

Glade

hitokiri

Eylül, 2012

İçindekiler

1	Glade Nedir?	2
2	Gtk Builder?	3
3	GtkBuilder İşlevleri	4
4	ÖRNEKLER	5

1 Glade Nedir?

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

Glade ile tasarlanmış kullanıcı arayüzleri XML biçiminde kaydedilir ve GtkBuilder nesnesi GTK arayüzü olarak biçimin 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 ulaşabilirsiniz: \n<http://www.pygtk.org/docs/pygtk/class-gtkbuilder.html>

4 ÖRNEKLER

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

```
1 $ python
2 >>> import gtk
3 >>> builder = gtk.Builder()
4 # add_from_string özelliğini kullanabilmek için uygulamamız için bir
5 # UI tanımı oluşturuyoruz.
6
7 >>> buffer = """<interface>
8 ...   <object class="GtkWindow" id="window1">
9 ...     <property name="can_focus">False</property>
10 ...    <property name="has_resize_grip">False</property>
11 ...    <child>
12 ...      <object class="GtkButton" id="button1">
13 ...        <property name="label" translatable="yes">button</property>
14 ...        <property name="visible">True</property>
15 ...        <property name="can_focus">True</property>
16 ...        <property name="receives_default">True</property>
17 ...        <property name="use_action_appearance">False</property>
18 ...      </object>
19 ...    </child>
20 ...  </object>
21 ... </interface>"""
22
23 #Oluşturduğumuz UI tanımını Builder'e ekliyoruz.
24
25 >>> builder.add_from_string(buffer)
26 >>> win = builder.get_object("window1")
27 >>> win.connect("delete-event", gtk.main_quit)
28 >>> win.show_all()
```

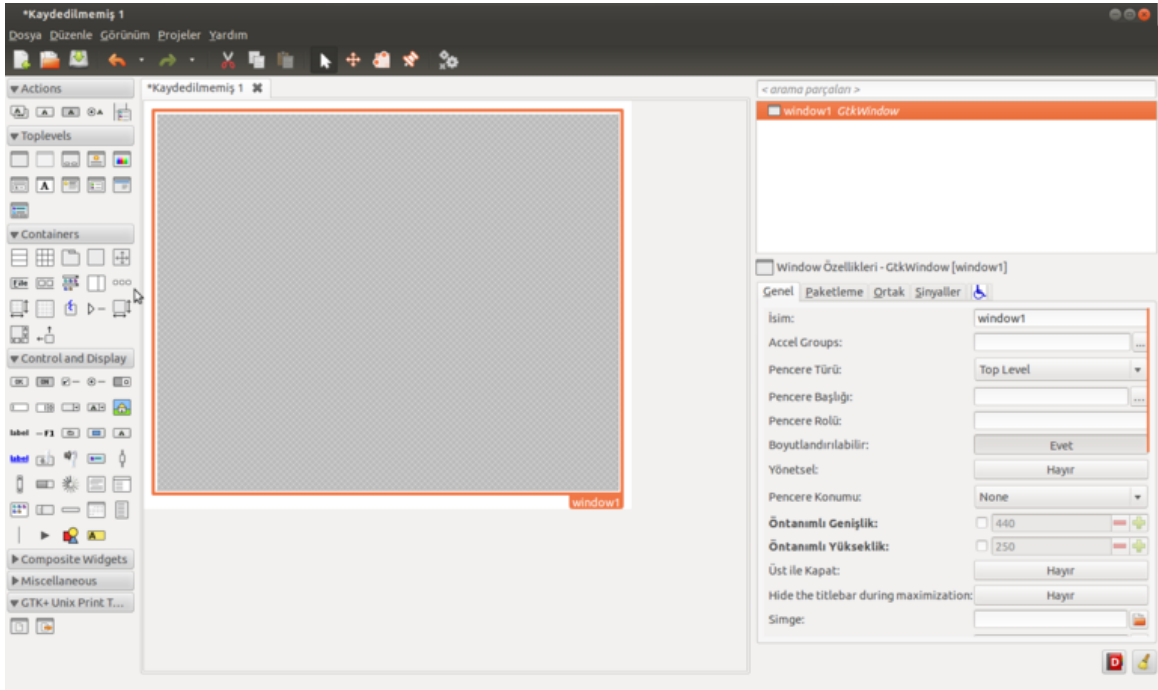
Şimdi gelelim Glade'i kullanarak yapacağımız örnek uygulamamıza. Bu bakımdan kamu yararını göz önünde tutarak şöyle boy-kilo indeksini hesaplayan bir uygulamacık tasarlayacağız.

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

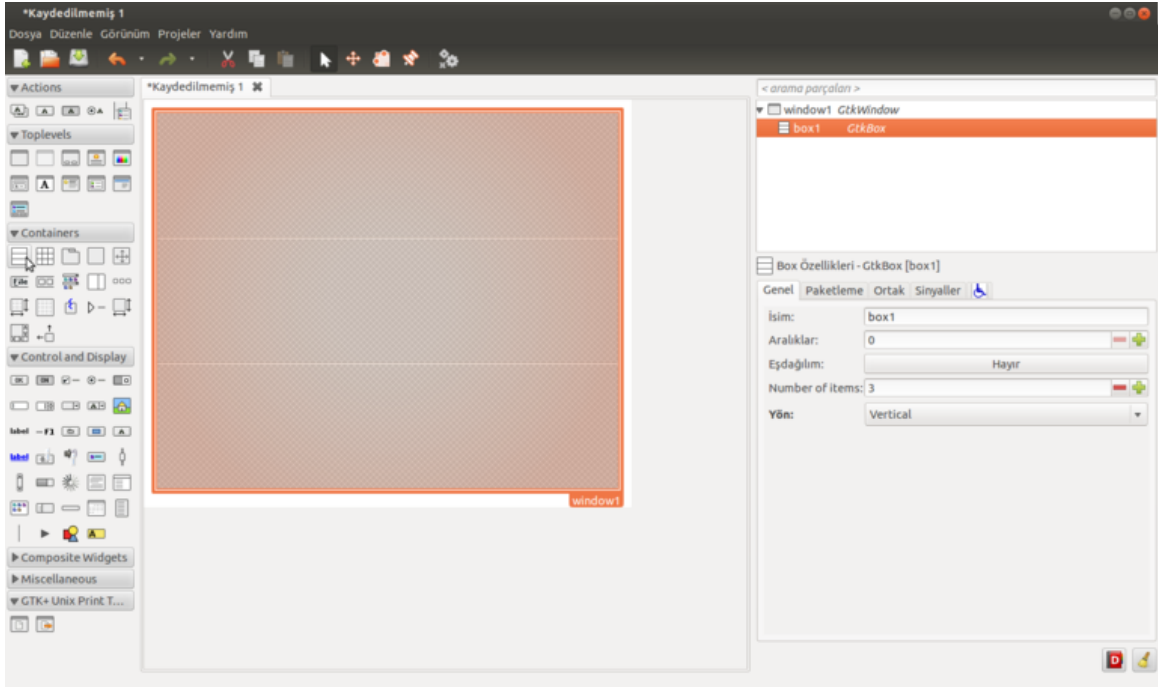
"Container" ana başlığı altından bir kutucuk eklemek ile işe başladık. Gtk araçlarımızı yerleştirdikten sonra şimdi gelelim sinyal yönetimine. Sinyalleri Glade üzerindeyken adlandırabilir ve daha sonrası için uygulamamızdan `_connect_signals` ile ayırıştırarak gtk aracını yeniden tanımlamadan kullanabiliriz.

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

```
1 #!/usr/bin/env python
2 # -*- coding: utf-8 -*-
3 # vim: ts=4:sw=4
4 from gi.repository import Gtk as gtk
5 #oran adlı bir sınıf oluşturulmuş ve __init__ değer olarak Builder aracını tanımlıyalım:
6 class oran(gtk.Builder):
7     def __init__(self):
8         gtk.Builder.__init__(self)
9         #Şimdi self haznesi üzerinden Builder aracını rahatlıkla kullanabiliriz.
10        #Dosyamızı _add_from_file ile Builder'e ekleyelim.
11        self.add_from_file("./boykilo.glade")
12        # Hesap işlemleri için gerek duyduğumuz gtk araçlarını
13        # uygulamamıza Builder üzerinden davet edelim.
14
15        self.label = self.get_object("label1")
16        self.boy = self.get_object("scale1")
17        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
23     #Pencerimiz gösterilmeye hazır.
24     self.get_object("window1").show_all()
25     #Kabaca basit bir işlemle şimdi hesaplamaya geldi sıra.
26     def islem(self,data):
27         # Tanımladığımız gtk araçlarında gerekli boy ve kilo bilgisini
28         # aldıktan sonra bilgilendirme hazır artık.
29         boy = self.boy.get_value()
30         kilo = self.kilo.get_value()
31         x = boy * boy
32         sonuc = float(kilo)/float(x)
33         if sonuc > 25:
34             while True:
35                 kilo -= 1.0
36                 xx = float(kilo)/float(x)
37                 if xx <= 25:
38                     ideal = kilo
39                     ver = self.kilo.get_value() - ideal
40                     break
41                 self.label.set_markup(\
42 """Boy - Kilo oranınız <span foreground="red" ><b><big>%s</big></b></span> ..
43 Sağlıklı kilo sınırınız olan <span foreground="red" ><b><big>25</big></b></span>'i geçmiş durumda.
44 Sağlıklı bir yaşam için olmanız gereken kilo <span foreground="red" ><b><big>%s</big></b></span>,
45 Vermeniz gereken kilo ise yaklaşık <span foreground="red" ><b><big>%s</big></b></span> kadar.. """)
46                 % ( str(sonuc)[0:4] , int(ideal) , int(ver) ) )
47         else:
48             self.label.set_markup(\
49 """Boy - Kilo oranınız <span foreground="#009021" ><b><big>%s</big></b></span> ..
50 Sağlıklı kilo sınırınız olan <span foreground="#009021"><b><big>25</big></b></span>'in altında .. """)
51                 % ( int(sonuc) ) )
52     oran()
53     gtk.main()

```

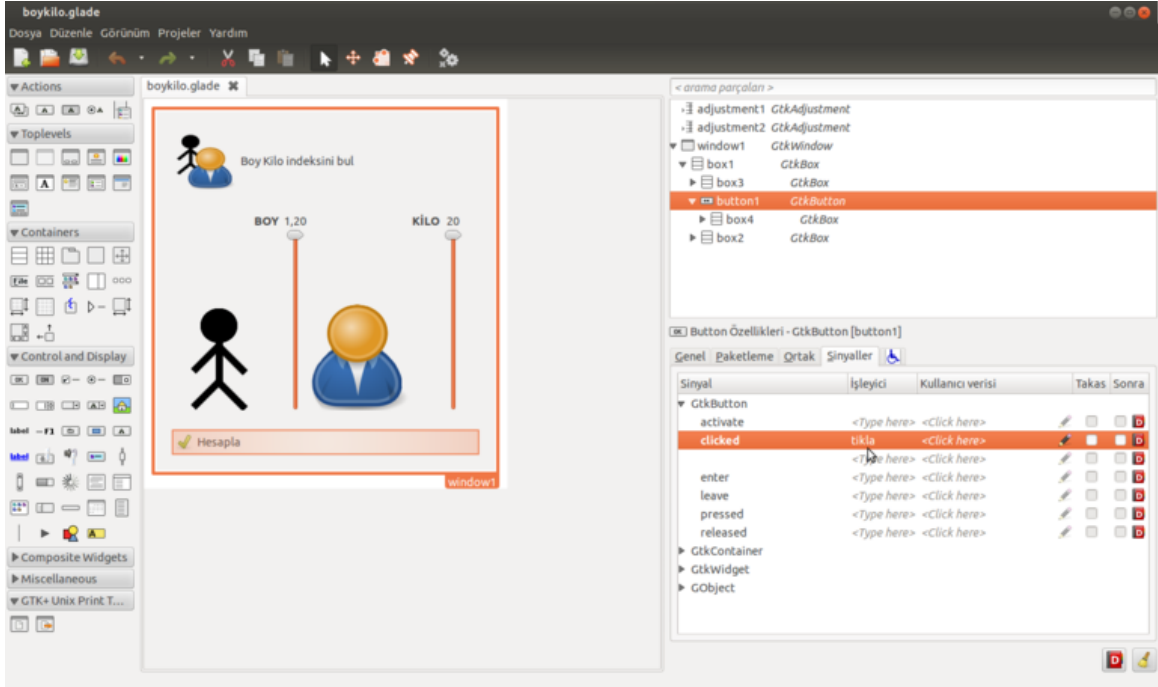
Python ve C için örneklere ekten ulaşabilirsiniz..

```

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.



Şekil 3: