Python programming — GUI

Finn Årup Nielsen

Department of Informatics and Mathematical Modelling
Technical University of Denmark
October 1, 2012

Graphical user interface

Tk

Python module `Tkinter`. Quick and dirty. (Langtangen, 2005, Chapters 6 & 11)

wxWidgets (wxWindows)

`wxPython` (Ubuntu package `python-wxtools`) and `PythonCard` (Ubuntu meta-package `pythoncard`)

Others

PyGTK for Gtk, PyQT and PySide for Qt

Tkinter Hello world

```
from Tkinter import *

def hello():
  """Callback function for button press."""
  print "Hello World"

# Main root window
root = Tk()

# A button with a callback function
Button(root, text="Press here", command=hello).pack()
# .pack() method is necessary to make the button appear

# Event loop
root.mainloop()
```

Tkinter Hello world

```
Python 2.6.5 (r265:79063, Apr 16 2010, 13:09:56)
[GCC 4.4.3] on linux2
Type "help", "copyright", "credits" or "license" for more inform
>>> from Tkinter import *
>>>
>>> def hello():
...     """Callback function for button press."""
...     print "Hello World"
...
>>> root = Tk()
>>> Button(root, text="Press here", command=hello).pack()
>>> root.mainloop()
Hello World
Hello World
```

Tkinter: menu, label

Build a GUI for showing one tweet stored in a Mongo database. One tweets at a time.

```
from Tkinter import *
import pymongo, re

pattern = re.compile(r'\bRT\b', re.UNICODE)
```

Should read from the Mongo database, and display some of the fields in a graphical user interface.

With a button the user should be able to click to the next item.

A callback function

```
counter = 0
def next(event=None):
  """Function that can be used as a callback function from a button."""
  global counter                # Global variable not so pretty
  counter += 1
  tweet = tweets.find_one(skip=counter)        # pymongo function
  while tweet.has_key('delete') or not pattern.search(tweet["text"]):
    counter += 1
    tweet = tweets.find_one(skip=counter)
  # Set gui texts
  gui_user["text"] = tweet["user"]["name"]
  gui_language["text"] = tweet["user"]["lang"]
  gui_followers["text"] = tweet["user"]["followers_count"]
  gui_text["text"] = tweet["text"]
```

Initial setup of window

```
root = Tk()

# Title of the window
root.title('Twitter in Mongo GUI')

# key associated with the 'next' callback function
root.bind('n', next)

# Build menubar
menubar = Menu(root)
filemenu = Menu(menubar, tearoff=0)
filemenu.add_command(label="Exit", command=root.destroy)
menubar.add_cascade(label="File", menu=filemenu)

# Get Mongo collection of tweets
tweets = pymongo.Connection().twitter.tweets
```

Set up of window components

```
# Using a 'Frame' to contain two 'Label's in a row
frame = Frame(root)
Label(frame, text="User:", width=10).pack(side=LEFT)
gui_user = Label(frame)
gui_user.pack(side=LEFT)
frame.pack(fill=X)

frame = Frame(root)
Label(frame, text="Language:", width=10).pack(side=LEFT)
gui_language = Label(frame)
gui_language.pack(side=LEFT)
frame.pack(fill=X)

frame = Frame(root)
Label(frame, text="Followers:", width=10).pack(side=LEFT)
```

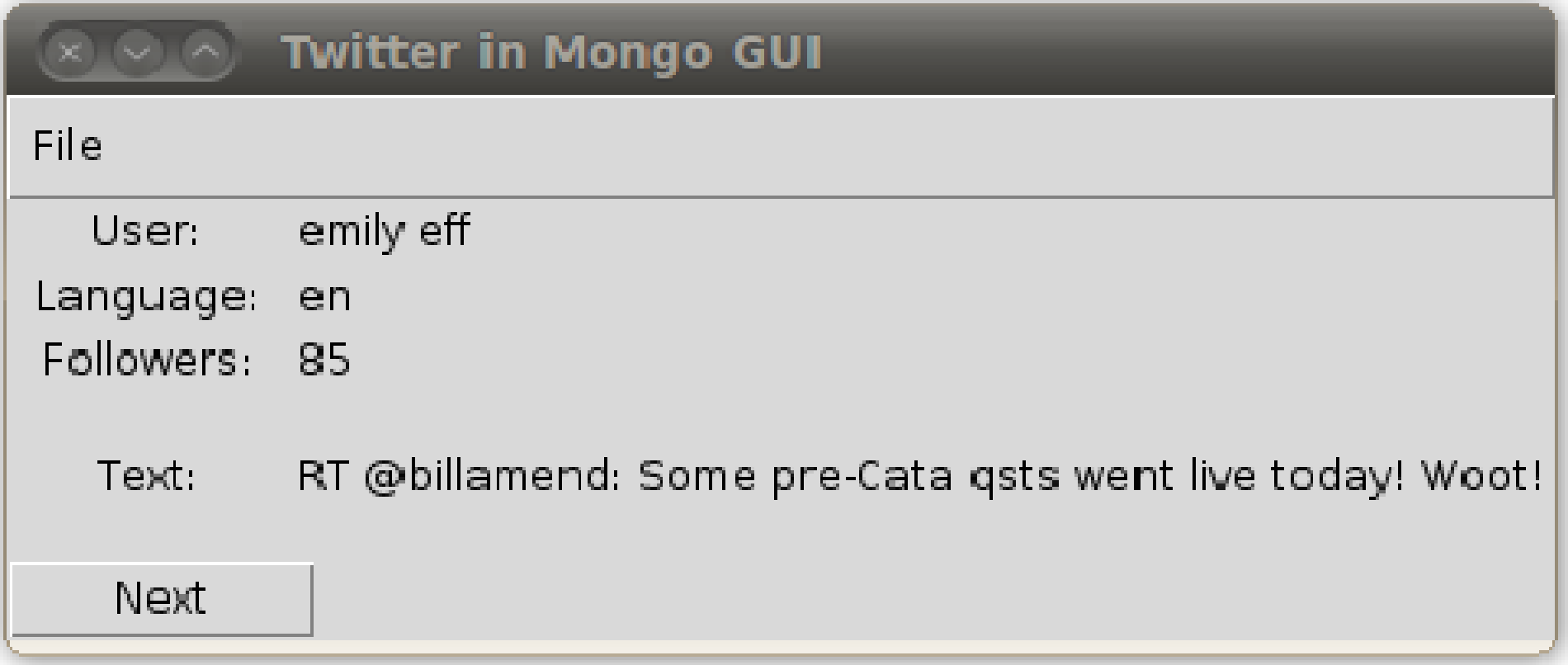
```
gui_followers = Label(frame)
gui_followers.pack(side=LEFT)
frame.pack(fill=X)

frame = Frame(root)
Label(frame, text="Text:", width=10).pack(side=LEFT)
gui_text = Label(frame)
gui_text.pack(side=LEFT)
frame.pack(fill=X)

# 'Next' button
frame = Frame(root)
Button(frame, text="Next", width=8, command=next).pack(side=LEFT)
frame.pack(fill=X)

root.config(menu=menubar)
root.mainloop()
```

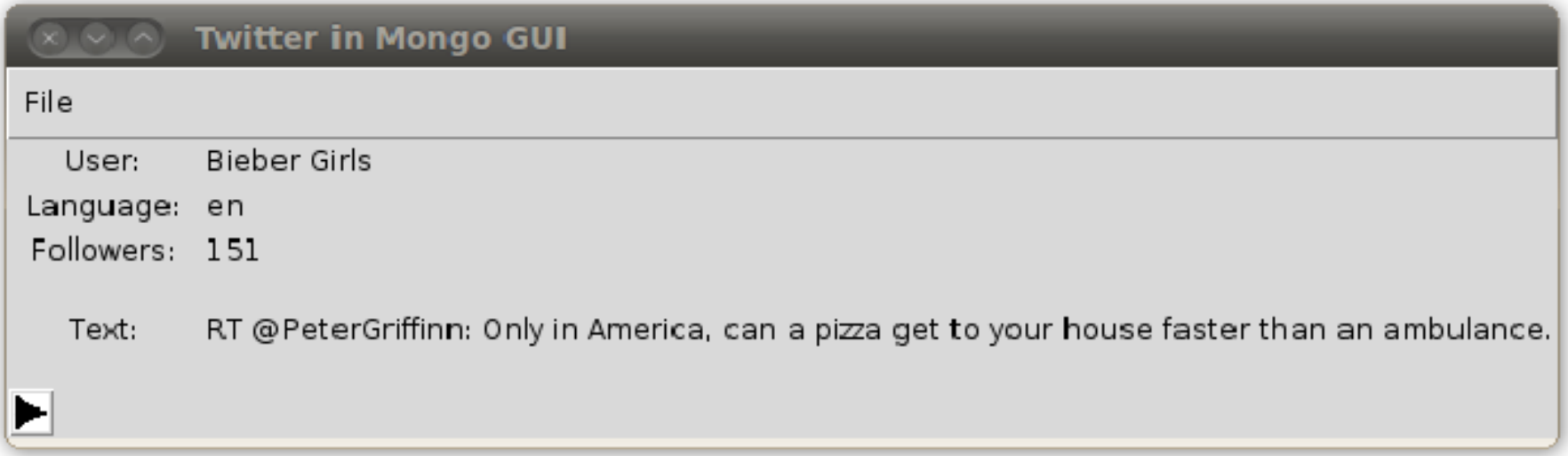
Rendered window



New ‘next’ button

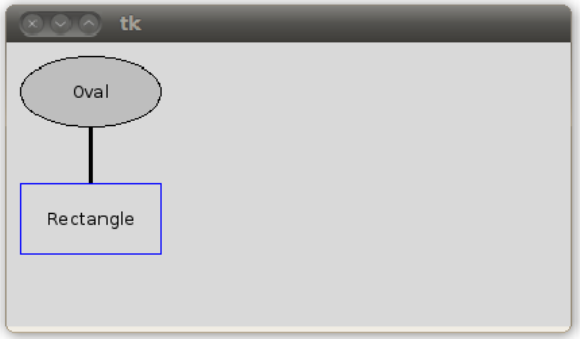
Brent Burley (Martelli et al., 2005, pages 432–434)

```
import base64
# Load a GIF file with an appropriate icon image
icon = base64.encodestring(open('Billeder/right.gif').read())
# Once you have read the GIF you can put the string in the code
iconImage = PhotoImage(master=root, data=icon)
Button(frame, image=iconImage, command=next).pack(side=LEFT)
```



Tkinter canvas

Drawing in the Tkinter interface with `canvas`.



Example from (Langtangen, 2005, pages 527+)

```
from Tkinter import *
root = Tk()
canvas = Canvas(root, width=400, height=200)
canvas.pack()
canvas.create_oval(10, 10, 110, 60, fill="grey")
canvas.create_text(60, 35, text="Oval")
canvas.create_rectangle(10, 100, 110, 150, outline="blue")
canvas.create_text(60, 125, text="Rectangle")
canvas.create_line(60, 60, 60, 100, width=3)
```

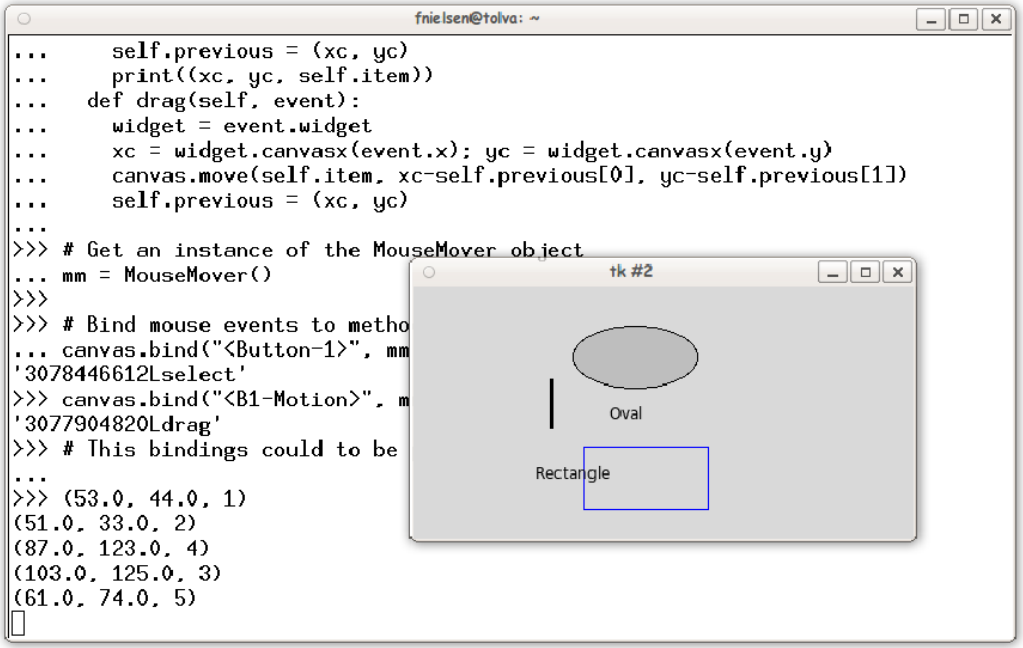
Mouse callback

```
class MouseMover():
  def __init__(self):
    self.item = 0; self.previous = (0, 0)
  def select(self, event):
    widget = event.widget                          # Get handle to canvas
    # Convert screen coordinates to canvas coordinates
    xc = widget.canvasx(event.x); yc = widget.canvasx(event.y)
    self.item = widget.find_closest(xc, yc)[0]          # ID for closest
    self.previous = (xc, yc)
    print((xc, yc, self.item))
  def drag(self, event):
    widget = event.widget
    xc = widget.canvasx(event.x); yc = widget.canvasx(event.y)
    canvas.move(self.item, xc-self.previous[0], yc-self.previous[1])
    self.previous = (xc, yc)
```

Canvas with mouse callback

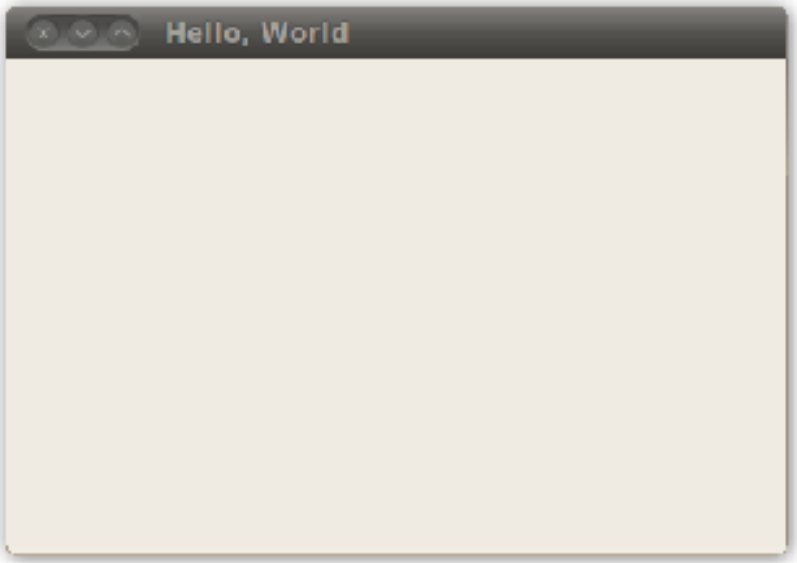
```
# Get an instance of the MouseMover object
mm = MouseMover()

# Bind mouse events to methods (could also be in the constructor)
canvas.bind("<Button-1>", mm.select)
canvas.bind("<B1-Motion>", mm.drag)
```



wxPython

http://www.wxpython.org



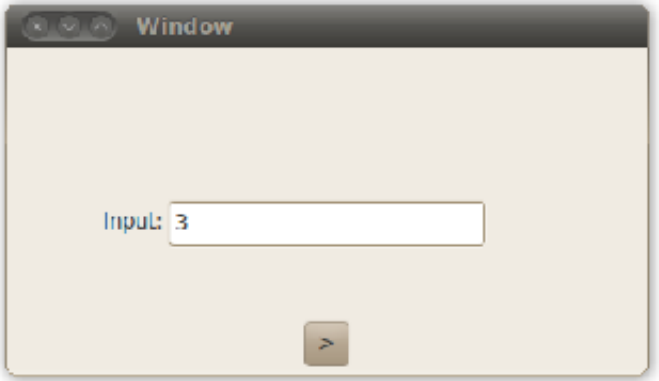

```
import wx

# Must always be set
app = wx.App()

# A window
frame = wx.Frame(None, title="Hello, World")
frame.Show()

# Event loop
app.MainLoop()
```

wx: event, get, set




```
import wx
app = wx.App()
class Incrementor(wx.Frame):
  def __init__(self, parent, id, title="Window"):
    wx.Frame.__init__(self, parent, id, title, size=(400, 200))
    wx.StaticText(self, -1, 'Input:', (60, 100))
    self.edit = wx.TextCtrl(self, value="0", pos=(100, 95), size=(200, 30))
    forward_button = wx.Button(self, -1, '>', pos=(185, 170), size=(30, 30)
    self.Bind(wx.EVT_BUTTON, self.forward, id=forward_button.GetId())
    self.Centre()              # Center in the middle of the screen
    self.Show(True)
  def forward(self, event):
    try: self.edit.SetValue(str(int(self.edit.GetValue())+1))
    except: pass

Incrementor(None, -1); app.MainLoop()
```

Python GTK

Gwibber: social network client written in Python and using the GTK module.

What to choose?

Tkinter. "De-facto standard GUI". May be included in standard install. Probably good for simple applications.

```
$ less /usr/lib/pymodules/python2.7/nltk/downloader.py
```

PyGTK (LGPL). Used in a range of programs on the Linux system. Gwibber, jokosher, update-manager, . . . : `$ less ‘which update-manager‘`

PyQT (GPL) for QT.

PySide (LGPL) for QT. Sponsored by Nokia.

wxPython

References

Langtangen, H. P. (2005). *Python Scripting for Computational Science*, volume 3 of *Texts in Computational Science and Engineering*. Springer. ISBN 3540294155.

Martelli, A., Ravenscroft, A. M., and Ascher, D., editors (2005). *Python Cookbook*. O'Reilly, Sebastopol, California, 2nd edition.