https://dfedorov.spb.ru

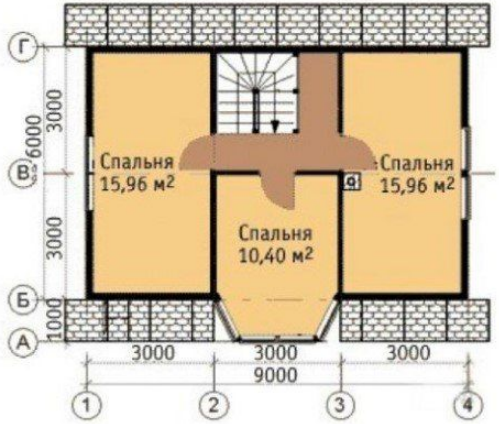
Объектно-ориентированный стиль программирования

Первым языком программирования, широко известным как «объектно-ориентированный», был язык **Simula**, спецификации которого были разработаны в 1965 году. Хабр

Алан Кэй придумал термин «объектно-ориентированное программирование», имея в виду язык программирования **Smalltalk** (1972). Этот язык разработали Алан Кэй, Дэн Инглз и другие сотрудники научно-исследовательского центра Xerox PARC в рамках проекта по созданию устройства Dynabook. Язык Smalltalk был более объектно-ориентированным, чем Simula. В Smalltalk всё является объектом, включая классы, целые числа и блоки (замыкания). Первоначальная реализация языка, Smalltalk-72, не имела возможностей создания подклассов. Эта возможность появилась в Smalltalk-76.

План 1-го этажа

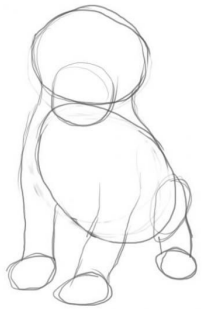
План 2-го этажа



Собака из
реального мира

Класс собаки

```
class Dog:
    pass
```

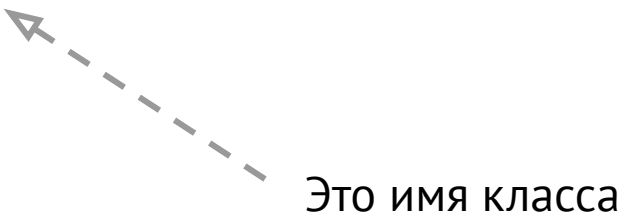


Экземпляр (объект) собаки

```
dog1 = Dog()
```

```
dog2 = Dog()
```

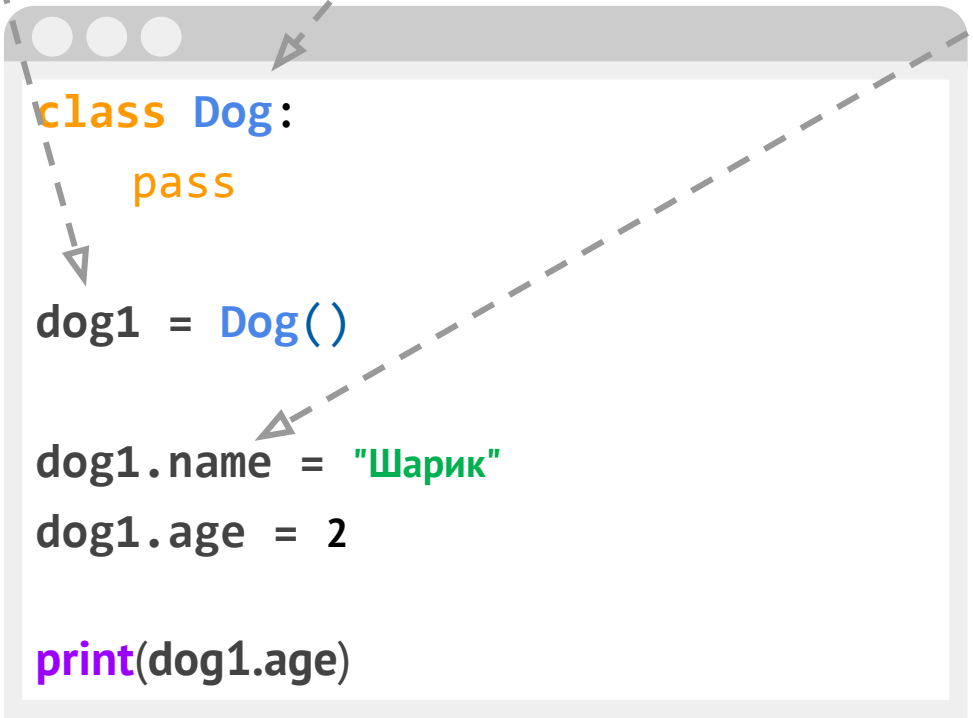
```
# Создали объект (экземпляр) типа (класса) int
a = int( )
```



Ссылка на объект (хранит адрес объекта типа Dog)

Имя класса (типа)

Поля (атрибуты) объекта, аналогия - СЛОВАРЬ



Состояние и поведение объектов

Метод класса Dog

```
class Dog:
    def bark(self):
        print("Woof")
```

Первый параметр всегда должен быть self

```
myDog = Dog()

myDog.name = "Spot"
myDog.weight = 20
myDog.age = 3
myDog.bark()
```

Вызов метода bark

Преобразуется в Dog.bark(myDog)

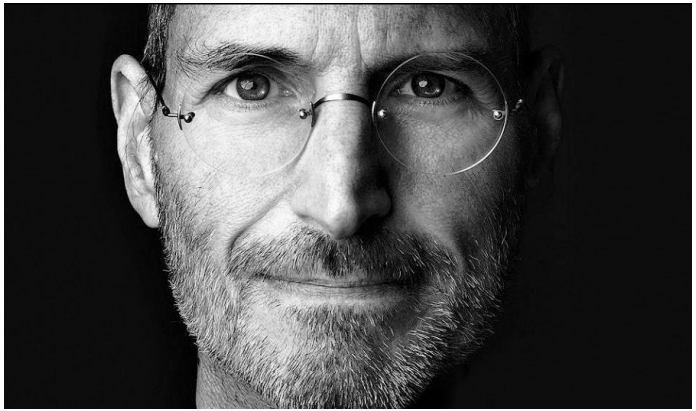
```
def bark(self):
    print("Woof says", self.name)
```

Классы в Python имеют особую функцию - **инициализатор**, вызываемую в момент создания экземпляра класса.

```
class Dog:
    # Вызывается в момент создания объекта этого типа
    def __init__(self):
        self.name = ""
        print("Родилась новая собака!")

# Создаем собаку (объект myDog класса Dog)
myDog = Dog()
```

В 1994 году Стив Джобс давал интервью журналу The Rolling Stone.
Джефф Гуделл: Объясните, пожалуйста, что такое объектно-ориентированный подход, используя простые термины.
Стив Джобс: Объекты похожи на людей. Они живые, дышащие сущности, которые обладают памятью и знаниями о том, как нужно действовать. И вместо того, чтобы взаимодействовать с ними на низком уровне, мы взаимодействуем с ними на очень высоком уровне абстракции.



Например, если для вас я объект-прачка, вы можете передать мне грязную одежду вместе с сообщением «постирай, пожалуйста, мою одежду». Я знаю, где в Сан-Франциско лучшая прачечная. Я говорю по-английски, в карманах у меня доллары. Я выхожу на улицу, ловлю такси и говорю таксисту, чтобы он отвёз меня в нужное место. Я стираю вещи, снова сажусь в такси и возвращаюсь. Я отдаю вам чистую одежду с сообщением: «вот ваша чистая одежда».
Вы понятия не имеете, как и какие шаги я проделал. Вы не знаете о том, что существует прачечная. Возможно, вы говорите по-французски и вы не можете поймать такси — вам нечем за него заплатить, долларов у вас нет. Тем не менее, я знал, как всё это сделать. А вам нет необходимости это знать. Вся комплексность была во мне запрятана, и мы смогли взаимодействовать на высоком уровне абстракции. В этом суть объектов — они сочетают в себе инкапсуляцию структурной сложности и высокоуровневый интерфейс.

Отсюда

coin_demo1.py

coin_demo2.py

coin_demo3.py

coin.py

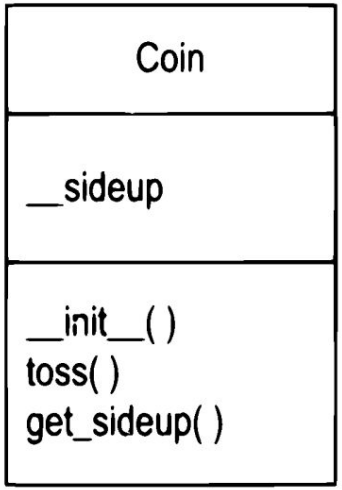
coin_demo4.py

bankaccount.py

account_test.py

bankaccount2.py

account_test2.py



Разбираемся с атрибутами объекта/класса:

class_car1.py

class_car2.py

class_car3.py

https://dfedorov.spb.ru