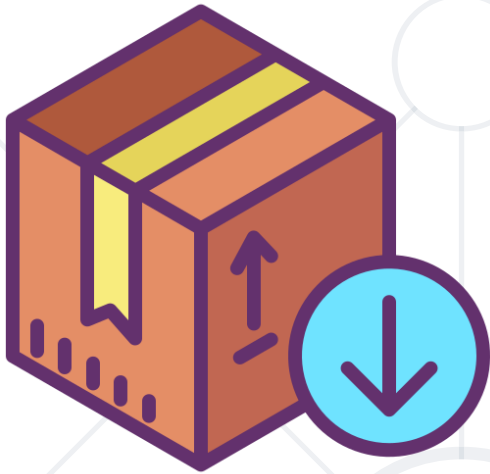


Modules

Python packages



SoftUni Team
Technical Trainers



SoftUni



Software University

<http://softuni.bg>

sli.do

#python-advanced

Table of Contents

1. What are modules
 - Definition and usage
2. Built-in modules
3. External modules
 - PIP
4. Custom modules





What are Modules?

Definition and Usage

- Simply put, a **module** is a file consisting of Python code
 - They are stored in **packages**
 - A **package** is a collection of Python modules
- Why use modules?
 - Keep Python files short and simple
 - Reuse code across multiple files by importing





Built-in Modules

- The Python interpreter has a number of **built-in** modules
- They are pre-installed and we can call them at any given time
- In order to call them we use the keyword - **import**

```
import random  
fruits = ["apple", "banana", "cherry"]  
  
random.choice(fruits)  
random.shuffle(fruits)
```

Different Ways to Import

```
import random as module_name  
module_name.randint(1, 10)
```

```
from random import choice as gimme_one, shuffle as mix  
gimme_one(["coke", "steak", "chips"])  
mix(["coke", "steak", "chips"])
```

```
from math import *  
sqrt(pi)
```


Problem: Calculate Logarithm

- Write a program that prints the calculated logarithm of any given number
- You will receive **2** inputs
 - The number (integer)
 - The base (if it is the word "natural" find the natural logarithm)
- Format the result up to the 2nd decimal digit and print it



Solution: Calculate Logarithm

```
from math import log
number = int(input())
base = input()
if base == "natural":
    print(f"{log(number):.2f}")
else:
    print(f"{log(number, int(base)):.2f}")
```

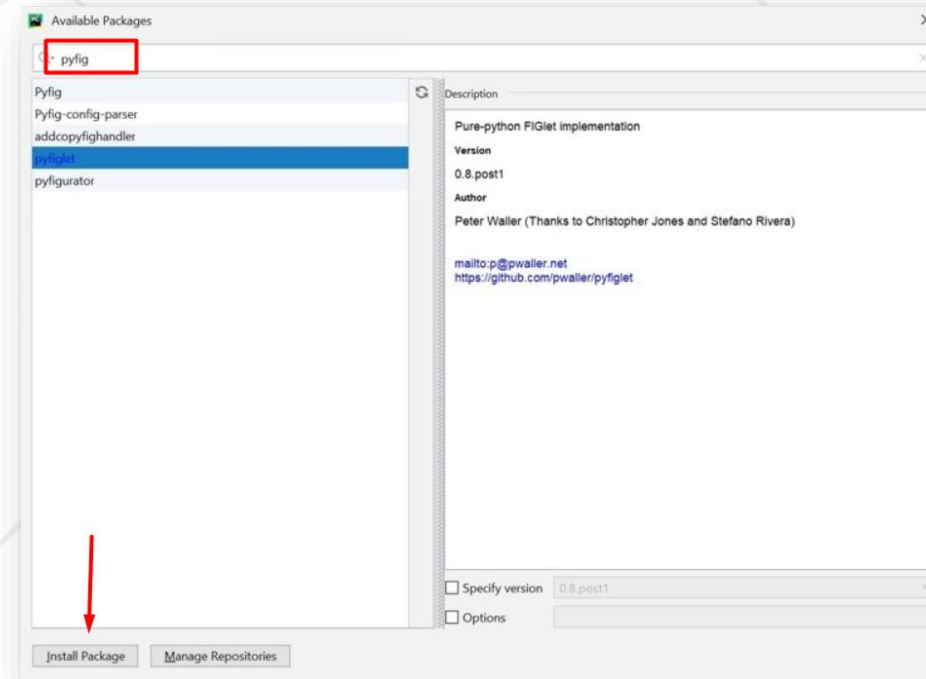
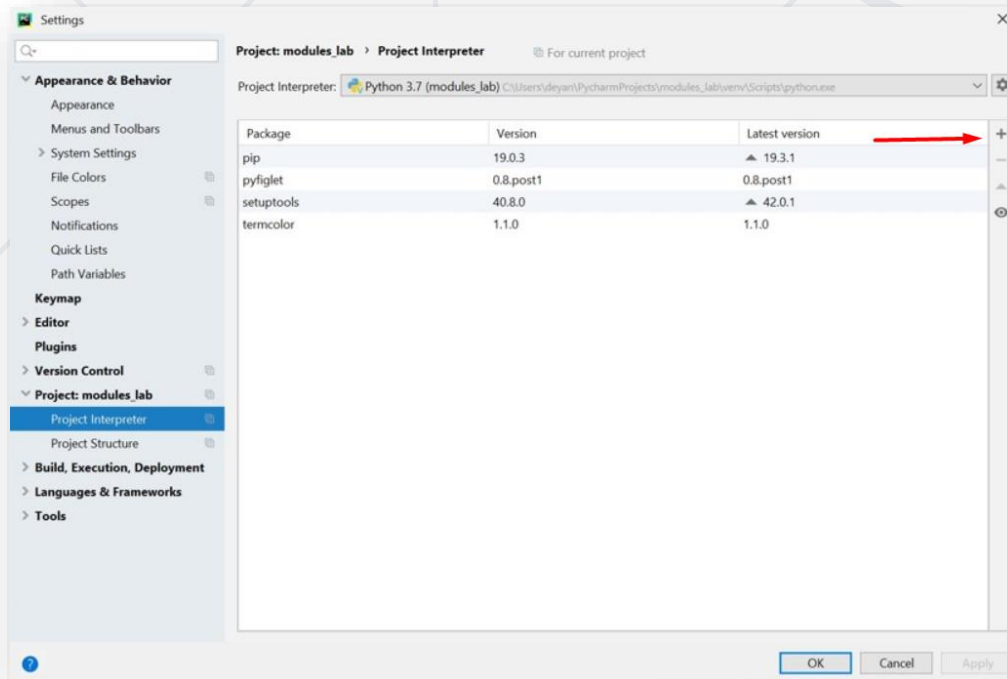


External Modules

PIP

Package Management System (PIP)

- External modules are downloaded from the internet by using **pip**
- In PyCharm you have **pip** already installed, so you can jump straight to installing them



- A **Requirements file** is just a list of pip install arguments placed in a file
- You should **not rely** on the items in the file being **installed** by pip in **any particular order**

```
### example-requirements.txt ###  
numpy  
matplotlib  
django == 3.0.1  
  
# Refer to other requirements files #  
-r other-requirements.txt
```

Requirements Files – Common Uses

- There are **4 common uses** of Requirements files:
 - hold the result from pip freeze
 - force pip to resolve dependencies properly
 - force pip to install an alternate version of a sub-dependency
 - override a dependency with a local patch that lives in version control



- Termcolor

```
from termcolor import colored
text = colored('Hello World!', 'red', attrs=['bold', 'underline'])
print(text)    # Hello World!
```

- PyFiglet

```
from pyfiglet import figlet_format
text =figlet_format("Python",font="isometric1")
print(text)
```

Problem: ASCII Art

- Using the **pyfiglet** package, write a program that **encrypts given words** by using the characters: "-|_/\()" to structure the word

```
from pyfiglet import figlet_format

def print_art(msg):
    ascii_art = figlet_format(msg)
    print(ascii_art)

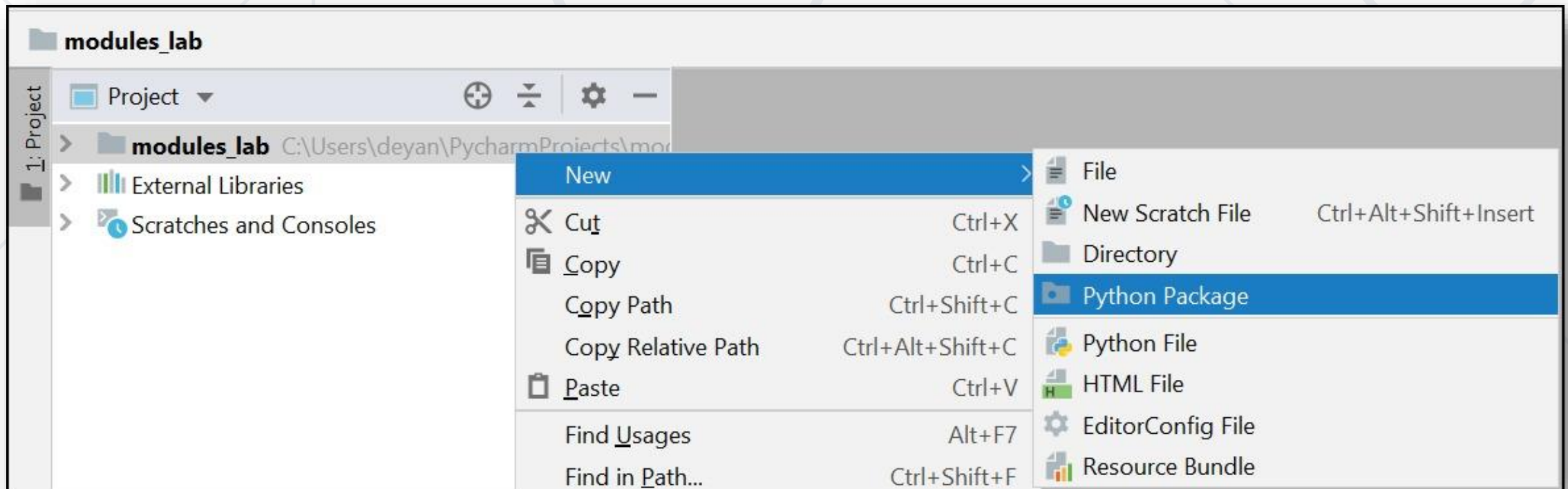
msg = input("What would you like to print? ")
print_art(msg)
```




Custom Modules

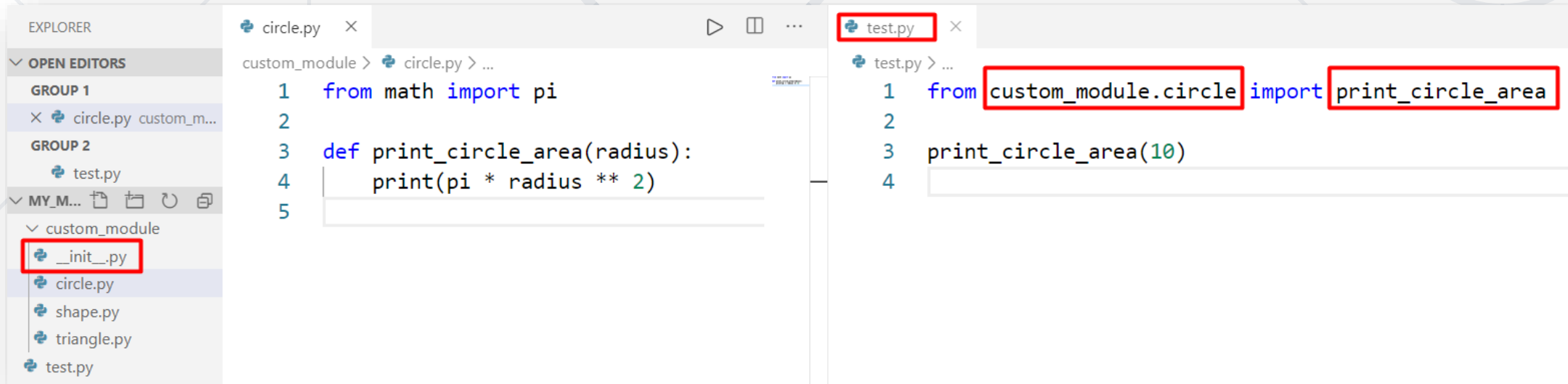
Creating a Module (Through PyCharm)

- Any **.py** file can be imported and used as a module
- In PyCharm there is a separate option to create a package(module)



The `__init__` File in a Module

- You can make **any folder** a **module** by just adding an **`__init__`** file in the folder
- The file **can be empty**



The screenshot shows the Visual Studio Code interface with two open files. On the left, the Explorer sidebar shows a project structure with a folder named `custom_module` containing `__init__.py`, `circle.py`, `shape.py`, `triangle.py`, and `test.py`. The `__init__.py` file is highlighted with a red box. The main editor shows the `circle.py` file with the following code:

```
1 from math import pi
2
3 def print_circle_area(radius):
4     print(pi * radius ** 2)
5
```

On the right, the `test.py` file is open, showing the following code:

```
1 from custom_module.circle import print_circle_area
2
3 print_circle_area(10)
4
```

The `test.py` tab is highlighted with a red box. The `from custom_module.circle import print_circle_area` line is also highlighted with a red box.

Problem: Triangle

- Create a module for printing a triangle
- You will receive an integer number which is the size of the triangle

3



```
1
1 2
1 2 3
1 2
1
```

4



```
1
1 2
1 2 3
1 2 3 4
1 2 3
1 2
1
```

Solution: Triangle

```
def print_triangle(size):  
    for row in range(1, size + 2):  
        print(*[col for col in range(1, row)])  
    for row in range(size, 0, -1):  
        print(*[col for col in range(1, row)])
```

```
from triangle import *  
  
size = int(input())  
print_triangle(size)
```

- A **module** is a file consisting of Python code
- There are a couple of ways to **import** a module:

```
import <module_name>
```

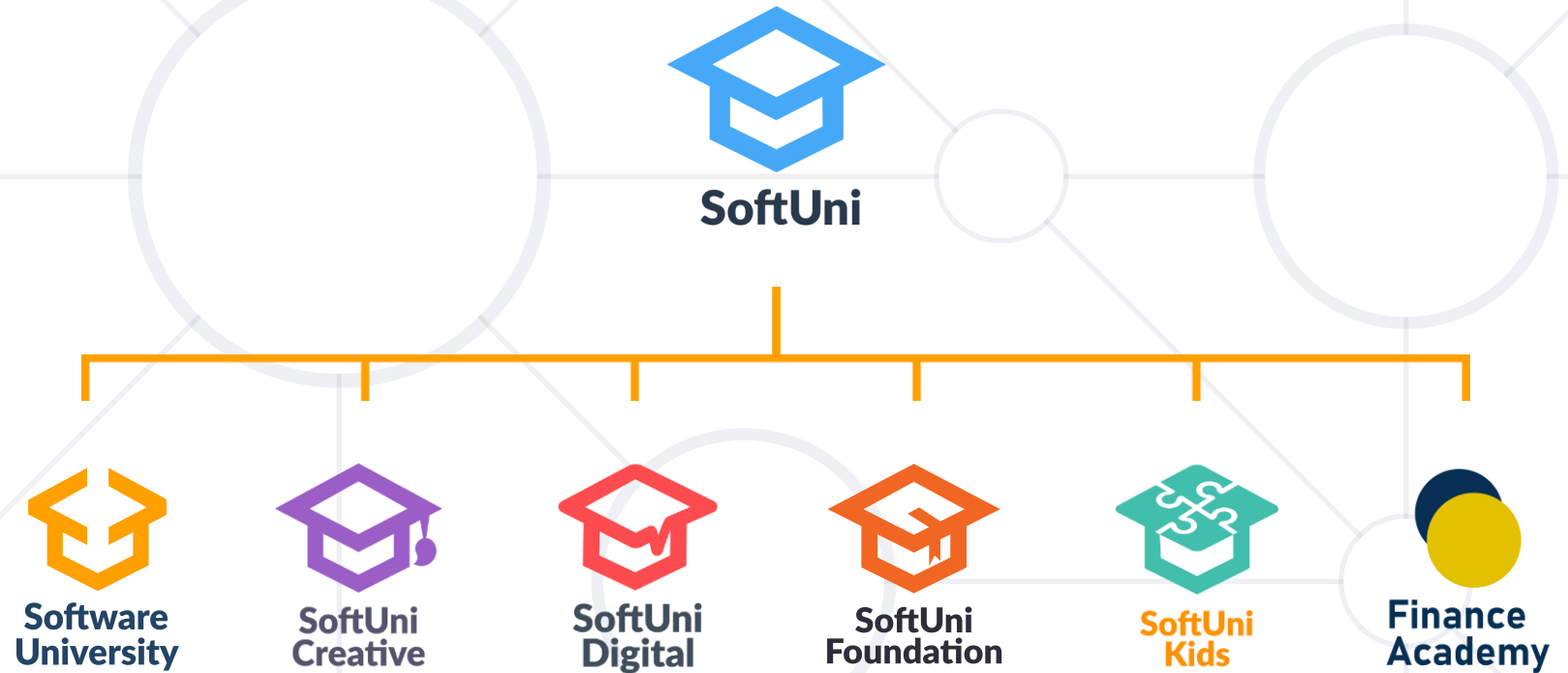
```
from <module_name> import <module_attribute>
```

```
import <module_name> as <custom_name>
```

- Modules make our code short, simple, and reusable



Questions?



SoftUni Diamond Partners



- Software University – High-Quality Education, Profession and Job for Software Developers
 - softuni.bg, softuni.org
- Software University Foundation
 - softuni.foundation
- Software University @ Facebook
 - facebook.com/SoftwareUniversity
- Software University Forums
 - forum.softuni.bg



- This course (slides, examples, demos, exercises, homework, documents, videos, and other assets) is **copyrighted content**
- Unauthorized copy, reproduction, or use is illegal
- © SoftUni – <https://about.softuni.bg>
- © Software University – <https://softuni.bg>

