24 Jam Belajar
Internet of Things (IoT)
dengan Raspberry Pi

A.R. Anom Besari, S.ST, M.Sc

Daftar Isi

DASAR

1. Berkenalan dengan Raspberry Pi
2. Instalasi Raspberry Pi
3. Selamat Datang di Raspbian OS
4. Familiar Linux Terminal
5. Bahasa Pemrograman Python
6. Akses Dunia Luar dengan GPIO
7. Pemrograman Multi-Thread
8. Pemrograman Jaringan Dasar

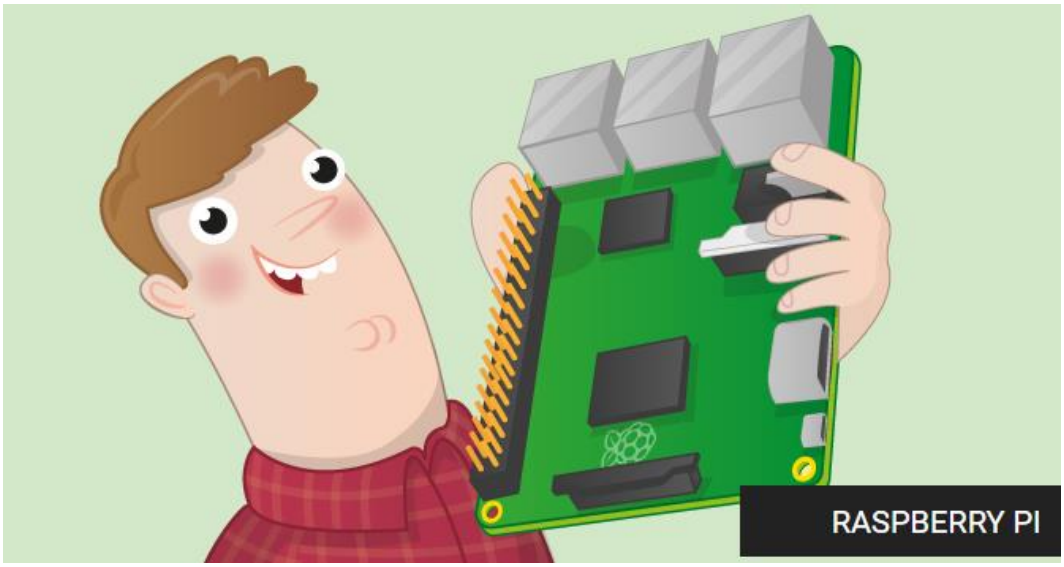
MENENGAH

9. Akuisisi Data Sensor
10. Kontrol Aktuator
11. Tampilan dan Interaksi Pengguna
12. Komunikasi Data
13. Koneksi ke Jaringan Lokal
14. Menyimpan ke Database MySQL.
15. Aplikasi Monitoring berbasis Website
16. Proyek IoT Menengah

LANJUT

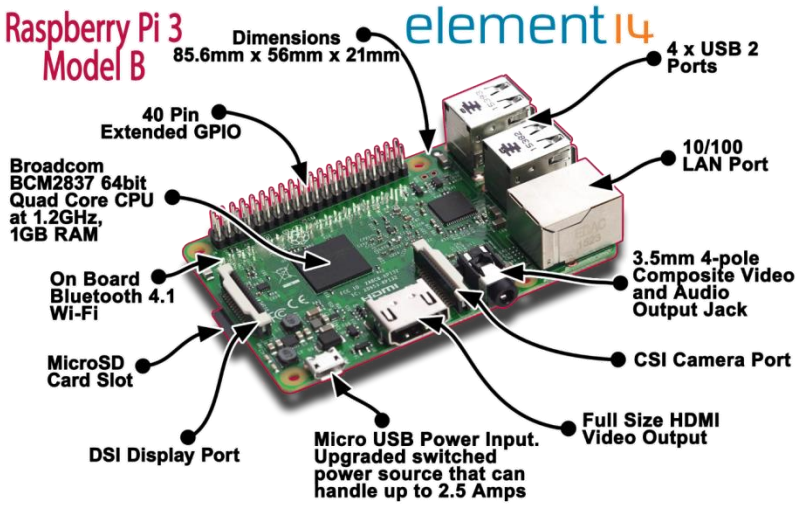
17. Desain Sistem Big Data
18. Penyimpanan Data ke Cloud
19. Analisis Data
20. Machine Learning
21. Aplikasi Monitoring berbasis Mobile
22. Fitur Spesial : Komputer Visi
23. Fitur Spesial : Robotika
24. Proyek IoT Lanjut

1 Berkenalan dengan Raspberry Pi



Raspberry Pi, sering disingkat dengan nama Raspi, adalah komputer papan tunggal (*single-board circuit*; SBC) yang seukuran dengan kartu kredit yang dapat digunakan untuk menjalankan program perkantoran, permainan komputer, dan sebagai pemutar media hingga video beresousi tinggi. Raspberry Pi dikembangkan oleh yayasan nirlaba, Rasberry Pi Foundation, yang digawangi sejumlah pengembang dan ahli komputer dari Universitas Cambridge, Inggris.

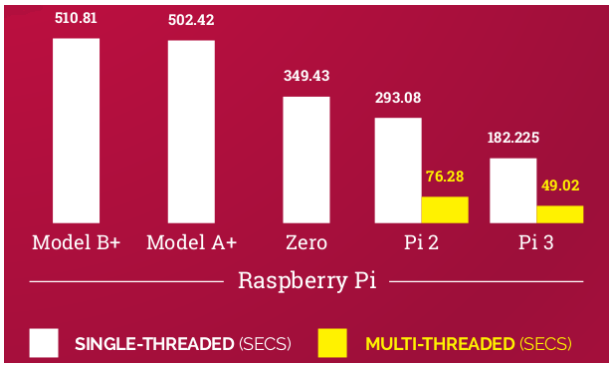
Ide dibalik Raspberry Pi diawali dari keinginan untuk mencetak pemrogram generasi baru. Seperti disebutkan dalam situs resmi Raspberry Pi Foundation, waktu itu Eben Upton, Rob Mullins, Jack Lang, dan Alan Mycroft, dari Laboratorium Komputer Universitas Cambridge memiliki kekhawatiran melihat kian turunnya keahlian dan jumlah siswa yang hendak belajar ilmu komputer. Mereka lantas mendirikan yayasan Raspberry Pi bersama dengan Pete Lomas dan David Braben pada 2009. Tiga tahun kemudian, Raspberry Pi Model B memasuki produksi massal. Dalam peluncuran pertamanya pada akhir Febuari 2012 dalam beberapa jam saja sudah terjual 100.000 unit. Pada bulan Februari 2016, Raspberry Pi Foundation mengumumkan bahwa mereka telah menjual 8 juta perangkat Raspi, sehingga menjadikannya sebagai perangkat paling laris di Inggris.



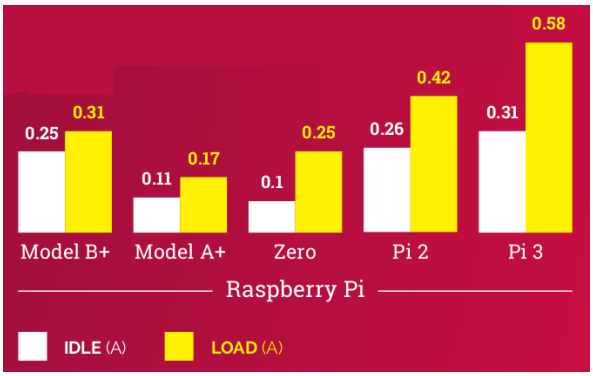
Berikut ini spesifikasi dari Raspberry Pi 3 :

- SoC: Broadcom BCM2837
- CPU: 4× ARM Cortex-A53, 1.2GHz
- GPU: Broadcom VideoCore IV
- RAM: 1GB LPDDR2 (900 MHz)
- Networking: 10/100 Ethernet, 2.4GHz 802.11n wireless
- Bluetooth: Bluetooth 4.1 Classic, Bluetooth Low Energy
- Storage: microSD
- GPIO: 40-pin header, populated
- Ports: HDMI, 3.5mm analogue audio-video jack, 4× USB 2.0, Ethernet, Camera Serial Interface (CSI), Display Serial Interface (DSI)

Dibangun dengan sistem-on-chip (SoC) Broadcom BCM2837 yang mencakup empat fitur : ARM Cortex-A53 *processing core* berjalan pada 1.2GHz dengan 32KB Level 1 dan 512 KB Level 2 cache memory, sebuah prosesor grafis VideoCore IV , dan terhubung dengan modul memori 1GB LPDDR2 di bagian belakang papan. Raspberry Pi terbaru menawarkan dukungan operasi *multi-threaded* dengan mengambil keunggulan dari empat *processing core* pada Pi 2 dan Pi 3. SysBench mengungkapkan pengembangan desain Raspberry Pi dari pertama hingga terakhir, menunjukkan kinerja single-threaded telah meningkat pesat, dan keunggulan terbesar pada saat dijalankannya program multi-threaded.

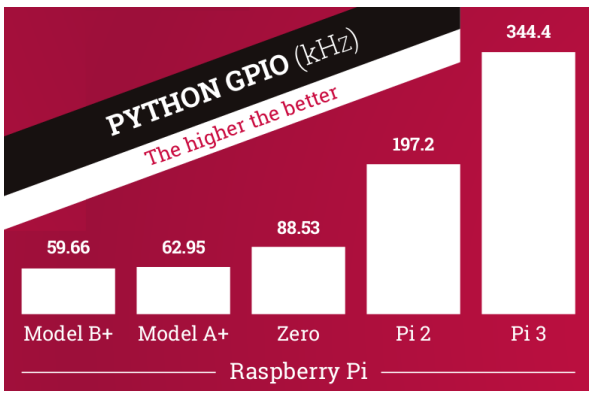


Anda tidak bisa mendapatkan kinerja ekstra dari Raspberry Pi tanpa beberapa pengorbanan. Raspberry Pi 3 mengambil daya lebih besar dari versi sebelumnya, baik pada saat mendapat beban kerja maksimum (*load*) maupun pada saat tidak ada beban (idle). Jika Anda mencari jenis Raspberry yang memiliki daya tahan baterai maksimum, mungkin Anda bisa memilih Model A + Pi Zero sebagai alternatif.



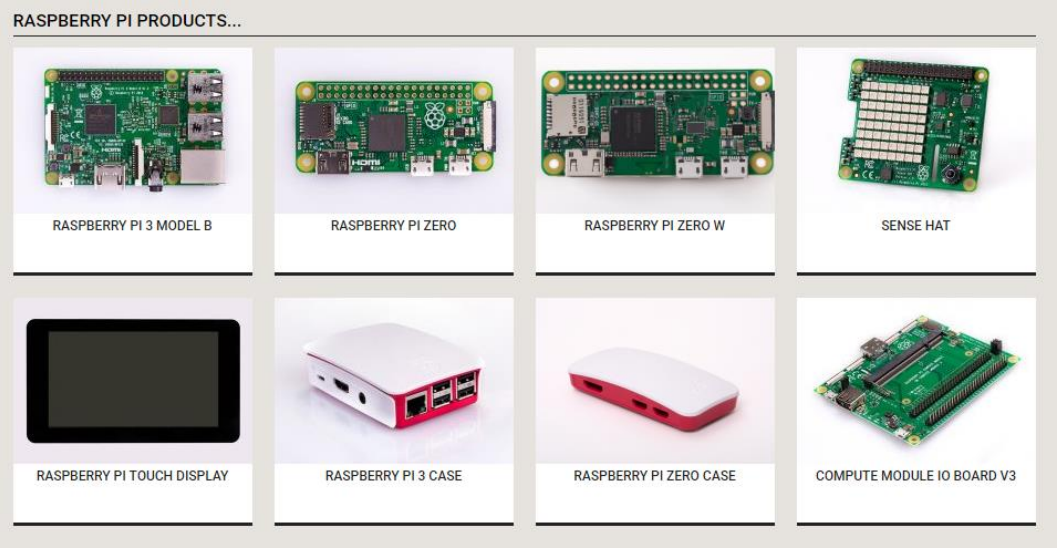
Chip Broadcom BCM43438 menyediakan 2.4GHz 802.11n wireless LAN, Bluetooth Low Energy, dukungan Bluetooth 4.1 Classic Radio. Tidak perlu untuk menghubungkan antena eksternal ke Raspberry Pi 3. Radio yang terhubung ke antena chip ini disolder langsung ke papan, untuk menjaga ukuran perangkat tetap keci;. Meskipun bertubuh mungil, antena ini mampu mengambil sinyal Wireless LAN dan Bluetooth, bahkan bisa menembus dinding. Raspberry Pi 3 menggunakan SMSC LAN9514 chip seperti pendahulunya, Raspberry Pi 2, dengan menambahkan konektivitas Ethernet 10/100 dan empat USB *channel* ke *print circuit board*. Seperti sebelumnya, chip SMSC menghubungkan ke SoC melalui saluran USB tunggal, bertindak sebagai adapter USB-to-Ethernet dan USB hub.

Raspberry Pi 3 memiliki fitur 40-pin general-purpose input-output (GPIO) dengan header yang sama seperti semua jenis Raspberry Pi Model B+ dan Model A+. Perangkat keras GPIO yang ada akan bekerja tanpa modifikasi; satu-satunya perubahan adalah switch UART tergabung pada pin GPIO, namun hal ini sudah ditangani secara internal oleh sistem operasi. Pin GPIO Raspberry Pi sering digunakan bersama dengan program berbasis Python, meskipun hal ini dapat menyebabkan terjadinya *bottleneck CPU*. Namun pada Raspberry terbaru hal ini sudah teratasi. Dalam tes ini, program sederhana yang menggunakan library RPi.GPIO digunakan untuk menghitung seberapa cepat mematikan pin dengan menggunakan *counter* frekuensi.

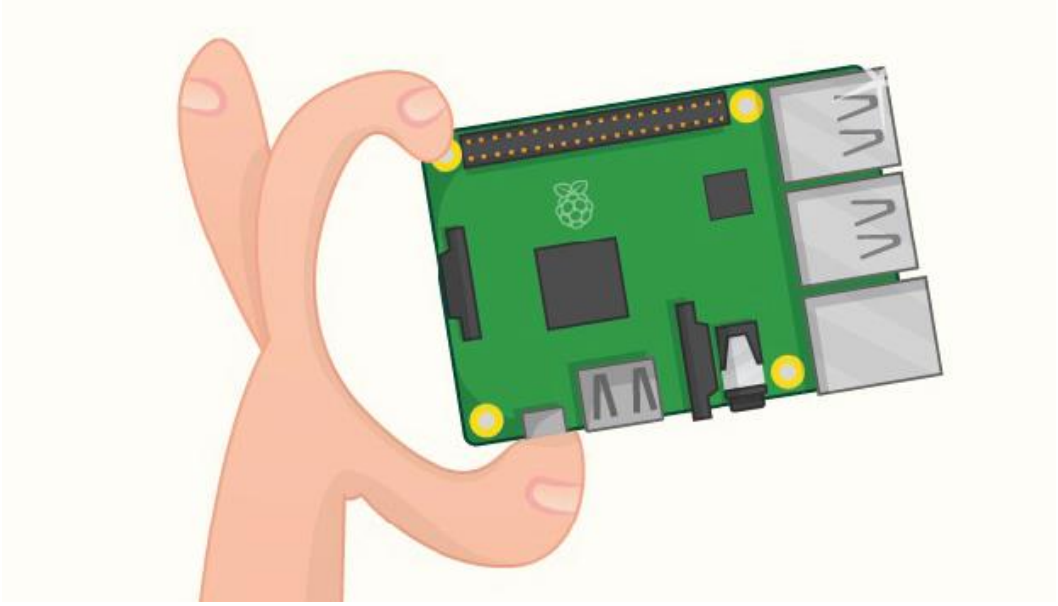


Raspberry Pi memiliki website di www.raspberrypi.org yang memiliki cukup banyak informasi tentang produk, aksesoris sampai dengan contoh proyek pembelajaran yang dapat dikerjakan dengan Raspberry Pi. Berikut ini adalah tampilan website Raspberry Pi.

Produk dan aksesoris Raspberry Pi dapat dilihat di website ini juga, ada banyak jenis Raspberry Pi yang dapat dipilih sesuai dengan kebutuhan. Aksesoris tambahan seperti Sense Hat, I/O Board Computer Module, Touch Displat sampai dengan Case tersedia untuk melengkapi Raspberry Pi dalam memenuhi keiinginan pengguna untuk merealisasikan ide kreatif dengan komputer mini ini.



2 Instalasi Raspberry Pi



Setelah Anda mengenal apa itu Raspberry Pi, mari kita mulai dengan memastikan perangkat yang diperlukan untuk memulai bekerja dengan Raspberry Pi.

- Menyiapkan peralatan pendukung
- Memasang sistem operasi Raspbian
- Melakukan pemasangan Raspberry Pi
- Menghubungkan Raspberry Pi dengan internet
- Menambahkan media penyimpanan tambahan
- Hubungkan headphone atau speaker untuk output audio

Menyiapkan Peralatan Pendukung

Sebelum Anda memasangkan apapun ke dalam Raspberry Pi, pastikan Anda memiliki peralatan yang Anda butuhkan seperti :

- **Raspberry Pi**
  Sudah pasti harus Anda sediakan ☺.

- **Monitor atau TV**
  Kebanyakan televisi dan monitor saat ini memiliki koneksi HDMI, hal ini akan memudahkan Anda untuk terhubung dengan Raspberry Pi.

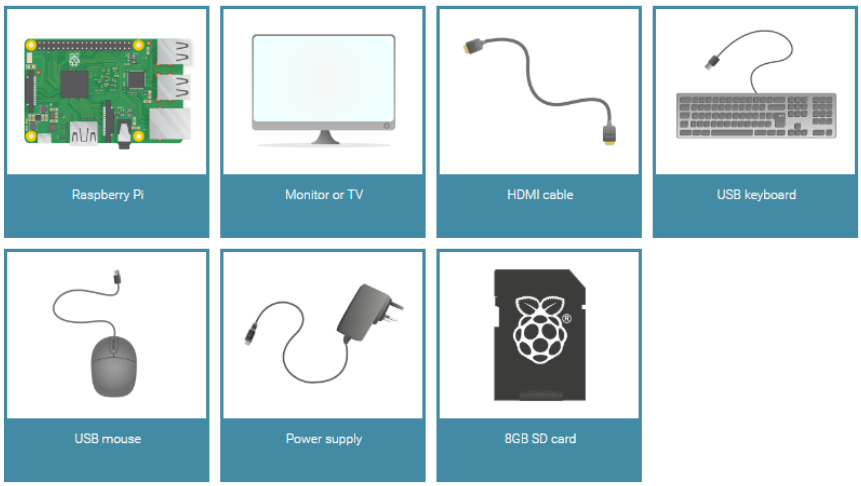
- **Kabel HDMI**
  Anda bisa mengubungkan Raspberry Pi ke monitor Anda secara langsung dengan menggunakan kabel HDMI. Jika monitor Anda memiliki konektor VGA atau DVI, maka yang Anda perlukan adalah konverter HDMI-to-VGA atau HDMI-to-DVI agar dapat terhubung dengan Raspberry Pi.

- **Keyboard dan mouse**

Keyboard dan mouse dengan standar USB dapat digunakan agar terhubung dengan Raspberry Pi secara "*plug and play*" (dapat bekerja tanpa driver tambahan). Cukup pasang keyboard dan mouse ke Raspberry Pi. Anda dapat menggunakan keyboard dan mouse bluetooth, namun tingkat keberhasilannya bervariasi tergantung pada model dan produsen.

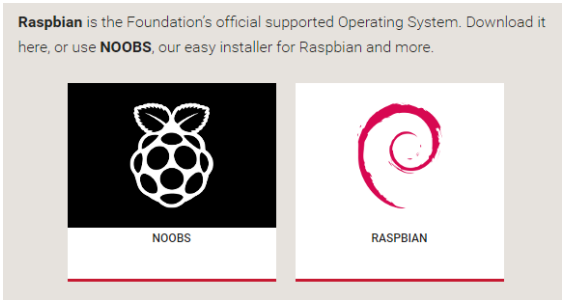
- **Micro USB power supply**
  Jika Anda menggunakan Raspberry Pi 3, maka disarankan agar Anda menggunakan power supply 5V, 2.5A. Ataupun Anda dapat menggunakan charger handphone, namun Anda harus memeriksa bahwa arus dan tegangan cukup untuk menghidupkan Raspberry Pi (5V/2.5A).

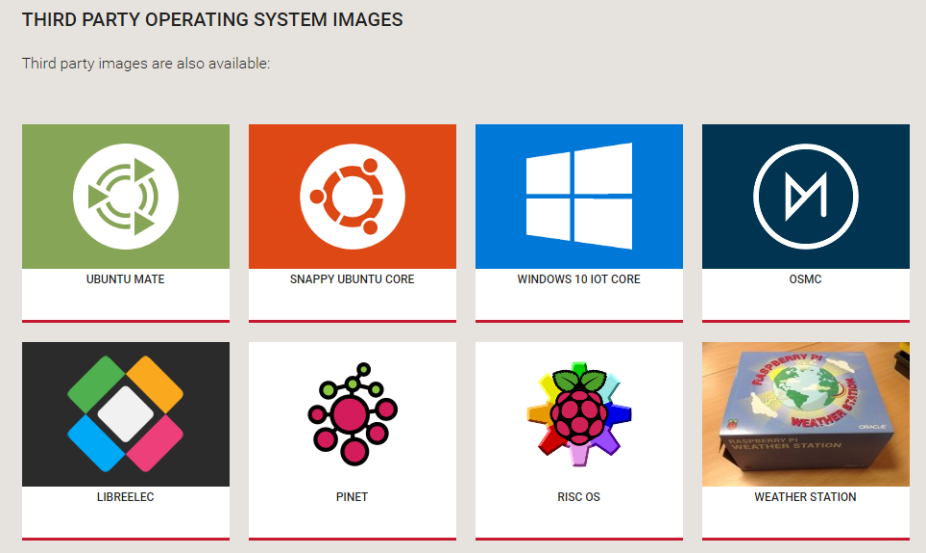
- **Micro SD card**
  Versi terbaru dari sistem operasi yang direkomendasikan untuk Raspberry Pi yaitu Raspbian OS, membutuhkan minimal kartu micro SD dengan kapasitas 8GB (atau lebih besar). Tidak semua kartu micro SD memiliki tingkat keberhasilan yang sama, disarankan yang memiliki spesifikasi Class 10. Berikutnya Anda harus mengikuti panduan konfigurasi software untuk belajar bagaimana untuk memasang sistem operasi ke kartu micro SD.



Memasang Sistem Operasi Raspbian

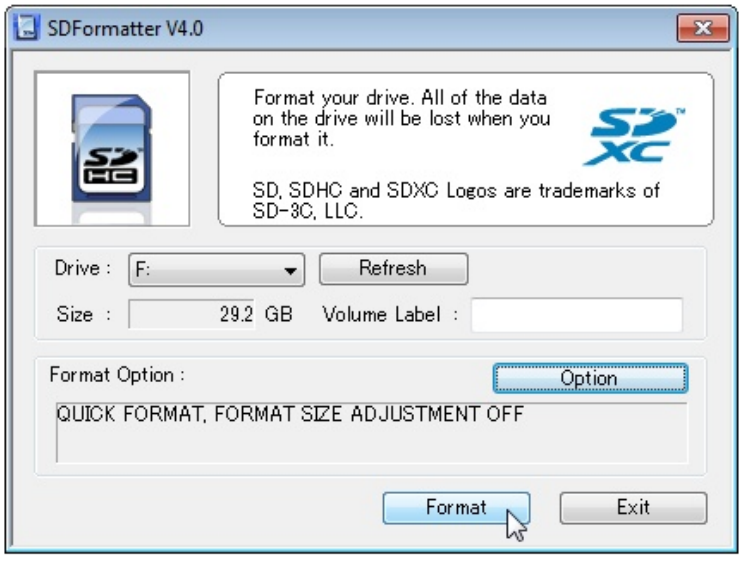
Setelah Anda memastikan semua peralatan fisik yang Anda butuhkan saat ini. Sekarang waktunya untuk memasang sistem operasi pada Raspberry Pi. Berikut ini adalah beberapa sistem operasi yang support dengan Raspberry Pi.





Sistem operasi yang direkomendasikan untuk digunakan pada Raspberry Pi disebut Raspbian. Raspbian adalah versi GNU / Linux yang dirancang khusus untuk bekerja dengan baik di hardware Raspberry Pi. Anda memiliki beberapa pilihan ketika datang untuk mendapatkan memegang salinan Raspbian.
Terlebih dulu pastikan Anda telah mem-format kartu micro SD dan Pastikan bahwa komputer Anda memiliki pembaca kartu SD, atau Anda dapat menggunakan *USB SD Card Reader*.

1. Kunjungi website *SD Association* dan download SD Formatter 4.0 (untuk Windows atau Mac) pada link berikut : https://www.sdcard.org/downloads/formatter_4/index.html
2. Ikuti petunjuk untuk menginstal perangkat lunak.
3. Masukkan kartu micro SD Anda ke komputer kemudian perhatikan huruf drive dimana kartu micro SD dialokasikan untuk itu, misalnya F:/.
4. Pada SD Formatter, pilih huruf drive untuk kartu micro SD dan lakukan format.



Installasi Raspbian dengan NOOBS

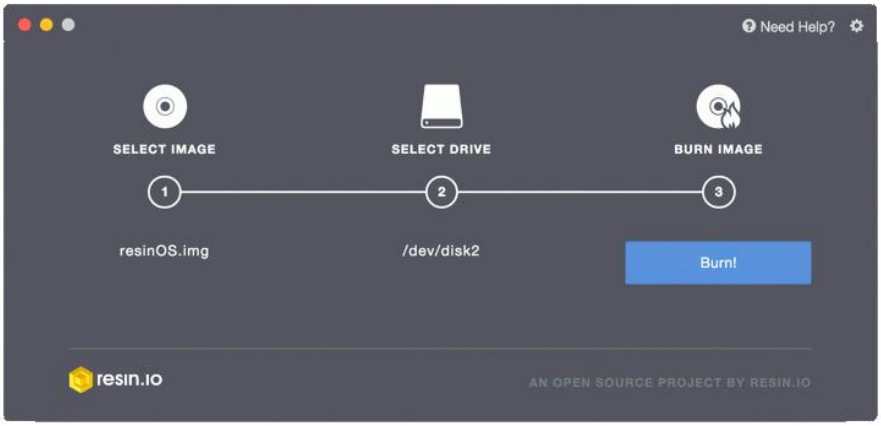
NOOBS singkatan *New Out Of Box Software*, dan jika Anda belum pernah bermain-main dengan GNU/Linux sebelumnya, maka disarankan Anda memulai dengan NOOBS.

1. Kunjungi halaman resmi Raspberry Pi Downloads. https://www.raspberrypi.org/downloads/
2. Klik pada NOOBS.
3. Klik pada tombol Download ZIP di bawah 'NOOBS (offline dan install via jaringan)', dan pilih folder untuk menyimpannya.
4. Ekstrak file dari ZIP.
5. Setelah kartu SD Anda telah diformat, copy semua file dalam folder NOOBS yang telah diekstrak ke drive kartu micro SD .
6. File yang diperlukan kemudian akan ditransfer ke kartu SD Anda.
7. Ketika proses ini selesai, keluarkan kartu SD dengan aman (*safely remove*) dan masukkan ke Raspberry Pi.

Installasi Raspbian dengan file image secara langsung

Alternatif lainnya selain menggunakan NOOBS untuk menginstal Raspbian adalah dengan mengunduh dan menginstal file image Raspbian secara langsung. Proses ini lebih cepat, utamanya jika Anda memerlukan file image untuk diinstal ke beberapa kartu micro SD semisal untuk kegiatan pelatihan.

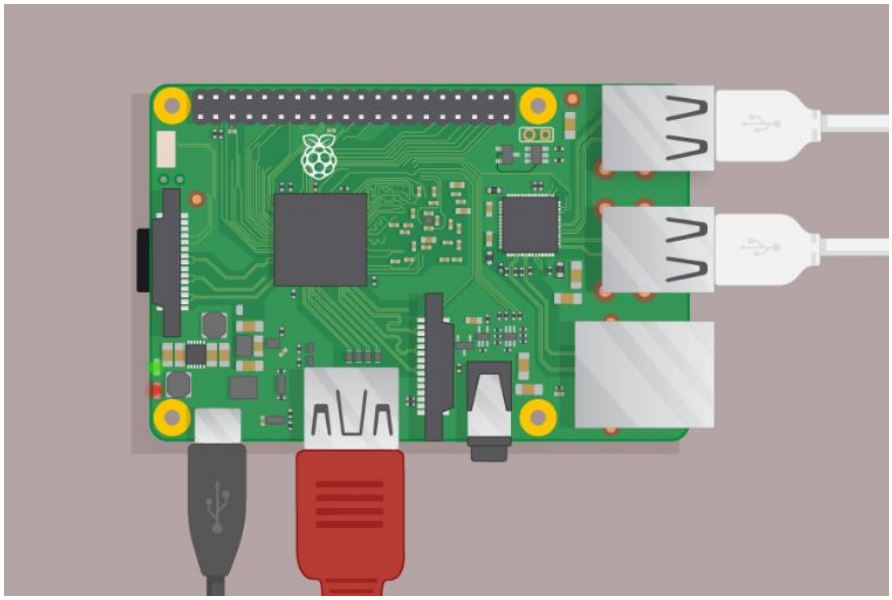
1. Menggunakan komputer dengan pembaca kartu SD, kunjungi halaman resmi Raspberry Pi Downloads.
2. Klik pada Raspbian.
3. Klik pada tombol Download ZIP di bawah 'Raspbian Jessie (*full desktop image*)', dan pilih folder untuk menyimpannya.
4. Ekstrak file dari ZIP.
5. Kunjungi etcher.io (http://www.etcher.io/), kemudian unduh dan pasang Etcher SD card image utility.
6. Jalankan Etcher SD card image dan pilih file image Raspbian yang telah di un-ZIP di komputer atau laptop.
7. Pilih drive kartu micro SD. Perhatikan bahwa perangkat lunak sudah dipilih untuk dipasang ke drive yang tepat.
8. Akhirnya, klik Burn untuk mentransfer Raspbian ke kartu micro SD. Anda akan melihat progress bar yang memberitahu Anda proses transfernya. Setelah selesai, utilitas akan secara otomatis melepaskan kartu micro SD sehingga aman untuk mengeluarkannya dari komputer.



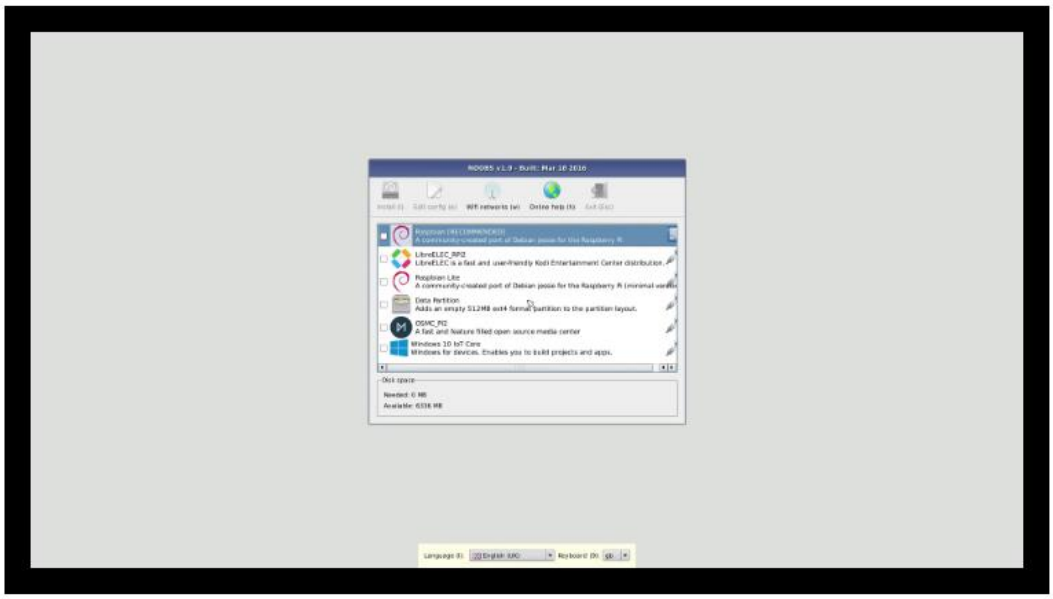
Sekarang Anda memiliki sistem operasi Raspbian, masukkan kartu micro SD Anda ke dalam Raspberry Pi. Ikuti cara pemasangan Raspberry Pi untuk pertama kalinya setelah bagian ini.

Memasang Raspberry Pi untuk pertama kalinya

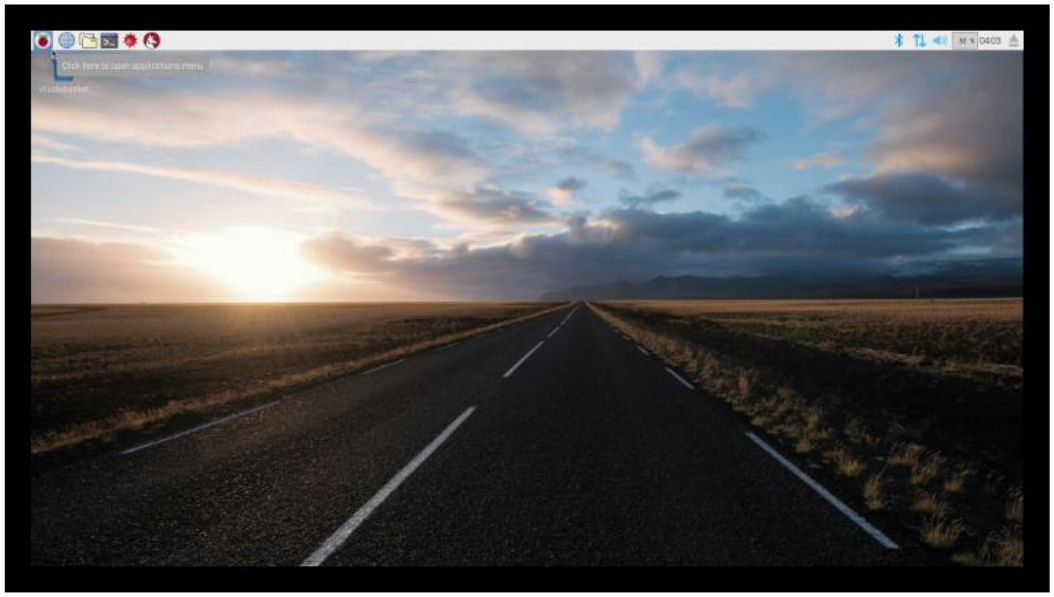
1. Mulailah dengan memasang kartu micro SD ke dalam slot pada Raspberry Pi.
2. Berikutnya, pasang keyboard dan mouse ke port USB pada Raspberry Pi.
3. Pastikan bahwa monitor atau TV telah dihidupkan, dan Anda telah memilih input yang tepat (misalnya HDMI 1, DVI, dll).
4. Hubungkan kabel HDMI dari Raspberry Pi ke monitor atau TV.
5. Jika Anda berniat untuk menghubungkan Raspberry Pi ke internet, pasang kabel Ethernet ke port Ethernet, atau menghubungkan modem atau dongle WiFi ke salah satu port USB (fitur WiFi sudah tersedia pada Raspberry Pi 3).
6. Bila Anda telah memasang semua kabel dan kartu SD dengan benar, berikutnya adalah menyalakan Raspberry Pi dengan menghubungkan catu daya USB mikro ke Raspberry Pi. Dengan demikian Raspberry Pi akan booting untuk masuk ke sistem operasi.



Jika Anda menggunakan NOOBS dan pertama kalinya Raspberry Pi dan kartu micro SD ini digunakan, maka Anda harus memilih sistem operasi dan akan dilakukan proses instalasi.



Jika Anda mengunduh file image Raspbian dan memasang ke kartu micro SD menggunakan etcher.io, maka Anda akan *booting* langsung ke desktop dari Raspbian dan tidak perlu menunggu.



Menghubungkan Raspberry Pi dengan internet

Mungkin Anda ingin menghubungkan Raspberry Pi ke jaringan lokal atau tersambung dengan internet. Anda dapat menggunakan beberapa pilihan, diantaranya menggunakan kabel Ehternet. Raspberry Pi memiliki Ethernet port disamping USB port. Jika Raspberry Pi dekat dengan sebuah router, access point, or switch, Anda dapat terhubung ke jaringan dengan kabel Ethernet.Jika Anda menghubungkan Ethernet cable ke Raspberry Pi dan sisi yang lain terhubung dengan access point, Raspberry Pi akan secara otomatis terhubung dengan jaringan.

Jika Anda menggunakan Raspberry Pi 3, sudah built-in WiFi. Jika Anda menggunakan versi Raspberry Pi sebelumnya, Anda memerlukan USB WiFi dongle. Beberapa jenis WiFi dongles, ketika terhubung dengan the Raspberry Pi, dapat secara mudah terhubung secara plug-and-play. Jenis yang lainnya memerlukan driver dan mungkin tidak kompatibel dengan Raspberry Pi. Pastikan dulu sebelum Anda membelinya.

Menghubungkan untuk output audio

Raspberry Pi dilengkapi dengan 3.5mm audio port. Hal ini memungkinkan Anda memasang speakers atau headphones ke Raspberry Pi, sehingga Anda dapat mendengarkan suara music atau program yang luar biasa pada "Sonic Pi". Pada Raspberry Pi 3 sudah terdapat fitur Bluetooth, Anda dapat mengakses Bluetooth speakers or headphones. Kesuksesan Anda bergantung dari jenis speakers yang Anda gunakan, jadi pastikan Anda sudah mencari tahu apakah peralatan tersebut dapat terkoneksi dengan Raspberry Pi.

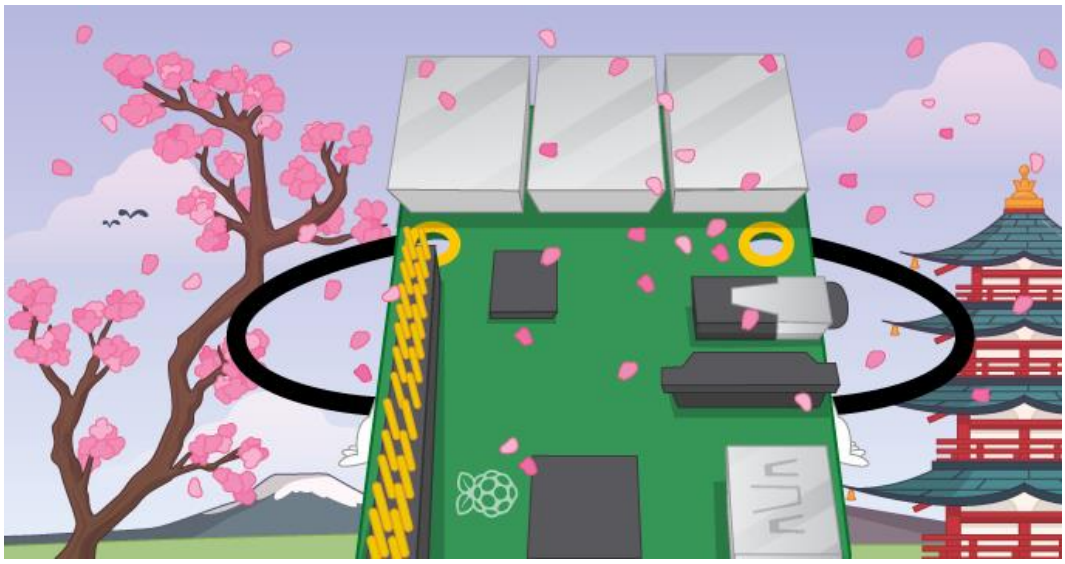
Menambahkan media penyimpanan tambahan

Media penyimpanan yang dapat Anda gunakan untuk keperluan minimalis di Raspberry Pi adalah kartu micro SD 8GB. Jika hal ini kurang. Ada beberapa pilihan untuk meningkatkan media penyimpanan dari Raspberry Pi.

Anda dapat menggunakan micro SD yang lebih besar kapasitasnya, saat ini yang paling besar 128 GB. Sebelum membelinya pastikan Anda sudah mengecek informasi bahwa spesifikasi micro SD tersebut kompatibel dengan Raspberry Pi.

Kapastias yang lebih besar adalah USB flash, atau kita kenal dengan flask-disk, saat ini yang paling besar 1TB. Ada banyak pilihan dengan menggunakan USB flash sebagai media penyimpanan sekunder (cadangan). Banyak tipe USB flash yang dapat langsung plug-and-play dengan Raspberry Pi.
Yang paling besar adalah external hard drives (hard-disk) yang dapat dikoneksikan via kabel USB cable untuk transfer data. Beberapa external hard drives sudah secara otomatis terhubung dengan power supply, dan hal ini tidak akan menjadi masalah. Namun beberapa diantaranya mendapatkan power supply via USB PORT, dan ini akan mengambil arus dari Raspberry Pi. Anda harus lebih hati-hati dengan hal ini. Bacalah petunjuk penggunaan apakah sesuai dengan Raspberry Pi. Beberapa external hard drives didesain secara khusus untuk bekerja di Raspberry Pi seperti WD PiDrive 314GB.

3. Sistem Operasi di Raspberry Pi



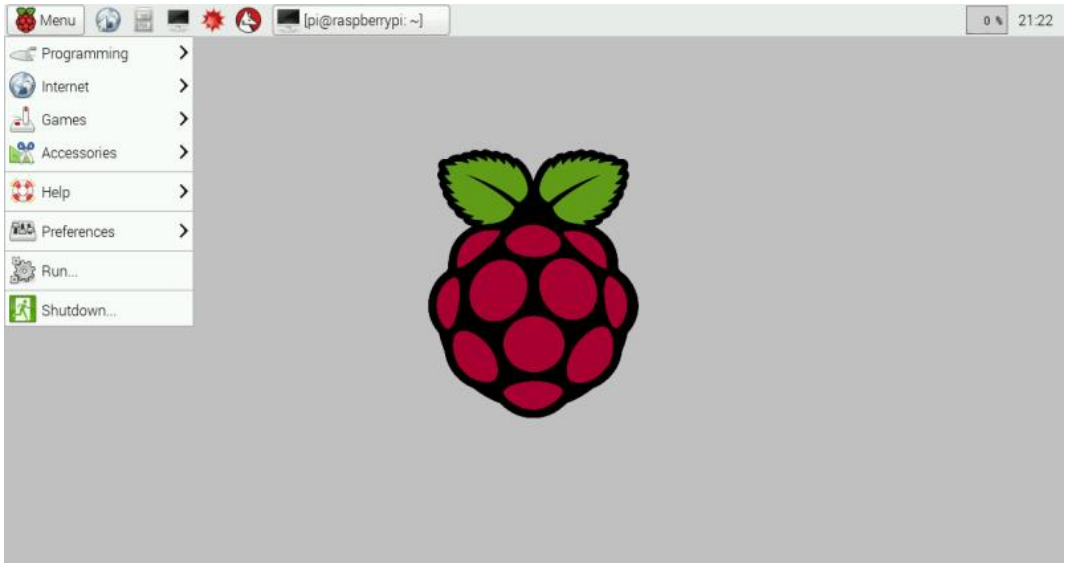
Sistem operasi (OS) adalah seperangkat program dan utilitas dasar yang membuat Raspberry Anda berjalan. Jika Anda baru mengenal Raspberry Pi, ada kemungkinan Anda mendownload atau membeli kartu SD yang di dalamnya sudah terpasang) NOOBS (*New Out Of the Box Software*). Perangkat lunak ini akan membantu Anda memulai beroperasi dengan Raspberry Pi dengan mudah. Di dalamnya mencakup banyak sistem operasi yang beraneka ragam. Yang harus Anda pilih tentu saja bergantung pada keperluan Anda. Inilah cara membuat pilihan yang tepat. NOOBS adalah cara yang bagus untuk menguji sistem operasi baru dan mengenal Raspberry Pi Anda, jadi bermanfaat untuk sedikit bereksperimen dengan mereka semua. Jika Anda mencari pilihan lain, ada satu ton proyek siap pakai lainnya yang tidak termasuk dalam NOOBS yang layak untuk dilihat.

Pemula disarankan memulai dengan NOOBS - Perangkat Lunak Baru di Luar Kotak. Anda dapat membeli kartu SD NOOBS pra-instal dari banyak pengecer, seperti Pimoroni, Adafruit dan The Pi Hut, atau download NOOBS di bawah ini dan ikuti panduan pengaturan perangkat lunak dan video panduan penyiapan NOOBS di halaman bantuan kami.

NOOBS adalah installer sistem operasi yang mudah yang berisi Raspbian. Ini juga menyediakan pilihan sistem operasi alternatif yang kemudian dapat diunduh dari internet dan kemudian dipasang.

NOOBS Lite berisi installer sistem operasi yang sama tanpa pre-loaded Raspbian. Ini menyediakan menu pilihan sistem operasi yang sama yang memungkinkan Raspbian dan gambar lainnya untuk didownload dan diinstal.

Raspbian (Raspberry Pi + Debian)



Raspbian adalah sistem operasi "resmi" dari Raspberry Pi dan oleh karena itu kebanyakan orang ingin memulai dari Raspbian. Raspbian adalah versi Linux yang dibangun khusus untuk Raspberry Pi, berbasis Debian yang dioptimalkan untuk perangkat keras Raspberry Pi. Raspbian tidak berafiliasi dengan *Raspberry Pi Foundation*. Raspbian diciptakan oleh tim pengembang kecil yang merupakan penggemar perangkat keras Raspberry Pi, yang berdedikasi untuk tujuan pendidikan *Raspberry Pi Foundation* dan Proyek Debian.

Raspbian dilengkapi dengan semua perangkat lunak yang Anda perlukan untuk setiap tugas dasar dengan komputer. Anda akan mendapatkan *LibreOffice* sebagai perangkat perkantoran (*office suite*), browser web, program email, dan beberapa alat untuk mengajarkan pemrograman kepada anak-anak (*scratch*) dan orang dewasa (*python, java, c/c++*). Bahkan didalamnya terdapat permainan *Minecraft*.

Raspbian menyediakan lebih dari sekedar OS biasa, Raspbian hadir dengan lebih dari 35.000 paket, perangkat lunak pra-kompilasi (*pre-compiled software*) yang digabungkan dalam format yang bagus untuk memudahkan pemasangan di Raspberry Pi Anda. Paket awal dari lebih dari 35.000 paket Raspbian yang dioptimalkan untuk kinerja terbaik di Raspberry Pi selesai pada bulan Juni 2012. Namun, Raspbian masih dalam pengembangan aktif dengan penekanan pada peningkatan stabilitas dan kinerja sebanyak mungkin paket Debian.

Raspbian adalah tulang punggung bagi hampir semua proyek *do-it-yourself* (DIY) yang ada di luar sana, jadi jika Anda ingin membuat sesuatu, sebaiknya Anda mulai dari Raspbian. Karena sistem operasi ini sangat banyak yang menggunakan, juga mudah menemukan panduan dan tips pemecahan masalah. Namun, jika Anda baru mengenal Linux, Raspbian akan sedikit membingungkan bagi Anda. "*Rpi Beginners wiki*" atau "Raspberry Pi Tutorials Channel" merupakan titik awal belajar yang disarankan, begitu juga berbagai sumber resmi di website Raspberry Pi. Referensi tersebut akan memandu Anda melalui semua hal yang Anda perlukan untuk menggunakan Raspbian.

OSMC (Open Source Media Center)

OSMC (Open Source Media Center) adalah perangkat lunak media center berbasis Kodi (sebelumnya XBMC), tapi lebih mudah untuk dipasang dan digunakan. Sebenarnya sama sekali tidak mirip Kodi, bahkan banyak yang berpendapat lebih baik, karena memberikan kemudahan penggunaan file-file multi-media. Jika Anda baru mengenal media center atau bukan termasuk orang teknik, OSMC adalah yang sistem operasi yang mungkin cocok buat Anda.

Dibandingkan Kodi, OSMC memiliki antarmuka sederhana dan mudah untuk dioperasikan. Anda mendapatkan menu di sisi kiri layar yang memungkinkan Anda memilih media (video/musik/gambar), masuk ke dalam pengaturan, atau melihat program lainnya. Semuanya tersusun secara rapi dan jelas. Tentu saja, Anda masih bisa memasang Kodi untuk *media stream* dan mengatur secara remote sehingga Anda tidak perlu menggunakan keyboard. Sebenarnya, OSMC memiliki preset untuk beberapa remote populer sehingga Anda bahkan tidak perlu menggaruk kepala Anda untuk mencoba memasangnya. Sedangkan untuk media lokal, Anda bisa memutar video dan foto dari USB flash drive.

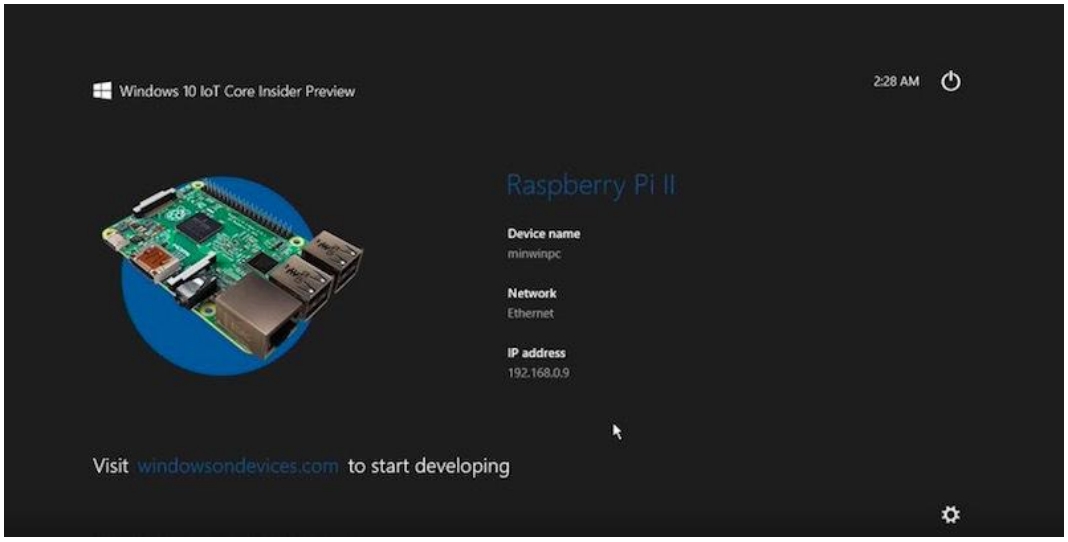
OSMC masih memiliki banyak ruang untuk bermain-main. Sistem operasi ini menjalankan versi lengkap Debian secara "*under the hood*", sehingga Anda dapat mengatur SSH, FTP, Samba Sharing, dan banyak lagi jika Anda pengguna tingkat lanjut.

OpenELEC (Open Embedded Linux Entertainment Center)

Jika OSMC tidak sesuai dengan kebutuhan Anda, OpenELEC (Open Embedded Linux Entertainment Center) patut dicoba. OpenELEC adalah modifikasi langsung dari Kodi, jadi jika Anda mengenal Kodi dan cara kerjanya, Anda seperti berada di rumah sendiri. Jika OSMC adalah pusat media yang kaya fitur dan dapat disesuaikan sesuai dengan yang Anda inginkan, OpenELEC dibangun untuk satu hal saja, yaitu : media bermain. Jika Anda memiliki satu ton film atau musik yang sudah ada di hard drive dan hanya menginginkan cara mudah memainkannya di televisi Anda, OpenELEC adalah pilihannya.

Kami sudah menggali OpenELEC sebelumnya, namun daya tarik utamanya adalah kecepatannya. OpenELEC mengambil Kodi dan menghapus banyak pilihan penyesuaian agar tetap cepat, namun terkesan mudah dan sederhana. Akan tetapi, OpenELEC ini tidak semaksimal OSMC, jadi Anda tidak dapat membuat perubahan pada sistem seperti mengubah kecepatan overclock Pi tanpa masuk ke dalam menu yang kompleks. OpenELEC juga membatasi akses ke layanan tertentu, seperti SSH, jadi tidak mudah untuk dilakukan pengaturan.

Windows 10 IoT Core



Windows 10 IoT adalah versi khusus Windows yang dibuat untuk Raspberry Pi. Namun ini bukan versi lengkap dari Windows. Sebagai gantinya, sistem operasi ini dibuat sebagai platform pengembangan untuk coder ataupun programmer untuk membuat perangkat dapat terhubung ke internet dengan menggunakan Raspberry Pi dan Windows 10. Windows 10 IoT hanya kompatibel dengan Windows 10 dan Anda tidak dapat melakukan apapun, kecuali jika Anda memiliki komputer lain yang terpasang dengan Windows 10 di dalamnya .

Saat Anda pertama kali masuk ke Windows 10 IoT, semua yang akan Anda lihat di layar Raspberry Pi Anda adalah seperti pada layar di atas. Anda tidak dapat mengendalikan atau melakukan sesuatu pada Pi dengan sendirinya. Untuk itu, Anda perlu mendownload dan menginstal Visual Studio di PC atau laptop Windows 10 Anda. Setelah Anda melakukannya, Anda dapat memprogram dan mengendalikan Raspberry Pi Anda dari Visual Studio di Windows 10. Ini berarti Anda dapat membuat lampu berkedip, terhubung ke tombol push, motor kontrol, dan banyak hal lainnya.

Untuk mulai menggunakan Windows IoT Core, Microsoft memiliki koleksi proyek luar biasa dan mengajarkan cara menggunakannya. Coba lihat proyek-proyek tersebut dan apakah ada yang menarik bagi Anda, kemudian coba untuk memutuskan apakah Windows 10 IoT Core layak untuk dipasang.

RISC OS

RISC OS tidak dibangun di Linux, juga tidak berasal dari prototipe *tinkerers electronic*. Sebagai gantinya, ini adalah sistem operasi tersendiri, agak aneh memang, tapi bisa menyenangkan untuk bermain-main dengannya. RISC OS tidak memiliki banyak kesamaan dengan sistem operasi lain seperti Linux, OS X, atau bahkan Windows. Awalnya RISC OS dirancang pada tahun 1987 dan berakar pada BBC Micro. RISC OS jauh lebih sederhana daripada sistem operasi modern. Sebuah aplikasi tunggal dapat mengambil alih keseluruhan sistem operasi, hanya berfungsi sebagai sistem pengguna tunggal (*single user*), aplikasi hanya berupa direktori dengan tanda seru di depan namanya, dan tidak memiliki banyak keamanan. RISC OS juga super ke drag and drop, dimana jika ingin save, anda drag icon "save as" ke folder. Pada dasarnya, ini adalah sistem operasi kecil yang membingungkan, tapi tetap menakjubkan.

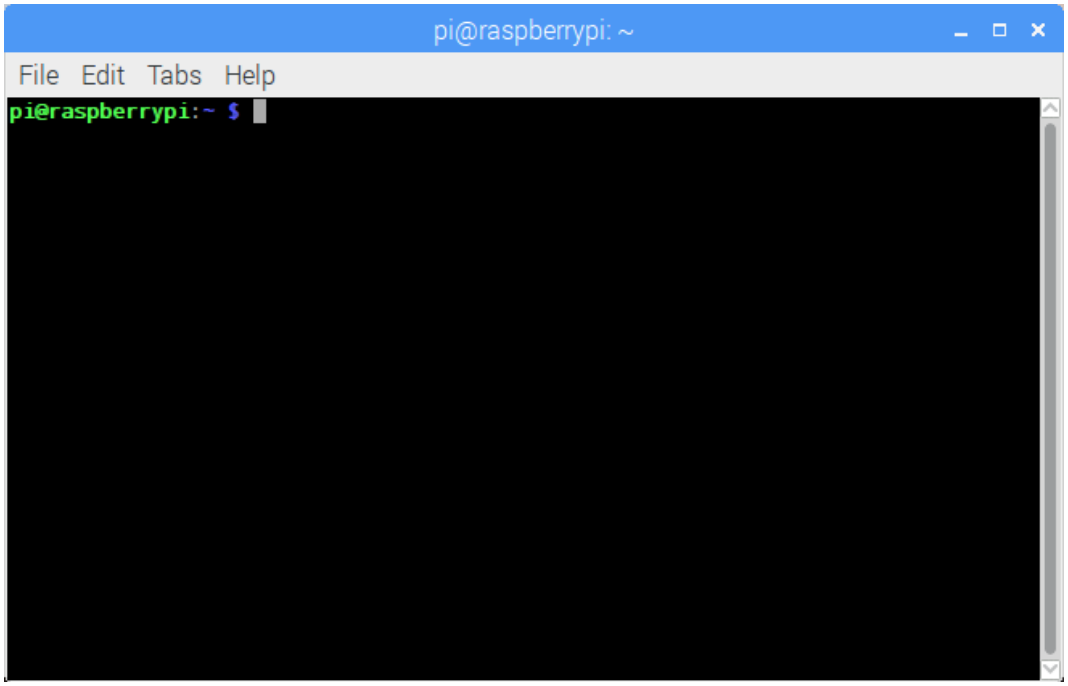
Kebanyakan orang tidak perlu menginstal RISC OS. Anda tidak dapat benar-benar menggunakannya sebagai sistem operasi utama, tidak ada banyak dukungan perangkat lunak modern, dan tidak sesuai dengan sistem operasi lain yang tersedia saat ini. Buat Anda yang penasaran, sangat menyenangkan untuk bermain-main dengannya. Untuk panduan pemula, *Ident Showcase* memiliki panduan yang bagus di YouTube, halaman selamat datang RISC OS memandu Anda melalui beberapa hal mendasar, atau melihat forum OS RISC untuk mendapatkan tip.

4. Familiar dengan Linux Terminal

Terminal atau biasa disebut *command-line* di komputer memungkinkan pengguna untuk mengendalikan sistem di dalam komputer, dalam hal ini adalah Raspberry Pi. Pengguna Windows mungkin telah menemukan pengguna *Command Prompt* atau *Powershell* dan Mac OS mungkin tidak mengenal Terminal. Semua perangkat lunak ini memungkinkan pengguna untuk secara langsung memanipulasi sistem di dalam komputer melalui penggunaan perintah. Perintah ini dapat digabungkan bersama dan atau digabungkan menjadi skrip yang rumit yang berpotensi menyelesaikan permasalahan dengan lebih efisien daripada perangkat lunak tradisional.

Membuka Jendela Terminal

Pada Raspberry Pi, dengan menjalankan sistem operasi Raspbian, aplikasi terminal secara *default* adalah LXTerminal. Hal ini dikenal sebagai 'terminal emulator', artinya bahwa LTXTerminal akan mengemulasi terminal gaya lama, menjadi lingkungan grafis. Aplikasi ini bisa ditemukan di desktop Raspberry Pi, jika Anda memulainya akan terlihat seperti ini:



Anda akan melihat prompt berikut ini:

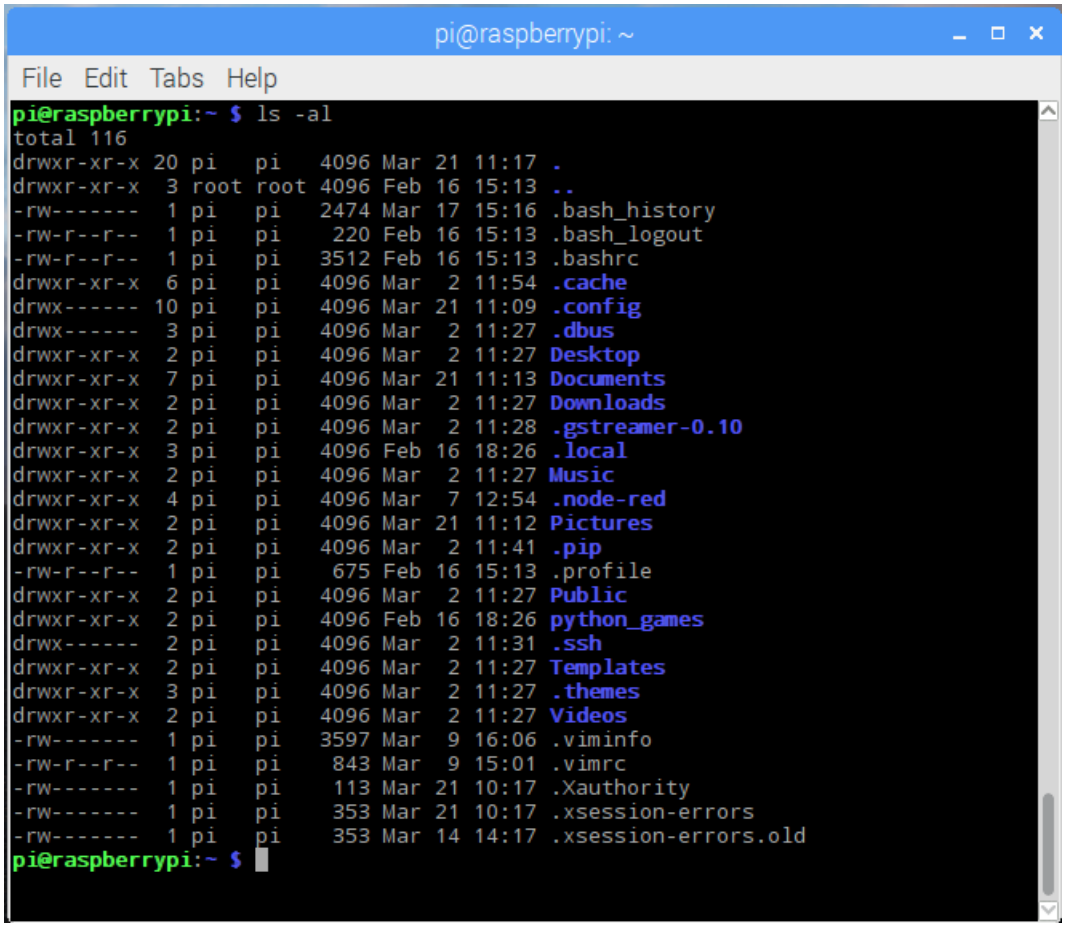
```
pi @ raspberrypi ~ $
```

Ini menunjukkan nama pengguna dan nama host dari Raspberry Pi. Disini username adalah “pi” dan nama hostnya adalah “raspberrypi”.

Sekarang, coba jalankan beberapa perintah. Ketik “pwd” pada direktori kerja sekarang, diikuti dengan tombol Enter. Perintah ini akan menampilkan sesuatu seperti / home / pi.

Navigasi dan Jelajahi Raspberry Pi Anda

Salah satu aspek utama penggunaan terminal adalah dapat menjelajah sistem file Anda. Pertama, jalankan perintah berikut: *ls -al*. Anda akan melihat tampilah seperti ini :



Perintah *ls* akan menampilkan isi direktori tempat Anda berada saat ini (direktori kerja Anda sekarang). Komponen *-al* dari perintah adalah apa yang dikenal sebagai “flags”. Flags memodifikasi perintah yang sedang dijalankan. Dalam kasus ini, “l” menampilkan isi direktori dalam daftar, menampilkan data seperti ukurannya dan kapan diedit terakhir, dan menampilkan semua file, termasuk yang diawali dengan huruf “a”, yang dikenal sebagai 'dotfiles'. Dotfiles biasanya bertindak sebagai file konfigurasi untuk perangkat lunak dan sebagaimana ditulis dalam teks, Dotfiles biasanya dapat diubah hanya dengan mengeditnya.

Untuk menjelajah ke direktori lain perintah untuk merubah direktori dengan menggunakan “cd”. Anda dapat menentukan direktori yang ingin Anda gunakan dengan jalur yang 'absolut' atau 'relatif'. Jadi jika Anda ingin berpindah ke direktori *python_games*, Anda bisa melakukan dengan *cd / home / pi / python_games* atau hanya *cd python_games* (jika Anda saat ini berada di */ home / pi*). Ada beberapa kasus khusus yang mungkin berguna yaitu tanda *tilde* ~ bertindak sebagai alias untuk direktori home Anda, jadi *~ / python_games* sama dengan */ home / pi / python_games*. Dan tanda dua titik “. .” adalah alias untuk direktori saat ini dan

direktori induknya masing-masing. Jika Anda berada di */ home / pi / python_games*, "cd .." akan membawa Anda ke / home / pi.

History dan *Auto-Complete*

Daripada mengetikkan setiap perintah, terminal memungkinkan Anda memanggil perintah sebelumnya yang telah Anda jalankan dengan menekan tombol atas (⬆) atau bawah (⬇) pada keyboard Anda. Jika Anda menulis nama file atau direktori sebagai bagian dari perintah maka menekan TAB akan mencoba untuk melengkapi secara otomatis nama dari apa yang Anda ketik. Misalnya, jika Anda memiliki file dalam direktori yang disebut *aLongFileName* kemudian menekan tab setelah mengetikkan karakter "a", Anda dapat memilih dari semua nama file dan direktori yang dimulai dengan direktori aktif, yang memungkinkan Anda memilih *aLongFileName*.

Perintah sudo *(Super User DO)*

Beberapa perintah yang membuat perubahan permanen pada keadaan sistem Anda mengharuskan Anda memiliki hak akses *root* atau yang biasa kita kenal dengan *admin* untuk menjalankannya. Perintah *sudo* untuk sementara memberi akun Anda (jika Anda belum login sebagai *root*) kemampuan untuk menjalankan perintah ini, asalkan nama pengguna Anda ada dalam daftar pengguna ('sudoers'). Bila Anda menambahkan *sudo* ke awal perintah dan tekan enter Anda akan dimintai kata sandi Anda, jika sudah masuk dengan benar maka perintah yang ingin Anda jalankan akan dijalankan dengan menggunakan hak akses *root*. Hati-hati meskipun, beberapa perintah yang memerlukan *sudo* tidak dapat memperbaiki kerusakan pada sistem anda, jadi hati-hati! Informasi lebih lanjut tentang *sudo* dan *root* user dapat anda cari pada *root linux*.

Memasang Perangkat Lunak Melalui "apt-get"

Anda tidak perlu mengunjungi "Pi Store" untuk mendownload software baru. Anda dapat menggunakan perintah *apt-get*, ini adalah sebuah *package manager* yang disertakan dengan distribusi Linux berbasis Debian (termasuk Raspbian). Hal ini memungkinkan Anda untuk menginstal dan mengelola paket perangkat lunak baru pada Raspberry Pi Anda. Untuk menginstal paket baru Anda akan mengetikkan *sudo apt-get install <package-name>* (di mana *<package-name>* adalah paket yang ingin Anda instal).

Menjalankan sudo *apt-get update* akan melakukan update pada daftar paket perangkat lunak yang tersedia di sistem anda. Jika versi baru dari sebuah paket tersedia maka *sudo apt-get upgrade* akan memperbarui paket lama ke versi yang baru. Akhirnya, *sudo apt-get remove <package-name>* akan menghapus paket tersebut dari sistem anda.Informasi lebih lanjut tentang ini dapat ditemukan di bagian penggunaan linux pada apt.

Perintah Berguna Lainnya

Ada beberapa perintah lain yang mungkin berguna bagi Anda, berikut ini tercantum di bawah ini:

cp : membuat salinan file dan meletakkannya di lokasi yang ditentukan (pada dasarnya seperti melakukan *copy-paste*), misalnya - *cp file_a / home / other_user /* akan menyalin *file_a* dari direktori *home* Anda ke pengguna *other_user* (Dengan asumsi Anda memiliki izin untuk menyalinnya di sana). Perhatikan bahwa jika targetnya adalah *folder*, nama filenya akan tetap sama, tapi kalau targetnya adalah nama file, maka akan diberikan file tersebut nama baru.

mv : memindahkan file dan meletakkannya di lokasi yang ditentukan (jadi di mana *cp* melakukan *copy-paste*, *mv* melakukan *cut-paste*). Penggunaannya mirip dengan *cp*, jadi *mv file_a / home / other_user /* akan memindahkan file *file_a* dari direktori *home* Anda ke pengguna yang ditentukan. Perintah *mv* juga digunakan untuk mengganti nama file, yaitu memindahkannya ke lokasi baru, mis. *mv hello.txt story.txt.*

rm : menghapus (*remove*) file yang ditentukan atau direktori saat digunakan dengan *-r*. File yang dihapus dengan cara ini umumnya tidak dapat dipulihkan.

mkdir : Ini membuat direktori baru, misalnha *mkdir new_dir* akan membuat direktori *new_dir* dalam direktori kerja sekarang.

cat : mencantumkan isi file, misalnya *cat some_file* akan menampilkan isi *some_file*.

Perintah lain yang mungkin berguna bisa ditemukan di halaman perintah.

Menemukan Informasi tentang Perintah

Untuk mengetahui lebih banyak informasi tentang perintah tertentu maka Anda dapat menjalankan perintah "*man"* (man adalah kependekan dari manual) yang diikuti oleh perintah yang ingin Anda ketahui lebih banyak (misalnya *man ls*). Halaman manual untuk perintah itu akan ditampilkan, termasuk informasi tentang flag untuk program tersebut dan kegunaannya. Beberapa halaman manual akan memberi contoh penggunaan.

5. Pemrograman Python

Raspberry Pi dengan sistem operasi Raspbian ataupun sistem operasi lainya yang berbasis Linux mendukung Python sebagai bahasa pemrograman utama selain bahasa pemrograman seperti C/C++ . Anda bisa menginstall *python* sebagai sarana mempelajari bahasa pemrograman Python. Untuk instalasi bisa di download disini : https://www.python.org/downloads/

Python adalah bahasa pemograman tingkat tinggi yang dapat digunakan secara luas di berbagai bidang. Python diciptakan pertama kali oleh Guido van Rossum pada tahun 1991. Sintaks dan fungsi pada Python dipengaruhi oleh beberapa bahasa seperti C, C++, Lisp, Perl dan Java. Oleh karena itu, kita dapat menemui konsep pemograman prosedural, fungsional dan *object-oriented* di Python. Python relatif mudah dipelajari bila dibandingkan dengan C++, Java dan PHP karena sintaks Python lebih singkat, lebih jelas dan mudah dipahami oleh programmer pemula.

Meskipun di Indonesia Python tidak sepopuler C/C++ ataupun Java, Python sangat patut untuk dipelajari karena banyak pilihan *library* Python yang bisa kita gunakan yang biasanya tidak dapat kita temui di bahasa lain. Pada bagian ini akan dijelaskan dasar-dasar dari pemograman Python. Buka terminal, lalu ketik perintah *python* untuk membuka console Python.

Membuat Variabel

Variabel di Python tidak menyertakan tipe data secara eksplisit, berbeda dengan C++ atau Java. Semua tipe data sama cara memasukkan nilainya ke dalam variabel.

```
>>> integer = 1
>>> float = 2.5
>>> string = "aku adalah string"
>>> boolean = False
>>> integer
1
>>> float
2.5
>>> string
"aku adalah string"
>>> boolean
False
```

String dapat diapit dengan tanda kutip tunggal (' ') atau tanda kutip ganda (" "). Kita juga bisa mendeklarasikan string *multiline* dengan mengapit string dalam tanda kutip tunggal atau tanda kutip ganda sebanyak tiga kali (''' ''' atau """ """).

```
>>> multiline = '''
... Aku seorang kapiten
... Mempunyai pedang panjang
... Kalau berjalan prok-prok-prok
... Aku seorang kapiten
... '''
...
>>> multiline
'Aku seorang kapiten
mempunyai pedang panjang
kalau berjalan prok-prok-prok
Aku seorang kapiten
'
```

Menulis Komentar

Python menggunakan tanda # untuk komentar. Sebuah komentar merupakan baris teks yang tidak dibaca sebagai kode oleh Python. Baris tersebut hanya diperuntukan untuk dibaca oleh manusia. Komentar membuat program yang Anda buat lebih mudah untuk dimengerti. Ketika Anda melihat kembali kode yang telah dibuat atau orang lain ingin berkolaborasi dengan Anda, maka mereka bisa membaca komentarnya dan dengan mudah mengartikan apa yang kode Anda buat. Contoh komentar satu baris:

```
#Variable berikut digunakan untuk menyimpan nomer misterius
#inilah gunanya komentar, kita jadi tahu apa fungsi dari variable ini

mysterious_variable = 42
```

Tanda # hanya akan mengkomentari satu baris saja. Sementara dalam penggunaannya tentunya akan memerlukan komentar dengan panjang lebih dari satu baris, memulai dengan tanda # setiap barisnya, tentunya akan sangat menyusahkan. Maka dari itu, untuk komentar multi-baris, Anda bisa memasukan seluruh blok komentar ke dalam satu set tanda yang terdiri dari tiga tanda petik dua (""""). Contoh penggunakan komentar multi-baris pada Python :

```
""" Ini adalah contoh
Komentar multi-baris
pada Python """

my_var = 123
```

Membuat Struktur Data : List

Pada Python, kita tidak dapat menemui konsep array. Sebagai gantinya, python menyediakan struktur data list. Inisialisasi untuk list sama seperti variabel biasa.

```
>>> list = [1, 2, 3, 4]
>>> list
[1, 2, 3, 4]
>>> list = [1.5, 2.4, 0.5]
>>> list
[1.5, 2.4, 0.5]
```

Kita juga dapat membuat deret integer pada list secara otomatis dengan fungsi *range*.

```
>>> list = range(5)
>>> list
[0, 1, 2, 3, 4]
>>> list = range(0,5)
>>> list
[0, 1, 2, 3, 4]
>>> list = range(0,10,2)
[0, 2, 4, 6, 8]
```

Berbeda dengan C++ dan Java, pada list kita dapat menampung elemen yang berbeda tipe data dalam satu variabel.

```
>>> list = [1, 'dua', 3.0]
>>> list
[1, 'dua', 3.0]
```

Cara akses elemen pada list sama seperti pada array, yaitu dengan menyertakan indeks elemen. Indeks dimulai dari nilai 0.

```
>>> list[0]
1
>>> list[1]
'dua'
```

List juga bisa mengandung list-list yang lain. Hal ini dikenal sebagai *multi-dimensional list*.

```
>>> list = [1, [1, 2, 3.0], [1.5, 'dua']]
```

Kita dapat menambahkan atau mengurangi elemen pada list yang sudah dibuat sebelumnya dengan menggunakan fungsi *append(), insert()* dan *remove().*

```
>>> list = [0, 1]
>>> list
[0, 1]
>>> list.append(2) # menambahkan elemen pada akhir list
>>> list
[0, 1, 2]
>>> list.insert(1, 'aku ditengah') # menambahkan elemen pada indeks 1
>>> list
[0, 'aku ditengah', 1, 2]
>>> list.remove('aku ditengah') # menghapus elemen yang nilainya 'aku ditengah'
>>> list
[0, 1, 2]
```

Untuk mengambil elemen pertama dari suatu list, kita dapat menggunakan *pop*.

```
>>> list = [0, 1]
>>> list.pop()
0
```

Membuat Struktur Data : Dictionary

Selain list, Python memiliki struktur data *dictionary*. Mirip seperti konsep *hash*, tiap elemen pada dictionary memiliki alias atau *key*.

```
>>> dict = {'satu': 1, 'dua': 2, 'tiga': 3}
```

Berbeda dengan list, cara akses nilai pada dictionary adalah dengan menyertakan nama key-nya.

```
>>> dict['satu']
1
>>> dict['dua']
2
```

Elemen pada dictionary dapat terdiri dari tipe data yang berbeda-beda, bahkan nama key juga tidak harus selalu string. Kita juga bisa memasukkan list sebagai elemen pada dictionary.

```
>>> dict = {'satu': 1.0, 'dua': [1, 'dua', [1, 2, 3]], 3: 'tiga'}
>>> dict[3]
'tiga'
>>> dict['dua']
[1, 'dua', [1, 2, 3]]
>>> dict['dua'][1]
['dua']
>>> dict['dua'][2]
[1, 2, 3]
>>> dict['dua'][2][1]
[2]
```

List dan dictionary adalah struktur data yang paling sering digunakan pada Python, namun struktur data pada Python tidak hanya terbatas pada itu saja. Untuk mempelajari struktur data lebih lanjut, Anda dapat membuka tutorial Python lebih lanjut.

Menggunakan If .. elif .. else ..

Fungsi *if* pada Python lebih sederhana penulisannya dibandingkan C++ atau Java. Untuk mengapit pernyataan di dalam *if* tidak menggunakan kurung kurawal. Penulisan pernyataan yang masuk pada fungsi *if* lebih menjorok ke dalam (menggunakan *whitespace*).

```
>>> var = 1
>>> if var < 2:
...     var = var + 1
... elif var < 3:
...     var = var + 2
... else:
...     var = 0
>>> var
2
```

Selain *case-sensitive*, Python sangat memperhatikan *whitespace* untuk memisahkan pernyataan dengan deklarasi fungsi. Oleh karena itu Anda harus selalu konsisten dalam memilih whitespace (pilih salah satu antara spasi atau tab). Pastikan juga semua indentasi pada fungsi memiliki jumlah whitespace yang sama.

Mengulang dengan For

Fungsi *for* sedikit berbeda dengan C++ atau Java, karena kita tidak menginisialisasi kondisi berhenti iterasi. Agar kita bisa menginisialisasi jumlah perulangan, kita bisa menggunakan range.

```
>>> var = 0
>>> for el in range(1, 10):
...     var = var + 1
...
>>> var
9
```

Untuk menghentikan atau melanjutkan iterasi pada suatu kondisi tertentu, kita dapat menggunakan break dan continue.

```
>>> var = 0
>>> for el in range(1, 10):
...     if var > 5:
...             break
...     elif var == 3:
...             continue
...     var = var + 1
...
>>> var
3
```

Di Python*, for* akan menjalankan pernyataan di dalamnya sebanyak elemen pada list dan akan berhenti setelah nilai indeks elemen melebihi nilai indeks maksimal pada list.

```
>>> list = ['maine coon', 'munchkin', 'siamese']
>>> var = 0
>>> for el in list:
...     var = var + 1
...
>>> var
2
```

Mengulang dengan While

Fungsi while akan terus secara iteratif menjalankan pernyataan di dalamnya selama kondisi pada while belum terpenuhi.

```
>>> var = 10
>>> while var > 0:
...     var = var - 1
...
>>> var
0
```

Mencetak dengan Print

Fungsi print digunakan untuk mencetak string pada terminal.

```
>>> print 'Halo dunia!'
Halo dunia!
```

Kita juga bisa mencetak string multiline dengan menggunakan fungsi yang sama.

```
>>> print '''
... Aku seorang kapiten
... Mempunyai pedang panjang
... Kalau berjalan prok-prok-prok
... Aku seorang kapiten
... '''
Aku seorang kapiten
Mempunyai pedang panjang
Kalau berjalan prok-prok-prok
Aku seorang kapiten
```

Menyambungkan 2 atau lebih string dapat dilakukan dengan menggunakan operator plus (+).

```
>>> print 'Halo ' + 'dunia!'
Halo dunia!
```

Untuk menambahkan keluaran yang bukan string, kita dapat menambahkan koma.

```
>>> print 'Kucing yang kami pelihara di rumah berjenis: ', ['main coon',
'siamese', 'munchkin']
Kucing yang kami pelihara di rumah berjenis: ['main coon', 'siamese', 'munchkin']
```

Selain itu, kita juga bisa memasukkan variabel di tengah-tengah string (*formatting*) dengan menggunakan tanda kurung kurawal {}.

```
>>> x = 1
>>> y = 2
>>> print '{} + {} = {}'.format(x, y, x + y)
1 + 2 = 3
```

Mengubah urutan output dapat dilakukan dengan memberikan indeks di dalam kurung kurawal.

```
>>> print '{1} + {0} = {2}'.format(x, y, x + y)
2 + 1 = 3
```

Indeks dapat diganti dengan key seperti pada *dictionary*.

```
>>> print '{0} + {1} = {hasil}'.format(x, y, hasil = x + y)
1 + 2 = 3
```

Bagi Anda yang familiar dengan format output yang mendeklarasikan tipe data secara eksplisit seperti pada C/C++, Anda juga bisa melakukan hal yang mirip di Python seperti dibawah ini:

```
>>> print '%d + %d = %d' % (x, y, x + y)
1 + 2 = 3
```

Membaca dan Menulis File

Untuk menulis teks pada file, kita dapat menggunakan write() dengan terlebih dahulu membuat file pada direktori.

```
>>> f = open('contoh_file.txt', 'w')
>>> for el in range(1,11):
...     f.write('Ini adalah baris teks {} pada file.\n'.format(el))
...
>>> f.close()
```

Untuk membaca keseluruhan teks pada file, kita dapat menggunakan read().

```
>>> f = open('contoh_file.txt', 'r')
>>> f.read()
Ini adalah baris teks 1 pada file.
Ini adalah baris teks 2 pada file.
Ini adalah baris teks 3 pada file.
Ini adalah baris teks 4 pada file.
Ini adalah baris teks 5 pada file.
>>> f.close()
```

Untuk membaca satu baris teks pada file, kita dapat menggunakan readline().

```
>>> f = open('contoh_file.txt', 'r')
>>> f.readline()
Ini adalah baris teks 1 pada file.
>>> f.close()
```

Untuk membaca satu per satu baris teks pada file, kita dapat menggunakan for.

```
>>> f = open('contoh_file.txt', 'r')
>>> for teks in f:
...     print teks
...
Ini adalah baris teks 1 pada file.
Ini adalah baris teks 2 pada file.
Ini adalah baris teks 3 pada file.
Ini adalah baris teks 4 pada file.
Ini adalah baris teks 5 pada file.
```

Apabila kita lihat secara teliti, ada perbedaan cara antara membuka file untuk menulis dan membaca. Parameter kedua yang dimasukkan pada fungsi *open()* adalah mode pembacaan file. Di bawah ini adalah daftar mode yang dapat digunakan pada fungsi *open()* :

- ‘r’ - hanya untuk membaca file (read)
- ‘w’ - hanya untuk menulis file (write)
- ‘a’ - untuk menambahkan teks pada akhir teks di dalam file (append)
- ‘r+’ - untuk membaca dan menulis file

Selalu gunakan fungsi *close()* setelah Anda membuka file untuk menghindari terjadinya memory leak

Jika kita membuka file dengan pilihan atribut “w”, bila file tersebut sudah ada, maka akan file tersebut akan dibuat kembali dan isi dari file tersebut akan otomatis terhapus. Jika kita ingin membuka file yang sudah ada kemudian menambahkan isi file tersebut kita gunakan pilihan atribut **“a”** (*append*).

Untuk menyimpan file data log, disarankan menggunakan format CSV (Comma Separated Value). Di dalam Python sudah disediakan modul CSV.

```
import csv
import datetime

temp = sensor()
temp = round(temp, 2)
now = datetime.datetime.now()
file = open("temperature.csv", "a")
writer = csv.writer(file, delimiter = ",")
data = [now, temp]
writer.writerow(data)
file.close()
```

Membuat Fungsi

Deklarasi fungsi pada Python dimulai dengan *def* dan diikuti dengan nama fungsi. Sama seperti deklarasi variabel, kita tidak perlu menambahkan tipe data secara eksplisit pada fungsi. Untuk fungsi yang mengembalikan nilai dapat menggunakan *return*.

```
>>> def sambung_string(str1, str2):
...     sambungan = str1 + ' dan ' + str2
...     return sambungan
```

Untuk memanggil fungsi, tulis nama fungsi setelah itu masukkan nilai parameter fungsinya jika ada.

```
>>> hasil = sambung_string('aku', 'kamu')
>>> print hasil
aku dan kamu
```

Kita juga bisa membuat nilai *default* pada parameter sehingga kita punya opsi untuk menimpa nilai default tersebut atau tidak.

```
>>> def sambung_string(str1, str2='kamu'):
...     sambungan = str1 + ' dan ' + str2
...     return sambungan
...
>>> hasil = sambung_string('aku')
>>> print hasil
aku dan kamu
>>> hasil_lain = sambung_string('aku', 'dia')
>>> print hasil_lain
aku dan dia
```

Semua parameter pada Python adalah *passed by reference.* Sehingga apabila di dalam fungsi ada operasi yang melibatkan variabel dengan nama yang sama dengan variabel di luar fungsi, maka nilai pada variabel di luar fungsi akan ikut berubah. Contoh:

```
>>> def tambah_elemen(elemen):
...     list.append(elemen)
...     print 'Output di dalam fungsi', list
...
>>> list = [1, 2]
>>> tambah_elemen(3)
Output di dalam fungsi [1, 2, 3]
>>> print 'Output di luar fungsi', list
Output di luar fungsi [1, 2, 3]
```

Membuat Program Object Oriented

Kita dapat mengimplementasikan konsep object oriented pada Python. Insialisasi kelas pada Python dapat dilihat pada contoh di bawah ini.

```
>>> class Kucing:
...     def __init__(self): # dobel underscore
...             self.jenis = 'domestik'
...             self.bulu = 'kuning'
...             self.buntut = 'pendek'
...             self.nama = 'Ambercat'
...     def deskripsi(self):
...             print 'Hai namaku {}. Aku adalah kucing berjenis {}. Warna buluku
{} dan buntutku {}.\n'.format(self.nama, self.jenis, self.bulu, self.buntut)
...
```

Pada contoh, kelas Kucing juga memiliki dua *method*, yaitu __init__() dan deskripsi(). Method __init__() adalah *constructor* yang akan selalu dipanggil setiap objek Kucing dibuat. Dalam contoh ini, __init__() akan membuat empat atribut, yaitu jenis, bulu, buntut dan nama. Setiap ingin membuat method, Anda harus selalu memasukkan parameter *self* pada deklarasi method.

Cara membuat objek dari kelas Kucing dapat dilihat pada contoh di bawah ini.

```
>>> meong = Kucing()
```

Cara untuk memanggil method pada objek adalah sebagai berikut.

```
>>> meong.deskripsi()
Hai namaku Ambercat. Aku adalah kucing berjenis domestik. Warna buluku kuning dan
buntutku pendek.
```

Supaya kita dapat mengeset nilai atribut jenis, bulu, buntut dan nama, kita dapat memodifikasi kelas Kucing menjadi seperti di bawah ini.

```
>>> class Kucing:
...     def __init__(self, jenis='domestik', bulu='kuning', buntut='pendek',
nama='Ambercat'): # dobel underscore
...             self.jenis = jenis
...             self.bulu = bulu
...             self.buntut = buntut
...             self.nama = nama
...     def deskripsi(self):
...             print 'Hai namaku {}. Aku adalah kucing berjenis {}. Warna buluku
{} dan buntutku {}.\n'.format(self.nama, self.jenis, self.bulu, self.buntut)
...
```

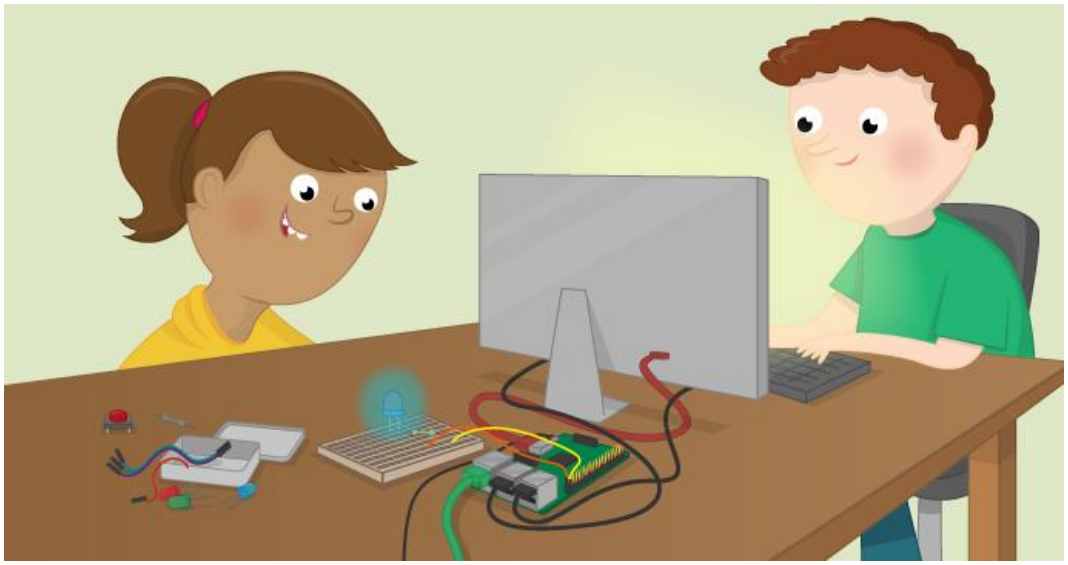
Untuk mengganti beberapa nilai atribut pada inisialisasi objek, kita dapat menggunakan cara berikut.

```
>>> meong = Kucing(bulu='putih', buntut='panjang')
>>> meong.deskripsi()
Hai namaku Ambercat. Aku adalah kucing berjenis domestik. Warna buluku putih dan
buntutku panjang.
```

Bagian ini belum mencakup seluruh fungsi dan konsep pada Python. Untuk mempelajari lebih lanjut mengenai Python, Anda dapat mengunjungi dokumentasi Python lebih lanjut.

```
print 'Selamat belajar!'
```

6. Akses Dunia Luar dengan GPIO



Salah satu alasan Raspberry Pi sangat populer digunakan adalah karena memiliki 40-pin GPIO yang memungkinkan pengguna untuk menghubungkan komponen elektronik dan mengendalikan mereka dengan sebuah program. Kode program untuk proyek komputasi fisik biasanya ditulis dengan Python, akan menjadi jauh lebih mudah jika kita menggunakan *library* yang sudah disediakan oleh GPIO Raspberry Pi.

Sebelum adanya library, untuk menghubungkan dan mengatur perangkat elektronik diperlukan banyak baris kode program. GPIO akan menjalankan semua kode ini untuk Anda, sehingga Anda dapat fokus pada pengendalian perangkat fisik. Hal ini akan membantu Anda untuk mengembangkan perangkat elektronika dengan sedikit kode program. Hal ini membuat programmer pemula lebih mudah untuk memahami. Buku ini, akan membantu Anda memulai *coding* dengan GPIO, membimbing Anda langkah demi langkah melalui beberapa proyek.

Membangun rangkaian elektronika dapat dengan mudah jika Anda tahu apa yang Anda lakukan, tetapi jika Anda membuat rangkaian elektronika, kemungkinan besar Anda harus mengacu pada diagram rangkaian secara umum. Cara yang lebih mudah bagi Anda adalah dengan melihat simbol dari rangkaian elektronika. Hal ini akan jauh lebih memudahkan Anda untuk membaca dan memahami simbol dari sebuah rangkaian. Berikut adalah contoh sederhana dari simbol rangkaian elektronika.

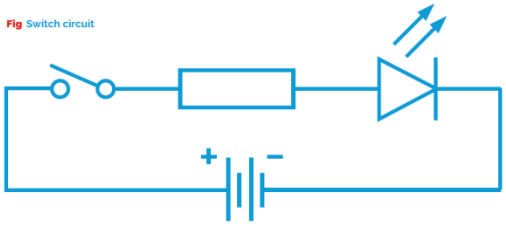
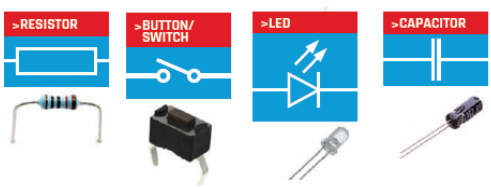
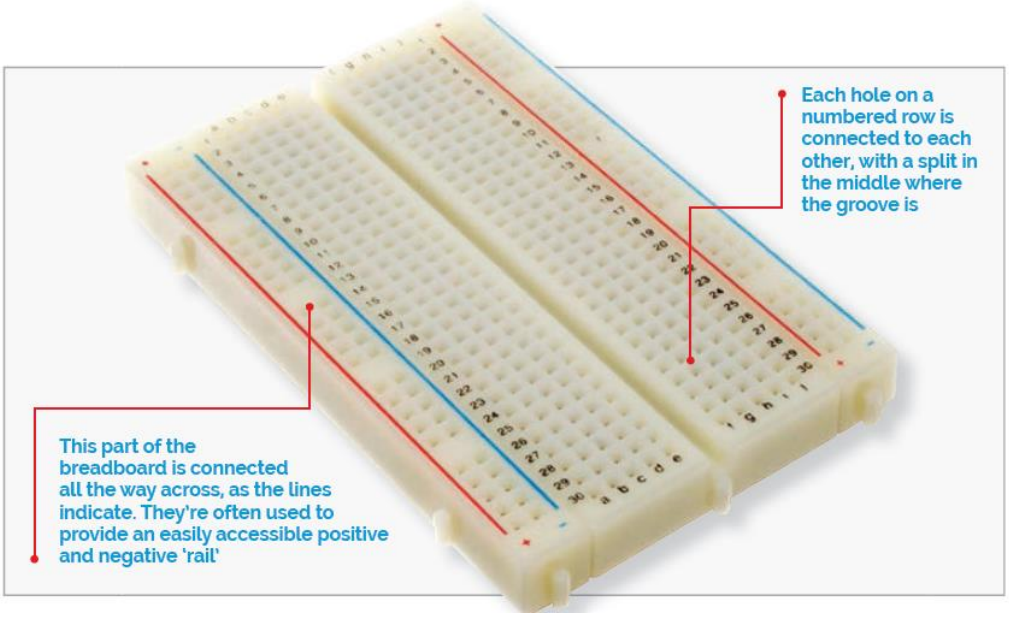


Fig Switch circuit



memiliki sumber daya (baterai di sirkuit ini), saklar, resistor, dan LED. Garis mewakili bagaimana sirkuit terhubung bersama-sama, baik melalui kawat atau cara lain. Beberapa komponen dapat digunakan dengan cara apapun bulat, seperti resistor atau switch. Namun, yang lain memiliki orientasi c spesifik, seperti LED. Dioda hanya membiarkan aliran listrik dari positif ke negatif; untungnya, LED kehidupan nyata memiliki penanda seperti kaki yang lebih panjang atau fl di tepi untuk menunjukkan sisi yang positif, membuat mereka lebih mudah untuk kawat sampai.



Instalasi RPi.GPIO

Untuk memulai menggunakan GPIO, pastikan bahwa *library* RPi.GPIO sudah terinstall pada Terminal di Raspberry Pi. Unduh dan pasang *library* RPi.GPIO dari terminal dengan menjalankan perintah sebagai berikut :

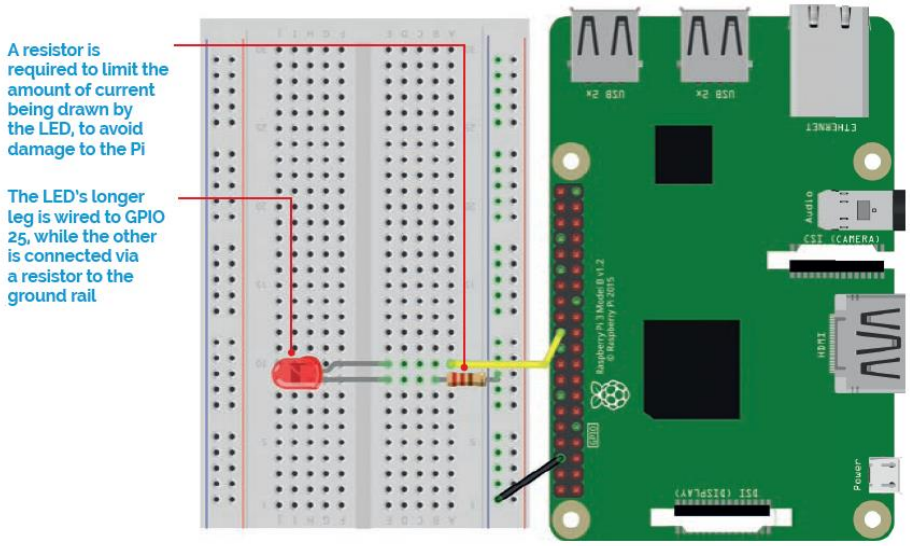
```
$ sudo apt-get install python-dev
$ sudo apt-get install python-rpi.gpio
```

Biasanya versi terbaru dari Raspberry Pi sudah terpasang RPi.GPIO di dalamnya. Dengan demikian perintah di atas hanya akan meng-*update* versi terbaru. *Library* RPi.GPIO ini akan digunakan di seluruh buku ini, karena menggunakan bahasa C yang dibungkus dengan Python membuat pekerjaan IO (*input-output*) menjadi lebih cepat. Namun, Pi tidak dirancang sebagai mikrokontroler, sehingga pin GPIO tidak akan merespon secepat Arduino.

Alternatif *library* lainnya yang dapat melakukan banyak pekerjaan yang sama seperti RPi.GPIO adalah **WiringPi**. Anda dapat mengetahui lebih lanjut tentang *library* ini di *Gordons Project*, karena tidak dibahas pada buku ini. Berikutnya kita akan mencoba beberapa fitur yang ada pada *library* RPi.GPIO.

Menyalakan LED

Ketika membangun rangkaian elektronika, pastikan bahwa kondisi Raspberry Pi OFF, ataupun minimal tidak terhubung langsung dengan rangkaian. Fitur *breadboard* terdapat kolom yang masing-masing terdiri lima lubang yang terhubung. Tempatkan kaki LED merah di kolom nomor yang berdekatan, seperti yang ditunjukkan pada Gambar. Perhatikan bahwa kaki lebih pendek dari LED adalah negatif (-); masukkan salah satu ujung resistor, kemudian tempatkan ujung lainnya di barisan luar ditandai (-) atau *ground* (GND). Gunakan kawat *jumper male-female* untuk menghubungkan pin GND pada Raspberry Pi. Gunakan kawat *jumper* untuk menghubungkan lubang di kolom kaki LED positif (+) ke GPIO pin 25. Rangkai seperti pada gambar di bawah ini :



Jalankan program di bawah ini :

```
$ sudo python
>>> import RPi.GPIO as GPIO
>>> GPIO.setmode(GPIO.BCM)
>>> GPIO.setup(25, GPIO.OUT)
>>> GPIO.output(25, True)
>>> GPIO.output(25, False)
```

Modifikasi program di atas dengan membuat file baru misal “blink.py” dengan menambahkan library “time” agar dapat mengatur waktu tunda (*delay*) pada saat LED mati dan nyala, begitu seterusnya.

```
import RPi.GPIO as GPIO
import time

GPIO.setmode(GPIO.BCM)
GPIO.setup(25, GPIO.OUT)

while (True):
        GPIO.output(25, True)
        time.sleep(0.5)
        GPIO.output(25, False)
        time.sleep(0.5)
```

Megatur Kecerahan LED

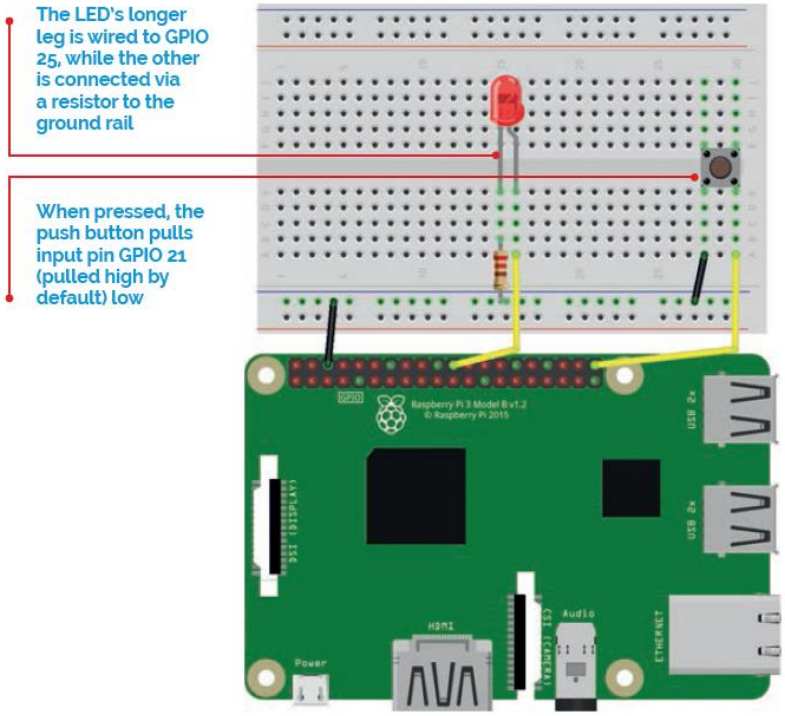
Masih dengan rangkaian yang sama. Cobalah program di bawah ini yang akan mengatur tingkat kecerahan LED mulai dari mati, redup sampai dengan nyala paling terang.

```
import RPi.GPIO as GPIO

led_pin = 25
GPIO.setmode(GPIO.BCM)
GPIO.setup(led_pin, GPIO.OUT)

pwm_led = GPIO.PWM(led_pin, 500)
pwm_led.start(100)

while True:
        duty_s = raw_input("Enter Brightness (0 to 100):")
        duty = int(duty_s)
        pwm_led.ChangeDutyCycle(duty)
```

Tantangan : Modifikasi program di atas agar otomatis dapat menyalakan dan mematikan LED secara bertingkat dari mulai mati, redup sampai paling terang, kemudian mati lagi, dan seterusnya.

Menghubungkan *Push Button*

Tambahkan tombol tekan (*push button*) pada *breadboard* bagian tengah, sehingga antara kakinya tidak ada yang berhubungan. Lalu hubungkan kabel *jumper male-female* dari kolom satu pin pada GPIO Raspberry Pi. Kemudian hubungkan kawat *jumper* male dari pin yang lain (pada sisi yang sama dari alur) ke GND (-). Akhirnya, hubungkan kawat *jumper male-female* dari yang terakhir ke pin GND pada Pi.



Tulis program di bawah ini, kemudian beri nama misalnya “switch.py”, dan jalankan.

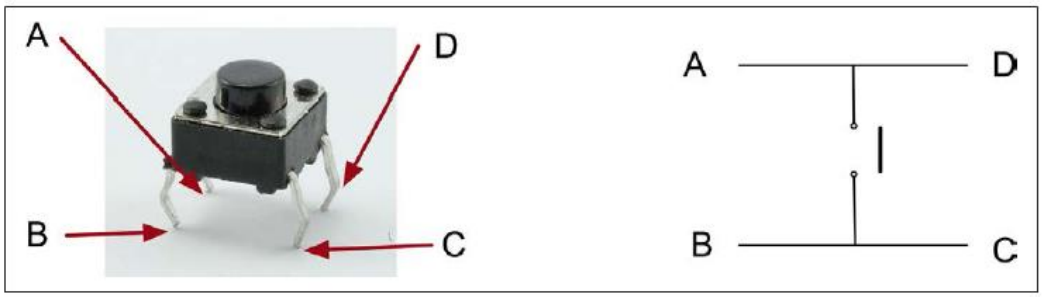
```
import RPi.GPIO as GPIO
import time

led_pin = 25
button_pin = 21
GPIO.setmode(GPIO.BCM)
GPIO.setup(led_pin, GPIO.OUT)
GPIO.setup(button_pin, GPIO.IN, pull_up_down=GPIO.PUD_UP)

while True:
        GPIO.output(25, False)
        input_state = GPIO.input(21)

        if input_state == False:
                GPIO.output(25, True)
                print(Tombol telah ditekan')
                time.sleep(0.2)
```

Kemudian jalankan program dengan perintah dibawah ini, dan coba tekan tombol 1 kali.

```
pi@raspberrypi~$ sudo python switch.py
Tombol telah ditekan
Tombol telah ditekan
Tombol telah ditekan
```

Ketika tombol ditekan, akan ada pesan “*Tombol telah ditekan*” pada terminal. Biasanya akan muncul beberapa kali,

Pada program di atas pin input diberikan “*pulled up 3.3V*” berdasarkan program **pull_up_down=GPIO.PUD_UP** pada saat melakukan pengaturan di GPIO.setup. Hal ini artinya Anda membaca input menggunakan **GPIO.input**, sedankan **False** akan mengembalikan kepada kondisi semula jika tombolnya ditekan. Dengan demikian logikanya berkebalikan.



Setiap pin GPIO dapat dikonfigurasi melalui program apakah memilih *pull-up* atau *pull-down resistors*. Ketika menggunakan pin GPIOsebagai input, Anda dapat mengkonfigurasi resistor ini sehingga satu / salah satu / tidak satu pun resitor yang aktif. Penggunaan parameter **pull_up_down** pada **GPIO.setup** adalah opsional.

Jika parameter ini diabaikan, maka tidak satu pun resistor akan diaktifkan. Hal ini akan mengakibatkan input mengambang, yang berarti bahwa nilainya tidak bisa dipastikan dan akan mengambang antara tinggi dan rendah tergantung pada *noise* dari sinyal listrik. Jika diatur ke **GPIO.PUD_UP**, resistor pull-up diaktifkan; jika sudah diatur untuk **GPIO.PUD_DOWN**, resistor pull-down diaktifkan.

Teknik *De-bouncing*

Kadang-kadang ketika Anda menekan tombol pada *switch*, hasil yang diharapkan terjadi lebih dari sekali, hal ini dikarenakan kontak saklar mengalami pantulan (*bounce*). Agar didapatkan kepastian bahwa yang Anda inginkan adalah 1 kali hasil, maka solusinya adalah perlu dibuat algoritma *de-bouncing*.

```
import RPi.GPIO as GPIO
import time

GPIO.setmode(GPIO.BCM)

switch_pin = 21
led_pin = 25

GPIO.setup(switch_pin, GPIO.IN, pull_up_down=GPIO.PUD_UP)
GPIO.setup(led_pin, GPIO.OUT)

led_state = False
old_input_state = True # pulled-up

while True:
        new_input_state = GPIO.input(switch_pin)
        if new_input_state == False and old_input_state == True:
                led_state = not led_state
                time.sleep(0.2)
        old_input_state = new_input_state
        GPIO.output(led_pin, led_state)
```

Menggunakan *Interrrupt*

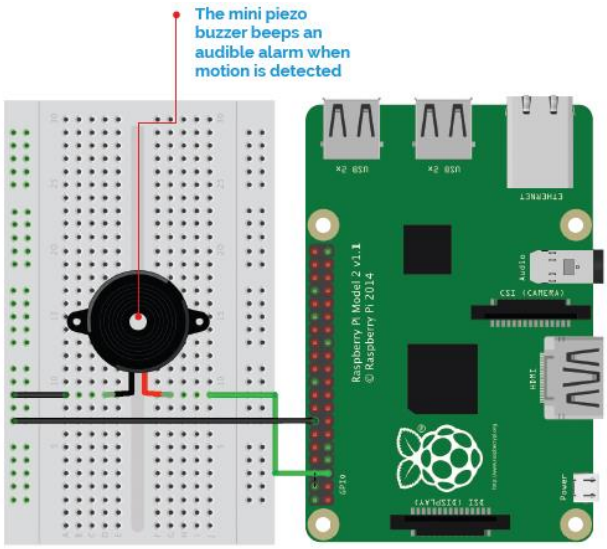
Jika Anda ingin menanggapi beberapa peristiwa (seperti push button yang ditekan), tanpa harus terus menerus mengecek (*polling*) untuk melihat apakah keadaan pin input telah berubah ataukah mendapat masukan. Hal ini akan mengakibatkan proses CPU akan meningkat, dan kinerjanya semakin melambat. Solusinya dapat menggunakan teknik interrupt dengan fungsi *add_event_detect* yang ada pada *Library* RPi.GPIO.

```
import RPi.GPIO as GPIO
import time

GPIO.setmode(GPIO.BCM)

def my_callback(channel):
        print('You pressed the button')

GPIO.setup(18, GPIO.IN, pull_up_down=GPIO.PUD_UP)
GPIO.add_event_detect(18, GPIO.FALLING, callback=my_callback)

i = 0
while True:
        i = i + 1
        print(i)
        time.sleep(1)
```

Jalankan program di atas, kemudian pada saat looping berjalan tekan tombol. Amati apa yang terjadi.

Mengaktifkan *Buzzer*

Jika Anda ingin membuat suara "dengung" dengan Raspberry Pi, Anda bisa menghubungkan pin GPIO dengan *buzzer piezo-electric*. Kebanyakan *buzzer piezo-electric* bekerja dengan baik jika ditunjukkan pada gambar dibawah ini.



Anda dapat menghubungkan pin buzzer langsung ke Raspberry Pi. Buzzer ini menggunakan sangat sedikit arus. Namun, jika Anda memiliki buzzer yang lebih besar atau hanya ingin bermain aman, maka tambakanlah resistor 470Ω antara pin GPIO dan ujung buzzer. Cobalah program "buzzer.py" berikut ini :

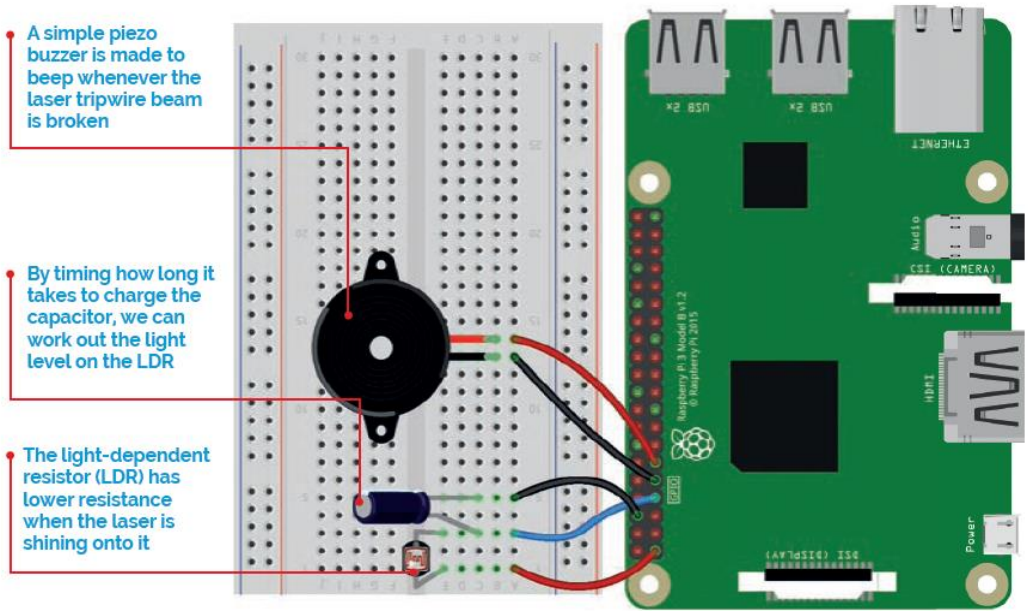
```
import RPi.GPIO as GPIO
import time

buzzer_pin = 18
GPIO.setmode(GPIO.BCM)
GPIO.setup(buzzer_pin, GPIO.OUT)

def buzz(pitch, duration):
        period = 1.0 / pitch
        delay = period / 2
        cycles = int(duration * pitch)

        for i in range(cycles):
                GPIO.output(buzzer_pin, True)
                time.sleep(delay)
                GPIO.output(buzzer_pin, False)
                time.sleep(delay)
        while True:
                pitch_s = raw_input("Enter Pitch (200 to 2000): ")
                pitch = float(pitch_s)
                duration_s = raw_input("Enter Duration (seconds): ")
                duration = float(duration_s)
                buzz(pitch, duration)
```

Ketika Anda menjalankan program, masukkan "*pitch*" dalam satuan Hz, dan masukkan durasi dalam satuan "second".

```
$ sudo python buzzer.py
Enter Pitch (2000 to 10000): 2000
Enter Duration (seconds): 20
```

Mengunakan LDR (Sensor Cahaya)

LDR (juga dikenal sebagai fotosel) adalah tipe resistor elektrik khusus yang resistannya sangat tinggi di saat gelap, namun berkurang saat cahaya bersinar di atasnya. Pasang LDR Anda ke *breadboard*, lalu tambahkan kapasitor 1uF. Penting untuk memastikan polaritas kapasitor yang benar kaki yang lebih panjang (positif) harus berada pada *breadboard* yang sama dengan satu kaki LDR. Sekarang hubungkan kolom ini (dengan kedua komponen) ke GPIO 4. Sambungkan kaki lain dari LDR ke pin 3V3, dan kaki kapasitor lainnya ke pin GND. Untuk membuat alarm, tambahkan buzzer ke papan. Sekali lagi, polaritasnya harus benar: sambungkan kolom bel yang lebih panjang ke GPIO 17, dan kaki yang lebih pendek ke pin GND.



Cobalah beberapa program berikut ini

```
from gpiozero import LightSensor

ldr = LightSensor(4)
while True:
        print(ldr.value)
```

```
from gpiozero import Buzzer

buzzer = Buzzer(17)
buzzer.beep()
```

```
from gpiozero import LightSensor, Buzzer
from time import sleep

ldr = LightSensor(4)
buzzer = Buzzer(17)
```

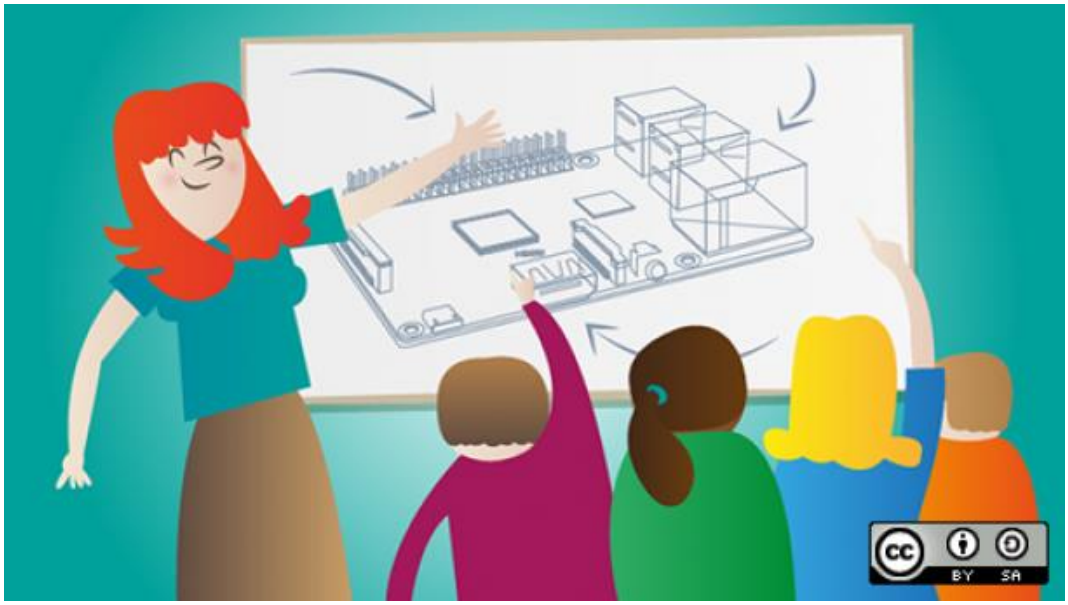
```
while True:
        sleep(0.1)
        if ldr.value < 0.5:
                buzzer.beep(0.5, 0.5, 8)
                sleep(8)
        else:
                buzzer.off()
```

Studi Kasus :

- Buatlah program untuk membuat aplikasi LED dan BUZZER untuk membantu orang menyeberang jalan. Sebelumnya LED HIJAU menyala, sedangkan LED KUNING, dan LED MERAH mati. Jika PUSH BUTTON ditekan, maka LED HIJAU mati, LED KUNING menyala kedip 3 kali, kemudian LED MERAH menyala selama 10 detik bersamaan dengan bunyi BUZZER. Setelah itu kembali normal seperti semula dengan LED HIJAU menyala.

- Buatlah program untuk mengambil data tingkat resistansi dari sensor LDR untuk mempengaruhi nyala lampu LED. Semakin besar resitansinya dari sensor LDR semakin redup nyala lampu LED, semakin kecil resistansinya semakin cerah nyala lampu LED.

7. Permrograman Multi-Threading



Satu kumpulan instruksi yang akan dieksekusi secara independen dinamakan thread. Thread adalah alur kontrol dari suatu proses atau sekumpulan perintah (instruksi) yang dapat dilaksanakan (dieksekusi) secara teratur dengan proses lainnya. Proses melakukan setiap langkah-langkah/intruksi yang berurutan, setiap intruksi untuk mengeksekusi baris kode/listing – listing program. Karena langkah-langkah yang berurutan itu, setiap langkah membutuhkan jumlah waktu tertentu.

Thread sangat berguna untuk membuat proses yang interaktif; misalnya pada permainan (game). Dengan menggunakan sejumlah thread, program tetap dapat menggerakkan sejumlah objek sembari memberikan kesempatan pemakai untuk melakukan tanggapan melalui keyboard. Web browser merupakan contoh lain penggunaan thread. Tanpa thread, Web browser akan menghentikan segala tanggapan terhadap pemakai ketika perangkat lunak tersebut sedang mengambil isi dari suatu URL.

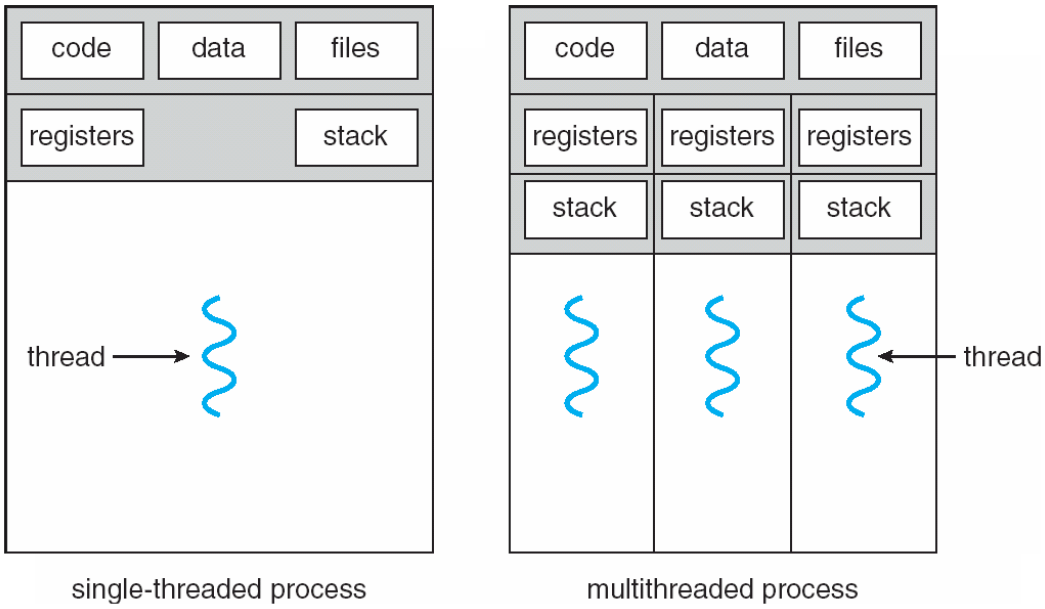
Multi-Threading berasal dari dua kata yakni multi dan thread. Multi bermakna banyak sementara thread bermakna benang atau alur. Multithreading adalah suatu kemampuan yang memungkinkan beberapa kumpulan instruksi atau proses dapat dijalankan secara bersamaan dalam sebuah program. Namun dalam praktik sebenarnya, multithread adalah sebuah metode yang memungkinkan sebuah program menjalankan dua atau lebih operasi dalam satu waktu. Sebenarnya ketika kita membuat sebuah program, secara otomatis kita sudah memiliki satu thread dalam program tersebut.

Single-Threading dan Multi-Threading

Single-Threading adalah sebuah lightweight process (proses sederhana) yang mempunyai thread tunggal yang berfungsi sebagai pengendali/ controller.

Multi-Threading adalah proses dengan thread yang banyak dan mengerjakan lebih dari satu tugas dalam satu waktu.

Menjalankan beberapa thread mirip dengan menjalankan beberapa program yang berbeda secara bersamaan. Beberapa thread dalam sebuah proses berbagi ruang data yang sama dengan thread induk dan karenanya dapat saling berbagi informasi atau berkomunikasi satu sama lain dengan lebih mudah daripada jika prosesnya terpisah.



Keuntungan Multi-Threading

Keuntungan dari sistem yang menerapkan multithreading dapat kita kategorikan menjadi 4 bagian:

1. Responsif
   Aplikasi interaktif menjadi tetap responsif meskipun sebagian dari program sedang diblok atau melakukan operasi lain yang panjang. Umpamanya, sebuah thread dari program *looping* membaca sensor dapat melayani permintaan pengguna misalnya input push button, sementara thread yang lain berusaha membaca sensor.

2. Berbagi sumber daya.
   Beberapa thread yang melakukan proses yang sama akan berbagi sumber daya. Keuntungannya adalah mengizinkan sebuah aplikasi untuk mempunyai beberapa thread yang berbeda dalam lokasi memori yang sama.

3. Ekonomis.
   Pembuatan sebuah proses memerlukan pengalokasian memori dan sumber daya. Alternatifnya adalah dengan menggunakan thread, karena thread membagi memori dan sumber daya yang dimilikinya sehingga lebih ekonomis untuk membuat thread. Secara umum pembuatan dan pengaturan proses akan memakan waktu lebih lama dibandingkan dengan thread.

4. Utilisasi arsitektur multiprosesor.
   Keuntungan dari multithreading dapat meningkat pada arsitektur multiprosesor, dimana setiap thread dapat berjalan secara paralel di atas procesor yang berbeda. Pada arsitektur processor tunggal, CPU menjalankan setiap thread secara bergantian tetapi hal ini berlangsung sangat cepat sehingga menciptakan ilusi paralel, tetapi pada kenyataanya hanya satu thread yang dijalankan CPU pada satu-satuan waktu.

Kerugian Multi-Threading

1. Multiple thread bisa mengganggu satu sama lain saat berbagi hardware resource, misalnya chace memory.

2. Execution time (waktu proses) dari sebuah single-thread tidak dapat diimprove (ditambah), tapi malah bisa diturunkan. Ini terjadi karena penurunan frekuensi yang dibutuhkan ketika terjadi pergantian thread yang berjalan.

3. Harus ada dukungan dari hardware ataupun software untuk melakukan multi-threading.

Multi-Threading pada Raspberry Pi

Processor yang ada dalam Raspberry Pi mendukung eksekusi paralel pada instruksi yang berhubungan dengan *floating-point*. Bukan multi-threading secara hardware untuk semua instruksi CPU, namun secara software pada program yang dibuat. Sistem operasi Raspbian mendukung program multi-threading. Perbedaan utama antara keduanya adalah bahwa multi-threading hardware memungkinkan eksekusi lebih dari satu pipa instruksi secara paralel, sementara multi-threading software terbatas pada eksekusi hanya satu urutan instruksi pada satu waktu, dan ada pinalti waktu setiap kali thread diaktifkan. Pinalti waktu tersebut yang tidak terjadi pada multi-threading hardware.

Terdapat beberapa library untuk algoritma multi-thread pada Python. Thread pada Python digunakan ketika eksekusi dari sebuah program (proses) disertai kondisi menunggu. Thread memungkinkan Python untuk mengeksekusi kode lainnya, sementara kode yang dieksekusi sebelumnya pada kondisi menunggu. Hal ini dapat disimulasikan menggunakan fungsi sleep().

Berikut ini adalah contoh thread yang mencetak angka 1-10 dengan menunggu 1 detik.

```
import threading
import time

def looping():
	for i in range(1, 11):
		time.sleep(1)
		print(i)

thread1 = threading.Thread(target = looping)
thread1.daemon = True
thread1.start()
```

Library yang disediakan Python menggunakan mekanisme penguncian sederhana, yang memungkinkan singkronisasi beberapa thread. Berikut ini contoh penggunaan mekanisme penguncian :

```
import threading
import time

def printTime(threadName, delay):
	while True :
		time.sleep(delay)
		lock.acquire()
		print "%s : %s" % (threadName,time.ctime(time.time()))
		lock.release()

lock = threading.lock()

thread1 = threading.Thread(target = printTime, args=("Thread#1",1))
thread1.daemon = True
thread1.start()

thread2 = threading.Thread(target = printTime, args=("Thread#2",2))
thread2.daemon = True
thread2.start()

try:
	while True:
		pass
except KeyboardInterrrupt:
	print "Selesai."
```

Berhati-hati ketika menggunakan "pass" di dalam endless loop, seperti pada contoh di atas yang merupakan lingkungan multi-thread. Hal ini akan menguras kerja CPU, disarankan mengganti dengan fungsi "time.sleep()". Cobalah menganti blok try-except diatas dengan program berikut :

```
try:
	while True:
		time.sleep(0.1)
except KeyboardInterrrupt:
	print "Selesai."
```

Menggunakan Event

Komunikasi antar thread dapat menggunakan "event". Satu thread dapat memberikan sinyal terhadap sebuah event, sementara thread lainnya menerima kemudian menghapusnya.

```
import threading
import time
```

```
e = threading.Event()

def signal_event():
	while True:
		e.set()
		print "Hello"
		time.sleep(1)

def wait_for_event():
	while True:
		e.wait()
		e.clear()
		print "World"

thread1 = threading.Thread(target = signal_event)
thread1.daemon = True
thread1.start()

thread2 = threading.Thread(target = wait_for_event)
thread2.daemon = True
thread2.start()

try:
	while True:
		time.sleep(0.1)
except KeyboardInterrrupt:
	print "Selesai."
```

Menggunakan Queue

Anda dapat membuat sebuah objek antrian dengan menggunakan "queue". Satu objek antrian dapat menampung banyak item. Berikut ini adalah beberapa method yang dapat digunakan untuk mengatur sebuah queue :

- get() : mengambil dan menghapus sebuah item dari antrian.
- put() : meletakkan sebuah item ke dalam antrian.
- qsize() : mengembalikan jumlah item yang terdapat dalam antrian.
- empty() : mengembalikan True jika antrian kosong, dan sebaliknya.
- full() : mengembalikan True jika antrian penuh, dan sebaliknya.

Berikut ini adalah contoh dimana satu thread mengirim sebuah karakter ke dalam antrian, dan thread yang lainnya membaca dan menampilkannya ke layar console :

```
import threading
import Queue
import time

def putQueue():
	while True:
		workQueue.put('a')
		time.sleep(1)

def getQueue():
	while True:
		if not workQueue.empty();
			ch = workQueue.get()
			print ch

workQueue = Queue.Queue(10)

thread1 = threading.Thread(target = putQueue)
thread1.daemon = True
thread1.start()
```

```
thread2 = threading.Thread(target = getQueue)
thread2.daemon = True
thread2.start()

try:
	while True:
		time.sleep(0.1)
except KeyboardInterrrupt:
	print “Selesai.”
```

Data yang ditukarkan dapat berupa “tupple”, seperti pada contoh berikut ini :

```
import threading
import Queue
import time

def putQueue():
	while True:
		workQueue.put(('a','hello', 12345)
		time.sleep(1)

def getQueue():
	while True:
		if not workQueue.empty();
			ch = workQueue.get()
			print ch[0], ch[1], ch[2]

workQueue = Queue.Queue(10)

thread1 = threading.Thread(target = putQueue)
thread1.daemon = True
thread1.start()

thread2 = threading.Thread(target = getQueue)
thread2.daemon = True
thread2.start()

try:
	while True:
		time.sleep(0.1)
except KeyboardInterrrupt:
	print “Selesai.”
```

Studi Kasus :

- Buatlah program untuk menjalankan beberapa fungsi sebagai berikut secara bersamaan :
    - Animasi nyala pada BARLED, menyalakan led ke 1 sampai 10 begitu seterusnya.
    - Baca sensor LDR secara terus menerus dan tampilkan di terminal.

- Modifikasi program di atas, dengan menambahkan fungsi agar pada saat animasi BARLED berjalan, jika PUSH BUTTON ditekan, maka animasi BARLED berhenti.

8. Pemrograman **Jaringan Dasar**

Pemrograman Socket

Python menyediakan dua tingkatan akses (*layer*) ke jaringan komputer yaitu tingkat bawah (*low level*) dan tingkat atas (*high level*). Pada tingkat bawah, kita dapat meng-akses *socket support* pada ranah sistem operasi, yang mana memungkinkan kita untuk meng-implementasi client dan server untuk *connection-oriented* dan *connectioniess protocol*. Python juga menyediakan library pada tingkat atas, ke protokol jaringan yang spesifik pada tingkatan aplikasi, seperti FTP, HTTP dan lainnya.

Socket adalah titik akhir dari kanal komunikasi dua arah. Socket dapat berkomunikasi di dalam sebuah proses, antar proses pada mesin yang sama, atau antar proses yang berbeda wilayah. Socket dapat di-implementaikan melalui sejumlah tipe kanal yang berbeda: UNlX domain socket, TCP, UDP, dan yang lainnya. Library socket menyediakan class spesifik untuk menangani transport, juga antarmuka yang umum untuk menangani setelahnya.

Socket UDP lebih sederhana dalam bekerja karena *connection-less*. Sebuah server UDP cukup memiliki sebuah socket dan menunggu data tiba, dan sebuah socket client mengirim data pada sebuah sebuah socket tanpa koneksi. Socket UDP digunakan pada komunikasi jaringan dimana kehandalan tidak terlalu penting dengan alasan tertentu.

Untuk membuat sebuah socket dalam Pyhton, kita dapat menggunakan method socket():

```
socket(Family, type[,protocol])
```

Parameter family memiliki pilihan AF__UNlX, AF_lNET atau AF_INET6. Ada beberapa tipe socket, dua yang utama adalah SOCK_STREAM untuk socket TCP dan SOCK__DGRAM untuk socket UDP.

Fungsi socket di atas, mengembalikan sebuah obyek socket yang dapat digunakan untuk memanggil fungsi-fungsi yang dimilikinya, seperti yang akan dijelaskan di bagian berikutnya dari tulisan ini.

Server socket method:

- **s.bind()** - mengingat alamat (hostname, port number) ke sebuah socket. Host dapat diisi sebuah string kosong, yang diartikan sebagai localhost, atau dapat diisi dengan “0,0,0,0” yang diartikan seluruh interface.
- **s.listen()** - men-setup dan memulai TCP listener. Fungsi ini memiliki parameter berupa jumlah client maksimum yang menunggu ketika server tengah menangani satu client, sebagai contoh: s.listen(S)
- **s.accept()** - secara pasif menerima koneksi TCP client, menunggu sampai koneksitiba.

Client socket method:

- **s.connect()** - meng-inisiasi koneksi TCP server, sebagai contoh: s.connect(“localhost',12345), untuk menghubungi localhost dengan port 12345; atau s.connect(('192.168.080”, 12345)) untuk menghubungi server dengan ip address 192.168.0.80 port 12345.

Bisa juga menggunakan fungsi socket.gethostbyname() untuk me-resolve ip address dari sebuah nama domain, sebagai contoh:

```
s = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
remote_ip = socket.gethostbyname()
s.connect(remote_ip, 86)
```

General socket method:

- **s.recv()** - menerima TCP message, sebagai contoh: s.recv(1024) - akan menerima data, maksimum 1024 byte.
- **s.send()** - mengirim TCP message, contoh: s.send(“Hello World').
- **s.recvfrom()** - menerima UDP message
- **s.sendto()** - mengirim UDP message
- **s.close()** - menutup soket
- **socket.gethostname()** - mengembalikan hostname.

Untuk menulis server internet, kita menggunakan fungsi soket yang tersedia dalam modul socket. Kemudian, obyek socket digunakan untuk memanggil fungsi lain untuk men-setup sebuah socket server. Sekarang, memanggil fungsi bind(hostname, port) untuk menentukan port pada sebuah host

Contoh Program UDP Server :

```
import socket

UDPSock = socket.socket(socket.AF_INET,socket.SOCK_DGRAM)

# Listen on port 5556 (to all IP addresses on this system)
listen_addr : ("",5556)
UDPSock.bind(listen_addr)

while True:
       data,addr : UDPSock.recvfrom(1024)
       print data.strip(),addr
       UDPSock.sendto("OK\n", addr)
```

Contoh Program UDP Client :

```
import socket

UDPSock = socket.socket(socket.AF_INET,socket.SOCK_DGRAM)
data = "Thank You\n"
addr = ("localhost",5556)
UDPSock.sendto(data,addr)
```

Contoh Program TCP Server :

```
import socket

BUFFER_SIZE = 20 # Normally 1924, but we want fast response

print 'TCP Server ...'
s = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
s.bind(("127.0.0.1",5005))
s.listen(1) % Listen 1 client
conn, addr = s.accept()
print 'Connection address:', addr
while True:
       data = conn.recv(BUFFER_SIZE)
       if not data:
              break
       print "received data:", data
              conn.send(data) % echo
conn.close()
```

Contoh Program TCP Client :

```
import socket
import threading
import time

TCP_IP = '127.0.0.1'
TCP_PORT = 5005
BUFFER_SIZE = 1024
MESSAGE = "Hello, World!"

def sockReceive():
       s : socket.socket(socket.AF"INET, socket.SOCK_STREAM)
       s.connect((TCP_IP, TCP_PORT))
       lock.acquire()
       print "Connect to Server"
       lock.release()
       while True:
              s.send(MESSAGE)
              data = s.recv(BUFFER_SIZE)
              lock.acquire()
```

```
                print "received data:", data
                lock.release()
                s.close()
                def printConsole():
        while True:
                lock.acquire()
                print MESSAGE
                lock.release()
                time.sleep(2)

lock = threading.Lock()
thread1 : threading.Thread(target = sockReceive)
thread1.daemon = True
thread1.start()

thread2 = threading.Thread(target = printConsole)
thread2.daemon = True
thread2.start()
while True:

time.sleep(0.1)
```

Mensetting IP Static pada Jaringan Lokal

Jika Anda ingin agar Raspberry Pi terhubung dalam jaringan lokal, anda dapat langsung terhubung dengan akses point/ wireless router dengan mengunakan DHCP, namun Anda akan mendapatkan alamat IP secara acak. Anda dapat mengkonfigurasi alamat IP yang ada pada Raspberry Anda secara manual. Yang harus Anda lakukan adalah melakukan beberapa konfigurasi alamat IP static. Berikut ini adalah beberapa konfigurasi yang harus dilakukan agar Anda dapat mengkonfigurasi alamat IP secara manual dengan menggunakan IP static.

Masuk ke terminal dan login dengan menggunakan username dan password standar

```
username : pi
password : raspberry
```

Perintah berikut akan melakukan pengecekan alamat IP.

```
$ ifconfig
```

Kita perlu memastikan antarmuka jaringan yang saat ini tersedia :

```
$ cat /etc/network/interfaces
```

Jika ada keterangan "`iface eth0 inet dhcp`", artinya saat ini alamat IP didapatkan secara DHCP oleh router (biasanya IP adressnya berubah-ubah). Hal ini yang ingin kita rubah menjadi static. Lakukan setting dengan membuka setting network interface nya dengan cara

```
$ nano /etc/network/interfaces
```

Ubah setting network interface seperti pada berikut ini :

```
iface eth0 inet static
address 192.168.1.101
network 192.168.1.0
```

```
netmask 255.255.255.0
broadcast 192.168.1.255
gateway 192.168.1.7
```

Lalu simpan setting network interface dengan cara tekan Ctrl+X lalu tekan Y. Kemudian restart service networking :

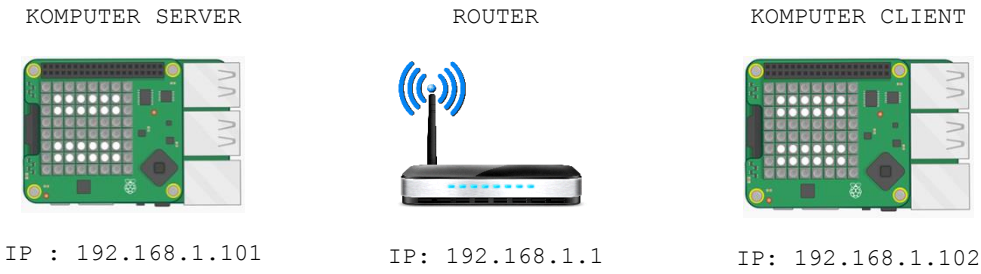
```
$ /etc/init.d/networking restart
```

Jika diperlukan bisa lakukan Restart si Raspi

```
$ reboot
```

Studi Kasus :

Project kali ini Anda memerlukan 2 buah Raspberry Pi dan sebuah akses point / wireless router. Kemudian lakukan setting alamat IP static seperti pada gambar dibawah ini :



Dengan menggunakan protokol TCP Client Server :

- Buatlah program untuk menyalakan LED yang ada pada SERVER dengan PUSH BUTTON yang ada pada CLIENT.

- Buatlah program untuk mengirimkan data sensor dari CLIENT ke SERVER kemudian oleh servier disimpan ke dalam sebuah file data log.

Referensi

- Richard Blum, *Python Programming for Raspberry Pi in 24 Hours*, Sams Teach Yourself. (2014)
- Simon Monk, *Raspberry Pi Cookbook : Software and Hardware Problems and Solutions*, O'Reilly Media (2014)
- Andi Dinata, *Physical Computing dengan Raspberry*, Elex Media Komputindo, Jakarta. (2016)
- Christianto Tjahyadi, *Internet of Things & Applications with Raspberry Pi*, Next System, Bandung. (2016)
- https://www.raspberrypi.org/
- https://id.wikipedia.org/wiki/Raspberry_Pi
- http://ambercat.rahmanda.net/code/2015/02/21/berkenalan-dengan-python.html
- http://lifehacker.com/the-best-operating-systems-for-your-raspberry-pi-projec-1774669829
- https://sangwidy.wordpress.com/web-design/oop-2/7-multi-threading/