

**Packages**

It is an encapsulation mechanism to group related modules into a single unit.

package is nothing but folder or directory which represents collection of Python modules.

Any folder or directory contains init .py file,is considered as a Python package.This file can be empty.

A package can contains sub packages also.

|  |  |  |
| --- | --- | --- |
| **x.py** |  | **y.py** |

Loan

**init .py**

**init .py**

**init .py**

**Home Loan**

**Vehicle Loan**

**n.py**

**m.py**

**File 1**

**File 1**

**File 1**



Loan

**init .py**

**File 1**

**init .py**

**File 1**

**init .py**

**Home Loan**

**Vehicle Loan**

**Module 1**

**Module 1**

**Module 1**

**Module 1**

**File 1**

The main advantages of package statement are

1. **We can resolve naming conflicts**
2. **We can identify our components uniquely**
3. **It improves modularity of the application Eg 1:**

D:\Python\_classes>

|-test.py

|-pack1

|-module1.py

|- init .py

init .py:

empty file

module1.py: def f1():

print("Hello this is from module1 present in pack1")

test.py (version-1): import pack1.module1 pack1.module1.f1()



test.py (version-2):

from pack1.module1 import f1 f1()

Eg 2:

D:\Python\_classes>

|-test.py

|-com

|-module1.py

|- init .py

|-durgasoft

|-module2.py

|- init .py

init .py: empty file module1.py:

def f1():

print("Hello this is from module1 present in com") module2.py:

def f2():

print("Hello this is from module2 present in com.durgasoft") test.py:

* 1. **from com.module1 import f1**

**2. from com.durgasoft.module2 import f2**

**3. f1()**

**4. f2()**

**5.**

**6. Output**

**7. D:\Python\_classes>py test.py**

**8. Hello this is from module1 present in com**

**9. Hello this is from module2 present in com.durgasoft**

Note: Summary diagram of library,packages,modules which contains functions,classes and variables.



Library

pack 1 pack 2 --------------- pack n

module 1 module 2 module n module 1 module 2 module n

function 1 function 2 function n variables classes