Python i GTK

Michał Nowikowski
godfryd @ gmail.com

Zimowisko Linuksowe
Puck 2005

Co to jest Python?

- język skryptowy i nie tylko
- doskonały do nauki i stosowania w dużych projektach
- dynamic-typed, object-oriented
- duże wsparcie w Internecie (społeczność)
- przenośny: wsparcie na kilkunastu platformach (w tym Linux)
- licencja kompatybilna z GPL, zatwierdzona przez OSI

Zastosowanie

- szybkie prototypowanie
- programowanie serwisów webowych
- zagnieżdżalny język skryptowy w aplikacjach
- przetwarzanie XMLa
- aplikacje bazodanowe
- aplikacje graficzne

Kto tego używa

- Google
- NASA
- Industrial Light & Magic
- Yahoo!
- RedHat
- Zope

Cechy języka

- interpretowany i interaktywny
- obiektowy:
  - wszystko jest obiektem

    a.__class__.__name__
  - polimorfizm, dziedziczenie
- skalowalny - poprzez hierarchiczny podział kodu: pakiety, moduły, klasy, funkcje
- obsługa wyjątków
- strong dynamic typing
- rozszerzalny i zagnieżdżalny
- notacja lambda (język funkcyjny jak Lisp)

Dostępne bibliteki / interfejsy

- XML: DOM, expat, XML-RPC, SOAP, Web Services
- RDBMS: MySQL, PostgreSQL, Oracle, ODBC, i inne
- Java (Jython)
- COM, DCOM (Excel, Word)
- SMTP, POP3, FTP, HTTP
- http://www.python.org/pypi (a'la CTAN lub CPAN)

Podstawowe typy

- liczbowe: int, long, float, complex
- łańcuchy znakowe (UTF-8)
- listy i dictionary:

```
l = [1, 2, 3, 4, 5]
l[:2]   # do drugiego elementu od 0 czyli [1, 2]
l[:]    # kopia listy
l[-1]   # ostatni
l[:-1]  # wszystkie bez ostatniego
d = {"a": 1, 3: "b"}
d[“a”]  # wynik 1
```

Przykład

```
def gcd(a, b):
    "greatest common divisor"
    while a != 0:
        a, b = b%a, a          #  podwójne podstawienie
    return b

help(gcd)
gcd(12, 15)
```

Przykład klasy

```
class Stack:
    "A well-known data structure"    # doc string
    def __init__(self):    # konstruktor
        self.items = []
    def push(self, x):
        self.items.append(x)    # bez limitu
    def pop(self):
        x = self.items[-1]    # co się stanie gdy jest pusty?
        del self.items[-1]
        return x
    def empty(self):
        return len(self.items) == 0
```

Tricki

- Zamiana, zamiast:
  tmp = a; a = b; b = tmp
- mamy:
  a, b = b, a

- Konwersja:
  bool("C++ sucks!")
  True

Biblioteki do GUI

- Tk
- wxWindows
- Qt
- Mac
- MFC
- GTK

GTK

- popularna biblioteka GUI
- przenośna – Linux, Windows

GTK Killer Apps

- GIMP
- Evolution
- Nautilus
- Galeon / Epiphany
- Gnumeric
- Gaim
- Inkscape
- Kino
- Totem
- I inne – popatrzeć na gnomefiles.org

Problemy z GTK

- dodawanie mnóstwa ficzerów jest pracochłonne w C
- ostatnie dyskusje co dalej z GNOME
- C++, C#, Java, Perl, Python

Python i GTK

- biblioteka bindująca pyGTK
- i inne do GNOME

GTK/Python Killer Apps

- instalator RedHata
- Gramps
- gdesklets
- Meld
- Revelation
- Eroaster
- PythonCAD

Prosty przykład na początek

- import bibliotek
  ```
  import pygtk; pygtk.require(“2.0”); from gtk import *
  gtk.main()
  ```
- tworzenie okienka i widgetów
  ```
  win = Window()
  vbox = VBox()
  win.add(vbox)
  b = Button(“press me”)
  l = Label(“labelka”)
  vbox.pack_start(b)
  vbox.pack_start(l)
  win.show_all()
  ```
- interaktywnie w gpython

Sygnały

```
def hello(widget, label):
    print “hello”
    label.set_text(“hello”)

s = b.connect(“clicked”, hello, l)
b.disconnect(s)
```

Responsiveness

- rozwiązania jednowątkowe:
  - timery

```
gtk.timeout_add(100, doer)
```

  - generatory

```
from __future__ import generator
def doer(label, n):
    i = n
    while i > 0:
        label.set_text(str(i))
        i = i - 1
        yield True
    yield False
gtk.timeout_add(100, doer().next)
```

Responsiveness cd.

- rozwiązanie wielowątkowe:

```
import threading
t =threading.Thread(target=doer)
t.start()
t.join()

gtk.threads_init()
gtk.threads_enter()
gtk.threads_leave()
```

- obsługa sygnałów automatycznie zakłada locka, natomiast idle, timeout i input handlers już nie

Glade

- biblioteka do budowania GUI w XMLu

```
import gtk.glade
xml = gtk.glade.XML(“a.glade”)
win = xml.get_widget(“window”)
b = xml.get_widget(“button”)
```

Glade cd.

- obsługa sygnałów

```
xml.signal_autoconnect({
    “on_win_destroy”: gtk.mainquit,
    “on_btn_clicked”: on_btn_clicked})
```

Biblioteki PyGTK

- GTK (GTK, GDK, GObject)
- GNOME (GNOME-VFS, GNOMEUI,
- GNOME-EXTRAS (gnomeapplet, gnomeprint, gtkhtml, egg, gtksourceview, gtkmozembed, gtkspell, wnck

Linki

- zagraniczne
  - http://www.python.org/
  - http://www.pygtk.org/
- polskie
  - http://pl.wikipedia.org/wiki/Python
  - http://python.kofeina.net/