Classes in Python

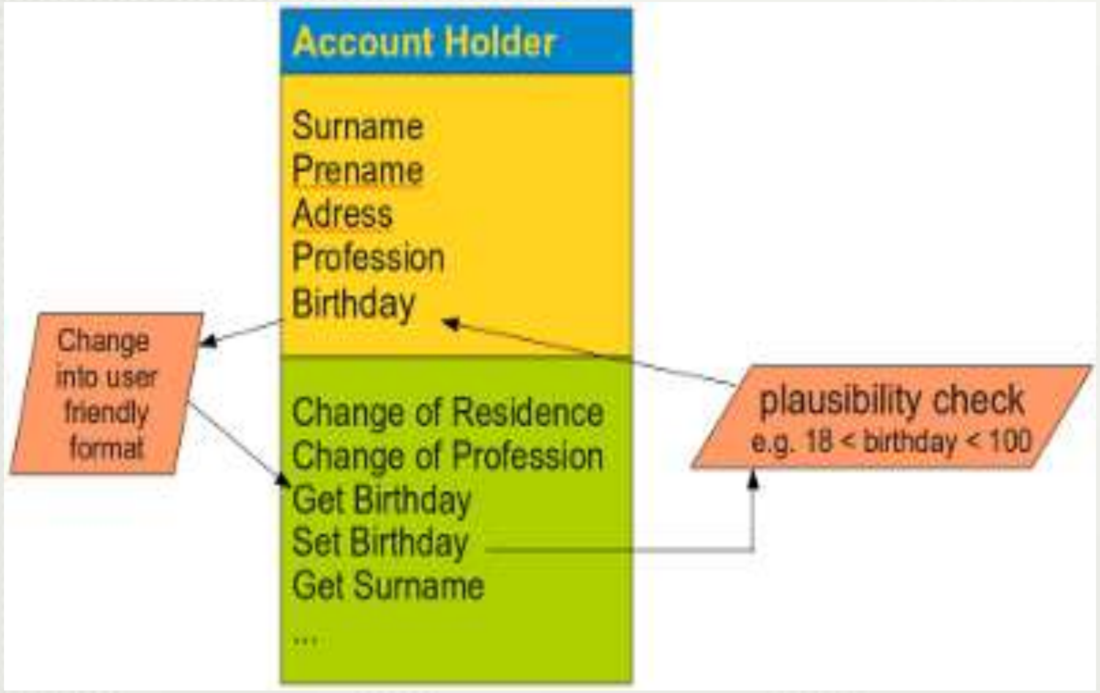
- This Session deals with
  - Encapsulation
  - Inheritance

Encapsulation

abstraction is achieved though encapsulation. Data hiding and encapsulation are the same concept

Encapsulation is the mechanism for restricting the access to some of an object's components,.

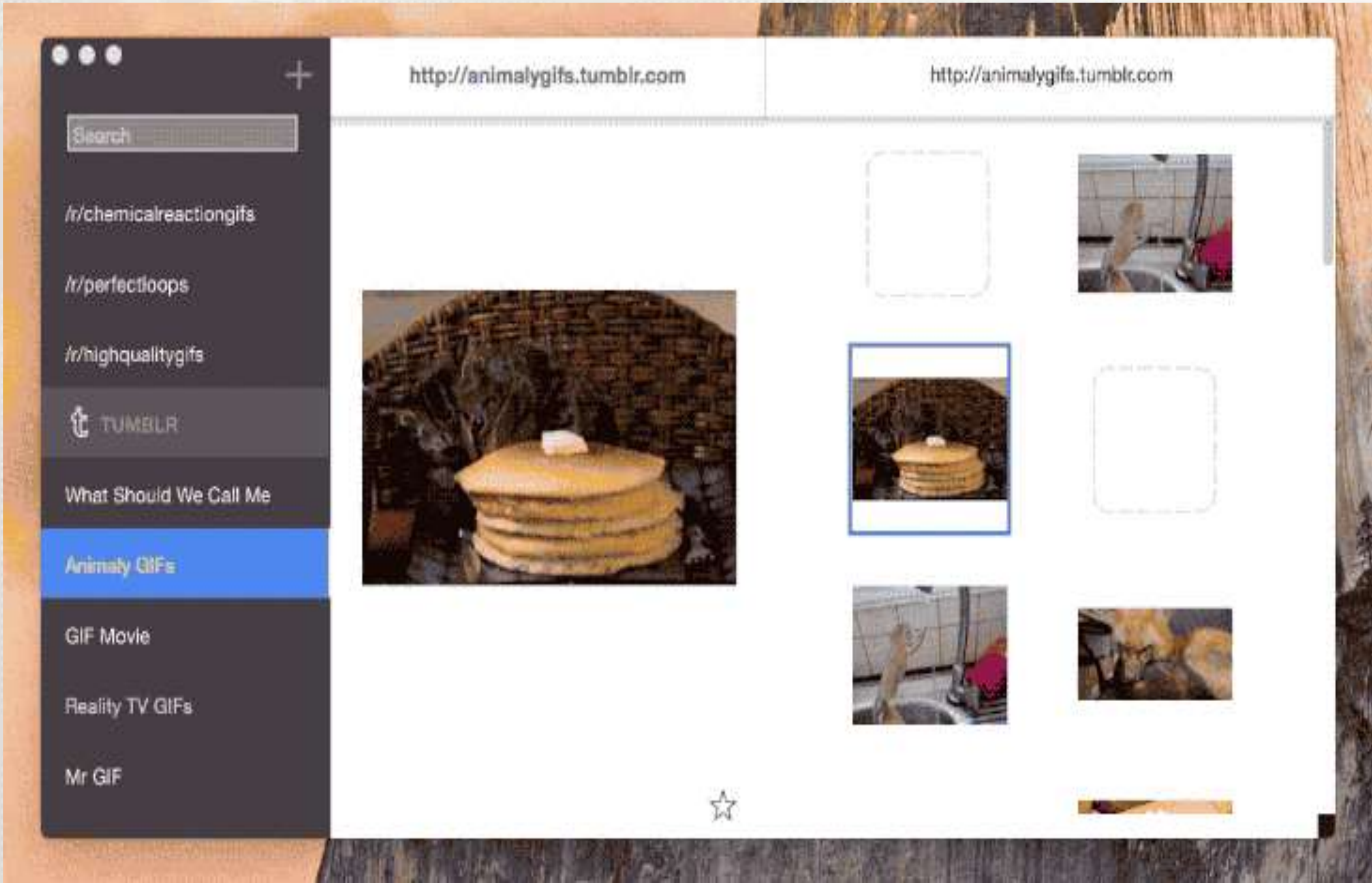
The internal representation of an object can't be seen from outside of the objects definition.



Kmit

TRISHUL

Encapsulation



Example

```
class Car:
   __maxspeed = 0
   __name = ""

   def __init__(self):
      self.__maxspeed = 200
      self.__name = "Supercar"

   def drive(self):
      print('driving. maxspeed ' + str(self.__maxspeed))

redcar = Car()
redcar.drive()
redcar.__maxspeed = 10  # will not change variable 
because its private
redcar.drive()
```

Output

```
driving. maxspeed 200
driving. maxspeed 200
```

Inheritance

Classes can inherit other classes.

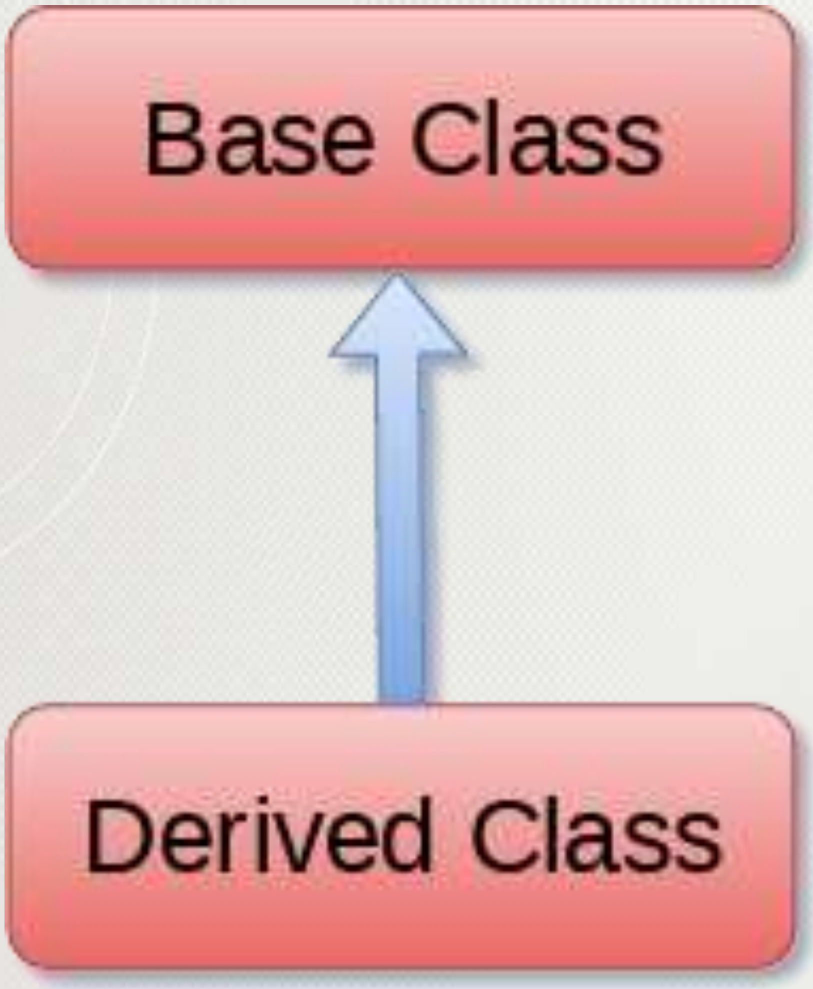
A class can inherit attributes and behaviour (methods) from other classes, called super-classes.

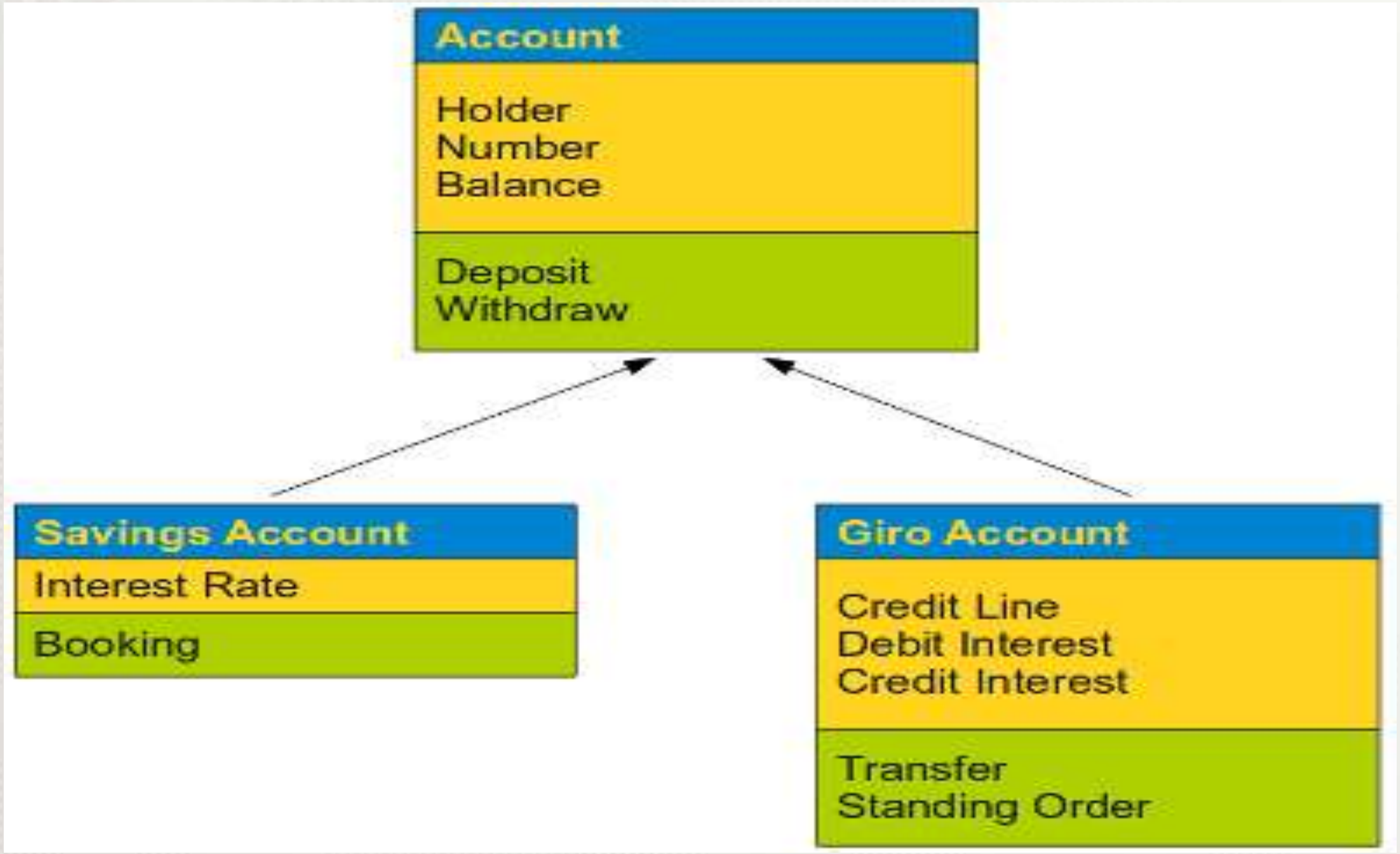
A class which inherits from super-classes is called a Sub-class.

Super-classes are sometimes called ancestors as well.

There exists a hierarchy relationship between classes.

Inheritance provides code reusability to the program


Base Class
Derived Class


Account
Holder
Number
Balance
Deposit
Withdraw
Savings Account
Interest Rate
Booking
Giro Account
Credit Line
Debit Interest
Credit Interest
Transfer
Standing Order

Exercise

```
class Person:
    def __init__(self, first, last):
        self.firstname = first
        self.lastname = last

    def Name(self):
        return self.firstname + " " + self.lastname
class Employee(Person):
    def __init__(self, first, last, staffnum):
        Person.__init__(self,first, last)
        self.staffnumber = staffnum

    def GetEmployee(self):
        return self.Name() + ", " +  self.staffnumber
x = Person("Sree", "Ram")
y = Employee("Ram", "Laxman", "1007")
print(x.Name())
print(y.GetEmployee())
```

Output

```
Sree Ram
Ram Laxman, 1007
```

Exercise-2

- Create a class Animal with speak method,create a class Dog with bark() method and inherit the method into the Dog class,create a class DogChild with eat method and inherit the method into DogChild class.
- Create a object for DogChild class and invoke all the methods.

Kmit TRISHUL

Exercise-2

python

```
class Animal:
    def speak(self):
        print("Animal speaking")
class Dog(Animal):
    def bark(self):
        print("dog barking")
class DogChild(Dog):
    def eat(self):
        print("Eating bread..")
d=DogChild()
d.speak()
d.bark()
d.eat()
```

Output

```
Animal speaking
dog barking
Eating bread..
```