



Python Package

In this article, you'll learn to divide your code base into clean, efficient modules using Python packages. Also, you'll learn to import and use your own or third party packages in your Python program.

Table of Contents

- [What are packages?](#)
- [Importing module from a package](#)

What are packages?

We don't usually store all of our files in our computer in the same location. We use a well-organized hierarchy of directories for easier access.

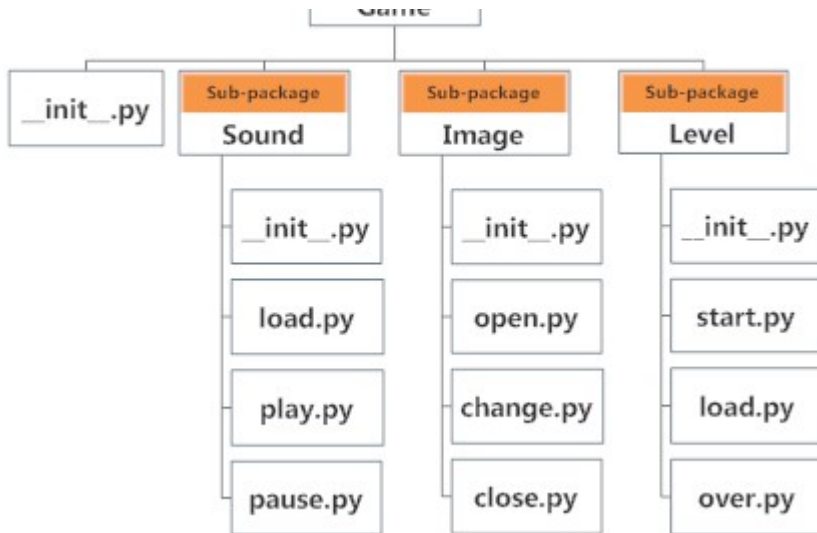
Similar files are kept in the same directory, for example, we may keep all the songs in the "music" directory. Analogous to this, Python has packages for directories and [modules](#) for files.

As our application program grows larger in size with a lot of modules, we place similar modules in one package and different modules in different packages. This makes a project (program) easy to manage and conceptually clear.

Similar, as a directory can contain sub-directories and files, a Python package can have sub-packages and modules.

A directory must contain a file named `__init__.py` in order for Python to consider it as a package. This file can be left empty but we generally place the initialization code for that package in this file.

Here is an example. Suppose we are developing a game, one possible organization of packages and modules could be as shown in the figure below.



Importing module from a package

We can import modules from packages using the dot (.) operator.

For example, if want to import the `start` module in the above example, it is done as follows.

```
import Game.Level.start
```

Now if this module contains a `function` named `select_difficulty()`, we must use the full name to reference it.

```
Game.Level.start.select_difficulty(2)
```

If this construct seems lengthy, we can import the module without the package prefix as follows.

```
from Game.Level import start
```

We can now call the function simply as follows.

```
start.select_difficulty(2)
```



```
from Game.Level.start import select_difficulty
```

Now we can directly call this function.

```
select_difficulty(2)
```

Although easier, this method is not recommended. Using the full [namespace](#) avoids confusion and prevents two same identifier names from colliding.

While importing packages, Python looks in the list of directories defined in `sys.path`, similar as for [module search path](#).

PREVIOUS

[PYTHON MODULES](#)

NEXT

[PYTHON NUMBERS AND TYPE CONVERSION](#)

Want to learn more Python for Data Science? Head over to DataCamp and try their free Python Tutorial

Python Tutorial

Python Introduction



Python Flow Control



Python Functions



Python Function

Function Argument

Python Recursion

[TUTORIAL](#)[EXAMPLES](#)[BUILT-IN FUNCTIONS](#)[Python Global Keyword](#)[Python Modules](#)[Python Package](#)[Take Quiz](#)[Python Datatypes](#)[Python Files](#)[Python Object & Class](#)[Advanced Topics](#)

Receive the latest tutorial to improve your programming skills.

[Join](#)

Get Latest Updates on Programiz

[Subscribe](#)[ABOUT](#)[CONTACT](#)[ADVERTISE](#)[C PROGRAMMING](#)[C++ PROGRAMMING](#)[R PROGRAMMING](#)

Copyright © by Programiz | All rights reserved | [Privacy Policy](#)