

## **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**

Copyright ©2007 by John Wiley & Sons, Inc. All rights reserved.

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

No part of this publication may be reproduced, stored in a retrieval system, or transmitted in any form or by any means, electronic, mechanical, photocopying, recording, scanning, or otherwise, except as permitted under Section 107 or 108 of the 1976 United States Copyright Act, without either the prior written permission of the Publisher, or authorization through payment of the appropriate per-copy fee to the Copyright Clearance Center, Inc., 222 Rosewood Drive, Danvers, MA 01923, (978) 750-8400, fax (978) 646-8600, or on the web at [www.copyright.com](http://www.copyright.com). Requests to the Publisher for permission should be addressed to the Permissions Department, John Wiley & Sons, Inc., 111 River Street, Hoboken, NJ 07030, (201) 748-6011, fax (201) 748-6008.

**Limit of Liability/Disclaimer of Warranty:** While the publisher and author have used their best efforts in preparing this book, they make no representations or warranties with respect to the accuracy or completeness of the contents of this book and specifically disclaim any implied warranties of merchantability or fitness for a particular purpose. No warranty may be created or extended by sales representatives or written sales materials. The advice and strategies contained herein may not be suitable for your situation. You should consult with a professional where appropriate. Neither the publisher nor author shall be liable for any loss of profit or any other commercial damages, including but not limited to special, incidental, consequential, or other damages.

For general information on our other products and services please contact our Customer Care Department with the U.S. at 877-762-2974, outside the U.S. at 317-572-3993 or fax 317-572-4002.

Wiley also publishes its books in a variety of electronic formats. Some content that appears in print, however, may not be available in electronic format.

***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

---

## PART I SUBMICRON SEMICONDUCTOR MANUFACTURE

1	Home	3
2	Overview	5
3	Environment Setup	7
4	Basic Syntax	9
5	Variable Type	11
6	Basic Operator	13
7	Decision Making	15
8	Loop	17
9	Numbers	19
10	Strings	21
11	Lists	23
12	Tuples	25
13	Dictionary	27
14	Functions	29
		vii



<b>15 Modules</b>	<b>31</b>
<b>16 Files I/O</b>	<b>33</b>
<b>17 Exceptions</b>	<b>35</b>
<b>18 Classess/Object</b>	<b>37</b>
<b>19 Reg Expression</b>	<b>39</b>
<b>20 CGI Programming</b>	<b>41</b>
<b>21 Databases Access</b>	<b>51</b>
<b>22 Sending Email</b>	<b>59</b>
<b>23 Multithreading</b>	<b>61</b>
<b>24 XML Processing</b>	<b>71</b>
<b>25 GUI Programming</b>	<b>81</b>
<b>26 Futher Expression</b>	<b>91</b>

# CONTENTS

---

List of Figures	xiii
List of Tables	xv
Foreword	xvii
Preface	xix
Acknowledgments	xxi
Acronyms	xxiii
Glossary	xxv
List of Symbols	xxvii
Introduction	xxix
<i>Catherine Clark, PhD.</i>	
References	xxix

## **PART I SUBMICRON SEMICONDUCTOR MANUFACTURE**

<b>1 Home</b>	<b>3</b>
	<b>ix</b>

<b>2</b>	<b>Overview</b>	<b>5</b>
<b>3</b>	<b>Environment Setup</b>	<b>7</b>
<b>4</b>	<b>Basic Syntax</b>	<b>9</b>
<b>5</b>	<b>Variabel Type</b>	<b>11</b>
<b>6</b>	<b>Basic Operator</b>	<b>13</b>
<b>7</b>	<b>Desicion Making</b>	<b>15</b>
<b>8</b>	<b>Loop</b>	<b>17</b>
<b>9</b>	<b>Numbers</b>	<b>19</b>
<b>10</b>	<b>Strings</b>	<b>21</b>
<b>11</b>	<b>Lists</b>	<b>23</b>
<b>12</b>	<b>Tuples</b>	<b>25</b>
<b>13</b>	<b>Dictionary</b>	<b>27</b>
<b>14</b>	<b>Functions</b>	<b>29</b>
<b>15</b>	<b>Modules</b>	<b>31</b>
<b>16</b>	<b>Files I/O</b>	<b>33</b>
<b>17</b>	<b>Exceptions</b>	<b>35</b>
<b>18</b>	<b>Clasess/Object</b>	<b>37</b>
<b>19</b>	<b>Reg Expression</b>	<b>39</b>
<b>20</b>	<b>CGI Programming</b>	<b>41</b>

<b>21</b>	<b>Databases Access</b>	<b>51</b>
<b>22</b>	<b>Sending Email</b>	<b>59</b>
<b>23</b>	<b>Multithreading</b>	<b>61</b>
<b>24</b>	<b>XML Processing</b>	<b>71</b>
<b>25</b>	<b>GUI Programming</b>	<b>81</b>
<b>26</b>	<b>Futher Expression</b>	<b>91</b>
	References	101
	References	103



## LIST OF FIGURES

---



# LIST OF TABLES

---

25.1	Ukuran	83
25.2	Ukuran	84
26.1	Ukuran	97
26.2	Ukuran	98





# 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 during 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 numerically 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 [1]. His first chip is shown in Fig. I. For comparison, Fig. I.2 shows a modern microprocessor chip, [4].

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

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

## 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

---

### ENVIRONMENT SETUP

---



## CHAPTER 4

---

### BASIC SYNTAX

---





## CHAPTER 5

---

### VARIABLE 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

---

### CLASSESS/OBJECT

---



## CHAPTER 19

---

## REG EXPRESSION

---



## CHAPTER 20

---

## NETWORKING

---





## CHAPTER 21

---

# 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 variannya). 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 mengizinkan 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 dikompilasi 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 pemrograman.

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, Python, 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 keamanan, 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; kecepatan 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 mengirimkannya 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 menyimpannya ke dalam suatu File atau secara otomatis mengirimkannya 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 gilirannya mengirimkannya kepada browser.

Program CGI pada prinsipnya bisa ditulis dalam bahasa pemrograman apa saja, namun kenyataannya 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 ekstensi python seperti ini AddHandler cgi-script .cgi .pl .asp .py
7. Cari baris ;Directory /\_ dan tambahkan ExecCGI pada list Options FollowSym-Links
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  
(tersebut 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 """


```

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

## CHAPTER 22

---

## DATABASES ACCESS

---

### **Pengertian Database**

Basis data adalah sekumpulan dari data yang telah disusun sesuai dengan aturan tertentu yang saling berhubungan sehingga memudahkan pengguna dalam mengelolanya juga memudahkan pengguna untuk memperoleh informasi. Selain itu ada juga yang menyebutkan bahwa database sebagai kumpulan file, tabel, atau arsip yang saling terhubung yang disimpan dalam media elektronik.

### **Manfaat Penggunaan Database**

#### **1. Kecepatan dan Kemudahan**

Database memiliki kemampuan dalam menyeleksi data sehingga menjadi suatu kelompok yang tersusun dengan cepat. Hal inilah yang akhirnya dapat menghasilkan informasi yang dibutuhkan secara cepat pula. Seberapa cepat pemrosesan data oleh database tergantung pada perancangan databasenya.

#### **2. Pemakaian Bersama-sama**

Suatu database bisa digunakan oleh siapa saja dalam suatu perusahaan. Sebagai contoh database mahasiswa dalam suatu perguruan tinggi dibutuhkan oleh be-

berapa bagian, seperti bagian admin, bagian keuangan, bagian akademik. Kesemua bidang tersebut membutuhkan database mahasiswa namun tidak perlu masing-masing bagian membuat databasenya sendiri, cukup database mahasiswa satu saja yang disimpan di server pusat. Nanti aplikasi dari masing-masing bagian bisa terhubung ke database mahasiswa tersebut.

### 3. Kontrol data terpusat

Masih berkaitan dengan point ke dua, meskipun pada suatu perusahaan memiliki banyak bagian atau divisi tapi database yang diperlukan tetap satu saja. Hal ini mempermudah pengontrolan data seperti ketika ingin mengupdate data mahasiswa, maka kita perlu mengupdate semua data di masing-masing bagian atau divisi, tetapi cukup di satu database saja yang ada di server pusat.

### 4. Menghemat biaya perangkat

Dengan memiliki database secara terpusat maka di masing-masing divisi tidak memerlukan perangkat untuk menyimpan database terhubung database yang dibutuhkan hanya satu yaitu yang disimpan di server pusat, ini tentunya memangkas biaya pembelian perangkat.

### 5. Keamanan Data

Hampir semua Aplikasi manajemen database sekarang memiliki fasilitas manajemen pengguna. Manajemen pengguna ini mampu membuat hak akses yang berbeda-beda disesuaikan dengan kepentingan maupun posisi pengguna. Selain itu data yang tersimpan di database diperlukan password untuk mengaksesnya.

### 6. Memudahkan dalam pembuatan Aplikasi baru

Dalam poin ini database yang dirancang dengan sangat baik, sehingga si perusahaan memerlukan aplikasi baru tidak perlu membuat database yang baru juga, atau tidak perlu mengubah kembali struktur database yang sudah ada. Sehingga Si pembuat aplikasi atau programmer hanya cukup membuat atau pengatur antarmuka aplikasinya saja.

Dengan segudang manfaat dan kegunaan yang dimiliki oleh database maka sudah seharusnya semua perusahaan baik itu perusahaan skala kecil apalagi perusahaan besar memiliki database yang dibangun dengan rancangan yang baik. Ditambah dengan pemanfaatan teknologi jaringan komputer maka manfaat database ini akan semakin besar. Penggunaan database sekaligus teknologi jaringan komputer telah banyak digunakan oleh berbagai macam perusahaan, contohnya saja perbankan yang memiliki cabang di setiap kotanya. Perusahaan Bank tersebut hanya memiliki satu database yang disimpan di server pusat, sedangkan cabang-cabangnya terhubung

melalui jaringan komputer untuk mengakses database yang terletak di sever pusat tersebut.

#### **Apa yang dimaksud dengan field, record, table, file, data & basis data?**

Field merupakan kumpulan dari karakter yang membentuk satu arti, maka jika terdapat field misalnya seperti KeteranganBarang atau JumlahBarang, maka yang dimunculkan dalam field tersebut harus yang berkaitan dengan keterangan barang dan jumlah barang. Atau field juga bisa disebut sebagai tempat atau kolom yang terdapat dalam suatu tabel untuk mengisikan nama-nama (data) field yang akan di isikan.

Record merupakan kumpulan field yang sangat lengkap, dan biasanya dihitung dalam satuan baris. Tabel merupakan kumpulan dari beberapa record dan juga field. File terdiri dari kumpulan field yang menunjukkan dari satu kesatuan data yang sejenis. Misalnya seperti file nama siswa berisikan data tentang semua nama siswa yang ada. Data adalah kumpulan fakta atau kejadian yang digunakan sebagai penyelesaian masalah dalam bentuk informasi. Basis data (database) terdiri dari dua kata, yaitu kata basis dan data. Basis merupakan tempat ataupun gudang, maupun wadah.

Data dapat disebut sebagai kumpulan dari fakta yang mewakili objek, misalnya seperti benda, manusia, barang dan sebagainya yang ditulis ke dalam bentuk angka, huruf, simbol, bunyi, teks, gambar ataupun gabungannya. Jadi dapat disimpulkan bahwa basis data merupakan kumpulan dari data-datayang terorganisasi yang saling berhubungan sedemikian rupa sehingga dapat dengan mudah disimpan, dimanipulasi, dan dipanggil oleh pemakainya. Karakter atau character yang ada didalam database merupakan bagian data yang terkecil, karakter tersebut dapat berupa karakter numerik, huruf ataupun karakter khusus (special characters) yang membentuk suatu item data atau field.

#### **Sifat-sifat database / basis data**

- Internal : kesatuan (integritas) dari file-file yang terlibat
- Terbagi/share : elemen-elemen database dapat dibagikan pada para user baik secara sendiri-sendiri maupun secara serentak dan pada waktu yang sama (concurrent sharing).

#### **Tipe Database / basis data**

Tipe Database Terdapat 12 tipe database, antara lain:

1. Operational database: Database ini menyimpan data rinci yang diperlukan untuk mendukung operasi dari seluruh organisasi. Mereka juga disebut subject-area databases (SADB), transaksi database, dan produksi database. Contoh: database pelanggan, database pribadi, database inventaris, akuntansi database.
2. Analytical database: Database ini menyimpan data dan informasi yang diambil dari operasional yang dipilih dan eksternal database. Mereka terdiri dari data

dan informasi yang dirangkum paling dibutuhkan oleh sebuah organisasi manajemen dan End-user lainnya. Beberapa orang menyebut analitis multidimensi database sebagai database, manajemen database, atau informasi database.

3. Data warehouse: Sebuah data warehouse menyimpan data dari saat ini dan tahun-tahun sebelumnya - data yang diambil dari berbagai database operasional dari sebuah organisasi.
4. Distributed database: Ini adalah database-kelompok kerja lokal dan departemen di kantor regional, kantor cabang, pabrik-pabrik dan lokasi kerja lainnya. Database ini dapat mencakup kedua segmen yaitu operasional dan user database, serta data yang dihasilkan dan digunakan hanya pada pengguna situs sendiri.
5. End-user database: Database ini terdiri dari berbagai file data yang dikembangkan oleh end-user di workstation mereka. Contoh dari ini adalah koleksi dokumen dalam spreadsheet, word processing dan bahkan download file.
6. External database: Database ini menyediakan akses ke eksternal, data milik pribadi online - tersedia untuk biaya kepada pengguna akhir dan organisasi dari layanan komersial. Akses ke kekayaan informasi dari database eksternal yang tersedia untuk biaya dari layanan online komersial dan dengan atau tanpa biaya dari banyak sumber di Internet.
7. Hypermedia databases on the web: Ini adalah kumpulan dari halaman-halaman multimedia yang saling berhubungan di sebuah situs web. Mereka terdiri dari home page dan halaman hyperlink lain dari multimedia atau campuran media seperti teks, grafik, gambar foto, klip video, audio dll.
8. Navigational database: Dalam navigasi database, queries menemukan benda terutama dengan mengikuti referensi dari objek lain.
9. In-memory databases: Database di memori terutama bergantung pada memori utama untuk penyimpanan data komputer. Ini berbeda dengan sistem manajemen database yang menggunakan disk berbasis mekanisme penyimpanan. Database memori utama lebih cepat daripada dioptimalkan disk database sejak Optimasi algoritma internal menjadi lebih sederhana dan lebih sedikit CPU mengeksekusi instruksi.
10. Document-oriented databases: Merupakan program komputer yang dirancang untuk aplikasi berorientasi dokumen. Sistem ini bisa diimplementasikan sebagai lapisan di atas sebuah database relasional atau objek database. Sebagai lawan dari database relasional, dokumen berbasis database tidak menyimpan data dalam tabel dengan ukuran seragam kolom untuk setiap record. Sebaliknya, mereka menyimpan setiap catatan sebagai dokumen yang memiliki karakteristik tertentu. Sejumlah bidang panjang apapun dapat ditambahkan ke dokumen. Bidang yang dapat juga berisi beberapa bagian data.

11. Real-time databases Real-time: Database adalah sistem pengolahan dirancang untuk menangani beban kerja negara yang dapat berubah terus- menerus. Ini berbeda dari database tradisional yang mengandung data yang terus- menerus, sebagian besar tidak terpengaruh oleh waktu.
12. Relational Database: Database yang paling umum digunakan saat ini. Menggunakan meja untuk informasi struktur sehingga mudah untuk mencari.

### Modul python untuk mengakses database MySQL

Untuk mengakses database MySQL dari Python, berikut adalah beberapa langkah sederhana. Yang pertama, server database MySQL harus siap dulu. Karenanya lakukan tahapan berikut ini.

1. Instal server mysql dengan menjalankan perintah `sudo apt-get install mysql-server`. Jangan lupa memasukkan password akun root untuk server MySQL.
2. Siapkan database dan tabel. Jalankan perintah `mysql -u root -p` dari terminal. Selanjutnya, kita akan masuk ke shell MySQL. Selanjutnya kita akan buat tabel yang skemanya seperti berikut ini (hanya ilustrasi saja).  

```
mysql> create database teman;
mysql> use teman;
mysql> create table alamat(id int not null auto_increment primary key, nama varchar(35), alamat text, telepon varchar(15), surat text);
```
3. ingin membuat user baru yang punya akses penuh ke database yang baru saja dibuat, lakukan tahapan berikut ini.  

```
mysql> create user 'andri'@'localhost' identified by '123456';
mysql> grant all on teman.* to 'andri'@'localhost' with grant option;
```
4. Langkah selanjutnya adalah membuat modul python untuk mengakses database tersebut. Untuk kasus ini, modul hanya diberi kemampuan untuk melihat seluruh isi tabel, sehingga tabel sebaiknya diisi dulu. Sedangkan kemampuan untuk melakukan operasi update dan delete dapat dibangun menggunakan pola yang sama. Modul tersebut seperti code berikut.

Modul terdiri dari dua fungsi, masing-masing `sambung` dan `selectall`. Fungsi pertama membutuhkan parameter terkait nama server, dan akun user serta mengembalikan variabel koneksi. Sedangkan fungsi `selectall` membutuhkan parameter koneksi yang diperoleh dari fungsi `sambung`, serta nama database dan nama tabel. Fungsi ini mengembalikan list yang berisi setiap row dalam tabel untuk kebutuhan lain yang belum terdefinisi dalam modul ini.

```
import MySQLdb
```



```

def sambung (host,user,passwd):
    mycon=MySQLdb.connect(host,user,passwd)
    return mycon

def selectall(mycon, dbname, table):
    mycur=mycon.cursor()
    mycur.execute('use ' + dbname)
    mycur.execute('select * from ' + table)
    rows=mycur.fetchall()
    a=[]
    for i in rows:
        nama=i[1]
        alamat=i[2]
        telepon=i[3]
        surat=i[4]
        print(nama + ' ' + alamat + ' ' + telepon + ' ' + surat)
        a.append(i)
    return a

```

Lalu, bagaimana menggunakannya? Untuk sementara, modul python ini hanya dapat diakses melalui shell python karena belum ada fungsi main yang terdefinisi. Hal ini disebabkan karena rancangan input/output masih seadanya. Karenanya, mari masuk ke shell python dengan menjalankan perintah python di terminal. Yang perlu diperhatikan, penggunaan shell python harus dilakukan dari directory di mana modul ini disimpan. Berikut adalah gambaran ketika berada dalam shell python dan menggunakan modul ini.

```

<<< from mymodul import *
<<< mycon=sambung("localhost","andri","123456")
<<< a=selectall(mycon,"teman","alamat")
Andri Jl. Sariosih, Sarijadi, Bandung 12450 08123456789 andri@poltekpos.ac.id
<<<

```

Di baris pertama, kita meng-import modul dari nama file, untuk selanjutnya meng-import semua fungsi yang ada di dalamnya. Selanjutnya, kita membuat variabel bernama mycon bertipe koneksi ke MySQL (dapat dilihat dengan cara menjalankan perintah type(mycon) dari shell python) dan meng-assigned nilainya dari memanggil fungsi sambung. Selanjutnya, variabel a di-assinged nilainya dari memanggil fungsi selectall. Terlihat bahwa fungsi selectall mencetak nilai yang diperoleh dari operasi select tabel.

Dengan ilustrasi ini, diharapkan dapat memberi inspirasi membuat modul python yang digunakan untuk sebuah aplikasi, misalnya dengan menjadikannya sebagai bagian dari hubungan SIGNAL-SLOT pada QT4. Semoga bermanfaat.

Dalam era informasi dimana kita hidup sekarang, kita dapat melihat seberapa banyak data dunia berubah. Kita pada dasarnya membuat, menyimpan, dan menarik data, secara ekstensif! Harusnya ada sebuah cara untuk menangani semua itu.

tidak dapat disebarkan kemana-mana tanpa adanya manajemen bukan? Di sini hadir Database Management System (DBMS). DBMS adalah sebuah sistem software yang memungkinkanmu untuk membuat, menyimpan, memodifikasi, menarik, dan penanganan lainnya terhadap sebuah data dari database. Sistem ini juga bervariasi dalam ukuran, mulai dari sistem kecil yang cukup berjalan pada komputer personal hingga yang lebih besar yang berjalan dalam mainframe.

### Python Database API

Python dapat berinteraksi dengan database. Namun, bagaimana itu dapat melakukannya? Python menggunakan apa yang disebut Python Database API dengan tujuan untuk menjadi antarmuka dengan database. API ini memungkinkan kita untuk memprogram database management system (DBMS) yang berbeda. Untuk DBMS yang berbeda itu, bagaimana pun juga, proses yang diikuti pada tingkatan code tetap sama, yaitu sebagai berikut:

- Membangun sebuah koneksi ke database pilihanmu.
- Membuat sebuah kursor untuk berkomunikasi dengan data.
- Memanipulasi data menggunakan SQL (berinteraksi).
- Memberitahu koneksi untuk entah menerapkan manipulasi SQL ke data dan membuatnya permanen (commit), atau memberitahunya untuk meninggalkan manipulasi itu (rollback), sehingga mengembalikan data ke keadaan sebelum interaksi terjadi.
- Menutup koneksi ke database.

Bagaimana caranya menampilkan data dari mysql menggunakan python. Saya asumsikan teman-teman sudah menginstall web server (XAMPP/ yang lainnya) dan python di komputer masing-masing. Setelah itu semua siap sekarang silahkan download mysql connector untuk python disini (Sesuaikan dengan versi python yang kalian punya). Setelah itu tinggal install.

Setelah selesai installasi, buka phpmyadmin dan buat database dengan nama terserah. Contohnya yaitu "trial " Lalu import sql berikut

trial.sql

Setelah di impor, lalu buka teks editor dan copas-kan code berikut

show-data.py

lalu simpan dengan nama terserah kalian (punya saya show-data.py) dan tinggal kalian run dari command prompt.

### Koneksi Database MySQL dengan Python

1. Pastikan sudah menginstall package libmysqlclient-dev dengan:  
apt-get install libmysqlclient-dev
2. Download modul MySQLdb.

3. Instal modul tersebut dengan cara (Pastikan login sebagai root ya):

```
$ gunzip MySQL-python-1.2.2.tar.gz
$ tar -xvf MySQL-python-1.2.2.tar
$ cd MySQL-python-1.2.2
$ python setup.py build
$ python setup.py install
```

4. Buat satu script seperti berikut (contoh: test.py):

```
import MySQLdb
# Open database connection
db = MySQLdb.connect("localhost","root","password","nama_database" )
# prepare a cursor object using cursor() method
cursor = db.cursor()
# execute SQL query using execute() method.
cursor.execute("SELECT VERSION()")
# Fetch a single row using fetchone() method.
data = cursor.fetchone()
print "Database version : %s " % data
# disconnect from server
db.close()
```

5. Test dengan code berikut:

```
python test.py
```

## CHAPTER 23

---

### SENDING EMAIL

---



## CHAPTER 24

---

# 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 (interrupted)*
- 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 (threadName, 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 mengekspos 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 mengembalikan 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 sebuah instance dari itu dan kemudian memulai *thread* baru dengan menerapkan *start()*, yang ada gilirannya 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 memungkinkan 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)
threads.append(thread2)

# Wait for all threads to complete
for 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 workQueue.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 25

---

# XML PROCESSING

---

### Python XML Processing

XML adalah bahasa open source portable yang memungkinkan pemrogram mengembangkan aplikasi yang dapat dibaca oleh aplikasi lain, terlepas dari sistem operasi dan bahasa pengembangannya.

#### Apa itu XML?

Extensible Markup Language (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 memarsing 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 bekerjasama dengan file besar. Di sisi lain, menggunakan DOM secara eksklusif benar-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 *ControlHandler* 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 memarsing 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 *startElementNS* dan *endElementNS* dipanggil. Disini, tag adalah elemen dan atribut adalah atribut.

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

Metode

Metode berikut membuat objek parsing baru dan mengembalikannya. Objek parsing dibuat 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 *ContentHandler*

- *ErrorHandler*

Jika ditentukan, *errorhandler* harus menjadi objek *ErrorHandler* SAX

- Metode *parseString*

Membuat parsing SAX dan mengurai string XML yang ditentukan :

```
xml.sax.parsestring(xmlstring, contenthandler[, errorhandler])
```

Berikut ini adalah detail nama dan parameter :

XMLstring

Nama dari string yang bisa dibaca

ContentHandler

Menjadi objek ContentHandler

ErrorHandler

Menjadi objek ErrorHandler 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 element 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 namespaces
    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 Object 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 digunakan saat sebuah dokumen diurai untuk mengetahui bagaimana menghasilkan bentuk serial dari isinya kemudian dapat ditulis ke sebuah file.

Untuk membuat instance elemen gunakan fungsi elemen constructor dan *SubElement()* pabrik.

Import `xml.etree.ElementTree` as `xml`

```
filename = "/home/abc/Desktop/test_xml.xml "
```

```
root = xml.Element( "Users "
```

```
userelement = xml.Element( "user "
```

```
root.append(userelement)
```

Bila menjalankan ini, akan menghasilkan sebagai berikut :

```
<Users>
  <user>
    <user>
  </Users>
```

Tambahkan anak-anak pengguna

```
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(userelement, "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 menggunakan fungsi *ElementTree*. Kemudian membuat elemen pengguna dan menambahkannya ke root. Selanjutnya membuat

*SubElement* dengan melewati elemen pengguna (userelement) ke *SubElement* beserta namanya seperti "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 Document :

```

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 " language=" 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 26

---

# 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 objek 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 26.1** Ukuran

Operator	Penjelasan
Button	Menampilkan 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 sejumlah 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 26.2** Ukuran

Karakter	Penjelasan
c	Sentimeter
i	Inci
m	Milimeter
p	Poin printer

Tkinter mengungkapkan panjang sebagai integer jumlah piksel.

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
- `wrplength`  
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 Tkinter, yaitu :

- Menggunakan string menentukan proporsi merah, hijau dan biru didigit heksadesimal. Misalnya " #ffff " putih, " #000000 " hitam dan " #000fff00 " hijau.
- Menggunakan lokal standar nama warna . warna-warna " white ", " black ", " green " dan " magenta " akan selalu tersedia.

Pilihan warna umum :

- `activebackground`  
Warna latar belakang 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
- `highlightcolor`  
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 tebal, normal untuk berat badan secara teratur
- Slant  
Italic untuk miring, roman untuk unslanted
- 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

Jangkar 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()
```

### Bitmaps

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 mengorganisir 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. Berikut 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 standar 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 Tkinter) ini bersifat open-source, maka banyak sekali komunitas-komunitas yang membahas Python dan Tkinter dan bisa belajar dan bertanya langsung dengan para ahli.

## CHAPTER 27

---

## 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 dianggap sebagai ekstensi.

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 perpustakaan yang terhubung secara dinamis).

#### 4.1 Pra-Persyaratan untuk Menulis Ekstensi

Untuk memulai ekstensi, 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 disukai karena tidak pernah diluar modul ekstensi mendefinisikan sebagai statis.

Fungsi C biasanya diberi nama dengan menggabungkan modul dan fungsi Python bersama-sama yang ditunjukkan 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 :

- **MI \_name**  
Nama fungsi yang digunakan penafsir Python saat digunakan dalam program Python
- **MI \_meth**  
Menjadi alamat ke fungsi yang memiliki salah satu tanda tangan yang dijelaskan dalam penelusuran sebelumnya
- **MI \_flags**  
Memberitahu penafsir yang mana dari tiga tanda tangan yang digunakan ml \_meth. Bendera ini biasanya memiliki nilai meth \_varargs. Bendera ini dapat digandakan dengan orod dengan meth \_keywords jika ingin memiarkan argumen kata kunci masuk ke fungsi. Ini juga bisa memiliki nilai meth \_noargs yang menunjukkan bahwa tidak ingin menerima argumen apa pun.
- **MI \_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 menggunakan saat menentukan fungsinya.

Fungsi inialisasi C umumnya memiliki strktur keseluruhan berikut :

```
PyMODINIT_FUNC initModule() {
    Py_InitModule3(func, module _methods, "docstring...");
}
```

Berikut adalah penjelasan fungsi Py\_InitModule :

- 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 setup.py sebagai berikut :

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

Sekarang gunakan perintah berikut yang aka melakukan semua kompilasi dan langkah penghubung 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 27.1** 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 27.2** 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," in 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," in *Int. Solid State Circuit Conf.*, Dig. Tech. Pap., p. 34 (1987).



