нетология

Объекты и классы
Инкапсуляция, наследование и полиморфизм

Олег
Булыгин

Олег Булыгин

IT-аудитор в ПАО Сбербанк

План занятия

1. Инкапсуляция
2. Наследование
3. Полиморфизм
4. Домашнее задание

Инкапсуляция

— это свойство системы, позволяющее объединить данные и методы, работающие с ними в классе, и скрыть детали реализации.

Как выглядит ваша программа

Скрываем всё

Стив Джобс был одним из первых, кто применил прием инкапсуляции к ПК, перекрыв доступ к внутреннему устройству компьютера обычному пользователю.

И такое решение небезосновательно.

С помощью корпуса мы обеспечиваем безопасность внутреннего содержимого ПК (отсутствие доступа к платам, проводам), но при этом мы можем взаимодействовать с ним, производя проверку корректности данных (порты для подключения наушников, HDMI, но не можем подключить к нему вилку). При этом, у нас остается возможность изменять внутренние характеристики ноутбука: объем памяти, процессор и т.д.

Модификаторы доступа

Модификаторы доступа в Python используются для модификации области видимости переменных по умолчанию.

Есть три типа модификаторов доступов в Python ООП:

- public,
- __private,
- _protected.

Доступ к переменным с модификаторами публичного доступа открыт из любой точки вне класса, доступ к приватным переменным открыт только внутри самого класса, и в случае с защищенными переменными, доступ открыт только внутри класса и дочерних классов.

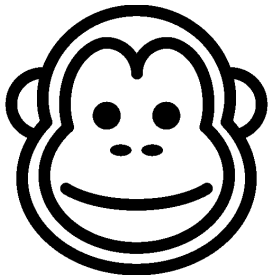
Но работают они только на словах.

Наследование

— это свойство системы, позволяющее описать новый класс на основе уже существующего с частично или полностью заимствованной функциональностью.

Primate

class Primate

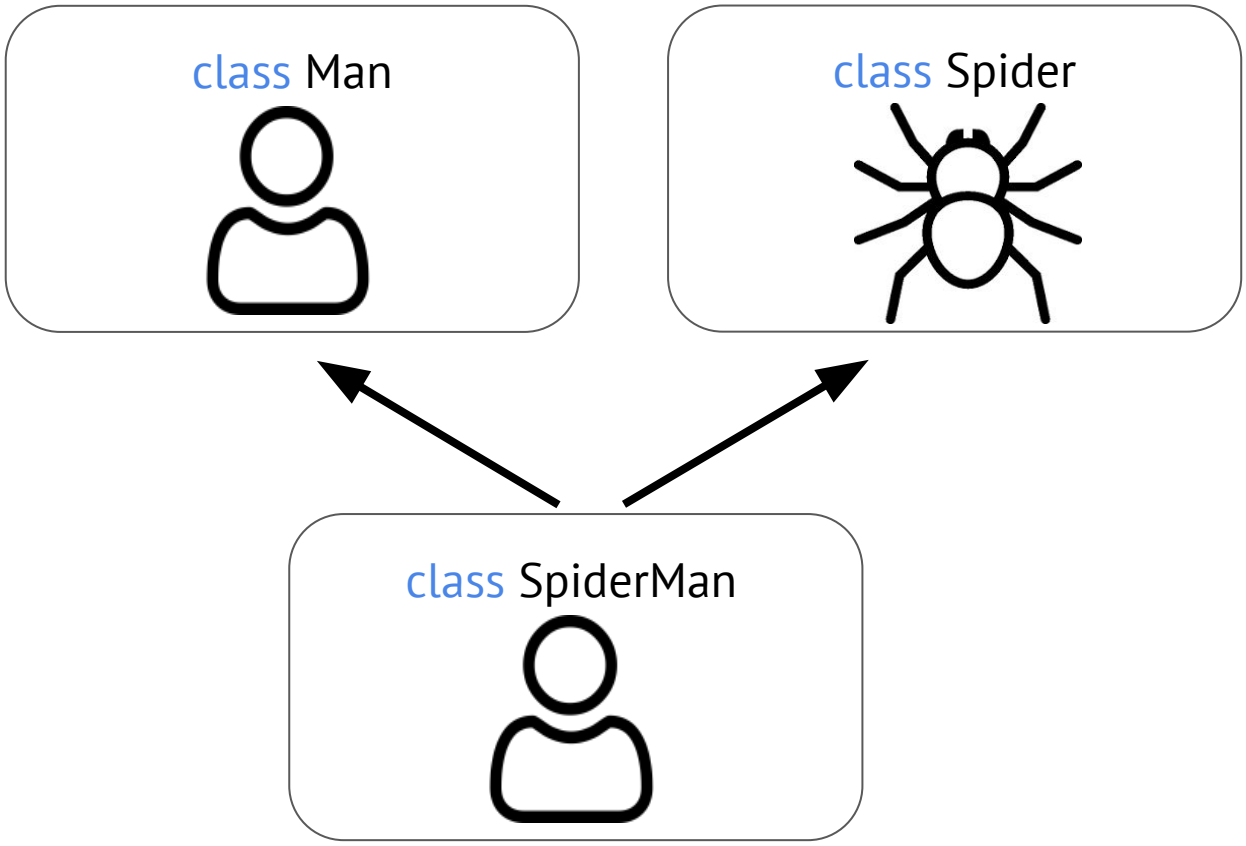


class Man

```
class Primate:

    def eat(self, food):
        …

    def run(self, time):
        …
```

```
class Man(Primate):
    name = …
    last_name = …

    def build(self, something):
        …
```

Множественное наследование



Что наследовать?

MRO (Method Resolution Order) — порядок, в котором Python ищет метод в иерархии классов.

```
class Primate:
    …

class Man(Primate):
    …

class Spider:
    …

class SpiderMan(Spider, Man):
    …

SpiderMan.mro()
```

Полиморфизм

— это свойство системы использовать объекты с одинаковым интерфейсом без информации о типе и внутренней структуре объекта.

Перегрузка и переопределение методов

Заставляем методы работать по-разному в зависимости от наличия параметров или исходя из того, из какого класса мы их вызываем.

```
class Primate:

    def eat(self, food):
        print(food)

class Man(Primate):

    def eat(self, food,
cooked=False):
        if cooked:
            print('cooked', food)
        else:
            print(food)
```

«Магия»

Откуда Python знает, какие действия нужно выполнить?

Инициализация и Конструирование:

- __new__(cls, other)
- __init__(self, other)
- __del__(self)

Операторы:

- __add__(self, other)
- __div__(self, other)
- __lt__(self, other)

И т.д.

Домашнее задание

Давайте посмотрим ваше домашнее задание.

- Вопросы по домашней работе задавайте **в чате** мессенджера .
- Задачи можно сдавать **по частям**.
- Зачёт по домашней работе проставляется после того, как **приняты все задачи**.

нетология

Задавайте вопросы и пишите отзыв о лекции!

Олег Булыгин