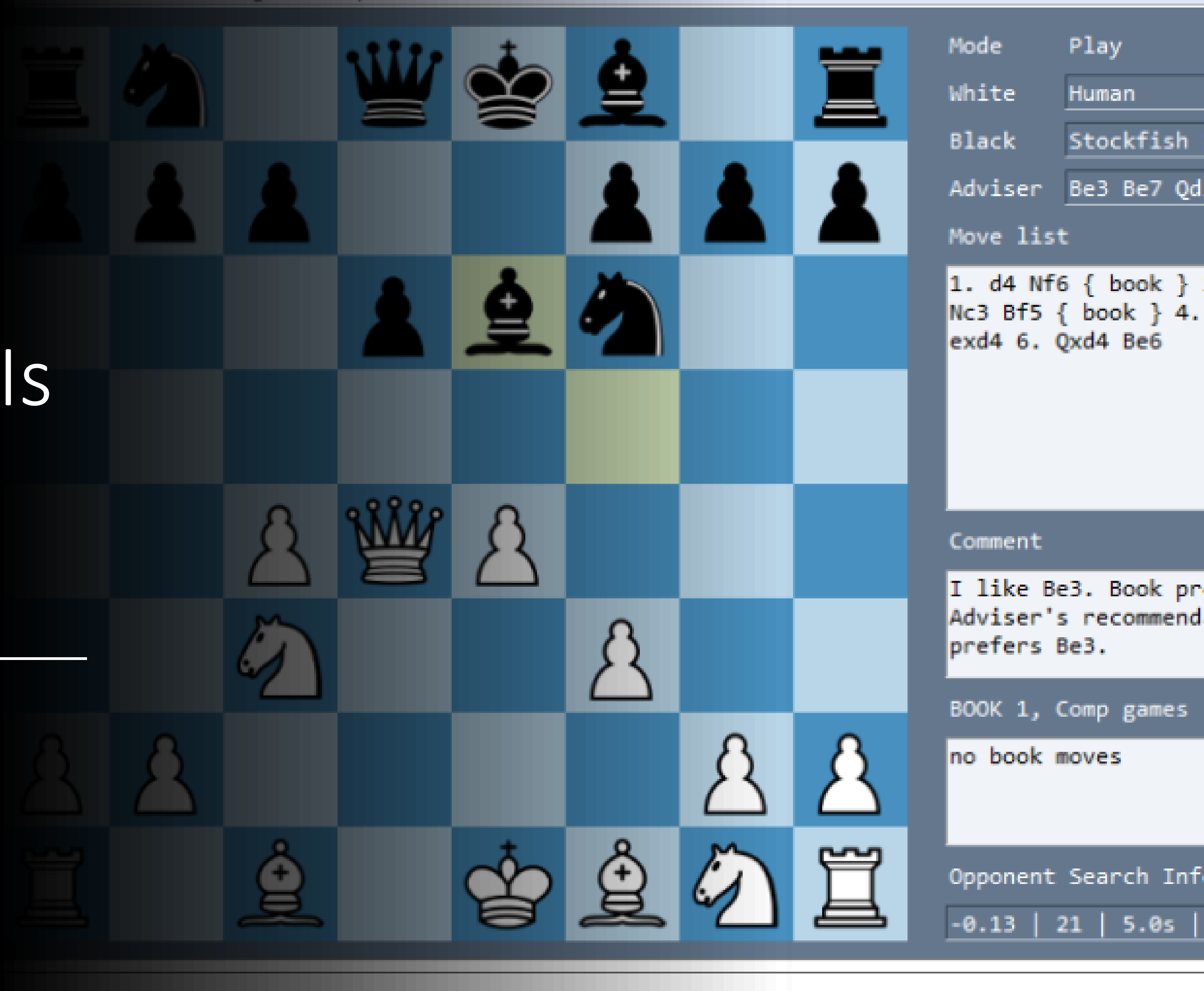
Attractive GUIs with PySimpleGUI

Ruud van der Ham, salabim.org



Mode Play

White Human

Black Stockfish

Adviser Be3 Be7 Qd

Move list

```
1. d4 Nf6 { book }
Nc3 Bf5 { book } 4.
exd4 6. Qxd4 Be6
```

Comment

```
I like Be3. Book pr
Adviser's recommend
prefers Be3.
```

BOOK 1, Comp games

```
no book moves
```

Opponent Search Inf

```
-0.13 | 21 | 5.0s |
```

My interest in PySimpleGUI

Quite a lot of experience in the **animation** part of tkinter, because of my salabim package

But, practically no experience with GUIs

I was in contact with the sole developer and maintainer of PySimpleGUI

And I had some projects that could use an attractive GUI.

So I started ...

And .. I am still a beginner in this field. So, please don’t shoot the messenger ...

Menu for today

Brief overview of GUIs in the Python world

Introduction PySimpleGUI

Two sample applications (hands on)

Gallery of PySimpleGUI application

Pros and cons

Conclusion

Introduction

GUIs are important

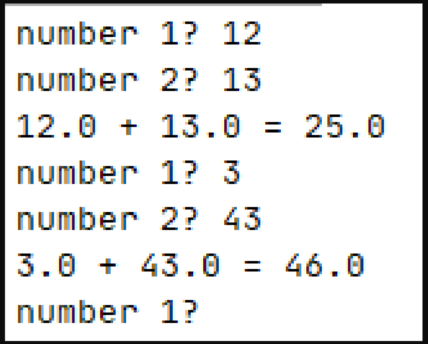
End users don’t want to /can’t use command line tools (especially under Windows)

Even for simple tasks an attractive GUI can be important

But ... GUIs are complicated and the domain of experts

Example of non GUI application

```
while True:
    number_1 = input("number 1? ")
    if number_1 == "":
        break
    number_1 = float(number_1)
    number_2 = float(input("number 2? "))
    print(f"{number_1} + {number_2} = {number_1 + number_2}")
```



```
number 1? 12
number 2? 13
12.0 + 13.0 = 25.0
number 1? 3
number 2? 43
3.0 + 43.0 = 46.0
number 1?
```

Example of a non GUI application: a CLI app

```
Usage: black [OPTIONS] [SRC]...

  The uncompromising code formatter.

Options:
  -l, --line-length INTEGER       How many characters per line to allow.
                                  [default: 88]

  -t, --target-version [py27|py33|py34|py35|py36|py37|py38]
                                  Python versions that should be supported by
                                  Black's output. [default: per-file auto-
                                  detection]

  --py36                          Allow using Python 3.6-only syntax on all
                                  input files.  This will put trailing commas
                                  in function signatures and calls also after
                                  *args and **kwargs. Deprecated; use
                                  --target-version instead. [default: per-file
                                  auto-detection]

  --pyi                           Format all input files like typing stubs
                                  regardless of file extension (useful when
                                  piping source on standard input).

  -S, --skip-string-normalization
                                  Don't normalize string quotes or prefixes.
  --check                         Don't write the files back, just return the
                                  status.  Return code 0 means nothing would
                                  change.  Return code 1 means some files
                                  would be reformatted.  Return code 123 means
                                  there was an internal error.

  --diff                          Don't write the files back, just output a
                                  diff for each file on stdout.

  --fast / --safe                 If --fast given, skip temporary sanity
                                  checks. [default: --safe]

  --include TEXT                  A regular expression that matches files and
                                  directories that should be included on
                                  recursive searches.  An empty value means
                                  all files are included regardless of the
                                  name.  Use forward slashes for directories
                                  on all platforms (Windows, too).  Exclusions
                                  are calculated first, inclusions later.
                                  [default: \.pyi?$]

  --exclude TEXT                  A regular expression that matches files and
                                  directories that should be excluded on
                                  recursive searches.  An empty value means no
                                  paths are excluded. Use forward slashes for
```

GUIs: tkinter

Comes with most Python installations

Feature-rich

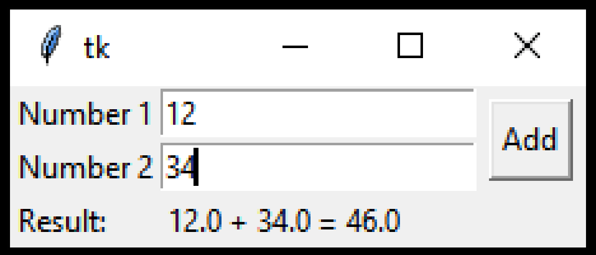
Flexible

Learning curve

Not very intuitive API

Example of a tkinter application

```
import tkinter as tk

def add_numbers():
    number_1 = float(entry_1.get())
    number_2 = float(entry_2.get())
    res = f"{number_1} + {number_2} = {number_1 + number_2}"
    result.set(res)

master = tk.Tk()
result = tk.StringVar()

tk.Label(master, text="Number 1").grid(row=0)
tk.Label(master, text="Number 2").grid(row=1)
tk.Label(master, text="Result:").grid(row=3)
tk.Label(master, text="", textvariable=result).grid(row=3, column=1)

entry_1 = tk.Entry(master)
entry_2 = tk.Entry(master)

entry_1.grid(row=0, column=1)
entry_2.grid(row=1, column=1)

b = tk.Button(master, text="Add", command=add_numbers)
b.grid(row=0, column=2, columnspan=2, rowspan=2, padx=5, pady=5)

tk.mainloop()
```



```
Number 1  12
Number 2  34
Result:   12.0 + 34.0 = 46.0
```

Other GUI frameworks

- wxPython The GUI Toolkit for Python
- PyQt / Qt for Python / PySide2
- kivy
- …

Powerful, learning curve, sometimes licensing costs

GUI builders

- PAGE
- Pygubu

Not very popular in the Python world

Interfacing rather complex

Maintenance of both GUI part and application

PySimpleGUI

Relatively new (start project 2 years ago)

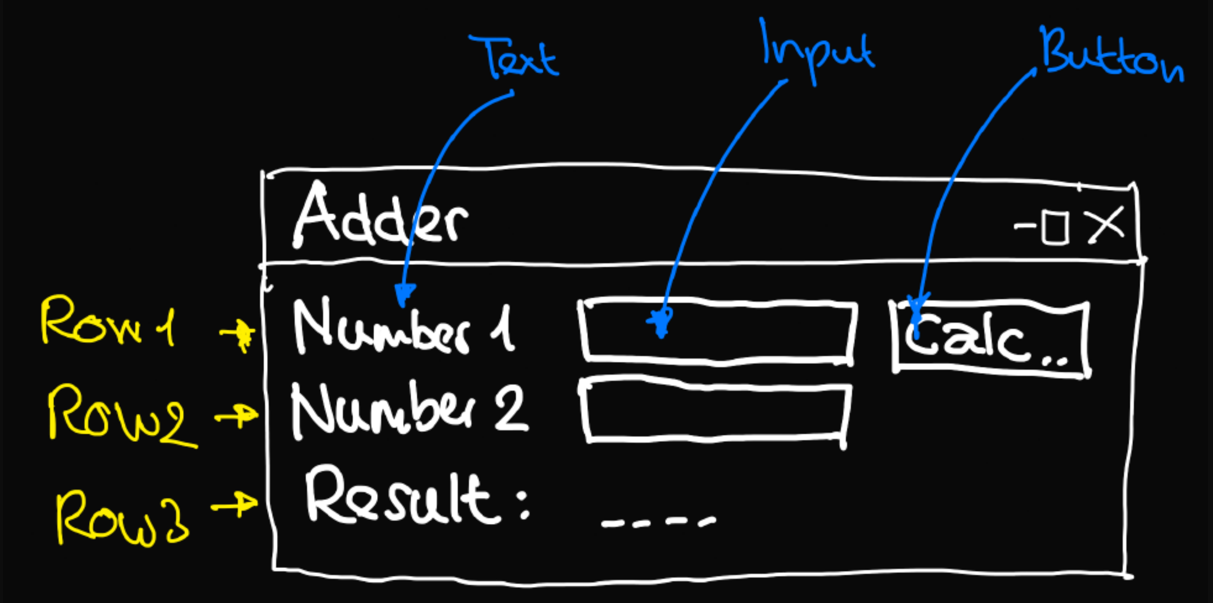
Very actively maintained (upto now)

Open source

Four platforms supported:

- tkinter
- PyQt
- wxPython
- web (Remi)

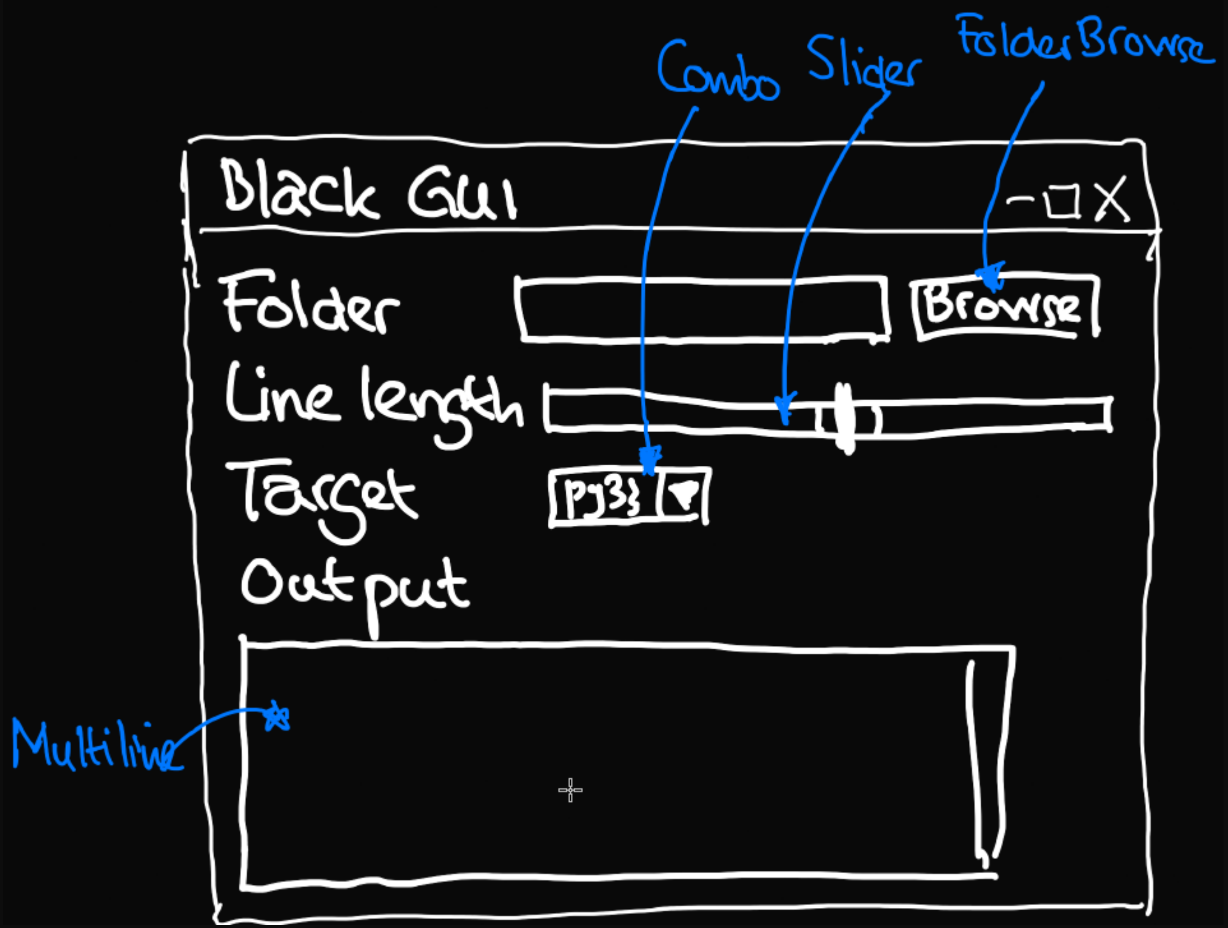
Let's build the adder application in the tkinter version of PySimpleGUI



PySimpleGUI

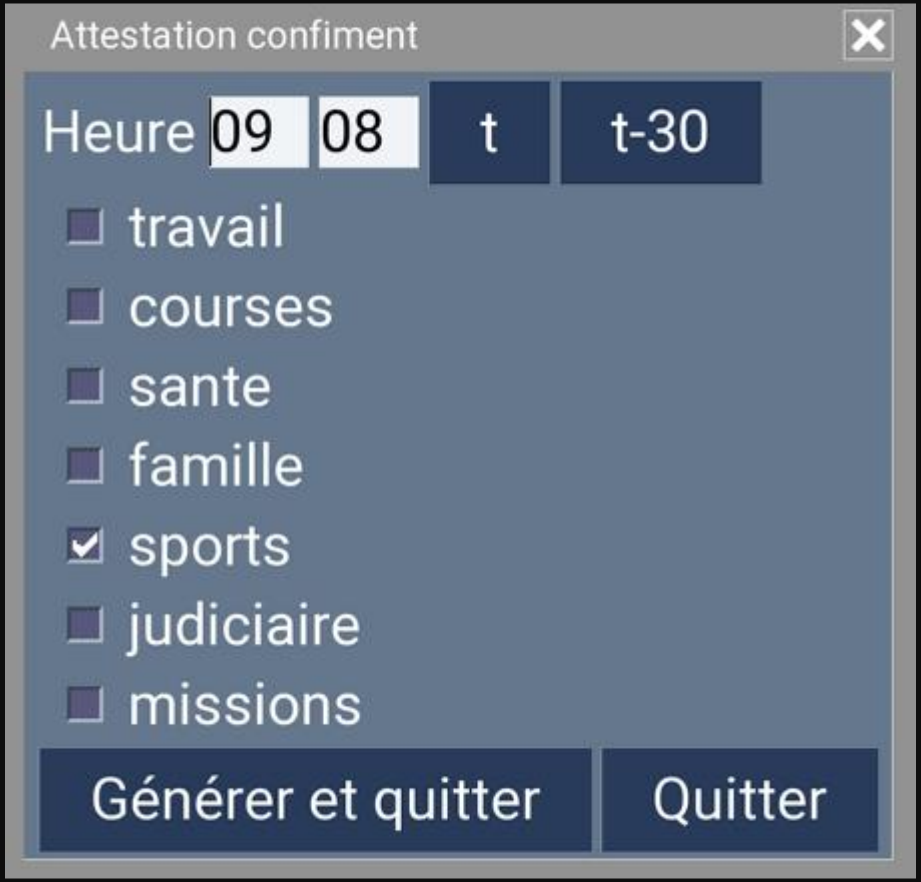
And now for another sample project:

A GUI for the Black formatter to replace the CLI



PySimpleGUI

We can even use PySimpleGUI run on Android phones:


Attestation confiment

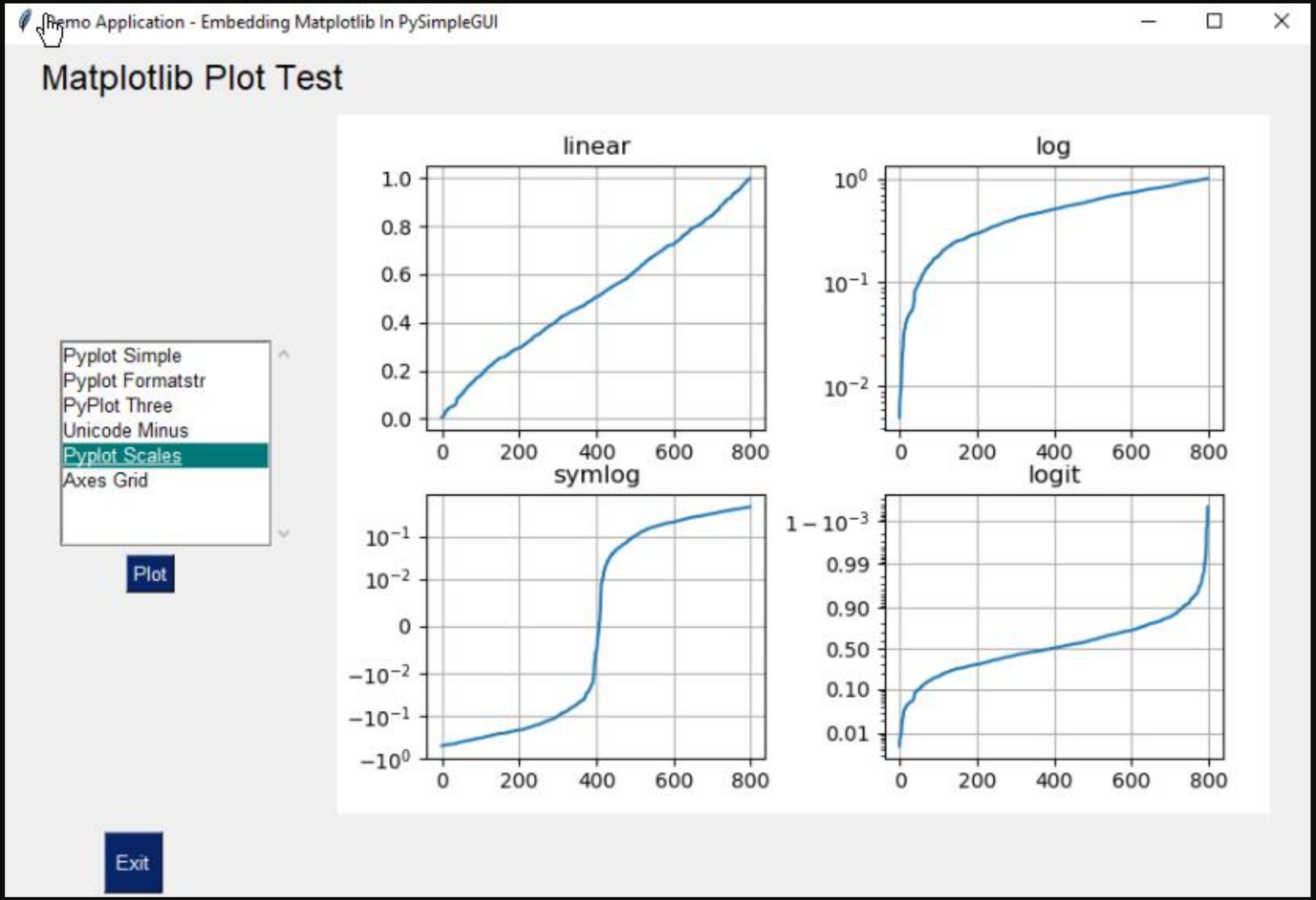
Heure 09 08 t t-30

- [ ] travail
- [ ] courses
- [ ] sante
- [ ] famille
- [x] sports
- [ ] judiciaire
- [ ] missions

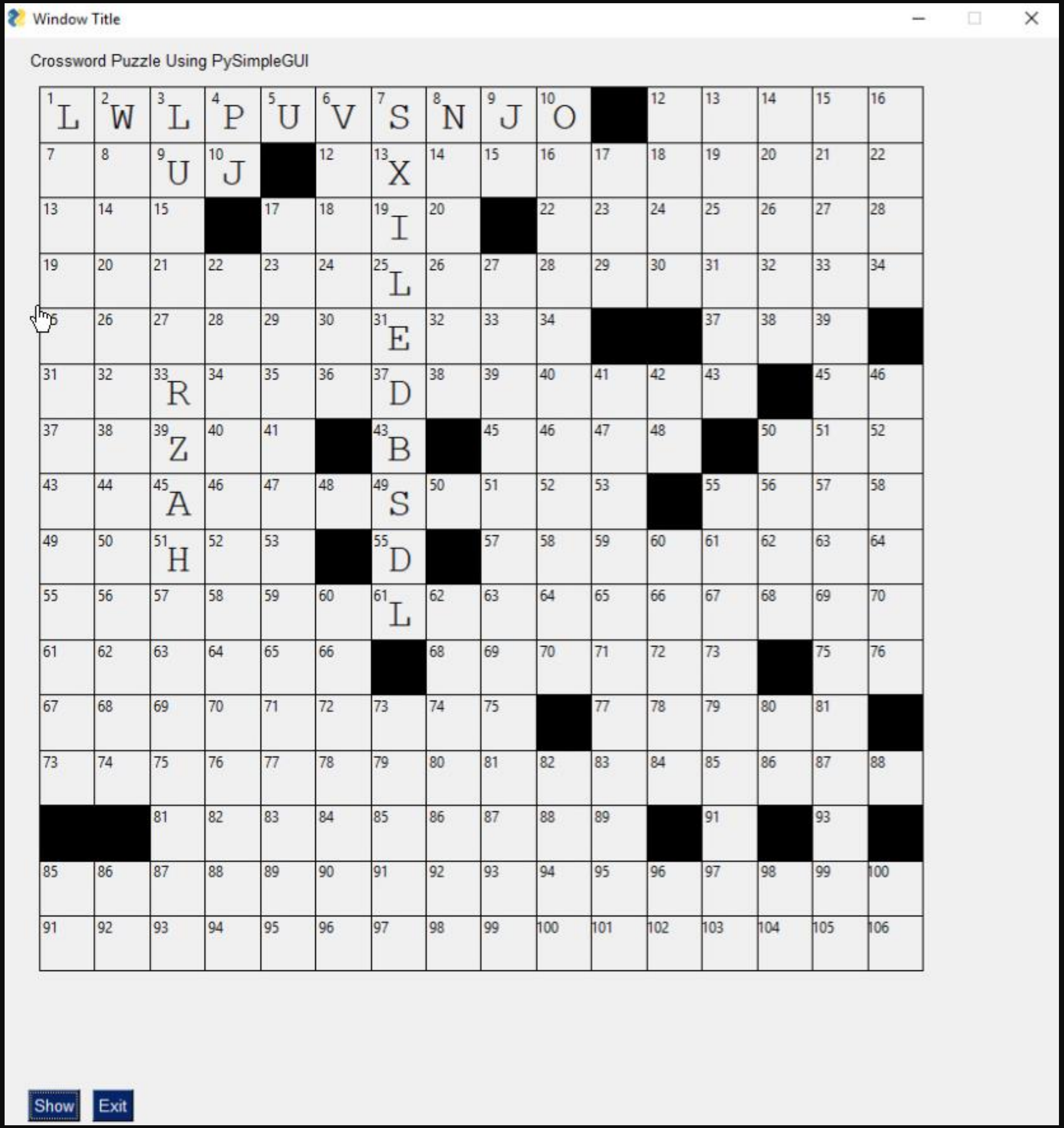
PySimpleGUI

Sample demo applications from the site:

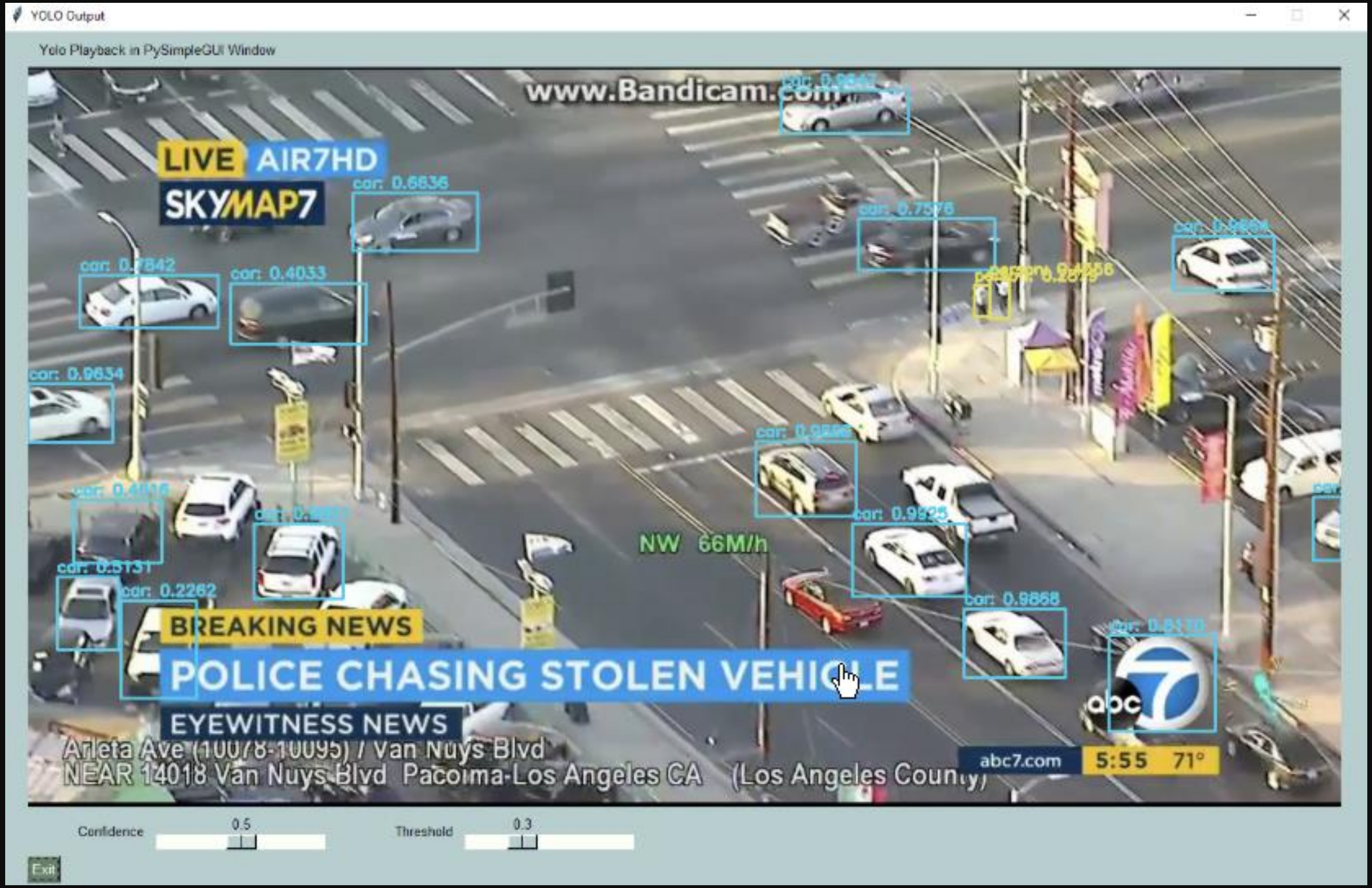
PySimpleGUI



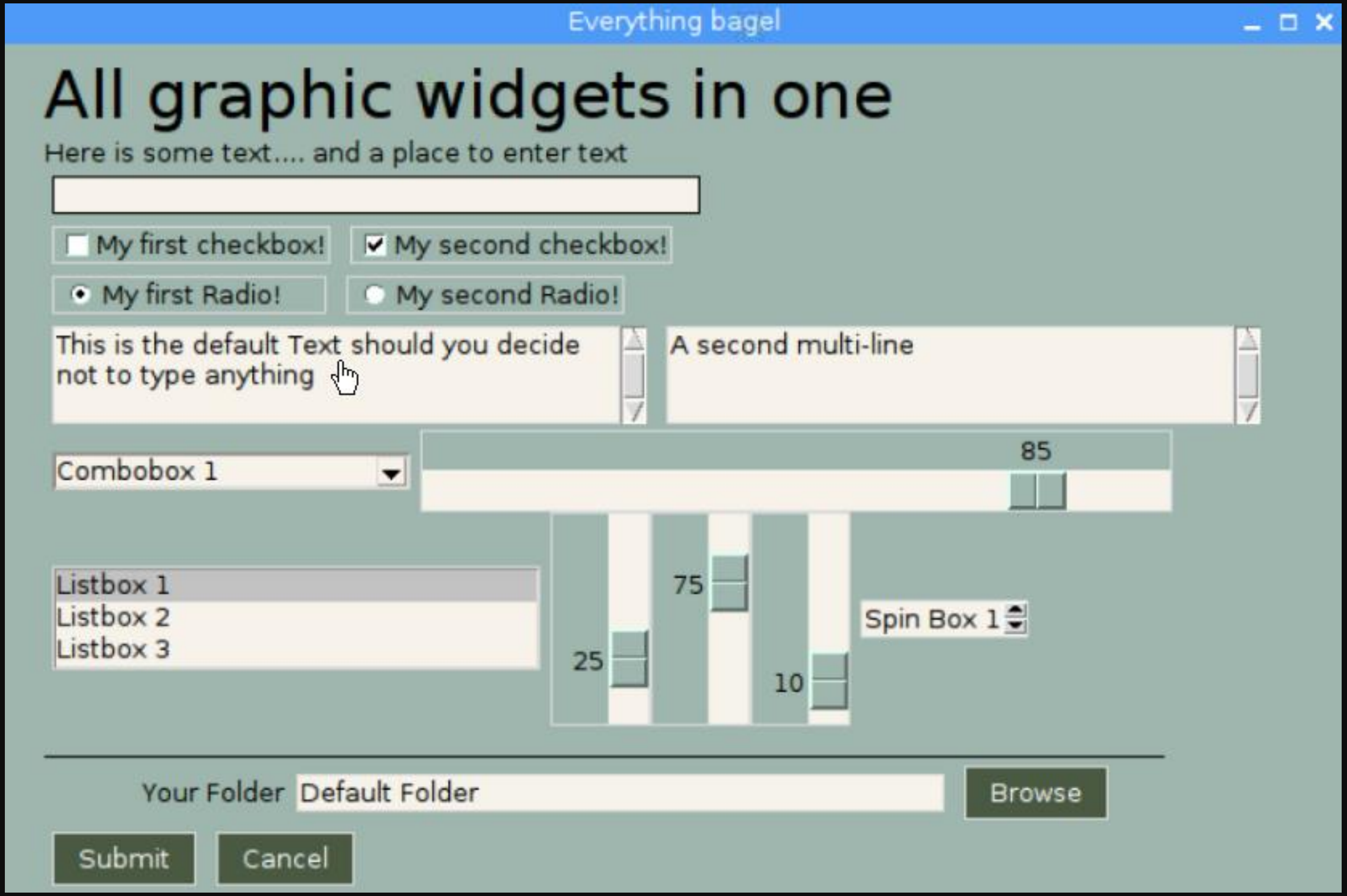
PySimpleGUI



PySimpleGUI



PySimpleGUI


Everything bagel

All graphic widgets in one

Here is some text.... and a place to enter text

This is the default Text should you decide not to type anything

A second multi-line

Your Folder Default Folder

What else?

I think I just know about 10% of the functionalitry of PySimpleGUI ...

It has so many widgets and features

Interfaces with matplotlib, OpenCV, PIL, ...

Supports threading

Cross platform (mostly)

Other ports, including Web/Remi

Very extensive and up-to-date documentation and cookbook

Great for beginners, but also experts that don’t want to spend too much time on the nitty gritty of native GUI packages

But,

You are limited to a number of prepackaged solutions, albeit many. Customization has its limits, by nature.

You can’t fully control everything.

No port for iPad/Pythonista (my favourite)

You have to rely on a package developed/maintained by one (devoted) person.

Conclusion

PySimpleGUI is worth more than a try.

Goto www.pysimplegui.com to find out for yourself.

Thank you

for your attention!

ruud@salabim.org