



PACKAGE

Package

- **Package** adalah teknik pengemasan **modul** dalam pemrograman Python.
- Dengan **package**, programmer dapat mengelompokkan atau **mengorganisasikan beberapa modul** yang telah dibuat untuk membangun suatu proyek pemrograman.
- **Package** merupakan struktur bertingkat (Hirarki) direktori yang mendefinisikan suatu aplikasi tunggal Python dan terdiri dari beberapa modul atau *sub-package*.

Tujuan Membuat Package

- Mengelompokkan modul-modul yang saling terkait.
- Menghindari konflik penamaan fungsi atau variabel dari suatu modul.
- Membuat struktur hierarki pada namespace (*Flat namespace*).

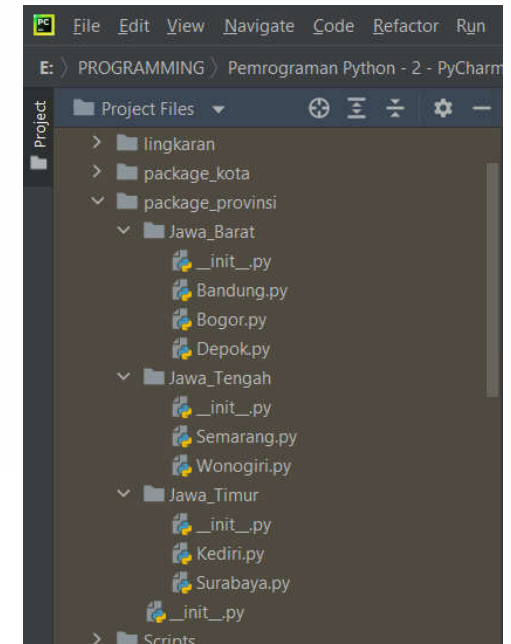
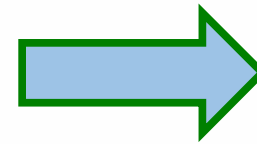
Struktur Direktori Package

```
namaPackage/  
  __init__.py  
  direktori1/  
    __init__.py  
    modul1.py  
    modul2.py  
  direktori2/  
    __init__.py  
    modul1.py  
    modul2.py  
  ...../  
    __init__.py  
  ...  
  direktoriN/  
    __init__.py  
    modul1.py  
    modul2.py
```

Pada setiap direktori **harus** disertakan **file kosong** (*empty* atau tidak ada programnya) **`__init__.py`** yang berfungsi **sebagai file initial**, sehingga seluruh file yang ada di dalamnya (termasuk sub-direktori) dikenali oleh program. Jika tidak, maka saat program dijalankan akan menampilkan **pesan error**.

Contoh Struktur Direktori “package_provinsi”

```
package_provinsi/           # Direktori_utama
  __init__.py
  Jawa_Barat/               # Sub-directori-1
    __init__.py
    Bandung.py
    Bogor.py
    Depok.py
  Jawa_Tengah/              # Sub-directori-2
    __init__.py
    Semarang.py
    Wonogiri.py
  Jawa_Timur/               # Sub-directori-3
    __init__.py
    Kediri.py
    Surabaya.py
```



E:\PROGRAMMING\Pemrograman Python - 2 - PyCharm\pycharm_workspace\package_provinsi

Name	Size	Type
__pycache__		File folder
Jawa_Barat		File folder
Jawa_Tengah		File folder
Jawa_Timur		File folder
__init__.py	0 KB	JetBrains PyCharm Community Edition

Modul-Modul Buatan Programmer

Simpan di sub-direktori provinsi "**Jawa_Tengah**"

Simpan di sub-direktori provinsi "**Jawa_Barat**"

```
# Modul Bandung.py
"\nWelcome to Bandung"      # Document string
def tampil():
    print("Selamat datang di Kota Bandung")
def run():
    print("Kota Bandung berada di Provinsi Jawa Barat")
```

```
# Modul Bogor.py
"\nWelcome to Bogor "      # Document string
def tampil():
    print("Selamat datang di Kota Bogor ")
def run():
    print("Kota Bogor berada di Provinsi Jawa Barat")
```

```
# Modul Depok.py
"\nWelcome to Depok "      # Document string
def tampil():
    print("Selamat datang di Kota Depok ")
def run():
    print("Kota Depok berada di Provinsi Jawa Barat")
```

```
# Modul Semarang.py
"\nWelcome to Semarang "      # Document string
def tampil():
    print("Selamat datang di Kota Semarang ")
def run():
    print("Kota Semarang berada di Provinsi Jawa Tengah")
```

```
# Modul Wonogiri.py
"\nWelcome to Wonogiri "      # Document string
def tampil():
    print("Selamat datang di Kota Wonogiri ")
def run():
    print("Kota Wonogiri berada di Provinsi Jawa Tengah")
```

Simpan di sub-direktori provinsi "**Jawa_Timur**"

```
# Modul Surabaya.py
"\nWelcome to Surabaya "      # Document string
def tampil():
    print("Selamat datang di Kota Surabaya ")
def run():
    print("Kota Surabaya berada di Provinsi Jawa Timur")
```

```
# Modul Kediri.py
"\nWelcome to Kediri "      # Document string
def tampil():
    print("Selamat datang di Kota Kediri ")
def run():
    print("Kota Kediri berada di Provinsi Jawa Timur")
```

Program “CallPackageProvinsi.py”

```
# Program Aplikasi Pemanggilan Package
from package_provinsi.Jawa_Barat import Bandung,Bogor,Depok
from package_provinsi.Jawa_Tengah import Semarang,Wonogiri
from package_provinsi.Jawa_Timur import Surabaya,Kediri

print(Bandung.__doc__)
Bandung.tampil()
Bandung.run()

print(Wonogiri.__doc__)
Wonogiri.tampil()
Wonogiri.run()

print(Surabaya.__doc__)
Surabaya.tampil()
Surabaya.run()
```

Simpan program ini
di luar direktori
“package_provinsi”

```
# Output
Welcome to Bandung
Selamat datang di Kota Bandung
Kota Bandung berada di Provinsi Jawa Barat

Welcome to Wonogiri
Selamat datang di Kota Wonogiri
Kota Wonogiri berada di Provinsi Jawa Tengah

Welcome to Surabaya
Selamat datang di Kota Surabaya
Kota Surabaya berada di Provinsi Jawa Timur
```

Atribut `__doc__`

- Atribut `__doc__` pada setiap modul merupakan catatan (berupa string) yang terdapat di dalam sebuah modul dan pada umumnya merupakan suatu keterangan atau pernyataan terkait sebuah modul.
- Untuk mengakses atribut tersebut digunakan perintah `nama_modul.__doc__`

JENIS PACKAGE/LIBRARY



1. Built-in Package/Library

- Dalam pembuatan aplikasi software dengan pemrograman Python, pada interpreter python telah disediakan (*Built-in*) berbagai macam package/library yang **dapat langsung digunakan** dengan memanggilnya menggunakan perintah “**import**.”
- **Dokumentasi** yang memberikan detail informasi untuk setiap package/library *built-in* Python atau sering disebut dengan *Python standard library*, dapat dibaca pada halaman website resmi Python, yaitu:

<https://docs.python.org/3.10/library/>

2. External Package/Library

- Untuk pembuatan aplikasi software dengan tujuan khusus dan lebih komplek, programmer Python diizinkan untuk menambahkan (install) package/library **eksternal** yang dibuat oleh pihak ketiga (*third-party*), yang dibagikan secara open (*Open source*), sehingga package/library eksternal tersebut dikenali oleh *interpreter Python* dan dapat digunakan.
- Package/library Python **eksternal** dapat di-download dari repository pembuatnya.
- Proses instalasi package/library Python **eksternal** dapat dilakukan **secara offline** dengan menggunakan file installer (.exe, .msi, .whl, .tar.gz, .rar, .zip) dan juga instalasi **secara online** menggunakan **PIP** melalui *Command prompt*.

Apa itu PIP?

PIP



Apa itu **PIP** ?

- **PIP** adalah *package manager* yang digunakan untuk melakukan instalasi *package/library* eksternal yang dibutuhkan oleh programmer dalam membangun suatu proyek software aplikasi Python.
- **PIP** singkatan dari *Preferred Installer Program* atau *Package Installer for Python*
- Mulai Python versi 3.4, PIP secara *default* telah diintegrasikan pada installer interpreter Python, sehingga dapat secara langsung digunakan tanpa perlu melakukan langkah instalasi seperti halnya jika kita menggunakan Python sebelum versi 3.4.

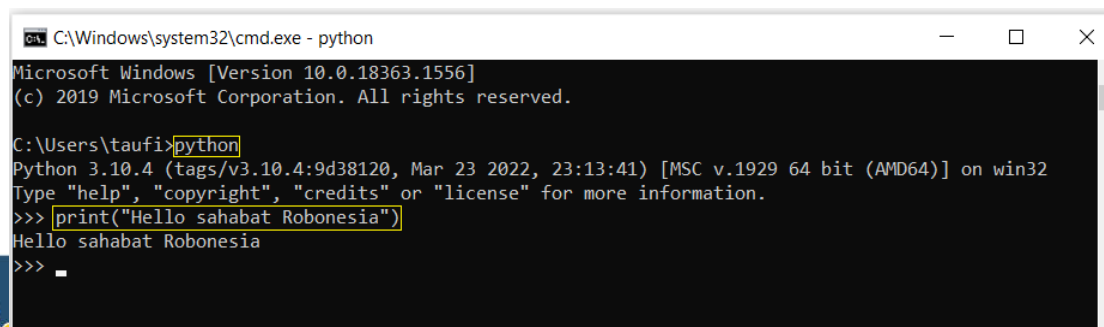
Instalasi Package/Library Eksternal dengan PIP



Instalasi Package/Library Eksternal dengan **PIP** (1)

Sebelum Instalasi, **pastikan**:

1. Komputer **terkoneksi** dengan jaringan **internet**.
2. Python bisa dijalankan via *command-prompt*:
 - a. Buka *command-prompt* (Window + R) >> ketik “cmd” >> tekan Ok.
 - b. Ketik perintah “**python**” kemudian tekan Enter.
 - c. Tuliskan program sederhana dan run, jika program dapat berjalan, berarti Python dapat dijalankan via *command-prompt*.



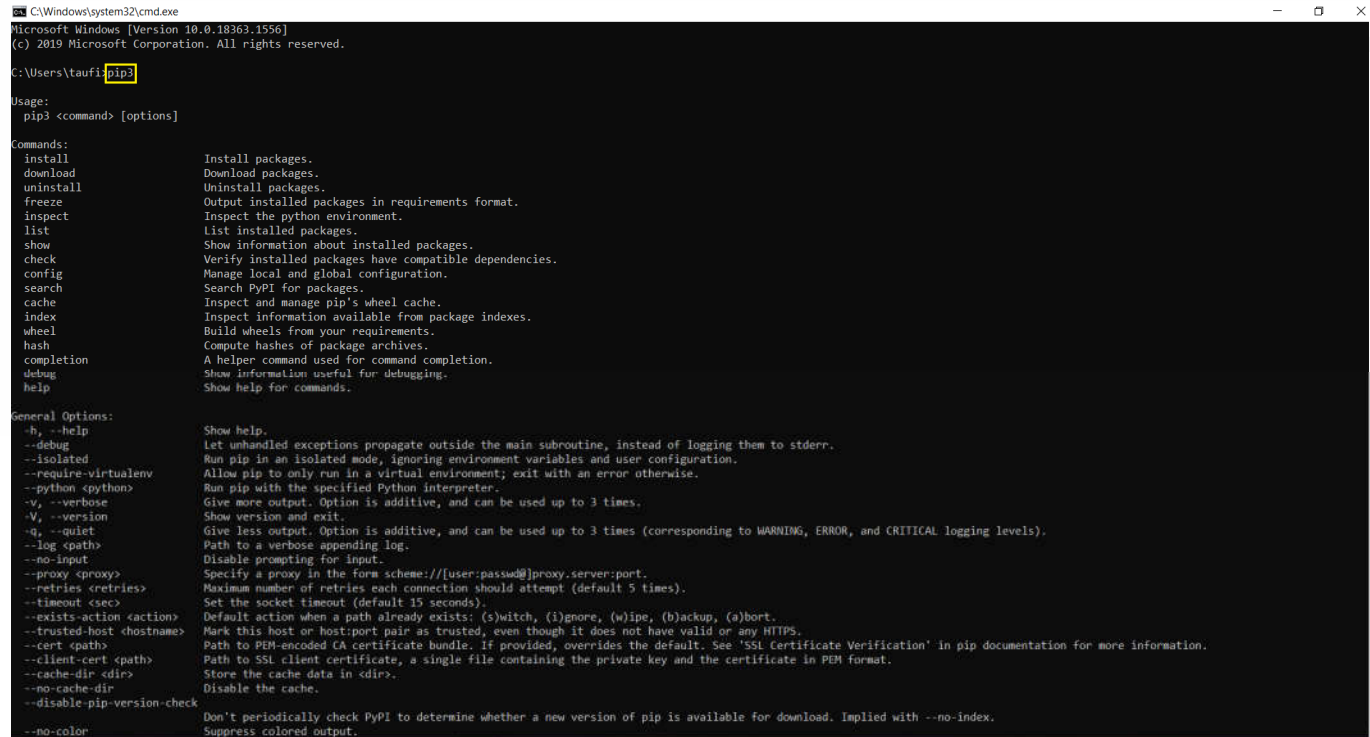
```
C:\Windows\system32\cmd.exe - python
Microsoft Windows [Version 10.0.18363.1556]
(c) 2019 Microsoft Corporation. All rights reserved.

C:\Users\taufi>python
Python 3.10.4 (tags/v3.10.4:9d38120, Mar 23 2022, 23:13:41) [MSC v.1929 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license" for more information.
>>> print("Hello sahabat Robonesia")
Hello sahabat Robonesia
>>> _
```

Instalasi Package/Library Eksternal dengan **PIP** (2)

Panggil PIP:

1. Buka command-prompt (Window + R) >> ketik "cmd" >> tekan Ok.
2. Ketik "**pip**" atau "**pip3**" untuk Python versi 3.4 dan setelahnya.
3. Jika tertampil seperti gambar di samping, maka artinya PIP telah terinstal pada Python.



```
C:\Windows\system32\cmd.exe
Microsoft Windows [Version 10.0.18363.1556]
(c) 2019 Microsoft Corporation. All rights reserved.

C:\Users\taufi>pip

Usage:
  pip3 <command> [options]

Commands:
  install             Install packages.
  download            Download packages.
  uninstall           Uninstall packages.
  freeze              Output installed packages in requirements format.
  inspect             Inspect the python environment.
  list                List installed packages.
  show                Show information about installed packages.
  check               Verify installed packages have compatible dependencies.
  config              Manage local and global configuration.
  search              Search PyPI for packages.
  cache               Inspect and manage pip's wheel cache.
  index               Inspect information available from package indexes.
  wheel               Build wheels from your requirements.
  hash                Compute hashes of package archives.
  completion           A helper command used for command completion.
  debug               Show information useful for debugging.
  help                Show help for commands.

General Options:
  -h, --help           Show help.
  --debug              Let unhandled exceptions propagate outside the main subroutine, instead of logging them to stderr.
  --isolated            Run pip in an isolated mode, ignoring environment variables and user configuration.
  --require-virtualenv Allow pip to only run in a virtual environment; exit with an error otherwise.
  --python <python>    Run pip with the specified Python interpreter.
  -v, --verbose         Give more output. Option is additive, and can be used up to 3 times.
  -V, --version         Show version and exit.
  -q, --quiet           Give less output. Option is additive, and can be used up to 3 times (corresponding to WARNING, ERROR, and CRITICAL logging levels).
  --log <path>         Path to a verbose appending log.
  --no-input            Disable prompting for input.
  --proxy <proxy>       Specify a proxy in the form scheme://[user:passwd@]proxy.server:port.
  --retries <retries>   Maximum number of retries each connection should attempt (default 5 times).
  --timeout <sec>      Set the socket timeout (default 15 seconds).
  --exists-action <action> Default action when a path already exists: (s)witch, (i)gnore, (w)ipe, (b)ackup, (a)bort.
  --trusted-host <hostname> Mark this host or host:port pair as trusted, even though it does not have valid or any HTTPS.
  --cert <path>         Path to PEM-encoded CA certificate bundle. If provided, overrides the default. See 'SSL Certificate Verification' in pip documentation for more information.
  --client-cert <path>  Path to SSL client certificate, a single file containing the private key and the certificate in PEM format.
  --cache-dir <dir>    Store the cache data in <dir>.
  --no-cache-dir        Disable the cache.
  --disable-pip-version-check Don't periodically check PyPI to determine whether a new version of pip is available for download. Implied with --no-index.
  --no-color            Suppress colored output.
```

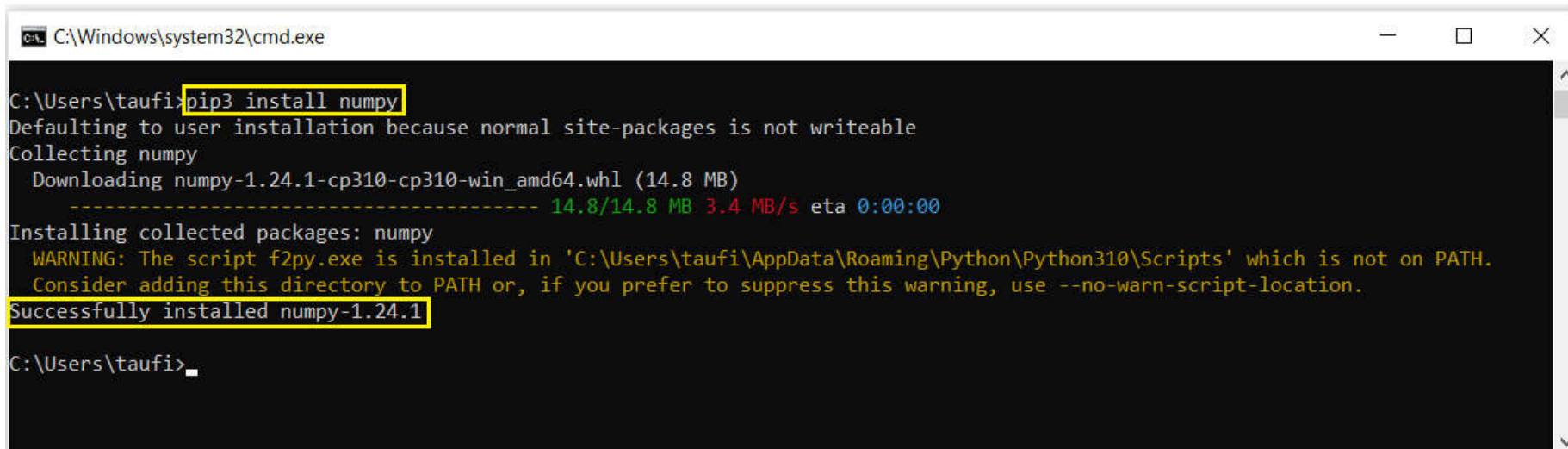

Instalasi Package/Library Eksternal dengan **PIP** (3)

Langkah Instalasi:

1. Pastikan **komputer terkoneksi dengan internet** (*Online*).
2. Tentukan package/library yang akan diinstal. Misal, **numpy**.
3. Untuk memulai instalasi. Ketik “**pip**” atau “**pip3**” diikuti perintah “**install**” dan **nama package/library**-nya. Kemudian tekan enter, sehingga proses instalasi package/library akan berjalan hingga selesai (*Successfully*).

```
pip install numpy
```

```
pip3 install numpy
```



```
C:\Windows\system32\cmd.exe

C:\Users\taufi>pip3 install numpy
Defaulting to user installation because normal site-packages is not writeable
Collecting numpy
  Downloading numpy-1.24.1-cp310-cp310-win_amd64.whl (14.8 MB)
    ----- 14.8/14.8 MB 3.4 MB/s eta 0:00:00
Installing collected packages: numpy
  WARNING: The script f2py.exe is installed in 'C:\Users\taufi\AppData\Roaming\Python\Python310\Scripts' which is not on PATH.
  Consider adding this directory to PATH or, if you prefer to suppress this warning, use --no-warn-script-location.
Successfully installed numpy-1.24.1

C:\Users\taufi>
```

Referensi mengenai PIP

- Installing Python Modules

<https://docs.python.org/3/installing/index.html#:~:text=pip%20is%20the%20preferred%20installer%20program.>

- Python PIP

https://www.w3schools.com/python/python_pip.asp

- pip (package manager)

[https://en.wikipedia.org/wiki/Pip_\(package_manager\)](https://en.wikipedia.org/wiki/Pip_(package_manager))

- Install PIP jika pada python belum tersedia PIP

<https://packaging.python.org/en/latest/tutorials/installing-packages/>

- pip

<https://pip.pypa.io/en/latest/>

- Commands

<https://pip.pypa.io/en/latest/cli/>

- The Python Standard Library

<https://docs.python.org/3/library/>

Terima Kasih