Python - Modules & Packages

- Simplicity
- Maintainability
- Reusability
- Scoping

Module:

- keep definitions in a file
- use them in various scripts

In [1]:
```
!cat fibo.py
```

```
# 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
```

In [2]:
```
import fibo
```

In [3]:
```
fibo.fib(100)
```
```
0 1 1 2 3 5 8 13 21 34 55 89
```

In [4]:
```
fibo.fib2(100)
```
Out[4]:
```
[0, 1, 1, 2, 3, 5, 8, 13, 21, 34, 55, 89]
```

In [5]:
```
from fibo import fib, fib2
```

In [6]:
```
fib(100)
```
```
0 1 1 2 3 5 8 13 21 34 55 89
```

In [7]:
```
fib2(100)
```
Out[7]:
```
[0, 1, 1, 2, 3, 5, 8, 13, 21, 34, 55, 89]
```

In [8]:
```
from fibo import *
```

In [9]:
```
fib(100)
```
```
0 1 1 2 3 5 8 13 21 34 55 89
```

In [10]:
```
fib2(100)
```
Out[10]:
```
[0, 1, 1, 2, 3, 5, 8, 13, 21, 34, 55, 89]
```

In [11]:
```
import fibo as myfib
```

In [12]:
```
myfib.fib(100)
```
```
0 1 1 2 3 5 8 13 21 34 55 89
```

In [13]:
```
myfib.fib2(100)
```

Out[13]:
```
[0, 1, 1, 2, 3, 5, 8, 13, 21, 34, 55, 89]
```

In [14]:
```
from fibo import fib as fibonacci
```

In [15]:
```
fibonacci(100)
```

```
0 1 1 2 3 5 8 13 21 34 55 89
```

In [16]:
```
from fibo import fib as myfib1, fib2 as myfib2
```

In [17]:
```
myfib1(100)
```

```
0 1 1 2 3 5 8 13 21 34 55 89
```

In [18]:
```
myfib2(100)
```

Out[18]:
```
[0, 1, 1, 2, 3, 5, 8, 13, 21, 34, 55, 89]
```

In [ ]:
```

```

In [19]:
```
dir(fibo)
```

Out[19]:
```
['__builtins__',
 '__cached__',
 '__doc__',
 '__file__',
 '__loader__',
 '__name__',
 '__package__',
 '__spec__',
 'fib',
 'fib2']
```

In [20]:
```
import math
dir(math)
```

Out[20]:
```
['__doc__',
 '__file__',
 '__loader__',
 '__name__',
 '__package__',
 '__spec__',
 'acos',
 'acosh',
 'asin',
 'asinh',
 'atan',
 'atan2',
 'atanh',
 'ceil',
 'copysign',
 'cos',
 'cosh',
 'degrees',
 'e',
 'erf',
 'erfc',
 'exp',
 'expm1',
 'fabs',
 'factorial',
 'floor',
 'fmod',
 'frexp',
 'fsum',
 'gamma',
 'gcd',
 'hypot',
 'inf',
 'isclose',
 'isfinite',
 'isinf',
 'isnan',
 'ldexp',
 'lgamma',
 'log',
 'log10',
 'log1p',
 'log2',
 'modf',
 'nan',
 'pi',
 'pow',
```

Packages

- **Hierarchical structuring of module namespace**
- **Helps avoid collision between module names**
- **https://docs.python.org/3/tutorial/modules.html#packages (https://docs.python.org/3/tutorial/modules.html#packages)**

```
sound/                          Top-level package
      __init__.py               Initialize the sound package
      formats/                  Subpackage for file format conversions
              __init__.py
              wavread.py
              wavwrite.py
              aiffread.py
              aiffwrite.py
              auread.py
              auwrite.py
              ...
      effects/                  Subpackage for sound effects
              __init__.py
              echo.py
              surround.py
              reverse.py
              ...
      filters/                  Subpackage for filters
              __init__.py
              equalizer.py
              vocoder.py
              karaoke.py
              ...
```

import individual modules

In [21]:
```
import sound.effects.echo
```

In [22]:
```
sound.effects.echo.echofilter("in", "out")
```

alternative way of importing the submodule

- **makes it available without its package prefix**

In [23]:
```
from sound.effects import echo
```

In [24]:
```
echo.echofilter("in", "out")
```

import the desired function or variable directly

In [25]:
```
from sound.effects.echo import echofilter
```

In [26]:
```
echofilter("in", "out")
```

In [27]:
```
from sound.effects.echo import echofilter as ef
```

In [28]:
```
ef("in", "out")
```

Importing * from a package

In [29]:
```
from sound.effects import *
```

In [30]:
```
sound.effects.reverse.test("foo")
```

```
---------------------------------------------------------------------------
AttributeError                            Traceback (most recent call last)
<ipython-input-30-d72f21242e41> in <module>
----> 1 sound.effects.reverse.test("foo")

AttributeError: module 'sound.effects' has no attribute 'reverse'
```

In [31]:
```
!cat sound/formats/__init__.py
```

```
__all__ = ["wavread", "wavwrite"]
```

In [32]:
```
from sound.formats import *
```

In [33]:
```
sound.formats.wavread.read("foo")
```

In [ ]:
```

```