

Institiúid Teicneolaíochta Cheatharlach



INSTITUTE *of*
TECHNOLOGY

CARLOW

At the Heart of South Leinster

IT Carlow

Bachelor of Software Development

Year 4

Portable GUI for ptpython shell

Research Manual

Student Name: **Inga Melkerte**

Student ID: **C00184799**

Supervisor: **Paul Barry**

Date: **31/10/16**

Table of Contents

Table of Contents	1
1. Purpose	3
2. Introduction	3
3. Research and discover ptpython	3
4. Research on available GUI toolkits in Python	5
4.1. Tkinter	5
4.1.1. Description	5
4.1.2. Platforms	5
4.1.3. Tutorials	5
4.1.4. Example	6
4.1.5. Code	7
4.2. TOGA	9
4.2.1. Description	9
4.2.2. Platforms	9
4.2.3. Tutorials	9
4.2.4. Example	10
4.2.5. Code	11
4.3. EasyGui and Tkinter	12
4.3.1. Description	12
4.3.2. Platforms	12
4.3.3. Tutorials	12
4.3.4. Example	13
4.3.5. Code	14
4.4. PyQt	15
4.4.1. Description	15
4.4.2. Platforms	15
4.4.3. Tutorials	16
4.4.4. Example	16
4.4.5. Code	17
4.5. Kivy	18
4.5.1. Description	18
4.5.2. Platforms	18

4.5.3. Tutorials	18
4.5.4. Example	18
4.5.5. Code	19
4.6. PySide	20
4.6.1. Description	20
4.6.2. Platforms	20
4.6.3. Tutorials	20
4.6.4. Example	21
4.6.5. Code	21
4.7. wxPython	22
4.7.1. Description	22
4.7.2. Platforms	22
4.7.3. Tutorials	22
4.7.4. Example	22
4.7.5. Code	22
5. Research Similar Projects	23
5.1. IDLE	23
5.2. THONNY	24
6. Virtual Python Environment builder	25
7. Mercurial - source control tool	25
8. Summary and Conclusion	26
Appendix A	27
Comparison of Python GUI framework toolkits	27
Reference	28

1. Purpose

The purpose of this document is:

- to research and get familiar with ptpython
- to research available GUI Toolkits in python and install them. Then simple application is developed using each GUI to find out how “easy” is to create graphical interface and discover is there enough documentation and tutorials available
- at the end of the research phase decide which GUI framework use to develop portable GUI for ptpython shell
- to choose source version control (research git and mercurial)
- to research similar graphical applications

2. Introduction

The goal of the final 4th year software development project is to develop portable GUI for ptpython shell which one can be used for learning purposes.

3. Research and discover ptpython

Developer of ptpython

Jonathan Slenders - Belgium software developer.

Description of ptpython

Ptpython is a better Python REPL built on top of prompt-toolkit. Prompt-toolkit is a library for designing command line interfaces. Basically, it can be a pure Python replacement for GNU readline. Ptpython is written in pure python where GNU readline in C. (GitHub, 2017).

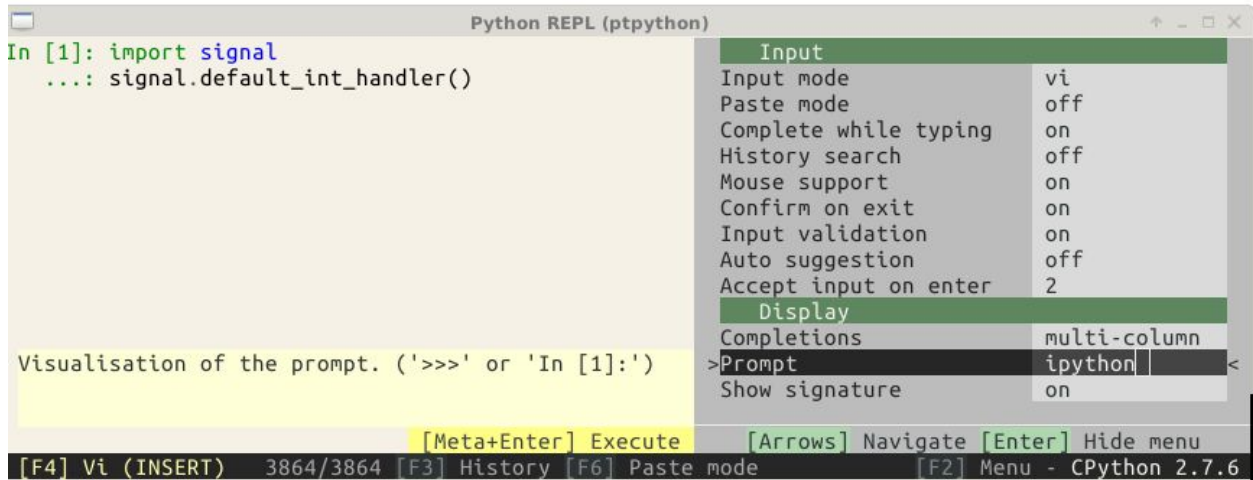
It implements most of the Emacs and Vi keybindings as well as reverse search and code completion and it supports double width characters. However, it can be even much more powerful. There is syntax highlighting of the input (from pygments library), autocompletion, multiline editing, support for toolbars, custom layouts, multiple input fields and integration with other event loops (e.g. asyncio). (GitHub, 2017).

To install ptpython

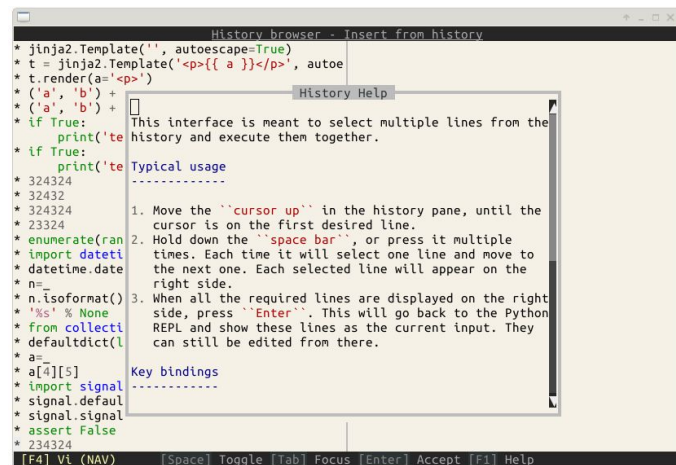
```
pip install ptpython
```

Cool ptython features

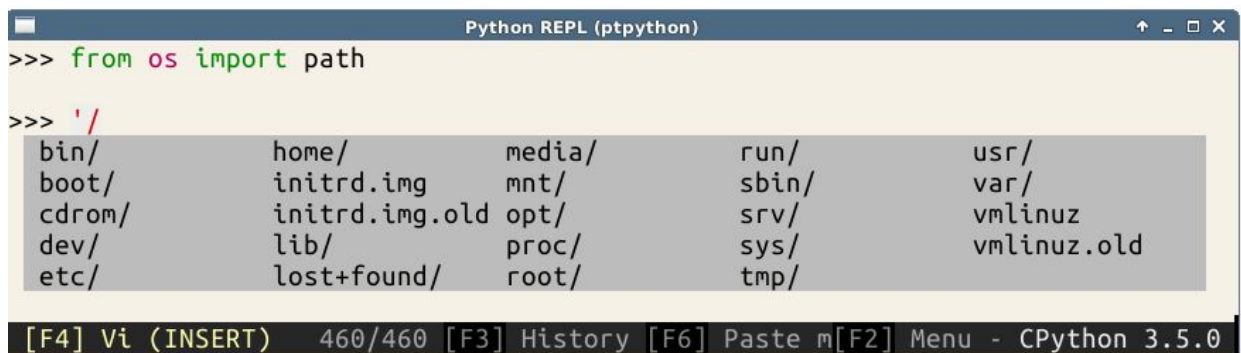
The configuration menu:



History page when F3 function key is pressed



Autocompletion



4. Research on available GUI toolkits in Python

For this project the application is being developed based on existing command-line interface by adding graphical user interface to it. The research shows that there are lots of options for the front end tools in python - more than 30 - such as PyJamas, Dabo, Kivy, PyForms, PyGame, Pyglet, Sugar etc. (Python Wiki, 2016). For deeper research 8 frameworks have been chosen (Tkinter, wxWidgets PYQT, Kivy, PySide, EasyGUI, wxPython and Toga) and at the end of the research the GUI Toolkit Comparison Table has been created. (see Appendix A for details).

4.1. Tkinter

4.1.1. Description

Tkinter is a thin object-oriented layer on top of Tcl/Tk. It has the advantage of being included with the Python standard library, making it the most convenient and compatible toolkit to program with. Tk offers native look and feel on all platforms (Reitz, 2016). Tkinter is implemented as a Python wrapper around a complete Tcl interpreter embedded in the Python interpreter. Tkinter name comes from Tk interface.(Wikipedia, 2016). In Python 2 it was called Tkinter (with capital T) where in Python 3 it was renamed - tkinter.

4.1.2. Platforms

- Windows
- Linux
- Mac

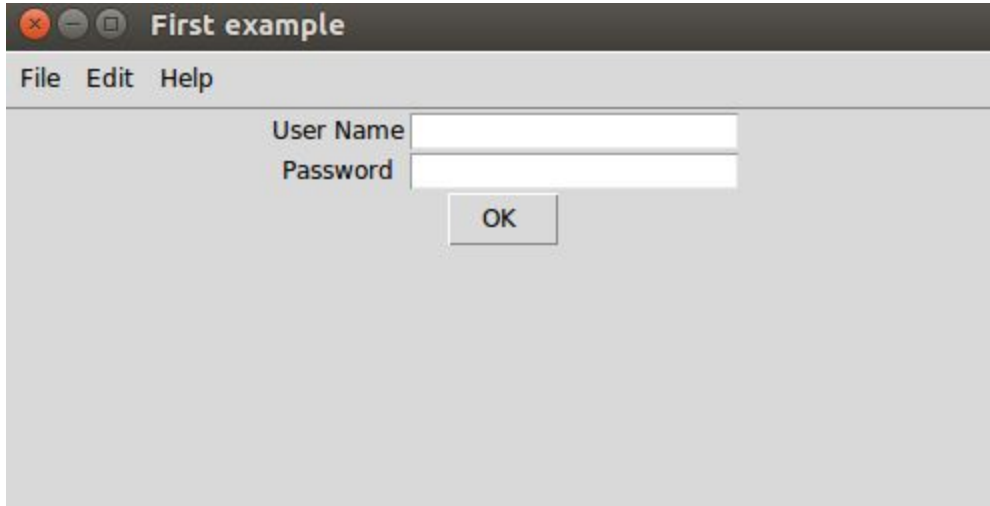
4.1.3. Tutorials

Lots of tutorials available online. Some of them -

- Tkinter. (Tkinter - Python Wiki, 2016)
- Python GUI Programming (tutorialspoint.com, 2016)
- An Introduction to Tkinter (Effbot, 2016)
- Building a GUI Application with Tkinter (Meyers, 2016)

4.1.4. Example

Window, title, menu, label, textbox, button



4.1.5. Code

```
from tkinter import *

def donothing():
    filewin = Toplevel(root)
    button = Button(filewin, text="Do nothing button Inga")
    button.pack()

def addEntry():
    phonenumber.append ([nameVar.get(), phoneVar.get()])
    setSelect ()

def show_entry_fields():
    print("First Name: %s\nLast Name: %s" % (e1.get(), e2.get()))

root = Tk()
root.title("First example")
root.geometry("500x200")
#frame1
frame1 = Frame(root)
frame1.pack()
menubar = Menu(root)
filemenu = Menu(menubar, tearoff=0)
filemenu.add_command(label="New", command=donothing)
filemenu.add_command(label="Open", command=donothing)
filemenu.add_command(label="Save", command=donothing)
filemenu.add_command(label="Save as...", command=donothing)
filemenu.add_command(label="Close", command=donothing)

filemenu.add_separator()

filemenu.add_command(label="Exit", command=root.quit)
menubar.add_cascade(label="File", menu=filemenu)
editmenu = Menu(menubar, tearoff=0)
editmenu.add_command(label="Undo", command=donothing)

editmenu.add_separator()
```



```
editmenu.add_command(label="Cut", command=donothing)
editmenu.add_command(label="Copy", command=donothing)
editmenu.add_command(label="Paste", command=donothing)
editmenu.add_command(label="Delete", command=donothing)
editmenu.add_command(label="Select All", command=donothing)

menubar.add_cascade(label="Edit", menu=editmenu)
helpmenu = Menu(menubar)#, tearoff=0)
helpmenu.add_command(label="Help Index", command=donothing)
helpmenu.add_command(label="About...", command=donothing)
menubar.add_cascade(label="Help", menu=helpmenu)
root.config(menu=menubar)

#frame2
frame2 = Frame(root)
frame2.pack()
Label(frame2, text="User Name").grid(row =0)
Label(frame2, text="Password").grid(row =1)

e1 = Entry(frame2)
e2 = Entry(frame2)

e1.grid(row = 0,column =1)
e2.grid(row = 1,column =1)

#frame3
frame3 = Frame(root)
frame3.pack()

b1 = Button(frame3,text=" OK  ",command=show_entry_fields)

b1.pack(side=RIGHT)

root.mainloop()
```

4.2. TOGA

4.2.1. Description

Toga is a part of BeeWare suite. Toga is a Python native, OS native, cross platform GUI toolkit. Toga consists of a library of base components with a shared interface to simplify platform-agnostic GUI development (Keith-Magee, 2016). Interesting features about Toga -it uses an approach that is new for widget toolkits, but well proven in computing: Cascading Style Sheets, (CSS). Other widget toolkits use different approaches - constraints, packing models, and grid-based models are all common (Keith-Magee, 2016).It is new tool , only 0.1.0 release. Documentation has only been added 24/3/17 (more detail in Appendix A).

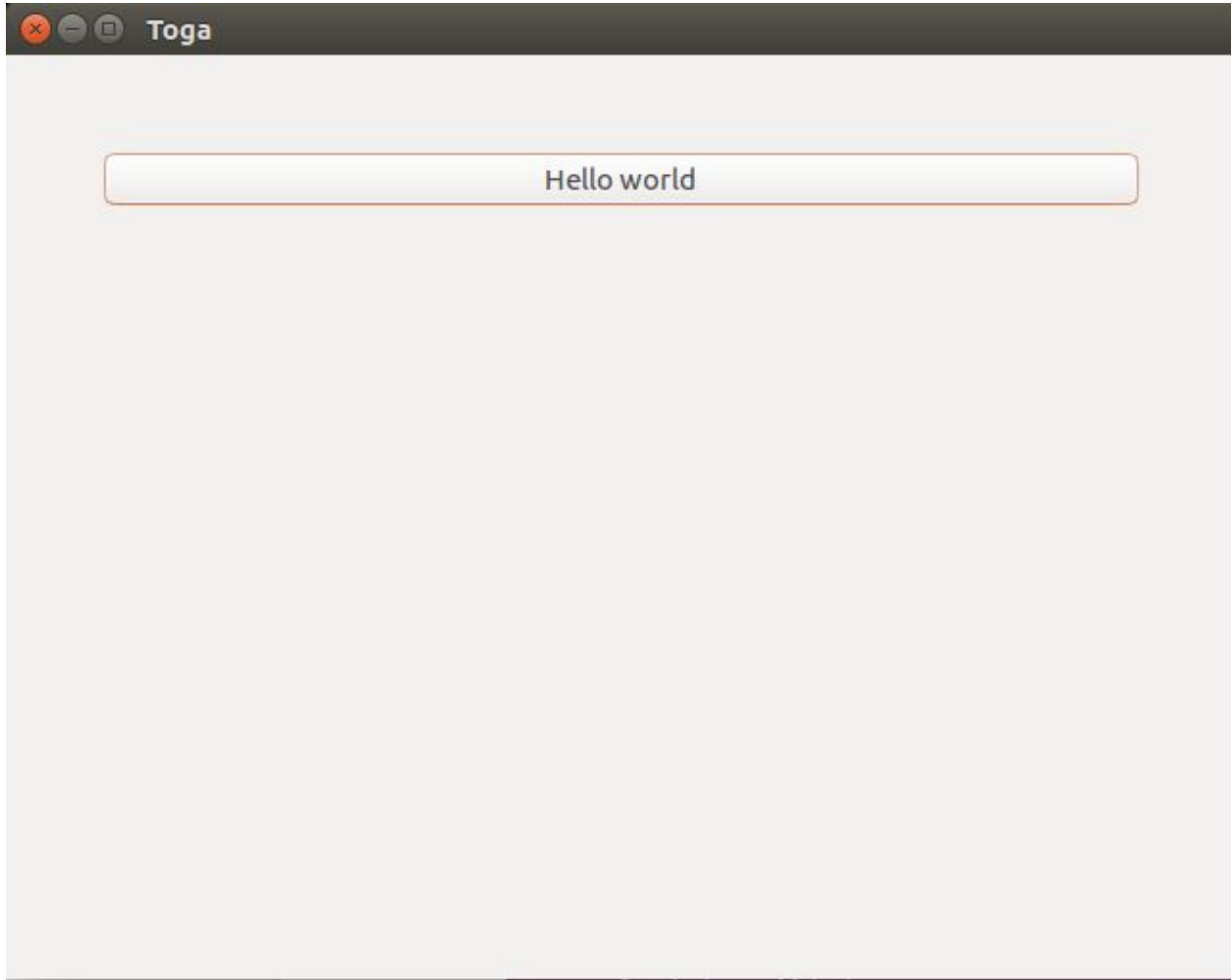
4.2.2. Platforms

- Windows
- Mac
- Linux
- Mobile Platforms -Android, iOS (Keith-Magee, 2016).

4.2.3. Tutorials

- A slightly less toy example — Toga 0.2.5.dev1 documentation (Keith-Magee, R., 2016)
- Toga Documentation. (Keith-Magee, R, 2017)
- Toga: yet another GUI toolkit on Python (YouTube, 2016)
- Widget Reference — Toga 0.2.5.dev1 documentation. (Widget Reference — Toga 0.2.5.dev1 documentation, 2017)
- Episode #79 Beeware Python Tools - [Talk Python To Me Podcast]. (Michael Kennedy, 2016)

4.2.4. Example



4.2.5. Code

```
import toga

def button_handler(widget):
    print("hello")

def build(app):
    box = toga.Box()

    button = toga.Button('Hello world', on_press=button_handler)
    button.style.set(margin=50)
    box.add(button)

    return box

if __name__ == '__main__':
    app = toga.App('First Example in Toga', 'org.pybee.helloworld', startup=build)
    app.main_loop()
```

4.3. EasyGui and Tkinter

4.3.1. Description

EasyGUI is a module for very simple, very easy GUI programming in Python. EasyGUI is different from other GUI generators in that EasyGUI is NOT event-driven. Instead, all GUI interactions are invoked by simple function calls (Zawadzki, 2016). EasyGui provides an easy-to-use interface for simple GUI interaction with a user. It does not require the programmer to know anything about tkinter, frames, widgets, callbacks or lambda. All GUI interactions are invoked by simple function calls that return results (EasyGui Tutorial, 2017).

It was difficult to install easygui but once I did managed to install it , it was very easy to use it. EasyGui documentation was updated only in 06/01/17 and that was a great help. To install in python or anaconda just type - *pip install --upgrade easygui. (on Linux)*.

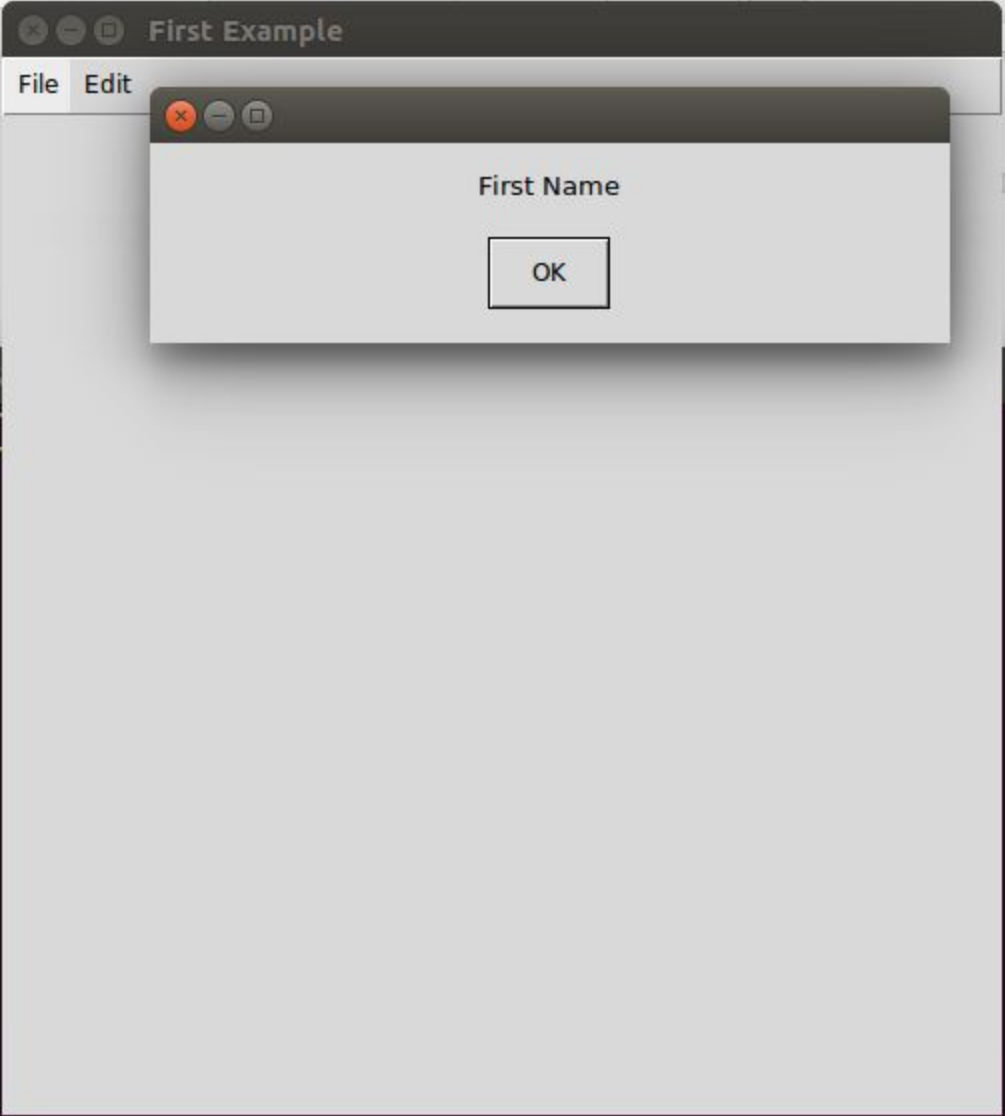
4.3.2. Platforms

- Windows
- Linux
- Mac

4.3.3. Tutorials

- easygui 0.98.0 - (Zawadzki, 2016)
- EasyGui Tutorial — easygui 0.97 (2014-12-20) documentation. (EasyGui Tutorial, 2017).
- easygui Documentation - (easygui dev team, 2017).

4.3.4. Example



4.3.5. Code

```
1 from easygui import *
2 from tkinter import *
3 import os
4
5
6 #function calls easygui msgbox
7 def newfile():
8     msgbox("First Name")
9
10 root = Tk()
11 root.title("First Example")
12 root.geometry("500x500")
13
14 #frame1 in tkinter
15 frame1 = Frame(root)
16 frame1.pack()
17 menubar = Menu(root)
18 filemenu = Menu(menubar, tearoff=0)
19 filemenu.add_command(label="new", command=newfile)
20 filemenu.add_command(label="save")
21 #filemenu.add_separator()
22 menubar.add_cascade(label="File", menu=filemenu)
23
24 editmenu = Menu(menubar, tearoff=0)
25 editmenu.add_command(label="copy")
26 editmenu.add_command(label="paste")
27 editmenu.add_command(label="cut")
28 menubar.add_cascade(label="Edit", menu=editmenu)
29 root.config(menu=menubar)
30
31 root.mainloop()
```

4.4. PyQt

4.4.1. Description

PyQt is a GUI widgets toolkit. It is a Python interface for Qt, one of the most powerful, and popular cross-platform GUI library (Tutorialspoint.com, 2016).

PyQt is dual licensed on all supported platforms under the GNU GPL v3 and the Riverbank Commercial License (EasyGui Tutorial, 2017).

PyQt is a GUI widgets toolkit. It is a Python interface for Qt, one of the most powerful, and popular cross-platform GUI library.

Users of Qt include Adobe, Google, IBM, Sharp, and Siemens. Adobe Photoshop Album and Google Earth are just two examples of widely used cross-platform Qt Applications (Tutorialspoint.com, 2016).

Problems installing PyQt

```
lnaga@lnaga-SATELLITE-C855-1J2:~$ pip3 install pyqt5
Collecting pyqt5
  Downloading PyQt5-5.7-cp35-cp35m-manylinux1_x86_64.whl (89.8MB)
    100% |#####| 89.8MB 18kB/s
Collecting sip (from pyqt5)
  Downloading sip-4.18.1-cp35-cp35m-manylinux1_x86_64.whl (60kB)
    100% |#####| 61kB 2.0MB/s
Installing collected packages: sip, pyqt5
Exception:
Traceback (most recent call last):
  File "/usr/local/lib/python3.5/dist-packages/pip/basecommand.py", line 215, in main
    status = self.run(options, args)
  File "/usr/local/lib/python3.5/dist-packages/pip/commands/install.py", line 317, in run
    prefix=options.prefix_path,
  File "/usr/local/lib/python3.5/dist-packages/pip/req/req_set.py", line 742, in install
    **kwargs
  File "/usr/local/lib/python3.5/dist-packages/pip/req/req_install.py", line 831, in install
    self.move_wheel_files(self.source_dir, root=root, prefix=prefix)
  File "/usr/local/lib/python3.5/dist-packages/pip/req/req_install.py", line 1032, in move_wheel_files
    isolated=self.isolated,
  File "/usr/local/lib/python3.5/dist-packages/pip/wheel.py", line 346, in move_wheel_files
    clobber(source, lib_dir, True)
  File "/usr/local/lib/python3.5/dist-packages/pip/wheel.py", line 324, in clobber
    shutil.copyfile(srcfile, destfile)
  File "/usr/lib/python3.5/shutil.py", line 115, in copyfile
    with open(dst, 'wb') as fdst:
PermissionError: [Errno 13] Permission denied: '/usr/local/lib/python3.5/dist-packages/sip.so'
lnaga@lnaga-SATELLITE-C855-1J2:~$
```

Got fixed by following answers in

<http://stackoverflow.com/questions/36175717/importerror-no-module-named-pyqt5-qtquick>
`sudo apt-get install python3-pyqt5.qtquick`

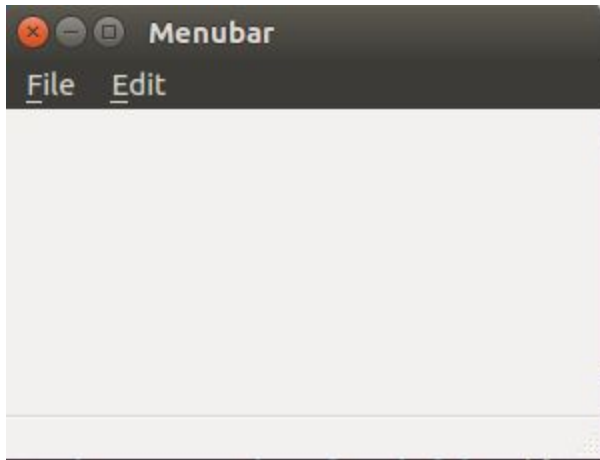
4.4.2. Platforms

- Windows
- Linux
- Mac
- Mobile - iOS, Android

4.4.3. Tutorials

- PyQt5 window – Python Tutorial. - (PyQt5 window, 2016).
- PyQt QMessageBox. - (Tutorialspoint.com, 2016).
- PyQt5 statusbar – Python Tutorial, (PyQt5 statusbar – Python Tutorial, 2016).

4.4.4. Example



4.4.5. Code

```
1 from easygui import *
2 from tkinter import *
3 import os
4
5
6 #function calls easygui msgbox
7 def newfile():
8     msgbox("First Name")
9
10 root = Tk()
11 root.title("First Example")
12 root.geometry("500x500")
13
14 #frame1 in tkinter
15 frame1 = Frame(root)
16 frame1.pack()
17 menubar = Menu(root)
18 filemenu = Menu(menubar, tearoff=0)
19 filemenu.add_command(label="new", command=newfile)
20 filemenu.add_command(label="save")
21 #filemenu.add_separator()
22 menubar.add_cascade(label="File", menu=filemenu)
23
24 editmenu = Menu(menubar, tearoff=0)
25 editmenu.add_command(label="copy")
26 editmenu.add_command(label="paste")
27 editmenu.add_command(label="cut")
28 menubar.add_cascade(label="Edit", menu=editmenu)
29 root.config(menu=menubar)
30
31 root.mainloop()
```

4.5. Kivy

4.5.1. Description

Kivy - Open source Python library for rapid development of applications that make use of innovative user interfaces, such as multi-touch apps (Kivy, 2016). Kivy is used for writing multi touch applications. for attractive, tablet-style interfaces (Kluyver, 2016).

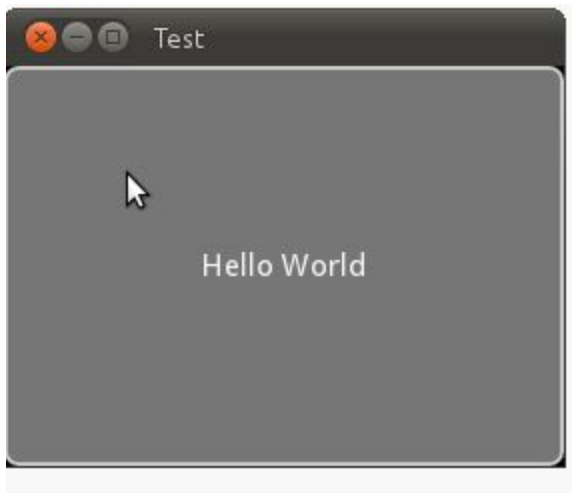
4.5.2. Platforms

- Windows
- Linux
- Mac
- Mobile (iOS, Android)

4.5.3. Tutorials

- Python Programming Tutorials. - (Harrison, 2016).
- Kivy crash course - YouTube. - (Taylor, A., 2016).

4.5.4. Example



1.Screenshot from (Kivy, 2016)

4.5.5. Code

```
from kivy.app import App
from kivy.uix.button import Button

class TestApp(App):
    def build(self):
        return Button(text='Hello World')

TestApp().run()
```

2.Screenshot from (Kivy, 2016)

4.6. PySide

4.6.1. Description

Qt is a cross-platform application framework from Qt Software (owned by Nokia). It features a large number of libraries providing services like network abstraction and XML handling, along with a very rich GUI package, allowing C++ developers to write their applications once and run them unmodified in different systems. PySide aims to provide Python developers access to the Qt libraries in the most natural way (PySide, 2016).

PySide is an open source software project providing Python bindings for the Qt framework. Qt is a cross-platform application and UI framework, allowing the developers to write applications once and deploy them across many operating systems without rewriting the source code, while Python is a modern, dynamic programming language with a vivid developer community.

Combining the power of Qt and Python, PySide provides the wealth of Qt framework for developers writing software in Python and presents a first-class rapid application development platform available on all major operating systems.

4.6.2. Platforms

- Windows
- Linux
- Mac

4.6.3. Tutorials

- About PySide - Qt Wiki. - (Qt Wiki, 2016).
- First programs in PySide. - (Bodnar, 2016).
- Overview — PySide 1.2.1 documentation. - (PySide, 2016).

4.6.4. Example



4.6.5. Code

```
1 #!/usr/bin/python
2
3 # Import PySide classes
4 import sys
5 from PySide.QtCore import *
6 from PySide.QtGui import *
7
8 # Create a Qt application
9 app = QApplication(sys.argv)
10 # Create a Label and show it
11 label = QLabel("<font color=red size=40>Simple Example in PySides</font>")
12 label.show()
13 # Enter Qt application main loop
14 app.exec_()
15 sys.exit()
```

4.7. wxPython

4.7.1. Description

wxPython is a Python wrapper for wxWidgets (which is written in C++), a popular cross-platform GUI toolkit. Qt Wiki. (2016).

wxPython is used for - PythonCard is a GUI construction kit for building cross-platform desktop applications on Windows, Mac OS X, and Linux, using the Python language (Python Wiki, 2016).

Problems installing in python3. Runs only in python2.7

4.7.2. Platforms

- Windows
- Linux
- Mac

4.7.3. Tutorials

- wxPython Tutorial. - (tutorialspoint.com, 2016.)
- API Documentation — wxPython Phoenix 3.0.3 documentation. - (wxPython Team, 2016).
- GUI Applications — The Hitchhiker's Guide to Python. - (Reitz, 2016).
- "Wxpython in Action." (Rappin, 2006).

4.7.4. Example



4.7.5. Code

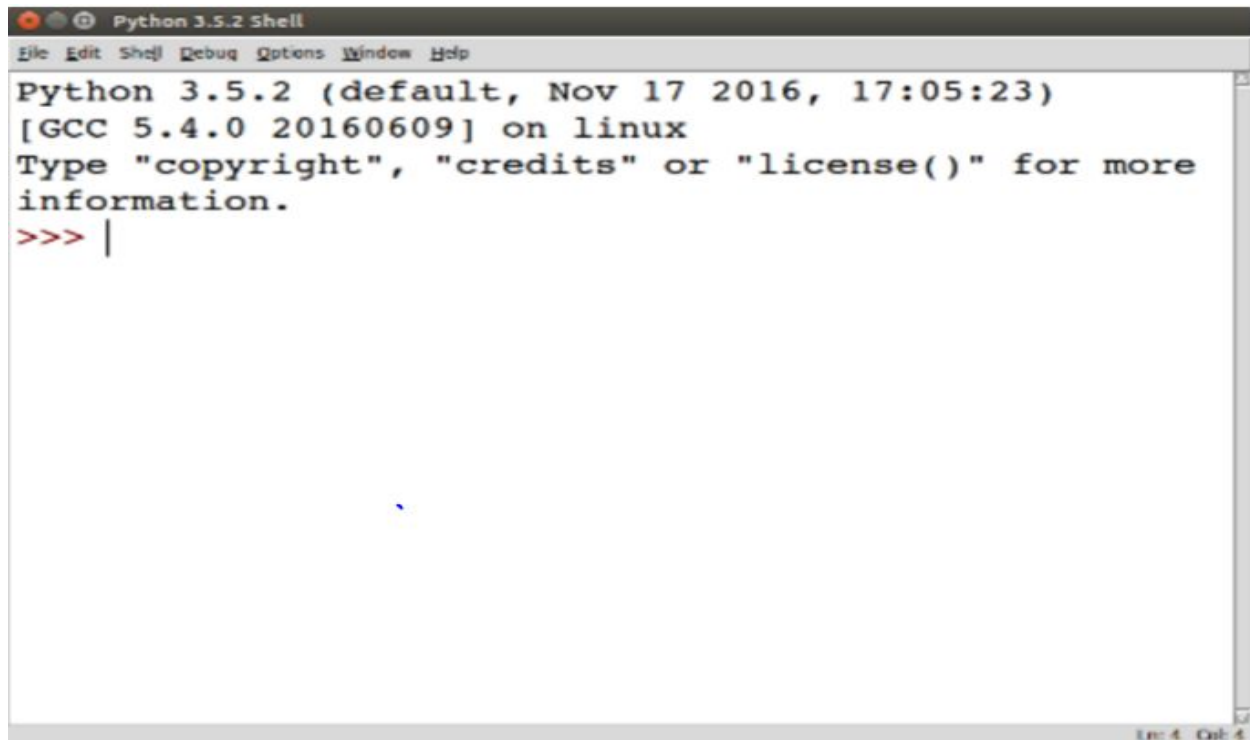
5. Research Similar Projects

5.1. IDLE

IDLE is Python's Integrated Development and Learning Environment.

IDLE has the following features:

- coded in 100% pure Python, using the **tkinter** GUI toolkit
- cross-platform: works mostly the same on Windows, Unix, and Mac OS X
- Python shell window (interactive interpreter) with colorizing of code input, output, and error messages
- multi-window text editor with multiple undo, Python colorizing, smart indent, call tips, auto completion, and other features
- search within any window, replace within editor windows, and search through multiple files (grep)
- debugger with persistent breakpoints, stepping, and viewing of global and local namespaces
- configuration, browsers, and other dialogs (24.6. IDLE, 2016)



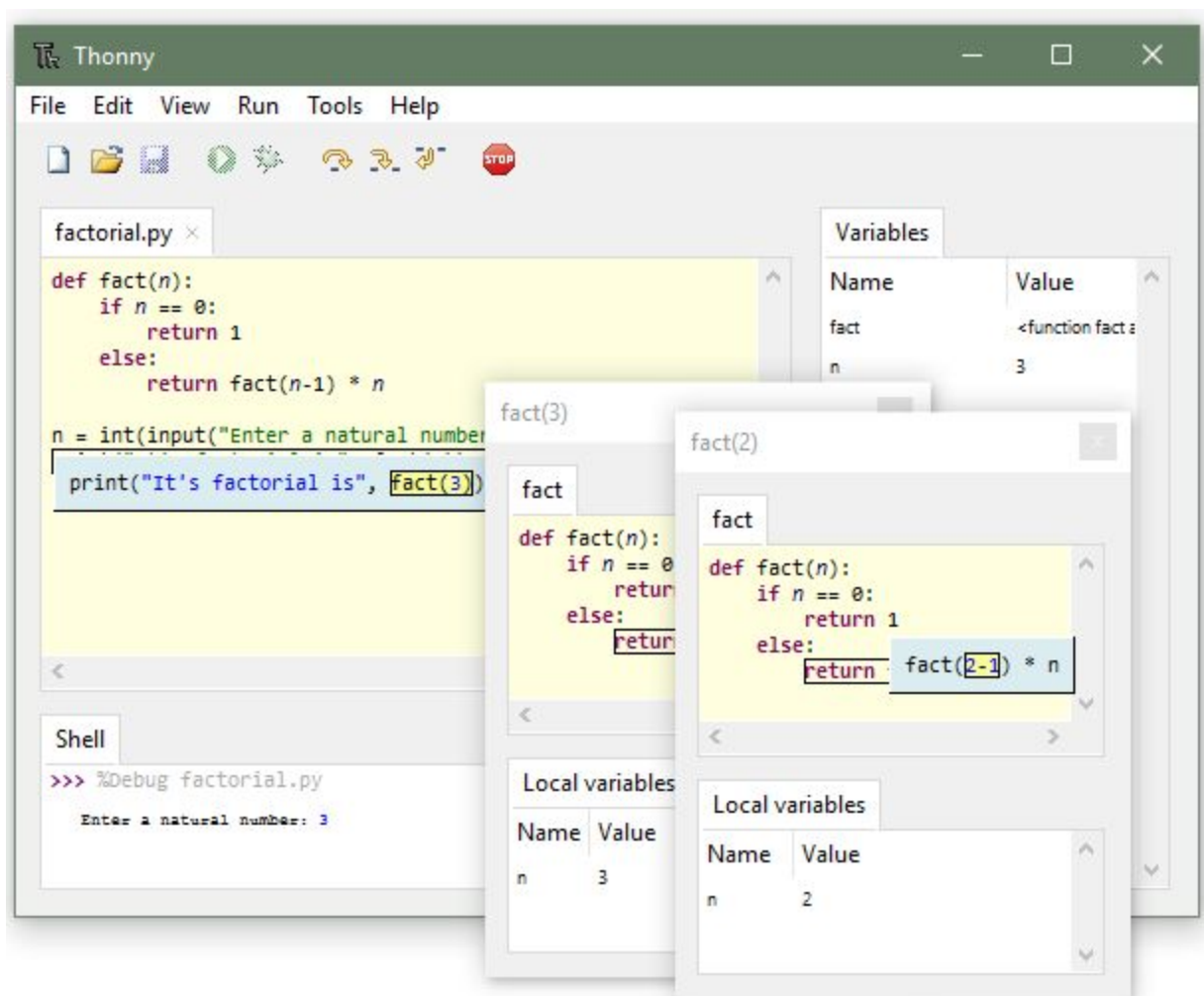
```
Python 3.5.2 Shell
File Edit Shell Debug Options Window Help
Python 3.5.2 (default, Nov 17 2016, 17:05:23)
[GCC 5.4.0 20160609] on linux
Type "copyright", "credits" or "license()" for more
information.
>>> |
```


5.2. THONNY

Thonny

One of interesting python IDE's for learning purposes is Thonny. It was developed by Aivar Annamaa who teaches introductory programming course in University of Tartu and he "seen that students, who don't have good understanding how their programs get executed, struggle the most with programming exercises." (Annamaa, 2016).

That's why he created Thonny. It is a Python IDE for learning programming. It can show step-by-step how Python executes your programs.



6. Virtual Python Environment builder

Virtual Python Environment builder - virtualenv - is a tool to create isolated Python environments.

The plan for first iteration was to display a simple Tkinter window when running ptpython and when F3 key is pressed to display history. The problem was the dependency of ptpython. When the code was changed and run , it was accessing the ptpython original which is installed in usr/lib/bin folder.

The virtualenv creates an environment that has its own installation directories, that doesn't share libraries with other virtualenv environments (and optionally does not access the globally installed libraries either). (Virtual Environments and Packages, 2016)

Venv advantages:

- Separation of packages installations - you can use different package sets for each project.
- Separation of python versions - you can use different python versions for each project. (Lipiński, 2016)

7. Mercurial - source control tool

“Mercurial is a free, distributed source control management tool. It efficiently handles projects of any size and offers an easy and intuitive interface.” (Mercurial SCM, 2016).

Mercurial was chosen to try out different source control version than Git. Mercurial website (Mercurial SCM, 2016) points that the advantages are:

- It is fast and powerful
- Mercurial efficiently handles projects of any size and kind.
- It is easy to learn
- You can follow our simple guide to learn how to revision your documents with Mercurial, or just use the quick start to get going instantly.

8. Summary and Conclusion

The research was done over one month period to research on lot of tools necessary to build portable GUI for ptython shell. First task was to research and get familiar with ptython shell. Ptython was installed on laptop with Linux OS. Lots of examples has been tried out using ptython shell to get familiar with it's cool features.

Mercurial was chosen as source version control. Similar applications has been researched - IDLE and Thonny.

The research was done also to find available GUI Toolkits in python and 8 frameworks got picked out - they were installed and simple example has been run to try them out. All these research findings has been summarised and compared to give an overview of available GUI frameworks in python and GUI Comparison Table has been created (see Appendix A).

Toga sounded like a very interesting and cool GUI framework to use because it offers native look and feel across platforms. Interesting features about Toga - it uses an approach that is new for widget toolkits, but well proven in computing: Cascading Style Sheets, (CSS) but because it is new tool (only 0.1.0 release) and they were no documentation when the research was done (documentation has only been added 24/3/17) Toga was not picked as the technology.

At the end of this research the decision was made that Tkinter GUI framework is going to be used to develop portable GUI for Ptython shell because it is already being included with the Python standard library, making it the most convenient and compatible toolkit to program with and because it offers native look and feel on all platforms.

Appendix A

Comparison of Python GUI framework toolkits

Reference

1. Annamaa, A. (2016). Thonny, Python IDE for beginners. [ONLINE] Available at: <http://thonny.cs.ut.ee/>. [Accessed 02 October 2016].
2. Bodnar, J. (2016). First programs in PySide. [ONLINE] Available at: <http://zetcode.com/gui/pysidetutorial/firstprograms/>. [Accessed 21 October 2016].
3. easygui Documentation 24.6. IDLE — Python 2.7.13 documentation. 2016. 24.6. IDLE — Python 2.7.13 documentation. [ONLINE] Available at: <https://docs.python.org/2/library/idle.html>. [Accessed 19 October 2016].
4. easygui dev team, (2017). easygui Documentation [ONLINE] Available at: <https://media.readthedocs.org/pdf/easygui/master/easygui.pdf> [Accessed 28 January 2017].
5. EasyGui Tutorial (2017). EasyGui Tutorial — easygui 0.97 (2014-12-20) documentation. [ONLINE] Available at: <http://easygui.sourceforge.net/tutorial.html#introduction>. [Accessed 01 October 2016].
6. Effbot .(2016)An Introduction to Tkinter (Work in Progress) [ONLINE] Available at: <http://effbot.org/tkinterbook>. [Accessed 04 October 2016].
7. GitHub. (2017). GitHub - jonathanslenders/ptpython: A better Python REPL. [ONLINE] Available at: <https://github.com/jonathanslenders/ptpython>. [Accessed 04 April 2017].
8. GuiProgramming - Python Wiki. [ONLINE] Available at: <https://wiki.python.org/moin/GuiProgramming>. [Accessed 29 October 2016].
9. Harrison, (2016). Python Programming Tutorials. [ONLINE] Available at: <https://pythonprogramming.net/kivy-language-tutorial/>. [Accessed 02 October 2016].
10. Keith-Magee, R.(2016) A slightly less toy example — Toga 0.2.5.dev1 documentation.[ONLINE]Available at: <https://toga.readthedocs.io/en/latest/tutorial/tutorial-1.html>. [Accessed 09 October 2016].

11. Keith-Magee, R.(2017) Toga Documentation. [ONLINE] Available at:<https://media.readthedocs.org/pdf/toga/latest/toga.pdf> [Accessed 30 March 2017].
12. Kivy, (2016). Kivy: Cross-platform Python Framework for NUI Development. [ONLINE] Available at: <https://kivy.org/>. [Accessed 28 October 2016].
13. Kluyver, T. (2016). So you want to write a desktop app in Python | Codel. [ONLINE] Available at: <http://takluyver.github.io/posts/so-you-want-to-write-a-desktop-app-in-python.html> . [Accessed 23 October 2016].
14. Mercurial SCM.(2016). Mercurial SCM. [ONLINE] Available at: <https://www.mercurial-scm.org/>. [Accessed 03 October 2016].
15. Meyers, C. (2016). Building a GUI Application with Tkinter [ONLINE] Available at: <http://www.openbookproject.net/py4fun/gui/tkPhone.html>. [Accessed 01 October 2016]
16. Michael Kennedy (@mkennedy). 2016. Episode #79 Beeware Python Tools - [Talk Python To Me Podcast]. [ONLINE] Available at: <https://talkpython.fm/episodes/show/79/beeware-python-tools>. [Accessed 07 October 2016].
17. PyQt5 statusbar – Python Tutorial, (2016). PyQt5 statusbar – Python Tutorial. [ONLINE] Available at: <https://pythonspot.com/en/pyqt5-statusbar/>. [Accessed 10 October 2016].
18. PyQt5 window, (2016). PyQt5 window – Python Tutorial. [ONLINE] Available at: <https://pythonspot.com/en/pyqt5-window/>. [Accessed 03 October 2016].
19. PySide, (2016). Overview — PySide 1.2.1 documentation. [ONLINE] Available at: <http://pyside.github.io/docs/pyside/>. [Accessed 23 October 2016].
20. Qt Wiki. (2016). About PySide - Qt Wiki. [ONLINE] Available at: https://wiki.qt.io/About_PySide. [Accessed 10 October 2016].
21. Rappin, N. (2006). *Wxpython in Action*. Edition. Manning Publications.
22. Reitz, K.(2016). GUI Applications — The Hitchhiker's Guide to Python. [ONLINE] Available at: <http://docs.python-guide.org/en/latest/scenarios/gui/>. [Accessed 24 October 2016].

23. Riverbank Computing Ltd , (2016). PyQt Class Reference. [ONLINE] Available at: <http://pyqt.sourceforge.net/Docs/PyQt4/classes.html>. [Accessed 04 October 2016].
24. Szymon Lipiński. 2016. Virtualenv Tutorial Part 2. [ONLINE] Available at: <http://www.simononsoftware.com/virtualenv-tutorial-part-2/>. [Accessed 21 October 2016].
25. Taylor, A. (2016). Kivy crash course - YouTube. [ONLINE] Available at: https://www.youtube.com/playlist?list=SPdNh1e1kmiPP4YApJm8ENK2yMlwF1_edq. [Accessed 01 October 2016].
26. TkInter - Python Wiki. (2016). TkInter - Python Wiki. [ONLINE] Available at: <https://wiki.python.org/moin/TkInter>. [Accessed 01 October 2016].
27. tutorialspoint.com. (2016.) wxPython Tutorial. [ONLINE] Available at: <https://www.tutorialspoint.com/wxpython/index.htm>. [Accessed 13 October 2016].
28. Tutorialspoint.com, (2016). PyQt QMessageBox. [ONLINE] Available at: https://www.tutorialspoint.com/pyqt/pyqt_qmessagebox.htm. [Accessed 03 October 2016].
29. Tutorialspoint.com, (2016). PyQt Introduction. [ONLINE] Available at: https://www.tutorialspoint.com/pyqt/pyqt_introduction.htm. [Accessed 01 October 2016].
30. tutorialspoint.com. (2016). Python GUI Programming (Tkinter). [ONLINE] Available at: http://www.tutorialspoint.com/python/python_gui_programming.htm. [Accessed 05 October 2016].
31. Widget Reference — Toga 0.2.5.dev1 documentation. (2017). Widget Reference — Toga 0.2.5.dev1 documentation. [ONLINE] Available at: <https://toga.readthedocs.io/en/latest/reference/index.html>. [Accessed 09 October 2016].
32. Wikipedia, (2016). Tkinter - Wikipedia. [ONLINE] Available at: <https://en.wikipedia.org/wiki/Tkinter>. [Accessed 11 October 2016].
33. 12. Virtual Environments and Packages — Python 3.6.1rc1 documentation. 2016. 12. Virtual Environments and Packages — Python 3.6.1rc1 documentation. [ONLINE] Available at: <https://docs.python.org/3/tutorial/venv.html>. [Accessed 11 October 2016].

34. wxPython Team, (2016). API Documentation — wxPython Phoenix 3.0.3 documentation. [ONLINE] Available at:
<https://wxpython.org/Phoenix/docs/html/main.html>. [Accessed 21 October 2016].
35. YouTube. (2016). Toga: yet another GUI toolkit on Python [ONLINE] Available at:
<https://www.youtube.com/watch?v=KlArzcl46Bo>. [Accessed 11 October 2016].
36. Zawadzki, A. (2016) easygui 0.98.0 [ONLINE] Available at:
<https://pypi.python.org/pypi/easygui/0.98.0> [Accessed 05 October 2016].