

Introducción a wxPython

<http://wxPython.org/>

Cross-Platform GUI Library

Lic. Marcelo Fernández

marcelo.fidel.fernandez@gmail.com - <http://blog.marcelofernandez.info>

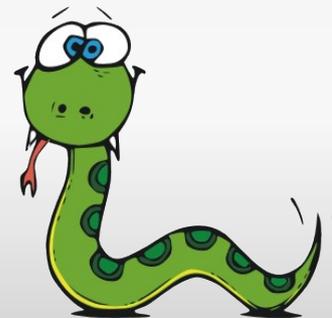
Publicado bajo Licencia Creative Commons - BY,
excepto las imágenes y logos tomadas de sitios de Internet



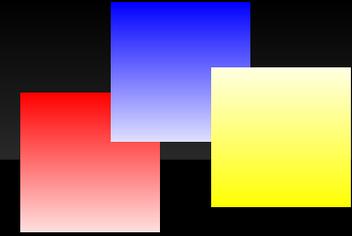
Introducción - GUIs en Python

- ¿Qué es una GUI?
- Bibliotecas: "Bindings" o "Wrappers"
- Disponibles en Python:
 - Tcl/Tk: Instalada por defecto.
 - **wxPython: wrapper de wxWidgets** (antes wxWindows)
 - PyGTK: wrapper de GTK.
 - PyQt: wrapper de QT.

(Ver comparativa en <http://python.org.ar/pyar/InterfacesGraficas>)



Introducción - wxPython



Lenguajes

Bindings

Mi Aplicación
Python

Aplicación X
Haskell

wxD

wxErlang

wxPerl

WxPython
Python/C++

wxRuby

wxHaskell

Otros

wxWidgets
C++

Windows

Win32 / Win64

Linux, Unix/GTK

wxGTK

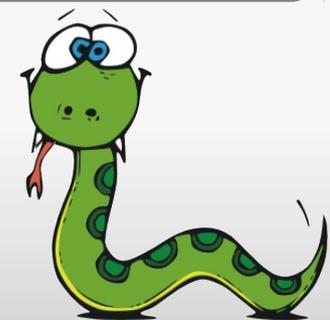
Mac OSX

wxMac

Otras

wxX11, wxDFB, wxMotif...

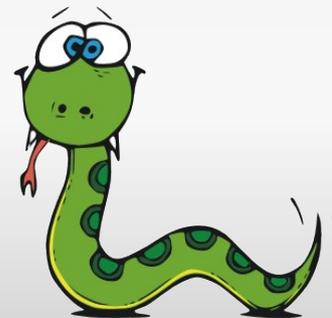
Widgets / Plataformas



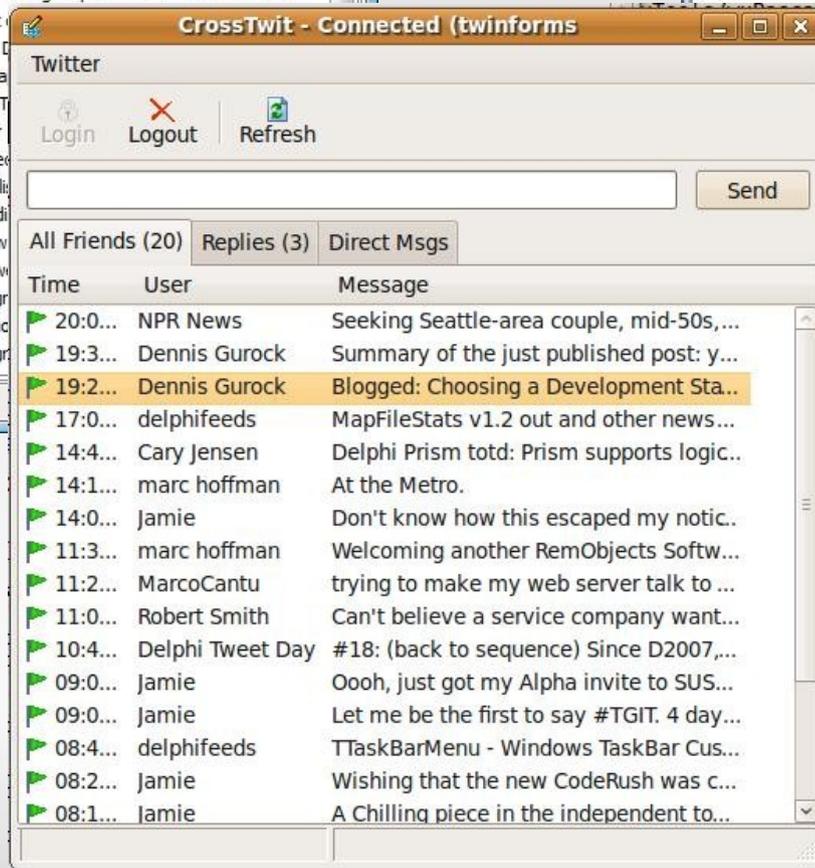
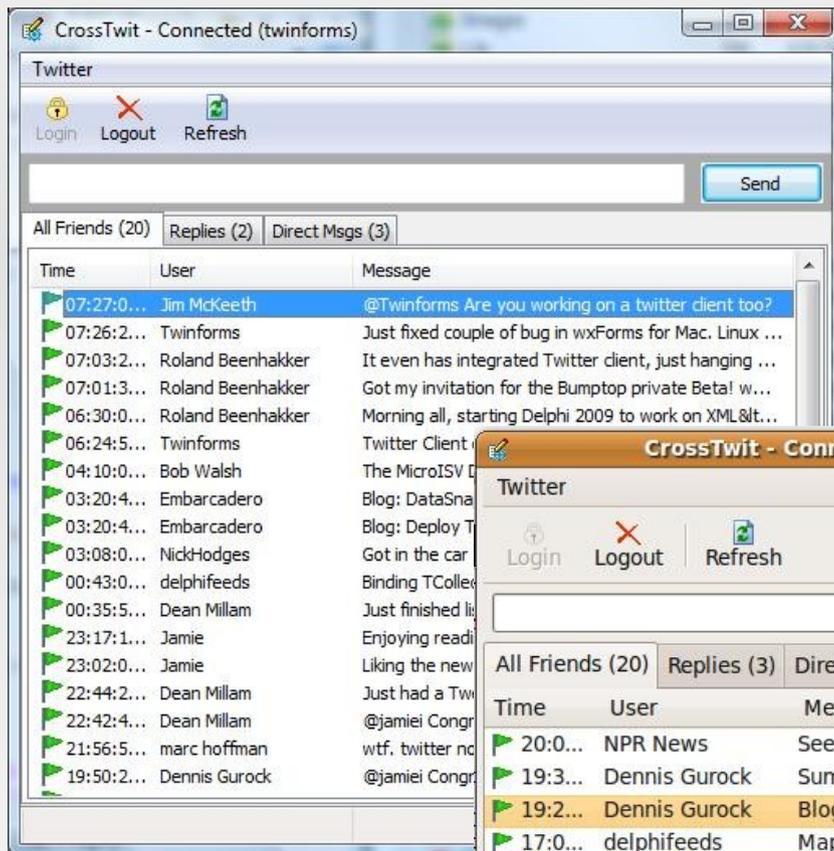
Ver más en http://wiki.wxwidgets.org/General_Information

Introducción - wxWidgets

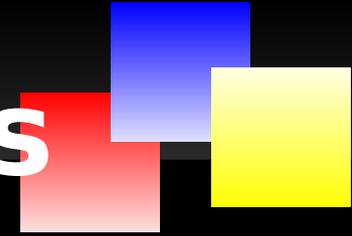
- wxPython es una biblioteca para usar **wxWidgets** (C++) desde Python
- Hereda sus características
 - Robusta, años evolucionando (1992).
 - Pensado para ser Multiplataforma desde el inicio.
 - Conserva el *Look and Feel* del entorno y su velocidad, ya que utiliza componentes GUI estándar de cada SO.
 - Permite embeber componentes nativos.
 - LGPL. Aplicaciones con cualquier licencia.
 - Windows, Linux, Mac y más con casi el mismo código fuente, sólo recompilando.



Introducción - wxWidgets



Introducción - wxWidgets



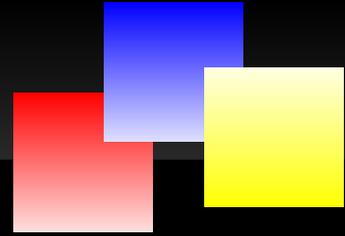
staccato005
File Edit View Project Generate Effect Analyze Help
20:00 21:00 22:00 23:00 24:00 25:00
staccato00
Left, 32000Hz
32-bit float
Mute Solo
L R
Project rate: 44100 Selection: 17:43.794309 - 18:31.633581

Fingertips (10)
File Edit View Project Generate Effect Analyze Help
1.920 1.930 1.940 1.950 1.960 1.970 1.980 1.990 2.000 2.010
Channel 1
Mono, 44100Hz
32-bit float
Mute Solo
L R
Channel 2
Mono, 44100Hz
32-bit float
Mute Solo
L R
Project rate: 44100 Selection: 0:01.958387 - 0:01.986387 (0:00.028000 min:sec) [Snap-To Off]

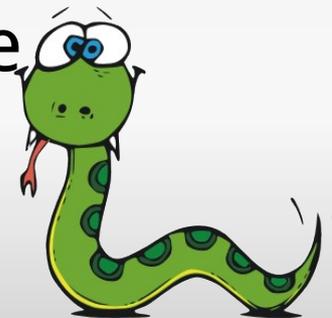
Moonglow
File Edit View Project Generate Effect Analyze Help
40 45 50
Moonglow
Stereo, 44100Hz
32-bit float
Mute Solo
L R
Project rate: 44100 Cursor: 0:00.000000 min:sec [Snap-To Off]



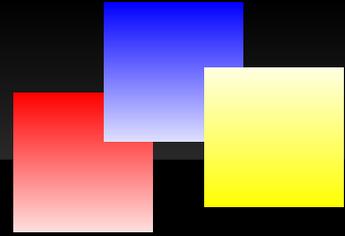
Introducción - wxPython



- wxPython amplía las ventajas de wxWidgets:
 - Es Python: más fácil de aprender y adecuado que C++.
 - Código "pythónico".
 - + Fácilmente extensible, **AGW** es un ejemplo.
 - Windows, Linux, Mac y más con el mismo código.
- Desventajas
 - Instalación: No está incluido en Python mismo.
 - Muchas capas de abstracción.
 - Curva de aprendizaje media/alta, lógicamente según la aplicación que se quiera desarrollar



Ejemplo 1 - Hola Mundo



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

```
import wx
```

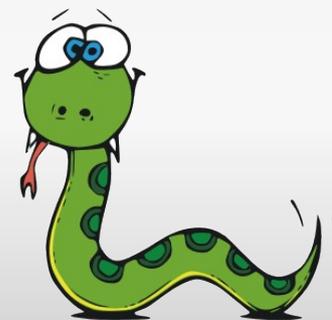
```
if __name__ == '__main__':
```

```
    app = wx.App()
```

```
    frame = wx.Frame(None, wx.ID_ANY, "Hola Mundo")
```

```
    frame.Show()
```

```
    app.MainLoop()
```



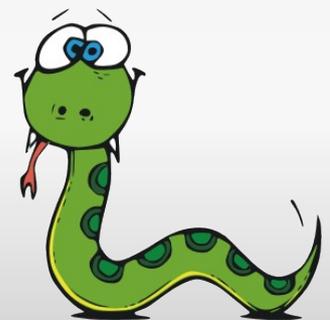
Ejemplo 2 - Estructura Base

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

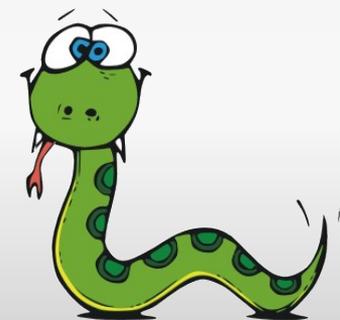
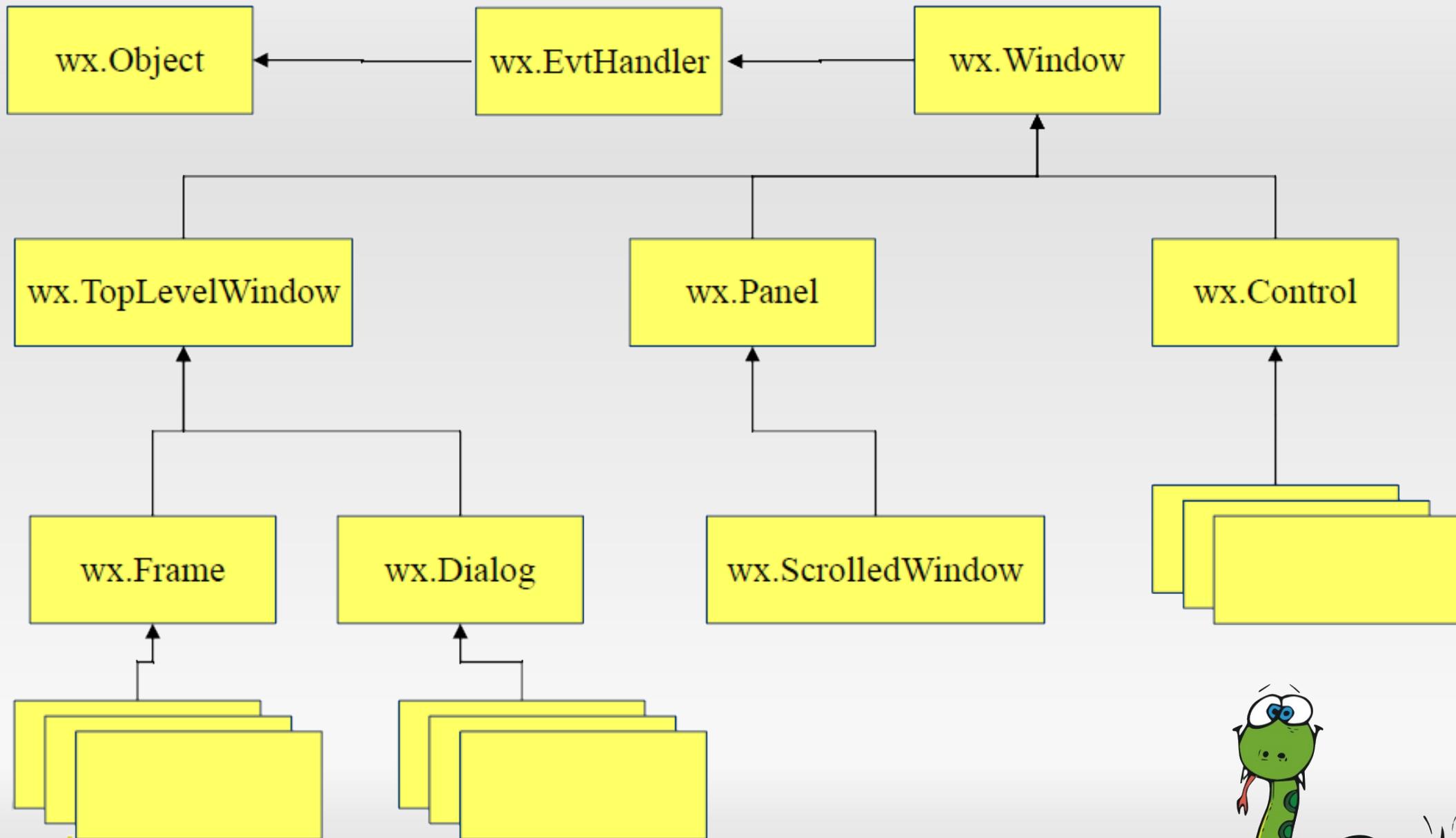
```
import wx
```

```
class TestFrame(wx.Frame):  
    def __init__(self, parent, title):  
        wx.Frame.__init__(self, parent, title=title, size=(200,100))  
        self.control = wx.TextCtrl(self, style=wx.TE_MULTILINE)
```

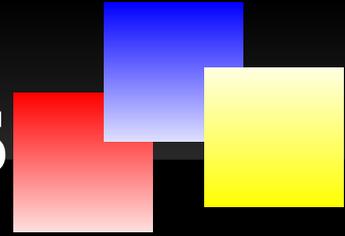
```
if __name__ == '__main__':  
    app = wx.App()  
    frame = TestFrame(None, 'Editor de Texto')  
    frame.Center()  
    frame.Show()  
    app.MainLoop()
```



Jerarquía de Clases - wxPython



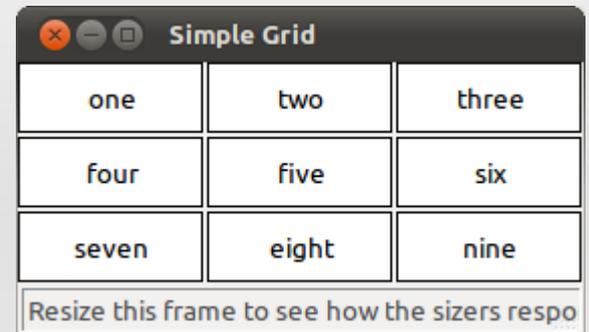
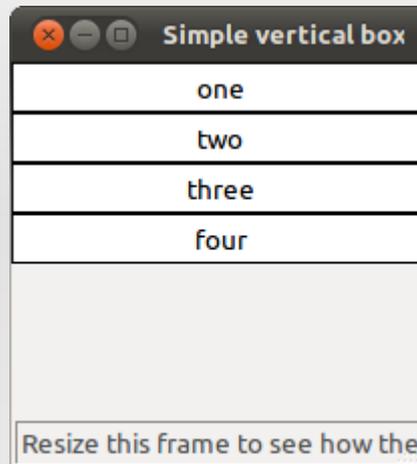
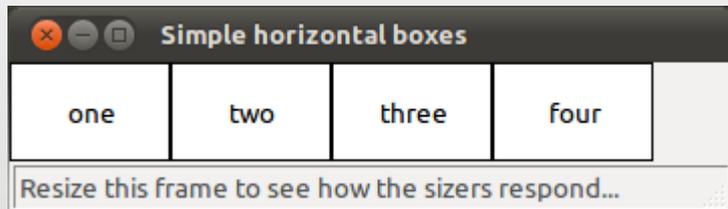
Organización de Widgets



- Organización Estática
 - Posicionamiento basado en pixels
 - Gestión manual de la ubicación de los componentes
 - Pero limitado por donde se lo mire:
 - Monitores y/o Resoluciones diferentes
 - Idiomas, SOs, "Skins", Tipografías diferentes
- Organización Dinámica
 - Mucho más util en ~~todos~~ el resto de los casos
 - wxWidgets lo provee mediante los Sizers
 - Diferentes algoritmos de posicionamiento, diferentes subclases de Sizer.

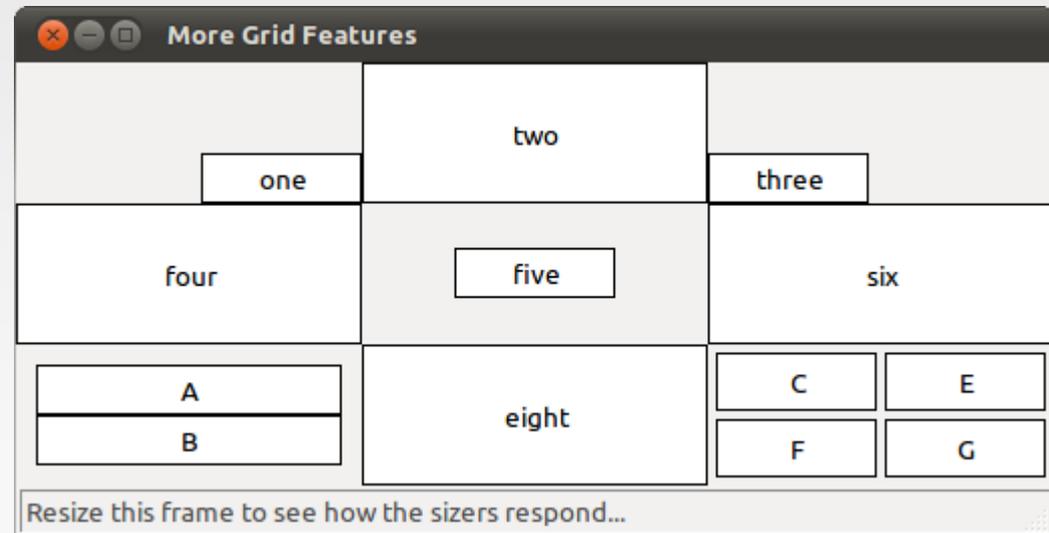
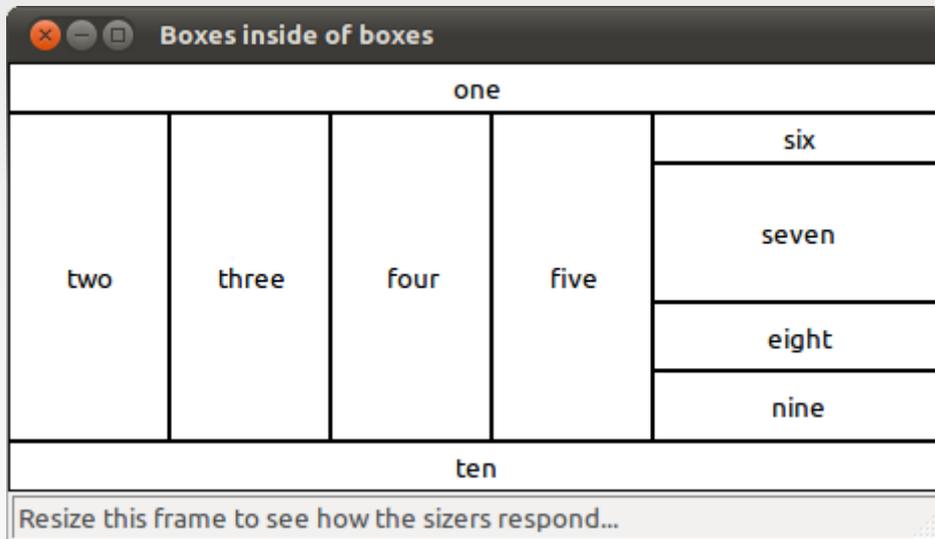


Posición Dinámica - Sizers

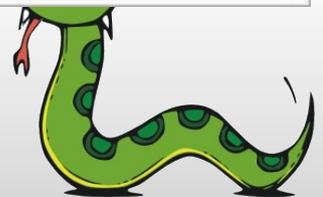


Horizontal, Vertical Boxes

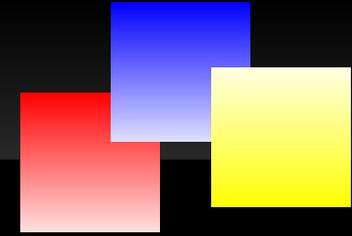
Grid



Sizers combinados y anidados



Ejemplo 3 - Sizers



```
# ... Dentro de la clase MainWindow
# Sizer de Botones
self.sizer_botones = wx.BoxSizer(wx.HORIZONTAL)
# ...

# Widget -- Caja de Texto
self.txtNotes = wx.TextCtrl(self, style=wx.TE_MULTILINE)

# Sizer de Grilla
self.sizer_form = wx.GridSizer(rows=2, cols=3)

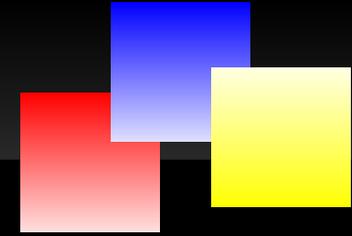
# ...

# Agrego cada Sizer al Sizer de la Ventana con su proporción y flags
self.sizer = wx.BoxSizer(wx.VERTICAL)
self.sizer.Add(self.sizer_botones, proportion=0, flag=wx.EXPAND)
self.sizer.Add(self.txtNotes, proportion=1, flag=wx.EXPAND)
self.sizer.Add(self.sizer_form, proportion=0)
self.SetSizer(self.sizer)

# ...
```



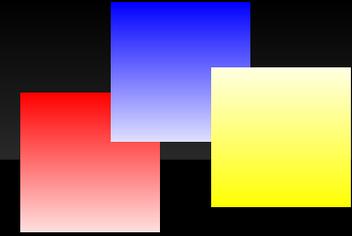
WxPython - Eventos



- wxPython está en un loop infinito, el `MainLoop()`, esperando que el usuario haga algo.
- Acción(Objeto) => Reacción = Evento
- `Bind()` permite que un evento invoque una función
- `window.Bind(wx.EVT_BUTTON, self.OnClick)`
- `def OnClick(self, event):`
 `print 'Click!' # event es una instancia de wx.Event`
- Siguen una jerarquía en algunos casos
- `event.Skip()` permite que el evento la siga



Ejemplo 4/1 - Eventos



```
# ... Dentro de la clase MainWindow – Ejemplo Mouse

# Defino botones Aceptar/Cancelar
self.btnAceptar = wx.Button(self,wx.ID_ANY,u'&Aceptar')

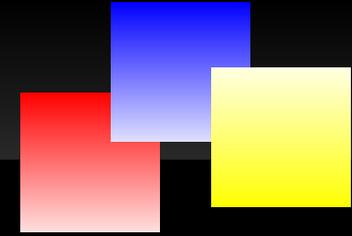
# Conecto el evento click a la función OnAceptar
self.btnAceptar.Bind(wx.EVT_BUTTON, self.OnAceptar)

# ...

def OnAceptar(self, event):
    dlg = wx.MessageDialog(self,u'¡Aceptar!',u'Test',wx.OK)
    dlg.ShowModal()
    dlg.Destroy()
```



Ejemplo 4/2 - Eventos

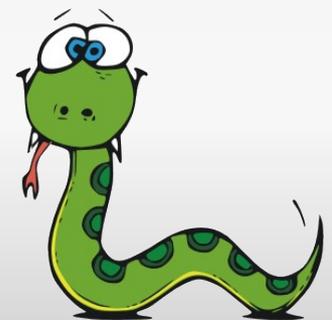


```
# ... Dentro de la clase MainWindow – Ejemplo Teclado
```

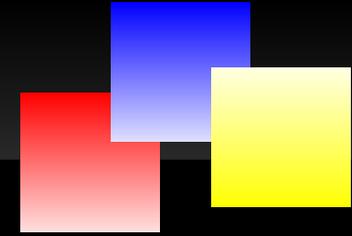
```
# Creo el widget TextCtrl  
self.txtNombre = wx.TextCtrl(self)
```

```
# Conecto el evento Key_Up a OnTxtNombre  
self.txtNombre.Bind(wx.EVT_KEY_UP, self.OnTxtNombre)  
# ...
```

```
def OnTxtNombre(self, event):  
    keycode = event.GetKeyCode()  
    print u'Tecleó en Nombre: ' + str(keycode)  
    if keycode == wx.WXK_TAB:  
        print u'Tab!'
```



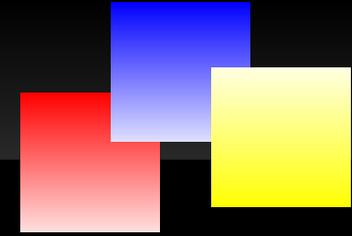
Repaso hasta aquí



- Con poco código se pueden hacer aplicaciones de escritorio multiplataforma, sin salir de Python...
- La instalación para un desarrollador es muy sencilla y para el cliente también.
- ¡Aplicaciones con *Look and Feel* nativo!
- Lo malo: **Este método no escala.**
- La "Capa Visual" y la "Lógica de Negocio" están mezcladas. Algo no va bien...



Diseñadores de GUI



- Ventajas

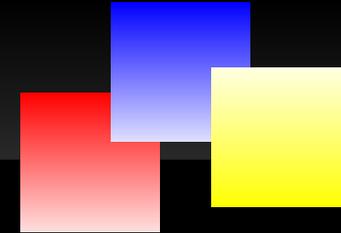
- Flexibilidad. Permiten separar el código de la vista.
- Permiten ver inmediatamente las ventanas con nuestros widgets y sin tener que escribir código.
- **¡Sí escala!**
- **XRC**: Formato estándar de wxWidgets para describir GUIs. Es un simple archivo XML.

- Desventajas

- No sirve en todos los casos: Formularios dinámicos.
- La carga del XML es un poco más lenta que si armamos la interfaz con código.



Boa Constructor



The screenshot displays the Boa Constructor Python IDE interface. The main window is titled "Boa Constructor - Python IDE - wxPython GUI Builder - Zope Editor". The interface includes a menu bar with options like "Nuevo", "Contenedores/Disposición", "Controles básicos", "Botones", "Controles de lista", "Librería", "Usuario", "Utilidades (Datos)", and "Zope". Below the menu bar is a toolbar with various icons for adding and managing controls.

The "Inspector" panel on the left shows the properties of the selected control, a wx.Button. The properties listed are:

Prop	Value
Class	wx.Button
Id	wxID_FRAME2BUTTON2
Label	u'button2'
Name	u'button2'
Position	wx.Point(248, 192)
Size	wx.Size(85, 29)
Style	0

The main editor window shows the code for the Frame2 class:

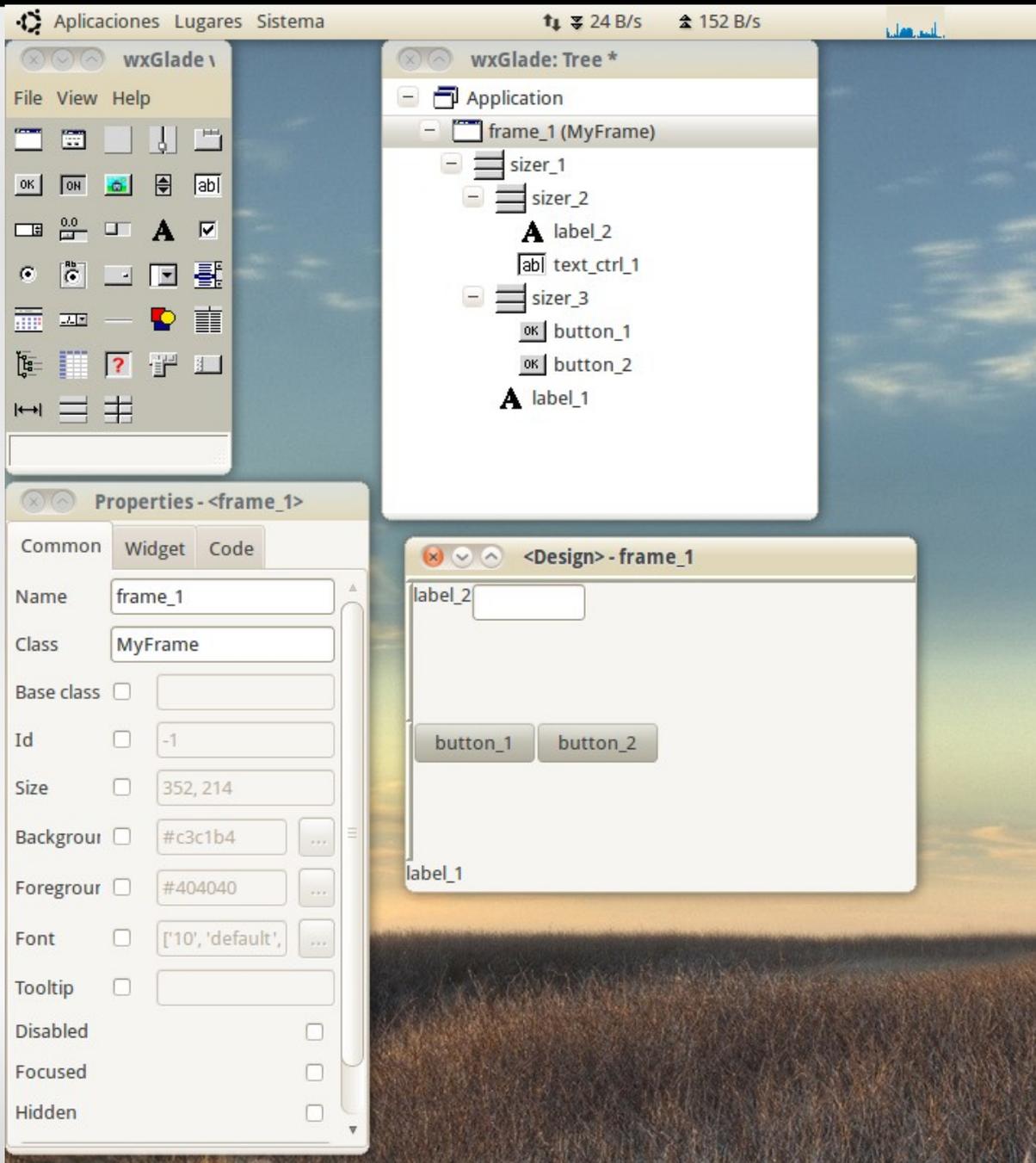
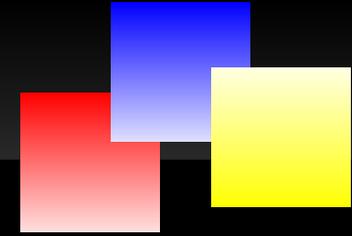
```
1 #Boa:Frame:Frame2
2
3 import wx
4
5 def create(parent):
6     return Frame2(parent)
7
8 [wxID_FRAME2] = [wx.NewId() for _init
9
10 class Frame2(wx.Frame):
11     def __init_ctrls(self, prnt):
12         wx.Frame.__init__(self, style=wx.DEFAULT_FRAME_STYLE, name='', parent=prnt, tit
13
14     def __init__(self, parent):
15         self._init_ctrls(parent)
16
```

The preview window, titled "Frame2", shows a graphical representation of the application. It contains a static text control labeled "staticText2" at the top. Below it are two text controls, "textCtrl1" and "textCtrl2". At the bottom, there are two buttons: "button1" and "button2". The "button2" control is highlighted with a dashed border, indicating it is the currently selected control.

Boa

Constructor

wxGlade

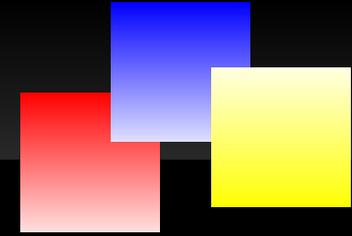


wxGlade

- Totalmente Visual
- Genera XRC y .py
- No soporta muchos de los controles actuales de wxPython
- Bastante rústico



XRCed



xrced: xrced.xrc
File Edit View Move Help

wxFrame
 wxBoxSizer
 wxStaticText
 wxTextCtrl
 wxGrid
 wxMenuBar
 wxStatusBar

class: wxFrame
name:

Attributes Look'n'Feel Style ExStyle Code

pos:
size:
title: Ventana de Ejemplo
 centered (default is OFF)

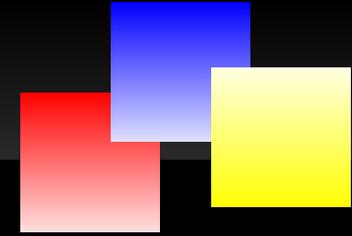
Ventana de Ejemplo
Titulo

Descargas
Documentos

Widget Catalog:

- Dialog: OK, Cancel, OK/Cancel, OK/Cancel/Apply, OK/Cancel/Apply/Help
- List: list, list book
- Page: page1
- Tree: tree book
- Choice: choice book
- StaticBox: StaticBox
- Table: Table with columns and rows
- Grid: Grid widget

wxFormBuilder



Aplicaciones Lugares Sistema 36 B/s 152 B/s EEUU sáb 8 de may, 02:00 marcelo

example - wxFormBuilder v3.1 - Beta

File Edit View Tools Help

Object Tree

- Example : Project
 - MainWindow : Frame
 - bSizer1 : wxBoxSizer
 - abc m_staticText1 : wxStaticText
 - abc m_textCtrl1 : wxTextCtrl
 - m_grid1 : wxGrid
 - m_menubar1 : wxMenuBar
 - m_statusBar1 : wxStatusBar

Component Palette

Common Additional Containers Menu/Toolbar Layout Forms

Editor

Ventana de Ejemplo

Titulo

	A	B	C	D	E
1	A-1	B-1	C-1	D-1	E-1
2	A-2	B-2	C-2	D-2	E-2
3	A-3	B-3	C-3	D-3	E-3
4	A-4	B-4	C-4	D-4	E-4
5	A-5	B-5	C-5	D-5	E-5

Object Properties

Properties Events

wxStaticText

name	m_staticText1
style	
label	Titulo
wrap	-1

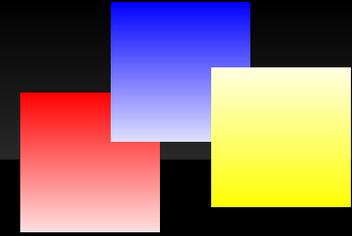
wxWindow

id	wxID_ANY
pos	-1; -1
size	-1; -1
Width	-1
Height	-1
minimum_size	-1; -1
maximum_size	-1; -1
font	-1; Default; ; Normal; N
fg	<input type="checkbox"/> Default
bg	<input type="checkbox"/> Default
window_name	
window_style	

Designer C++ XRC

Property Modified! /home/marcelo/Documentos/Charlas/wxPython/example.fbp Name: m_staticText1 | Class: wxStaticText

Ejemplo 5 - XRC



```
import wx
import wx.xrc as xrc

class EditorApp(wx.App):

    def OnInit(self):
        # Cargo el XRC y el frame principal
        self.res = xrc.XmlResource('example.xrc')
        self.frame = self.res.LoadFrame(None, 'MainWindow')

        # Obtengo la referencia al texto, útil
        self.txtTexto = xrc.XRCCTRL(self.frame, 'txtTexto')

        # Conecto los eventos ...
        self.frame.Bind(wx.EVT_MENU, self.OnSalir, id=xrc.XRCID('mnuSalir'))
        self.frame.Show()
        return True

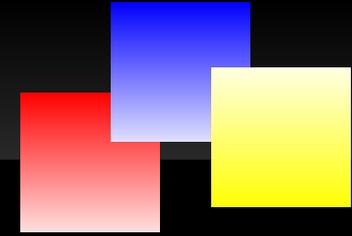
    # ... Defino las demás funciones / callbacks en la App

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

¡Con esta línea armé todo el form!



wxPython Demo

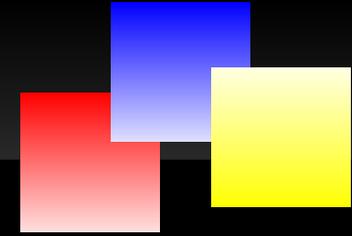


The screenshot shows a window titled "wxPython: (A Demonstration)" with a menu bar (File, Demo, Help) and a tabbed interface. The left sidebar lists various demo components, with "RibbonBar" selected. The main area displays a "Pure-Python RibbonBar" containing a "wxPython Ribbon Sample Application". This application features a ribbon with tabs: "Examples", "Appearance", "Empty Page", and "Another Page". The "Examples" tab is active, showing a toolbar, selection tools (Expand Vertically, Expand Horizontally, Contract), and shape tools (Circle, Triangle, Cross, Square, Other Polygon). A large "iDemo!" text is overlaid on the application window. At the bottom, a "Demo Log Messages" panel shows the following text:

```
OnAppActivate: False
OnActivate: True
OnActivate: False
```



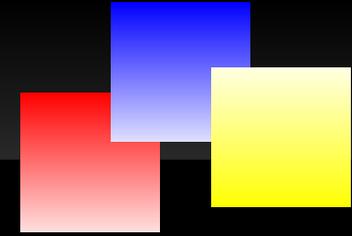
Y como si fuera poco...



- Visualización e impresión de HTML simple
- Print Framework, con vista previa y configuración
- Clipboard y drag and drop
- Ayuda en línea. Gran comunidad alrededor
- Librería de graficación de objetos vectoriales: OGL.
- Soporte para Cairo y OpenGL (GLCanvas)
- Texto enriquecido (RTF) y "estilizado" (STC)
- Animaciones y multimedia
- Programación multiproceso, Unicode, componentes personalizados, wx.AUI



Links



- Sitio oficial: <http://www.wxpython.org>
- Libro de referencia: wxPython in Action (Manning)
- Wiki Comunidad: <http://wiki.wxpython.org>
- Listas de correo:
 - wxPython-users
 - wx-users
 - PyAr - <http://www.python.com.ar> ;-)
- <http://www.zetcode.com/wxpython/>
- <http://pablotilli.com.ar/>

