my_combo_box.currentText()
        if i != "Pick a color":
            self.color = Color(my_rgb_dict[i])
            self.color.show()


app = QApplication(sys.argv)
main = Window()
main.show()
sys.exit(app.exec_())
```

Results from Task 1 & 2







Task3: We are both working on our note for the quiz.

Bonus Task

```
'''
Authors: Cassandra Cabrera & Mike Menendez
Date: March 2, 2020
Purpose: Bonus Task of Lab 10
    - Create a color picker widget
    - When selected display color with RGB and hex values
'''
import sys
from PyQt5.QtWidgets import QApplication, QWidget, QPushButton, QColorDialog
from PyQt5.QtGui import QIcon
from PyQt5.QtCore import pyqtSlot
from PyQt5.QtGui import QColor

class DisplayColor(QWidget):

    def __init__(self):
        super().__init__()
        self.title = 'CST205: The color you selected!'
        self.left = 10
        self.top = 10
        self.width = 320
        self.height = 200
        self.initUI()

    def initUI(self):
        self.setWindowTitle(self.title)
        self.setGeometry(self.left, self.top, self.width, self.height)

        color = QColorDialog.getColor()

        if color.isValid():
            self.initColor(color)

        self.show()

    def initColor(self,color):
        super().__init__()
        self.setWindowTitle(self.title)
        self.setGeometry(self.left, self.top, self.width, self.height)

        self.color = color

        # Set window background color
        self.setAutoFillBackground(True)
        p = self.palette()
        p.setColor(self.backgroundRole(),
            QColor(self.color))
        self.setPalette(p)

        self.show()


if __name__ == '__main__':
    app = QApplication(sys.argv)
    ex = DisplayColor()
    sys.exit(app.exec_())
```

Results from Bonus: