Python course

Objective

This course provides an introduction into advanced features of Python such as classes and objects, installing and using third party libraries (python-Qt, python-vlc, ...) and some advanced programming constructs such as state engines and how to coordinate asynchronous program flows.

The course is highly practical - we show the concepts by designing a simple media player application which implements all the concepts we teach. To make things more interesting we implement two different UI types, a local GUI and a web GUI.

I'm not the best imaginative namegiver, but for reference, let's call our wonderful product *MyPlayer*. Any ideas for a better name?

Prerequisites

Windows/Linux (advanced user)

Python (basic programming skills: Variables & statements, control structures, functions).

Basic Git usage (Setting up repo, commit, branch)

A lot of persevearance, googling and pushing through! Explanations are sometimes sparse on purpose!

Requirements

Average Laptop / Desktop machine plus peripherals.

OS: Windows (10) or (K)Ubuntu Linux 18

Python 3

Python IDE (PyClipse)

Programmer friendly editor (Windows: Notepad++, Linux: Kate or GEdit)

TortoiseGit (Windows) + deps, Just Git (Linux)

Agenda

1. Installation
    1. Python 3
    2. PyClipse
    3. python-qt
    4. python-vlc
    5. Git + friends
2. Program structure
    1. Components and their purpose
3. The GUI
    1. Visual design
    2. Architecture: Delegate pattern
    3. Logic, GUI-Only
4. The controller
    1. State machine
    2. Integrating the GUI
5. The backend
    1. The audiosystem (vlc) - Introduction
        1. Instantiating
        2. Functions
        3. Events
    2. Integrating vlc into backend
    3. Playlist
6. The controller II
    1. Integrating the backend into controller
    2. Integrating volume slider
    3. Passing runtime info to GUI (Track position/info)
7. Web GUI, visual design
8. Web service
    1. Web GUI delegate
    2. Comms channel to web client (web sockets)
9. Integrating WEB GUI
    1. The Javascript library
    2. Trying it out

Lesson 1: Installation (2019-08)

Objective: Set up the Development environment: Python+libs, git, IDE

Linux

We use KUbuntu, 64 bit, and assume a working system. For the installation you need an open Terminal (emulator). KUbuntu offers you a program for that (aptly named **`Konsole`**). There you will type everything.

VLC player

That's for your media player backend to play your muzac.

```
~:$  sudo apt install vlc
```

Qt

This is the framework which puts your local GUI on your screen.

```
~:$  sudo apt install qt4-designer
```

git

A source code management system. You need it to keep track of your project.

```
~:$  sudo apt install git
~:$  sudo apt install git-gui
~:$  sudo apt install git-man
```

Python + libs

You also need python plus some libs to bind qt and vlc into it.

```
~:$  sudo apt install python3
~:$  sudo apt install pyside-tool
~:$  sudo apt install python-qt4
~:$  sudo apt install python3-pip
~:$  sudo pip3 install python-vlc
```

Java

Needed for LiClipse. We'll be using the openJDK which ships with KUbuntu. Unfortunately the official Oracle Java8 can't be installed via repository, and it seems OpenJDK is enough. Check whether you have Java installed. On my laptop I get:

```
~:$  java -version

~:$  java version "1.8.0_201"

~:$  Java(TM)  SE  Runtime  Environment  (build  1.8.0_201-b09)
     Java HotSpot(TM) 64-Bit Server VM (build 25.201-b09, mixed mode)
```

If it tells you it doesn't have java, you need to install it:

```
~:$  sudo apt install default-jdk
```

Your system will download and install the standard Kubuntu JDK

Python IDE (LiClipse)

We'll be using LiClipse. It comes as a trial version with full functionality. After 14 days you will get a tender reminder (popup box) that you really *should* buy a license (software will continue to work though). It will cost $80, and I'd recommend buying it as it supports the developer (devs do need to eat, too - even if they say that their food is code).

Head over to https://www.liclipse.com/download.html. You'll probably have a 64 bit version of KUbuntu, so download the 64 bit version. You will get a file with the ending `.tar.gz` (e.g. `liclipse_5.2.4_linux.gtk.x86_64.tar.gz`). For your info - that's a compressed tar archive.

Once the file has downloaded, make yourself a directory `/home/your-user-name/bin/liclipse` and copy the file into that one. Then

- open the folder in Dolphin
- right click it
- choose `Extract -> Extract archive here, autodetect subfolder`

This will create another sub folder. To start LiClipse `cd` into that sub folder and execute `LiClipse`. (Hint) - you can make that a menu item in your KDE startup menu.

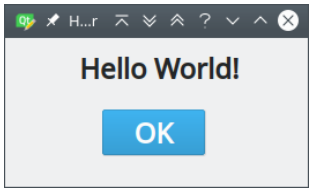
⇒ LiClipse comes with git support inbuilt so you can use git from inside your dev environment. Outside of LiClipse you can use the TortoiseGit features directly inside the Windows explorer, via right click on a (repository) folder.

Windows (10)

Whilst we could install everything directly into Windows there's a better way by setting up a virtual machine with a Kubuntu guest and then to use that as development machine. For the VM environment you can use VMWare (expensive, but if you have it, why not) or Virtualbox. With respect to setting up a VM and installing Linux, this would exceed the scope of this guide. Suffice to say that you should give the VM a harddisk with at least 30 GB and at least 4GB of memory. Also, the VM guest (i.e. VM thing you set up) needs a working connection to the internet and a

working audio output. When you have the base system working, follow the Linux part of this chapter (above) to get your dev environment ready.

Task: Using `QtDesigner` and `pyuic`, make python put a helloworld window onto your screen! Your window should look something like this:



Lesson 2: Program structure

Objective: Show the general structure of the player program.

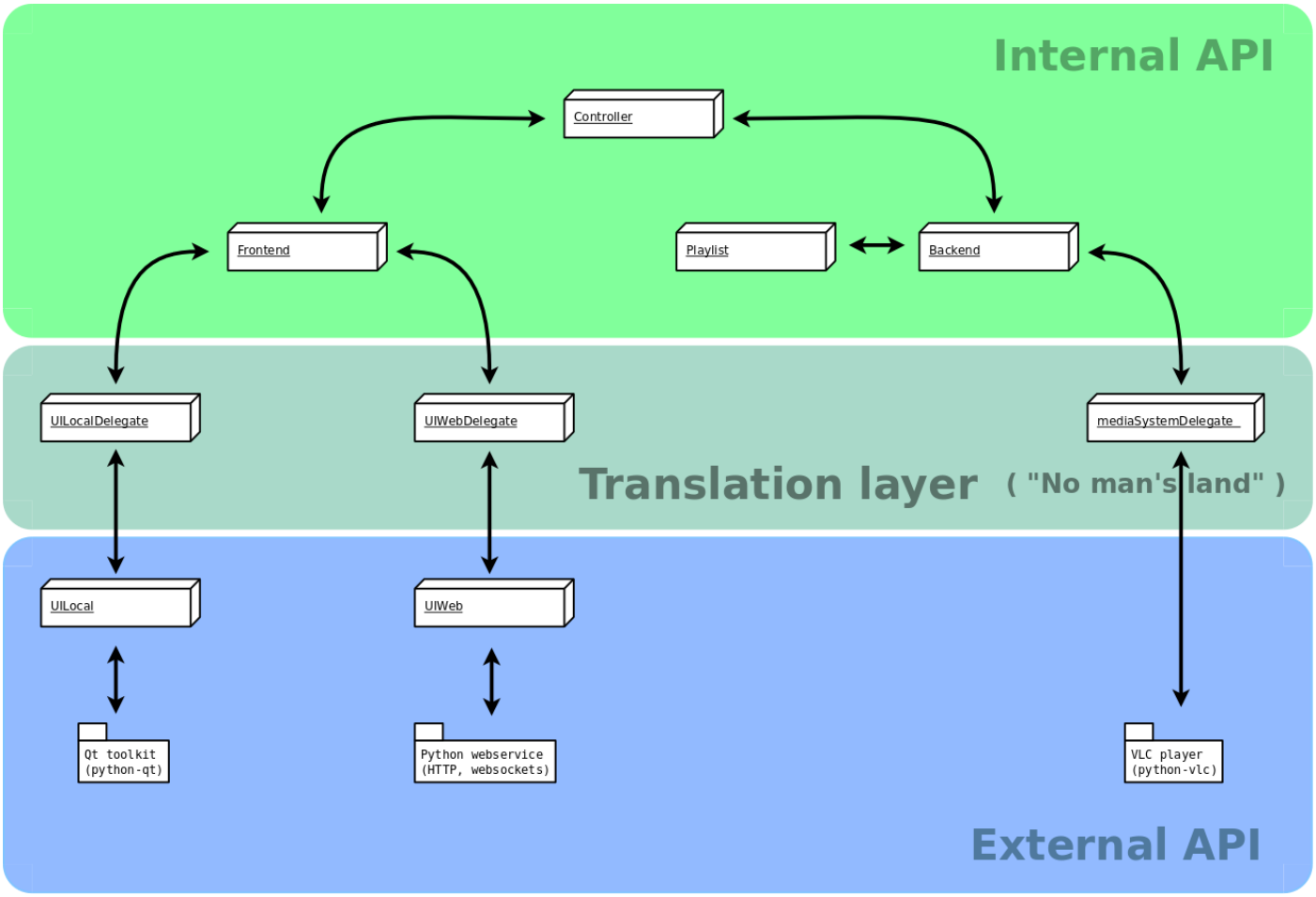


fig 1: Program overview

This chapter is a first get-to-know-me in broad strokes. We present the overall design of our player application. Subsequent chapters will go into much more detail, but this one wants to present the "big picture".

A good program design is hard to get to and needs experience! In today's interconnected world everything is super complicated! Design flaws are hard to beat once the project grows, so it's very important to get things right early! Several principles guide the development of MyPlayer.

- *Separation of concerns*. Example from the real world: A well designed company structure will result in employees who are competent in what they do and communicate well. Software design is similar! We take take a big problem, split it up into sub problems then write components that tackle those sub problems independently and work well together. Result: Program works well and can be maintained by dummies!

  MyPlayer realizes this principle by splitting up the whole application into smaller independent components. Furthermore, we isolate the components in separate layers. We have a system layer which operates the system parts (Local GUI, web GUI, VLC player), a control layer which represents the player's view of everything (controller, frontend and backend facades) and a translation layer (delegates) which translates what the control

components want to a "language" which the system components understand (and vice versa).

- Talking about "language" - *API, API, and again, API !* Did I say API? Good separation of concerns is only half the story. What good is the independence if the stuff doesn't talk to each other? We need a well defined interaction between the different components. In programmer's mumbo jumbo such interaction way is called *API* (*Application programming interface*). API thinking is all over the application.
  - Each component provides a clean interface - a bunch of methods to set getters and setters (methods that get and set stuff inside the component) and event handlers - which respond to other components wanting something. All methods follow a consistent naming scheme. For example, getters always start with `get`.
  - The components in the system layer all have very different requirements and thus very different methods. To present a more consistent API of the system components we use a layer of delegates that "translate" between the resp. system components and the control components. As a result the control components have a consistent view onto the system components. For example, the two frontends (Local GUI and web GUI) are very different in what they do and how they, but thanks to their respective delegates they look the same to the Frontend facade. This approach reduces complexity so we can pretty much plug any frontend into the running application and access them all same way.

    For the media player component (VLC) we also use a delegate; technically this isn't necessary, but we still use it in case we want to use a different media framework. Plus, it's just more consistent - now *any* system component talks to our application via a delegate. This means one less special case and one thing less to consider!

Getting a design right is the hardest part of software design. This setup is just an attempt to make the application easier to understand and maintain. We meet the design challenge by splitting the whole setup into separate components and providing a clearly defined interaction between those components.

Some code to layout

Pardon the weird language, but *layout* is jargonese for the way our source code is organized! In our longish introduction we talked about layers and modules. This reflects directly in how we set up the packages and modules - or in everyday speech - the folder structure and what source files we put there. I'm sure there are more interesting ways of laying out code, but I sugggest we organize it somewhat along the layers and modules we presented above. Fig. 2 shows what it looks like.

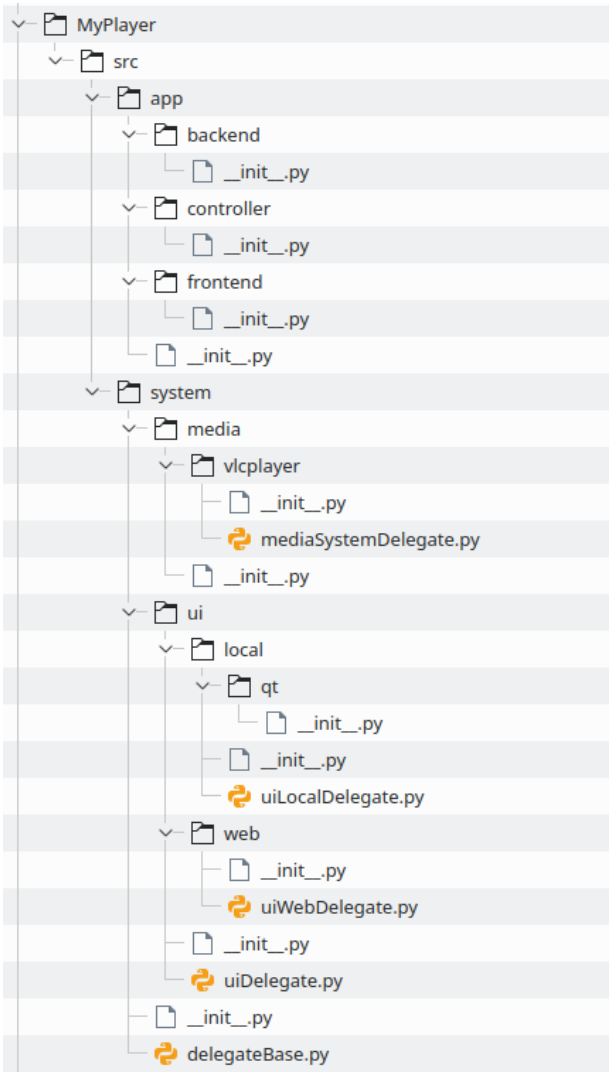


fig 2: Source layout

Noticed something? Blimey, where are those delegates???? If you look closely you will see them lumped with the respective system components! The local GUI has got a **`UILocalDelegate`**, the web UI has got a **`UIWebDelegate`** and the VLC driver has got a **`MediaSystemDelegate`**. The app layer components will use the delegates, and *only* the delegates to use the various system components. We'll see the details of all this later, but for now we note that the app layer talks in appanese with the delegates who translate that stuff into systemese for the system layer (and vice versa). If we pull this off correctly then it will be a breeze to integrate the system components into the app components!

The astute reader will have observed that there's no test code! That's right - but this guide is already demanding enough! Throwing in unit testing would be like adding rocks to your full grain muesli!

Task: In LiClipse, set up the project and create a structure of python modules implementing the program structure we explained.

As a plus, maybe make a GUI and let python execute it.

Using **`Konsole`**, put the project under git control.

Lesson 3: The GUI

Objective: Design the GUI (visually), build integration into player application and show a simplistic approach to the GUI logic

Let's start with the simple stuff up front. There's always time to complicate, so let's complicate later! We want to design the user interface first[1].

On your dev system (that virtual machine we set up in part 1) start QtDesigner (`Start menu -> Development`) and use it to create a user interface like this:

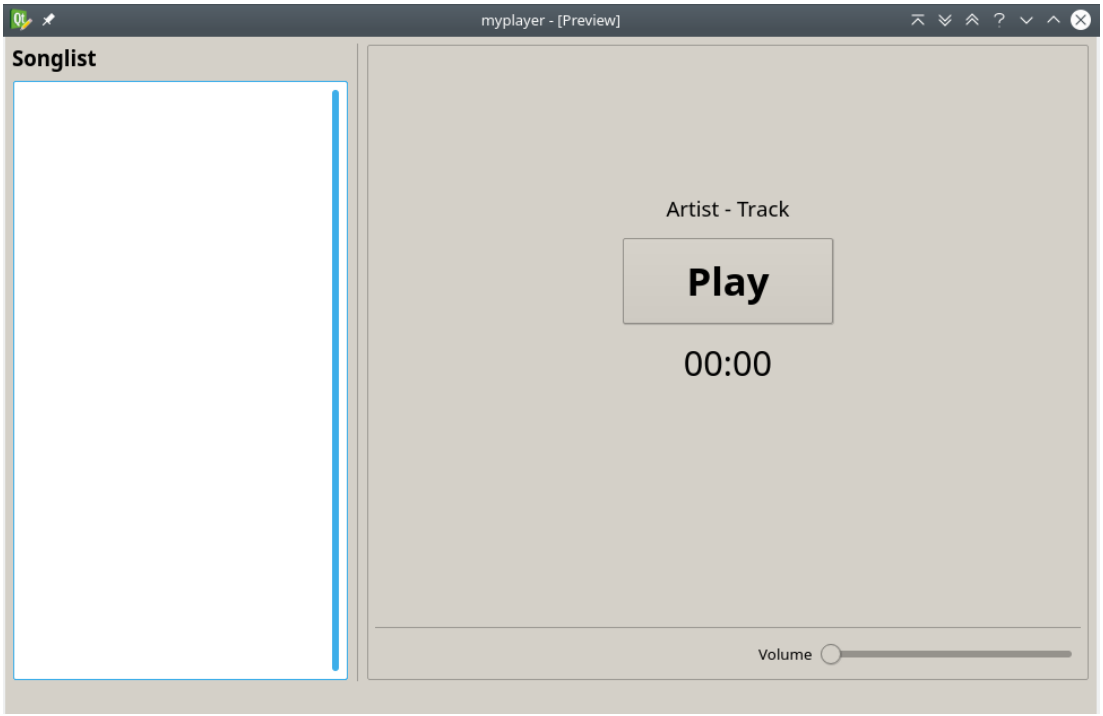


fig 3: GUI (local)

Some of the controls need specific names, so that the GUI works with the code we develop later. You can see those on my designer (fig. 4).

1 In my own software work I often found that I was *actually* more productive when I designed the GUI first. It sort of was the next best thing to designing my program customer focused.

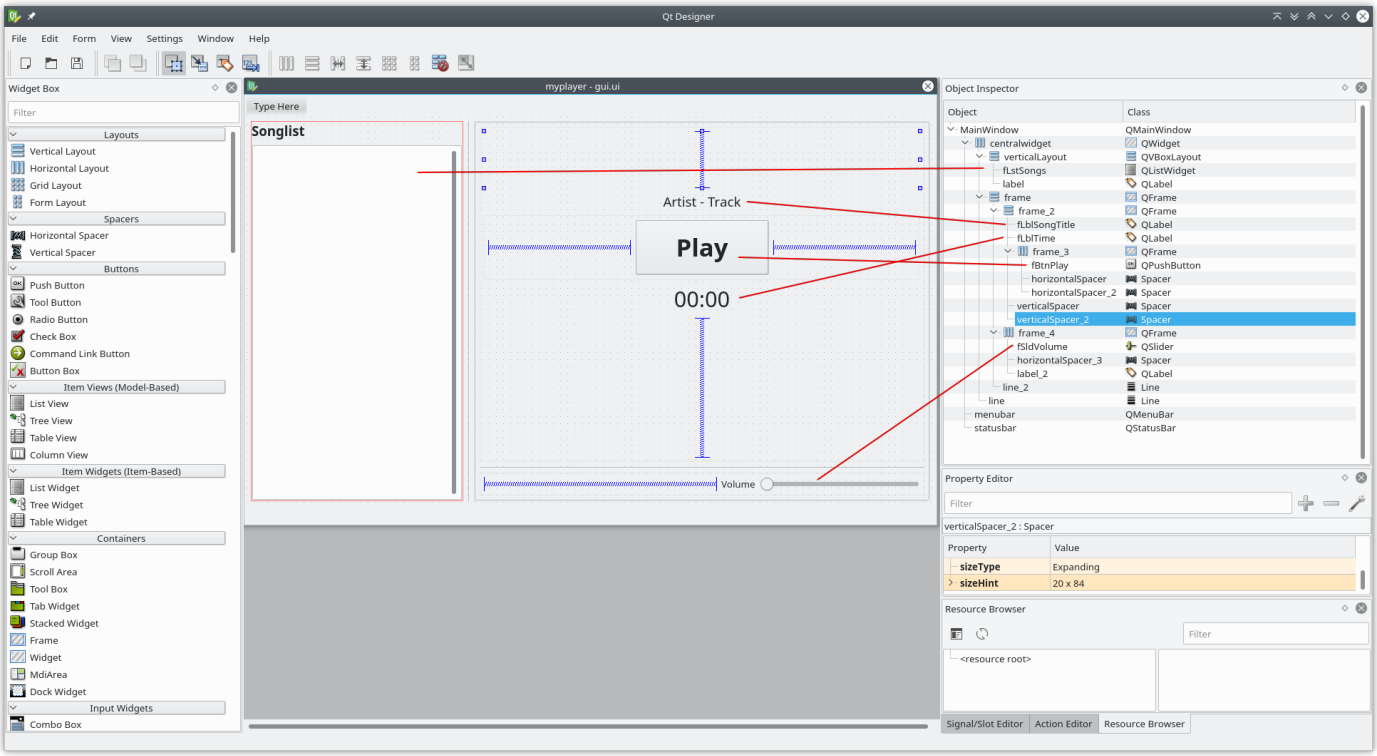


fig 4: Developing UI in QtDesigner

Once you are done with the design you can save the final `.ui` file into your project (I stuffed it into a new directory `MyPlayer/src/app/frontend/_assets`, so let's use that one).

With that done we need to translate the `.ui` file into python code. Open `konsole`, `cd` into that `_assets` directory and do:

```
~:$  pyuic4 -x -o ./gui.py ./gui.ui
```

You can look at the result by listing that directory's contents:

```
~:$  ls -la
total 32
drwxrwxr-x 2 peter peter 4096 Aug 20 18:00 .
drwxrwxr-x 3 peter peter 4096 Aug 20 17:53 ..
-rw-rw-r-- 1 peter peter 9062 Aug 20 18:00 gui.py  <---- python file
-rw-rw-r-- 1 peter peter 9792 Aug 20 17:56 gui.ui
```

The new python file is your local GUI. When you run this script you will see MyPlayer's main window[2]. Run it and try out the various controls on it (button, slider, ...).

There's been an event in China...

As you operated the GUI you noticed ... nothing interesting! Button gets pressed, Slider moves... but nothing much else! What's wrong? GUI's gone on strike? Not really, but in order for stuff to happen you need to write some *event handlers* and connect them with the GUI elements we want

2 For the script's source, see Appendix A, Listing: Lesson 3, gui.py.

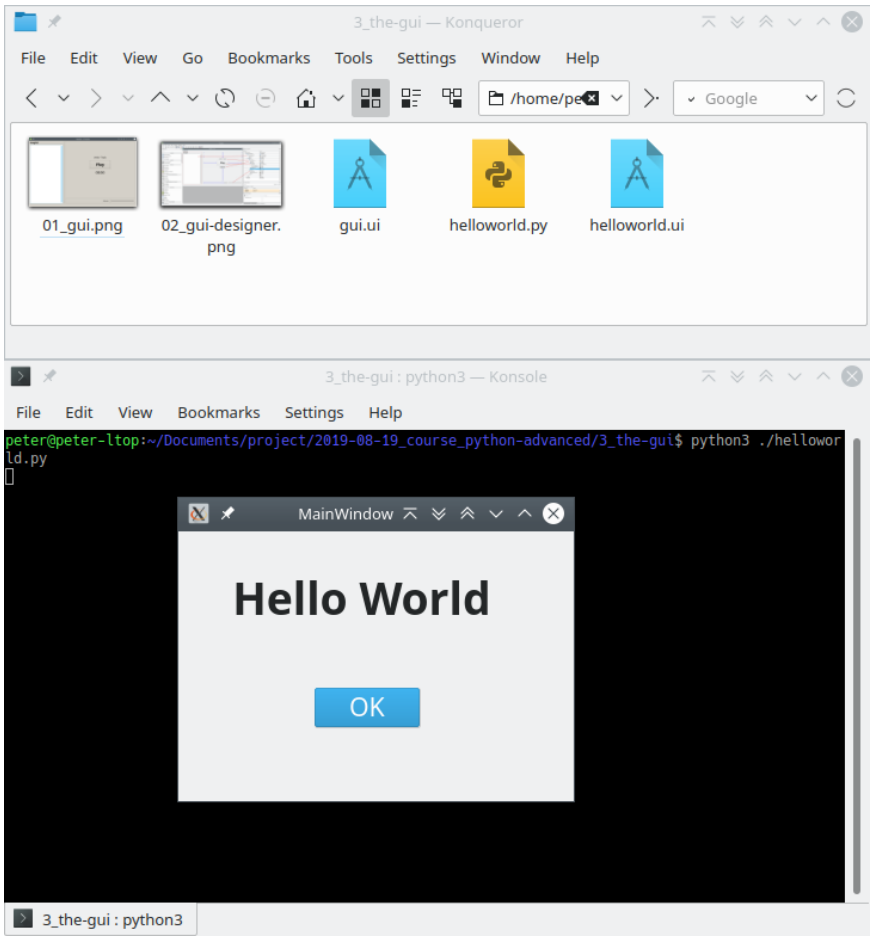
to use. Once connected an event handler will do whatever we program into it when the corresponding GUI element experiences the event. Let's illustrate my skimpy explanation with an example and revisit that homework after lesson 1 (HelloWorld). Like our GUI here the HelloWorld thingy was dead as a Dodo! You could hit that OK button and nothing happens! Let's declare war on lazy GUIs and have it show "How are you?" when we click that OK button!

For starters, let's see the source code generated by my **`pyuic`** compiler:

```
1  # -*- coding: utf-8 -*-
2
3  # Form implementation generated from reading ui file './helloworld.ui'
4  #
5  # Created by: PyQt4 UI code generator 4.12.1
6  #
7  # WARNING! All changes made in this file will be lost!
8
9  from PyQt4 import QtCore, QtGui
10
11 try:
12     _fromUtf8 = QtCore.QString.fromUtf8
13 except AttributeError:
14     def _fromUtf8(s):
15         return s
16
17 try:
18     _encoding = QtGui.QApplication.UnicodeUTF8
19     def _translate(context, text, disambig):
20         return QtGui.QApplication.translate(context, text, disambig, _encoding)
21 except AttributeError:
22     def _translate(context, text, disambig):
23         return QtGui.QApplication.translate(context, text, disambig)
24
25 class Ui_MainWindow(object):
26     def setupUi(self, MainWindow):
27         MainWindow.setObjectName(_fromUtf8("MainWindow"))
28         MainWindow.resize(325, 223)
29         self.centralwidget = QtGui.QWidget(MainWindow)
30         self.centralwidget.setObjectName(_fromUtf8("centralwidget"))
31         self.btnOK = QtGui.QPushButton(self.centralwidget)
32         self.btnOK.setGeometry(QtCore.QRect(110, 130, 90, 35))
33         font = QtGui.QFont()
34         font.setPointSize(16)
35         self.btnOK.setFont(font)
36         self.btnOK.setObjectName(_fromUtf8("btnOK"))
37         self.lblMsg = QtGui.QLabel(self.centralwidget)
38         self.lblMsg.setGeometry(QtCore.QRect(20, 30, 261, 51))
39         font = QtGui.QFont()
40         font.setPointSize(28)
41         font.setBold(True)
42         font.setWeight(75)
43         self.lblMsg.setFont(font)
44         self.lblMsg.setAlignment(QtCore.Qt.AlignCenter)
45         self.lblMsg.setObjectName(_fromUtf8("lblMsg"))
46         MainWindow.setCentralWidget(self.centralwidget)
```

```
47
48          self.retranslateUi(MainWindow)
49          QtCore.QMetaObject.connectSlotsByName(MainWindow)
50
51      def retranslateUi(self, MainWindow):
52          MainWindow.setWindowTitle(_translate("MainWindow", "MainWindow", None))
53          self.btnOK.setText(_translate("MainWindow", "OK", None))
54          self.lblMsg.setText(_translate("MainWindow", "Hello World", None))
55
56
57  if __name__ == "__main__":
58      import sys
59      app = QtGui.QApplication(sys.argv)
60      MainWindow = QtGui.QMainWindow()
61      ui = Ui_MainWindow()
62      ui.setupUi(MainWindow)
63      MainWindow.show()
64      sys.exit(app.exec_())
```

This code doesn't do anything interesting, apart from showing a pretty window when running it in python. Let's do that:



Apart from a few basics this GUI doesn't do anything when you click on it - The button turns blue when you click on it and the application quits when you click that little cross thingy on the top right corner. But we want the thing to do something interesting when we click that button! Let's say that "something" is: Changing "Hello World" to "How are you?". We need a few things to accomplish that.

Firstly, we need to create an event handler. For that we need to edit the python file which **`pyuic`** generated (**`helloworld.py`**). You could do that in LiClipse, but let's keep things simple and use Kate instead (Start button -> Utilities -> Kate). So - open **`helloworld.py`** in Kate. You may want to delete that offputting warning at the top ("all changes will be lost"). This is relevant when you recreate **`helloworld.py`** by running **`pyuic`** again. No worries, we won't do that, so you might as well get rid of that confusing warning.

Then in Kate edit your code so it looks like this:

```
1   # -*- coding: utf-8 -*-
2
3   # Form implementation generated from reading ui file './helloworld.ui'
4   #
5   # Created by: PyQt4 UI code generator 4.12.1
6   #
7
8   from PyQt4 import QtCore, QtGui
9
10  try:
11      _fromUtf8 = QtCore.QString.fromUtf8
12  except AttributeError:
13      def _fromUtf8(s):
14          return s
15
16  try:
17      _encoding = QtGui.QApplication.UnicodeUTF8
18      def _translate(context, text, disambig):
19          return QtGui.QApplication.translate(context, text, disambig, _encoding)
20  except AttributeError:
21      def _translate(context, text, disambig):
22          return QtGui.QApplication.translate(context, text, disambig)
23
24  class Ui_MainWindow(object):
25
26      def Handle_OK_Click():
27          print ("Ouch you clicked me!")
28
29      def setupUi(self, MainWindow):
30          MainWindow.setObjectName(_fromUtf8("MainWindow"))
31          MainWindow.resize(325, 223)
32          self.centralwidget = QtGui.QWidget(MainWindow)
33          self.centralwidget.setObjectName(_fromUtf8("centralwidget"))
34          self.btnOK = QtGui.QPushButton(self.centralwidget)
35          self.btnOK.setGeometry(QtCore.QRect(110, 130, 90, 35))
36          font = QtGui.QFont()
37          font.setPointSize(16)
```

```
38          self.btnOK.setFont(font)
39          self.btnOK.setObjectName(_fromUtf8("btnOK"))
40          self.lblMsg = QtGui.QLabel(self.centralwidget)
41          self.lblMsg.setGeometry(QtCore.QRect(20, 30, 261, 51))
42          font = QtGui.QFont()
43          font.setPointSize(28)
44          font.setBold(True)
45          font.setWeight(75)
46          self.lblMsg.setFont(font)
47          self.lblMsg.setAlignment(QtCore.Qt.AlignCenter)
48          self.lblMsg.setObjectName(_fromUtf8("lblMsg"))
49          MainWindow.setCentralWidget(self.centralwidget)
50
51          self.btnOK.clicked.connect (self.Handle_OK_Click)
52
53          self.retranslateUi(MainWindow)
54          QtCore.QMetaObject.connectSlotsByName(MainWindow)
55
56      def retranslateUi(self, MainWindow):
57          MainWindow.setWindowTitle(_translate("MainWindow", "MainWindow", None))
58          self.btnOK.setText(_translate("MainWindow", "OK", None))
59          self.lblMsg.setText(_translate("MainWindow", "Hello World", None))
60
61
62  if __name__ == "__main__":
63      import sys
64      app = QtGui.QApplication(sys.argv)
65      MainWindow = QtGui.QMainWindow()
66      ui = Ui_MainWindow()
67      ui.setupUi(MainWindow)
68      MainWindow.show()
69      sys.exit(app.exec_())
```

What happens here? In lines **26** and **27** we implemented a function **Handle_OK_Click ()**. Then, in line 51 we connect that function to the OK button so that each time you click that OK button we call the function **Handle_OK_Click ()**. In GUI speak, stuff like a button click is called an *event*. The method **Handle_OK_Click ()** handles that event and has hence become the *event handler* for all clicks on the OK button. There are many different types of events - for example, sliding a slider, selecting a list box entry, setting a checkbox and more galore. Any widget (button, checkbox, listbox, ...) needs an event handler for each type of event we want to handle. As an astute reader you have no doubt noticed that the event handler just prints "Ouch, you clicked me!" onto the terminal... As an exercise, implement the proper behaviour I mentioned above!

And for another example! Suppose you have a form with a text box and a checkbox and want to respond to the user typing into the text box and setting the checkbox,

- You need to provide two event handlers[3] (similar to example above in lines **26** and **27**):
  - One for handling state changes in the checkbox,
  - the other one for handling text changes in the text box

3 As you don't want to handle anything else you don't need any other event handlers.

- You need to connect each event handler to its widget (similar to example above in line `51`):
  - You want one handler to run when the user clicks the button, so you would connect that one to the clicked event of the button.
  - You want the other handler to run when the user changes text in the text box, so you would connect that one to the `textChanged` event of the text box[4].

Here is our wonder application, sporting a line edit and a checkbox - Ladies and gentlemen, put your hands together for the great Frobrinatorrrrr!

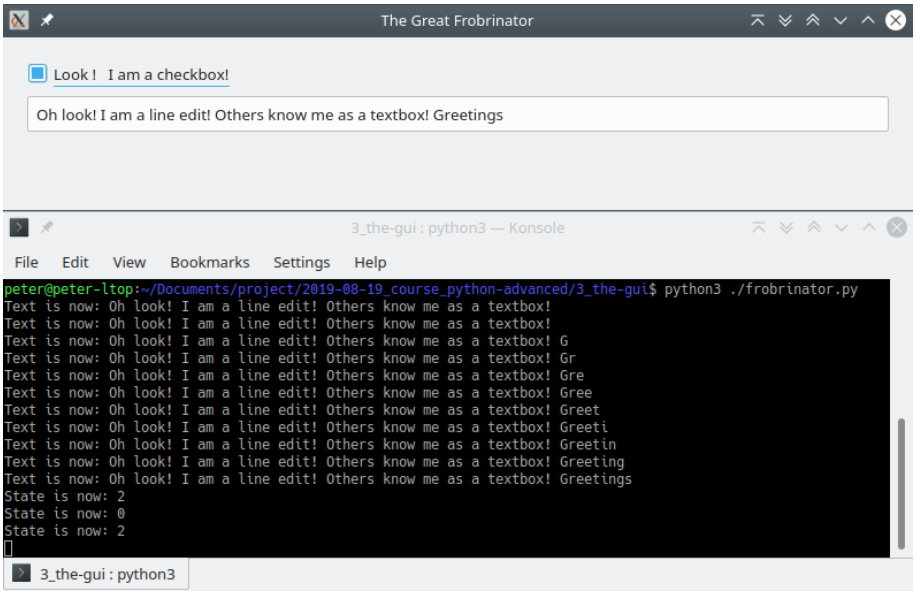


fig 5: The Great Frobri-thing (Hear the sound of applause!)

```
1   # -*- coding: utf-8 -*-
2
3   # Form implementation generated from reading ui file './frobrinator.ui'
4   #
5   # Created by: PyQt4 UI code generator 4.12.1
6
7   from PyQt4 import QtCore, QtGui
8
9   try:
10      _fromUtf8 = QtCore.QString.fromUtf8
11  except AttributeError:
12      def _fromUtf8(s):
13          return s
14
15  try:
16      _encoding = QtGui.QApplication.UnicodeUTF8
17      def _translate(context, text, disambig):
18          return QtGui.QApplication.translate(context, text, disambig, _encoding)
19  except AttributeError:
20      def _translate(context, text, disambig):
21          return QtGui.QApplication.translate(context, text, disambig)
```

4 Sorry, I mislead you... in Qt the textbox isn't called a textbox... it's actually called a line edit!

```
22
23 class Ui_MainWindow(object):
24
25     def Handle_LineEdit_TextChanged (self):
26         print ("Text is now: %s" % self.lineEdit.text ())
27
28     def Handle_CheckBox_Toggled (self, state):
29         print ("State is now: %s" % state)
30
31     def setupUi(self, MainWindow):
32         MainWindow.setObjectName(_fromUtf8("MainWindow"))
33         MainWindow.resize(790, 147)
34         self.centralwidget = QtGui.QWidget(MainWindow)
35         self.centralwidget.setObjectName(_fromUtf8("centralwidget"))
36         self.checkBox = QtGui.QCheckBox(self.centralwidget)
37         self.checkBox.setGeometry(QtCore.QRect(20, 20, 181, 23))
38         self.checkBox.setObjectName(_fromUtf8("checkBox"))
39         self.lineEdit = QtGui.QLineEdit(self.centralwidget)
40         self.lineEdit.setGeometry(QtCore.QRect(20, 50, 751, 33))
41         self.lineEdit.setObjectName(_fromUtf8("lineEdit"))
42         MainWindow.setCentralWidget(self.centralwidget)
43         self.menubar = QtGui.QMenuBar(MainWindow)
44         self.menubar.setGeometry(QtCore.QRect(0, 0, 790, 31))
45         self.menubar.setObjectName(_fromUtf8("menubar"))
46         MainWindow.setMenuBar(self.menubar)
47         self.statusbar = QtGui.QStatusBar(MainWindow)
48         self.statusbar.setObjectName(_fromUtf8("statusbar"))
49         MainWindow.setStatusBar(self.statusbar)
50
51         self.lineEdit.textChanged.connect (self.Handle_LineEdit_TextChanged)
52         self.checkBox.stateChanged.connect (self.Handle_CheckBox_Toggled)
53
54         self.retranslateUi(MainWindow)
55         QtCore.QMetaObject.connectSlotsByName(MainWindow)
56
57     def retranslateUi(self, MainWindow):
58         MainWindow.setWindowTitle(
59             _translate("MainWindow", "The Great Frobrinator", None))
60         self.checkBox.setText(
61             _translate("MainWindow", "Look !   I am a checkbox!", None))
62         self.lineEdit.setText(
63             _translate(
64                 "MainWindow",
65                 "Oh look! I am a line edit! Others know me as a textbox!",
66                 None
67                 )
68             )
69
70 if __name__ == "__main__":
71     import sys
72     app = QtGui.QApplication(sys.argv)
73     MainWindow = QtGui.QMainWindow()
74     ui = Ui_MainWindow()
```

```
75      ui.setupUi(MainWindow)
76      MainWindow.show()
77      sys.exit(app.exec_())
```

What can you put into the event handlers? Your imagination is the limit... you could set the form's color to yellow when you click the button or minimize it when you type "hello" into the text box! Our example above is boring - all it does is to print a message onto the terminal!

So... we didn't really have an event in China (Admit you found the heading kool, otherwise I will attach an event handler to your left earlobe!), but we have plenty of events on our forms, and you are the expert when it comes to handle them! To recap -

- On your GUI you place various widgets,
- Each widget can send events, depending on what the user does with it
- To give your GUI life you have to create event handlers (each handler a function) and connect those to the form's widgets.
- You only need to create (and connect) those handlers you need

Our GUI

After all that long fluff, let's turn our attention back to the GUI *we* use. We presented the looks at the beginning of this chapter. Here is the source code as created by pyuic (I removed a lot of the generated code, because I want to show our parts):

```
1   from PyQt4 import QtCore, QtGui
2
3   ... snip ...
4
5   class Ui_MainWindow(object):
6       def setupUi(self, MainWindow):
7           ... snip ...
8
9       def retranslateUi(self, MainWindow):
10          ... snip ...
11
12  if __name__ == "__main__":
13      import sys
14      ... snip ...
```

Firstly, we need some event handlers that make the form alive. We need three event handlers (meaning: functions):

- An event handler which captures mouse (left) clicks on the `Play button`,
- An event handler which captures item selections on the left hand `playlist`,
- An event handler which captures value changes on the `volume slider` (i.e. user moves the slider).

We implement these functions as methods of the `Ui_MainWindow` class, and we mark them as private by putting an underscore in front of them[5]:

5 It's just a convention saying to others "Please don't use". In current python every method is (really) public.

```
1   from PyQt4 import QtCore, QtGui
2
3   ... snip ...
4
5   class Ui_MainWindow(object):
6       """
7       The local GUI.
8       """
9       def _Handle_BtnPlay_Click (self):
10          """
11          Event handler: User clicked "Play" button.
12          """
13          ... to be added later ...
14
15      def _Handle_LstPlaylist_Select (self, item):
16          """
17          Event handler: User selected a song on the playlist.
18          """
19          ... to be added later ...
20
21      def _Handle_SldVolume_Move (self, currentValue):
22          """
23          Event handler: User changed volume setting.
24          """
25          ... to be added later ...
26
27      def setupUi(self, MainWindow):
28          ... snip ...
29
30          self.fBtnPlay.clicked.connect          (self._Handle_BtnPlay_Click)
31          self.fLstPlaylist.itemClicked.connect  (self._Handle_LstPlaylist_Select)
32          self.fSldVolume.sliderMoved.connect    (self._Handle_SldVolume_Move)
33
34      def retranslateUi(self, MainWindow):
35          ... snip ...
36
37  if __name__ == "__main__":
38      import sys
39      ... snip ...
```

As you see, we have three little event handlers and their evil little sisters (connection statements in `setupUI()`). Note that from here forthwith we do some better source commenting. This is going to be the standard. Good source commenting makes code much much more readable - even to yourself after five years!

Here is the full code of our GUI so far:

```
1  # -*- coding: utf-8 -*-
2
3  # Form implementation generated from reading ui file './gui.ui'
4  #
5  # Created by: PyQt4 UI code generator 4.12.1
6
7  from PyQt4 import QtCore, QtGui
8
9  try:
10     _fromUtf8 = QtCore.QString.fromUtf8
11 except AttributeError:
12     def _fromUtf8(s):
13         return s
14
15 try:
16     _encoding = QtGui.QApplication.UnicodeUTF8
17     def _translate(context, text, disambig):
18         return QtGui.QApplication.translate(context, text, disambig, _encoding)
19 except AttributeError:
20     def _translate(context, text, disambig):
21         return QtGui.QApplication.translate(context, text, disambig)
22
23 class Ui_MainWindow(object):
24     def _Handle_BtnPlay_Click (self):
25          """
26          Event handler: User clicked "Play" button.
27          """
28          ... to be added later ...
29
30     def _Handle_LstPlaylist_Select (self, item):
31          """
32          Event handler: User selected a song on the playlist.
33          """
34          ... to be added later ...
35
36     def _Handle_SldVolume_Move (self, currentValue):
37          """
38          Event handler: User changed volume setting.
39          """
40          ... to be added later ...
41
42     def setupUi(self, MainWindow):
43         MainWindow.setObjectName(_fromUtf8("MainWindow"))
44         MainWindow.resize(965, 600)
45         self.centralwidget = QtGui.QWidget(MainWindow)
46         self.centralwidget.setObjectName(_fromUtf8("centralwidget"))
47         self.horizontalLayout = QtGui.QHBoxLayout(self.centralwidget)
48         self.horizontalLayout.setObjectName(_fromUtf8("horizontalLayout"))
49         self.verticalLayout = QtGui.QVBoxLayout()
50          self.verticalLayout.setSizeConstraint(QtGui.QLayout.SetDefaultConstraint
   )
51         self.verticalLayout.setContentsMargins(0, 0, 0, -1)
52         self.verticalLayout.setObjectName(_fromUtf8("verticalLayout"))
```

```
53          self.label = QtGui.QLabel(self.centralwidget)
54          font = QtGui.QFont()
55          font.setPointSize(14)
56          font.setBold(True)
57          font.setWeight(75)
58          self.label.setFont(font)
59          self.label.setTextFormat(QtCore.Qt.PlainText)
60          self.label.setObjectName(_fromUtf8("label"))
61          self.verticalLayout.addWidget(self.label)
62          self.fLstSongs = QtGui.QListWidget(self.centralwidget)
63          font = QtGui.QFont()
64          font.setPointSize(14)
65          font.setBold(True)
66          font.setWeight(75)
67          self.fLstSongs.setFont(font)
68          self.fLstSongs.setVerticalScrollBarPolicy(QtCore.Qt.ScrollBarAlwaysOn)
69           self.fLstSongs.setHorizontalScrollBarPolicy(QtCore.Qt.ScrollBarAlwaysOff
    )
70          self.fLstSongs.setUniformItemSizes(False)
71          self.fLstSongs.setSelectionRectVisible(True)
72          self.fLstSongs.setObjectName(_fromUtf8("fLstSongs"))
73          self.verticalLayout.addWidget(self.fLstSongs)
74          self.horizontalLayout.addLayout(self.verticalLayout)
75          self.line = QtGui.QFrame(self.centralwidget)
76          self.line.setFrameShape(QtGui.QFrame.VLine)
77          self.line.setFrameShadow(QtGui.QFrame.Sunken)
78          self.line.setObjectName(_fromUtf8("line"))
79          self.horizontalLayout.addWidget(self.line)
80          self.frame = QtGui.QFrame(self.centralwidget)
81          sizePolicy = QtGui.QSizePolicy(
82              QtGui.QSizePolicy.Minimum, QtGui.QSizePolicy.Preferred)
83          sizePolicy.setHorizontalStretch(0)
84          sizePolicy.setVerticalStretch(0)
85          sizePolicy.setHeightForWidth(
86              self.frame.sizePolicy().hasHeightForWidth())
87          self.frame.setSizePolicy(sizePolicy)
88          self.frame.setMinimumSize(QtCore.QSize(640, 0))
89          self.frame.setFrameShape(QtGui.QFrame.StyledPanel)
90          self.frame.setFrameShadow(QtGui.QFrame.Raised)
91          self.frame.setLineWidth(0)
92          self.frame.setObjectName(_fromUtf8("frame"))
93          self.verticalLayout_3 = QtGui.QVBoxLayout(self.frame)
94          self.verticalLayout_3.setObjectName(_fromUtf8("verticalLayout_3"))
95          self.frame_2 = QtGui.QFrame(self.frame)
96          self.frame_2.setFrameShape(QtGui.QFrame.NoFrame)
97          self.frame_2.setFrameShadow(QtGui.QFrame.Raised)
98          self.frame_2.setObjectName(_fromUtf8("frame_2"))
99          self.verticalLayout_2 = QtGui.QVBoxLayout(self.frame_2)
100         self.verticalLayout_2.setObjectName(_fromUtf8("verticalLayout_2"))
101         spacerItem = QtGui.QSpacerItem(
102             20, 84, QtGui.QSizePolicy.Minimum, QtGui.QSizePolicy.Expanding)
103         self.verticalLayout_2.addItem(spacerItem)
104         self.fLblSongTitle = QtGui.QLabel(self.frame_2)
```

```
105         font = QtGui.QFont()
106         font.setPointSize(14)
107         self.fLblSongTitle.setFont(font)
108         self.fLblSongTitle.setAlignment(QtCore.Qt.AlignCenter)
109         self.fLblSongTitle.setObjectName(_fromUtf8("fLblSongTitle"))
110         self.verticalLayout_2.addWidget(self.fLblSongTitle)
111         self.frame_3 = QtGui.QFrame(self.frame_2)
112         self.frame_3.setFrameShape(QtGui.QFrame.NoFrame)
113         self.frame_3.setFrameShadow(QtGui.QFrame.Raised)
114         self.frame_3.setObjectName(_fromUtf8("frame_3"))
115         self.horizontalLayout_2 = QtGui.QHBoxLayout(self.frame_3)
116         self.horizontalLayout_2.setObjectName(_fromUtf8("horizontalLayout_2"))
117         spacerItem1 = QtGui.QSpacerItem(
118             197, 20, QtGui.QSizePolicy.Expanding, QtGui.QSizePolicy.Minimum)
119         self.horizontalLayout_2.addItem(spacerItem1)
120         self.fBtnPlay = QtGui.QPushButton(self.frame_3)
121         self.fBtnPlay.setMinimumSize(QtCore.QSize(188, 78))
122         font = QtGui.QFont()
123         font.setPointSize(26)
124         font.setBold(True)
125         font.setWeight(75)
126         self.fBtnPlay.setFont(font)
127         self.fBtnPlay.setCheckable(False)
128         self.fBtnPlay.setObjectName(_fromUtf8("fBtnPlay"))
129         self.horizontalLayout_2.addWidget(self.fBtnPlay)
130         spacerItem2 = QtGui.QSpacerItem(
131            197, 20, QtGui.QSizePolicy.Expanding, QtGui.QSizePolicy.Minimum)
132         self.horizontalLayout_2.addItem(spacerItem2)
133         self.verticalLayout_2.addWidget(self.frame_3)
134         self.fLblTime = QtGui.QLabel(self.frame_2)
135         font = QtGui.QFont()
136         font.setPointSize(23)
137         self.fLblTime.setFont(font)
138         self.fLblTime.setAlignment(QtCore.Qt.AlignCenter)
139         self.fLblTime.setObjectName(_fromUtf8("fLblTime"))
140         self.verticalLayout_2.addWidget(self.fLblTime)
141         spacerItem3 = QtGui.QSpacerItem(
142            20, 200, QtGui.QSizePolicy.Minimum, QtGui.QSizePolicy.Expanding)
143         self.verticalLayout_2.addItem(spacerItem3)
144         self.verticalLayout_3.addWidget(self.frame_2)
145         self.line_2 = QtGui.QFrame(self.frame)
146         self.line_2.setFrameShape(QtGui.QFrame.HLine)
147         self.line_2.setFrameShadow(QtGui.QFrame.Sunken)
148         self.line_2.setObjectName(_fromUtf8("line_2"))
149         self.verticalLayout_3.addWidget(self.line_2)
150         self.frame_4 = QtGui.QFrame(self.frame)
151         self.frame_4.setFrameShape(QtGui.QFrame.NoFrame)
152         self.frame_4.setFrameShadow(QtGui.QFrame.Raised)
153         self.frame_4.setObjectName(_fromUtf8("frame_4"))
154         self.horizontalLayout_3 = QtGui.QHBoxLayout(self.frame_4)
155         self.horizontalLayout_3.setObjectName(_fromUtf8("horizontalLayout_3"))
156         spacerItem4 = QtGui.QSpacerItem(
157            40, 20, QtGui.QSizePolicy.Expanding, QtGui.QSizePolicy.Minimum)
```

```
158         self.horizontalLayout_3.addItem(spacerItem4)
159         self.label_2 = QtGui.QLabel(self.frame_4)
160         self.label_2.setObjectName(_fromUtf8("label_2"))
161         self.horizontalLayout_3.addWidget(self.label_2)
162         self.fSldVolume = QtGui.QSlider(self.frame_4)
163         sizePolicy = QtGui.QSizePolicy(
164         QtGui.QSizePolicy.Fixed, QtGui.QSizePolicy.Fixed)
165         sizePolicy.setHorizontalStretch(0)
166         sizePolicy.setVerticalStretch(0)
167         sizePolicy.setHeightForWidth(
168         self.fSldVolume.sizePolicy().hasHeightForWidth())
169         self.fSldVolume.setSizePolicy(sizePolicy)
170         self.fSldVolume.setMinimumSize(QtCore.QSize(225, 0))
171         self.fSldVolume.setBaseSize(QtCore.QSize(0, 0))
172         self.fSldVolume.setAcceptDrops(False)
173         self.fSldVolume.setTracking(True)
174         self.fSldVolume.setOrientation(QtCore.Qt.Horizontal)
175         self.fSldVolume.setInvertedAppearance(False)
176         self.fSldVolume.setInvertedControls(False)
177         self.fSldVolume.setObjectName(_fromUtf8("fSldVolume"))
178         self.horizontalLayout_3.addWidget(self.fSldVolume)
179         self.verticalLayout_3.addWidget(self.frame_4)
180         self.horizontalLayout.addWidget(self.frame)
181         MainWindow.setCentralWidget(self.centralwidget)
182         self.menubar = QtGui.QMenuBar(MainWindow)
183         self.menubar.setGeometry(QtCore.QRect(0, 0, 965, 31))
184         self.menubar.setObjectName(_fromUtf8("menubar"))
185         MainWindow.setMenuBar(self.menubar)
186         self.statusbar = QtGui.QStatusBar(MainWindow)
187         self.statusbar.setObjectName(_fromUtf8("statusbar"))
188         MainWindow.setStatusBar(self.statusbar)
189
190         self.fBtnPlay.clicked.connect          (self._Handle_BtnPlay_Click)
191         self.fLstPlaylist.itemClicked.connect  (self._Handle_LstPlaylist_Select)
192         self.fSldVolume.sliderMoved.connect    (self._Handle_SldVolume_Move)
193
194         QtCore.QMetaObject.connectSlotsByName(MainWindow)
195
196
197
198       def retranslateUi(self, MainWindow):
199           MainWindow.setWindowTitle(_translate("MainWindow", "myplayer", None))
200           self.label.setText(_translate("MainWindow", "Songlist", None))
201           self.fLblSongTitle.setText(
202              _translate("MainWindow", "Artist - Track", None))
203           self.fBtnPlay.setText(_translate("MainWindow", "Play", None))
204           self.fLblTime.setText(_translate("MainWindow", "00:00", None))
205           self.label_2.setText(_translate("MainWindow", "Volume", None))
206
207
208   if __name__ == "__main__":
209       import sys
210       app = QtGui.QApplication(sys.argv)
```

```
211         MainWindow = QtGui.QMainWindow()
212         ui = Ui_MainWindow()
213         ui.setupUi(MainWindow)
214         MainWindow.show()
215         sys.exit(app.exec_())
```

Task: Implement some behaviour in the new GUI:

- At application startup have the program load three items into the left hand `playlist` and set the `Play button` disabled.
- When you click an item in the `playlist` set the `Play button` enabled. Then write the text of what you clicked into the terminal (via **`print(...)`**).
- When you click the (enabled) `Play button`, and it has "Play" written on it, change its text to "Pause".
- When you click the (enabled) `Play button`, and it has "Pause" written on it, change its text to "Play".
- When you move the slider, have it write its current value to the terminal (via **`print (...)`**).

The GUI reloaded

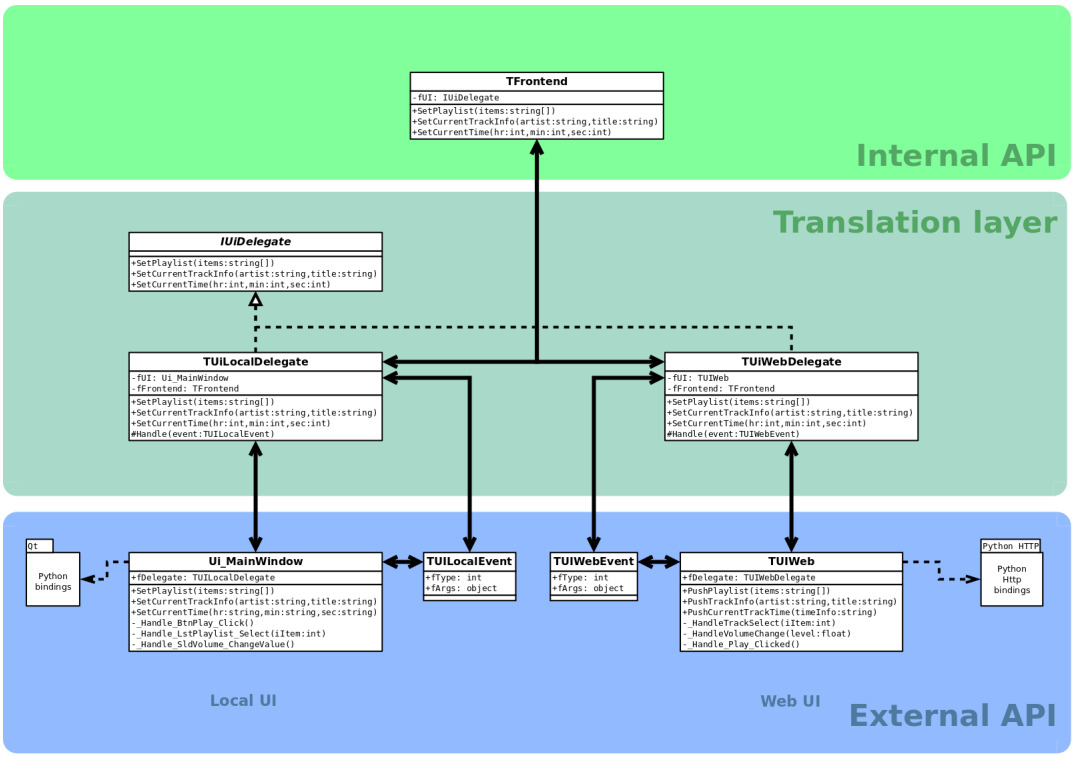
So far our GUI stands by itself. The GUI is the app! As you did your last task you will have noticed that the code grew more and more complex! You may get a hunch that this will become much more complex as you go on. For example, you could bind the vlc part into the GUI and then integrate everything, so you hear the music you select. It's a fast way of getting some results, and it's good to test something quickly or do some prototype. But if we want to develop this further (and we will) then we should really not cram everything into the GUI code - it would become a bloated unmanageable mess! The mistake we'd make would be quite simply that we are trying to integrate the application into the GUI! What we should do instead is to integrate the GUI into our application! This little reversal means a lot for the GUI code as it will mean a much smaller and tidier code - making the thing manageable and maintainable! And it opens a world of opportunities what else we could integrate into our application! For example, as you have seen in chapter 2 there is also another, web based UI to come!

The first step to a good GUI is that we give it a voice that speaks in a large application. What was that code word again, back in chapter 2? Wasn't there something about APIs?

APIs, delegates and singleness

As you will see, our GUI is a highly specialized thing. Its main purpose is to interact with the user by capturing user actions (events: clicks, slides, ...) and changing its appearance (changing some text, coloring some element blue, ...). Because it's so specialized it needs a special set of functions to express that. Unfortunately we cannot allow this special-ness to spread to the entire application! Especially, if we also integrate another UI (that web based one) which is highly specialized as well! That's why we provide each of these special components with a *delegate* - that is a class which makes them both appear the same to our application. We call it a delegate because it represents the specialized UI components, so both of them present the same API to our application! The result will be an *enormously* more readable and maintainable code!

Enough with the preface, let's do it! Here's what I mean:



Here we put some flesh and bones on the big picture we presented back in lesson 2. Basically, we have the external API (UI components), the internal API (our program) and the delegates as the glue between the two. You can see that the components in the external API use their own specialized set of methods. For example, you would call `SetPlaylist` to fill the playlist in the local UI whilst you would call `PushPlaylistItems` in the web UI.

Meanwhile, the Delegates provide identical methods for filling the playlist, namely `SetPlaylist`. This means that whatever UI we use, from our internal API calls `SetPlaylist`. This concept can scale indefinitely - we can add as many different UIs as we want, and thanks to the Delegates we always call `SetPlaylist` to fill the playlist widget.

To enforce consistency we provide an *interface* - `IUiDelegate` - and have all delegates *implement* this interface. `IUiDelegate` contains nothing but method stubs, and each delegate must implement the methods prescribed by the interface. You will notice that `IUiDelegate` provides only setters (`SetPlaylistItem`, `SetCurrentTrackInfo`, `SetCurrentTime`), but none of the `Handle` methods. This is because the setters are called from our application whilst the handlers are called by the resp. frontends. The handlers are as specialized as the frontends which call them. The handlers can still talk to our application by calling the appropriate method in the `TFrontend` class.

Let's talk about `TFrontend`! We now have a unified view on *all* UIs, so why not bundle the whole access via one object? When the application runs we will provide a single, application wide, object of type `TFrontend` which channels all our wishes to whatever UI is running at the moment. To the application (internal API) the single `TFrontend` object becomes our *facade* of the the whole UI framework. The internal API does not need to know what kind of UI it operates, it just tells the facade what it wants done on the UI. For example, if our application wants to set the title and artist of the current track it calls frontend's method `SetCurrentTrackInfo` which then calls `SetCurrentTrackInfo` of the current delegate which then calls the appropriate method(s) in the current UI.

`TFrontend` is the last step to make UI access easy as we now have one object via which we talk to any UI. Note the difference - I said, "one *object*", not just "one class". Application wide, there is one and *only* one object (menaing - instance of a class) that we use for our UI work. This pattern is called *singleton pattern* (singleton being that one object), and it makes the whole UI access *insanely* easy for our application!

Lesson 4: The controller

Objective: Show the concept of a state machine and integrate the GUI into it. Demonstrate how the state machine controls the GUI.

So far we have implemented the program's logic in the GUI. This has several drawbacks, most notably that it is inflexible (as we are bound to a single GUI only) and makes the program convoluted, i.e hard to maintain. The way forward is to follow the "separation of concerns" principle by exporting the program's logic into its own part. We shall explore a viable route by introducing a controller which utilizes a state engine at its core. A state machine is usually shown as the diagram in fig 6.

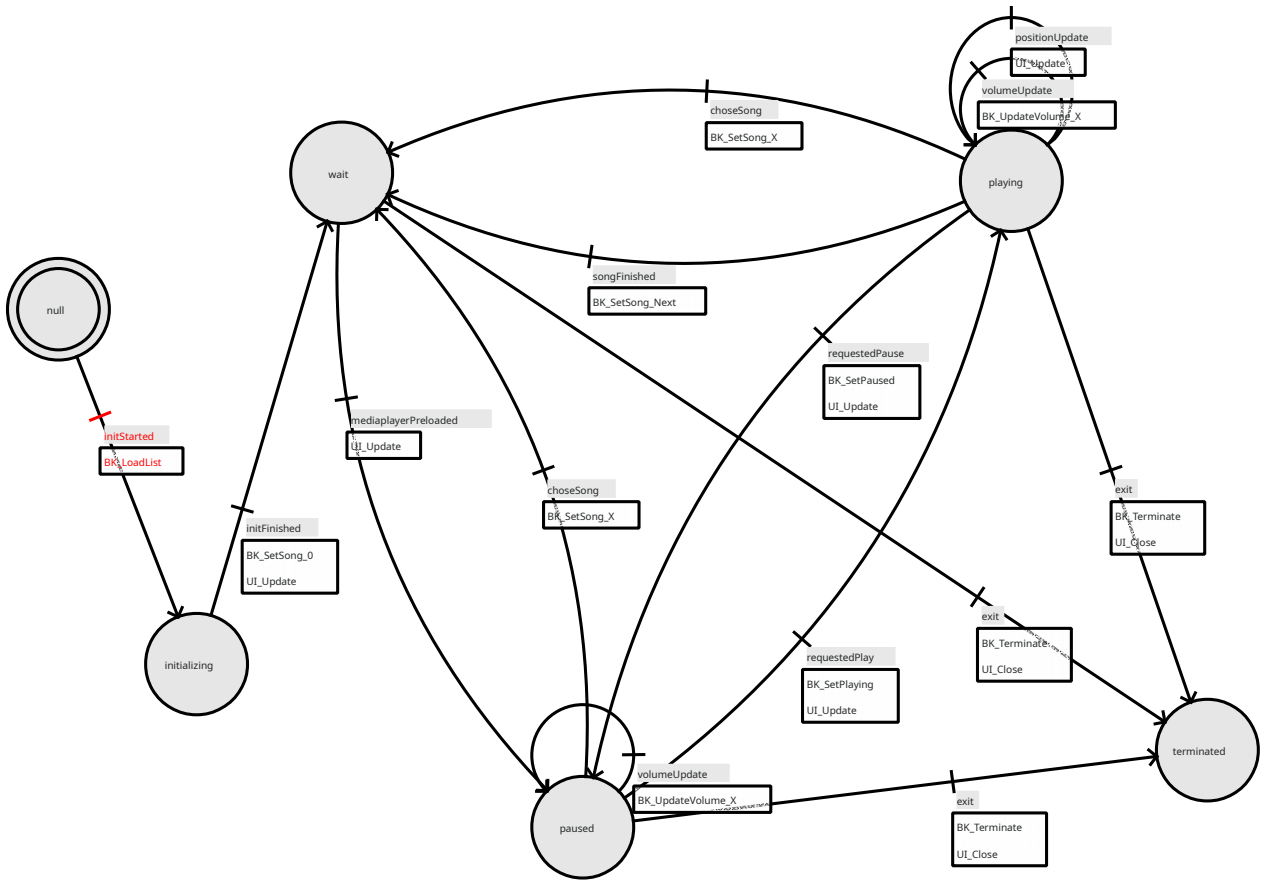
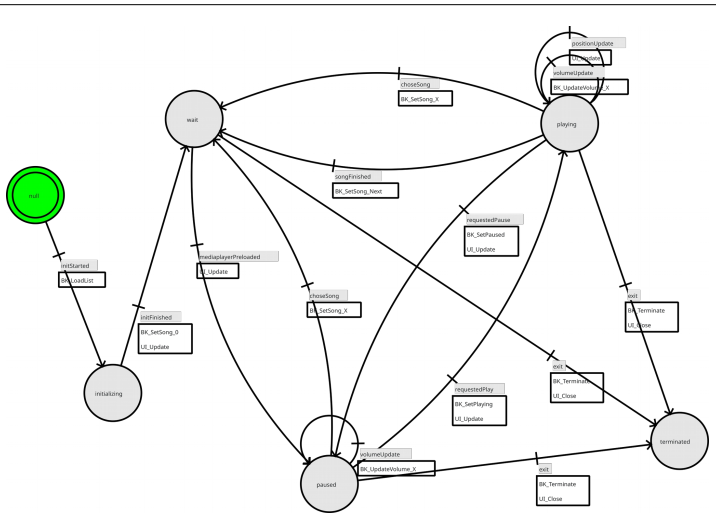
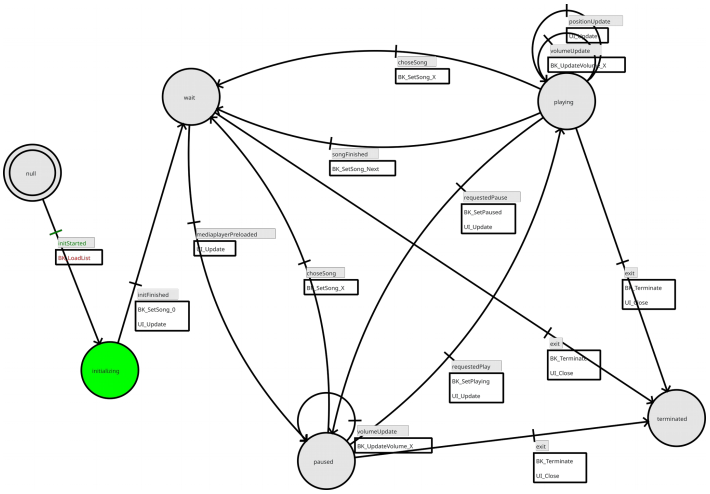
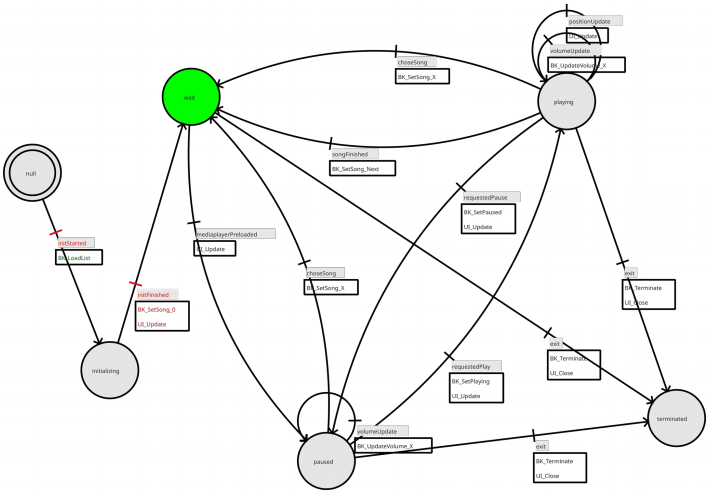
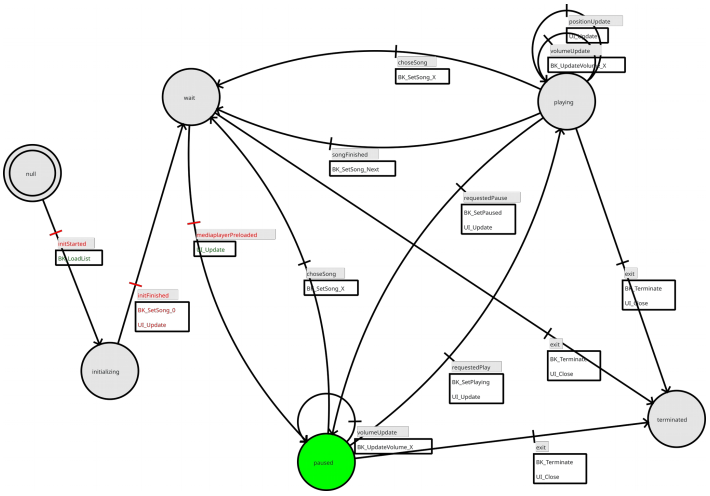
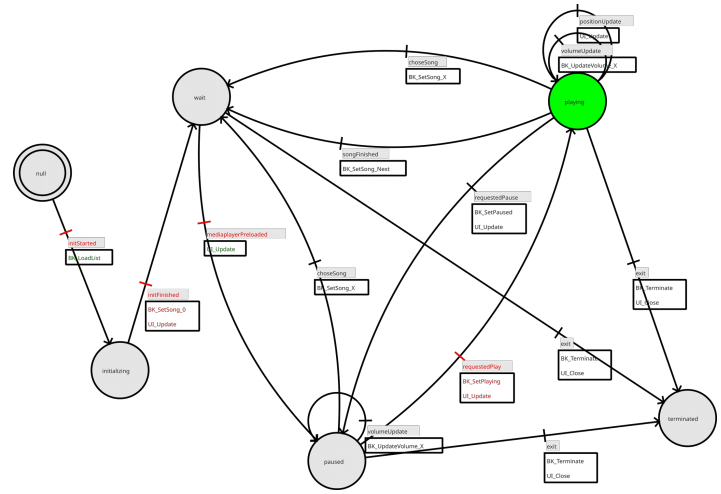
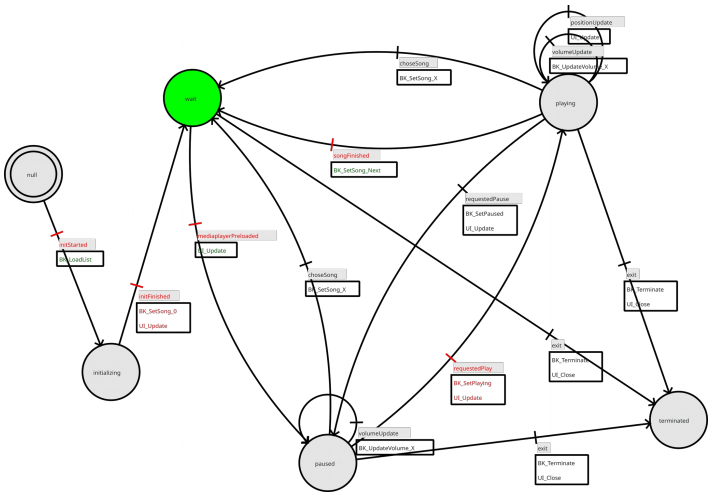
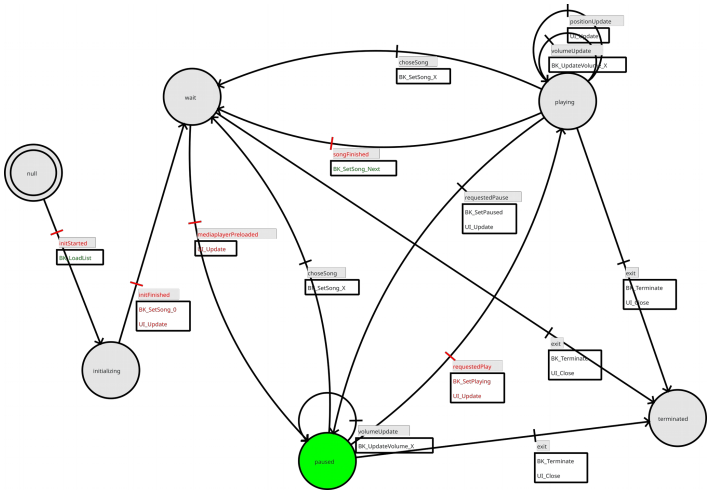


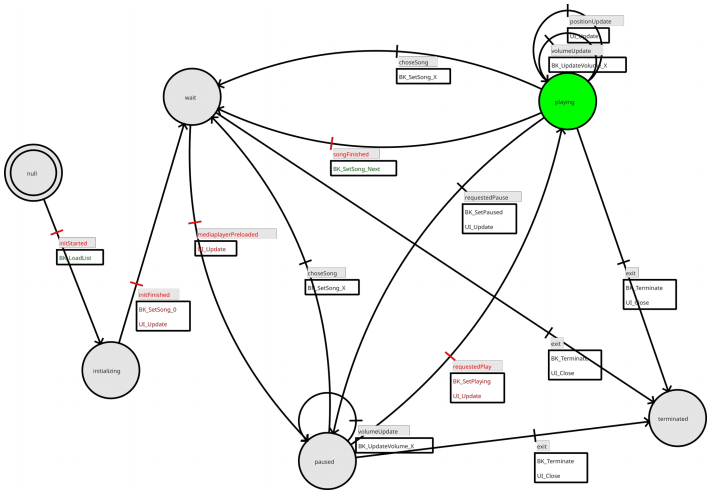
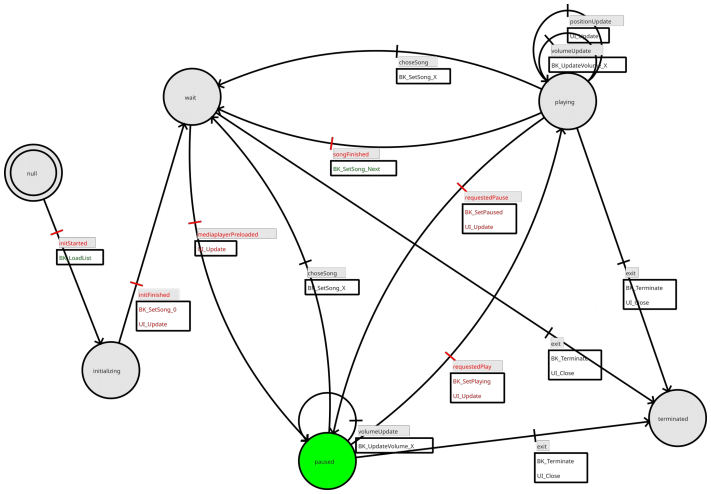
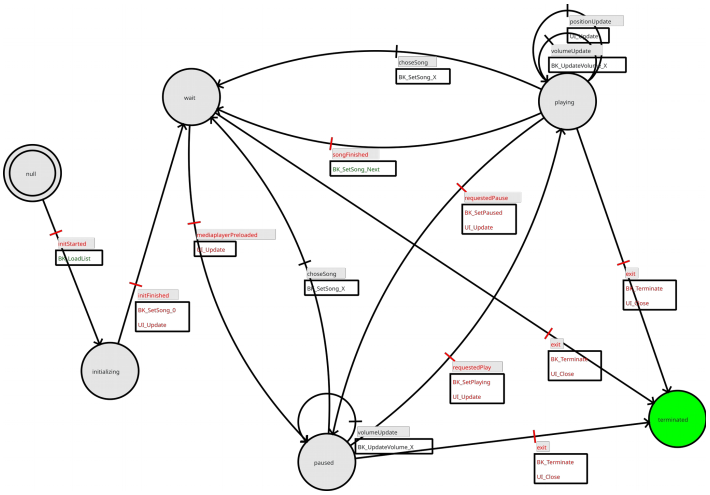
fig 6: state machine

A state machine provides fine grained control of complex applications, also with asynchronous processes. Our media player uses plenty of those! Here is a hypothetical program run and how the state engine controls the player.

At program start we set the state engine to state **`null`**.	
We issue event **`initStarted`**. This sets the engine into state **`initializing`**. State machine calls **`BK_LoadList ()`** to load the songs into the playlist.	
List has loaded. We issue event **`initFinished`**. This sets the state machine into state **`wait`**. State machine calls **`BK_SetSong (0)`** to have the backend load the first song into the media layer. Song loading is asynchronous, i.e. we wait for another event from the backend when the media layer has actually loaded the song.	

Media layer has loaded the song. Backend issues the event **`mediaplayerPreload`**. This sets the state engine into state **`paused`**. State machine calls **`UI_Update ()`** to update the frontend (UI).	
User clicked the big Play button which issues a **`requestPlay`** event. This sets the state engine into state **`playing`**. State machine first calls **`BK_SetPlaying ()`** (media layer plays track), then **`UI_Update ()`** to update the Frontend (UI).	

Song has finished, so the backend layer issues a **`songFinished`** event. This sets the state machine into state **`wait`**. State machine calls BK_SetSong_Next () to have the backend load the next song into the media layer.	
Media layer has loaded the next song. Backend issues the event **`mediaplayerPreload`**. This sets the state engine into state **`paused`**. State machine calls **`UI_Update ()`** to update the frontend (UI).	

User clicked the Play button which issues a **`requestPlay`** event. This sets the state engine into state **`playing`**. State machine first calls **`BK_SetPlaying ()`** (media layer plays track), then **`UI_Update ()`** to update the Frontend (UI).	
User clicked the Pause button which issues a **`requestPause`** event. This sets the state engine into state **`paused`**. State machine first calls **`BK_SetPaused ()`** (media layer plays track), then **`UI_Update ()`** to update the Frontend (UI).	
User closed the application which issues an **`exit`** event. This sets the state engine into state **`terminated`**. State machine first calls **`BK_Terminate ()`** which cleans up any traces (close audio streams etc.) and updates the frontend (UI) so it shows that we exit.	

Lesson 5: The backend

Objective: We build the backend layer which controls audio system and provides a simple playlist management.

Lesson 6: The controller II

Objective: We integrate controller and audio backend and connect it with the GUI.

Lesson 7: WEB GUI, visual design

Objective: We design a simple web GUI (HTML + CSS).

Lesson 8: Player, web service

Objective: We design the player's web backend (Web + web socket server + delegate).

Lesson 9: Integrating web GUI and web service.

Objective: We design a javascript library which communicates with the player's web service to send user requests (track choice, volume updates, ...), receive server information (track infos, track position, volume, ...) and update the web GUI

Appendix A (Source code)

Listing: Lesson 3, gui.py

```
216    # -*- coding: utf-8 -*-
217
218    # Form implementation generated from reading ui file './gui.ui'
219    #
220
221    from PyQt4 import QtCore, QtGui
222
223    try:
224        _fromUtf8 = QtCore.QString.fromUtf8
225    except AttributeError:
226        def _fromUtf8(s):
227            return s
228
229    try:
230        _encoding = QtGui.QApplication.UnicodeUTF8
231        def _translate(context, text, disambig):
232                return  QtGui.QApplication.translate(context,  text,  disambig,
   _encoding)
233    except AttributeError:
234        def _translate(context, text, disambig):
235            return QtGui.QApplication.translate(context, text, disambig)
236
237    class Ui_MainWindow(object):
238        def setupUi(self, MainWindow):
239            MainWindow.setObjectName(_fromUtf8("MainWindow"))
240            MainWindow.resize(965, 600)
241            self.centralwidget = QtGui.QWidget(MainWindow)
242            self.centralwidget.setObjectName(_fromUtf8("centralwidget"))
243            self.horizontalLayout = QtGui.QHBoxLayout(self.centralwidget)
244            self.horizontalLayout.setObjectName(_fromUtf8("horizontalLayout"))
245            self.verticalLayout = QtGui.QVBoxLayout()
246             self.verticalLayout.setSizeConstraint(QtGui.QLayout.SetDefaultConstra
   int)
247            self.verticalLayout.setContentsMargins(0, 0, 0, -1)
248            self.verticalLayout.setObjectName(_fromUtf8("verticalLayout"))
249            self.label = QtGui.QLabel(self.centralwidget)
250            font = QtGui.QFont()
251            font.setPointSize(14)
252            font.setBold(True)
253            font.setWeight(75)
254            self.label.setFont(font)
255            self.label.setTextFormat(QtCore.Qt.PlainText)
256            self.label.setObjectName(_fromUtf8("label"))
257            self.verticalLayout.addWidget(self.label)
258            self.fLstSongs = QtGui.QListWidget(self.centralwidget)
259            font = QtGui.QFont()
260            font.setPointSize(14)
```

```
261             font.setBold(True)
262             font.setWeight(75)
263             self.fLstSongs.setFont(font)
264              self.fLstSongs.setVerticalScrollBarPolicy(QtCore.Qt.ScrollBarAlwaysOn
    )
265              self.fLstSongs.setHorizontalScrollBarPolicy(QtCore.Qt.ScrollBarAlways
    Off)
266             self.fLstSongs.setUniformItemSizes(False)
267             self.fLstSongs.setSelectionRectVisible(True)
268             self.fLstSongs.setObjectName(_fromUtf8("fLstSongs"))
269             self.verticalLayout.addWidget(self.fLstSongs)
270             self.horizontalLayout.addLayout(self.verticalLayout)
271             self.line = QtGui.QFrame(self.centralwidget)
272             self.line.setFrameShape(QtGui.QFrame.VLine)
273             self.line.setFrameShadow(QtGui.QFrame.Sunken)
274             self.line.setObjectName(_fromUtf8("line"))
275             self.horizontalLayout.addWidget(self.line)
276             self.frame = QtGui.QFrame(self.centralwidget)
277                       sizePolicy  =  QtGui.QSizePolicy(QtGui.QSizePolicy.Minimum,
    QtGui.QSizePolicy.Preferred)
278             sizePolicy.setHorizontalStretch(0)
279             sizePolicy.setVerticalStretch(0)
280              sizePolicy.setHeightForWidth(self.frame.sizePolicy().hasHeightForWidt
    h())
281             self.frame.setSizePolicy(sizePolicy)
282             self.frame.setMinimumSize(QtCore.QSize(640, 0))
283             self.frame.setFrameShape(QtGui.QFrame.StyledPanel)
284             self.frame.setFrameShadow(QtGui.QFrame.Raised)
285             self.frame.setLineWidth(0)
286             self.frame.setObjectName(_fromUtf8("frame"))
287             self.verticalLayout_3 = QtGui.QVBoxLayout(self.frame)
288             self.verticalLayout_3.setObjectName(_fromUtf8("verticalLayout_3"))
289             self.frame_2 = QtGui.QFrame(self.frame)
290             self.frame_2.setFrameShape(QtGui.QFrame.NoFrame)
291             self.frame_2.setFrameShadow(QtGui.QFrame.Raised)
292             self.frame_2.setObjectName(_fromUtf8("frame_2"))
293             self.verticalLayout_2 = QtGui.QVBoxLayout(self.frame_2)
294             self.verticalLayout_2.setObjectName(_fromUtf8("verticalLayout_2"))
295                spacerItem = QtGui.QSpacerItem(20, 84, QtGui.QSizePolicy.Minimum,
    QtGui.QSizePolicy.Expanding)
296             self.verticalLayout_2.addItem(spacerItem)
297             self.fLblSongTitle = QtGui.QLabel(self.frame_2)
298             font = QtGui.QFont()
299             font.setPointSize(14)
300             self.fLblSongTitle.setFont(font)
301             self.fLblSongTitle.setAlignment(QtCore.Qt.AlignCenter)
302             self.fLblSongTitle.setObjectName(_fromUtf8("fLblSongTitle"))
303             self.verticalLayout_2.addWidget(self.fLblSongTitle)
304             self.frame_3 = QtGui.QFrame(self.frame_2)
305             self.frame_3.setFrameShape(QtGui.QFrame.NoFrame)
306             self.frame_3.setFrameShadow(QtGui.QFrame.Raised)
307             self.frame_3.setObjectName(_fromUtf8("frame_3"))
308             self.horizontalLayout_2 = QtGui.QHBoxLayout(self.frame_3)
```

```
309             self.horizontalLayout_2.setObjectName(_fromUtf8("horizontalLayout_2")
    )
310             spacerItem1 = QtGui.QSpacerItem(197, 20, QtGui.QSizePolicy.Expanding,
    QtGui.QSizePolicy.Minimum)
311             self.horizontalLayout_2.addItem(spacerItem1)
312             self.fBtnPlay = QtGui.QPushButton(self.frame_3)
313             self.fBtnPlay.setMinimumSize(QtCore.QSize(188, 78))
314             font = QtGui.QFont()
315             font.setPointSize(26)
316             font.setBold(True)
317             font.setWeight(75)
318             self.fBtnPlay.setFont(font)
319             self.fBtnPlay.setCheckable(False)
320             self.fBtnPlay.setObjectName(_fromUtf8("fBtnPlay"))
321             self.horizontalLayout_2.addWidget(self.fBtnPlay)
322             spacerItem2 = QtGui.QSpacerItem(197, 20, QtGui.QSizePolicy.Expanding,
    QtGui.QSizePolicy.Minimum)
323             self.horizontalLayout_2.addItem(spacerItem2)
324             self.verticalLayout_2.addWidget(self.frame_3)
325             self.fLblTime = QtGui.QLabel(self.frame_2)
326             font = QtGui.QFont()
327             font.setPointSize(23)
328             self.fLblTime.setFont(font)
329             self.fLblTime.setAlignment(QtCore.Qt.AlignCenter)
330             self.fLblTime.setObjectName(_fromUtf8("fLblTime"))
331             self.verticalLayout_2.addWidget(self.fLblTime)
332               spacerItem3 = QtGui.QSpacerItem(20, 200, QtGui.QSizePolicy.Minimum,
    QtGui.QSizePolicy.Expanding)
333             self.verticalLayout_2.addItem(spacerItem3)
334             self.verticalLayout_3.addWidget(self.frame_2)
335             self.line_2 = QtGui.QFrame(self.frame)
336             self.line_2.setFrameShape(QtGui.QFrame.HLine)
337             self.line_2.setFrameShadow(QtGui.QFrame.Sunken)
338             self.line_2.setObjectName(_fromUtf8("line_2"))
339             self.verticalLayout_3.addWidget(self.line_2)
340             self.frame_4 = QtGui.QFrame(self.frame)
341             self.frame_4.setFrameShape(QtGui.QFrame.NoFrame)
342             self.frame_4.setFrameShadow(QtGui.QFrame.Raised)
343             self.frame_4.setObjectName(_fromUtf8("frame_4"))
344             self.horizontalLayout_3 = QtGui.QHBoxLayout(self.frame_4)
345             self.horizontalLayout_3.setObjectName(_fromUtf8("horizontalLayout_3")
    )
346              spacerItem4 = QtGui.QSpacerItem(40, 20, QtGui.QSizePolicy.Expanding,
    QtGui.QSizePolicy.Minimum)
347             self.horizontalLayout_3.addItem(spacerItem4)
348             self.label_2 = QtGui.QLabel(self.frame_4)
349             self.label_2.setObjectName(_fromUtf8("label_2"))
350             self.horizontalLayout_3.addWidget(self.label_2)
351             self.fSldVolume = QtGui.QSlider(self.frame_4)
352                         sizePolicy   =   QtGui.QSizePolicy(QtGui.QSizePolicy.Fixed,
    QtGui.QSizePolicy.Fixed)
353             sizePolicy.setHorizontalStretch(0)
354             sizePolicy.setVerticalStretch(0)
```

```
355             sizePolicy.setHeightForWidth(self.fSldVolume.sizePolicy().hasHeightFo
    rWidth())
356             self.fSldVolume.setSizePolicy(sizePolicy)
357             self.fSldVolume.setMinimumSize(QtCore.QSize(225, 0))
358             self.fSldVolume.setBaseSize(QtCore.QSize(0, 0))
359             self.fSldVolume.setAcceptDrops(False)
360             self.fSldVolume.setTracking(True)
361             self.fSldVolume.setOrientation(QtCore.Qt.Horizontal)
362             self.fSldVolume.setInvertedAppearance(False)
363             self.fSldVolume.setInvertedControls(False)
364             self.fSldVolume.setObjectName(_fromUtf8("fSldVolume"))
365             self.horizontalLayout_3.addWidget(self.fSldVolume)
366             self.verticalLayout_3.addWidget(self.frame_4)
367             self.horizontalLayout.addWidget(self.frame)
368             MainWindow.setCentralWidget(self.centralwidget)
369             self.menubar = QtGui.QMenuBar(MainWindow)
370             self.menubar.setGeometry(QtCore.QRect(0, 0, 965, 31))
371             self.menubar.setObjectName(_fromUtf8("menubar"))
372             MainWindow.setMenuBar(self.menubar)
373             self.statusbar = QtGui.QStatusBar(MainWindow)
374             self.statusbar.setObjectName(_fromUtf8("statusbar"))
375             MainWindow.setStatusBar(self.statusbar)
376 
377             self.retranslateUi(MainWindow)
378             QtCore.QMetaObject.connectSlotsByName(MainWindow)
379 
380         def retranslateUi(self, MainWindow):
381             MainWindow.setWindowTitle(_translate("MainWindow", "myplayer", None))
382             self.label.setText(_translate("MainWindow", "Songlist", None))
383             self.fLblSongTitle.setText(_translate("MainWindow", "Artist - Track",
    None))
384             self.fBtnPlay.setText(_translate("MainWindow", "Play", None))
385             self.fLblTime.setText(_translate("MainWindow", "00:00", None))
386             self.label_2.setText(_translate("MainWindow", "Volume", None))
387 
388 
389     if __name__ == "__main__":
390         import sys
391         app = QtGui.QApplication(sys.argv)
392         MainWindow = QtGui.QMainWindow()
393         ui = Ui_MainWindow()
394         ui.setupUi(MainWindow)
395         MainWindow.show()
396         sys.exit(app.exec_())
```

Listing: Lesson 3, gui.py with event handlers

```
397     # -*- coding: utf-8 -*-
398 
399     # Form implementation generated from reading ui file './gui.ui'
400     #
401 
402     import sys
403 
```

```
404    from PyQt5 import QtCore, QtGui, QtWidgets
405
406    class TUILocal:
407        def __init__ (self, delegate):
408            self.fApp      = None
409            self.fDelegate = delegate
410            self.fWndMain  = None
411
412        def Handle_List_Click (self, item):
413            pass
414
415        def Handle_Play_Click (self):
416            pass
417
418        def Handle_Volume_Moved (self, currentValue):
419            pass
420
421        def UIStart (self):
422            self.fDelegate.Handle_HasInitStarted ()
423            self.fApp        = QtWidgets.QApplication (sys.argv)
424            self.fWndMain    = QtWidgets.QMainWindow ()
425            self._setupUi (self.fWndMain)
426            self.fWndMain.show ()
427            self.fDelegate.Handle_HasInitFinished ()
428            self.fApp.exec ()
429
430        def _setupUi(self, mainWindow):
431            mainWindow.setObjectName("MainWindow")
432            mainWindow.resize(965, 600)
433            self.centralwidget = QtWidgets.QWidget(mainWindow)
434            self.centralwidget.setObjectName("centralwidget")
435            self.horizontalLayout = QtWidgets.QHBoxLayout(self.centralwidget)
436            self.horizontalLayout.setObjectName("horizontalLayout")
437            self.verticalLayout = QtWidgets.QVBoxLayout()
438             self.verticalLayout.setSizeConstraint(QtWidgets.QLayout.SetDefaultCon
   straint)
439            self.verticalLayout.setContentsMargins(0, 0, 0, -1)
440            self.verticalLayout.setObjectName("verticalLayout")
441            self.label = QtWidgets.QLabel(self.centralwidget)
442            font = QtGui.QFont()
443            font.setPointSize(14)
444            font.setBold(True)
445            font.setWeight(75)
446            self.label.setFont(font)
447            self.label.setTextFormat(QtCore.Qt.PlainText)
448            self.label.setObjectName("label")
449            self.verticalLayout.addWidget(self.label)
450            self.fLstSongs = QtWidgets.QListWidget(self.centralwidget)
451            font = QtGui.QFont()
452            font.setPointSize(14)
453            font.setBold(True)
454            font.setWeight(75)
455            self.fLstSongs.setFont(font)
```

```
456             self.fLstSongs.setVerticalScrollBarPolicy(QtCore.Qt.ScrollBarAlwaysOn
    )
457             self.fLstSongs.setHorizontalScrollBarPolicy(QtCore.Qt.ScrollBarAlways
    Off)
458             self.fLstSongs.setUniformItemSizes(False)
459             self.fLstSongs.setSelectionRectVisible(True)
460             self.fLstSongs.setObjectName("fLstSongs")
461             self.verticalLayout.addWidget(self.fLstSongs)
462             self.horizontalLayout.addLayout(self.verticalLayout)
463             self.line = QtWidgets.QFrame(self.centralwidget)
464             self.line.setFrameShape(QtWidgets.QFrame.VLine)
465             self.line.setFrameShadow(QtWidgets.QFrame.Sunken)
466             self.line.setObjectName("line")
467             self.horizontalLayout.addWidget(self.line)
468             self.frame = QtWidgets.QFrame(self.centralwidget)
469                 sizePolicy = QtWidgets.QSizePolicy(QtWidgets.QSizePolicy.Minimum,
    QtWidgets.QSizePolicy.Preferred)
470             sizePolicy.setHorizontalStretch(0)
471             sizePolicy.setVerticalStretch(0)
472             sizePolicy.setHeightForWidth(self.frame.sizePolicy().hasHeightForWidt
    h())
473             self.frame.setSizePolicy(sizePolicy)
474             self.frame.setMinimumSize(QtCore.QSize(640, 0))
475             self.frame.setFrameShape(QtWidgets.QFrame.StyledPanel)
476             self.frame.setFrameShadow(QtWidgets.QFrame.Raised)
477             self.frame.setLineWidth(0)
478             self.frame.setObjectName("frame")
479             self.verticalLayout_3 = QtWidgets.QVBoxLayout(self.frame)
480             self.verticalLayout_3.setObjectName("verticalLayout_3")
481             self.frame_2 = QtWidgets.QFrame(self.frame)
482             self.frame_2.setFrameShape(QtWidgets.QFrame.NoFrame)
483             self.frame_2.setFrameShadow(QtWidgets.QFrame.Raised)
484             self.frame_2.setObjectName("frame_2")
485             self.verticalLayout_2 = QtWidgets.QVBoxLayout(self.frame_2)
486             self.verticalLayout_2.setObjectName("verticalLayout_2")
487                                 spacerItem    =    QtWidgets.QSpacerItem(20,    84,
    QtWidgets.QSizePolicy.Minimum, QtWidgets.QSizePolicy.Expanding)
488             self.verticalLayout_2.addItem(spacerItem)
489             self.fLblSongTitle = QtWidgets.QLabel(self.frame_2)
490             font = QtGui.QFont()
491             font.setPointSize(14)
492             self.fLblSongTitle.setFont(font)
493             self.fLblSongTitle.setAlignment(QtCore.Qt.AlignCenter)
494             self.fLblSongTitle.setObjectName("fLblSongTitle")
495             self.verticalLayout_2.addWidget(self.fLblSongTitle)
496             self.frame_3 = QtWidgets.QFrame(self.frame_2)
497             self.frame_3.setFrameShape(QtWidgets.QFrame.NoFrame)
498             self.frame_3.setFrameShadow(QtWidgets.QFrame.Raised)
499             self.frame_3.setObjectName("frame_3")
500             self.horizontalLayout_2 = QtWidgets.QHBoxLayout(self.frame_3)
501             self.horizontalLayout_2.setObjectName("horizontalLayout_2")
502                                 spacerItem1    =    QtWidgets.QSpacerItem(197,    20,
    QtWidgets.QSizePolicy.Expanding, QtWidgets.QSizePolicy.Minimum)
```

```
503             self.horizontalLayout_2.addItem(spacerItem1)
504             self.fBtnPlay = QtWidgets.QPushButton(self.frame_3)
505             self.fBtnPlay.setMinimumSize(QtCore.QSize(188, 78))
506             font = QtGui.QFont()
507             font.setPointSize(26)
508             font.setBold(True)
509             font.setWeight(75)
510             self.fBtnPlay.setFont(font)
511             self.fBtnPlay.setFocusPolicy (QtCore.Qt.NoFocus)
512             self.fBtnPlay.setCheckable(False)
513             self.fBtnPlay.setObjectName("fBtnPlay")
514             self.horizontalLayout_2.addWidget(self.fBtnPlay)
515                                 spacerItem2    =    QtWidgets.QSpacerItem(197,    20,
    QtWidgets.QSizePolicy.Expanding, QtWidgets.QSizePolicy.Minimum)
516             self.horizontalLayout_2.addItem(spacerItem2)
517             self.verticalLayout_2.addWidget(self.frame_3)
518             self.fLblTime = QtWidgets.QLabel(self.frame_2)
519             font = QtGui.QFont()
520             font.setPointSize(23)
521             self.fLblTime.setFont(font)
522             self.fLblTime.setAlignment(QtCore.Qt.AlignCenter)
523             self.fLblTime.setObjectName("fLblTime")
524             self.verticalLayout_2.addWidget(self.fLblTime)
525                                 spacerItem3    =    QtWidgets.QSpacerItem(20,    200,
    QtWidgets.QSizePolicy.Minimum, QtWidgets.QSizePolicy.Expanding)
526             self.verticalLayout_2.addItem(spacerItem3)
527             self.verticalLayout_3.addWidget(self.frame_2)
528             self.line_2 = QtWidgets.QFrame(self.frame)
529             self.line_2.setFrameShape(QtWidgets.QFrame.HLine)
530             self.line_2.setFrameShadow(QtWidgets.QFrame.Sunken)
531             self.line_2.setObjectName("line_2")
532             self.verticalLayout_3.addWidget(self.line_2)
533             self.frame_4 = QtWidgets.QFrame(self.frame)
534             self.frame_4.setFrameShape(QtWidgets.QFrame.NoFrame)
535             self.frame_4.setFrameShadow(QtWidgets.QFrame.Raised)
536             self.frame_4.setObjectName("frame_4")
537             self.horizontalLayout_3 = QtWidgets.QHBoxLayout(self.frame_4)
538             self.horizontalLayout_3.setObjectName("horizontalLayout_3")
539                                  spacerItem4    =    QtWidgets.QSpacerItem(40,    20,
    QtWidgets.QSizePolicy.Expanding, QtWidgets.QSizePolicy.Minimum)
540             self.horizontalLayout_3.addItem(spacerItem4)
541             self.label_2 = QtWidgets.QLabel(self.frame_4)
542             self.label_2.setObjectName("label_2")
543             self.horizontalLayout_3.addWidget(self.label_2)
544             self.fSldVolume = QtWidgets.QSlider(self.frame_4)
545                  sizePolicy  =  QtWidgets.QSizePolicy(QtWidgets.QSizePolicy.Fixed,
    QtWidgets.QSizePolicy.Fixed)
546             sizePolicy.setHorizontalStretch(0)
547             sizePolicy.setVerticalStretch(0)
548              sizePolicy.setHeightForWidth(self.fSldVolume.sizePolicy().hasHeightFo
    rWidth())
549             self.fSldVolume.setSizePolicy(sizePolicy)
550             self.fSldVolume.setMinimumSize(QtCore.QSize(225, 0))
```

```
551             self.fSldVolume.setBaseSize(QtCore.QSize(0, 0))
552             self.fSldVolume.setAcceptDrops(False)
553             self.fSldVolume.setTracking(True)
554             self.fSldVolume.setOrientation(QtCore.Qt.Horizontal)
555             self.fSldVolume.setInvertedAppearance(False)
556             self.fSldVolume.setInvertedControls(False)
557             self.fSldVolume.setObjectName("fSldVolume")
558             self.horizontalLayout_3.addWidget(self.fSldVolume)
559             self.verticalLayout_3.addWidget(self.frame_4)
560             self.horizontalLayout.addWidget(self.frame)
561             mainWindow.setCentralWidget(self.centralwidget)
562             self.menubar = QtWidgets.QMenuBar(mainWindow)
563             self.menubar.setGeometry(QtCore.QRect(0, 0, 965, 30))
564             self.menubar.setObjectName("menubar")
565             mainWindow.setMenuBar(self.menubar)
566             self.statusbar = QtWidgets.QStatusBar(mainWindow)
567             self.statusbar.setObjectName("statusbar")
568             mainWindow.setStatusBar(self.statusbar)
569 
570             self.fBtnPlay.clicked.connect           (self.Handle_Play_Click     )
571             self.fSldVolume.sliderMoved.connect     (self.Handle_Volume_Moved   )
572             self.fLstSongs.itemClicked.connect      (self.Handle_List_Click     )
573 
574             mainWindow.setWindowTitle               ("myplayer")
575             self.label.setText                      ("Songlist")
576               self.fLblSongTitle.setText                ("Ivan Torrent - Beyond
    love")
577             self.fBtnPlay.setText                   ("Play")
578             self.fLblTime.setText                   ("00:00")
579             self.label_2.setText                    ("Volume")
580 
581             QtCore.QMetaObject.connectSlotsByName(mainWindow)
```