

I/O Channels, Serialization: pickle

28 Oct 2010

CMPT140

Dr. Sean Ho

Trinity Western University

Quiz 4 (5min, 10pts)

- Name 3 out of the 4 **modes** in which a file may be **opened** (excluding “binary modes”). Also indicate the letter codes used to specify the file mode to Python:
`open('input.txt', mode)` [3]
- What is **aliasing**? [3]
Write Python **code** to demonstrate an alias.
- What are **accessor/mutator** methods? Why are they a **good** idea? [4]

Quiz 4: answers #1-2

- Name 3 out of the 4 **modes** in which a file may be **opened** (excluding “binary modes”). **[3]**

- **read ('r'), write ('w'), append ('a'), both ('r+')**

- What is **aliasing**? **[3]**

Write Python **code** to demonstrate an alias.

- Multiple **names** which refer to the same **object**.

```
apples = [ 'Fuji', 'Gala', 'Honeycrisp' ]
```

```
appleAlias = apples
```

Quiz 4 (5min, 10pts)

- What are **accessor/mutator** methods?
Why are they a **good** idea? [4]
 - Protect access to **attributes** of an **object** [1]
 - Accessor: **read**. Mutator: **write**. [1]
 - Attributes should be made **private** [1]
 - Prevent code **outside** the class from direct access to attributes:
e.g., **error-checking** before allowing an attribute to be modified [1]

I/O channels



- Abstractly, a **stream** of input comes over a **channel** from a **source**
 - e.g., source can be keyboard, file, program,...
- A stream is output over a channel to a **sink**
 - e.g., sink can be screen, file, program, etc.
- **I/O channels** (file descriptors, file handles) can be opened in one of three **modes**:
 - **Read**, **write**, and **read/write**
- **Default**: source is keyboard, sink is screen
- Can **redirect** channels to other source/sink

Standard I/O channels

- In the standard `sys` library: `import sys`
- Standard **Input**: `sys.stdin`
 - Default: keyboard
- Standard **Output**: `sys.stdout`
 - Default: screen, but we can **redirect** to a file
- Standard **Error**: `sys.stderr`
 - Default: screen (in case stdout is redirected)
- Alternative to `print()`: `sys.stdout.write()`
 - Newlines must be explicit
- Alternative to `input()`: `sys.stdin.readline()`

Redirecting standard I/O

- You can **redirect** the standard I/O channels just by **reassigning** them:
- e.g., make **print()** go to a **file**:

```
import sys
```

```
old_stdout = sys.stdout          # save stdout
```

```
with open('log.txt', 'w') as sys.stdout: # redirect
```

```
    print( 'Hello!' )           # goes to file
```

```
...
```

```
sys.stdout = old_stdout        # restore stdout
```

Serialization

- Python's **I/O** framework is flexible enough to support I/O with any kind of **stream**:
 - Files, **network** sockets, other programs (**IPC**), **hardware** drivers (e.g., robotics)
- To send data, it must be **serialized**: converted into a **stream** of bytes which we can **.write()** to a file handle or I/O stream
- Different data **types** serialize differently
 - If we make our own **new** data types, we have to **specify** how to serialize
- **Pickle** is Python's framework for serialization

How to pickle/unpickle?

- `import pickle`
 - **Open** the file (read or write mode)
 - **Write** the object: `pickle.dump(obj, file)`
 - **Read** an object: `obj = pickle.load(file)`
- Pickled objects can be **interspersed** with regular text in the file; you just have to `seek()` to the right spot where the pickled object should be
- **Get** the pickled object without writing to file:
- `pickledObj = pickle.dumps(obj)`
 - This is just a **string**; you can then `write()` it

What can be pickled?

- None, True, False
- ints, floats, complex, strings,
- Tuples, lists, sets, dictionaries, but only if all elements are picklable
- Functions defined at the top level of a module
- Classes (user-defined types) that are defined at the top level of a module
- Instances of such classes, but only if all attributes are picklable
 - `__dict__` tells you what an object contains

For more information

- Python **Tutorial** ch7 on I/O:
 - <http://docs.python.org/py3k/tutorial/inputoutput.html>
- Python **I/O** Library reference:
 - <http://docs.python.org/lib/bltin-file-objects.html>
- Python **pickle** library reference:
 - <http://docs.python.org/library/pickle.html>