



Introduzione a PyGTK

Giuliani Vito, Ivan

kratorius@lugbari.org

<http://zeta-puppis.com>

LinuxDay 2007





Programmazione delle GUI

- La creazione delle GUI é spesso un'operazione macchinosa
- Esistono diverse librerie grafiche che *semplificano* il lavoro:
 - GTK
 - wxWidgets
 - QT
 - ...





La libreria grafica GTK

- La libreria GTK¹ (the GIMP Toolkit) é una libreria per la creazione di interfacce grafiche.
- Caratteristiche:
 - Multiplatforma
 - Si basano su *eventi* e *segnali*
 - Esistono *bindings* per la maggior parte dei linguaggi di programmazione

¹<http://www.gtk.org>

Cos'è Glade

- Glade è un tool RAD (Rapid Application Development) per lo sviluppo di GUI
- Permette di creare le GUI *disegnanvole*
- Per alcuni linguaggi², genera anche automaticamente lo skeleton delle classi per la gestione della finestra

²il supporto per Python è ancora sperimentale



Come funziona

- Viene prodotto un file XML che é la descrizione dell'interfaccia
- Si *comunica* con il file prodotto attraverso la libreria **libglade**

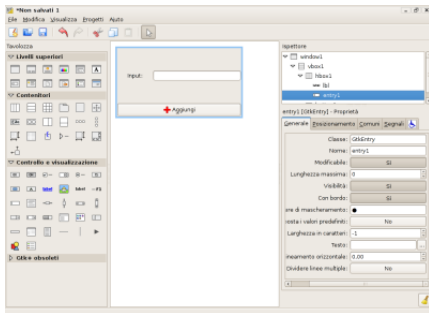


Figura: Uno screenshot di Glade 3



Quando usarlo

- su piccoli progetti

o meglio

- su GUI non troppo complicate



Come funzionano le GTK

- Le GTK sono basate sul concetto di **evento**.
- Questo significa che esiste un ciclo infinito che *cattura* gli eventi che avvengono all'interno della finestra

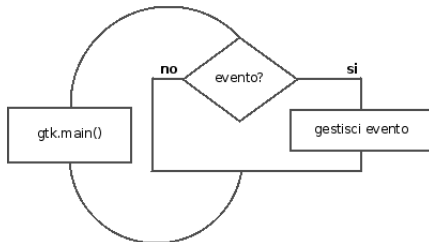


Figura: Schema dell'architettura ad eventi



Eventi e segnali

Esempi di eventi:

- Click
- Spostamento
- Movimento del mouse
- Pressione di un tasto
- ...



Eventi e segnali

- Non c'è bisogno di gestire manualmente tutti gli eventi perché per i più comuni ci sono comportamenti di default
- Nel momento in cui bisogna gestire un evento (es: click su un pulsante) si utilizzano i **segnali**.

quindi

- Un **segnale** viene emesso in base ad un azione di un utente





Funzioni di callback

- La gestione dei segnali si basa sul meccanismo delle funzioni di **callback**
- Si *connette* un **segnale** a una **funzione** che verrà chiamata nel momento in cui il segnale viene emesso
- Questa funzione é chiamata funzione di **callback**
- A seconda del segnale, verranno passati opportuni parametri



Cos'è un widget

- Ogni elemento delle GTK
- Possono essere combinati tra di loro per comporre altri widget

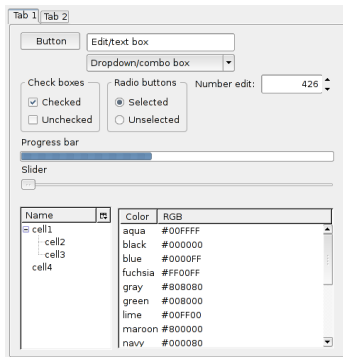


Figura: Vari widget in una finestra

gtk.Window

Il modello *base* di finestra

Costruttore:

`gtk.Window(type=gtk.WINDOW_TOPLEVEL)`

dove `type` può assumere i valori:

- `gtk.WINDOW_TOPLEVEL`
(il valore di default), una finestra *normale*
- `gtk.WINDOW_POPUP`,
una finestra speciale come un tooltip

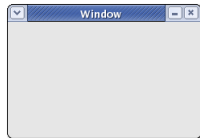


Figura:
Screenshot di
una
gtk.Window

gtk.Label

É un etichetta di testo

Costruttore: `gtk.Label(str=None)`

dove `str` é la stringa da mostrare



Figura:
Screenshot di
una `gtk.Label`

gtk.Button

Il pulsante generico

Costruttore: `gtk.Button(label=None, stock=None, use_underline=True)`

dove

- `label` é il testo da inserire nel pulsante
- `stock` é l'immagine predefinita da usare (se specificata anche il testo mostrato sar  quello predefinito, ignorando l'attributo `label`)
- e `use_underline` indica se bisogna interpretare `label` come un acceleratore

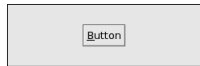


Figura:

Screenshot di un `gtk.Button`

gtk.Entry



Figura: Screenshot di un gtk.Entry

Una casella di testo

Costruttore: `gtk.Entry(max=0)`

dove `max` é la lunghezza massima di caratteri inseribili.

gtk.Frame



Figura: Screenshot di un gtk.Frame

Una cornice

Costruttore: `gtk.Frame(label=None)`
dove `label` é l'etichetta che verrà utilizzata.

I box

I box non sono altro che contenitori per widget. Principalmente ne esistono due:

- `gtk.HBox`: i widget inseriti all'interno verranno disposti orizzontalmente, da sinistra verso destra
- `gtk.VBox`: i widget inseriti all'interno verranno disposti verticalmente, dall'alto verso il basso



Cosa serve

Per avere una panoramica generale sui componenti e le funzioni base di PyGTK costruiremo un piccolo programma d'esempio. Gli strumenti necessari sono:

- Python: <http://www.python.org>
- PyGTK: <http://www.pygtk.org>

Il nostro programma di esempio sarà il *classico* “hello world”, arricchito da qualche funzione aggiuntiva.

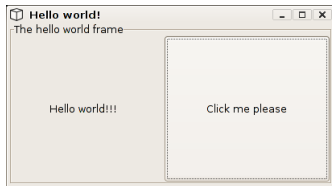


Figura: Il nostro “hello world”

Il preambolo

Ogni programma in Python, per utilizzare le librerie PyGTK, deve importare i moduli della libreria. Questo avviene con le seguenti righe:

```
import pygtk
import gtk
```



La classe principale

Nella creazione del programma, é stata utilizzata una classe chiamata `Main` che creerà la finestra e ne gestirá gli eventi/segnali.

Questa classe verrà istanziata non appena parte il programma:

```
def main():
    try:
        Main()
    except KeyboardInterrupt:
        print "Closed by CTRL+C"
        return True
    return False

if __name__ == '__main__':
    sys.exit(main())
```



L'istanziamento della classe

- Tutte le classi in Python possono disporre di un metodo `__init__()` che viene chiamato quando la classe viene istanziata.
- Il nostro `__init__()` si occuperà di chiamare le varie funzioni (metodi della classe) che *disegneranno* la finestra.
- Una volta che la finestra sarà stata *disegnata*, verrà invocato il metodo `main` della libreria GTK.



L'inizializzazione della classe

Nel momento in cui la classe viene istanziata, viene chiamato il metodo `__init__`:

```
class Main:
    def __init__(self):
        self.setupWindow()      # creiamo la finestra
        self.setupFrame()      # il frame
        self.setupContainer()  # l'hbox
        self.setupLabel()      # l'etichetta
        self.setupButton()     # il pulsante
        self.setupSignals()    # e i segnali

        self.wndMain.show_all()
        gtk.main()
```



L'inizializzazione della classe

Nel momento in cui la classe viene istanziata, viene chiamato il metodo `__init__`:

```
class Main:
    def __init__(self):
        self.setupWindow()      # creiamo la finestra
        self.setupFrame()      # il frame
        self.setupContainer()  # l'hbox
        self.setupLabel()      # l'etichetta
        self.setupButton()     # il pulsante
        self.setupSignals()    # e i segnali

        self.wndMain.show_all()
        gtk.main()
```



L'inizializzazione della classe

Nel momento in cui la classe viene istanziata, viene chiamato il metodo `__init__`:

```
class Main:
    def __init__(self):
        self.setupWindow()      # creiamo la finestra
        self.setupFrame()      # il frame
        self.setupContainer()  # l'hbox
        self.setupLabel()      # l'etichetta
        self.setupButton()     # il pulsante
        self.setupSignals()    # e i segnali

        self.wndMain.show_all()
        gtk.main()
```



L'inizializzazione della classe

Nel momento in cui la classe viene istanziata, viene chiamato il metodo `__init__`:

```
class Main:
    def __init__(self):
        self.setupWindow()      # creiamo la finestra
        self.setupFrame()      # il frame
        self.setupContainer()  # l'hbox
        self.setupLabel()      # l'etichetta
        self.setupButton()     # il pulsante
        self.setupSignals()    # e i segnali

        self.wndMain.show_all()
        gtk.main()
```





L'inizializzazione della classe

Nel momento in cui la classe viene istanziata, viene chiamato il metodo `__init__`:

```
class Main:
    def __init__(self):
        self.setupWindow()      # creiamo la finestra
        self.setupFrame()      # il frame
        self.setupContainer()  # l'hbox
        self.setupLabel()      # l'etichetta
        self.setupButton()     # il pulsante
        self.setupSignals()    # e i segnali

        self.wndMain.show_all()
        gtk.main()
```



L'inizializzazione della classe

Nel momento in cui la classe viene istanziata, viene chiamato il metodo `__init__`:

```
class Main:
    def __init__(self):
        self.setupWindow()      # creiamo la finestra
        self.setupFrame()       # il frame
        self.setupContainer()   # l'hbox
        self.setupLabel()       # l'etichetta
        self.setupButton()      # il pulsante
        self.setupSignals()     # e i segnali

        self.wndMain.show_all()
        gtk.main()
```





L'inizializzazione della classe

Nel momento in cui la classe viene istanziata, viene chiamato il metodo `__init__`:

```
class Main:
    def __init__(self):
        self.setupWindow()      # creiamo la finestra
        self.setupFrame()       # il frame
        self.setupContainer()   # l'hbox
        self.setupLabel()       # l'etichetta
        self.setupButton()      # il pulsante
        self.setupSignals()     # e i segnali

self.wndMain.show_all()
gtk.main()
```



L'inizializzazione della classe

Nel momento in cui la classe viene istanziata, viene chiamato il metodo `__init__`:

```
class Main:
    def __init__(self):
        self.setupWindow()      # creiamo la finestra
        self.setupFrame()       # il frame
        self.setupContainer()   # l'hbox
        self.setupLabel()       # l'etichetta
        self.setupButton()      # il pulsante
        self.setupSignals()     # e i segnali

    self.wndMain.show_all()
    gtk.main()
```



Creiamo la finestra

```
def setupWindow(self):  
    self.wndMain = gtk.Window()  
    self.wndMain.set_default_size(400, 200)  
    self.wndMain.set_title>Hello world!)  
    self.wndMain.set_position(gtk.WIN_POS_CENTER)
```

Con queste istruzioni:

- Creiamo la finestra
- Impostiamo la grandezza
- Impostiamo il titolo
- Impostiamo la posizione sullo schermo



Creiamo la finestra

```
def setupWindow(self):  
    self.wndMain = gtk.Window()  
    self.wndMain.set_default_size(400, 200)  
    self.wndMain.set_title>Hello world!)  
    self.wndMain.set_position(gtk.WIN_POS_CENTER)
```

Con queste istruzioni:

- Creiamo la finestra
- Impostiamo la grandezza
- Impostiamo il titolo
- Impostiamo la posizione sullo schermo

Creiamo la finestra

```
def setupWindow(self):  
    self.wndMain = gtk.Window()  
    self.wndMain.set_default_size(400, 200)  
    self.wndMain.set_title>Hello world!)  
    self.wndMain.set_position(gtk.WIN_POS_CENTER)
```

Con queste istruzioni:

- Creiamo la finestra
- Impostiamo la grandezza
- Impostiamo il titolo
- Impostiamo la posizione sullo schermo



Creiamo la finestra

```
def setupWindow(self):  
    self.wndMain = gtk.Window()  
    self.wndMain.set_default_size(400, 200)  
    self.wndMain.set_title>Hello world!)  
    self.wndMain.set_position(gtk.WIN_POS_CENTER)
```

Con queste istruzioni:

- Creiamo la finestra
- Impostiamo la grandezza
- Impostiamo il titolo
- Impostiamo la posizione sullo schermo



Creiamo il frame

```
def setupFrame(self):  
    self.frame = gtk.Frame("The hello world frame")  
    self.wndMain.add(self.frame)
```

In questo modo creiamo il **frame** e lo aggiungiamo alla **finestra principale**.



Creiamo l'HBox

```
def setupContainer(self):  
    self.hbox = gtk.HBox(spacing=10)  
    self.frame.add(self.hbox)
```

In questo modo creiamo l'**HBox** e lo aggiungiamo all'interno dell'**frame**.



Creiamo l'etichetta

```
def setupLabel(self):  
    self.label = gtk.Label("Hello world!!!")  
    self.hbox.add(self.label)
```

In questo modo creiamo l'**etichetta** e la aggiungiamo come *primo* elemento all'interno dell'**HBox**.



Creiamo il pulsante

```
def setupButton(self):  
    self.button = gtk.Button("Click me please")  
    self.hbox.add(self.button)
```

In questo modo creiamo il **pulsante** e lo aggiungiamo come *secondo* elemento all'interno dell'**HBox**.

Gestiamo i segnali

A questo punto bisogna *connettere* (o *collegare*) i segnali alle relativi funzioni di callback:

```
def setupSignals(self):  
    self.wndMain.connect('destroy', self.closeWindow)  
    self.button.connect('clicked', self.buttonClicked)
```



I gestori dei segnali

```
def closeWindow(self, wnd):  
    gtk.main_quit()
```

```
def buttonClicked(self, button):  
    self.label.set_text("The button has been clicked!")
```

Mentre l'evento `destroy` chiamerà la funzione `closeWindow` che non fa altro che chiudere l'applicazione chiamando `gtk.main_quit()`, l'evento `clicked` invocando `buttonClicked` cambierà il testo all'interno della label precedentemente inserita.



Riferimenti

- **Sito ufficiale di Python**
<http://www.python.org>
- **Sito ufficiale della libreria PyGTK**
<http://pygtk.org>
- **Sito ufficiale delle librerie GTK**
<http://gtk.org>



Licenza

Queste slide sono rilasciate con licenza GFDL³

³<http://www.gnu.org/copyleft/fdl.html>