**학교 홍보용 디지털 사이니지 만들기**

2019067501 나노광전자학과 박정현

2020076317 ICT 융합학부 지승윤

2022005269 ICT 융합학부 어선영

2022088031 ICT 융합학부 배수연

<구현 코드>

import os

import sys

import queue

import platform

from PyQt5 import QtWidgets, QtGui, QtCore

import vlc

class MiniPlayer(QtWidgets.QMainWindow):

"""Stripped-down PyQt5-based media player class to sync with "master" video.

"""

def \_\_init\_\_(self, data\_queue, master=None):

QtWidgets.QMainWindow.\_\_init\_\_(self, master)

self.setWindowTitle("Mini Player")

self.statusbar = self.statusBar()

self.statusbar.showMessage("Ready")

# Create a basic vlc instance

self.instance = vlc.Instance()

self.media = None

# Create an empty vlc media player

self.mediaplayer = self.instance.media\_player\_new()

self.init\_ui()

self.open\_file()

self.timer = QtCore.QTimer(self)

self.timer.setInterval(10)

self.timer.timeout.connect(self.update\_ui)

self.data\_queue = data\_queue

self.timer.start()

# 추가 점수 2. Mini\_player가 항상 제일 위에 있도록 설정 - 창을 항상 위에 유지하기 위해 창 플래그를 설정합니다.

self.setWindowFlag(QtCore.Qt.WindowStaysOnTopHint)

def init\_ui(self):

"""Set up the user interface

"""

self.widget = QtWidgets.QWidget(self)

self.setCentralWidget(self.widget)

# In this widget, the video will be drawn

if platform.system() == "Darwin": # for MacOS

self.videoframe = QtWidgets.QMacCocoaViewContainer(0)

else:

self.videoframe = QtWidgets.QFrame()

self.palette = self.videoframe.palette()

self.palette.setColor(QtGui.QPalette.Window, QtGui.QColor(0, 0, 0))

self.videoframe.setPalette(self.palette)

self.videoframe.setAutoFillBackground(True)

self.vboxlayout = QtWidgets.QVBoxLayout()

self.vboxlayout.addWidget(self.videoframe)

self.widget.setLayout(self.vboxlayout)

def open\_file(self):

"""Open a media file in a MediaPlayer

"""

filename = "ERICA.mp4"

# getOpenFileName returns a tuple, so use only the actual file name

self.media = self.instance.media\_new(filename)

# Put the media in the media player

self.mediaplayer.set\_media(self.media)

# Parse the metadata of the file

self.media.parse()

# Set the title of the track as the window title

self.setWindowTitle("{}".format(self.media.get\_meta(0)))

# The media player has to be 'connected' to the QFrame (otherwise the

# video would be displayed in it's own window). This is platform

# specific, so we must give the ID of the QFrame (or similar object) to

# vlc. Different platforms have different functions for this

if platform.system() == "Linux": # for Linux using the X Server

self.mediaplayer.set\_xwindow(int(self.videoframe.winId()))

elif platform.system() == "Windows": # for Windows

self.mediaplayer.set\_hwnd(int(self.videoframe.winId()))

elif platform.system() == "Darwin": # for MacOS

self.mediaplayer.set\_nsobject(int(self.videoframe.winId()))

# Start playing the video as soon as it loads

self.mediaplayer.play()

def update\_ui(self):

self.update\_statusbar()

try:

val = self.data\_queue.get(block=False) # 데이터 큐에서 다음 값을 가져옵니다.

except queue.Empty:

val = None # 큐가 비어있을 경우 val을 None으로 설정합니다.

if val is not None:

if val == '<':

self.mediaplayer.set\_rate(self.mediaplayer.get\_rate() \* 0.5) # 재생 속도를 절반으로 줄입니다.

return

if val == '>':

self.mediaplayer.set\_rate(self.mediaplayer.get\_rate() \* 2) # 재생 속도를 두 배로 증가시킵니다.

return

if val == 'P':

self.mediaplayer.play() # 재생을 시작합니다.

return

if val == 'p':

self.mediaplayer.pause() # 재생을 일시 정지합니다.

return

if val == 'S':

self.mediaplayer.stop() # 재생을 중지합니다.

return

val = int(val)

if val != self.mediaplayer.get\_time():

self.mediaplayer.set\_time(val) # 재생 시간을 받은 값으로 설정합니다.

state = self.mediaplayer.get\_state() # 현재 미디어 플레이어의 상태를 가져옵니다.

if state == vlc.State.Ended: # 추가 점수 1. Mini\_player 종료 전까지 youtube 영상 무한 반복

self.mediaplayer.stop() # 미디어 재생이 종료된 경우 재생을 중지합니다.

self.mediaplayer.play() # 재생을 다시 시작합니다.

QtCore.QTimer.singleShot(10, self.update\_ui) # 10밀리초 후에 다음 update\_ui 호출을 예약합니다.

def update\_statusbar(self):

mtime = QtCore.QTime(0, 0, 0, 0)

time = mtime.addMSecs(self.mediaplayer.get\_time())

self.statusbar.showMessage(time.toString())

def main():

"""Entry point for our simple vlc player

"""

app = QtWidgets.QApplication(sys.argv)

data\_queue = queue.Queue()

player = MiniPlayer(data\_queue)

player.show()

player.resize(480, 480)

sys.exit(app.exec\_())

if \_\_name\_\_ == "\_\_main\_\_":

main()

<코드 변경 내용>

1. **파일 선택 기능 변경:**

기존의 mini\_play.py코드에서는 파일 선택을 위해 **QFileDialog.getOpenFileName()**을 사용하여 파일을 열었습니다. **open\_file** 메서드를 수정하여 파일을 선택하기 위해 **QtWidgets.QFileDialog.getOpenFileName** 대신 **filename** 변수에 직접 파일 경로를 설정했습니다.

filename = "ERICA.mp4"

1. **추가 점수 1. 영상 무한 반복:**

변경한 코드에서는 **vlc.State.Ended** 상태에 도달하면 미디어 재생을 중지하고 다시 시작하는 방식으로 영상을 무한 반복하는 기능을 추가했습니다. **update\_ui** 메서드에 해당 로직을 추가했습니다. **update\_ui** 메서드에 미디어 플레이어 상태를 확인하는 코드를 추가했습니다. 상태가 **vlc.State.Ended**인 경우, 미디어 플레이어를 정지하고 다시 재생합니다. 이 코드를 통해, 동영상이 끝에 도달했을 때 다시 시작됩니다.

if state == vlc.State.Ended: #영상의 끝에 도달한 경우

self.mediaplayer.stop() # 미디어 재생이 종료된 경우 재생 중지

self.mediaplayer.play() # 재생 다시 시작

1. **추가 점수 2. Mini Player가 항상 최상위에 유지;**

변경된 코드에서는 **setWindowFlag** 메서드를 사용하여 **MiniPlayer** 창이 항상 최상위에 나타나도록 설정했습니다. **QtCore.Qt.WindowStaysOnTopHint** 플래그를 사용했습니다. 아래 코드를 추가하여 창이 항상 최상위에 유지되도록 창 플래그를 설정했습니다. **self.setWindowFlag(QtCore.Qt.WindowStaysOnTopHint)**를 통해 플레이어 창이 항상 다른 창 위에 유지됩니다.

self.setWindowFlag(QtCore.Qt.WindowStaysOnTopHint)

1. **큐의 효과적인 데이터 처리**

**update\_ui** 메서드를 수정하여 데이터 큐가 비어있는 경우에 대해 더 효과적으로 처리하도록 변경했습니다. 큐가 비어있을 때 예외를 발생시키지 않고 **val**에 **None**을 할당하고 별도로 처리하도록 수정했습니다. 이렇게 함으로써 큐가 비어있을 때 예외가 발생하는 것을 미리 방지했습니다.

except queue.Empty:

val = None # 큐가 비어있을 경우 val을 None으로 설정