МОСКОВСКИЙ ГОРОДСКОЙ УНИВЕРСИТЕТ
МГПУ

Как **Python** управляет модулями и пакетами. Работа с импортом

МОСКОВСКИЙ ГОРОДСКОЙ УНИВЕРСИТЕТ МГПУ

Python Packages

Модули

mod1.py

```python
def foo():
    print('[mod1] foo()')

class Foo:
    pass
```

mod2.py

```python
def bar():
    print('[mod2] bar()')

class Bar:
    pass
```

mod3.py

```python
def baz():
    print('[mod3] baz()')

class Baz:
    pass
```

mod4.py

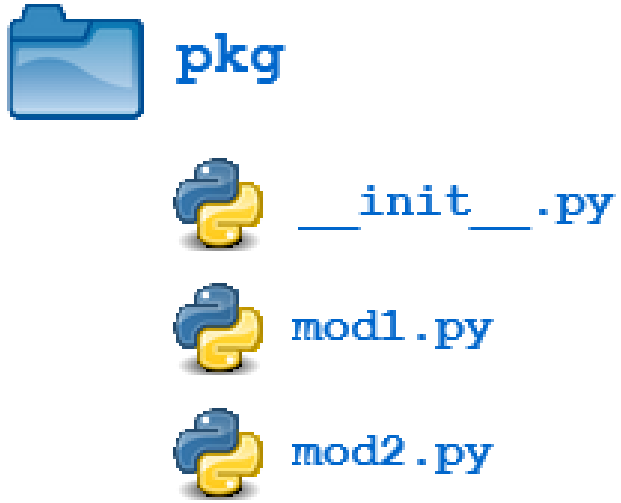
```python
def qux():
    print('[mod4] qux()')

class Qux:
    pass
```

МОСКОВСКИЙ ГОРОДСКОЙ УНИВЕРСИТЕТ МГПУ

Пакеты позволяют иерархически структурировать пространство имен модулей с помощью **записи через точку** . Точно так же, как **модули** помогают избежать конфликтов между именами глобальных переменных, **пакеты** помогают избежать конфликтов между именами модулей.

МОСКОВСКИЙ ГОРОДСКОЙ УНИВЕРСИТЕТ МГПУ



Здесь есть каталог с именем **pkg**, который содержит два модуля, **mod1.py** и **mod2.py**

```
import pkg
```

Если указанный файл **__init__.py** присутствует в каталоге пакета, он вызывается при импорте пакета или модуля в пакете.

```
import pkg
```

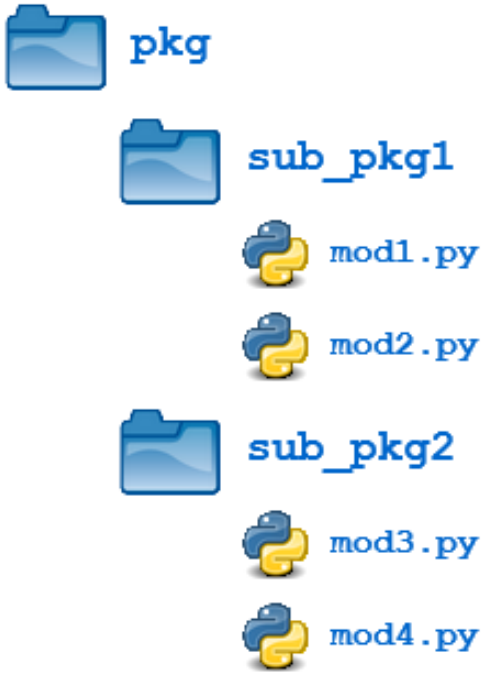
__init__.py

```python
print(f'Invoking __init__.py for {__name__}')
import pkg.mod1, pkg.mod2
```

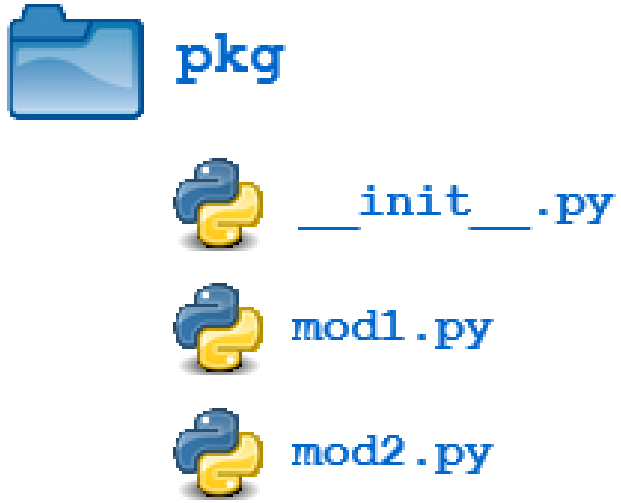
ИНИЦИАЛИЗИРУЮТСЯ МОДУЛИ**mod1.py** и **mod2.py**

Подпакеты. Пакеты могут содержать вложенные **подпакеты** произвольной глубины.

МОСКОВСКИЙ ГОРОДСКОЙ УНИВЕРСИТЕТ МГПУ



Здесь есть каталог с именем **pkg**, который содержит два модуля, **mod1.py** и **mod2.py**



Если указанный файл **__init__.py** присутствует в каталоге пакета, он вызывается при импорте пакета или модуля в пакете.

1. Соглашения об именах

МОСКОВСКИЙ ГОРОДСКОЙ УНИВЕРСИТЕТ МГПУ

Стиль Python во многом регулируется набором документов, называемых **Python Enhancement Proposals**, сокращенно **PEP** .

Модули должны иметь короткие имена со строчными буквами. В имени модуля можно использовать символы подчеркивания, если это улучшает читабельность. Пакеты Python также должны иметь короткие имена со строчными буквами, хотя использование символов подчеркивания не рекомендуется.

1. Соглашения об именах

МОСКОВСКИЙ ГОРОДСКОЙ УНИВЕРСИТЕТ
МГПУ

модули называются по именам файлов,

пакеты именуются по именам их каталогов

2. Модули и пакеты

МОСКОВСКИЙ ГОРОДСКОЙ УНИВЕРСИТЕТ МГПУ

Помимо сбора модулей в каталог, есть еще одно требование — определить каталог как пакет. Ключом к разработке пакета является добавление **__init__.py** файла в этот каталог.

Рекомендуемая структура

Пакет верхнего уровня **examplepackage** с тремя подпакетами:

- **common**
- **model**
- **tests**

- У нас также есть каталог **resources**, но он содержит только изображения и т. д. Это НЕ пакет, так как он не содержит файла **__init__.py**.

```
examplepackage-git/
├── examplepackage
│   ├── __init__.py
│   ├── app.py
│   ├── common
│   │   ├── __init__.py
│   │   └── utils.py
│   ├── model
│   │   ├── __init__.py
│   │   ├── nice_model.py
│   │   └── sub_model.py
│   ├── tests
│   │   ├── __init__.py
│   │   └── test_project.py
│   └── resources
├── LICENSE.md
├── MANIFEST.in
├── README.md
├── setup.py
└── venv
    ├── bin
    ├── include
    ├── lib
    ├── lib64 -> lib
    ├── pyvenv.cfg
    └── share
```

МОСКОВСКИЙ ГОРОДСКОЙ УНИВЕРСИТЕТ МГПУ

3. Как работает импорт

МОСКОВСКИЙ ГОРОДСКОЙ УНИВЕРСИТЕТ МГПУ

Python imports ищет каталоги, перечисленные в **sys.path**, используя **текущий рабочий каталог**, за которым следуют каталоги, перечисленные в **PYTHONPATH** переменной среды.

ПРИМЕР 1 smart_door.py

МОСКОВСКИЙ ГОРОДСКОЙ УНИВЕРСИТЕТ

```
# smart_door.py
def close():
    print("Ahhhhhhhhhhhh.")

def open():
    print("Thank you for making a simple door very happy.")
```

Если мы хотим запустить функцию **open()**, мы должны сначала импортировать модуль **smart_door**. Самый простой способ сделать это

ПРИМЕР 1 smart_door.py

МОСКОВСКИЙ ГОРОДСКОЙ УНИВЕРСИТЕТ МГПУ

```
import smart_door
smart_door.open()
smart_door.close()
```

Если мы хотим запустить функцию **open()**, мы должны сначала импортировать модуль **smart_door**. Самый простой способ сделать это

ПРИМЕР 1 smart_door.py

МОСКОВСКИЙ ГОРОДСКОЙ УНИВЕРСИТЕТ МГПУ

Если мы хотим иметь возможность использовать **open()** функцию без необходимости постоянно использовать имя модуля, мы можем сделать

```
from smart_door import open
open()
```

ПРИМЕР 1 smart_door.py

МОСКОВСКИЙ ГОРОДСКОЙ УНИВЕРСИТЕТ МГПУ

«Пусть у меня в модуле сотни функций, и я хочу использовать их все!» Это момент, когда многие разработчики сходят с рельсов, делая это

```
from smart_door import *
```

Импорт в Ваш проект

Пример файловой структуры проекта

В моем **nice_model** модуле, определенном с помощью examplepackage/model/nice_model.py, хочу использовать MyGenerator класс. Этот класс определен в examplepackage/common/utils.py. Как мне добраться до него?

Поскольку я определил **examplepackage** как пакет и организовал свои модули в подпакеты, на самом деле это довольно просто. В **nice_model.py** говорим

```
from examplepackage.common.utils import MyGenerator
```

Это называется **абсолютным импортом**. Он начинается с пакета верхнего уровня, examplepackage, и переходит в **common** пакет, где ищет файлы **utils.py**.

```
examplepackage-git/
├── examplepackage
│   ├── __init__.py
│   ├── app.py
│   ├── __main__.py
│   ├── common
│   │   ├── __init__.py
│   │   └── utils.py
│   ├── model
│   │   ├── __init__.py
│   │   ├── nice_model.py
│   │   └── sub_model.py
│   ├── tests
│   │   ├── __init__.py
│   │   └── test_project.py
│   └── resources
├── LICENSE.md
├── MANIFEST.in
├── README.md
├── setup.py
└── venv
    ├── bin
    ├── include
    ├── lib
    ├── lib64 -> lib
    ├── pyvenv.cfg
    └── share
```

МОСКОВСКИЙ ГОРОДСКОЙ УНИВЕРСИТЕТ МГПУ

Импорт в Ваш проект

Альтернативный вариант

```
from ..common.utils import MyGenerator
```

Существует много споров о том, использовать ли **абсолютный** или **относительный** импорт. Единственная важная часть заключается в том, что результат *очевиден* — **не должно быть никакой тайны, откуда что-то берется**.

```
examplepackage-git/
├── examplepackage
│   ├── __init__.py
│   ├── app.py
│   ├── __main__.py
│   ├── common
│   │   ├── __init__.py
│   │   └── utils.py
│   ├── model
│   │   ├── __init__.py
│   │   ├── nice_model.py
│   │   └── sub_model.py
│   ├── tests
│   │   ├── __init__.py
│   │   └── test_project.py
│   └── resources
├── LICENSE.md
├── MANIFEST.in
├── README.md
├── setup.py
└── venv
    ├── bin
    ├── include
    ├── lib
    ├── lib64 -> lib
    ├── pyvenv.cfg
    └── share
```

МОСКОВСКИЙ ГОРОДСКОЙ УНИВЕРСИТЕТ МГПУ

МОСКОВСКИЙ ГОРОДСКОЙ УНИВЕРСИТЕТ
МГПУ

__main__.py

__main__.py

Это специальный файл в пакете верхнего уровня, который выполняется, когда запускаем пакет непосредственно с помощью Python.
Пакет **examplepackage** можно запустить из корня репозитория с ключем **python -m examplepackage**.

```
from examplepackage import app

if __name__ == '__main__':
    app.run()
```

```
examplepackage-git/
├── examplepackage
│   ├── __init__.py
│   ├── app.py
│   ├── __main__.py
│   ├── common
│   │   ├── __init__.py
│   │   └── utils.py
│   ├── model
│   │   ├── __init__.py
│   │   ├── nice_model.py
│   │   └── sub_model.py
│   ├── tests
│   │   ├── __init__.py
│   │   └── test_project.py
│   └── resources
├── LICENSE.md
├── MANIFEST.in
├── README.md
├── setup.py
└── venv
    ├── bin
    ├── include
    ├── lib
    ├── lib64 -> lib
    ├── pyvenv.cfg
    └── share
```

МОСКОВСКИЙ ГОРОДСКОЙ УНИВЕРСИТЕТ МГПУ

Задание 1

Создать модули и через обращение к ним решить задачу

fact.py

```
def fact(n):
    return 1 if n == 1 else n * fact(n-1)

if (__name__ == '__main__'):
    import sys
    if len(sys.argv) > 1:
        print(fact(int(sys.argv[1])))
```

Подсчитать 10!

Задание 2

Создать пакет и через обращение к нему решить задачу

МОСКОВСКИЙ ГОРОДСКОЙ УНИВЕРСИТЕТ МГПУ

```
# Fibonacci numbers module

def fib(n):    # write Fibonacci series up to n
    a, b = 0, 1
    while a < n:
        print(a, end=' ')
        a, b = b, a+b
    print()

def fib2(n):   # return Fibonacci series up to n
    result = []
    a, b = 0, 1
    while a < n:
        result.append(a)
        a, b = b, a+b
    return result
```

Подсчитать n<10 fib(n)
n>10 fib2(n)

Задание 3

Решить задачу в VS code
и провести отладку кода

МОСКОВСКИЙ ГОРОДСКОЙ УНИВЕРСИТЕТ МГПУ

```
import matplotlib.pyplot as plt
import numpy as np

x = np.linspace(0, 20, 100)  # Create a list of evenly-spaced numbers over the range
plt.plot(x, np.sin(x))       # Plot the sine of each x point
plt.show()                   # Display the plot
```

https://github.com/BosenkoTM/Python-Programming-A-101/blob/main/lectures/pract_on_lect_8_vs_code_vs_python.pdf

МОСКОВСКИЙ ГОРОДСКОЙ УНИВЕРСИТЕТ
МГПУ

129226, г. Москва, 2-й Сельскохозяйственный проезд, 4
info@mgpu.ru
+7 (499) 181-24-62
www.mgpu.ru

Университет твоих возможностей