Python-разработчик

Множественное наследование

skypro

Сегодня на уроке

План урока

1. Рассмотрим, что такое абстрактные классы.
2. Узнаем, как наследоваться от нескольких классов и что такое MRO.
3. Познакомимся с коллекцией `__slots__`.

Абстрактный класс

Абстрактный класс —
класс, который содержит один или несколько абстрактных методов.

определение

```
from abc import ABC

class Employee(ABC):
    pass
```

Запрещает пользователю создавать экземпляры этого класса.

Абстрактный метод

Абстрактный метод — метод, который имеет объявление, но не имеет реализации.

определение

```
from abc import ABC, abstractmethod

class Employee(ABC):

    @abstractmethod
    def work(self):
        pass
```

Вынуждает пользователя переопределять себя в дочернем классе.

Задача

Создать абстрактный класс Employee
и два дочерних класса Developer и Accountant.

Задача

Создание абстрактного класса.

Флоу решения:

1. Импортировать модуль `abc`
2. Создать класс `Employee(ABC)`
3. Создать в `Employee` методы
   с декоратором `@abstractmethod`

Множественное наследование

Множественное наследование — возможность класса иметь более одного родительского класса.

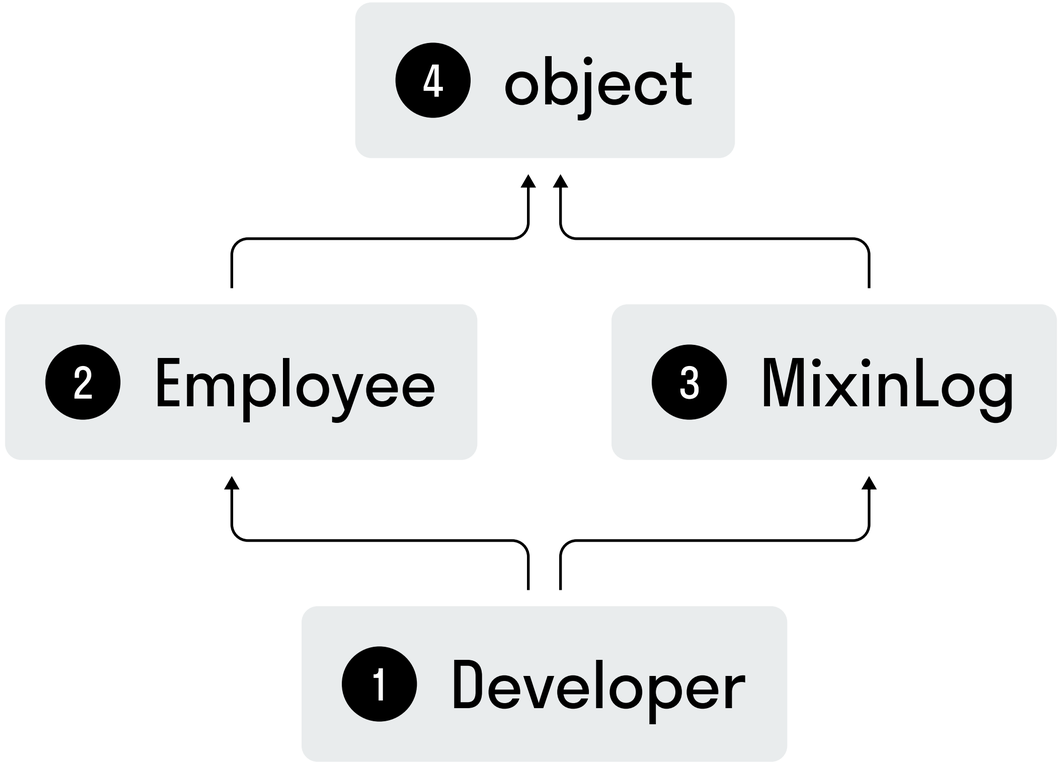
определение

```
from abc import ABC


class Employee(ABC):
    pass


class SalaryMixin:
    pass


class Develop(SalaryMixin, Employee):
    pass
```

MRO

MRO (от англ. Method Resolution Order) — алгоритм, который определяет порядок обхода классов при поиске методов и атрибутов в иерархии наследования.

определение



Порядок наследования можно посмотреть в коллекции `__mro__`:

```
DeveloperClass.__mro__
```

Developer → Employee → MixinLog → object

Миксины

Миксины (от англ. mixins) — классы, которые предназначены для использования в качестве дополнительного функционала в других классах.

определение

```
class MixinLog:
    ID = 1

    def __init__(self):
        self.id = self.ID
        MixinLog.ID += 1

    def order_log(self):
        print(f'{self.id}-й сотрудник')
```

Задача

Добавить возможность логирования порядка найма разработчиков.

Задача

Логирование.

Флоу решения:

1. Создать класс-миксин
2. Добавить его в список родителей `Developer`
3. Добавить `super().__init__` в `Employee`

Коллекции __slots__

Коллекция `__slots__` перечисляет, какие свойства могут быть у экземпляров класса.

Наследуется, только если у дочернего класса тоже прописаны `__slots__`.

Преимущества использования `__slots__`:

- Ограничение создаваемых локальных свойств.
- Уменьшение занимаемой памяти.
- Ускорение работы с локальными свойствами.

```
class PointSlots:
    __slots__ = ('x', 'y')
    def __init__(self, x, y):
        self.x = x
        self.y = y
```

Задача

Создать два класса точки на плоскости,
замерить скорость обращения к атрибутам
с и без использования коллекции `__slots__`.

Задача

Точки на плоскости.

Флоу решения:

1. Создать класс
2. Прописать в кортеже `__slots__`
3. Замерить время через `timeit.timeit(pts.get_set_del)`

Подведем итоги

Итоги урока

1. С помощью абстрактных классов и методов можно определять набор обязательных методов для реализации в дочерних классах.

Итоги урока

1. С помощью абстрактных классов и методов можно определять набор обязательных методов для реализации в дочерних классах.
2. Миксины — примеси функциональности, которые можно добавлять к любому классу, который наследуется от другого класса.

Итоги урока

1. С помощью абстрактных классов и методов можно определять набор обязательных методов для реализации в дочерних классах.
2. Миксины — примеси функциональности, которые можно добавлять к любому классу, который наследуется от другого класса.
3. Множественное наследование добавляет к классам новые фунциональности.

Итоги урока

1. С помощью абстрактных классов и методов можно определять набор обязательных методов для реализации в дочерних классах.
2. Миксины — примеси функциональности, которые можно добавлять к любому классу, который наследуется от другого класса.
3. Множественное наследование добавляет к классам новые фунциональности.
4. Порядок поиска методов можно посмотреть в атрибуте `__mro__`.

Итоги урока

1. С помощью абстрактных классов и методов можно определять набор обязательных методов для реализации в дочерних классах.
2. Миксины — примеси функциональности, которые можно добавлять к любому классу, который наследуется от другого класса.
3. Множественное наследование добавляет к классам новые фунциональности.
4. Порядок поиска методов можно посмотреть в атрибуте `__mro__`.
5. `__slots__` предназначен для фиксации разрешенных атрибутов и сокращает использование ресурсов для обработки объектов.

Спасибо!

skypro