Laporan Praktikum PBO

Disusun oleh:

Muhammad Widyantoro Wiryawan | 121140183

Program Studi Teknik Informatika

Institut Teknologi Sumatera

Lampung Selatan

2023

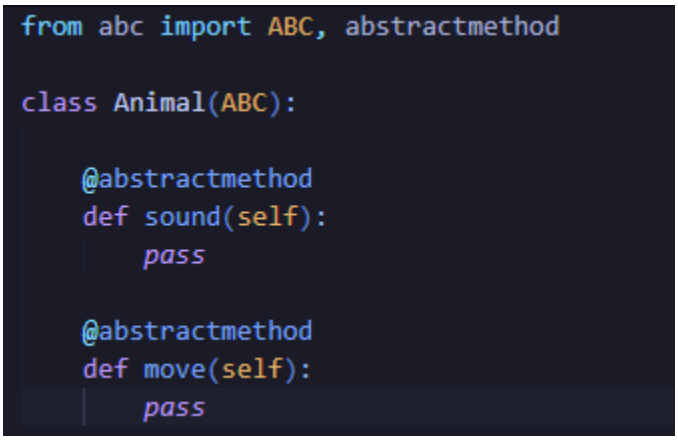
Ringkasan

1. **Kelas abstrak**
   Kelas abstrak dapat dianggap sebagai kumpulan method untuk kelas lain. kelas tersebut menentukan sekumpulan metode yang harus diimplementasikan di child kelas mana pun yang berasal dari kelas abstrak. Kelas abstrak dapat berisi satu atau lebih metode abstrak. Metode yang memiliki deklarasi tetapi tidak ada implementasi disebut sebagai abstrak. Seperti pada gambar berikut.

```
class Aset(ABC):
    def __init__(self, x, y, width, image):
        #enkapsulasi (protected)
        self._x = x
        self._y = y
        self._width = width
        self._image = image
        #

    #abstraksi
    @abstractmethod
    def draw(self, game_display):
        pass
    #

    @abstractmethod
    def get_rect(self):
        pass
    #

    def move(self):
        pass
```

   Berdasarkan kode tersebut kelas turunan dari kelas abstrak harus menerapkan semua method abstrak yang telah dideklarasikan pada kelas abstrak. Tujuan dari kelas abstrak yaitu memberikan sebuah konsep dasar yang harus diikuti oleh kelas turunannya.

2. **Interface**
   Interface adalah sebuah method dan properti yang ada pada kelas. Interface dapat dikatakan sebagai perilaku yang menginplemetasikan method pada sebuah objek.



```
from abc import ABC, abstractmethod

class Animal(ABC):

    @abstractmethod
    def sound(self):
        pass

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

```python
class Dog(Animal):

    def sound(self):
        return "Woof"

    def move(self):
        return "Walking"

# Membuat objek dari kelas Dog
dog = Dog()
print("Sound Anjing:", dog.sound())
print("Cara Bergerak Anjing:", dog.move())
```

3. **Metaclass**
   Metaclass kelas yang mendefinisikan bagaimana kelas berperilaku. Kelas itu sendiri merupakan turunan dari metaclass. Kelas dalam Python menentukan bagaimana instance kelas berperilaku.

```python
class Parent(type):
    def __new__(cls, name, bases, bts):
        # Modifikasi perilaku pembuatan kelas
        bts['isi'] = 'Ini adalah metaclass.'

        # Membuat dan mengembalikan kelas baru
        return super().__new__(cls, name, bases, bts)

# Membuat kelas menggunakan metaclass kustom
class Meta(metaclass=Parent):
    pass

# Mencetak atribut dari kelas yang dibuat
print(Meta.isi)  # Output: Ini adalah metaclass.
```

Kesimpulan

Interface adalah sebuah kontrak yang mendefinisikan metode-metode yang harus ada dalam suatu kelas atau objek. Interface tidak memberikan implementasi konkret dari metode-metode tersebut, tetapi hanya mendefinisikan metode dengan nama, parameter, dan tipe kembalian. Interface digunakan ketika kita ingin menggunakan konsep polimorfisme. Dengan begitu, memungkinkan kita menulis kode yang dapat digunakan Kembali secara flesibel.

Kelas abstrak adalah kelas yang tidak dapat langsung dipakai tetapi berfungsi sebagai kerangka kerja untuk kelas turunannya. Kelas abstrak mendefinisikan metode yang harus diterapkan oleh kelas turunannya. Ketika kita ingin mendefinisikan framework umum yang harus diikuti oleh kelas turunan, kita harus menggunakan kelas abstrak. Perbedaan utama antara kelas abstrak dan interface adalah bahwa kelas abstrak dapat memiliki implementasi konkret untuk banyak metode, sedangkan interface hanya mendefinisikan tanda tangan metode tanpa implementasi konkret.

Kelas konkret adalah kelas yang dapat langsung dipakai dan memiliki implementasi konkret dari metode yang didefinisikan di dalamnya. Ketika kita ingin membuat objek yang dapat digunakan secara langsung dan telah menerapkan perilaku tertentu, kita harus menggunakan kelas konkret. Kelas konkret menyediakan implementasi konkret untuk metode yang didefinisikan dalam kelas tersebut, hal ini memungkinkan objek yang dibuat dari kelas itu untuk secara langsung melakukan tugas tertentu.

Metaclass adalah kelas yang digunakan untuk membuat dan mengontrol perilaku kelas lain (kelas). Metaclass memungkinkan kita untuk mengubah dan mengelola cara kelas dibuat dan dijalankan. Saat kita ingin mengubah perilaku instance, menambah atau mengubah metode di kelas, atau mengimplementasikan fitur tambahan, kita harus menggunakan metaclass. Perbedaan utama antara metaclass dan inheritance biasa adalah bahwa metaclass beroperasi pada tingkat pembuatan kelas, sedangkan inheritance beroperasi pada tingkat objek.

Daftar Pustaka

bestharadhakrishna. (2021, May 19). *Abstract Classes in Python*. Retrieved from https://www.geeksforgeeks.org/: https://www.geeksforgeeks.org/abstract-classes-in-python/

Choudhari, P. (2022, Februari 7). *What are metaclasses in Python*. Retrieved from https://www.python-engineer.com/: https://www.python-engineer.com/posts/metaclasses-python/

Metaclasses, P. M. (2018). *Pengantar Metaclass PythonIntroduction to Python Metaclasses*. Retrieved from https://www.datacamp.com/: https://www.datacamp.com/tutorial/python-metaclasses#rdl

N, A. (2021, Oktober 17). *How to Use Abstract Classes in Python*. Retrieved from https://towardsdatascience.com/: https://towardsdatascience.com/how-to-use-abstract-classes-in-python-d4d2ddc02e90

naina024. (2020, Maret 26). *Python-interface module*. Retrieved from https://www.geeksforgeeks.org/: https://www.geeksforgeeks.org/python-interface-module/