Glade

hitokiri

Nisan, 2016

İçindekiler

1 Glade Nedir?

Glade, Gnome Masaüstü ortamına, GTK+ araç kiti için hızlı ve kolay arayüzler sağlamak üzere geliştirlmiş bir RAD aracıdır.

Glade ile tasarlanmış kullanıcı arayüzleri XML biçeminde kaydedilir ve GtkBuilder nesnesi GTK arayüzü olarak biçemin dinamikleşmesini sağlar.

GtkBuilder kullanarak Glade dosyalarını kullanan sayısız programlama dili vardır. Başta C olmak üzere C++, C#, Vala, Perl, Python, Java…

Glade, özgür bir yazılımdır ve GNU/GPL ile lisanslanmıştır.

2 Gtk Builder?

Glade tasarımcısıyla arayüzler oluşturmak şöyle bir kenarda dursun, peki ama bu dosyalar programda nasıl yer alır? Bunun için öncelikle dikkat etmemiz gereken GtkBuilder'dır.
GtkBuilder, kaydedilen Glade .xml arayüz dosyalarının, programcı tarafından widget, window, dialog gibi gtk araçları olarak kullanılmasına imkân verir.

3 GtkBuilder İşlevleri

- add_from_file
UI tanımını içeren dosyayı ayrıştırır.

- add_from_string
UI tanımını içeren karakter grubunu ayrıştırır.

- add_objects_from_file
UI tanımını içeren dosya içinden belirtilen gtk araçlarını ayrıştırır.

- add_objects_from_string
UI tanımını içeren karakter dizesi içinden belirtilen gtk araçlarını ayrıştırır.

- get_object
Ayrıştırılmış dosya veya karakter dizesi içinden ismi tanımlanan gtk aracını verir.

- get_objects
Ayrıştırılmış dosya veya karakter dizesi içinde bulunan tüm gtk araçlarının bir liste olarak bilgisini verir.

_connect_signals
Adlandırılmış sinyallerin gtk aracının yeniden tanımlanmasına gerek duyulmadan uygulama tarafından erişilmesini sağlar.

Başlıca GtkBuilder işlevleri bunlardır. Tüm diğer seçenekler için gerekli bilgiye şuradan ulaşabilisiniz: \
http://www.pygtk.org/docs/pygtk/class-gtkbuilder.html

4 ÖRNEKLER

Glade ile oluşturulan basit bir pencere Python ile şu şekilde kullanılabilir:

```
$ python
>>> import gtk
>>> builder = gtk.Builder()
# add_from_string özelliğini kullanabilmek için uygulamamız için bir
# UI tanımı oluşturuyoruz.

>>> buffer = """<interface>
...   <object class="GtkWindow" id="window1">
...     <property name="can_focus">False</property>
...     <property name="has_resize_grip">False</property>
...     <child>
...       <object class="GtkButton" id="button1">
...         <property name="label" translatable="yes">button</property>
...         <property name="visible">True</property>
...         <property name="can_focus">True</property>
...         <property name="receives_default">True</property>
...         <property name="use_action_appearance">False</property>
...       </object>
...     </child>
...   </object>
... </interface>"""

#Oluşturduğumuz UI tanımını Builder'e ekliyoruz.

>>> builder.add_from_string(buffer)
>>> win = builder.get_object("window1")
>>> win.connect("delete-event",gtk.main_quit)
>>> win.show_all()
```

Şimdi gelelim Glade'i kullanarak yapıcağımız örnek uygulamacığımıza. Bu bakımdan kamu yararını göz önünde tutarak şöyle boy-kilo indeksini hesaplayan bir uygulamacık tasarlayacağız.

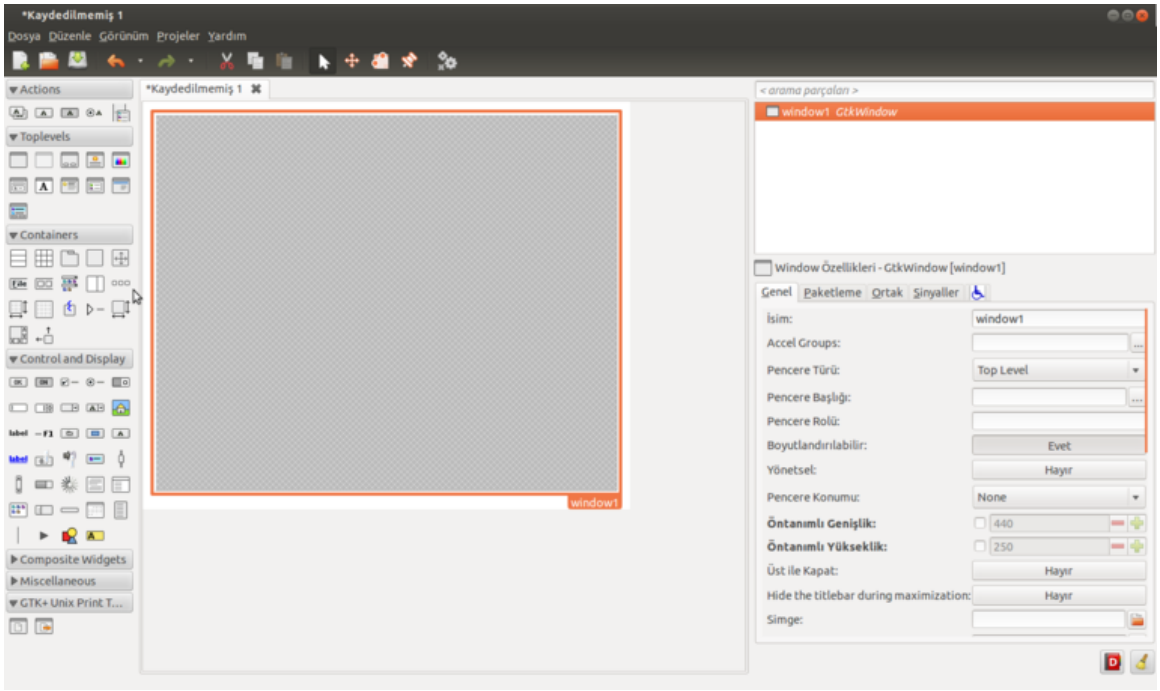
Öncelikle Glade'yi açalım. Bunun için Alt+F2'ye basıp "glade" yazıyoruz.
İşte karşımızda:

"Container" ana başlığı altından bir kutucuk eklemek ile işe başladık. Gtk araçlarımızı yerleştirdikten sonra şimdi gelelim sinyal yönetimine. Sinyalleri Glade üzerindeyken adlandırabilir ve daha sonrası için uygulamacığımızdan _connect_signals ile ayrrıştırarak gtk aracını yeniden tanımlamadan kullanabiliriz.

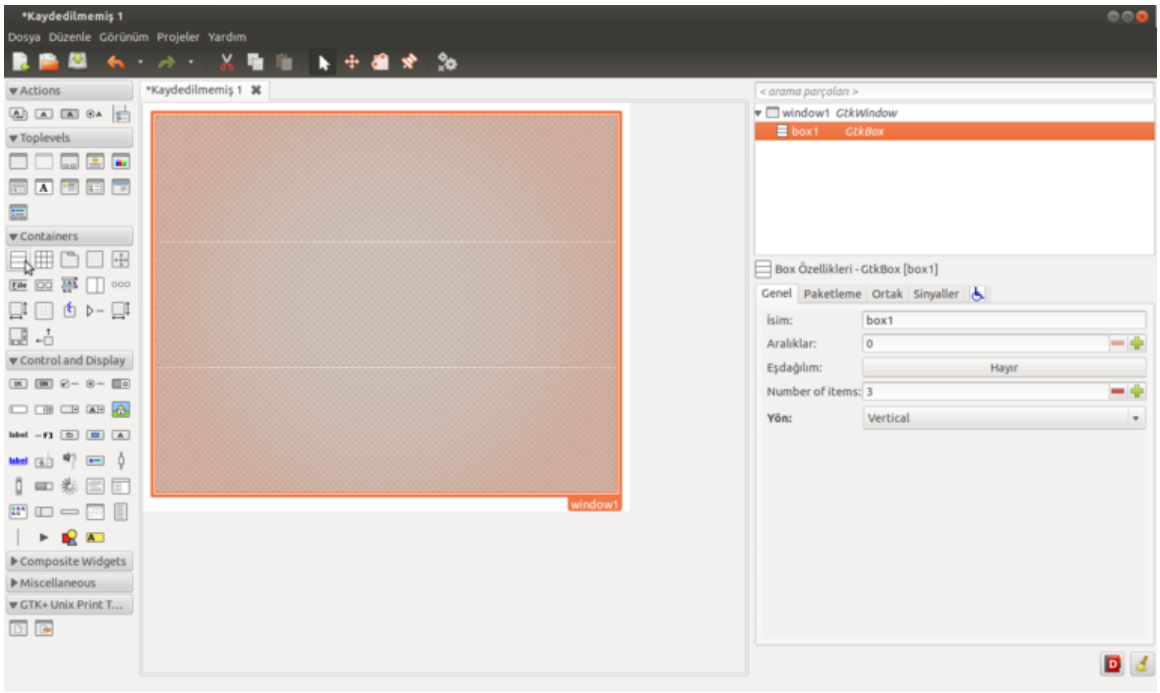
Şimdi gelelim Python betiğimizin içeriğine. Öncelikle gtk kütüphanelerini yükleyelim. 12.04'ün ruhuna uygun olarak Glade, gtk3'ün yapılandırmalarıyla geliyor. Bu yüzden Python için gi deposu üzerinden gtk kütüphanelerini yüklemek sorun yaşamamamızı sağlayacaktır.

```
#!/usr/bin/env python
# -*- coding: utf-8 -*-
# vim: ts=4:sw=4
from gi.repository import Gtk as gtk
#oran adlı bir sınıf oluşturalım ve __init__ değer olarak Builder aracını tanımlıyalım:
class oran(gtk.Builder):
    def __init__(self):
        gtk.Builder.__init__(self)
        #Şimdi self haznesi üzerinden Builder aracını rahatlıkla kullanabiliriz.
        #Dosyamızı _add_from_file ile Builder'e ekleyelim.
        self.add_from_file("./boykilo.glade")
        # Hesap işlemleri için gerek duyduğumuz gtk araçlarını
        # uygulamamıza Builder üzerinden davet edelim.

        self.label = self.get_object("label1")
        self.boy = self.get_object("scale1")
        self.kilo = self.get_object("scale2")
```



Şekil 1:



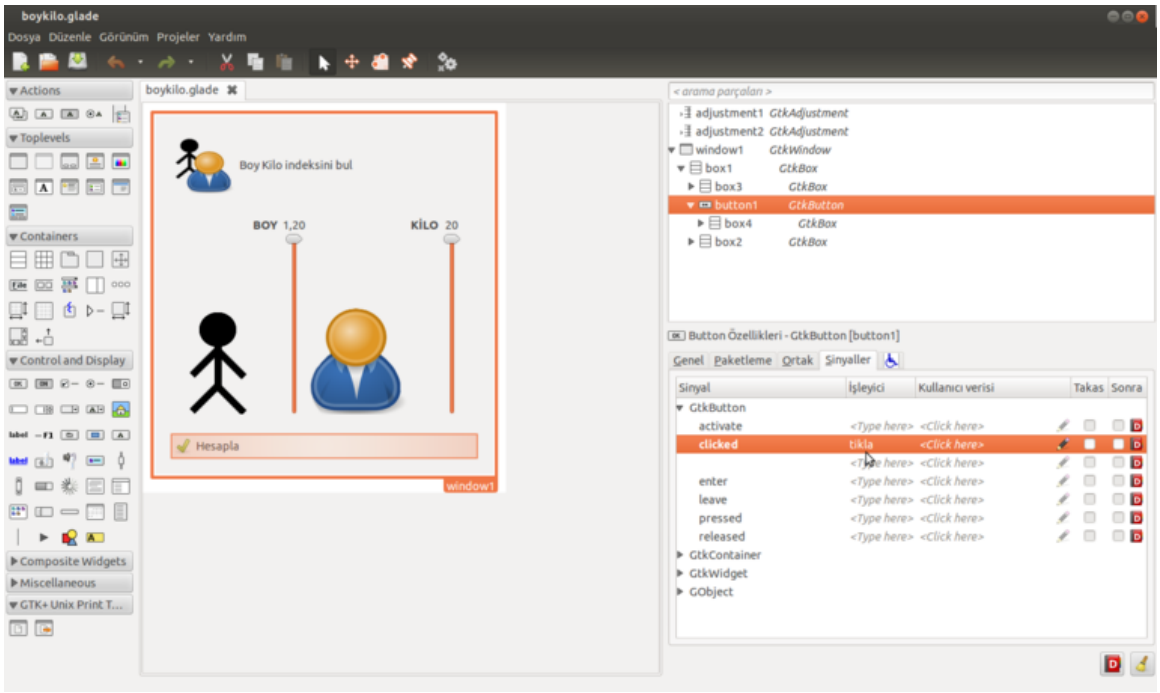
Şekil 2:

```
18          #Şimdiyse adlandırdığımız sinyal isimlerine fonksiyonlarımızın hangileri
19          #olduğu connect_signals ile belirtelim.
20          self.connect_signals({ "tikla":self.islem ,
21                                 "kapat" : gtk.main_quit })
22          #Pencerimiz gösterilmeye hazır.
23          self.get_object("window1").show_all()
24          #Kabaca basit bir işlemle şimdi hesaplamaya geldi sıra.
25      def islem(self,data):
26          # Tanımladığımız gtk araçlarında gerekli boy ve kilo bilgisini
27          # aldıktan sonra bilgilendirme hazır artık.
28          boy = self.boy.get_value()
29          kilo = self.kilo.get_value()
30          x = boy * boy
31          sonuc = float(kilo)/float(x)
32          if sonuc > 25:
33              while True:
34                  kilo -= 1.0
35                  xx = float(kilo)/float(x)
36                  if xx <= 25:
37                      ideal = kilo
38                      ver = self.kilo.get_value() - ideal
39                      break
40              self.label.set_markup(\
41  """Boy - Kilo oranınız <span foreground="red" ><b><big>%s</big></b></span> ..
42  Sağlıklı kilo sınırınız olan <span foreground="red" ><b><big>25</big></b></span>'i geçmiş
        durumda.
43  Sağlıklı bir yaşam için olmanız gereken kilo<span foreground="red" ><b><big>%s</big></b></
        span>,
44  Vermeniz gereken kilo ise yaklaşık <span foreground="red" ><b><big>%s</big></b></span> kadar
        .."""
45              % ( str(sonuc)[0:4]  , int(ideal) , int(ver) ) )
46          else:
47              self.label.set_markup(\
48  """Boy - Kilo oranınız <span foreground="#009021" > <b><big>%s</big></b></span> ..
49  Sağlıklı kilo sınırınız olan <span foreground="#009021"><b><big>25</big></b></span>'in
        altında .."""
50              % ( int(sonuc) ) )
51  oran()
52  gtk.main()
```

Python ve C için örneklere ekten ulaşabilirsiniz..

```
1  gcc mixer.c -o mixer `pkg-config --libs --cflags gtk+-2.0`
2  gcc mixer.c -o mixer `pkg-config --libs --cflags gtk+-3.0`
```

komutları ile c için derleme işlemini gerçekleştirebilirsiniz.


boykilo.glade
Dosya Düzenle Görünüm Projeler Yardım
Actions
Toplevels
Containers
Control and Display
Composite Widgets
Miscellaneous
GTK+ Unix Print T...
boykilo.glade
Boy Kilo indeksini bul
BOY 1,20
KİLO 20
Hesapla
window1
< arama parçaları >
adjustment1 GtkAdjustment
adjustment2 GtkAdjustment
window1 GtkWindow
box1 GtkBox
box3 GtkBox
button1 GtkButton
box4 GtkBox
box2 GtkBox
Button Özellikleri - GtkButton [button1]
Genel Paketleme Ortak Sinyaller
Sinyal
İşleyici
Kullanıcı verisi
Takas
Sonra
GtkButton
activate
clicked
tikla
enter
leave
pressed
released
GtkContainer
GtkWidget
GObject

Şekil 3: