Python

Intermediate-level workshop

Alexandru Cristiean, PAL, University of Sussex, 30 Oct 2020

Covering today

- Structuring code in Python
- Importing modules
- Mutable and immutable objects
- Dictionaries and tuples
- List comprehension
- IPython and notebooks

Recommended resource: https://docs.python.org/3/tutorial/

Python structure and imports

Structure of a program

- Code blocks
  - commands (docs: Using the Python interpreter, Interactive mode)
  - function body
  - class definition
  - module (docs: Modules)
- Packages (docs: Packages)
  - pip - package installer for Python
  - pypi - Python Package Index

Python structure and imports

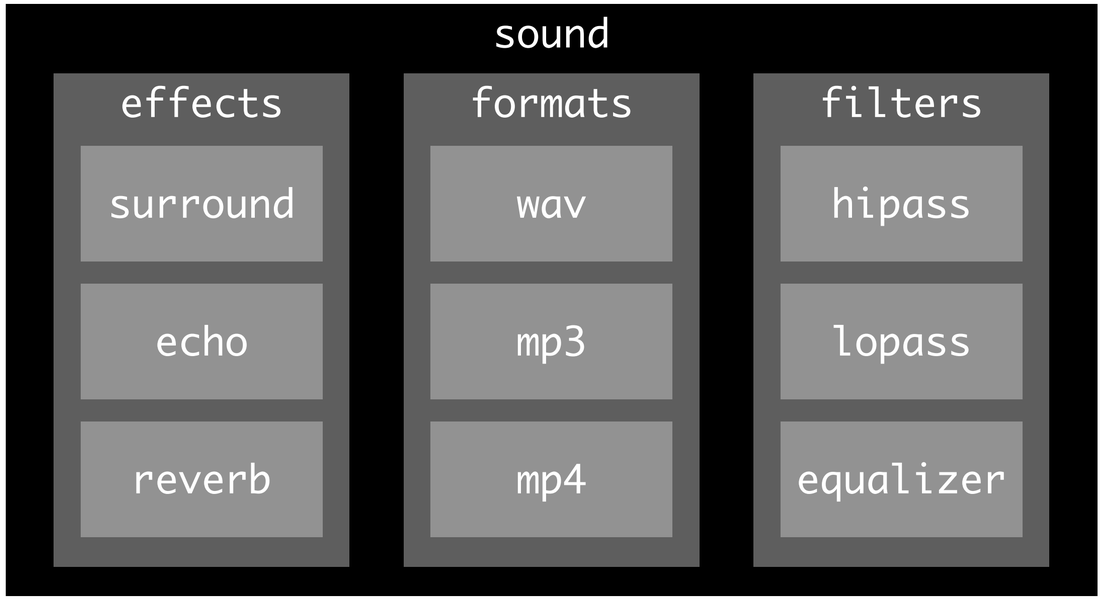
Imports

- Modules:
  - `import numpy, matplotlib, time`
  - `import numpy as np` (bind another name to the imported module)
  - `from numpy import pi` (import names from a module)
  - `from numpy import pi as PI`
  - `from numpy import *` (import all names a module defines)

Python structure and imports

Imports

- Packages:
  - `import sound.effects.echo` (import individual modules from package)
  - Intra-package references (e.g. below: statements in the `surround` module)
    - `from . import echo`
    - `from .. import formats`
    - `from ..filters import equalizer`



Mutable and immutable

- Immutable objects
  - `bool`, `int`, `float`, `str`
  - tuples `(1, 2, 2)`
  - range `range(1, 21)`
  - frozenset `frozenset(iterable)`
- Mutable objects
  - list `[1, 2, 2]`
  - set `{1, 2, 3}`
  - dict `{"United Kingdom": 44}`

Dictionaries and tuples

- Tuples immutability
  - finite
  - memory efficient
  - hashable
- Dictionary keys
  - Immutable keys allowed ✅ `a_dict = {"Sussex": True, (1, 2, 3): True, range(1,10): True}`
  - Mutable keys not allowed ⛔ `a_dict = {[1, 2, 1]: True, {1, 2, 3}: True, {1: True}: True}`

List comprehension

Lists

Instead of:

```
squares = []
for x in range(5):
  squares.append(x**2)
```

We can write:

```
[x**2 for x in range(5)]
```

List comprehension

Lists

- Note, we can also re-write **nested for-loops as list comprehensions**:

```
li = []
for number in range(10):
   for exponent in range(3):
      li.append(number ** exponent)
```

is equivalent to:

```
[number ** exponent for exponent in range(3) for number in range(10)]
```

- Note, we can include **conditionals within list comprehensions**:

```
[(x, y) for x in [1,2,3] for y in [3,2,1] if x!=y]
```

- Note, we can have **nested list comprehensions** (doc: Nested List Comprehensions):

```
[[x+y for y in range(5)] for y in range(10)]
```

List comprehension

Tuples and dictionaries

- Tuple comprehensions
  - `tuple(x for x in range(10))`
  - Note, the notation `(x for x in range(10))` is taken by generator expressions (see Stack Overflow question Why is there no tuple comprehension in Python?)
- Dictionary comprehensions
  - ```
    # all even ints in range (0, 10) to their squared value
    {x: x**2 for x in range(10) if x % 2 == 0}
    ```
    ```

IPython and notebooks

- Interactive computing - software accepts input from user as it runs
- IPython
- Project Jupyter (Julia, Python, R)
- Jupyter notebook
- Interfaces for cloud (Colaboratory, SageMaker Notebooks, Azure Notebooks)