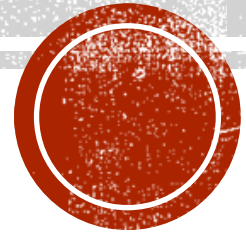
LEARNING PROGRESS REVIEW 3

Dinda Adilfi Wirahmi

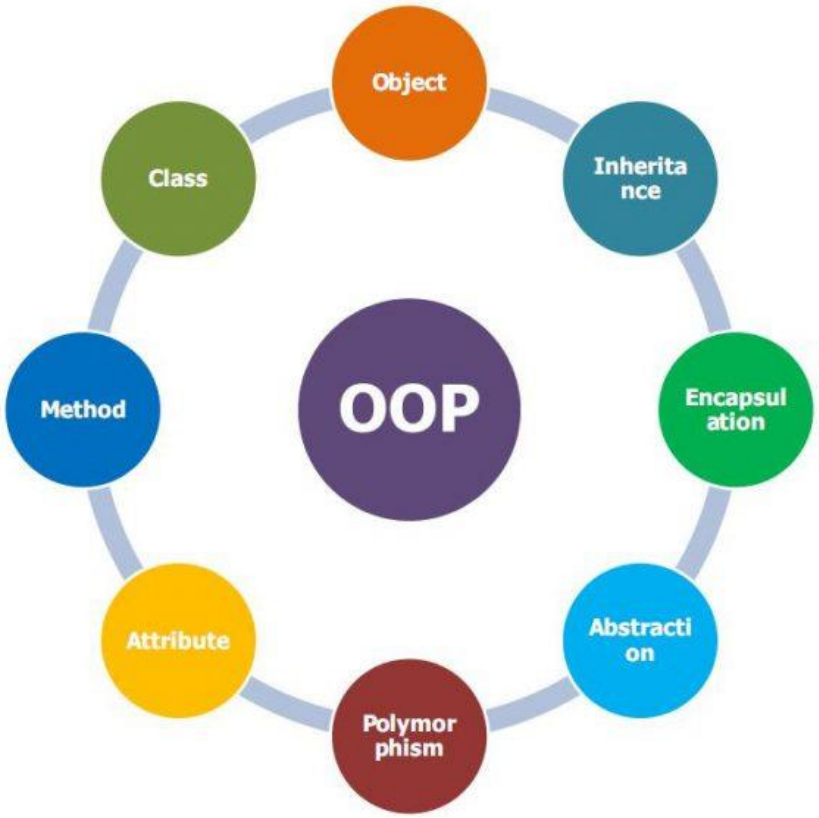
DigitalSkola

PENGENALAN OOP (*OBJECT-ORIENTED PROGRAMMING*)



DigitalSkola

APA ITU OOP?



Dalam OOP terdapat konsep yang disebut '*class*' yang memungkinkan pemrogram menyusun kode-kode perangkat lunak. Karena penggunaan *'class'* dan *'object'*, pemrograman menjadi mudah dipahami dan dikodekan.

DigitalSkola

KENAPA BUTUH OOP?

Karena kemampuan *encapsulation* yang dapat mengkaitkan *variables* dan *functions* bersama dalam satu kesatuan unit yaitu *object*. Jadi user tidak perlu lagi menggunakan nama variabel atau function dengan awalan *prefix* tertentu.

DigitalSkola

"class Kendaraan" merupakan *blueprint*

```python
class Kendaraan:
    bahan_bakar = "bensin"

    def __init__(self, nama, warna, jumlah_roda):
        self.nama = nama
        self.warna = warna
        self.jumlah_roda = jumlah_roda

    def maju(self):
        print("Kendaraan", self.nama, "maju")

    def mundur(self):
        print("Kendaraan", self.nama, "mundur")

    @classmethod
    def isi_bahan_bakar(cls):
        print(cls.bahan_bakar, 'sedang diisi')
```

"bahan_bakar" merupakan class attributes

"self.nama" merupakan instance attributes

Instance Method harus memiliki parameter "self" dan bisa mengakses instance variable & class variable dari class dimana method tsb di definisikan

@classmethod = terikat dengan Class pemanggil methodnya, tidak bisa akses instance variables

DigitalSkola

Inheritance. Class Mobil merupakan inheritance dari Class Kendaraan

```
class Mobil(Kendaraan):
    bahan_bakar = "Solar"
    def belok(self, belok=None):
        if belok == None:
            print("tidak bisa belok karena tidak ada arah")
        else:
            print(f"{self.nama} belok {belok}")
class Pesawat(Kendaraan):
    bahan_bakar = "Avtur"

mobil = Kendaraan('Brv', 'Putih', 4,)
pesawat = Kendaraan('Garuda', 'Silver', 3)
```

Overloading : sebuah class memiliki nama method yang sama dengan parameter beda

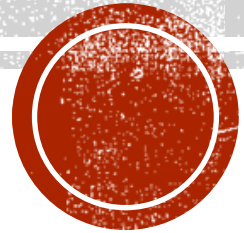
Overriding. Memungkinkan subclass memiliki behaviour berbeda dari parent class

“mobil” dan “pesawat”Object & class dari Kendaraan

Kendaraan() = Instantiation

DigitalSkola

NETWORKING



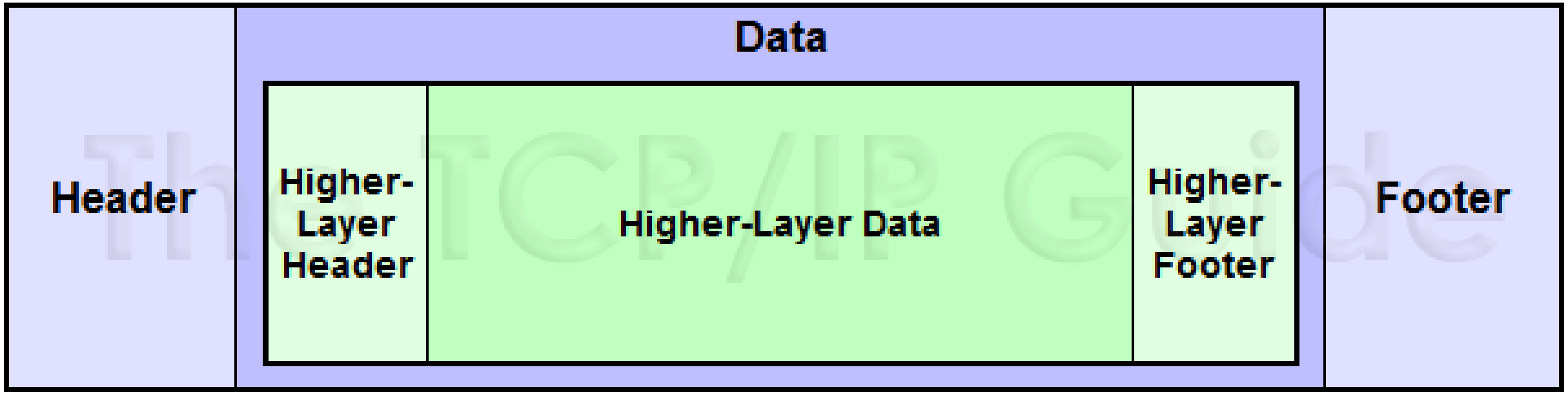
DigitalSkola

MENGAPA NETWORKING?

- *Transfer data* : *data engineer* memahami untuk dapat mentransfer data dalam volume besar
- *Multiprocessing Data* : *Data engineering* diharapkan untuk mengolah data dengan cepat (*velocity*). User dapat mengolah data dengan *multiprocessing* (menggunakan beberapa komputer).
- *Data engineer* diharapkan mempunyai pengetahuan mengenai *cloud* yang merupakan tempat *processing unit* untuk melakukan pengolahan data

DigitalSkola

FORMAT *MESSAGE NETWORK*



DigitalSkola

- *Header*: Informasi yang ditempatkan sebelum data aktual. *Header* biasanya berisi sejumlah kecil *byte* informasi untuk mengkomunikasikan fakta-fakta penting tentang data yang ada dalam pesan, diinterpretasikan dan digunakan.
- Data: Data aktual yang akan ditransmisikan, sering disebut muatan pesan. Sebagian besar pesan berisi beberapa data dari satu bentuk atau lainnya, tetapi beberapa sebenarnya tidak berisi.
- *Footer*: Informasi yang ditempatkan setelah data.

NETWORK STANDADRD

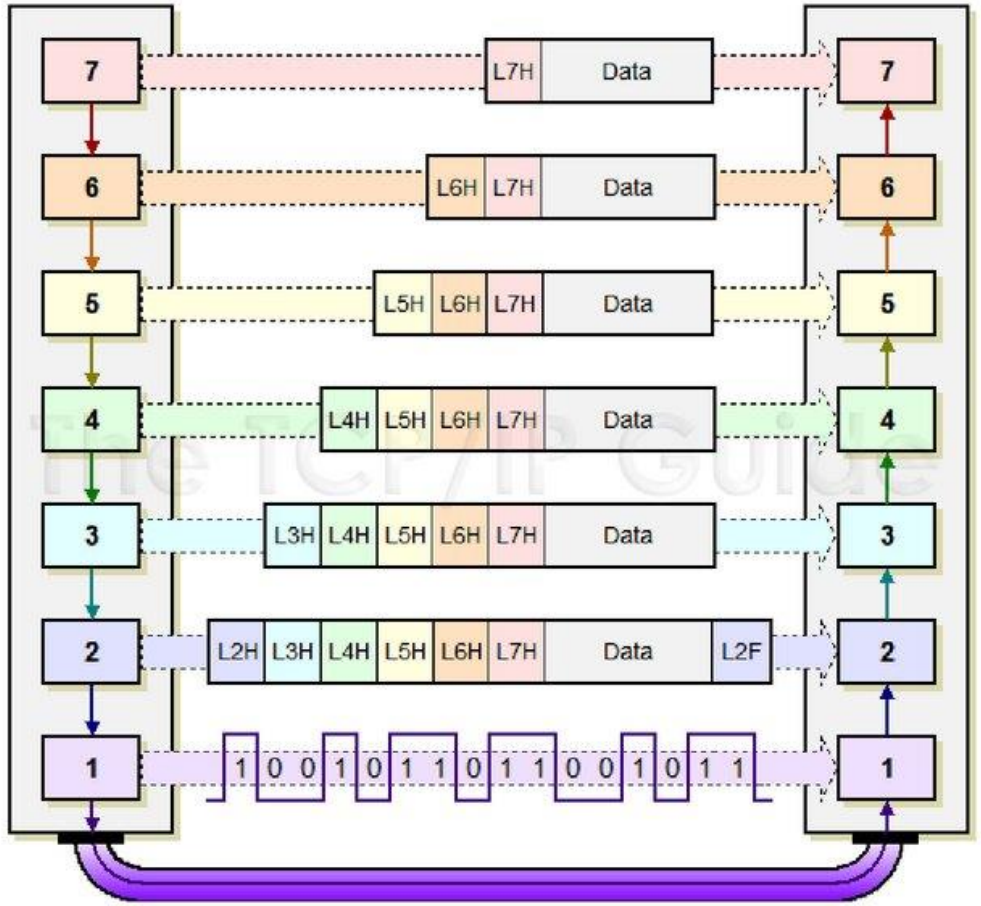
	OSI Layer	TCP/IP	Datagrams are called
Software	**Layer 7** Application	HTTP, SMTP, IMAP, SNMP, POP3, FTP	Upper Layer Data
	Layer 6 Presentation	ASCII Characters, MPEG, SSL, TSL, Compression (Encryption & Decryption)	
	Layer 5 Session	NetBIOS, SAP, Handshaking connection	
	Layer 4 Transport	TCP, UDP	Segment
	Layer 3 Network	IPv4, IPv6, ICMP, IPSec, MPLS, ARP	Packet
Hardware	**Layer 2** Data Link	Ethernet, 802.1x, PPP, ATM, Fiber Channel, MPLS, FDDI, MAC Addresses	Frame
	Layer 1 Physical	Cables, Connectors, Hubs (DLS, RS232, 10BaseT, 100BaseTX, ISDN, T1)	Bits

DigitalSkola

Communication between Layers

Moving from the top - messages get larger and larger. A message is passed down, and the lower layer adds a header to it. This is "encapsulation".

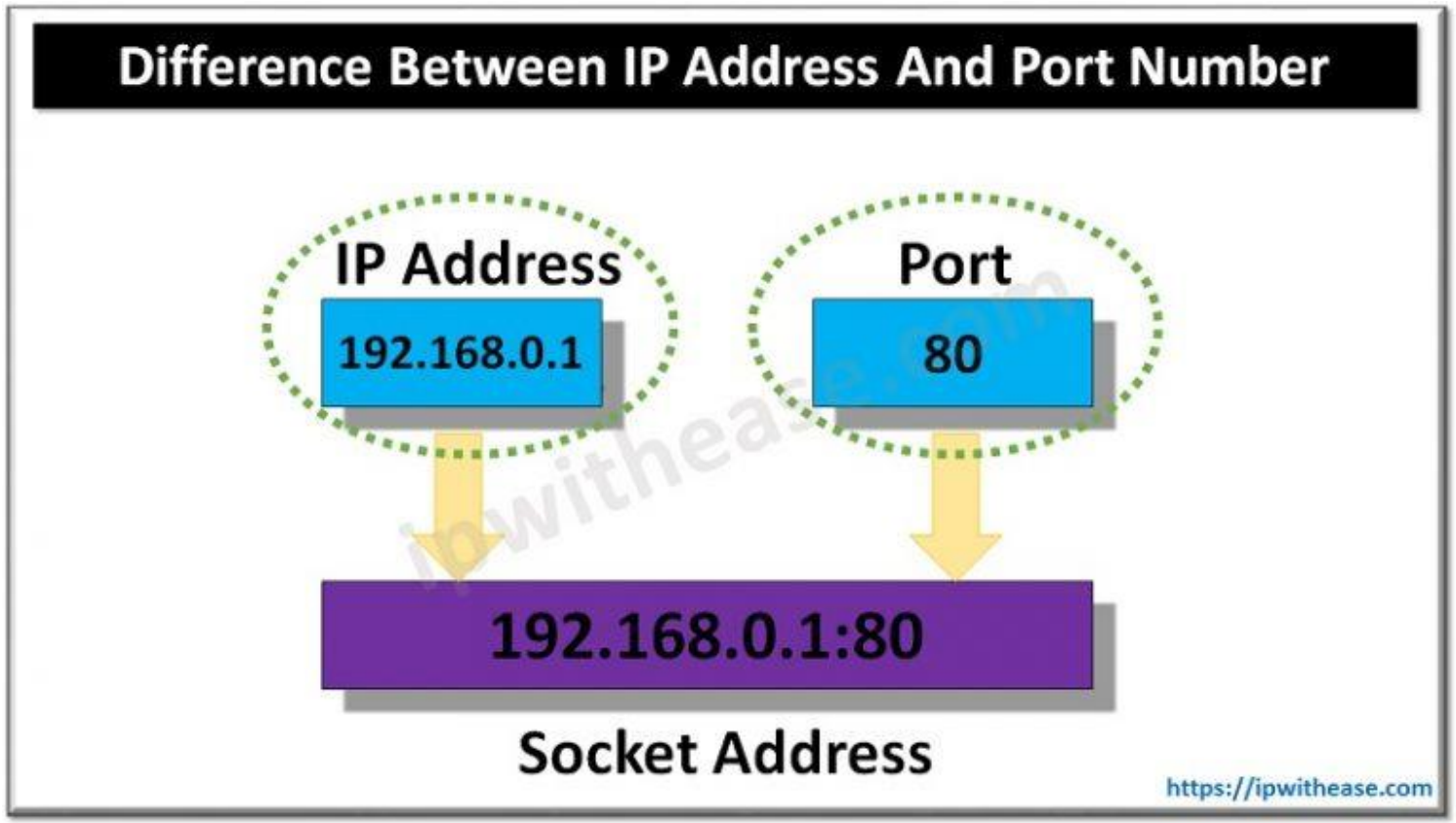
Moving from the bottom - messages get smaller and smaller. Each upper layer receives the data message from the layer below, and then strips off it's own header and passes the data up. This is "decapsulation".



Setiap data yang melewati setiap layer, akan dienkapssulasi dengan HEADER dan FOOTER dari data.

DigitalSkola

IP ADDRESS & PORT



IP ADDRESS

- Sebuah *IP Address* terdiri dari empat angka, masing-masing berisi 1-3 digit, dengan satu titik (.) dimana titik tersebut yang memisahkan setiap nomor atau serangkaian angka. Masing-masing dari empat nomor dapat berkisar dari 0 sampai 255.
- Alamat IP dapat berupa statis atau dinamis.

DigitalSkola

PORT

Well-Known Port Numbers

Service, Protocol, or Application	Port Number	TCP or UDP
FTP (File Transfer Protocol)	20, 21	TCP
SSH (Secure Shell Protocol)	22	TCP
Telnet	23	TCP
SMTP (Simple Mail Transfer Protocol)	25	TCP
DNS (Domain Name System	53	UDP
TFTP	69	UDP
HTTP	80	TCP
POP3	110	TCP
IMAP4	143	TCP
HTTPS	443	TCP

- Nomor *port* atau *Port number* pada jaringan komputer merupakan angka *biner* sepanjang 16 bit yang berfungsi sebagai nomor untuk layanan yang digunakan didalam jaringan komputer.

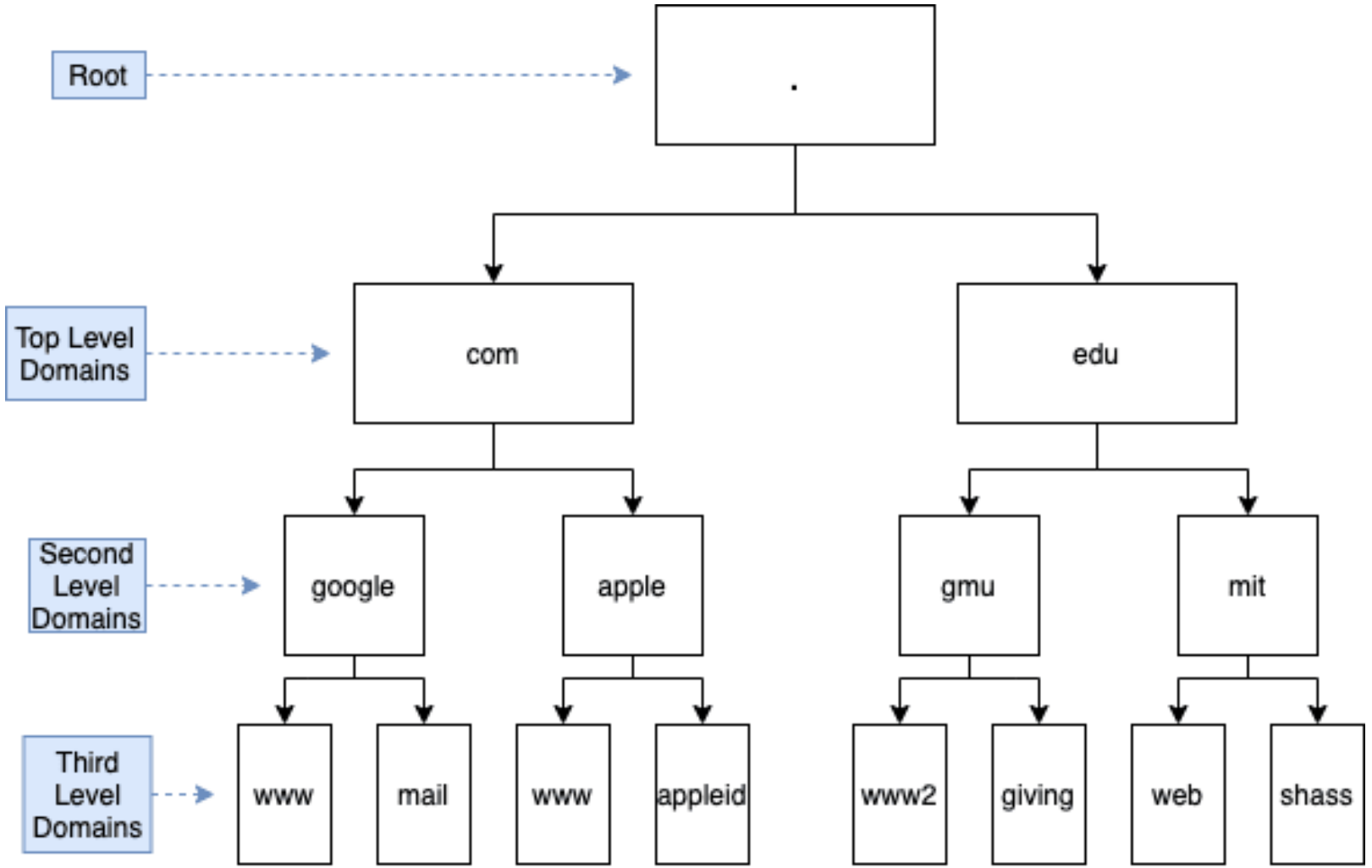
DigitalSkola

DNS

- *Domain Name Server* atau DNS adalah sebuah sistem yang menghubungkan *Uniform Resource Locator (URL)* dengan *Internet Protocol Address (IP Address).*
- Normalnya, untuk mengakses *internet*, user perlu mengetikkan *IP Address* sebuah *website*. DNS adalah sistem yang meringkas pekerjaan ini. Misalkan, *user* ingin mengakses *Google*. Tidak perlu menulis 172.217.0.142 ke dalam *address bar*, cukup tinggal memasukkan alamat ***Google.com***.

DigitalSkola

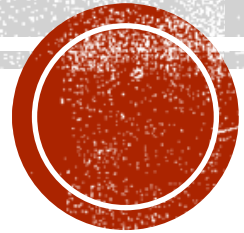
DNS HIERARCHY



Graphic created by Blake Khan (blakekhan.com)

DigitalSkola

LINUX



DigitalSkola

KENAPA LINUX?

- *Powerful* untuk melakukan pengolahan data yang besar
- Mudah untuk dihandle karena 1 *script* bisa *handle* banyak *server* secara otomatis
- Banyak *tools data engineer* yang *running well* di *linux* (*Hadoop, spark, airflow*)
- 90% *public cloud* berjalan di atas *linux*

DigitalSkola

LINUX BASIC COMMAND

- ls untuk melihat isi direktori
- mkdir untuk membuat folder

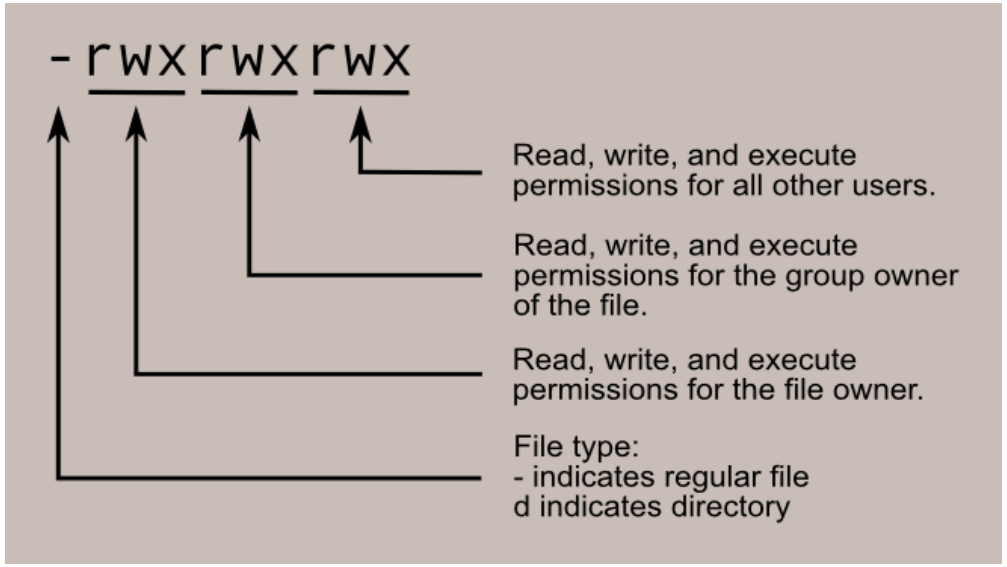
  contoh : mkdir *Finance* (membuat folder bernama Finance)
- cd (change directory) untuk pindah ke direktori tertentu
- pwd untuk melihat *current position* direktori
- touch untuk membuat *file* dalam folder
- vim dan nano untuk melakukan *writing* dan *editing file*
- cat (membaca seluruh file), less(membaca dalm *mode editor*), head(membaca 100 row ke atas), tail(membaca 100 row ke bawah)
- cp (*copy file*), mv(*move file*), rm(*remove file*), rmdir(*remove folder*)
- grep untuk melakukan pencarian di dalam *file*

  contoh : cat nama_file | grep yang_mau_dicari

FILE MODE AND PERMISSION

- Keunggulan *linux* lainnya adalah dapat memberikan akses tertentu ke *user* tertentu

Misal : Ada *user* tertentu yang tidak boleh mengakses *file finance* maka dapat menggunakan *file mode and permission*



DigitalSkola

USER

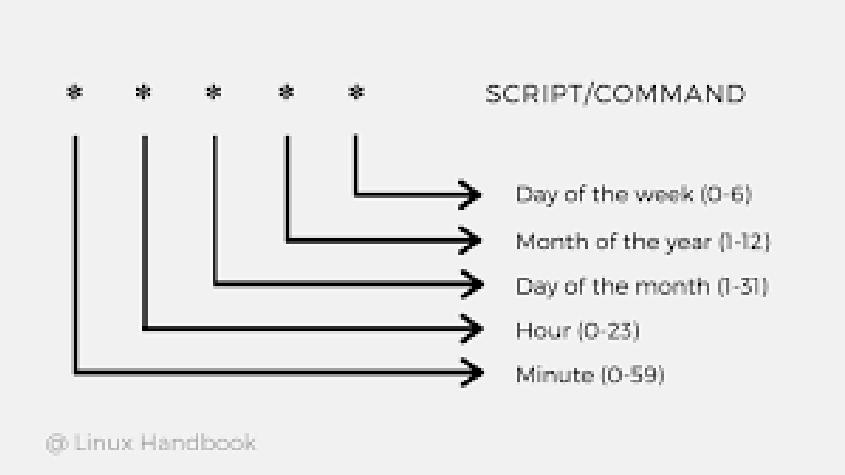
- *User* biasa
- *User root (super user)* memiliki *privilege* yang lebih tinggi daripada *user* biasa salah satunya yaitu bisa mengubah konfigurasi *user* lain

sudo su merupakan perintah untuk login sebagai *root*

sudo untuk menjalankan perintah sebagai *root*

DigitalSkola

CRONTAB



- Memungkinkan komputer untuk menjalankan perintah sesuai dengan jadwal yang diinginkan (*scheduler*)
- Misal :

00 02 * * * sql finance.sql

Data *finance* akan dikirim jam 2 pagi setiap harinya

DigitalSkola

thank you!

DigitalSkola