SURVEY METHODOLOGY

SURVEY METHODOLOGY
This is the Subtitle

Robert M. Groves
Universitat de les Illes Balears

Floyd J. Fowler, Jr.
University of New Mexico

A JOHN WILEY & SONS, INC., PUBLICATION

Published by John Wiley & Sons, Inc., Hoboken, New Jersey.
Published simultaneously in Canada.

Library of Congress Cataloging-in-Publication Data:

Survey Methodology / Robert M. Groves . . . [et al.].
p. cm.—(Wiley series in survey methodology)
"Wiley-Interscience."
Includes bibliographical references and index.
ISBN 0-471-48348-6 (pbk.)
1. Surveys—Methodology. 2. Social sciences—Research—Statistical methods. I. Groves, Robert M. II. Series.

HA31.2.S873 2007
001.4'33—dc22 2004044064
Printed in the United States of America.

10 9 8 7 6 5 4 3 2 1

To my parents

CONTRIBUTORS

MASAYKI ABE, Fujitsu Laboratories Ltd., Fujitsu Limited, Atsugi, Japan

L. A. AKERS, Center for Solid State Electronics Research, Arizona State University, Tempe, Arizona

G. H. BERNSTEIN, Department of Electrical and Computer Engineering, University of Notre Dame, Notre Dame, South Bend, Indiana; formerly of Center for Solid State Electronics Research, Arizona State University, Tempe, Arizona

CONTENTS IN BRIEF

CONTENTS

LIST OF FIGURES

LIST OF TABLES

FOREWORD

This is the foreword to the book.

PREFACE

This is an example preface. This is an example preface. This is an example preface. This is an example preface.

R. K. WATTS

Durham, North Carolina
September, 2007

ACKNOWLEDGMENTS

From Dr. Jay Young, consultant from Silver Spring, Maryland, I received the initial push to even consider writing this book. Jay was a constant “peer reader” and very welcome advisor durying this year-long process.

To all these wonderful people I owe a deep sense of gratitude especially now that this project has been completed.

G. T. S.

ACRONYMS

ACGIH	American Conference of Governmental Industrial Hygienists
AEC	Atomic Energy Commission
OSHA	Occupational Health and Safety Commission
SAMA	Scientific Apparatus Makers Association

GLOSSARY

NormGibbs	Draw a sample from a posterior distribution of data with an unknown mean and variance using Gibbs sampling.
pNull	Test a one sided hypothesis from a numberically specified posterior CDF or from a sample from the posterior
sintegral	A numerical integration using Simpson's rule

SYMBOLS

A Amplitude
& Propositional logic symbol
a Filter Coefficient

$\mathcal{B}$ Number of Beats

INTRODUCTION

CATHERINE CLARK, PHD.

Harvard School of Public Health
Boston, MA, USA

The era of modern began in 1958 with the invention of the integrated circuit by J. S. Kilby of Texas Instruments [**?**]. His first chip is shown in Fig. I. For comparison, Fig. I.2 shows a modern microprocessor chip, [**?**].

This is the introduction. This is the introduction. This is the introduction. This is the introduction. This is the introduction. This is the introduction.

$$ABC\mathcal{DEF}\alpha\beta\Gamma\Delta\sum_{def}^{abc} \tag{I.1}$$

REFERENCES

1. J. S. Kilby, "Invention of the Integrated Circuit," *IEEE Trans. Electron Devices,* **ED-23,** 648 (1976).
2. R. W. Hamming, *Numerical Methods for Scientists and Engineers*, Chapter N-1, McGraw-Hill, New York, 1962.
3. J. Lee, K. Mayaram, and C. Hu, "A Theoretical Study of Gate/Drain Offset in LDD MOSFETs" *IEEE Electron Device Lett.,* **EDL-7**(3). 152 (1986).

PART I

SUBMICRON SEMICONDUCTOR MANUFACTURE

CHAPTER 1

HOME

CHAPTER 2

OVERVIEW

CHAPTER 3

ENVIRONTMENT SETUP

CHAPTER 4

BASIC SYNTAX

CHAPTER 5

VARIABEL TYPE

CHAPTER 6

BASIC OPERATOR

CHAPTER 7

DESICION MAKING

CHAPTER 8

LOOP

CHAPTER 9

NUMBERS

CHAPTER 10

STRINGS

CHAPTER 11

LISTS

CHAPTER 12

TUPLES

CHAPTER 13

DICTIONARY

CHAPTER 14

FUNCTIONS

CHAPTER 15

MODULES

CHAPTER 16

FILES I/O

CHAPTER 17

EXCEPTIONS

CHAPTER 18

CLASESS/OBJECT

CHAPTER 19

REG EXPRESSION

CHAPTER 20

CGI PROGRAMMING

Common Gateway Interface atau disingkat CGI merupakan standar untuk menghubungkan berbagai program aplikasi ke halaman web. CGI mirip dengan program komputer yang menjadi perantara antara standar HTML yang menjadikan tampilan web dengan program lain, seperti basis data (database). Hasil yang diperoleh dari proses pencarian dikirimkan kembali ke halaman web untuk ditampilkan dalam format HTML.

CGI (Common Gateway Interface) adalah bentuk dari hubungan interaktif di mana client (browser) bisa mengirimkan suatu masukan kepada server, dan server mengolah masukan tersebut serta mengembalikannya kepada client (browser). Contoh sederhana adalah saat kita menggunakan sebuah mesin pencari. Saat kita menuliskan keyword dan menekan tombol Search maka browser akan mengirimkan keyword tersebut ke server. Keyword tersebut lalu diolah oleh server dan server mengirimkan data hasil pengolahan (yang sesuai dengan keyword yang kita masukkan) ke browser kita. Jadi yang akan kita lihat pada browser adalah hanya data yang sesuai dengan keyword yang kita masukkan.

Untuk dapat menggunakan CGI syarat yang utama adalah server dengan sistem operasi UNIX (beserta variantnya). Namun perlu kita perhatikan bahwa tidak se-

mua server UNIX (gratis) mampu menangani dan melayani CGI. Server-server yang melayani penempatan web yang berlayanan gratis seperti Geocities dan Homepage, tidak akan mengijinkan penggunaan script CGI dalam web kita. Untuk itu kita bisa mencoba Virtual Avenue, Tripod, atau Hypermart.

Program CGI ditulis dengan menggunakan bahasa yang dapat dimengerti oleh sistem misalnya C/C++, Fortran, Perl, Tcl, Visual Basic, dan lain-lain. Pemilihan bahasa yang digunakan tergantung dari sistem yang digunakan. Jika bahasa pemrograman yang digunakan seperti C atau Fortran maka program-program yang kita buat harus dikompile terlebih dahulu sebelum dijalankan sehingga pada server akan terdapat source code dan program hasil kompilasi. Berbeda jika bahasa yang digunakan yaitu bahasa script seperti PERL, TCL, atau Unix Shell maka hanya akan terdapat script itu sendiri (tanpa ada source code). Jika dibandingkan saat ini banyak orang yang lebih memilih untuk menggunakan script CGI daripada menggunakan bahasa pemrograman karena lebih mudah untuk di-compile dan dimodifikasi.

Pada awalnya CGI merupakan salah satu yang mendekati aplikasi server-side programming. Program CGI yang paling sering digunakan yaitu C++ dan perl. CGI merupakan bagian dari web server yang dapat berkomunikasi dengan program lain yang ada di server. Dengan CGI web server dapat memanggil program yang dibuat dari berbagai bahasa pemrograman (Common). Interaksi antara pengguna dengan berbagai aplikasi, misalnya database, dapat dijembatani oleh CGI (Gateway).

CGI (Common Gateway Interface) merupakan skrip tertua dalam bidang pemrograman web. Skrip bisa didefinisikan sebagai rangkaian dari beberapa instruksi program. Untuk membuat skrip yang dapat dijalankan pada web diperlukan pengetahuan pemograman.

CGI sendiri telah muncul sejak teknologi web diperkenalkan di dunia pada awal tahun 1990, bersama dengan kemunculan CERN, web server pertama di dunia. CGI disediakan sebagai tool atau perlengkapan untuk membuat program web. CGI digunakan untuk membuat program-program tampilan web yang lebih interaktif, koneksi ke basis data, bahkan membuat permainan (game).

CGI pada masa-masa awalnya dibuat dengan bahasa C, bahasa yang juga digunakan untuk membuat web server pertama yaitu, CERN. CGI kemudian diadopsi oleh NCSA (National Central for Supercomputing Application) web server, dan hingga kini masih digunakan pada Apache Web Server, web server yang paling banyak digunakan oleh komunitas internet saat ini.

Walaupun demikian CGI bisa juga direalisasikan dengan banyak bahasa pemrograman lain. Mulai dari C, Perl, Phyton, PHP, Tcl/Tk, hingga skrip shell pada UNIX/LINUX.

CGI seringkali digunakan sebagai mekanisme untuk mendapatkan informasi dari user melalui fill out form, mengakses basis data (database), atau menghasilkan halaman yang dinamis. meskipun secara prinsip mekanisme CGI tidak memiliki lubang keamana, program atau skrip yang dibuat sebagai CGI dapat memiliki lubang keamanan ataupun tidak sengaja). Potensi lubang keamanan yang digunakan dapat terjadi dengan CGI antara lain :

Seorang pemakai yang nakal dapat memasang skrip CGI sehingga dapat mengirimkan berkas kata kunci (password) kepada pengunjung yang mengeksekusi CGI tersebut.

Program CGI dipanggil berkali-kali sehingga server menjadi terbebani karena harus menjalankan beberapa program CGI yang menghabiskan memori dan CPU cycle dari web server.

Sebuah aplikasi web berkomunikasi dengan perangkat lunak client melalui HTTP. HTTP, sebagai protokol yang berbicara menggunakan request dan response menjadikan aplikasi web bergantung kepada siklus ini untuk menghasilkan dokumen yang ingin diakses oleh pengguna. Secara umum, aplikasi web yang akan kita kembangkan harus memiliki satu cara untuk membaca HTTP Request dan mengembalikan HTTP Response ke pengguna.

Pada pengembangan web tradisional, kita umumnya menggunakan sebuah web server seperti Apache HTTPD atau nginx sebagai penyalur konten statis seperti HTML, CSS, Javascript, maupun gambar. Untuk menambahkan aplikasi web kita kemudian menggunakan penghubung antar web server dengan program yang dikenal dengan nama CGI (Common Gateway Interface).

CGI diimplementasikan pada web server sebagai antarmuka penghubung antara web server dengan program yang akan menghasilkan konten secara dinamis. Program-program CGI biasanya dikembangkan dalam bentuk script, meskipun dapat saja dikembangkan dalam bahasa apapun. Contoh dari bahasa pemrograman dan program yang hidup di dalam CGI adalah PHP.

Untuk melihat dengan lebih jelas cara kerja CGI, perhatikan penjelasan berikut:

item Web Server yang berhadapan langsung dengan pengguna, menerima HTTP Request dan mengembalikan HTTP Response.

item Untuk konten statis seperti CSS, Javascript, gambar, maupun HTML web server dapat langsung menyajikannya sebagai HTTP Response kepada pengguna. Konten dinamis seperti program PHP maupun Perl disajikan melalui CGI. CGI Script kemudian menghasilkan HTML atau konten statis lainnya yang akan disajikan sebagai HTTP Response kepada pengguna.

Meskipun terdapat banyak pengembangan selanjutnya dari CGI, ilustrasi sederhana di atas merupakan konsep inti ketika awal pengembangan CGI. Umumnya aplikasi web dengan CGI memiliki kelemahan di mana menjalankan script CGI mengharuskan web server untuk membuat sebuah proses baru. Pembuatan proses baru biasanya akan menggunakan banyak waktu dan memori dibandingkan dengan ek-

sekusi script, dan karena setiap pengguna yang terkoneksi akan mengakibatkan hal ini terhadap server performa aplikasi akan menjadi kurang baik.

CGI sendiri menyediakan solusi untuk hal tersebut, misalnya FastCGI yang menjalankan aplikasi sebagai bagian dari web server. Bahasa lain juga menyediakan alternatif dari CGI, misalnya Java yang memiliki Servlet. Servlet pada Java merupakan sebuah program yang menambahkan fitur dari server secara langsung. Jadi pada pemrograman dengan Servlet, kita akan memiliki satu web server di dalam program kita, dan pada web server tersebut akan ditambahkan fitur-fitur spesifik aplikasi web kita.

KELEBIHAN CGI

Kelebihan yang dimiliki CGI antara lain :

1. Skrip CGI dapat ditulis dalam bahasa apa saja, namun barangkali sekitar 90 % program CGI yang ada di tulis dalam Perl

2. Protokol CGI yang sederhana

3. Kefasihan Perl dalam mengolah teks, menjadikan menulis sebuah program CGI cukup mudah dan cepat.

4. Meski tertua hingga saat ini menurut survey dari Netcraft sekitar 70 % aplikasi di web masih menggunakan CGI. Ini berarti, lebih dari separuh situs Web dinamik yang ada dibangun dengan CGI.

KELEMAHAN CGI

Salah satu kelemahannya ialah kecepatan yang rendah. Untuk menghasilkan keluaran program CGI, overhead yang harus ditempuh cukup besar, Dalam kasus CGI Perl, prosesnya sebagai berikut :

1. Web server terlebih dahulu akan menciptakan sebuah proses baru dan menjalankan interpreter Perl.

2. Perl kemudian mengkompilasi script CGI tersebut, baru kemudian menjalankan skrip.

Keseluruhan siklus ini terjadi untuk setiap request. Dengan kata lain, terlalu banyak waktu yang dibuang untuk menciptakan proses dan tidak ada cache skrip yang telah dikompilasi.

Namun demikian, mungkin ini tidak lagi menjadi kendala di saat teknologi hardware untuk server sudah sedemikian maju; kecepata prosesor saat ini sudah cukup tinggi. Jika situs web menerima kurang dari sepuluh hingga dua puluh ribu hit CGI per hari, rata-rata mesin web server UNIX yang ada sekarang ini mampu menanganinya dengan baik.

Dalam kasus CGI Perl, prosesnya sbb:

- Web server terlebih dahulu akan menciptakan sebuah proses baru dan menjalankan interpreter Perl.

- Perl kemudian mengkompilasi script CGI tersebut, baru kemudian menjalankan skrip.

Keseluruhan siklus ini terjadi untuk setiap request. Dengan kata lain, terlalu banyak waktu dibuang untuk menciptakan proses dan tidak ada cache skrip yang telah dikompilasi.
Jika sebuah situs web menerima kurang dari sepuluh hingga dua puluh ribu hit CGI per hari, rata-rata mesin web server Unix yang ada sekarang ini mampu menanganinya dengan baik.
Angka ini relatif, bergantung pada:

- Tingkat pembebanan mesin web server untuk melakukan pekerjaan lain (misalnya, mengirim mail dan menjalankan server database)

- Aplikasi CGI itu sendiri (sebab beberapa aplikasi CGI berupa skrip tunggal berukuran besar hingga waktu loading-nya cukup lama; umumnya aplikasi CGI yang rumit memecah diri menjadi skrip-skrip terpisah untuk mengurangi waktu loading).

- Cepat atau lambatnya penampilan halaman web yang diterima klien akan lebih bergantung pada koneksi jaringan.

Penerapan CGI

Penerapan CGI yang paling umum adalah dalam pemrosesan . Umumnya, form dipergunakan untuk dua kegunaan utama . Yang sederhana adalah form yang dipakai untuk mengumpulkan informasi dari pengguna dan mengirimkanya ke server. Namun form juga bisa dipakai untuk keperluan yang lebih ”canggih” seperti timbal balik antara pengguna dan server, misalnya form yang memberikan sedaftar pilihan dokumen dalam server kepada pengguna untuk dipilih. Program CGI di server dibuat untuk mengolah informasi ini dan kemudian mengirimkan dokumen - dokumen yang sesuai dengan pilihan pengguna.

Contoh nyata penerapan CGI untuk dokumen dinamis ini misalnya suatu ”buku tamu”. Pengguna memasukkan informasi seperti nama, alamat, alamat e-mail, dan komentar-komentarnya ke dalam form. Setelah server menerima informasi-informasi tadi, program CGI dapat menyimpanya ke dalam suatu File atau secara otomatis mengirimkanya lewat e-mail ke suatu alamat. Program CGI juga bisa menampilkan dokumen yang berisi informasi yang
baru saja dikirimkan oleh pengguna tadi sembari memberikan ucapan terima kasih atas partisipasinya.

Penerapan lain dari CGI adalah sebuah gateway. Artinya adalah program yang dipergunakan sebagai penghubung untuk mengakses informasi yang tidak dapat secara langsung dibaca oleh program browser pengguna.Contoh yang nyata adalah gateway yang menghubungkan antara web server dengan dengan suatu database server yang besar semacam oracle atau DB2, yang memang dapat dilakukan dengan mempergunakan bahasa pemrograman Perl dan DBI extentionta sehingga web server bisa memberikan query dalam SQL (structured query language, yaitu bahasa yang dipakai untuk melakukan pendefinisian maupun manipulasi terhadap database) ke server database Oracle. Setelah informasi dari database keluar, program CGI mengubahnya ke dalam bentuk yang bisa dibaca browser (HTML) dan web server pada giliranya mengirimkanya kepada browser.

Program CGI pada prinsipnya bisa ditulis dalam bahasa pemrograman apa saja, namun kenyataanya tidak semua bahasa pemrograman cocok untuk pemrograman CGI. Penerapan CGI dapat sangat kompleks, dan untuk membuat suatu program CGI menuntut pengetahuan teknis yang cukup tinggi akan pemrograman.

Keamanan pada CGI

CGI dapat menimbulkan lubang keamanan, karena program CGI dapat dijalankan di server lokal dari luar sistem (remote) oleh siapa saja. Apabila program CGI tidak didisain dan dikonfigurasi dengan baik, maka akan terjadi lubang keamanan. Kesalahan yang dapat terjadi antara lain:

- program CGI mengakses berkas (file) yang seharusnya tidak boleh di akses. Misalnya pernah terjadi kesalahan dalam program phf sehingga digunakan oleh orang untuk mengakses berkas password dari server WW.
- runaway CGI-script, yaitu program berjalan di luar kontrol sehingga mengabiskan CPU cycle dari server WWW

Lubang Keamanan CGI

Beberapa contoh lubang keamanan pada CGI

- CGI dipasang oleh orang yang tidak berhak
- CGI dijalankan berulang-ulang untuk menghabiskan resources (CPU, disk): DoS
- Masalah setuid CGI di sistem UNIX, dimana CGI dijalankan oleh userid web server
- Penyisipan karakter khusus untuk shell expansion
- Kelemahan ASP di sistem Windows
- Guestbook abuse dengan informasi sampah (pornografi)

- Akses ke database melalui perintah SQL (SQL injection).

Web Programming Python

Python adalah bahasa pemrograman dinamis yang mendukung pemrograman berorientasi obyek. Python dapat digunakan untuk berbagai keperluan pengembangan perangkat lunak dan dapat berjalan di berbagai platform sistem operasi. Seperti halnya bahasa pemrograman dinamis, python seringkali digunakan sebagai bahasa skrip dengan interpreter yang teintergrasi dalam sistem operasi. Saat ini kode python dapat dijalankan pada sistem berbasis:

- Linux/Unix
- Windows
- Mac OS X
- Java Virtual Machine
- OS/2
- Amiga
- Palm
- Symbian (untuk produk-produk Nokia)

Python didistribusikan dengan beberapa lisensi yang berbeda dari beberapa versi. Lihat sejarahnya di Python Copyright. Namun pada prinsipnya Python dapat diperoleh dan dipergunakan secara bebas, bahkan untuk kepentingan komersial. Lisensi Python tidak bertentangan baik menurut definisi Open Source maupun General Public License (GPL).

Python merupakan bahasa pemrograman yang mendukung pengembangan aplikasi berbasis desktop dan juga aplikasi berbasis web. Biasanya kalau berhubungan dengan WEB maka orang akan berfikir framework yang digunakan. Tentunya ada beberapa framework yang bisa digunakan untuk membangun aplikasi web berbasis python ini antara lain adalah Django, Web2py, Cherrypy dan lain-lain. Masing-masing framework memiliki aturan khusus dalam penulisan syntax. Framework tersebut mengadopsi struktur yang sama seperti pemrograman CGI. Untuk lebih jelasnya mari kita pelajari pemrograman CGI.

Common Gateway Interface atau disingkat CGI adalah suatu standar untuk menghubungkan berbagai program aplikasi ke halaman web. CGI mirip sebuah program komputer yang menjadi perantara antara standar HTML yang menjadikan tampilan web dengan program lain, seperti basis data (database). Hasil yang diperoleh dari proses pencarian dikirimkan kembali ke halaman web untuk ditampilkan dalam format HTML.

Python menyediakan modul CGI yang bisa digunakan untuk membuat aplikasi berbasis web. Tentunya python tidak kalah dengan pemrograman berbasis web lain seperti Java, PHP dan lain2. Mari kita lakukan percobaan untuk membuat web dengan menggunakan python.
Hal Yang paling utama sebelum membuat aplikasi adalah mempersiapkan beberapa komponen aplikasi diantaranya adalah :

1. Menginstal Program Python
2. Menginstal Program Web Server Seperti Apache2 atau Xampp
3. Setelah kedua program berhasil di instal maka langkah selanjutnya adalah mengkonfigurasi file httpd.conf yang berada pada directory web server, pada kesempatan ini saya menggunakan Xampp.
4. Buka directory Xampp dan masuk ke folder apache → Conf dan cari file httpd.conf
5. Buka file httpd.conf menggunakan notepad
6. Cari baris AddHandler cgi-script .cgi .pl .asp pada file setelah itu tambahkan extensi python seperti ini AddHandler cgi-script .cgi .pl .asp .py
7. Cari baris ¡Directory /¿ dan tambahkan ExecCGI pada list Options FollowSymLinks
8. Setelah itu simpan
9. Selanjutnya kita akan mencoba membuat halaman web dasar pada python
10. Buka Notepad dan ketikkan script dbawah ini :

```
#!/Python27/python
print ”Content-type:text/html”
print
print ’¡html¿’
print ’¡head¿’
print ’¡title¿WEB Python ¡/title¿’
print ’¡/head¿’
print ’¡body¿’
print ’¡h1¿¡center¿Tutorial Web Programming Python Bagian 1 Python¡/center¿¡/h1¿’
print
print
print ’¡h2¿¡center¿Selamat Belajar Bagi Para Pecinta Python¡/h2¿¡/center¿’
print ’¡/body¿’
```

print '¡/html¿'

pada script diatas jangan lupa menuliskan posisi directory python.exe (#!/Python27/python)

setelah itu simpan pada directory xampp folder cgi-bin dengan nama web.py (terserah nama apa saja asalhkan ekstensinya .py)

11. Buka browser dan ketikkan localhost/cgi-bin/web.py pada url dan lihatlah hasilnya

Membuat Kamus Menggunakan CGI Python

Pertama yang kita butuhkan adalah sebuah kosa kata yang akan digunakan sebagai database, kosa kata tersebut kita convert kedalam format JSON. Untuk prosesnya sebagai berikut. Buatlah sebuah kosa kata bahasa indonesia dan bahasa inggris pada excel dengan header inggris dan indonesia.

Jika sudah save as kedalam format .csv lalu di convert ke dalam format .json proses convert bisa dilakukan secara online disini dan hasilnya akan seperti berikut dan simpan dengan nama kamus.json

Selanjutnya kita mulai membuat script, buat sebuah file pada folder cgi-bin diserver localhost, tutorial ini menggunakan OS linux, ketikan script berikut.

```
#!/usr/bin/python
import cgi
import cgitb; cgitb.enable()
import simplejson as json

print "Content-type: text/html"
print

print """
¡html¿
¡head¿¡title¿CGI Script¡/title¿¡/head¿
¡body¿
 ¡h1¿ Kamus sederhana dengan cgi python¡/h1¿
 ¡form method="post" action="index.cgi"¿
  Bahasa Indonesia¡br/¿
  ¡input type="text" name="kata"/¿¡/p¿
  ¡input type="submit" name="submit" value="Terjemahkan"/¿¡/p¿
 ¡/form¿
 Bahasa Inggris¡br/¿
"""

form = cgi.FieldStorage() #variable form
cari _kata = form.getvalue("kata") #variable mengambil nilai dari input
```

```
location _database = open('/home/develop/DW/kamus.json', 'r') #membuka kosa
kata bahasa inggris
bhs _inggris = json.load(location _database)

if cari _kata:
  for bhs _indonesia in cari _kata.split(' '):
   for arti _kata in bhs _inggris:
    if arti _kata["indonesia"] == bhs _indonesia.replace(' ',''):
       hasil = arti _kata['inggris']
      break
   else:
      hasil = "arti kata tidak ditemukan"

   print """
   ¡input type="text" name="hasil" value=" %s"/¿
   ¡/body¿
   ¡/html¿
   """ % cgi.escape(hasil)
```

Jika sudah save dengan nama kamus.cgi sebagai contoh dan buka browser ketikan pada url http//localhost/cgi-bin/kamus.cgi jika muncul form input coba di tester ketikan nama kata dalam bahasa indonesia.

CHAPTER 21

DATABASES ACCESS

CHAPTER 22

SENDING EMAIL

CHAPTER 23

MULTITHREADING

Menjalankan beberapa *thread* mirip dengan menjalankan beberapa program yang berbeda secara bersamaan, namun dengan manfaat berikut :

- Beberapa *thread* dalam proses berbagi ruang data yang sama dengan benang induk dan karena dapat saling berbagi informasi atau berkomunikasi satu sama lain dengan lebih muda daripada jika prosesnya terpisah
- *thread* terkadang disebut proses ringan dan tidak membutuhkan banyak memori atas, mereka lebih murah daripada proses.

Sebuah *thread* memiliki permulaan, urutan eksekusi dan sebuah kesimpulan. Ini memiliki pointer perintah yang melacak dari mana dalam konteksnya saat ini berjalan.

- Hal ini dapat dilakukan sebelum pre-*empted* (*inturrepted*)
- Untuk sementara dapat ditunda sementara *thread* lainnya yang sedang berjalan ini disebut unggul.

1.1 Memulai *Thread* Baru

Untuk melakukan *thread* lain, perlu memanggil metode berikut yang tersedia dimodul *thread* :

Thread.start _new _thread (function, args [, kwargs])

Pemanggilan metode ini memungkinkan cara cepat dan tepat untuk membuat *thread* baru di linux dan window.

Pemanggilan metode segera kembali dan anak *thread* dimulai dan fungsi pemanggilan dengan daftar *args* telah berlalu. Saat fungsi kembali ujung *thread* akan berakhir.

Disini, *args* adalah tupel argumen. Gunakan tupel kosong untuk memanggil fungsi tanpa melewati argumen. *Kwargs* adalah kamus opsional argumen kata kunci.

Contoh :

```
#!/usr/bin/python

Import thread
Import time

# Define a function for the thread
Def print _time (threadNamw, delay):
        Count = 0
        While count ¡5:
        Time.sleep(delay)
        Count +=1
        Print " %s : %s " % (threadName, time.ctime(time.time()))

# Create two thread as follows
try:
thread.start _new _thread(print _time, ( "Thread-1 ", 2, ))
thread.start _new _thread(print _time, ( "Thread-2 ", 4,))
except:
  print "Error: unable to start thread "

while 1:
pass
```

Bila kode diatas dieksekusi, maka menghasilkan hasil sebagai berikut :

```
Thread-1 : Thu Jan 22 15:42:17 2009

Thread-1 : Thu Jan 22 15:42:19 2009

Thread-2 : Thu Jan 22 15:42:19 2009

Thread-1 : Thu Jan 22 15:42:21 2009

Thread-2 : Thu Jan 22 15:42:23 2009

Thread-1 : Thu Jan 22 15:42:23 2009

Thread-1 : Thu Jan 22 15:42:23 2009

Thread-1 : Thu Jan 22 15:42:25 2009

Thread-2 : Thu Jan 22 15:42:27 2009

Thread-2 : Thu Jan 22 15:42:31 2009

Thread-2 : Thu Jan 22 15:42:35 2009
```

Meskipun sangat efektif untuk benang tingkat rendah, namun modul *thread* sangat terbatas dibandingkan dengan modul yang baru.

1.2 Modul Threading

Modul threading yang lebih baru disertakan dengan Python 2.4 memberikan jauh lebih kuat, dukungan tingkat tinggi untuk *thread* dari modul *thread* dibahas pada bagian sebelumnya.

The *threading* modul mengekpos semua metode dari *thread* dan menyediakan beberapa metode tambahan :

- **threading.activeCount()**

  Mengembalikan jumlah objek *thread* yang aktif

- **threading.currentThread()**

  Mengembalikan jumlah objek *thread* dalam kontrol benang pemanggil

- **threading.enumerate()**

Mengembalikan daftar semua benda *thread* yang sedang aktif

Selain metode, modul *threading* memiliki *thread* kelas yang mengimplementasikan *threading*. Metode yang disediakan oleh *thread* kelas adalah sebagai berikut :

- **run()**

  Metode adalah titik masuk untuk *thread*

- **start()**

  Metode dimulai *thread* dengan memanggil metode run

- **join([time])**

  Menunggu benang untuk mengakhiri

- **isAlive()**

  Metode memeriksa apakah *thread* masih mengeksekusi

- **getName()**

  Metode mengambalikan nama *thread*

- **setName()**

  Metode menetapkan nama *thread*

1.3 Membuat *Thread* Menggunakan *Threading* Modul

Untuk melaksanakan *thread* baru menggunakan *threading* harus melakukan hal berikut :

- Mendefinisikan subclass dari *thread* kelas
- Menimpa _init _ (self [args]) metode untuk menambahkan argumen tambahan
- Menimpa run(self[args]) metode untuk menerapkan apa *thread* harus dilakukan ketika mulai

Setelah membuat baru *thread* subclass, dapat membuah seuah instance dari itu dan kemudian memulai *thread* baru dengan menerapkan *start(),* yang ada gilirinnya panggilan *run()* metode.

Contoh :

```
#!/usr/bin/python

import threading
import time

exitFlag = 0

class myThread (threading.Thread):
        def _init _(self, threadID, name, counter) :
    threading.Thread. _init _(self)
        self.threadID = threadID
        self.name = name
self.counter = counter
def run (self) :
        print "Starting " + self.name
        print _time(self.name, self.counter, 5)
        print "Exiting "+ self.name
def print _time(threadName, delay, counter):
while counter:
        if exitFlag:
                threadName.exit()
        time.sleep(delay)
        print " %s: %s " % (threadName, time.ctime(time.time()))
counter -= 1

# Create new threads
thread1 = myThread(1, "Thread-1 ", 1)
thread2 = myThread(2, "Thread-2 ", 2)

# Start new threads
thread1.start()
thread2.start()
print "Exiting Main Thread "
```

Ketika kode diatas dijalankan, menghasilkan hasil sebagai berikut:

```
Starting Thread-1
Starting Thread-2
Exiting Main Thread
Thread-1 : Thu Mar 21 09:10:03 2013
Thread-1 : Thu Mar 21 09:10:04 2013
Thread-2 : Thu Mar 21 09:10:04 2013
```

Thread-1 : Thu Mar 21 09:10:05 2013
Thread-2 : Thu Mar 21 09:10:06 2013
Thread-1 : Thu Mar 21 09:10:07 2013
Exiting Thread-1
Thread-2 : Thu Mar 21 09:10:08 2013
Thread-2 : Thu Mar 21 09:10:10 2013
Thread-2 : Thu Mar 21 09:10:12 2013
Exiting Thread=2

1.4 Sinkronisasi *Thread*

Threading modul disediakan dengan Python termasuk sederhana untuk menerapkan mekanisme bahwa memungkinkan untuk menyinkronkan *thread* penguncian. Sebuah kunci baru dibuat dengan memanggil *lock()* metode yang mengembalikan kunci baru.

The *acquire (blocking)* metode objek kunci baru digunakan untuk memaksa *thread* untuk menjalankan serempak. Opsional *blocking* parameter memungkikan untuk mengontrol apakah *thread* menunggu untuk mendapatkan kunci.

Jika *blocking* diatur ke 0, *thread* segera kembali dengan nilai 0 jika kunci tidak dapat diperoleh dan dengan 1 jika kunci dikuisisi. Jika pemblokiran diatur ke 1, blok dan menunggu kunci yang akan dirilis.

The *release()* metode objek kunci baru digunakan untuk melepaskan kunci ketika tidak lagi diperlukan.

Contoh:

```
#!/usr/bin/python

import threading
import time

class myThread (threading.Thread):
 def _init _(self, threadID, name, counter):
   threading.Thread. _init _(self)
   self.threadID = threadID
   self.name = name
   self.counter = counter
 def run(self)
   print "Starting "+ self.name
   # Get lock to synchronize threads
   ThreadLock.acquire()
   print _time(self.name, self.counter, 3)
   # Free lock to realease next thread
   ThreadLock.release()

 Def print _time(threadName, delay, counter):
  while counter:
   time.sleep(delay)
```

```
   print " %s: %s " % (threadName, time.ctime(time.time()))
   counter -= 1
 threadLock = threading.Lock()
 threads = []

# Create new threads
thread1 = myThread(1, "Thread-1,1 )
thread2 = myThread(2, "Thread-2,2 )

# Start new Threads
thread1.start()
thread2.start()

# Add threads to thread list
threads.append(thread1)
thread2.append(thread2)

# Wait for all threads to complete
Fort t in threads:
   t.join()
print "Exiting Main thread "
```

Bila kode diatas dieksekusi, maka menghasilkan sebagai berikut :

```
Starting Thread-1
Starting Thread-2
Thread-1: Thu Mar 21 09:11:28 2013
Thread-1: Thu Mar 21 09:11:29 2013
Thread-1: Thu Mar 21 09:11:30 2013
Thread-2: Thu Mar 21 09:11:32 2013
Thread-2: Thu Mar 21 09:11:34 2013
Thread-2: Thu Mar 21 09:11:36 2013
Exiting Main Thread
```

1.5 Multithreaded Antrian Prioritas

The queue modul memungkinkan untuk membuat objek antrian baru yang dapat menampung jumlah tertentu item. Ada metode berikut untuk mengontrol antrian :

- **get()**

  Menghapus dan mengembalikan item dari antrian

- **put()**

  Menambahkan item ke antrian

- **qsize()**

  Mengembalikan jumlah item yang saat ini dalam antrian

- **empty()**

  Mengembalikan benar jika antrian kosong jika tidak, salah

- **full()**

Mengembalikan benar jika antrian penuh jika tidak, salah

Contoh:

```
#!/usr/bin/python

import Queue
import threading
import time

exitFlag = 0

class myThread (threading.Thread):
  def _init_(self, threadID, name, q):
     threading.Thread. _init_(self)
    self.name = name
    self.q = q
def run(self):
      print "Starting "+ self.name
      process_data(self.name, self.q)
      print "Exiting "+ self.name

def process_data(threadName, q):
    while not exitFlag:
    queueLock.acquire()
    if not workQueu.empty():
        data = q.get()
        queueLock.release()
        print " %s processing %s " % (threadName, data)
     else:
        queueLock.release()
        time.sleep(1)

threadList = [ "Thread-1 ", "Thread-2 ", "Thread-3 "]
nameList = [ "One ", "Two ", "Three ", "Four ", "Five "]
queueLock = threading.Lock()
workLock = Queue.Queue(10)
threads = []
threadID = 1
```

```
# Create new threads
For tName in threadList:
   thread = myThread(threadID, tName, workQueue)
   thread.start()
   thread.append(thread)
   threadID +=1

# Fill the queue
queueLock.acquire()
for word in nameList:
   workQueue.put(word)
queueLock.release()

# Wait for queue to empty
while not workQueue.empty():
pass

# Notify threads it?s time to exit
exitFlag = 1

# Wait for all threads to complete
For t in threads:
   t.join()
print ”Exiting Main Thread ”
```

Bila kode diatas dieksekusi, maka menghasilkan hasil sebagai berikut:

```
Starting Thread-1
Starting Thread-2
Starting Thread-3
Thread-1 processing One
Thread-2 processing Two
Thread-3 processing Three
Thread-1 processing Four
Thread-2 processing Five
Exiting Thread-3
Exiting Thread-1
Exiting Thread-2
Exiting Main Thread
```

CHAPTER 24

XML PROCESSING

Python XML Processing

XML adalah bahasa open source portable yang mungkinkan pemrogram mengemangkan aplikasi yang dapat dibaca oleh aplikasi lain, terlepas dari sistem operasi dan bahasa pengembangnya.

Apa itu XML?

Extensible Markup Languange (XML) adalah bahasa markup seperti HTML atau SGML. Ini direkomendasikan oleh World Wide Web Consortium dan tersedia sebagai standar terbuka.

XML sangat berguna untuk mencatat data berukuran kecil dan menengah tanpa memerlukan tulang punggung berbasis SQL.

2.1 Arsitektur Parsing XML dan API

Perpustakaan standar Python menyediakan seperangkat antarmuka minimal tapi berguna untuk bekerja dengan XML.

Dua API yang paling dasar dan umum digunakan untuk data XML adalah antarmuka SAX dan DOM.

API sederhana untuk XML (SAX): mendaftarkan panggilan kemali untuk acara yang diminati dan kemudian membiarkan parser berjalan melalui dokumen. Ini berguna bila dokumen berukuran besar atau memiliki keterbatasan memori, ini memparsing file tidak pernah tersimpan dalam memori.

API Document Objek Model (DOM): ini adalah rekomendasi World Wide Web Consortium dimana keseluruhan file dibaca ke memori dan disimpan dalam bentuk hierarkies (tree-based) untuk mewakili semua fitur dokumen XML.

SAX jelas tidak bisa memproses informasi secepat DOM saat bisa bekerjadengan file besar. Di sisi lain, menggunakan DOM secara eklusifenar-benar dapat membunuh sumber daya, terutama jika digunakan pada banyak file kecil.

SAX hanya bisa dibaca sementara DOM mengizinkan perubahan pada file XML. Kedua API yang berbeda ini saling melengkapi satu sama lain, tidak ada alasan mengapa tidak dapat menggunakannya untuk proyek besar.

Contoh:

```
¡collection shelf=”New Arrivals”¿
¡movie title=”Enemy Behind”¿
  ¡type¿War, Thriller¡/type¿
  ¡format¿DVD¡/format¿
  ¡year¿2003¡/year¿
  ¡rating¿PG¡/rating¿
  ¡stars¿10¡/stars¿
  ¡description¿Talk about a US-Japan war¡/description¿
¡/movie¿
¡movie title=”Transformers”¿
  ¡type¿Anime, Science Fiction¡/type¿
  ¡format¿DVD¡/format¿
  ¡year¿1989¡/year¿
  ¡rating¿R¡/rating¿
  ¡stars¿8¡/stars¿
  ¡description¿A schientific fiction¡/description¿
¡/movie¿
  ¡movie title=”Trigun”¿
  ¡type¿Anime, Action¡/type¿
  ¡format¿DVD¡/format¿
  ¡episodes¿4¡/episodes¿
  ¡rating¿PG¡/rating¿
  ¡stars¿10¡/stars¿
  ¡description¿Vash the Stampede!¡/description¿
¡/movie¿
¡movie title=”Ishtar”¿
  ¡type¿Comedy¡/type¿
  ¡format¿VHS¡/format¿
  ¡rating¿PG¡/rating¿
  ¡stars¿2¡/stars¿
```

¡description¿Viewable boredom¡/description¿
¡/movie¿
¡/collection¿

2.2 Parsing XML dengan API SAX

SAX adalah antarmuka standar untuk parsing XML berbasis event. Parsing XML dengan SAX umumnya mengharuskan untuk membuat C*ontrolHandler* dengan subclassing xml.sax *controlhandler*.

ControlHandler menangani tag dan atribut tertentu dari XML. Objek *ControlHandler* menyediakan metode untuk menangani berbagai aktivitas parsing. Parsing memanggil metode *ControlHandler* saat memparsing file XML.

Metode *startDocument* dan *endDocument* disebut awal dan akhir setiap elemen. Jika parsing tidak dalam mode namespace, metode *startElement* (tag attribute) dan *endElement* (tag) dipanggil. Jika tidak, metode yang sesuai *startElemenNS* dan *endElemenNS* dipanggil. Disini, tah adalah tag elemen dan atriut adalah atribut.

Berikut ini metode penting untuk memahami sebelum melanjutkan ke materi berikutnya :
Metode

Metode berikut membuat objek parsing baru dan mengembalikannya. Objek parsing diuat akan menjadi tipe parsing pertama yang ditemukan sistem.

xml.sax.make _parser([parser _list])

Berikut adalah detail parameternya :

Parser _list : pilihan argumen yang terdiri dari daftar parsing untuk digunakan yang semuanya harus menerapkan metode *make _parse*

Metode

Metode berikut membuat parsing SAX dan menggunakannya untuk mengurai dokumen

xml.sax.parser(xmlfile, contenthandler[, errorhandler])

Berikut adalah detail dari parameternya:

- *Xmlfile*

  Ini adalah nama file XML yang bisa dibaca.

- *ContentHandler*

  Ini harus menjadi objek *ContenHandler*

- *ErrorHandler*

  Jika ditentukan, e*rrorhandler* harus menjadi objek *ErrorHandler* SAX

- Metode *parseString*

Membuat parsing SAX dan mengurai string XML yang ditentukan :

xml.sax.parsertring(xmlstring,contenthandler[, errorhandler])

Brikut ini adalah detail nama dar parameter :

XMLstring

Nama dari string yang bisa dibaca

ContentHandler

Menjadi objek ContenHandler

ErrorHandler

Menjadi objek ErorHandler SAX

Contoh :

```
#!/usr/bin/python

import xml.sax

class MovieHandler( xml.sax.ContentHandler ):
  def __init__(self):
    self.CurrentData = ""
    self.type = ""
    self.format = ""
    self.year = ""
    self.rating = ""
    self.stars = ""
    self.description = ""

  # Call when an element starts
  def startElement(self, tag, attributes):
    self.CurrentData = tag
    if tag == "movie":
      print "*****Movie*****"
      title = attributes["title"]
      print "Title:", title

  # Call when an elements ends
  def endElement(self, tag):
    if self.CurrentData == "type":
      print "Type:", self.type
    elif self.CurrentData == "format":
      print "Format:", self.format
    elif self.CurrentData == "year":
      print "Year:", self.year
    elif self.CurrentData == "rating":
      print "Rating:", self.rating
    elif self.CurrentData == "stars":
      print "Stars:", self.stars
```

```
      elif self.CurrentData == ”description”:
        print ”Description:”, self.description
      self.CurrentData = ””

    # Call when a character is read
    def characters(self, content):
      if self.CurrentData == ”type”:
        self.type = content
      elif self.CurrentData == ”format”:
        self.format = content
      elif self.CurrentData == ”year”:
        self.year = content
      elif self.CurrentData == ”rating”:
        self.rating = content
      elif self.CurrentData == ”stars”:
        self.stars = content
      elif self.CurrentData == ”description”:
        self.description = content

if ( __name__ == ”__main__”):

    # create an XMLReader
    parser = xml.sax.make_parser()
    # turn off namepsaces
    parser.setFeature(xml.sax.handler.feature_namespaces, 0)

    # override the default ContextHandler
    Handler = MovieHandler()
    parser.setContentHandler( Handler )

    parser.parse(”movies.xml”)
```

Ini akan menghasilkan hasil sebagai berikut:

```
*****Movie******
*****Movie*****
Title: Enemy Behind
Type: War, Thriller
Format: DVD
Year: 2003
Rating: PG
Stars: 10
Description: Talk about a US-Japan war
*****Movie*****
Title: Transformers
```

Type: Anime, Science Fiction
Format: DVD
Year: 1989
Rating: R
Stars: 8
Description: A schientific fiction
*****Movie*****
Title: Trigun
Type: Anime, Action
Format: DVD
Rating: PG
Stars: 10
Description: Vash the Stampede!
*****Movie*****
Title: Ishtar
Type: Comedy
Format: VHS
Rating: PG
Stars: 2

2.3 Parsing XML dengan API DOM

Document Ovject Model (DOM) adalah API lintas bahasa dari World Wide Web Consortium (W3C) untuk mengakses dan memodifikasi dokumen XML.

DOM sangat berguna untuk aplikasi akses acak. SAX hanya memungkinkan melihat satu bit dokumen sekaligus. Jika melihat satu elemen SAX, tidak memiliki akses ke yang lain.

Berikut adalah cara termudah untuk memuat dokumen XML dengan cepat dan membuat objek minidom menggunakan modul xml.dom. Objek minidom menyediakan metode parsing sederhana yang dengan cepat memuat pohon DOM dari file XML.

Contoh frase memanggil fungsi parsing (file [,parsing]) dari objek minidokumen untuk mengurai file XML yang ditunjuk oleh file ke objek pohon DOM.

```
#!/usr/bin/python

from xml.dom.minidom import parse
import xml.dom.minidom

# Open XML document using minidom parser
DOMTree = xml.dom.minidom.parse("movies.xml")
collection = DOMTree.documentElement
if collection.hasAttribute("shelf"):
  print "Root element : %s" % collection.getAttribute("shelf")

# Get all the movies in the collection
movies = collection.getElementsByTagName("movie")
```

```
# Print detail of each movie.
for movie in movies:
  print "*****Movie*****"
  if movie.hasAttribute("title"):
    print "Title: %s" % movie.getAttribute("title")

  type = movie.getElementsByTagName('type')[0]
  print "Type: %s" % type.childNodes[0].data
  format = movie.getElementsByTagName('format')[0]
  print "Format: %s" % format.childNodes[0].data
  rating = movie.getElementsByTagName('rating')[0]
  print "Rating: %s" % rating.childNodes[0].data
  description = movie.getElementsByTagName('description')[0]
  print "Description: %s" % description.childNodes[0].data
```

Ini akan menghasilkan hasil sebagai berikut :

```
Root element : New Arrivals
*****Movie*****
Title: Enemy Behind
Type: War, Thriller
Format: DVD
Rating: PG
Description: Talk about a US-Japan war
*****Movie*****
Title: Transformers
Type: Anime, Science Fiction
Format: DVD
Rating: R
Description: A schientific fiction
*****Movie*****
Title: Trigun
Type: Anime, Action
Format: DVD
Rating: PG
Description: Vash the Stampede!
*****Movie*****
Title: Ishtar
Type: Comedy
Format: VHS
Rating: PG
Description: Viewable boredom
```

2.4 Membangun Parsing Document XML menggunakan Python

Python mendukung untuk bekerja dengan berbagai bentuk markup data terstruktur. Selain mengurai xml.etree. *ElementTree* mendukung pembuatan dokumen

XML yang terbentuk dengan baik dari objek elemen yang dibangun dalam aplikasi. Kelas elemen digunakakan saat sebuah dokumen diurai untuk mengetahui bagaimana menghasilkan bentuk serial dari isinya kemudian dapat ditulis ke sebuah file.

Untuk membuat instance elemeb gunakan fungsi elemen contructor dan *SubElemen()* pabrik.

```
Import xml.etree.ElementTree as xml

filename = ”/home/abc/Desktop/test _xml.xml ”
toot = xml.Element( ”Users ”)
userelement = xml.Element( ”user ”)
root.append(userelement)
```

Bila menjalankan ini, akan menghasilkan sebagai berikut :

```
¡Users¿
        ¡user¿
        ¡user¿
¡/Users¿
```

Tambahkan anak-anak pegguna

```
Uid = xml.SubElement(userelement, ”uid ”)
Uid.text = ”1 ”

FirstName = xml.SubElement(userelement, ”FirstName ”)
FirstName.text = ”testuser ”

LastName = xml.SubElement(userelement, ”LastName ”
LastName.text = ”testuser ”

Email = xml.SubElement(userelement, ”Email ”)
Email.text = mailto:testuser@test.comtestuser@test.com

state = xml.SubElement(userelemet, ”state ”)
state.text = ”xyz ”

location = xml.SubElement(userelement, ”location)
location.text = abc

tree = xml.ElementTree(root)
with open(filename, ”w ”) as fh:
tree.write(fh)
```

Pertama buat elemen root dengan mengunakan fungsi *ElementTree*. Kemudian membuat elemen pegguna dan menambahkannya ke root. Selanjutnya membuat

SubElement dengan melewatkan elemen pengguna (userelement) ke *SubElemen* beserta namanya seperto "FirstName ". Kemudian untuk setiap *SubElement* tetapkan properti teks untuk memberi nilai. Di akhir, membuat *ElementTree* dan menggunakannya untuk menulis XML ke file.

Jika menjalankan ini akan menjadi sebagai berikut :

¡users¿
    ¡user¿
        ¡uid¿1¡/uid¿
        ¡FirstName¿testuser¡/FirstName¿
        ¡LastName¿testuser¡/LastName¿
        ¡state¿xyz¡/state¿
        ¡location¿abc¡/location¿
    ¡/user¿
¡/Users¿

Parsing XML Documen :

```
import xml.etree.ElementTree as ET
tree = ET.parse(Your _XML _file _path)
root = tree.getroot()
```

Disini *getroot()* akan mengembalikan elemen dari dokumen XML

```
¡Users version= ”1.0 ” languange= ”SPA ”¿
        ¡user¿
                ¡uid¿1¡/uid¿
                ¡FirstName¿testuser¡/FirstName¿
                ¡LastName¿testuser¡/LastName¿
                ¡Email¿testuser@tes.com/Email¿
                ¡state¿xyz¡/state¿
                ¡location¿abc¡/location¿
        ¡/user¿
¡/Users¿
```

CHAPTER 25

GUI PROGRAMMING

Python menyediakan berbagai pilihan untuk mengembangkan antarmuka pengguna grafis (GUIs). Yang paling tercantum dibawah ini :

- Tkinter

  Antarmuka Python ke toolkit Tk GUI dikirimkan dengan Python.

- wxPython

  antarmuka Python open-source untuk wxWindows

- Jpython

  Port Python untuk java yang memberikan Python script akses tanpa batas ke perpustakaan kelas java pada mesin lokal

3.1 Tkinter Pemrograman

Tkinter adalah perpustakaan GUI standar untuk Python. Python bila dikombinasikan dengan Tkinter menyediakan cara yang mudah dan cepat untuk membuat aplikasi GUI. Tkinter menyediakan antarmuka berorientasi ojek yang kuat untuk toolkit Tk GUI.

Membuat aplikasi GUI menggunakan Tkinter adalah tugas yang mudah. Yang diperlukan adalah melakukan langkah-langkah sebagai berikut :

- Mengimpor Tkinter modul
- Buat jendela utama aplikasi GUI
- Tambahkan satu atau lebih dari widget tersebut diatas ke aplikasi GUI
- Masukkan acara loop utama untuk mengambil tindakan terhadap setiap peristiwa dipicu oleh pengguna

Contoh :
#!/usr/bin/python

import Tkinter
top = Tkinter.Tk()
# Code to add widgets will go here...
top.mainloop()

3.2 Tkinter Widget

Tkinter menyediakan berbagai kontrol seperti tombol, label dan kotak teks yang digunakan dalam aplikasi GUI. Kontrol ini biasanya disebut widget.

Saat ini ada 15 jenis widget di Tkinter. Menyajikan widget serta penjelasan singkat pada tabel berikut ini :

Table 25.1 Ukuran

Operator	Penjelasan
Button	SMenampilkan tombol dalam aplikasi
Canvas	Menggambar bentuk seperti garis, oval, poligon dan persegi panjang dalam aplikasi
Checkbutton	Menampilkan sejumlah pilihan sebagai kotak centang. Pengguna dapat memilih beberapa pilihan pada suatu waktu
Entry	Menampilka bidang garis teks tunggal untuk menerima nilai-nilai dari pengguna
Frame	Wadah untuk mengatur widget lainnya
Label	Memberikan keterangan garis single untuk widget lainnya. Hal ini berisi gambar
Listbox	Menyediakan daftar pilihan kepada pengguna
Menubutton	Menampilkan menu dalam aplikasi
Menu	Memberikan berbagai perintah untuk pengguna. Perintah-perintah ini terkandung di dalam MenuButton
Message	Menampilkan bidang teks multiline untuk menerima nilai-nilai dari pengguna
RadioButton	Menampilkan sejumah pilihan sebagai tombol radio. Pengguna dapat memilih hanya satu pilihan pada suatu waktu
Scale	Menyediakan widget slide
Scrollbar	Menambah kemampuan bergulir ke berbagai widget seperti kotak daftar
Text	Menampilka teks dalam beberapa garis
Toplevel	Menyediakan wajah jendela terpisah
PanedWindow	Wadah yang mengandung sejumlah panel disusun horizontal atau vertikal
LabelFrame	Wadah widget sederhana. Bertindak sebagai spacer atau wajah untuk layout jendela kompleks
TkMessageBox	Menampilkan kotak pesan dalam aplikasi
Spinbox	Memilih sejumlah tetap nilai-nilai

Beberapa atribut umu sebagai ukuran, warna dan font ditentukan. Berikut adalah beberapa atriut standarm :

1. Ukuran

Berbagai panjang, lebar, dan dimensi lain dari widget digambarkan dalam banyak unit yang berbeda seperti :

- Jika menetapkan dimensi ke integer diasumsikan dalam piksel
- Menentukan unit dengan menentukan dimensi untuk string yang berisi sejumlah diikuti oleh :

Table 25.2 Ukuran

Karakter	Penjelasan
c	Sentimeter
i	Inci
m	Milimeter
p	Poin printer

Tkinter mengungkapkan panjang sebagai integer jumlah pik-

sel. Berikut ini adalah daftar pilihan panjang umum:

- borderwidth

  Lebar batas yang memberikan tampilan tiga dimensi untuk widget

- highlightthickness

  Lebar puncak persegi panjang ketika widget memiliki fokus

- padX padY

  Ruang tambahan widget dari manajer tata letak luar minimum widget perlu menampilkan isinya di x dan y arah

- selectborderwidth

  Lebar perbatasan tiga dimensi disekitar dipilih item widget

- wraplength

  Panjang garis maksimum untuk widget yang melakukan kata membungkus

- height

  Tinggi diinginkan widget

- underline

  Indeks karakter untuk menggarisawahi dalam teks widget

- width
- Lebar diinginkan widget

Warna
Tkinter memiliki warna dengan string. Ada dua cara umum untuk menentukan warna di Tkiter, yaitu :

- Menggunakan string menentukan proporsi merah, hijau dan biru didigit heksadesimal. Misalnya " #ffff " putih, " #000000 " hitam dan " #000fff000 " hijau.
- Menggunakan lokal standar nama warna . warna-warna "white ", "black ", "green " dan "magenta " akan selalu tersedia.

Pilihan warna umum :

- activebackground

  Warna latar berlakang untuk widget ketika widget aktif

- activeforeground

  Warna depan untuk widget ketika widget aktif

- background

  Merepresentasikan sebagai *bg*

- disableforeground

  Warna depan untuk widget ketika widget dinonaktifkan

- foreground

  Merepresentasikan fg

- highlightbackground

  Warna latar belakang dari daerah puncak ketika widget memiliki fokus

- hightlightcolor

  Warna depan dari wilayah puncak ketika widget memiliki fokus

- selectbackground

  Warna latar belakang untuk item yang dipilih dari widget

- selectforeground

  Warna depan untuk item yang dipilih dari widget

- Font

  Sebagai tupel yang elemen pertama adalah keluarga font diikuti dengan string yang berisi satu atau lebih gaya pengubah tebal,miring, garis bawah dan overstrike.

  Contoh :

- (”Helvetica ”, ”16 ”-point Helvetica biasa
- (”Times ”, ”24 ”, ”beranimiring ”) untuk 24-point kali miring tebal

Dapat membuat ”font object ” dengan mengimpor modul tkFont dan menggunakan kelas konstruktor font nya :

```
Import tkFont
Font = tkFont.Font (option, ....)
```

Berikut adalah daftar pilihan :

- Family

  Font nama keluarga sebagai string

- Size

  Font tinggi sebagai integer dalam poin

- Weight

  Bold untuk teal, normal untuk berat badan secara teratur

- Slant

  Italic untuk miring, roman untuk unstlanted

- Underline

  1 untuk teks yang digarisbawahi, 0 untuk normal

- Overstrike

  1 untuk teks telak, 0 untuk normal

  Jika berjalan di bawah X window system, dapat menggunakan salah satu nama font X. Sebagai contoh, font bernama ”-*lucidatypewriter-medium-r-*-*-*-140-*-*-* ” adalah favorit fixed-width font penulis untuk digunakan pada layar.

- Jangkar

  Jangkar digunakan untuk mendefinisikan mana teks diposisikan relatif terhadap titik acuan. Berikut adalah daftar kemungkinan konstanta yang dapat digunakan :

- NW
- N
- NE
- W
- TENGAH

- E
- SW
- S
- SE

Jika menggunakan tengah sebagai jangkar tek, tek akan ditengahkan horizontal dan vertikal disekitar titik referensi.

Jangkar NW akan posisi teks sehingga titik referensi bertepatan dengan laut sudut kotak berisi teks

Jangakr W akan pusat teks secara vertikal disekitar titik referensi dengan tepi kiri kotak teks yang melewati titik itu dan sebagainya.

Jika membuat widget kecil didalam bingkai besar dan menggunakan jangkar = SE pilihan, widget akan ditempatkan disudut kanan bawah gambar. Jika menggunakan anchor = N sebaliknya widget akan dipusatkan disepanjang tepi atas.

Gaya relief

Widget mengacu pada efek 3-D simulasi terbaru disekitar bagian luar widget. Berikut adalah daftar konstanta yang mungkin dapat digunakan untuk atribut:

- Datar
- Dibesarkan
- Cekung
- Alur
- Punggung bukit

Contoh :

```
From Tkinter import *
Import Tkinter

top = Tkinter.Tk()
B1 = Tkinter.Button(top, text= ”FLAT ”, relief=FLAT)
B2 = Tkinter.Button(top, text= ”RAISED ”, relief=RAISED)
B3 =Tkinter.Button(top, text= ”SUNKEN ”, relief=SUNKEN)
B4=Tkinter.Button(top, text= ”GROOVE ”, relief=GROOVE)
B5=Tkinter.Button(top, text= ”RIDGE ”, relief=RIDGE)

B1.pack()
B2.pack()
B3.pack()
B4.pack()
B5.pack()
```

top.mainloop()

Britmaps

Ada beberapa jenis bitmap yang tersedia, diantaranya:

- Kesalahan
- Gray75
- Gray50
- Gray12
- Jam Pasir
- Info
- Questhead
- Perantanyaan
- Peringatan

Contoh:

```
From Tkinter import *
Import Tkinter

Top = Tkinter.Tk()

B1 = Tkinter.Button(top, text = "error ", relief=RAISED, n bitmap= "error ")
B2 = Tkinter.Button(top, text = "hourglass ", relief=RAISED, n bitmap= "hourglass
")
B3 = Tkinter.Button(top, text = "info ", relief=RAISED, n bitmap= "info ")
B4 = Tkinter.Button(top, text = "question ", relief=RAISED, n bitmap= "question
")
B5 = Tkinter.Button(top, text = "warning ", relief=RAISED, n bitmap= "warning
")

B1.pack()
B2.pack()
B3.pack()
B4.pack()
B5.pack()
top.mainloop()
```

Kursor

Berikut daftar menarik :

- Panah
- Lingkaran

- Jam
- Menyebrang
- Dotbox
- Bertukar
- Fluer
- Jantung
- Manusia
- Tikus
- Bajak laut
- Tamah
- Antar jemput
- Perekat
- Laba-laba
- Kaleng semprot
- Bintang
- Target
- Tcross
- Melakukan perjalanan
- Menonton

Contoh :
From Tkinter import *
Import Tkinter

Top = Tkinter.Tk()

B1 = Tkinter.Button(top, text = ”circle ”, relief=RAISED, n bitmap= ”circle ”)
B2 = Tkinter.Button(top, text = ”plus ”, relief=RAISED, n bitmap= ”plus ”)

B1.pack()
B2.pack()
top.mainloop()

3.3 Manajemen Geometri

Semua widget tkinter memiliki akses ke metode manajemen geometri tertentu, yang memiliki tujuan menggorganisir widget diseluruh wilayah widget induk. Tkinter mengekspos kelas manager geometri berikut :

- Metode the *pack()*

  Manajer geometri ini mengatur widget diblok sebelum menempatkan mereka di widget induk

- Metode the *grid()*

  Manajer geometri ini mengatur widget dalam struktur tabel seperti di widget induk

- Metode the *place()*

Manajer geometri ini mengatur widget dengan menempatkan dalam posisi tertentu dalam widget induk

3.4 Manfaat Tkinter

Tkinter sangat sederhana. Erikut manfaat Tkinter dibandingkan GUI toolkit :

- Tkinter mudah diakses oleh siapa saja. (Accessibilty) Tkinter merupakan toolkit

  yang ringan dan satu-satunya solusi GUI yang paling sederhana untuk Python sampai saat ini. Cukup menuliskan beberapa baris kode Python untuk membuat aplikasi GUI sederhana dengan Tkinter. Untuk menambahkan komponen baru pada Tkinter, dapat membuatnya dalam kode Python atau menambahkan paket ekstensi seperti Pmw, Tix, atau ttk.

- Tkinter mudah digunakan di semua platform (Portability) Sebuah program Python

  yang dibangun menggunakan Tkinter dapat berjalan dengan baik di semua platform sistem operasi seperti Microsoft Windows, Linux, dan Macintosh. Dan dari segi tampilan window, akan terlihat sama dengan standar platform yang digunakan.

- Tkinter selalu tersedia di Python (Availability) Tkinter merupakan modul stan-

  dar pada pustaka Python. Sebagian besar paket instalasi Python sudah langsung berisi Tkinter. Khusus untuk beberapa distro Linux, perlu menambahkan paket Tkinter secara terpisah. Pada Windows, bisa langsung menggunakan Tkinter sesaat setelah menginstal paket instalasi Python.

- Dokumentasi Tkinter sangat LUAR BIASA (Documentation) Python (plus Tk-

  inter) ini bersifat open-source, maka banyak sekali komunitas-komunitas yng membahas Python dan Tkinter dan bisa belajar dan bertanya langsung dengan para ahli.

CHAPTER 26

FUTHER EXPRESSION

FURTHER EXPRESSION

Stiap kode yang dituliskan menggunakan bahasa yang dikompilasi seperti C, C++ atau Java dapat diintegrasikan ke skrip Python lainnya. Kode ini diagnggap sebagai ektensi.

Modul ekstensi Python tidak lebih dari sekedar perpustakaan C biasa. Pada mesin Unix, perpustakaan ini biasanya diakhiri dengan .so (untuk objek bersama). Pada mesin windows, biasanya melihat .dll (untuk perpusatkaan yang terhubung secara dinamis).

4.1 Pra-Persyaratan untuk Menulis Ekstensi

Untuk memulai ektensi, memerlukan file header Python. Pada mesin Unix, biasanya memerlukan instalasi paket khusus pengembang seperti python 2-5.

Pengguna window mendapatkan header ini sebagai bagian dari paket saat menggunakan pemasang Python biner.

Harus memiliki pengetahuan yang baik tentang C atau C++ untuk menulis ekstensi Python menggunakan pemrograman C.

Untuk melihat modul ekstensi Python, perlu mengelompokkan kode menjadi empat bagian :

- File header Python h
- Fungsi C yang ingin ditampilkan sebagai antarmuka dari modul
- Sebuah tabel memetakan nama-nama fungsi saat pengembang Python melihat ke fungsi C didalam modul ekstensi
- Fungsi inilisasi

Perlu menyertakan file header Python.h di file sumber C memberi akses ke API Python internal digunakan untuk menghitung modul ke penerjamah.

Menyertakan header Python.h sebelum header lain yang mungkin dibutuhkan. Mengikuti termasuk dengan fungsi yang ingin dipanggil dari Python.

Tanda tangan penerapan C fungsi selalu mengambil salah satu dari tiga bentuk berikut :

```
static PyObject *MyFunction( PyObject *self, PyObject *args );
 static PyObject *MyFunctionWithKeywords(PyObject *self,

                    PyObject *args,
                    PyObject *kw);
static PyObject *MyFunctionWithNoArgs( PyObject *self );
```

Masing-masing deklarasi seelumnya mengembalikan objek Python. Tidak ada yang namanya fungsi void dengan Python seperti ada di C. Jika ingin fungsi mengembalikan nilai, Python. Header Python mendefinisikan makro. Py _Return _None yang melakukan ini.

Nama-nama fungsi C bisa menjadi apapun yang disuka karena tidak pernah diluar modul ekstensi mendefinisikan sebagai statis.

Fungi Cbiasanya diberi nama dengan menggabnungkan modul dan fungsi Python bersama-sama yang ditunjukan disini :

```
static PyObject *module _func(PyObject *self, PyObject *args) {
  /* Do your stuff here. */
  Py _RETURN _NONE;
}
```

Ini adalah fungsi Python yang disebut func didalam modul-modul. Memasukkan petunjuk ke fungsi C ke dalam tabel metode untuk modul yang biasanya muncul selanjutnya dikode sumber tael pemetaan metode.

Tabel metode ini adalah susunan sederhana dari struktur PyMethodSef. Struktur itu terlihat seperti ini :

```
struct PyMethodDef {
```

```
  char *ml_name;
  PyCFunction ml_meth;
  int ml_flags;
  char *ml_doc;
};
```

Inilai uraian anggota struktur ini :

- Ml_name

  Nama fungsi yang digunakan penafsir Python saat digunakan dalam program Python

- Ml_meth

  Menjadi alamaat ke fungsi yang memiliki salah satu tanda tangan yang dijelaskan dalam penelusuran sebelumnya

- Ml_flags

  Memberitahu penafsir yang mana dari tiga tanda tangan yang digunakan ml_meth. Bendera ini biasanya mmiliki nilai meth_varargs. Bendera ini dapat digandakan dengan ored dengan meth_keywords jika ingin memiarkan argumen kata kunci masuk ke fungsi. Ini juga bisa memiliki nilai meth_noargs yang menunjukan bahwa tidak ingin menerima argumen apa pun.

- Ml_doc

  Ini adalah docstring untuk fungsi yang bisa jadi NULL jika tidak ingin menulisnya.

Tabel ini perlu diakhiri dengan sentinel yang terdiri dari NULL dan 0 untuk anggota yang sesuai.

Contoh:

```
static PyMethodDef module_methods[] = {
  { "func", (PyCFunction)module_func, METH_NOARGS, NULL },
  { NULL, NULL, 0, NULL }
};
```

Bagian terakhir dari modul ekstensi adalah fungsi inialisasi. Fungsi ini dipanggil oleh juru bahasa Python saat modul diisikan. Hal ini diperlukan agar fungsi diberi nama intiModule dimana modul adalah nama modul.

Fungsi inialisasi perlu diekspor dari perpustakaan yang akan dibangun. Header Python mendefinisikan PyMODINIT_Func untuk memasukkan

mantra yang sesuai agar terjadi pada lingkungan tertentu tempat menyuusun. Yang harus dilakukan adalah mengunakan saat menentukan fungsinya.

Fungsi inialisasi C umumnya memiliki strktur keseluruhan berikut :

PyMODINIT _FUNC init*Module*() {

Py _InitModule3(*func*, *module* _methods, ”docstring...”);

}

Berikut adalah penjelasan fugsi Py _IntiModule :

- Func

  Ini adalah fungsi yang akan diekspor

- Module

  Ini adalah nama tabel pemetaan yang didefinisikan diatas

- Docstring

Ini adalah komentar yang ingin diberikan diekstensi

Menempatkan ini semua bersama-sama terlihat sebagai berikut :

```
#include ¡Python.h¿

static PyObject *module _func(PyObject *self, PyObject *args) {
  /* Do your stuff here. */
  Py _RETURN _NONE;
}

static PyMethodDef module _methods[] = {
  { ”func”, (PyCFunction)module _func, METH _NOARGS, NULL },
  { NULL, NULL, 0, NULL }
};

PyMODINIT _FUNC initModule() {
  Py _InitModule3(func, module _methods, ”docstring...”);
}
```

Contoh :

```
#include ¡Python.h¿

static PyObject* helloworld(PyObject* self)
{
   return Py _BuildValue(”s”, ”Hello, Python extensions!!”);
}
```

```
static char helloworld _docs[] =
   "helloworld( ): Any message you want to put here!! $n$n";

static PyMethodDef helloworld _funcs[] = {
   {"helloworld", (PyCFunction)helloworld,
    METH _NOARGS, helloworld _docs },
   {NULL }
};

void inithelloworld(void)
{
   Py _InitModule3("helloworld", helloworld _funcs,
               "Extension module example!");
}
```

Disini fungsi Py _BuildValue digunakan untuk membangun nilai Python.

4.2 Membangun dan Menginstal Ekstensi

Distutils paket membuatnya sangat mudah mendistribusikan modul Python, baik Python murni dan modul ekstensi dengan cara standar. Modul didistribusikan dalam bentuk sumber dan dibangun dan diinstal melalui skrip setup yang iasa disebut seup.py sebagai berikut :

```
from distutils.core import setup, Extension
setup(name='helloworld', version='1.0', $n$
    ext _modules=[Extension('helloworld', ['hello.c'])]).
```

Sekarang gunakan perintah berikut yang aka melalakukan semua kompilasi dan langkah penghubunh yang diperlukan dengan perintah dan bendera penyusun dan penghubung yang benar dan menyalin perpustakaan dinamis yang dihasilkan ke dalam direktori yang sesuai .

Contoh :

```
$ python setup.py install
```

Pada sistem berbasis Unix kemungkinan besar perlu menjalankan perintah ini sebagai root agar meminta izin untuk menulis ke direktori paket situs. Ini biasanya tidak menjadi masalah pada window.

Setelah menginstal ekstensi, akan dapat mengimpor dan memanggil ekstensi tersebut di skrip Python sebagai berikut :

```
#!/usr/bin/python
import helloworld
```

print helloworld.helloworld()

Ini akan menghasilkan hasil sebagai berikut :
Hello, Python extensions!!

Seperti kemungkinan besar ingin mendefinisikan fungsi yang menerima argumen, dapat menggunakan salah satu tanda tangan lain untuk fungsi C. Sebagai contoh, fungsi berikut yang menerima beberapa parameter akan didefinisikan seperti ini :

```
static PyObject *module_func(PyObject *self, PyObject *args) {
  /* Parse args and do something interesting here. */
  Py_RETURN_NONE;
}
```

Tabel metode yang berisi entri untuk fungsi baru akan terlihat seperti ini :

```
static PyMethodDef module_methods[] = {
  { "func", (PyCFunction)module_func, METH_NOARGS, NULL },
  { "func", module_func, METH_VARARGS, NULL },
  { NULL, NULL, 0, NULL }
};
```

Menggunakan fungsi API PyArg_ParseTuple untuk mengekstrak argumen dari satu pointer PyObject yang dikirimkan ke fungsi C. Argumen pertama untuk PyArg_ParseTuple adalah args argumen. Ini adalah objek yang akan parsing. Argumen kedua adalah string format yang menggambarkan argumen saat mengharapkannya muncul. Setiap argumen diwakili oleh satu atau lebih karakter dalam format string sebagai berikut :

```
static PyObject *module_func(PyObject *self, PyObject *args) {
  int i;
  double d;
  char *s;

  if (!PyArg_ParseTuple(args, "ids", &i, &d, &s)) {
    return NULL;
  }

  /* Do something interesting here. */
  Py_RETURN_NONE;
}
```

Mengkompilasi versi baru dari modul dan mengimpornya memungkinkan untuk memanggil fungsi baru dengan sejumlah argumen dari jenis apa pun :

```
module.func(1, s="three", d=2.0)
module.func(i=1, d=2.0, s="three")
module.func(s="three", d=2.0, i=1)
```

Berikut adalah tanda tangan standar untuk fungsi PyArg _ParseTuple:
int PyArg _ParseTuple(PyObject* tuple,char* format,...)

Fungsi ini mengembalikan 0 untuk kesalahan, dan nilai tidak sama dengan 0 untuk kesuksesan. Tuple adalah PyObject * yang merupakan argumen kedua dari fungsi C. Format berikut adalah string C yang menggambarkan argumen wajib dan opsional. Berikut adalah daftar kode format untuk fungsi PyArg _ParseTuple:

Table 26.1 Ukuran

Karakter	Penjelasan
c	Sentimeter
i	Inci
m	Milimeter
p	Poin printer

Py _BuildValue mengambil format string seperti PyArg _ParseTuple. Alih-alih menyampaikan alamat nilai yang sedang bangun, melewati nilai sebenarnya. Berikut adalah contoh yang menunjukkan bagaimana menerapkan fungsi tambah :

```
static PyObject *foo _add(PyObject *self, PyObject *args) {
  int a;
  int b;

  if (!PyArg _ParseTuple(args, ”ii”, &a, &b)) {
    return NULL;
  }
  return Py _BuildValue(”i”, a + b);
}
```

Ini adalah apa yang akan terlihat seperti jika diimplementasikan dengan Python :

```
def add(a, b):
  return (a + b)
```

Mengembalikan dua nilai dari fungsi sebagai berikut, ini akan dipicu menggunakan daftar dengan Python :

```
static PyObject *foo _add _subtract(PyObject *self, PyObject *args) {
  int a;
  int b;

  if (!PyArg _ParseTuple(args, ”ii”, &a, &b)) {
    return NULL;
```

```
  }
  return Py _BuildValue("ii", a + b, a - b);
}
```

Ini adalah apa yang akan terlihat seperti jika diimplementasikan dengan Python :

```
def add _subtract(a, b):
  return (a + b, a - b)
```

Berikut adalah tanda tangan standar untuk fungsi Py _BuildValue :

```
PyObject* Py _BuildValue(char* format,...)
```

Format berikut adalah string C yang menggambarkan objek Python untuk dibangun. Argumentasi berikut Py _BuildValue adalah nilai C dari mana hasilnya dibuat. Hasil PyObject * adalah referensi baru.

Berikut daftar tabel string kode yang umum digunakan, yang nol atau lebihnya digabungkan ke dalam format string :

Table 26.2 Ukuran

Code	C Type	Meaning
c	char	String Python dengan panjang 1 menjadi huruf C.
d	double	Pelampung Python menjadi C ganda.
f	float	Pelampung Python menjadi pelampung C.
i	int	Int Python menjadi int int
l	long	Sebuah int Python menjadi panjang C.
L	long long	Sebuah int Python menjadi C panjang panjang
O	PyObject*	Gets non-NULL meminjam referensi ke argumen Python
s	char*	Python string tanpa nulls tertanam ke C char *
s	char*+int	Setiap string Python ke alamat dan panjang C
t	char*+int	Read-only penyangga segmen tunggal ke alamat C dan panjangnya
U	PyUNICODE*	Python Unicode tanpa nulls tertanam ke C
u	PPyUNICODE*int+	Setiap alamat dan panjang Python Unicode C
w	char*+int	Membaca / menulis penyangga segmen tunggal ke alamat dan panjang C
Z	char*	Seperti s, juga menerima None (set C char * ke NULL).
z	char*+int	Seperti s, juga menerima None set C char * ke NULL
(...)	Poin printer	Urutan Python diperlakukan sebagai satu argumen per item.
—	as per ...	Argumen berikut bersifat opsional.
:		Format akhir, diikuti dengan nama fungsi untuk pesan error.
;		Format akhir, diikuti oleh seluruh pesan kesalahan teks.

Kode {... } membangun kamus dari sejumlah nilai C, kunci dan nilai bergantian. Misalnya, Py _BuildValue (" {issi }", 23, "zig", "zag", 42) mengembalikan kamus seperti {23: 'zig', 'zag': 42 } Python.

Setiap blok memori yang dialokasikan dengan malloc () pada akhirnya harus dikembalikan ke genangan memori yang tersedia dengan satu panggilan untuk membebaskan (). Penting untuk menelepon gratis () pada waktu yang tepat. Jika alamat blok dilupakan tapi gratis () tidak dipanggil untuk itu, memori yang ditempatinya tidak dapat digunakan kembali sampai program berakhir. Ini disebut kebocoran memori. Di sisi lain, jika sebuah program memanggil gratis () untuk satu blok dan kemudian terus menggunakan blok tersebut, itu menciptakan konflik dengan penggunaan ulang blok melalui panggilan malloc () yang lain. Ini disebut dengan menggunakan memori yang dibebaskan. Ini memiliki konsekuensi buruk yang sama seperti merujuk pada data yang tidak diinisiasi - dump inti, hasil yang salah, crash misterius.

Karena Python membuat penggunaan malloc () dan gratis (), dibutuhkan strategi untuk menghindari kebocoran memori dan juga penggunaan memori yang bebas. Metode yang dipilih disebut penghitungan referensi. Prinsipnya sederhana: setiap objek berisi sebuah counter, yang bertambah saat referensi ke objek disimpan di suatu tempat, dan yang dikurangi saat referensi itu dihapus. Saat counter mencapai nol, referensi terakhir ke objek telah dihapus dan objeknya dibebaskan.

REFERENCES

1. J. S. Kilby, "Invention of the Integrated Circuit," *IEEE Trans. Electron Devices,* **ED-23,** 648 (1976).
2. R. W. Hamming, *Numerical Methods for Scientists and Engineers*, Chapter N-1, McGraw-Hill, New York, 1962.
3. J. Lee, K. Mayaram, and C. Hu, "A Theoretical Study of Gate/Drain Offset in LDD MOSFETs" *IEEE Electron Device Lett.,* **EDL-7**(3). 152 (1986).
4. A. Berenbaum, B. W. Colbry, D.R. Ditzel, R. D Freeman, and K.J. O'Connor, "A Pipelined 32b Microprocessor with 13 kb of Cache Memory," it Int. Solid State Circuit Conf., Dig. Tech. Pap., p. 34 (1987).

REFERENCES

[Kil76] J. S. Kilby, "Invention of the Integrated Circuit," *IEEE Trans. Electron Devices,* **ED-23,** 648 (1976).

[Ham62] R. W. Hamming, *Numerical Methods for Scientists and Engineers*, Chapter N-1, McGraw-Hill, New York, 1962.

[Hu86] J. Lee, K. Mayaram, and C. Hu, "A Theoretical Study of Gate/Drain Offset in LDD MOSFETs" *IEEE Electron Device Lett.,* **EDL-7**(3). 152 (1986).

[Ber87] A. Berenbaum, B. W. Colbry, D.R. Ditzel, R. D Freeman, and K.J. O'Connor, "A Pipelined 32b Microprocessor with 13 kb of Cache Memory," it Int. Solid State Circuit Conf., Dig. Tech. Pap., p. 34 (1987).