Python Fundamentals day 55

Today's Agenda

- Polymorphism
- Duck Typing

Polymorphism

Polymorphism in python defines methods in the child class that have the same name as the methods in the parent class. In inheritance, the child class inherits the methods from the parent class. Also, it is possible to modify a method in a child class that it has inherited from the parent class.

Polymorphism establishes 1:many relation. This means certain line of code giving different output, or performing different task.

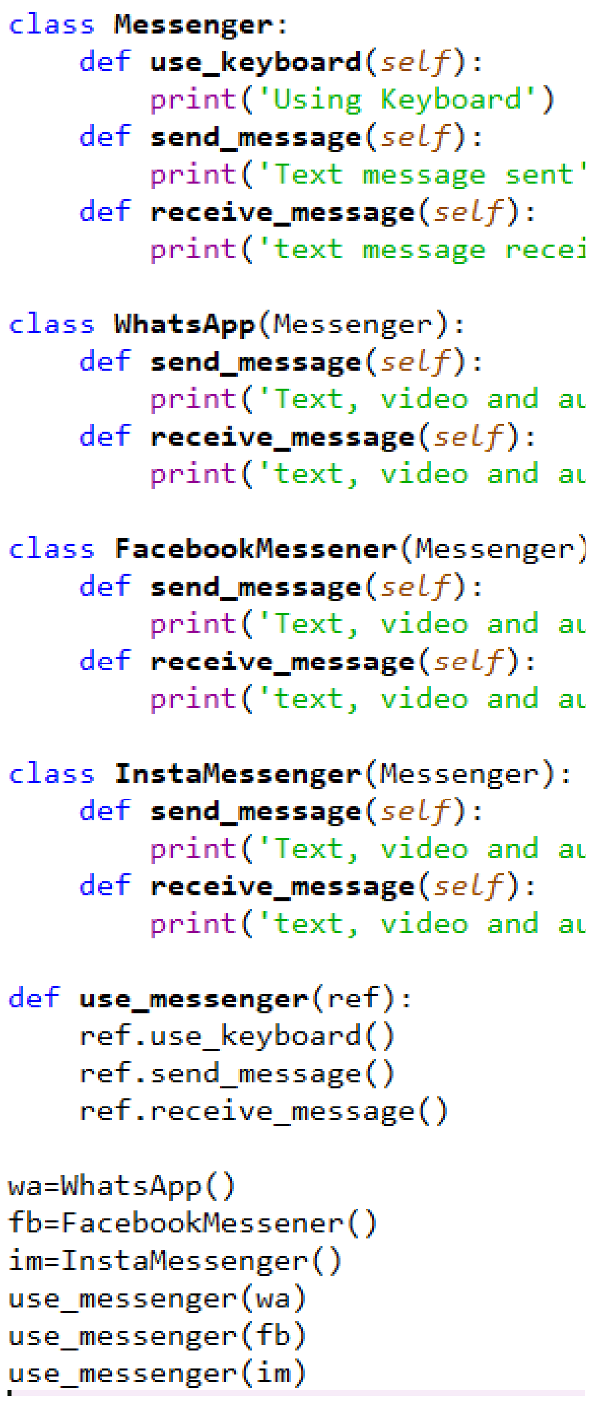
Let us look at the below example which is non-polymorphic and then convert the same into a code following polymorphism

```python
class Messenger:
    def use_keyboard(self):
        print('Using Keyboard')
    def send_message(self):
        print('Text message sent')
    def receive_message(self):
        print('text message received')

class WhatsApp(Messenger):
    def send_message(self):
        print('Text, video and audio message sent using WA')
    def receive_message(self):
        print('text, video and audio message received using WA')

class FacebookMessener(Messenger):
    def send_message(self):
        print('Text, video and audio message sent using FBM')
    def receive_message(self):
        print('text, video and audio message received using FBM')

class InstaMessenger(Messenger):
    def send_message(self):
        print('Text, video and audio message sent using Insta')
    def receive_message(self):
        print('text, video and audio message received using Insta')

wa=WhatsApp()
fb=FacebookMessener()
im=InstaMessenger()

wa.use_keyboard()
wa.send_message()
wa.receive_message()

fb.use_keyboard()
fb.send_message()
fb.receive_message()

im.use_keyboard()
im.send_message()
im.receive_message()
```

Output:

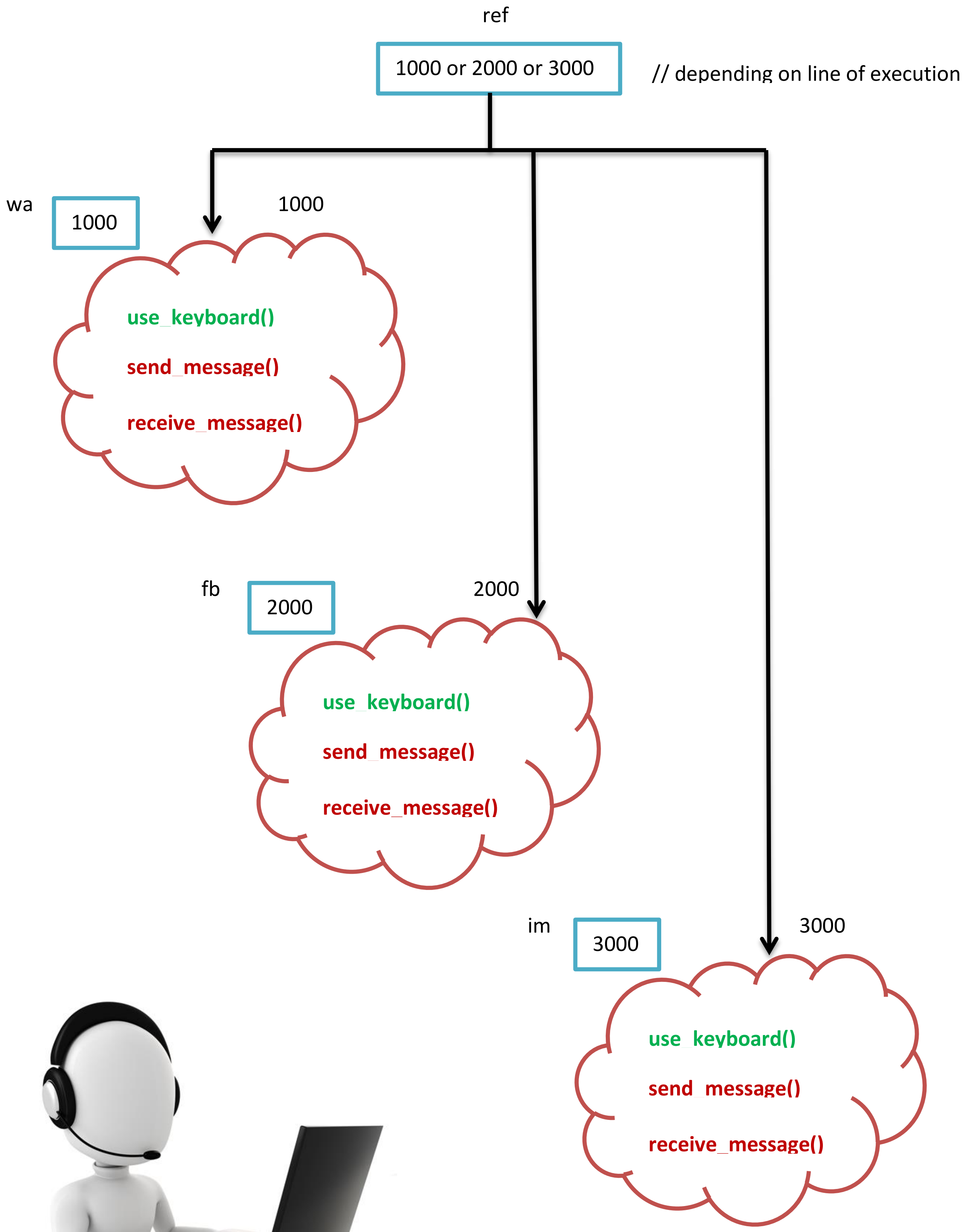
```
In [1]: runfile('C:/Users/rooman/OneDrive/Desktop/python/
test.py', wdir='C:/Users/rooman/OneDrive/Desktop/python')
Using Keyboard
Text, video and audio message sent using WA
text, video and audio message received using WA
Using Keyboard
Text, video and audio message sent using FBM
text, video and audio message received using FBM
Using Keyboard
Text, video and audio message sent using Insta
text, video and audio message received using Insta
```

Now let us write the code implementing polymorphism, to achieve code flexibility and code reusability.



```
class Messenger:
    def use_keyboard(self):
        print('Using Keyboard')
    def send_message(self):
        print('Text message sent'
    def receive_message(self):
        print('text message recei

class WhatsApp(Messenger):
    def send_message(self):
        print('Text, video and a
    def receive_message(self):
        print('text, video and a

class FacebookMessener(Messenger)
    def send_message(self):
        print('Text, video and a
    def receive_message(self):
        print('text, video and a

class InstaMessenger(Messenger):
    def send_message(self):
        print('Text, video and a
    def receive_message(self):
        print('text, video and a

def use_messenger(ref):
    ref.use_keyboard()
    ref.send_message()
    ref.receive_message()

wa=WhatsApp()
fb=FacebookMessener()
im=InstaMessenger()
use_messenger(wa)
use_messenger(fb)
use_messenger(im)
```

Let us trace before seeing the output



Output:

```
In [2]: runfile('C:/Users/rooman/OneDrive/Desktop/py
test.py', wdir='C:/Users/rooman/OneDrive/Desktop/pyt
Using Keyboard
Text, video and audio message sent using WA
text, video and audio message received using WA
Using Keyboard
Text, video and audio message sent using FBM
text, video and audio message received using FBM
Using Keyboard
Text, video and audio message sent using Insta
text, video and audio message received using Insta
```

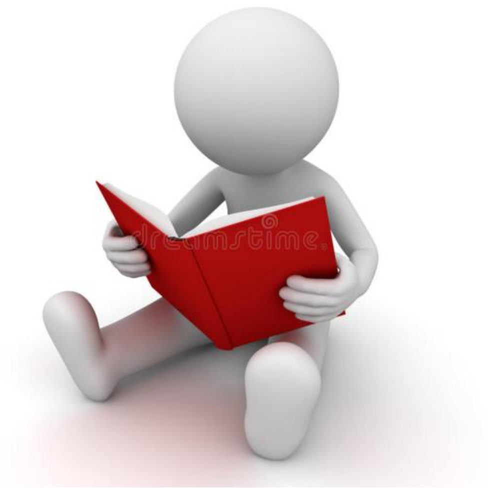
Great! But what if we want to include specialized methods? Duck typing is the solution. Didn't understand? Let us explore

Duck typing

In duck typing, you do not check types at all. Instead, you check for the presence of a given method or attribute. Like shown in the below example

```python
class Messenger:
    def use_keyboard(self):
        print('Using Keyboard')
    def send_message(self):
        print('Text message sent')
    def receive_message(self):
        print('text message received')

class WhatsApp(Messenger):
    def send_message(self):
        print('Text, video and audio message sent using WA')
    def receive_message(self):
        print('text, video and audio message received using WA')
    def send_live_location(self):
        print('Live Location sent using WA')

class FacebookMessener(Messenger):
    def send_message(self):
        print('Text, video and audio message sent using FBM')
    def receive_message(self):
        print('text, video and audio message received using FBM')
    def use_builtin_apps(self):
        print('using built-in apps using FBM')
```

```python
class InstaMessenger(Messenger):
    def send_message(self):
        print('Text, video and audio message sent using Insta')
    def receive_message(self):
        print('text, video and audio message received using Insta')
    def add_filters(self):
        print('Filters using insta')

def use_messenger(ref):
    ref.use_keyboard()
    ref.send_message()
    ref.receive_message()

    if type(ref)== WhatsApp:
        ref.send_live_location()
    if type(ref)==FacebookMessener:
        ref.use_builtin_apps()
    if type(ref)==InstaMessenger:
        ref.add_filters()

wa=WhatsApp()
fb=FacebookMessener()
im=InstaMessenger()

use_messenger(wa)
use_messenger(fb)
use_messenger(im)
```



Output:

```
In [3]: runfile('C:/Users/rooman/OneDrive/Desktop/python/
test.py', wdir='C:/Users/rooman/OneDrive/Desktop/python')
Using Keyboard
Text, video and audio message sent using WA
text, video and audio message received using WA
Live Location sent using WA
Using Keyboard
Text, video and audio message sent using FBM
text, video and audio message received using FBM
using built-in apps using FBM
Using Keyboard
Text, video and audio message sent using Insta
text, video and audio message received using Insta
Filters using insta
```

Great! So now we know how to make code flexible with specialized methods as well.