KORISNIČKA SUČELJA

Aleksandar Maksimović
IRB

uvod u wxPython

- wxPython
  - OS
    - MS Windows (Windows 98 i novije)
    - Unix, Linux + gtk (Gnome Toolkit)
    - Mac OS X 10.2.3
  - Python ver 2.3
  - wxPython toolkit - više verzija
  - tekst editor

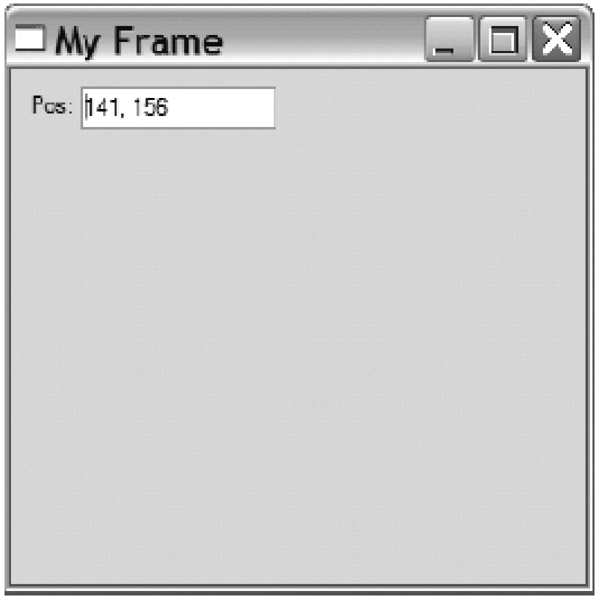
uvod wxpython

sample.py

vjezbe7

```
#!/bin/env python
import wx
class MyFrame(wx.Frame):

    def __init__(self):
        wx.Frame.__init__(self, None, -1, "My Frame", size=(300, 300))
        panel = wx.Panel(self, -1)
        panel.Bind(wx.EVT_MOTION,  self.OnMove)
        wx.StaticText(panel, -1, "Pos:", pos=(10, 12))
        self.posCtrl = wx.TextCtrl(panel, -1, "", pos=(40, 10))

    def OnMove(self, event):
        pos = event.GetPosition()
        self.posCtrl.SetValue("%s, %s" % (pos.x, pos.y))

if __name__ == '__main__':
    app = wx.PySimpleApp()
    frame = MyFrame()
    frame.Show(True)
    app.MainLoop()
```

uvod wxpython



sample.py

Label - StaticText

Entry - TextCtrl

Tkinter $\longrightarrow$ wxPython

wx.Frame.__init__ - wx konstruktor

wx.Panel

wx.EVT_MOTION - događaj

uvod

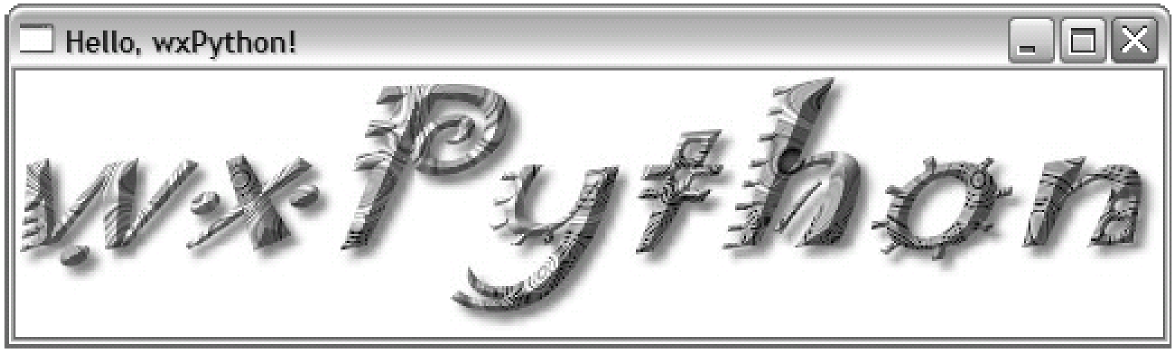


Figure 1.2
Running `hello.py`
on Windows

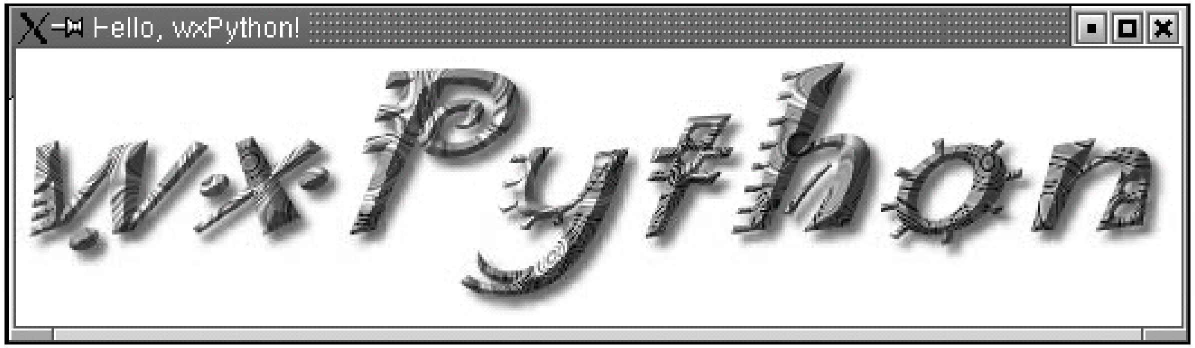


Figure 1.3
Running `hello.py`
on Linux

minimalni wxpy program

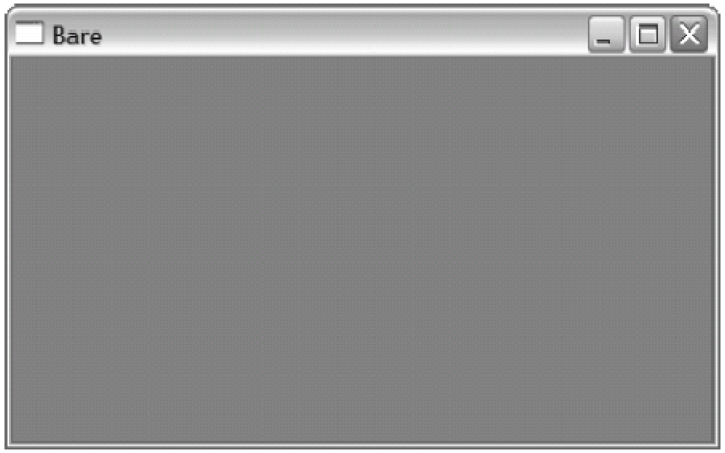
bare.py

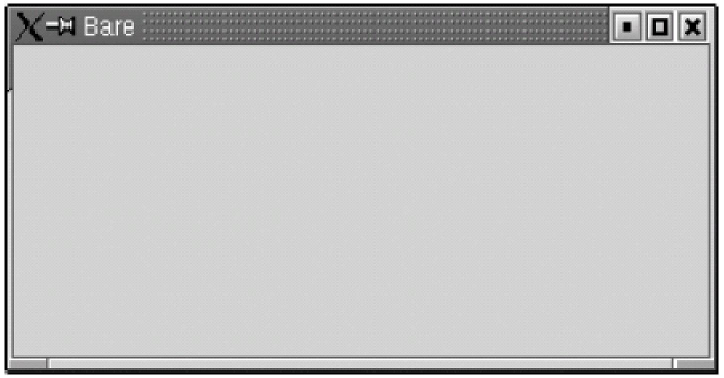
```
import wx

class App(wx.App):

    def OnInit(self):
        frame = wx.Frame(parent=None, title='Bare')
        frame.Show()
        return True

app = App()
app.MainLoop()
```

Provjerimo da li wxpython radi.

Program napravi prazan frame (okvir) i prikaže ga.

minimalni wxpy program





Sve linije koda u primjeru su neophodne.

Ilustrira 5 koraka potrebnih za svaki wxPython program:

1. Import wxPython paket
2. Naslijedi wx.App klasu (subklasa)
3. Definiraj konstruktor (__init__)
4. Kreiraj instancu klase (App())
5. Napravi glavnu petlju (Mainloop())

1. uvod

1. Importiranje wxPythona (modul wx)

```
import wx
```

2. Koristimo wx klase, funkcije ili konstante stavljanjem imena wx kao prefiks

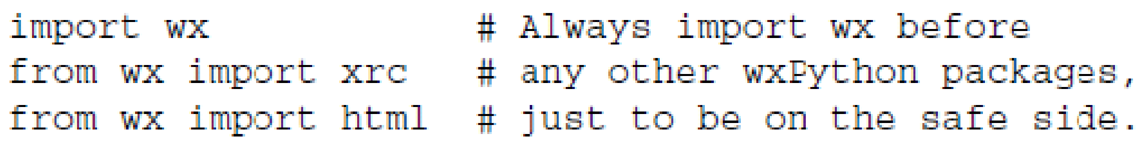
```
class App(wx.App):
```

3. wx moramo importirati prije svih ostalih klasa, funkcija iz wxPythona

1. uvod

- Stari stil (NE KORISTIMO)
  - from wxPython import wx # DEPRECATED
  - from wxPython.wx import * # DON'T DO THIS ANY MORE

Ako ne importiramo prvo wx neke klase neće dobro raditi, npr. xrc



```
import wx                   # Always import wx before
from wx import xrc          # any other wxPython packages,
from wx import html         # just to be on the safe side.
```

1. uvod

- Ostale pakete i dalje importiramo kako želimo

```
import sys
import wx
import os
from wx import xrc
import urllib
```

2. uvod

- wxPython program mora imati
  - 1 objekt aplikacije (wx.App), mora biti instanca wx.App ili subklasa (nasljedi wx.App) koja definira metodu OnInit(). Metodu OnInit() koristi wx.App prilikom kreiranja objekta.
  - 1 ili više frame objekata wx.Frame

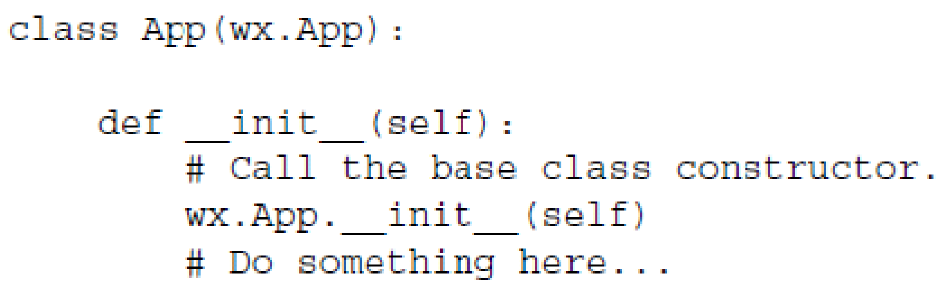
Subklasa:

```
class MyApp(wx.App):

    def OnInit(self):
        frame = wx.Frame(parent=None, id=-1, title="Bare")
        frame.Show()
        return True
```

Show - prikazuje ili skriva Frame (prozor)

3. Konstruktor

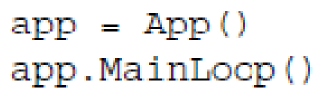
- Nismo definirali konstruktor
  - kad __init__ metoda nije definirana Python automatski zove konstruktor od klase iznad (roditelja) wx.App.__init__()
  - ako definiramo konstruktor, moramo zvati konstruktor od wx.App klase



```
class App(wx.App):

    def __init__(self):
        # Call the base class constructor.
        wx.App.__init__(self)
        # Do something here...
```

4 i 5 Aplikacija i petlja

- Konačni korak je stvaranje instance aplikacije i pozivanje metode MainLoop()



```
app = App()
app.MainLoop()
```

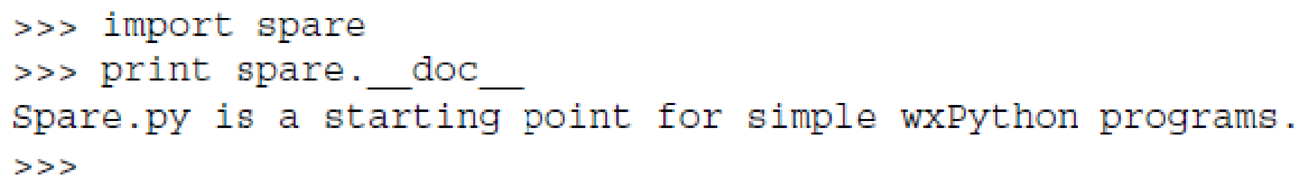
wxPython preuzima kontrolu i odgovara na događaje.

spare.py

```
#!/usr/bin/env python   ❶

"""Spare.py is a starting point for a wxPython prcgram."""   ❷

import wx

class Frame(wx.Frame):   ❸
    pass

class App(wx.App):

    def OnInit(self):
        self.frame = Frame(parent=None, title='Spare')   ❹
        self.frame.Show()
        self.SetTopWindow(self.frame)   ❺
        return True

if __name__ == '__main__':   ❻
    app = App()
    app.MainLoop()
```

uvod

1. unix OS poziva interpreter, inače je komentar
2. docstring - opisuje program



```
>>> import spare
>>> print spare.__doc__
Spare.py is a starting point for simple wxPython programs.
>>>
```

3. promjenili smo stvaranje Frame objekta, sada je Frame subklasa od wx.Frame klase
4. varijabla .frame sadrži instancu Frame

uvod

5. SetTopWindow() metoda postavlja self.frame kao "glavni" prozor. Metoda naslijeđena iz wx.App klase.
6. Ako je modul "glavni" program izvrši linije

```
if __name__ == '__main__':
    app = App()
    app.MainLoop()
```

hello.py

```
#!/usr/bin/env python      ❶ Shebang

"""Hello, wxPython! program."""    ◁— Docstring describes the code

import wx    ◁— Import the  wxPackage
                                ❷ wx.Frame subclass
class Frame(wx.Frame):      ◁—┘
    """Frame class that displays an image."""

    def __init__(self, image, parent=None, id=-1,       ❸ Image parameter
                 pos=wx.DefaultPosition,
                 title='Hello, wxPython!'):
        """Create a Frame instance and display image."""
        temp = image.ConvertToBitmap()                          ❹
        size = temp.GetWidth(), temp.GetHeight()
        wx.Frame.__init__(self, parent, id, title, pos, size)   Displaying
        self.bmp = wx.StaticBitmap(parent=self, bitmap=temp)    the image
```

hello.py

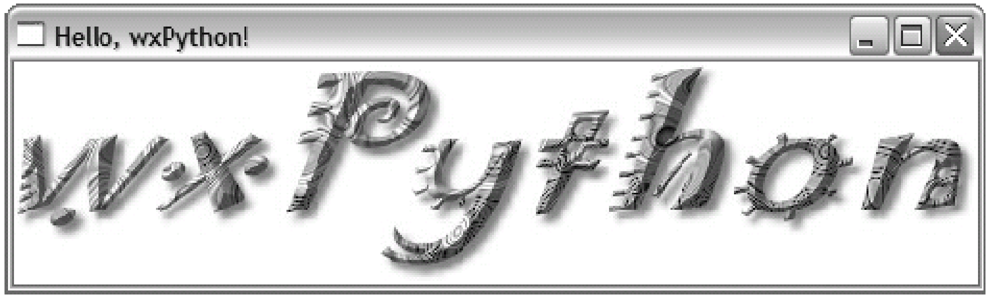
```
class App(wx.App):
    """Application class."""

    def OnInit(self):
        image = wx.Image('wxPython.jpg', wx.BITMAP_TYPE_JPEG)
        self.frame = Frame(image)

        self.frame.Show()
        self.SetTopWindow(self.frame)
        return True

def main():
    app = App()
    app.MainLoop()

if __name__ == '__main__':
    main()
```

5 **wx.App subclass**

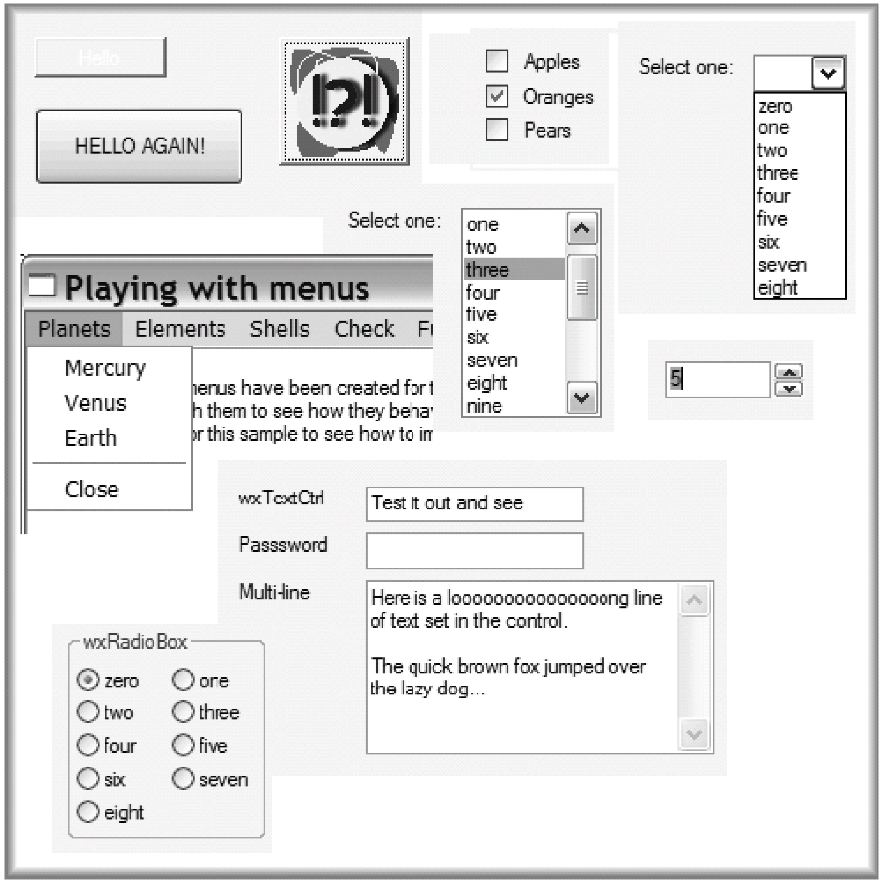
6 **Image handling**

7 **main() function**

8 **Import vs. execute**



Mogućnosti wxPythona

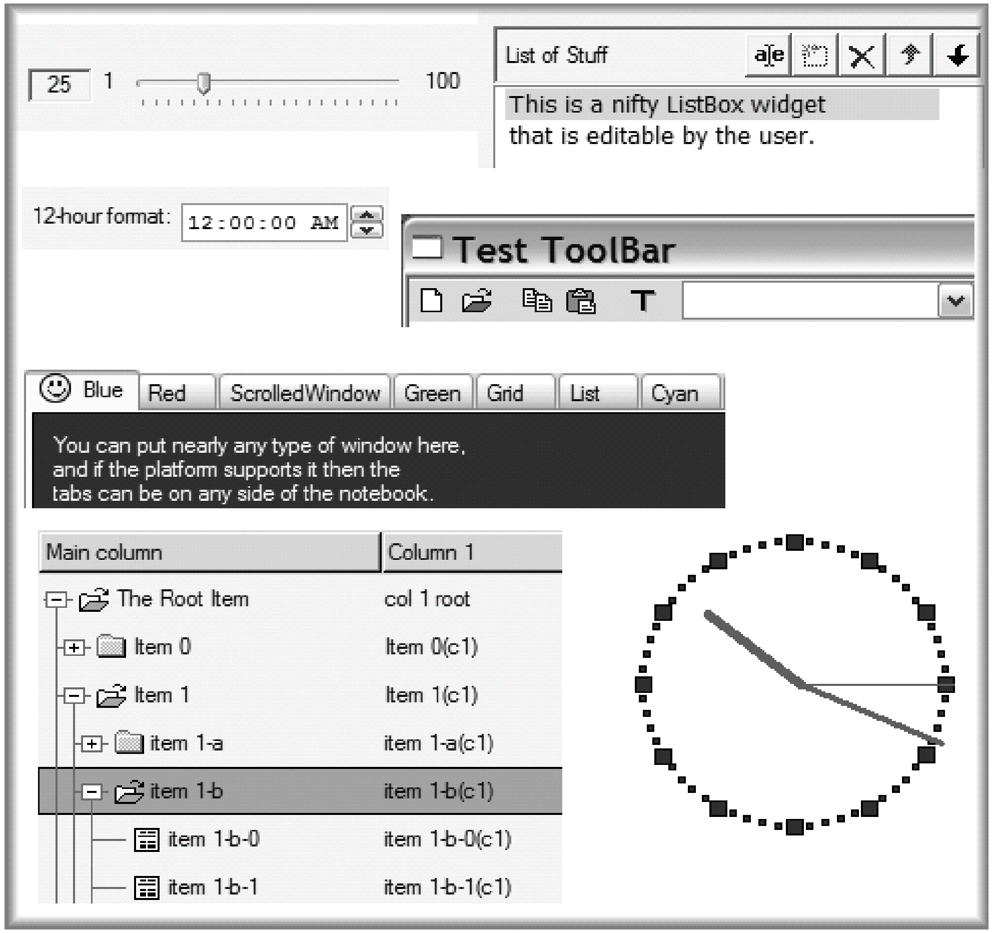


Here is a looooooooooooooong line of text set in the control.

The quick brown fox jumped over the lazy dog...

osnovni widgeti

Mogućnosti wxPythona

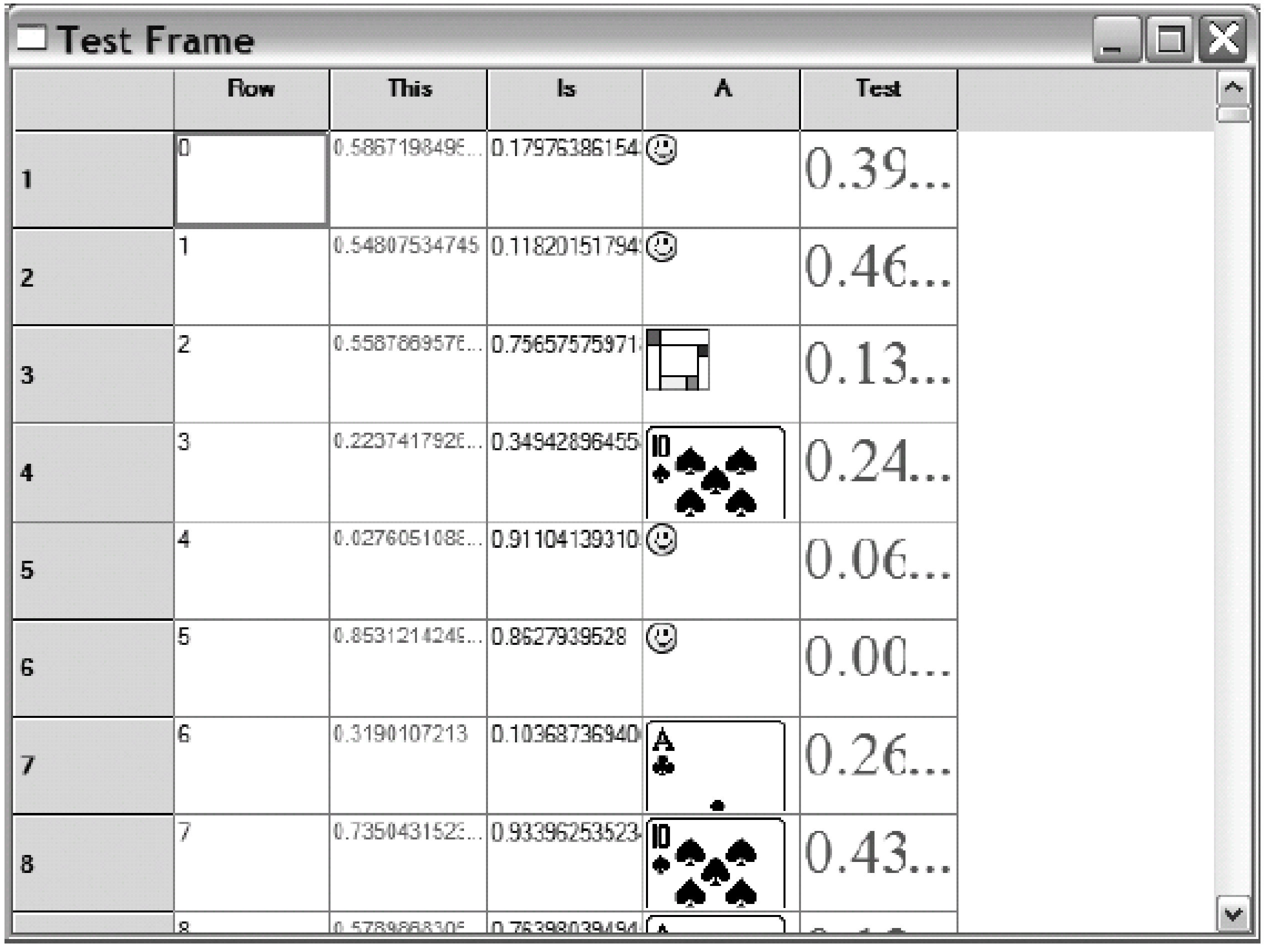


naprednije kontrole

tree list

analogni sat

Mogućnosti wxPythona


Test Frame

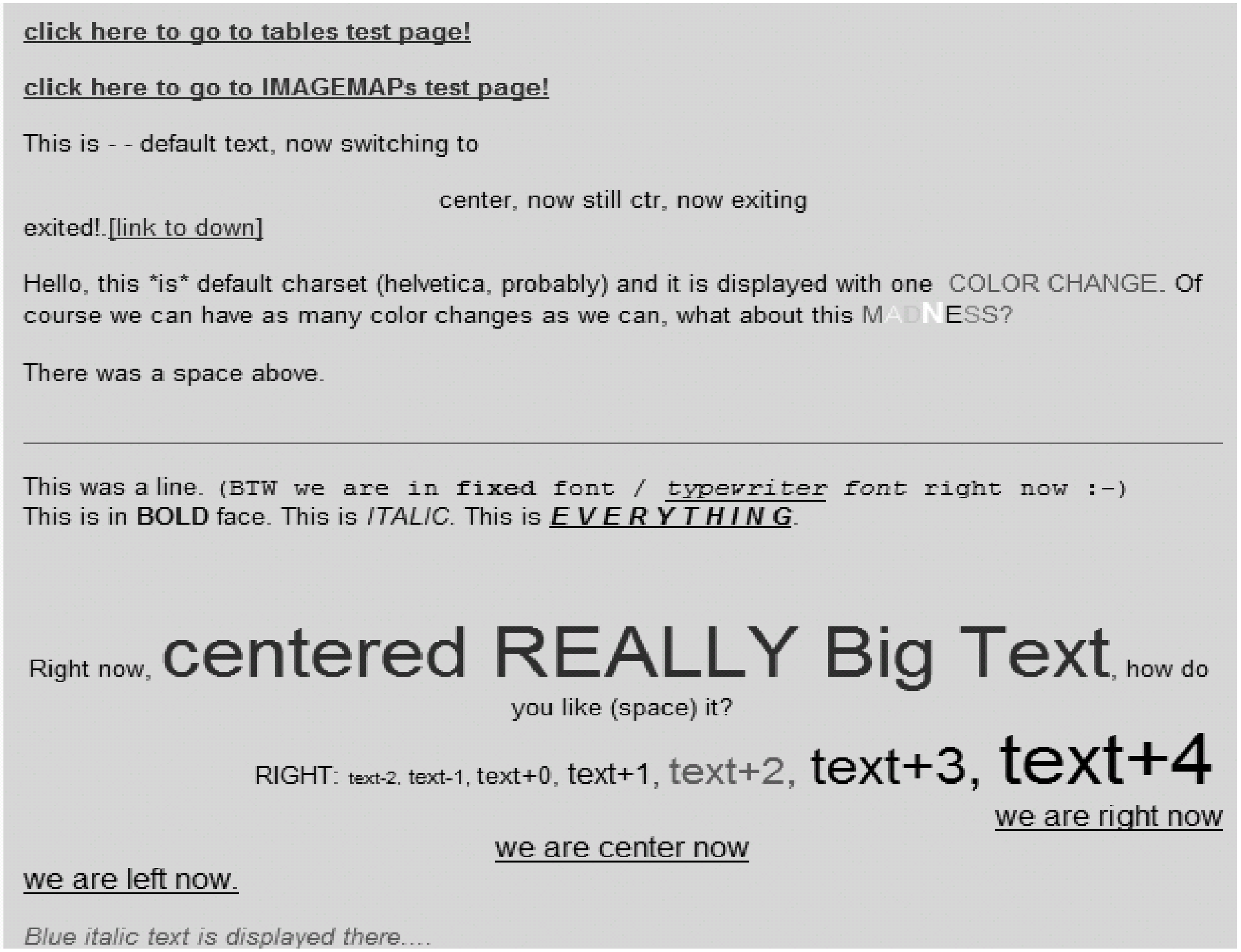
	Row	This	Is	A	Test
1	0	0.5867198495...	0.17976386154	☺	0.39...
2	1	0.54807534745	0.11820151794	☺	0.46...
3	2	0.5587869576...	0.75657575971	▨	0.13...
4	3	0.2237417928...	0.34942896455	10♠	0.24...
5	4	0.0276051088...	0.91104139310	☺	0.06...
6	5	0.8531214249...	0.8627939528	☺	0.00...
7	6	0.3190107213	0.10368736940	A♣	0.26...
8	7	0.7350431523...	0.93396253523	10♠	0.43...
	8	[illegible]	[illegible]		

grid

prikazivanje ćelija

s odabranim pozadinama

Mogućnosti wxPythona



click here to go to tables test page!

click here to go to IMAGEMAPs test page!

This is - - default text, now switching to

center, now still ctr, now exiting

exited!.[link to down]

Hello, this *is* default charset (helvetica, probably) and it is displayed with one COLOR CHANGE. Of course we can have as many color changes as we can, what about this MADNESS?

There was a space above.

This was a line. `(BTW we are in fixed font / typewriter font right now :-)`

This is in **BOLD** face. This is *ITALIC*. This is ***E V E R Y T H I N G***.

Right now, centered REALLY Big Text, how do you like (space) it?

RIGHT: text-2, text-1, text+0, text+1, text+2, text+3, text+4

we are right now

we are center now

we are left now.

Blue italic text is displayed there....

HTML mogućnosti

wx.HTMLwindow

Hello world program

```
import wx

class MyApp(wx.App):

    def OnInit(self):
        frame = MyFrame("Hello World", (50, 60), (450, 340))
        frame.Show()
        self.SetTopWindow(frame)
        return True

class MyFrame(wx.Frame):

    def __init__(self, title, pos, size):
        wx.Frame.__init__(self, None, -1, title, pos, size)
        menuFile = wx.Menu()
        menuFile.Append(1, "&About...")
        menuFile.AppendSeparator()
        menuFile.Append(2, "E&xit")
        menuBar = wx.MenuBar()
        menuBar.Append(menuFile, "&File")
        self.SetMenuBar(menuBar)
```

FRAME

IZBORNIK

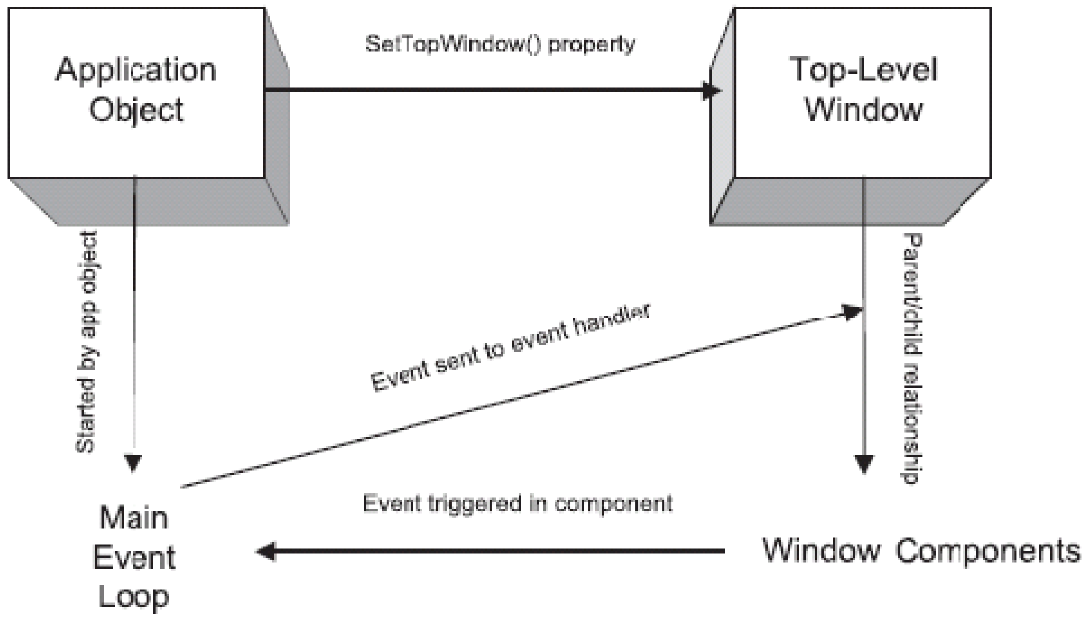
```
        self.CreateStatusBar()
        self.SetStatusText("Welcome to wxPython!")
        self.Bind(wx.EVT_MENU, self.OnAbout, id=1)
        self.Bind(wx.EVT_MENU, self.OnQuit, id=2)

    def OnQuit(self, event):
        self.Close()

    def OnAbout(self, event):
        wx.MessageBox("This is a wxPython Hello world sample",
                "About Hello World", wx.OK | wx.ICON_INFORMATION, self)



if __name__ == '__main__':
    app = MyApp(False)
    app.MainLoop()
```

Traka statusna

Dijalog

wxPython aplikacija

- application object - objekt aplikacije iz wx.App
  - poziva glavnu petlju
  - odziv na događaje koji inače nisu napravljeni
  - sadrži glavni prozor i glavnu petlju



objekt aplikacije

1. Definira se subklasa
2. Napisati metodu OnInit() u subklasi
3. U glavnom dijelu programa napraviti instancu klase
4. Pozvati MainLoop() metodu koja prenosi kontrolu programa na wxPython

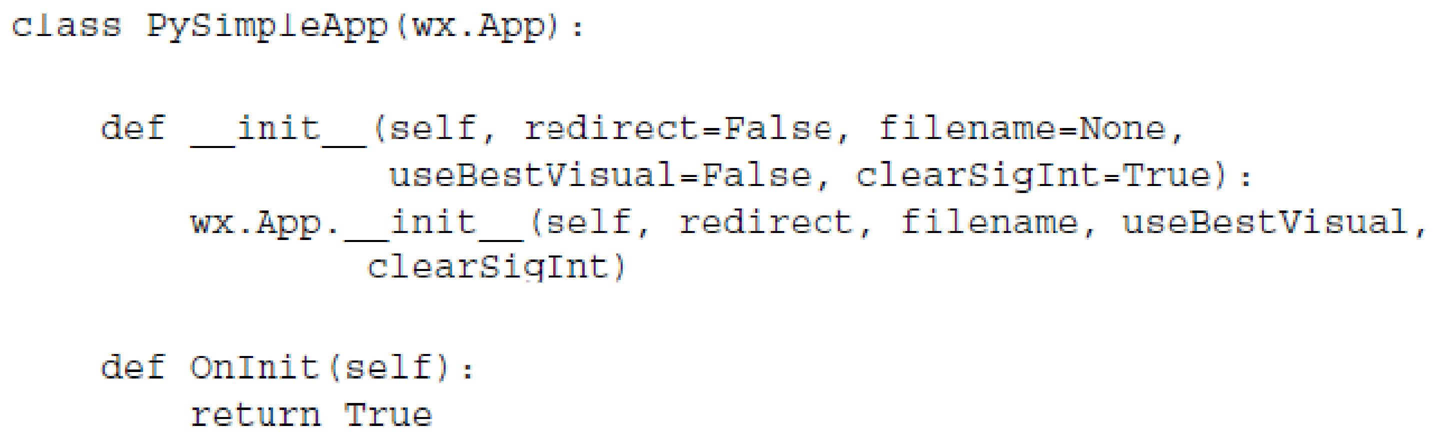
Metoda OnInit() je dio wxPythona, koristimo za sve potrebne postavke (inicijalizacije), a ne u __init__ metodi (konstruktoru). Ako koristimo konstruktor moramo pozvati konstruktor od objekta aplikacije

```
wx.App.__init__(self)
```

U OnInit() napravimo tipično barem 1 Frame objekt

wx.App subklasa

- Kada možemo izostaviti subklasu od wx.App? Obično radimo subklasu kako bi mogli definirati Frame u OnInit()
  - kada imamo samo jedan Frame, objekt aplikacije je trivijalan
  - koristimo wx.PySimpleApp klasu definiranu u wxPythonu.



```
class PySimpleApp(wx.App):

    def __init__(self, redirect=False, filename=None,
                 useBestVisual=False, clearSigInt=True):
        wx.App.__init__(self, redirect, filename, useBestVisual,
                 clearSigInt)

    def OnInit(self):
        return True
```

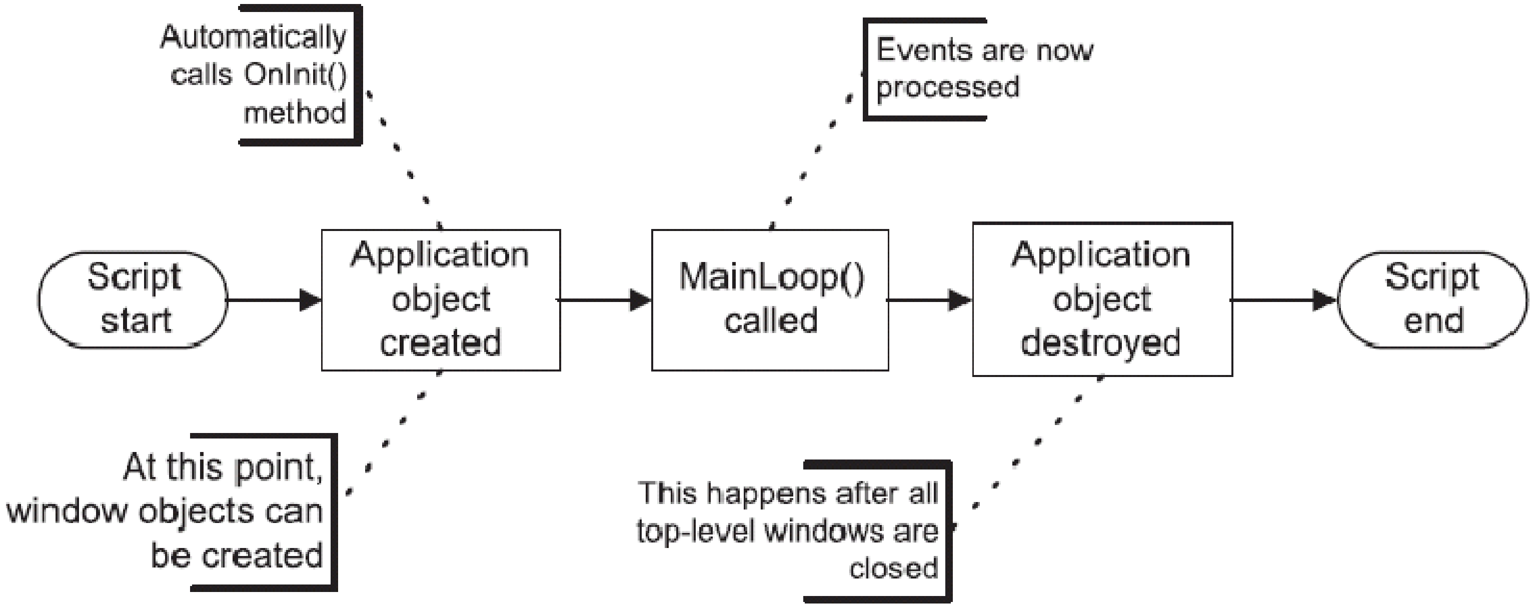
wx.PySimpleApp primjena

Klasu PySimpleApp jednostavno koristimo

```
if __name__ == '__main__':
    app = wx.PySimpleApp()
    frame = MyNewFrame(None)
    frame.Show(True)
    app.MainLoop()
```

Život objekta u wxPythonu

Zatvaranjem prozora završava MainLoop(), ne mora se podudarati s programom.


Automatically calls OnInit() method
Events are now processed
Script start
Application object created
MainLoop() called
Application object destroyed
Script end
At this point, window objects can be created
This happens after all top-level windows are closed

redirekcija u wxPythonu

startup.py

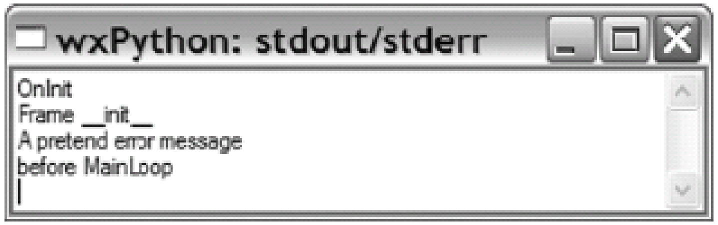
```
#!/usr/bin/env python

import wx
import sys

class Frame(wx.Frame):

    def __init__(self, parent, id, title):
        print "Frame __init__"
        wx.Frame.__init__(self, parent, id, title)

class App(wx.App):

    def __init__(self, redirect=True, filename=None):
        print "App __init__"
        wx.App.__init__(self, redirect, filename)
```

Koristi sys.stdout i sys.stderr, standardni izlazi za poruke i pogreške.

wxPython pod MS Windows kontrolira ove izlaze i zamjenjuje ih prozorom.

redirekcija u wxPythonu

```
    def OnInit(self):
        print "OnInit"        <-- Writing to stdout
        self.frame = Frame(parent=None, id=-1, title='Startup')  <-- Creating the frame
        self.frame.Show()
        self.SetTopWindow(self.frame)
        print >> sys.stderr, "A pretend error message"    <-- Writing to stderr
        return True

    def OnExit(self):
        print "OnExit"

if __name__ == '__main__':
    app = App(redirect=True)      ❶  Text redirection starts here
    print "before MainLoop"
    app.MainLoop()          ❷  The main event loop is entered here
    print "after MainLoop"
```



```
OnInit
Frame __init__
A pretend error message
before MainLoop
```

```
if __name__ == '__main__':
    app = App(redirect=True)      ❶  Text redirection starts here
    print "before MainLoop"
    app.MainLoop()          ❷  The main event loop is entered here
    print "after MainLoop"
```

Glavni prozor

Korisnik vidi program kao "glavni prozor" - top-level window

Glavni prozor obično dobivamo iz wx.Frame ili wx.Dialog klasa

Postoji veliki broj već definiranih dijaloga u wx.Dialog klasi

"top-level" prozor je bilo koji widget bez roditelja

Samo jedan je "glavni prozor" - pomoću metode SetTopWindow()

Default: prvi Frame u wx.App postaje "glavni" prozor

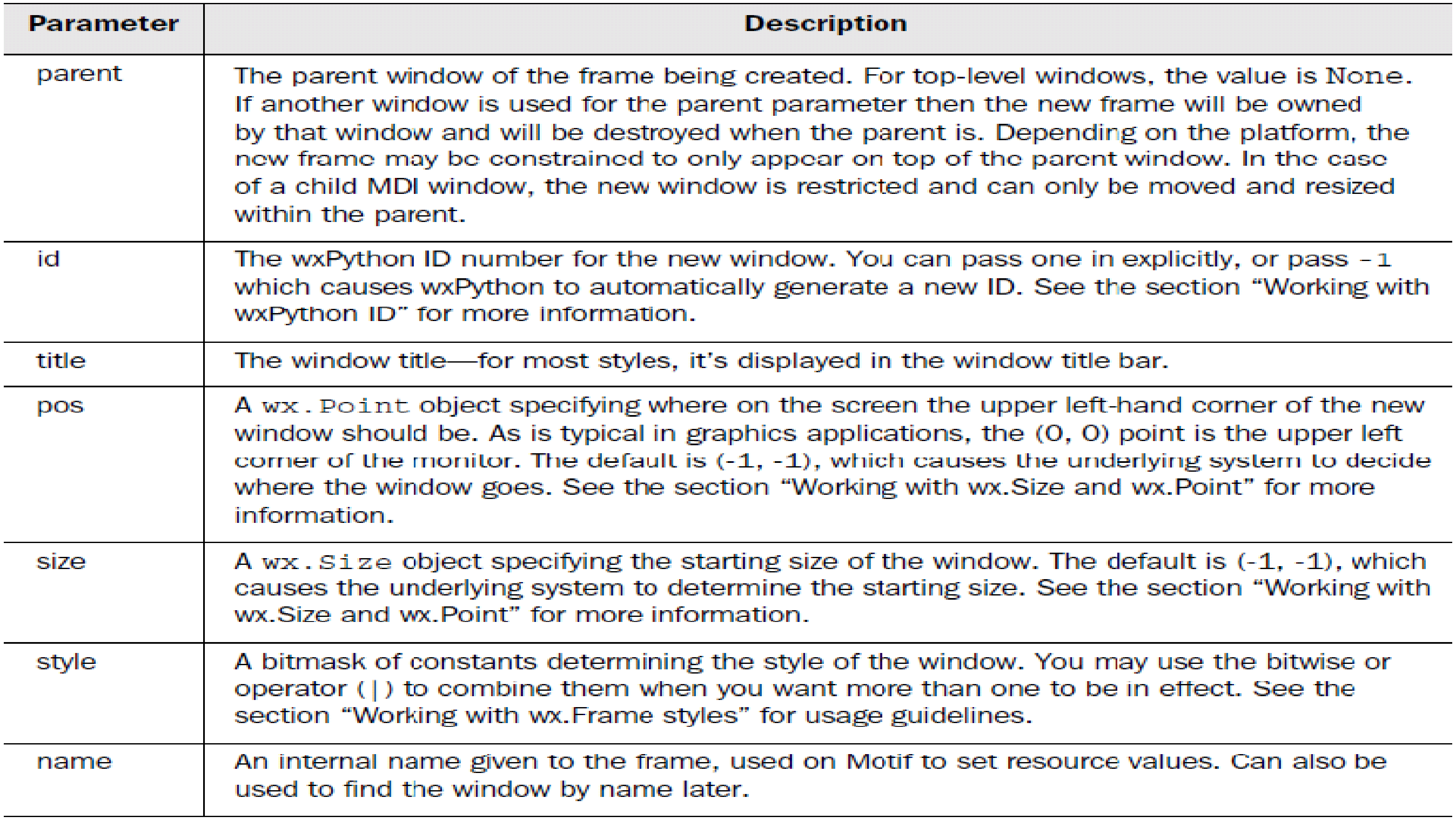
wx.Frame=prozor

- GUI korisnik vidi Frame kao prozor
- wx.Frame je roditelj svih Frame objekata u wxPythonu
- Subklasa od wx.Frame koja ima __init__ metodu mora zvati konstruktor koji ima opcije

```
wx.Frame(parent, id=-1, title="", pos=wx.DefaultPosition,
        size=wx.DefaultSize, style=wx.DEFAULT_FRAME_STYLE,
        name="frame")
```

To su parametri koje možemo poslati konstruktoru wx.Frame.__init__()

Frame parametri



Parameter	Description
parent	The parent window of the frame being created. For top-level windows, the value is None. If another window is used for the parent parameter then the new frame will be owned by that window and will be destroyed when the parent is. Depending on the platform, the new frame may be constrained to only appear on top of the parent window. In the case of a child MDI window, the new window is restricted and can only be moved and resized within the parent.
id	The wxPython ID number for the new window. You can pass one in explicitly, or pass `-1` which causes wxPython to automatically generate a new ID. See the section "Working with wxPython ID" for more information.
title	The window title—for most styles, it's displayed in the window title bar.
pos	A `wx.Point` object specifying where on the screen the upper left-hand corner of the new window should be. As is typical in graphics applications, the (0, 0) point is the upper left corner of the monitor. The default is (-1, -1), which causes the underlying system to decide where the window goes. See the section "Working with wx.Size and wx.Point" for more information.
size	A `wx.Size` object specifying the starting size of the window. The default is (-1, -1), which causes the underlying system to determine the starting size. See the section "Working with wx.Size and wx.Point" for more information.
style	A bitmask of constants determining the style of the window. You may use the bitwise or operator (`|`) to combine them when you want more than one to be in effect. See the section "Working with wx.Frame styles" for usage guidelines.
name	An internal name given to the frame, used on Motif to set resource values. Can also be used to find the window by name later.

Frame

Id prozora, cijeli broj koji mora biti jedinstven u programu

NewId() generira id.

```
id = wx.NewId()
frame = wx.Frame.__init__(None, id)
```

Ne zanima nas id

```
frame = wx.Frame.__init__(None, -1)
id = frame.GetId()
```

Klase wx.Point i wx.Size.

```
point = wx.Point(10, 12)
```

(0,0) je default

```
x = point.x
y = point.y
```

eksplicitno definiramo veličinu

i položaj.

```
frame = wx.Frame(None, -1, pos=(10, 10), size=(100, 100))
```

Dinamička promjena položaja

```
frame.SetPosition((2, 3))
```

Stil Frame objekta

wx.DEFAULT_FRAME_STYLE BITMASKE

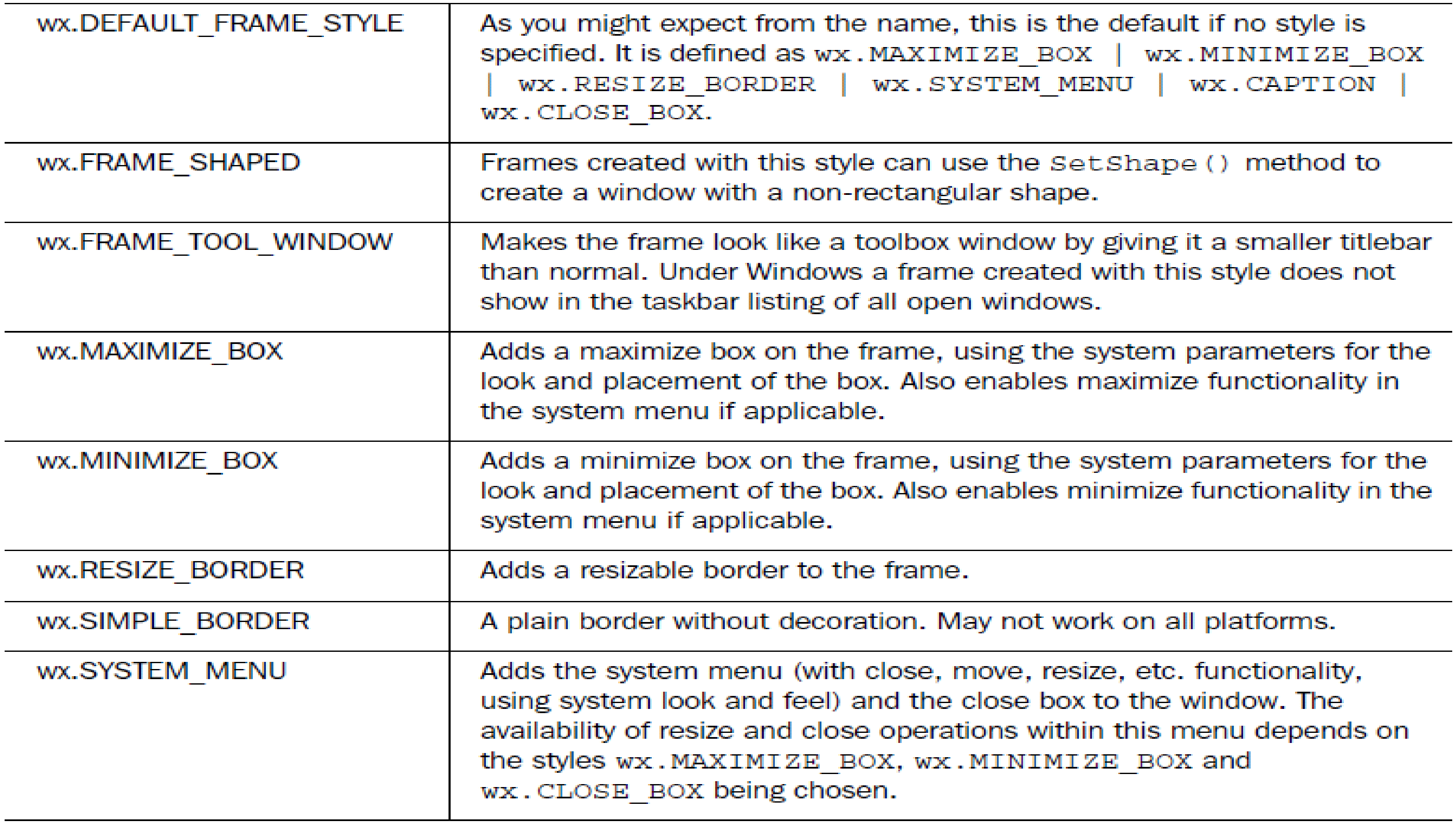
```
wx.MAXIMIZE_BOX | wx.MINIMIZE_BOX | wx.RESIZE_BORDER |
wx.SYSTEM_MENU | wx.CAPTION | wx.CLOSE_BOX
```

Default stil modificiran tako da se ne može promjeniti veličina prozora

```
wx.DEFAULT_FRAME_STYLE ^ (wx.RESIZE_BORDER | wx.MINIMIZE_BOX |
wx.MAXIMIZE_BOX)
```

Style	Description
wx.CAPTION	Adds a title bar on the frame, which displays the frame's `Title` property.
wx.CLOSE_BOX	Instructs the system to display a close box on the frame's title bar, using the system defaults for placement and style. Also enables the close item on the system menu if applicable.

stilovi



wx.DEFAULT_FRAME_STYLE	As you might expect from the name, this is the default if no style is specified. It is defined as `wx.MAXIMIZE_BOX | wx.MINIMIZE_BOX | wx.RESIZE_BORDER | wx.SYSTEM_MENU | wx.CAPTION | wx.CLOSE_BOX`.
wx.FRAME_SHAPED	Frames created with this style can use the `SetShape()` method to create a window with a non-rectangular shape.
wx.FRAME_TOOL_WINDOW	Makes the frame look like a toolbox window by giving it a smaller titlebar than normal. Under Windows a frame created with this style does not show in the taskbar listing of all open windows.
wx.MAXIMIZE_BOX	Adds a maximize box on the frame, using the system parameters for the look and placement of the box. Also enables maximize functionality in the system menu if applicable.
wx.MINIMIZE_BOX	Adds a minimize box on the frame, using the system parameters for the look and placement of the box. Also enables minimize functionality in the system menu if applicable.
wx.RESIZE_BORDER	Adds a resizable border to the frame.
wx.SIMPLE_BORDER	A plain border without decoration. May not work on all platforms.
wx.SYSTEM_MENU	Adds the system menu (with close, move, resize, etc. functionality, using system look and feel) and the close box to the window. The availability of resize and close operations within this menu depends on the styles `wx.MAXIMIZE_BOX`, `wx.MINIMIZE_BOX` and `wx.CLOSE_BOX` being chosen.

primjeri

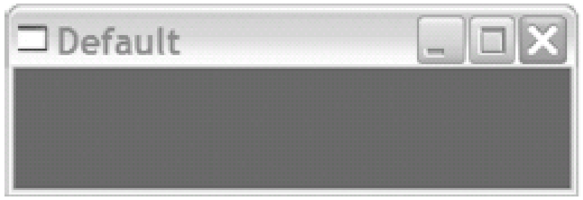


Figure 2.4 A frame created with the default style

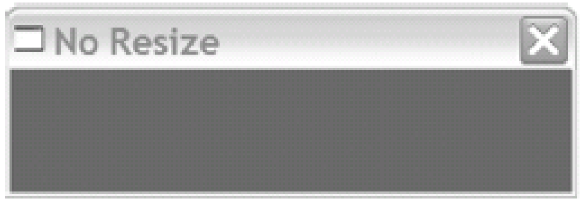


Figure 2.5 A frame created to be non-resizable. Notice the lack of minimize/maximize buttons.

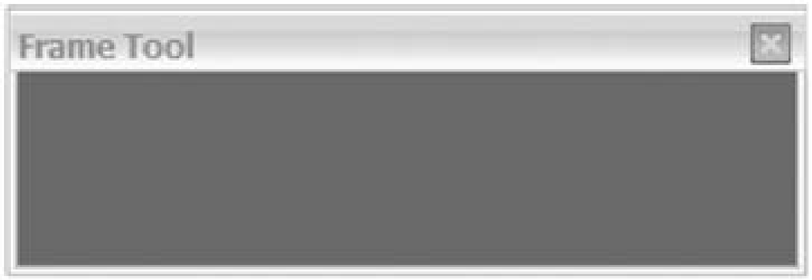


Figure 2.6 A toolbar frame, with a smaller title bar and no system menu

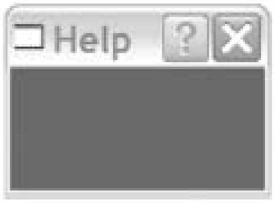
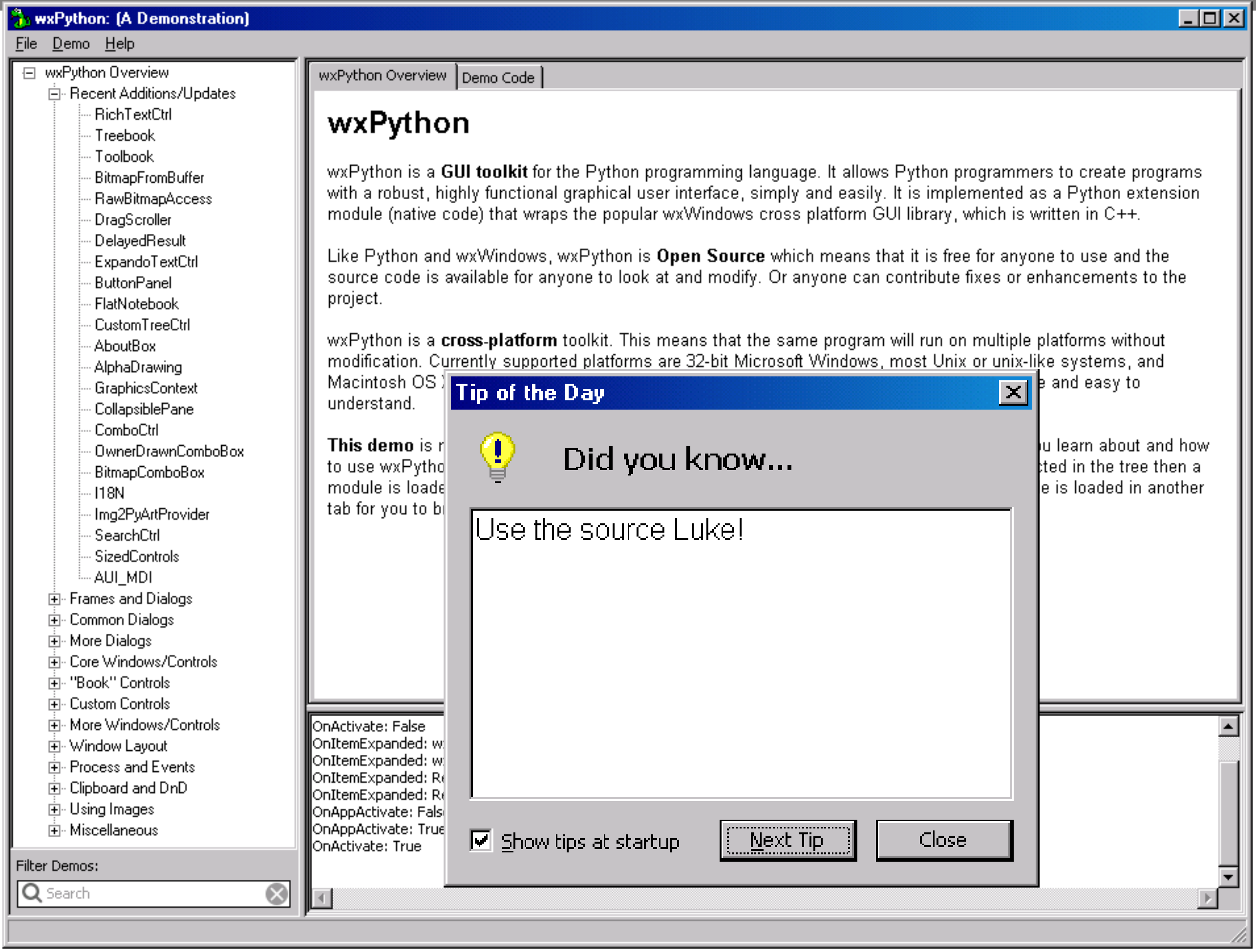


Figure 2.7 A frame with a help button

demo wxpython



wxPython

wxPython is a **GUI toolkit** for the Python programming language. It allows Python programmers to create programs with a robust, highly functional graphical user interface, simply and easily. It is implemented as a Python extension module (native code) that wraps the popular wxWindows cross platform GUI library, which is written in C++.

Like Python and wxWindows, wxPython is **Open Source** which means that it is free for anyone to use and the source code is available for anyone to look at and modify. Or anyone can contribute fixes or enhancements to the project.

wxPython is a **cross-platform** toolkit. This means that the same program will run on multiple platforms without modification. Currently supported platforms are 32-bit Microsoft Windows, most Unix or unix-like systems, and Macintosh OS X e and easy to understand.

This demo is u learn about and how to use wxPytho cted in the tree then a module is loade e is loaded in another tab for you to b

Use the source Luke!

```
OnActivate: False
OnItemExpanded: w
OnItemExpanded: w
OnItemExpanded: R
OnItemExpanded: R
OnAppActivate: Fals
OnAppActivate: True
OnActivate: True
```

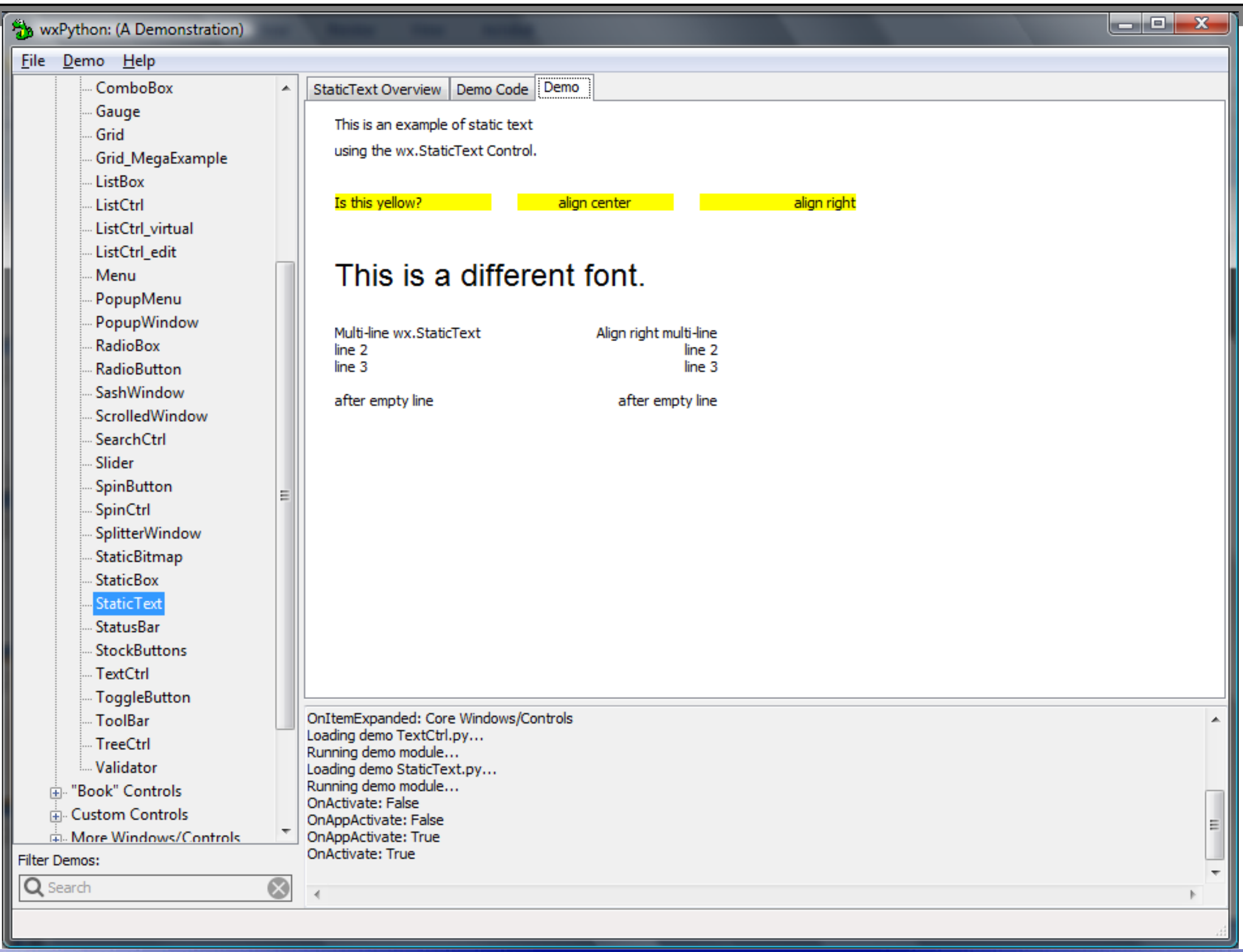
demo.py

direktorij s

dokmentima od

wxpythona

Demo StaticText



This is an example of static text
using the wx.StaticText Control.

Is this yellow? align center align right

This is a different font.

```
Multi-line wx.StaticText                Align right multi-line
line 2                                                  line 2
line 3                                                  line 3

after empty line                              after empty line
```

```
OnItemExpanded: Core Windows/Controls
Loading demo TextCtrl.py...
Running demo module...
Loading demo StaticText.py...
Running demo module...
OnActivate: False
OnAppActivate: False
OnAppActivate: True
OnActivate: True
```

Demo staticText

```
import  wx
USE_GENERIC = 0
if USE_GENERIC:
    from wx.lib.stattext import GenStaticText as StaticText
else:
    StaticText = wx.StaticText
```

```
def runTest(frame, nb, log):
    panel = TestPanel(nb)
    return panel
```

```
class TestPanel(wx.Panel):
    def __init__(self, parent):
        wx.Panel.__init__(self, parent, -1)
        ##self.SetBackgroundColour("sky blue")

        StaticText(self, -1, "This is an example of static text", (20, 10))
        StaticText(self, -1, "using the wx.StaticText Control.", (20, 30))
……………………………………………………….
```

```
if __name__ == '__main__':
    import sys,os
    import run
    run.main(['', os.path.basename(sys.argv[0])]
        …..+ sys.argv[1:])
```

Demo StaticText

```
import  wx
StaticText = wx.StaticText
class MyFrame(wx.Frame):
    def __init__(self,titlew="My Frame",size=(500,300)):
        wx.Frame.__init__(self, None, -1, titlew, size=size)
#       panel=TestPanel(self)
```

```
class TestPanel(wx.Panel):
    def __init__(self, parent):
        wx.Panel.__init__(self, parent, -1)
……………………………………………
```

```
if __name__ == '__main__':
        app = wx.PySimpleApp()
        frame = MyFrame(size=(200,300))
#       panel=TestPanel(frame)
        runTest(None,frame,None)
        frame.Show(True)

        app.MainLoop()
```

zadatak

- Prostudiraj primjere sample.py i spare.py. Napiši aplikaciju koja će imati subklasu Xapp od wx.App i subklasu Xframe od wx.Frame. U klasi Xapp inicijaliziraj Xframe. U klasi Xframe napravi statično polje "Prezime", zatim TextCtrl u kojem se upisuje prezime. Ispiši naredbom print prezime (koristi GetValue() metodu)
- Postavi ova dva polja jedno ispod drugog.
- Pokreni demo.py demo program od wxpythona
- Kako dobijemo help u python interpreteru za widget wx.TextCtrl. Da li je ova informacije korisna?
- Promjeni atribute od prozora iz prve aplikacije tako da 1) gumb za zatvaranje prozora ne radi, 2) ukloni naslov i sistemske gumbe na prozoru.