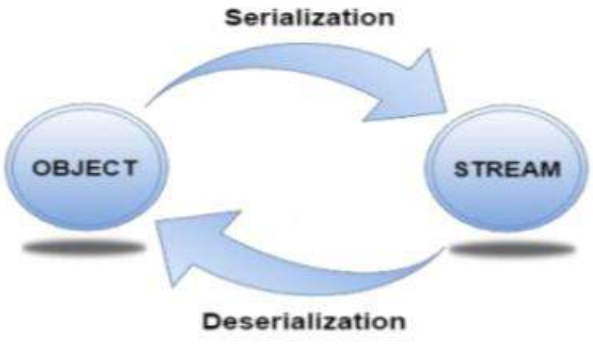
pickle

Python object serialization

The pickle module implements binary protocols for **serializing** and **de-serializing** a Python object structure.

"Pickling" is the process whereby a Python object hierarchy is converted into a byte stream, and ***"unpickling"*** is the inverse operation, whereby a byte stream is converted back into an object hierarchy.



Pickling (and unpickling) is alternatively known as "**serialization**", "**marshalling**," or "**flattening**"; however, to avoid confusion, the terms used here are "**pickling**" and "**unpickling**".

The pickle module provides the following functions to make the pickling process more convenient:

pickle.**dump**(*obj*, *file*)

Write the pickled representation of the object *obj* to the open file object *file*.

pickle.**load**(*file*)

Read the pickled representation of an object from the open file object *file* and return the reconstituted object hierarchy specified therein.

What can be pickled?

- Booleans,
- Integers,
- Floats,
- Complex numbers,
- (normal and Unicode) Strings,
- Tuples,
- Lists,
- Sets, and
- Dictionaries that ontain picklable objects.

All the above can be pickled, but you can also do the same for classes and functions.

Example on serializing

```
idno = int(input("Employee IDNO :"))
name = input("Employee NAME : ")
salary = float(input("Employee SALARY : "))
import pickle
file = open("employee_info.txt","wb")
pickle.dump(idno,file)
pickle.dump(name,file)
pickle.dump(salary,file)
file.close()
print("Data Written to File")
```

Output

```
Employee IDNO :101

Employee NAME : Ravi

Employee SALARY : 185000.00

Data Written to File
```

Example on de-serializing

```
import pickle
file = open("employee_info.txt","rb")
id = pickle.load(file)
na = pickle.load(file)
sal = pickle.load(file)
```

```
print(id,na,sal)
file.close()
```

Output

```
101 Ravi 185000.0
```

Example Script to Write list set, tuple and dictionary object's into a File

```
# list data
    my_friends = ['Ravi','Kumar','Krishna','Mohan']
# set data
    contacts =
{9876543210,9876543211,9876543212,9876543213}
# tuple data
    nick_name = ('aa','bb','cc','dd')
# dict Data
    details = {"idno":101,"name":"Ravi","salary":125000.00}

import pickle
file = open("contacts1.txt","wb")
pickle.dump(my_friends,file)
pickle.dump(contacts,file)
pickle.dump(nick_name,file)
pickle.dump(details,file)
file.close()
print("Data Written to File")
```

Example Script to Read list set, tuple and dictionary object's From a File

```
import pickle
file = open("contacts1.txt","rb")

my_friends = pickle.load(file)
print(my_friends)
print(type(my_friends))

cont = pickle.load(file)
print(cont)
print(type(cont))

nick = pickle.load(file)
print(nick)
print(type(nick))

det = pickle.load(file)
print(det)
print(type(det))

file.close()
print("Thanks")
```

Example Script to Write Employee object into a File

```
class Employee:
    comp_name = "Sathya"
    comp_cno = 9052492329
    def assign(self,id,na):
        self.idno = id
        self.name = na
    def display(self):
        print(Employee.comp_name)
        print(Employee.comp_cno)
        print(self.idno)
        print(self.name)

e1 = Employee()
e1.assign(101,"Ravi")
e1.display()

e2 = Employee()
e2.assign(102,"kumar")
e2.display()

import pickle
file = open("employee.txt","wb")
pickle.dump(e1,file)
pickle.dump(e2,file)
file.close()
print("Employee Objects written to File")
```

Example Script to Read a Employee Object From a File

```
class Employee:
    comp_name = "Sathya"
    comp_cno = 9052492329
    def assign(self,id,na):
        self.idno = id
        self.name = na
    def display(self):
        print(Employee.comp_name)
        print(Employee.comp_cno)
        print(self.idno)
        print(self.name)

import pickle
file = open("employee.txt","rb")

data = pickle.load(file)
data.display()
print("----------------")
data = pickle.load(file)
data.display()
```