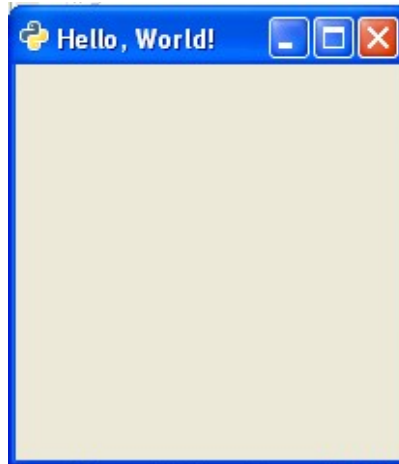


# PyGTK

هنتاول فى ال tutorial دى pygtk



عايزين نعمل window مشابه لى ونخليها متسنتره (فى منتصف الشاشة) اول ماتنشئ  
1- استدعى ال gtk

```
import gtk
```

2- انشئ class يورث ال gtk

```
class Window(gtk.Window):  
  
    def __init__(self):  
        super(Window, self).__init__(gtk.WINDOW_TOPLEVEL)  
        self.__init_comp()  
  
        self.show_all()
```

3- هنا بنقول ان ال window دى TOPLEVEL مش POPUP  
بنستدعى ال \_\_init\_comp طريقة لإنشاء الواجهة

.show\_all  
الطريقة show\_all بتعرض كل ال components داخل ال Window وهنا مش فى غيرها بس اتعود تستخدمها  
لأنك هتبقه تحط ويدجنس كثير جواها

```
.init_comp  
  
    def __init_comp(self):  
        self.set_title("Hello, World!")  
        self.set_position(gtk.WIN_POS_CENTER)
```

.set\_title(new\_title)

بتستخدم فى تغيير ال title على النافذة

.set\_position(pos)

بتستخدم هنا لتحديد ال مكان الخاص بالنافذة  
ولها عدة قيم زى

gtk.WIN\_POS\_CENTER

بتسنتر النافذة عند انشاءها

gtk.WIN\_POS\_CENTER\_ALWAYS

هيتم سنترتها عند اى تغيير فى ال size

gtk.WIN\_POS\_MOUSE

هيتم اظهار النافذة عند مكان الماوس الحالى

```
.set_size_request(h,w)
```

لتحديد ارتفاع وعرض النافذة

```
if __name__=="__main__":  
    w=Window()  
    gtk.main()
```

وبس كدا



هنا عندنا نافذة وفيها button واحد مكتوب عليه click me طيب جميل

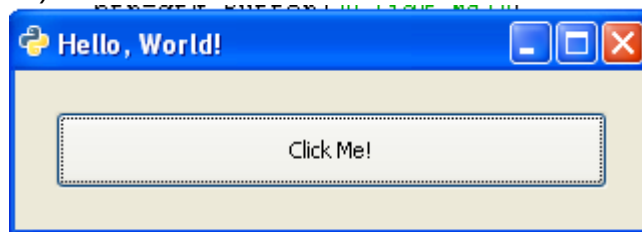
```
class Window(gtk.Window):  
  
    def __init__(self):  
        super(Window, self).__init__(gtk.WINDOW_TOPLEVEL)  
        self.__init_comp()  
  
        self.show_all()  
  
    def __init_comp(self):  
        self.set_title("Hello, World!")  
        self.set_border_width(20)  
        self.set_position(gtk.WIN_POS_CENTER)  
        btn=gtk.Button("Click Me!")  
        self.add(btn)
```

التعريف العام لل button

```
button = gtk.Button(label=None, stock=None)
```

وبس كدا  
تقدر تتحكم فى ال border\_width بإستخدام

```
.set_border_width(width)
```



عايزين بقة يظهر مسج لطيفة كدا اول مانضغط على ال button دا

```
def __on_btn_clicked(self, widget, data):  
    md=gtk.MessageDialog(self, gtk.DIALOG_DESTROY_WITH_PARENT,  
gtk.MESSAGE_INFO, gtk.BUTTONS_OK, "Hi!")  
    print widget  
    print data  
    md.run()  
    md.destroy()
```

دى اسمها callback يعنى طريقة هيتم تنفيذها عند حدوث شئ معين زى الضغط على  
MessageDialog نيجى لل  
اول معامل هو ال parent  
تانى معامل فيه خصائص الديالوج

gtk.DIALOG\_MODAL

لو اه فهو اللي هيصطاد اى ايفنت يحصل من الكيبورد (يمنع الوصول لل نافذة الأصلية إلا بعد انتهائه)

gtk.DIALOG\_DESTROY\_WITH\_PARENT

هيتقفل فى حال قفل ال parent

gtk.DIALOG\_NO\_SEPARATOR

مش هيطهر خط فاصل بين الرسالة وال buttons بتوع الرسالة

تالت معامل هو نوع المسج

هل معلومة او تحذير او سؤال او خطأ

gtk.MESSAGE\_INFO

معلومة

gtk.MESSAGE\_WARNING

تحذير

gtk.MESSAGE\_QUESTION

سؤال

gtk.MESSAGE\_ERROR

خطأ

ال button

تمام كدا بس انا شغلت الكود ومش فى حاجة حصلت (:  
فعلا لأننا لسه مش ربطنا ال callback بال signal  
بكل بساطة اكتب التالى

```
btn.connect("clicked", self.__on_btn_clicked, None)
```



وبس كدا طبعا انت مدايق من None وابه المتغيرات اللي معرفها انا فى ال callback دى أصلا ؟ ايه widget, ?? data

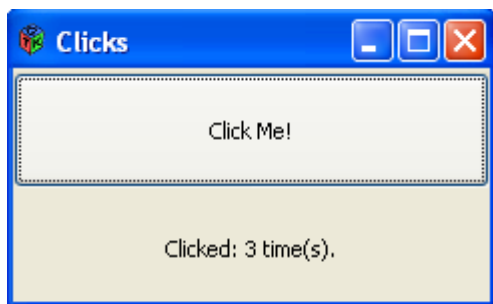
طيب تمام جدا قبل ماتسألنى السؤال دا تقدر تعمل حاجة حلوة قوى بإنك تجرب print على widget وعلى data فى ال callback

<gtk.Button object (GtkButton) at 0xb803f0>

None

ال widget دا ال button فى مثالنا  
ال data هى ال None تقدر طبعاً تستبدلها بأى حاجة المهم انها تكون شئ هيفيدك.. هنتعرف  
عليها اكثر لاحقاً...

## Clicks



لاحظ ان النافذة متقسمة لجزئين راسي اول جزء فيه button والثانى فيه label بيتكتب عليه عدد  
مرات الضغط على ال button

احنا ممكن نكتبها بالطريقة المعتادة وممكن نكتبها بطريقة كثير افضل باستخدام ال OOP  
تمام الأول عشان نخليهم متقسمين فى شكل معين افقى او رأسى بنستخدم Hbox (اختصار ل  
Horizontal Box) او Vbox (اختصار ل Vertical Box) تمام ؟ قشطة



ال vertical box بياخد ال widgets او بوكسز تانية فى صورة rows صفوف

ال horizontal box بياخد ال widgets او البوكسز التانى فى صورة Columns عواميد



لإنشاء box بتبدأ اولاً ب homogeneous ودى معناها هل كل الأجزاء متساوية فى العرض او الطول  
ليها ترجمة بإسم متجانسة اعتقد مناسبة ؟ والمعامل التانى لتحديد عرض الفاصل  
vbox=gtk.VBox(False, 2)

:packing

pack\_start(child, expand, fill, padding)

لوضع ال widget من الشمال لليمين او من فوق لتحت "صورة فطرية!"

pack\_end(child, expand, fill, padding)

لوضع ال widget من اليمين للشمال او من تحت لفوق وهى موجود لل Hbox وال Vbox

expand: هل عايزه يكبر مع اى زيادة فى حجم النافذة؟  
 fill: فى حال التصغير هل يتم اخفاء جزء منه؟ وليس تصغيره  
 padding: الهامش حوله

-استخدام الجداول

|   | 0 | 1 | 2 |
|---|---|---|---|
| 0 | + | + | + |
|   |   |   |   |
| 1 | + | + | + |
|   |   |   |   |
| 2 | + | + | + |

لإنشاء جدول بنشئته كالتالى

`gtk.Table( rows, columns, homogeneous )`

عدد الصفوف وعدد الأعمدة وهل متجانسين او لأ

لإضافة child باستخدام ال attach method

`.attach( child, left_attach, right_attach, top_attach,  
 bottom_attach, xoptions, yoptions, xpadding, ypadding)`

child: هو اللى هيثم اضافته فى الجدول

left\_attach: العمود على يسار المكان

right\_attach: العمود على يمين المكان

top\_attach: الصف فوق المكان

bottom\_attach: الصف تحت المكان

مثال للتوضيح

|   | 0 | 1                    | 2 |
|---|---|----------------------|---|
| 0 | + | +                    | + |
|   |   |                      |   |
| 1 | + | +                    | + |
|   |   | ████████████████████ |   |
| 2 | + | +                    | + |

لو عايزين نخط widget معين فى الكورنر اليمين من جدول زى مانت شايف 2X2

بيقع بين الخطين الرأسين 1 و 2 وهما دول ال left\_attach, right\_attach

بيقع بين الخطين الأفقيين 1 و 2 وهما دول ال top\_attach, bottom\_attach

xoptions: الرولز ل x

gtk.FILL: لو الجدول اكبر من الويدجت فالويدجت هيثمدد ليشغل المساحة

gtk.EXPAND: هنا الجدول هيثمدد اذا كان فى مساحة فى ال window

gtk.SHRINK: اذا تم تصغير المساحة المتاحة للويدجت "مع تصغير الجدول مثلا" هيثم تصغيره

yoptions: الرولز ل y مشابهه ل x

xpadding: الهامش من ناحية ال x

ypadding: الهامش من ناحية ال y

لو انت مرضى مع الخيارات الأساسية تقدر تستخدم attach\_defaults

```
attach_defaults( child, left_attach, right_attach, top_attach,
bottom_attach )
```

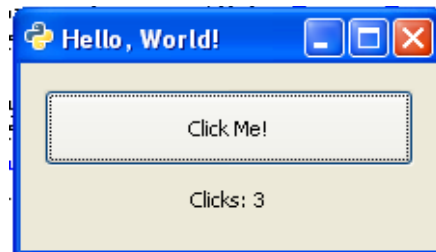
ودي هتخليك تدخل ال left, right, top, bottom attach فقط والباقي هيكون بالديفولت لل x,y options

gtk.FILL | gtk.EXPAND

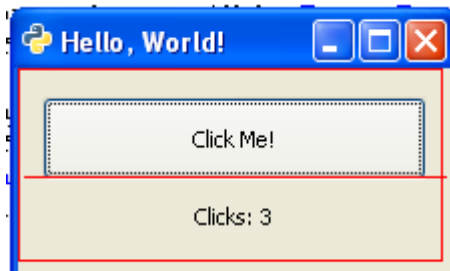
وال x,y padding هيكون 0

فين ال fixed ؟ موجود لعرض ال ويدجتس بتحديد ال مكان على الفورم ولكن "استخدام السابق افضل من حيث حماية طريقة وضعك لل ويدجتس من حيث التمدد والإنكماش وكدا"

تعالى نعمل مثال clicks بصورة واضحة



التصميم



المستطيل الأحمر الكبير عبارة عن vbox وجواه صفين الصف الأول فيه button والثاني فيه label الأول عندنا متغير clicks\_ يعبر عن عدد الضغطات

```
class Window(gtk.Window):
    def __init__(self):
        super(Window, self).__init__(gtk.WINDOW_TOPLEVEL)
        self.__init_comp()
        self.__clicks=0
        self.show_all()
```

```

def getClicks(self):
    return self.__clicks

def setClicks(self, value):
    self.__clicks = value

def delClicks(self):
    del self.__clicks

clicks = property(getClicks, setClicks, delClicks, "Clicks's Docstring")
 ثانيا التصميم |
def __init_comp(self):
    self.set_title("Hello, World!")
    self.set_position(gtk.WIN_POS_CENTER)
    self.set_border_width(12)
 هنا بنحدد خصائص النافذة title, position, border_width |

    mvbox=gtk.VBox(False, 0)
 بننشئ Vertical Box عشان نضم فيه button, label |
    btnclicks=gtk.Button("Click Me!")
 بننشئ ال Button مكتوب عليه Click Me |
    lbl=gtk.Label("Clicks: ")
 بننشئ Label مكتوب عليه Clicks |

    mvbox.pack_start(btnclicks, True, True, 2)
    mvbox.pack_start(lbl, True, True, 0)
 بنضيف ال btnclicks, lbl لل mvbox |

    self.add(mvbox)
 بنضيف ال mvbox لل window |

    btnclicks.connect("clicked", self.__on_btnclicks_clicked, lbl, None)
 بنربط ال clicked signal الخاصة ب btnclicks بطريقة بإسم |
__on_btnclicks_clicked
 احنا مصممينها كالتالى |
def __on_btnclicks_clicked(self, widget, lblclicks, data):
    self.__clicks += 1
    print widget, lblclicks, data
    lblclicks.set_text("Clicks: "+str(self.__clicks))

    وبس كذا!
    المعامل الأول widget بيغير عن ال receiver لل signal
    الثانى ال lblclicks بيغير عن ال label اللى عايزين نغيره
    التالت ال data بيغير عن اى داتا اضافية
    كل اللى هيحصل اتنا هنزود عدد ال __clicks ونعدل ال تكست على ال lblclicks
    بإستخدام
    .set_text(newtext)

```

## الهيكليّة

### GObject

```

|
+GtkObject
+GtkWidget
| +GtkMisc

```

```

| | +GtkLabel
| | | `GtkAccelLabel
| | +GtkArrow
| | `GtkImage
+GtkContainer
| | +GtkBin
| | | +GtkAlignment
| | | +GtkFrame
| | | | `GtkAspectFrame
| | | +GtkButton
| | | | +GtkToggleButton
| | | | | `GtkCheckButton
| | | | | `GtkRadioButton
| | | | `GtkOptionMenu
| | | +GtkItem
| | | | +GtkMenuItem
| | | | +GtkCheckMenuItem
| | | | | `GtkRadioMenuItem
| | | | +GtkImageMenuItem
| | | | +GtkSeparatorMenuItem
| | | | `GtkTearoffMenuItem
| | | +GtkWindow
| | | | +GtkDialog
| | | | | +GtkColorSelectionDialog
| | | | | +GtkFileSelection
| | | | | +GtkFontSelectionDialog
| | | | | +GtkInputDialog
| | | | | `GtkMessageDialog
| | | | `GtkPlug
| | | +GtkEventBox
| | | +GtkHandleBox
| | | +GtkScrolledWindow
| | | `GtkViewport
+GtkBox
| | | +GtkButtonBox
| | | | +GtkHButtonBox
| | | | `GtkVButtonBox
| | | +GtkVBox
| | | | +GtkColorSelection
| | | | +GtkFontSelection
| | | | `GtkGammaCurve
| | | `GtkHBox
| | | +GtkCombo
| | | `GtkStatusbar
+GtkFixed
+GtkPaned
| | | +GtkHPaned
| | | | `GtkVPaned
+GtkLayout
+GtkMenuShell
| | | +GtkMenuBar
| | | | `GtkMenu
+GtkNotebook
+GtkSocket
+GtkTable
+GtkTextView
+GtkToolbar
`GtkTreeView

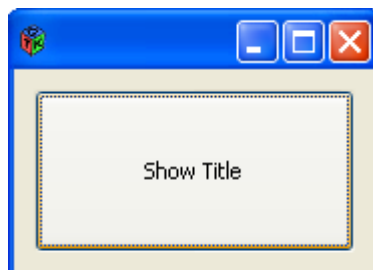
```

```
| +GtkCalendar
| +GtkDrawingArea
| | `GtkCurve
| +GtkEditable
| | +GtkEntry
| | `GtkSpinButton
| +GtkRuler
| | +GtkHRuler
| | `GtkVRuler
| +GtkRange
| | +GtkScale
| | | +GtkHScale
| | | `GtkVScale
| | `GtkScrollbar
| | +GtkHScrollbar
| | `GtkVScrollbar
| +GtkSeparator
| | +GtkHSeparator
| | `GtkVSeparator
| +GtkInvisible
| +GtkPreview
| `GtkProgressBar
+GtkAdjustment
+GtkCellRenderer
| +GtkCellRendererPixbuf
| +GtkCellRendererText
| +GtkCellRendererToggle
+GtkItemFactory
+GtkTooltips
`GtkTreeViewColumn
```

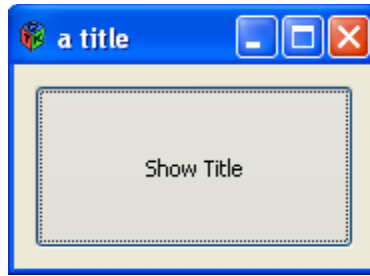
## *Toggle Button*

```
toggle_button = gtk.ToggleButton(label=None)
```

مشابه لل check box ودا ليه حالتين True او False



هنا وهو مش متنشط يعني False



هنا هو متنشط یعنی active  
اول ما يضغط عليه بيرسل toggled signal

```
import gtk

class Window(gtk.Window):

    def __init__(self):
        super(Window, self).__init__(gtk.WINDOW_TOPLEVEL)
        self.__init_comp()

        self.show_all()

    def __init_comp(self):
        self.set_title("Hello, World!")
        self.set_position(gtk.WIN_POS_CENTER)

        mvbox=gtk.VBox(False, 0)
        togbtn=gtk.ToggleButton("Show Title")
        togbtn.set_active(True)
        togbtn.connect("toggled", self.__on_toggled)

        mvbox.pack_start(togbtn, True, True, 2)

        self.add(mvbox)

    def __on_toggled(self, widget):
        if self.title.strip():
            self.set_title(" ")
        else:
            self.set_title("Hello, World!")

if __name__=="__main__":
    w=Window()
    gtk.main()
```

CheckBox

```
check_button = gtk.CheckButton(label=None)
```

الصورة العامة لإنشاءه



نفس نظام السابق ولكن بشكل مختلف ليس اكثر

```
import gtk

class Window(gtk.Window):

    def __init__(self):
        super(Window, self).__init__(gtk.WINDOW_TOPLEVEL)
        self.__init_comp()

        self.show_all()

    def __init_comp(self):
        self.set_title("Hello, World!")
        self.set_position(gtk.WIN_POS_CENTER)

        mvbox=gtk.VBox(False, 0)
        chkbtn=gtk.CheckButton("Show Title")
        chkbtn.set_active(True)
        chkbtn.connect("toggled", self.__on_toggled)

        mvbox.pack_start(chkbtn, True, True, 2)

        self.add(mvbox)

    def __on_toggled(self, widget):
        if self.title.strip():
            self.set_title(" ")
        else:
            self.set_title("Hello, World!")

if __name__=="__main__":
    w=Window()
    gtk.main()
```

ويس كدا

RadioButton

الصورة العامة لإنشاءه

```
radio_button = gtk.RadioButton(group=None, label=None)
```

لازم تظبط ال group دى عشان يشتغل بصورة سليمة فبكل بساطة خليها None لأول radio button وبعد كدا خليها ال radio button اللى انشئ اول واحد!  
هننشئ حاجة مشابهه لدى



التصميم

النافذة فيها 2 radio buttons وفيها Horizontal Separator و button

```
class Window(gtk.Window):
    def __init__(self):
        super(Window, self).__init__(gtk.WINDOW_TOPLEVEL)
        self.__init_comp()
        self.gender="Male"
        self.show_all()

    def __init_comp(self):
        self.set_title("Hello, World!")
        self.set_position(gtk.WIN_POS_CENTER)
        self.set_border_width(12)

        mvbox=gtk.VBox(False, 0)
        rd1=gtk.RadioButton(None, "Male")
        rd1.set_active(True)

        rd1.connect("toggled",self.__on_radio_toggled, "Male")

        rd2=gtk.RadioButton(rd1, "Female")
        rd2.connect("toggled",self.__on_radio_toggled, "Female")

        mvbox.pack_start(rd1, False, False, 2)
        mvbox.pack_start(rd2, False, False, 2)

        mvbox.pack_start(gtk.HSeparator(), True, True, 0)
        btninfo=gtk.Button("OK!")
```

```

btninfo.connect("clicked", self.__on_btninfo_clicked)

mvbox.pack_start(btninfo, False, False, 3)
self.add(mvbox)

def __on_radio_toggled(self, w, data):
    self.gender=data

def __on_btninfo_clicked(self, w):
    md=gtk.MessageDialog(self, gtk.DIALOG_DESTROY_WITH_PARENT,
gtk.MESSAGE_INFO, gtk.BUTTONS_OK, self.gender)
    md.run()
    md.destroy()

```

لاحظ هنا انشئنا Hseparator (فاصل افقى) وظيفناه مباشرة  
 ال signal المهمة هنا هى toggled لل radio buttons

## Adjustment

هى مش ويدجت ولكن بيستخدم فى تخزين ونقل معلومات محددة لإعدادات ويدجتس  
 معينة زي السكرولبارز والسينرز والراينجز وغيرهم  
 لعمل adjustment object بننشئه كالتالى

```
Adjustment( value, lower, upper, step_increment,
page_increment, page_size )
```

تقدر تعتبرها ك model لويدجت وهو بيعرضها لك فى ال view بتاعته على كل حال  
 هنشوف

ال value القيمة الأساسية

ال lower اقل قيمة

ال upper اعلى قيمة

ال step\_increment مقدار الزيادة

## Scale

فى منها افقى وفي رأسى لإنشاء الأوبجكتس منها  
 VScale( adjustment )

```
VScale( min, max, step )
```

```
HScale( adjustment );
```

```
HScale min, max, step );
```

يا إما تباصى adjustment او تباصى اقل واكبر قيمة والزيادة لإظهار القيمة مع ال scale او لأ تقدر تستخدم draw\_value او set\_draw\_value وتديهم قيمة true او false فى حالة الإخفاء -- هى افتراضيا..

digits/set\_digits لتحديد عدد الأرقام بعد العلامة العشرية اللى عايزها تظهر set\_value\_pos(pos) تقدر تستخدمهم لتحديد المكان اللى هيظهر عليه قيمة ال value ودى بتاخذ القيم

```
gtk.POS_LEFT  
gtk.POS_RIGHT  
gtk.POS_TOP  
gtk.POS_BOTTOM
```



مثلا لعمل المثال دا همنشئه كالتالى

```
class Window(gtk.Window):  
  
    def __init__(self):  
        super(Window, self).__init__(gtk.WINDOW_TOPLEVEL)  
        self.__init_comp()  
  
        self.show_all()  
  
    def __init_comp(self):  
        self.set_title("Hello, World!")  
        self.set_position(gtk.WIN_POS_CENTER)  
        self.set_border_width(12)  
  
        adj=gtk.Adjustment(5,1, 101)  
        hscale=gtk.HScale(adj)  
        hscale.set_digits(0)
```

```

mvbox=gtk.VBox(False, 0)
mvbox.pack_start(hscale, True, True, 0)
mvbox.pack_start(gtk.HSeparator(), True, True, 0)

self.add(mvbox)

```

update\_policy

بتحدد امته يتم تعديل ال value بال adjustment الخاصة بال range widget وترسل ال value\_changed signal بتاخذ قيم مثل

gtk.UPDATE\_CONTINUOUS

بترسل عند حدوث اقل تغيير ممكن فى ال range

gtk.UPDATE\_DISCONTINUOUS

بترسل لما المستخدم يسبب الماوس ويكون ال range ثابت

gtk.UPDATE\_DELAYED

بترسل بمجرد ان المستخدم يسبب الماوس او يتوقف عن الحركة لفترة صغيرة وبينم التحكم فى ال policy من خلال

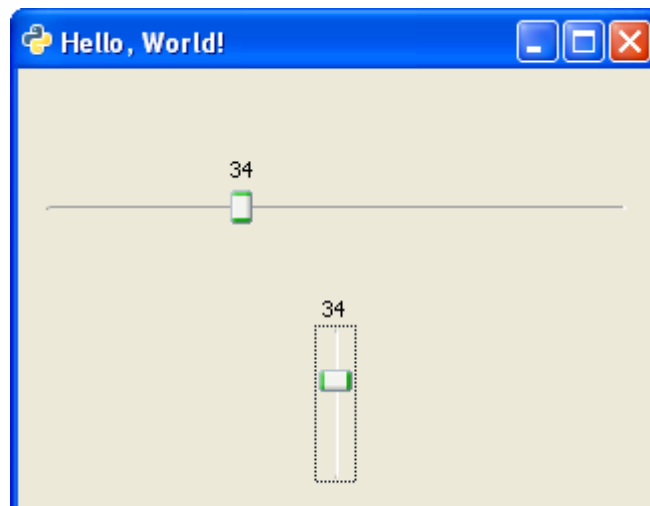
.set\_update\_policy(up\_policy)

للحصول على ال adjustment الخاصة بال range استخدم

.get\_adjustment

لتعديل ال adjustment استخدم

.set\_adjustment(adj)



هنا عندنا 2 scales واحدة افقى والثانية رأسى والإثنين هندیلهم adjustment واحدة عشان التغيير فيهم بيقة على التوازى

```
class Window(gtk.Window):
```

```
def __init__(self):
```

```

super(Window, self).__init__(gtk.WINDOW_TOPLEVEL)
self.__init_comp()

self.show_all()

def __init_comp(self):
    self.set_title("Hello, World!")
    self.set_position(gtk.WIN_POS_CENTER)
    self.set_border_width(12)

    adj=gtk.Adjustment(5, 1, 101)
    hscale=gtk.HScale(adj)
    hscale.set_digits(0)
    vscale=gtk.VScale(adj)
    vscale.set_digits(0)

    mvbox=gtk.VBox(False, 0)
    mvbox.pack_start(hscale, True, True, 0)
    mvbox.pack_start(vscale, True, True, 2)

    self.add(mvbox)

```

`.set_digits(num)`

بتحدد عدد الأرقام المطلوبة بعد العلامة (خليناها 0)



نيجى لمثال تانى هنا عندنا Scale و SpinButton عايزين نربطهم ان لما يتغير قيمة اى منهم يتعدل فى الثانية  
 هنا هنستخدم ال adjustment object للإثنين بحيث ان يتعدل قيمة ال value فيها للإثنين (بما انها ال model اللى بيعرضه كل من ال Scale, SpinButton)

كود المثال

```

class Window(gtk.Window):
    def __init__(self):
        super(Window, self).__init__(gtk.WINDOW_TOPLEVEL)
        self.__init_comp()

        self.show_all()

```

```

def __init_comp(self):
    self.set_title("Hello, World!")
    self.set_position(gtk.WIN_POS_CENTER)
    self.set_border_width(12)

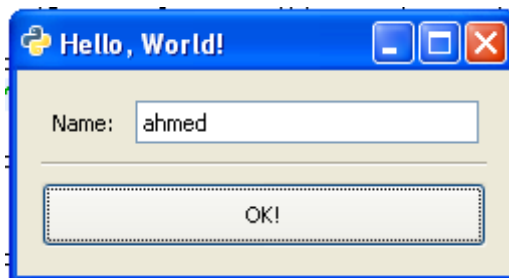
    adj=gtk.Adjustment(5,1, 101)
    hscale=gtk.HScale(adj)
    hscale.set_digits(0)
    spin=gtk.SpinButton(adj)
    spin.set_digits(0)

    mvbox=gtk.VBox(False, 0)
    mvbox.pack_start(hscale, True, True, 0)
    mvbox.pack_start(spin, True, True, 2)
    mvbox.pack_start(gtk.HSeparator(), True, True, 0)

    self.add(mvbox)

```

## Label/Entry



ال Label يستخدم لعرض تكست ما بدون الحاجة لتغييره من المستخدم  
 ال Text Entry مشابه لل تكست فيلد بيدخل فيه المستخدم قيمة مطلوبة مثلا...

التصميم



الكود

```

class Window(gtk.Window):

    def __init__(self):
        super(Window, self).__init__(gtk.WINDOW_TOPLEVEL)
        self.__init_comp()

        self.show_all()

    def __init_comp(self):
        self.set_title("Hello, World!")

```

```

self.set_position(gtk.WIN_POS_CENTER)
self.set_border_width(12)

mvbox=gtk.VBox(False, 0)

lblname=gtk.Label("Name:")
nameentry=gtk.Entry()

hbox=gtk.HBox(False, 0)
hbox.pack_start(lblname, True, True, 0)
hbox.pack_start(nameentry, True, True, 4)

mvbox.pack_start(hbox, True, True, 0)
mvbox.pack_start(gtk.HSeparator(), True, True, 2)

btnok=gtk.Button("OK!")
btnok.connect("clicked", self.__on_btnok_clicked , nameentry)

mvbox.pack_start(btnok, True, True, 2)
self.add(mvbox)

```

```

def __on_btnok_clicked(self, w,e, data=None):
    print e.get_text()

```

Entry:

.set\_text(newtext)

لتغيير ال text

.insert\_text(text, \_from)

بتصيف text من عند نقطة ال from\_

.select\_region(\_from,\_to)

بتطلل منطقة معينة تبدأ من from\_ وتنتهى ب to\_

.set\_max\_length(maxlen)

بتحدد بيها اقصى عدد حروف لل entry

.set\_editable(bool)

هل يقدر المستخدم يعدل فيها؟

Label:

.set\_text(text)

تعديل ال text

.get\_text()

اعادة ال text

.set\_justify(just)

بتاخذ قيم كالتالى

JUSTIFY\_LEFT يسار

JUSTIFY\_RIGHT يمين

JUSTIFY\_CENTER المنتصف

.set\_line\_wrap(bool)

هل ي wrap السطور او لأ؟

.set\_markup(markup)

تخزين markup

gtk.Arrow

يستخدم ليشير الى اتجاه ما لبرنامج (بوضع رأس السهم)



```
arrow = gtk.Arrow(arrow_type, shadow_type)
```

```
arrow.set(arrow_type, shadow_type)
```

arrow\_type بتعبر عن نوع السهم اعلى ، اسفل ، يمين ، يسار

```
ARROW_UP  
ARROW_DOWN  
ARROW_LEFT  
ARROW_RIGHT
```

و shadow\_type بتحدد نوع ال shadow

```
SHADOW_IN  
SHADOW_OUT # the default  
SHADOW_ETCHED_IN  
SHADOW_ETCHED_OUT
```

مثال

Create an Arrow widget with the specified parameters #

```
# and pack it into a button  
def create_arrow_button(arrow_type, shadow_type):  
    button = gtk.Button()  
    arrow = gtk.Arrow(arrow_type, shadow_type)  
    button.add(arrow)  
    button.show()  
    arrow.show()  
    return button  
  
class Arrows(object):  
    def __init__(self):  
        # Create a new window  
        window = gtk.Window(gtk.WINDOW_TOPLEVEL)  
  
        window.set_title("Arrow Buttons")  
  
        # It's a good idea to do this for all windows.  
        window.connect("destroy", lambda x: gtk.main_quit())  
  
        # Sets the border width of the window.  
        window.set_border_width(10)  
  
        # Create a box to hold the arrows/buttons  
        box = gtk.HBox(False, 0)  
        box.set_border_width(2)  
        window.add(box)  
  
        # Pack and show all our widgets  
        box.show()  
  
        button = create_arrow_button(gtk.ARROW_UP, gtk.SHADOW_IN)
```

```

box.pack_start(button, False, False, 3)

button = create_arrow_button(gtk.ARROW_DOWN, gtk.SHADOW_OUT)
box.pack_start(button, False, False, 3)

button = create_arrow_button(gtk.ARROW_LEFT, gtk.SHADOW_ETCHED_IN)
box.pack_start(button, False, False, 3)

button = create_arrow_button(gtk.ARROW_RIGHT, gtk.SHADOW_ETCHED_OUT)
box.pack_start(button, False, False, 3)

window.show()

```

في الدالة

```

# and pack it into a button
def create_arrow_button(arrow_type, shadow_type):
    button = gtk.Button();
    arrow = gtk.Arrow(arrow_type, shadow_type);
    button.add(arrow)
    button.show()
    arrow.show()
    return button

```

بنقوم بإنشاء button يحوي arrow قمنا بإنشاءه من خلال ال arrow\_type, shadow\_type  
واضفناه لل button بإستخدام ال add method

gtk.Tooltips

بتستخدم لتحديد ال tooltip ( نص مساعد على الويدجت)  
تنشئ كالتالي

```

tooltips = gtk.Tooltips()

```

تقوم بتحديد نص التلميح بإستخدام ال set\_tip method

```

tooltips.set_tip(widget, tip_text)

```

widget هو الويدجت المطلوب تحديد ال tip له  
tip\_text النص

فقط قم بإضافة ال create\_arrow\_button

```

class Tooltips:
    def __init__(self):
        # Create a new window
        window = gtk.Window(gtk.WINDOW_TOPLEVEL)

        window.set_title("Tooltips")

        # It's a good idea to do this for all windows.
        window.connect("destroy", lambda w: gtk.main_quit())

        # Sets the border width of the window.
        window.set_border_width(10)

        # Create a box to hold the arrows/buttons
        box = gtk.HBox(False, 0)
        box.set_border_width(2)

```

```

window.add(box)

# create a tooltips object
self.tooltips = gtk.Tooltips()

# Pack and show all our widgets
box.show()

button = create_arrow_button(gtk.ARROW_UP, gtk.SHADOW_IN)
box.pack_start(button, False, False, 3)
self.tooltips.set_tip(button, "SHADOW_IN")

button = create_arrow_button(gtk.ARROW_DOWN, gtk.SHADOW_OUT)
box.pack_start(button, False, False, 3)
self.tooltips.set_tip(button, "SHADOW_OUT")

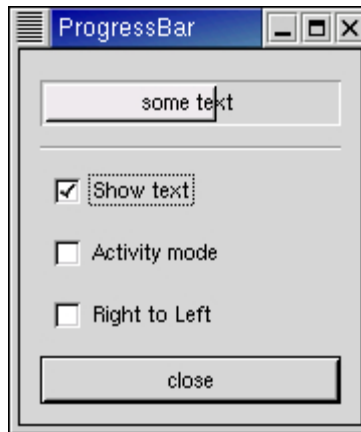
button = create_arrow_button(gtk.ARROW_LEFT, gtk.SHADOW_ETCHED_IN)
box.pack_start(button, False, False, 3)
self.tooltips.set_tip(button, "SHADOW_ETCHED_IN")

button = create_arrow_button(gtk.ARROW_RIGHT, gtk.SHADOW_ETCHED_OUT)
box.pack_start(button, False, False, 3)
self.tooltips.set_tip(button, "SHADOW_ETCHED_OUT")

window.show()

```

gtk.ProgressBar



هو ويدجت يستخدم لعرض تقرير عن الحالة

لإنشاءه

```
progressbar = gtk.ProgressBar(adjustment=None)
```

في حال عدم تحديد ال adjustment ه يتم انشاءها

```
.set_fraction(fraction)
```

لتحديد ال fraction وهى الكم المنتهى

```
.set_orientation(orientation)
```

لتحديد اتجاه ملء الزيادة

PROGRESS\_LEFT\_TO\_RIGHT من اليسار لليمين  
 PROGRESS\_RIGHT\_TO\_LEFT من اليمين للييسار  
 PROGRESS\_BOTTOM\_TO\_TOP من أسفل لأعلى

PROGRESS\_TOP\_TO\_BOTTOM من أعلى لأسفل

.get\_text()

للحصول على النص الظاهر على ال progressbar

.set\_text(to)

تحديد النص الظاهر على ال progressbar إلى to

.pulse()

لتشير حدوث تغيير في ال progressbar

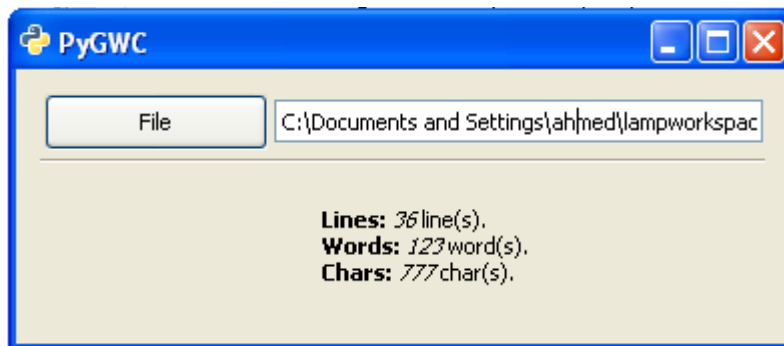
```
def progress_timeout(pbobj):
```

```
    if pbobj.activity_check.get_active():
        pbobj.pbar.pulse()
    else:
        # Calculate the value of the progress bar using the
        # value range set in the adjustment object
        new_val = pbobj.pbar.get_fraction() + 0.01
        if new_val > 1.0:
            new_val = 0.0
        # Set the new value
        pbobj.pbar.set_fraction(new_val)

    # As this is a timeout function, return TRUE so that it
    # continues to get called
    return True
```

.activity\_check.get\_active()

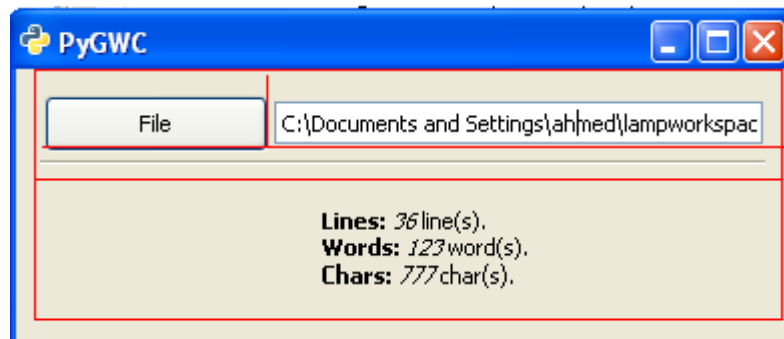
هل progressobject نشط أولاً

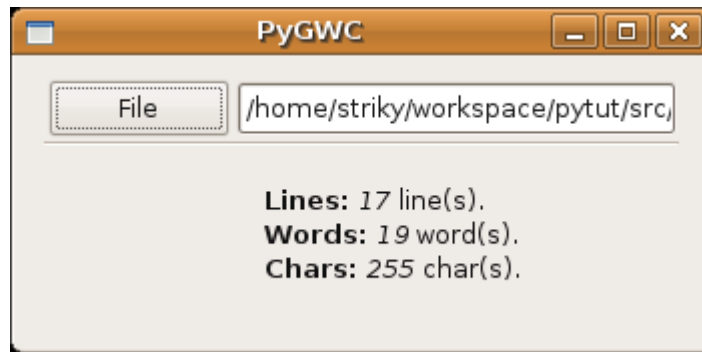


الأول نكتب السرفيس

```
class WordCounter(object):  
  
    def __init__(self):  
        pass  
  
    def set_file(self, p):  
        self.filepath=p  
        fileobj=file(p, "r")  
        self.txt=fileobj.read()  
  
        fileobj.close()  
  
    def get_nwords(self):  
        return self.txt.count(" ")+1  
  
    def get_nlines(self):  
        return self.txt.count("\n")  
  
    def get_nchars(self, count_spaces=True):  
        if count_spaces:  
            return len(self.txt)  
        else:  
            return len(self.txt)-self.get_nwords()  
  
    def get_info_as_markup(self, spaces=True):  
        s="""  
        <b>Lines:</b> <i>%s</i> line(s).  
        <b>Words:</b> <i>%s</i> word(s).  
        <b>Chars:</b> <i>%s</i> char(s).  
        """%(self.get_nlines(), self.get_nwords(), self.get_nchars(spaces))  
        return s
```

التصميم





الكود

```

class Window(gtk.Window):
    def __init__(self):
        super(Window, self).__init__(gtk.WINDOW_TOPLEVEL)
        self.__init_comp()
        self.wc=WordCounter()

        self.show_all()

    def __init_comp(self):
        self.set_title("PyGWC")
        self.set_position(gtk.WIN_POS_CENTER)
        self.set_border_width(12)

        mvbox=gtk.VBox(False, 0)
        btnfile=gtk.Button("File")
        btnfile.connect("clicked", self.select_file)
        self.fileentry=gtk.Entry()
        self.fileentry.set_editable(False)
        hbox=gtk.HBox(False, 0)

        hbox.pack_start(btnfile, True, True, 2)
        hbox.pack_start(self.fileentry, True, True, 1)

        mvbox.pack_start(hbox, True, True, 0)
        mvbox.pack_start(gtk.HSeparator(), True, True, 2)

        self.lblinfo=gtk.Label()
        mvbox.pack_start(self.lblinfo, True, True, 0)

        self.add(mvbox)
  
```

هنربط ال clicked signal الخاصة بال btnfile ب select\_file callback المعرفة كالتالى

```

:(def select_file(self,w

sel = gtk.FileChooserDialog("Open..",
    self,
    gtk.FILE_CHOOSER_ACTION_OPEN,
    (gtk.STOCK_CANCEL, gtk.RESPONSE_CANCEL,
     gtk.STOCK_OPEN, gtk.RESPONSE_OK)
    )
  
```

```

sel.set_default_response(gtk.RESPONSE_OK)

res=sel.run()
if res==gtk.RESPONSE_OK:
    self.wc.set_file(sel.get_filename())
    self.fileentry.set_text(sel.get_filename())
    self.lblinfo.set_markup(self.wc.get_info_as_markup(spaces=True))

else:
    print "Dialog with RESPONSE_CANCEL!"

sel.destroy()

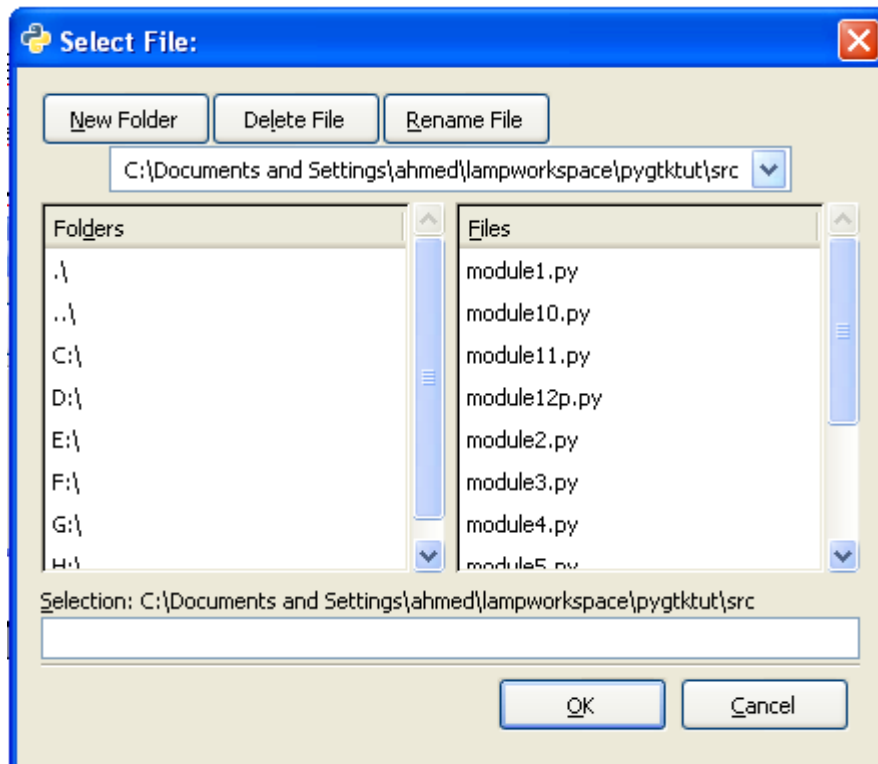
```

هنا بننشئ اوبيجت من ال FilechooserDialog (لأختيار الملفات او الفولدرات) ونحدد العنوان بال constructor واخبرناه بعنوان النافذة ،ونوع فعلها (ACTION\_OPEN لفتح الملفات وليست للحفظ) وانواع ردود الفعل العائدة منها وهي OK, CANCEL وايضا ال stocks

```

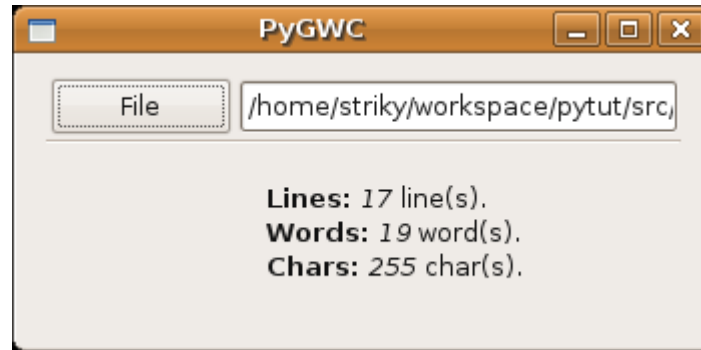
sel = gtk.FileChooserDialog("Open",
                             self,
                             gtk.FILE_CHOOSER_ACTION_OPEN,
                             (gtk.STOCK_CANCEL, gtk.RESPONSE_CANCEL,
                              gtk.STOCK_OPEN, gtk.RESPONSE_OK)
                             )

```



هيتم تدمير ال sel ويتم تحدد اسم الفايل الخاص بال WordCounter object نعدل التكتست على ال fileentry لإسم الفايل اخيرا نخط ال markup الناتج من ال wc.get\_info\_as\_markup على ال lblinfo بإستخدام ال

set\_markup method



ملحوظة: لا تقم بإستخدام FileSelection (تم تعديل المثال ليستخدم FileChooserDialog) `gtk.ComboBox`



قائمة منسدلة بتشمل مجموعة من الإختيارات ، فى مثالنا هنا مجموعة من اسماء التوزيعات

```
class Window(gtk.Window):
```

```
def __init__(self):
    super(Window, self).__init__()
    self.__init_comp() #prepare components

    self.show_all()

def __init_comp(self):

    self.vbox=gtk.VBox(False, 2)

    entries = ["Slackware", "Ubuntu", "Mandriva", "Debian"]
    cbo=gtk.combo_box_new_text()
    map(cbo.append_text, entries)

    cbo.connect("changed", self._on_cboselection_changed)
    self.vbox.pack_start(cbo, False, False)

    self.add(self.vbox)

def _on_cboselection_changed(self, widget, *args):
    print widget.get_active_text()
```

الطريقة السريعة هي انشاءها باستخدام `combo_box_new_text` (بتجهز الكومبو بتكست رندرر) استدعى الطريقة `append_text` على كل المدخلات `entries` باستخدام `map`

اربط ال `changed` signal ب `on_cboselection_changed_` وللحصول على العنصر النشط فى القائمة استخدام `()get_active_text`

Menus



هناك فى المثال نافذة بشريط قوائم فيه قائمة واحدة `File` ويتشمل 3 عناصر `open, save, quit`

```
import gtk
```

```
class Window(gtk.Window):
```

```
    def __init__(self):
        super(Window, self).__init__()
        self.__init_comp() #prepare components

        self.show_all()

    def __init_comp(self):

        self.vbox=gtk.VBox(False, 2)

        mbar=gtk.MenuBar()
        self.vbox.pack_start(mbar, False, False, 2)

        self.connect("delete_event", lambda w,e: gtk.main_quit())

        file_item=gtk.MenuItem("File")
        fmenu=gtk.Menu()
        file_item.set_submenu(fmenu)

        open_item = gtk.MenuItem("Open")
        save_item = gtk.MenuItem("Save")
        quit_item = gtk.MenuItem("Quit")

        map(fmenu.append, [open_item,save_item, quit_item])

        open_item.connect("activate", self._on_item_activated, "Open")
        save_item.connect("activate", self._on_item_activated, "Save")
        quit_item.connect("activate", self._on_item_activated, "Quit")

        mbar.append(file_item)
```

```
self.add(self.vbox)
```

خطوات متتالية لإنشاء القوائم  
-1 شريط القوائم من الصف MenuBar وإضافته للصندوق الرأسى

```
mbar=gtk.MenuBar()  
self.vbox.pack_start(mbar, False, False, 2)
```

-2 العنصر الذى سيحوى القائمة وهنا هو File وينشئ من الصف MenuItem

```
file_item=gtk.MenuItem("File")
```

-3 القائمة من الصف Menu

```
fmenu=gtk.Menu()
```

-4 ضم القائمة

```
file_item.set_submenu(fmenu)
```

-5 إنشاء عناصر القائمة وإضافتهم باستخدام الطريقة append

```
open_item = gtk.MenuItem("Open")  
save_item = gtk.MenuItem("Save")  
quit_item = gtk.MenuItem("Quit")
```

```
map(fmenu.append, [open_item,save_item, gtk.SeparatorMenuItem(),  
quit_item])
```

لاحظ اننا انشأنا فاصل افقى SeparatorMenuItem ليفصل بين عنصرى فتح وحفظ وعنصر اغلاق  
-6 ربط ال activate signal بال callback المناسبة

```
open_item.connect("activate", self._on_item_activated, "Open")  
save_item.connect("activate", self._on_item_activated, "Save")  
quit_item.connect("activate", self._on_item_activated, "Quit")
```

التي عرفناها كالتالى مثلا

```
def _on_item_activated(self, w, action):  
    print "Calling %s"%action
```

-7 اخيرا ضم العنصر الذى يحوى القائمة الى شريط القوائم  
mbar.append(file\_item)

يوجد طريقة اسهل وهى باستخدام ال UIManager توفر عليك الكثير من الكتابة

Gladizer

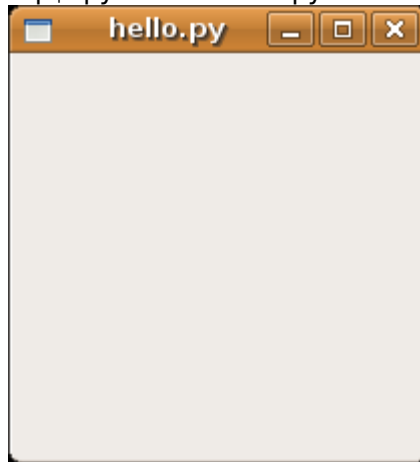
تقدر تصمم واجهات ممتازة باستخدام glade وبدون الحاجة لكتابة الكثير من الأكواد فيما يتعلق بشكل الواجهة بالإستعانة ب gladizer ، ملف ال glade الواجهة الممثلة بصورة XML لاتعلم بايئون عنها شئ سوا انها ملف XML فيجب ان تخبرها بأنها وصف لواجهة برنامج ما وانك تريد ان تستخدمها فى تطبيقك. تستطيع القيام بذلك يدويا، او بإستخدام gladizer او أى اداة مشابهة مثل glade2py, gladeX الخ

-1 افتح glade وانشئ نافذة وقم بحفظ الملف ب hello.glade  
هيكون الكود مشابه للتالى

```
<?xml version="1.0" encoding="UTF-8" standalone="no"?>  
<!DOCTYPE glade-interface SYSTEM "glade-2.0.dtd">  
<!--Generated with glade3 3.4.5 on Tue Dec 2 06:07:51 2008 -->  
<glade-interface>  
  <widget class="GtkWindow" id="window1">  
    <child>  
      <placeholder/>  
    </child>  
  </widget>
```

</glade-interface>

استدعى gladizer ليقوم بالربط  
striky@striky-desktop:~/Desktop\$ gladizer.py -f hello.glade -p Python > hello.py  
قم بالتنفيذ  
striky@striky-desktop:~/Desktop\$ python hello.py



تصفح الكود الناتج من جلايزر ستجده مشابه للتالى

```
#!/bin/python
##CODE GENERATED by gladizer 1.2

import pygtk
pygtk.require('2.0')

import gtk, gtk.glade

class MainWindow(object):
    def __init__(self):

        #Widget tree..
        self.wTree=gtk.glade.XML('hello.glade')

        #connect signals and handlers.
        self.wTree.signal_autoconnect(self)

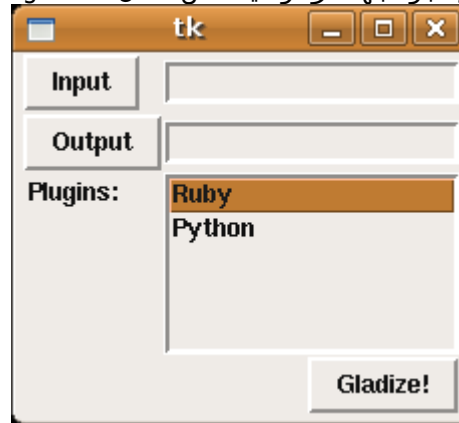
        self._window1=self.wTree.get_widget('window1')
        self._window1.show()
# run main loop

def main():
    mainwindow = MainWindow()
    #mainwindow.window1.show()
    gtk.main()

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

وبعد ذلك قم بإضافة ال callbacks كما تحب :

تستطيع ايضا استخدام gladizer بواجهة رسومية من خلال gladizerguitk



صفحة gladizer

[/http://sourceforge.net/projects/gladizer](http://sourceforge.net/projects/gladizer)

ملحوظة هناك بعض الأمثلة محسنة من <http://pygtk.org/pygtk2tutorial/index.html> وهذا الملف ليس بديلا لها، ولكن تحت المراجعة.

```
To be written...
#UI Manager
#Toolbars/Statusbar
#text view
#tree view
#Simple text editor.
```