LAPORAN PRAKTIKUM PEMROGRAMAN BERORIENTASI OBJEK (PBO)

PRAKTIKUM 5

2411102441250

Malik Sabarullah Akbar

FAKULTAS SAINS DAN TEKNOLOGI

PROGRAM STUDI S1 TEKNIK INFORMATIKA

UNIVERSITAS MUHAMMADIYAH KALIMANTAN TIMUR

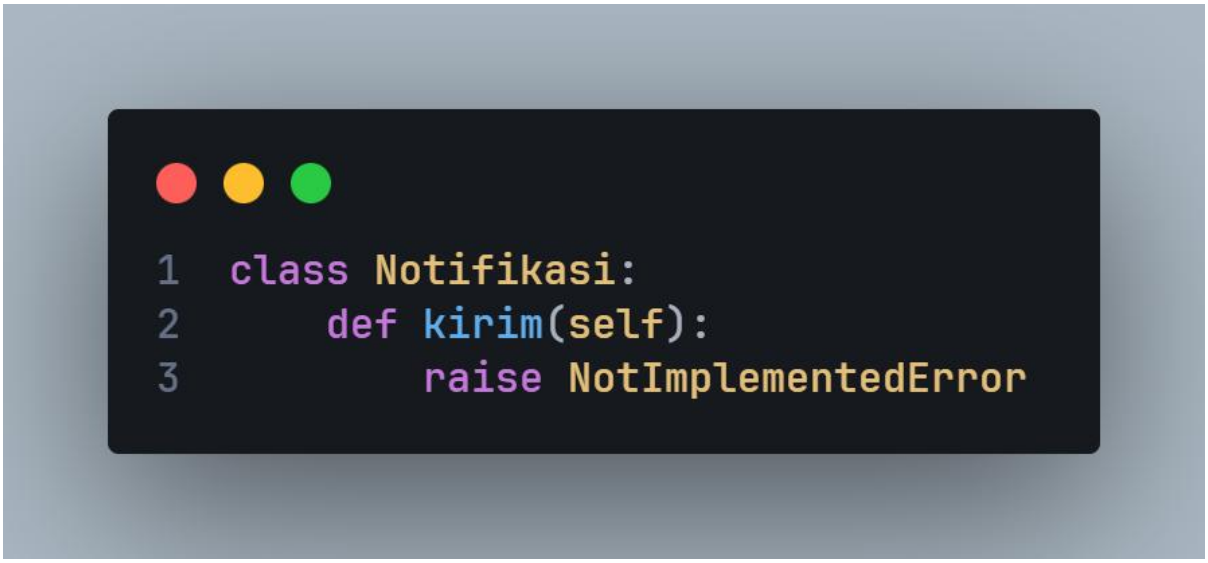
EMAIL : itsmalik57@gmail.com

Polymorphism

Studi Kasus - Sistem Notifikasi Multi-Channel

Anda diminta untuk merancang sebuah sistem yang bisa mengirim notifikasi melalui berbagai kanal (Email, SMS, Push Notification). Gunakan polymorphism agar sistem mudah dikembangkan jika ada kanal baru di masa depan.

1. Buat sebuah Parent Class abstrak bernama Notifikasi di file latihan_5.py.
    a. Class ini memiliki satu method bernama kirim(pesan) yang melempar NotImplementedError.



2. Buat tiga Child Classes yang mewarisi dari Notifikasi: Email, SMS, dan PushNotif.
    a. Setiap class anak harus melakukan override pada method kirim(pesan).
    b. Implementasi kirim(pesan) di setiap class harus mencetak pesan yang spesifik
    b. sesuai kanalnya.
        - Email. Mencetak "[EMAIL] Mengirim: '[isi pesan]'"
        - SMS. Mencetak "[SMS] Mengirim: '[isi pesan]'"
        - PushNotif. Mencetak "[PUSH] Mengirim: '[isi pesan]'"

```python
class Email(Notifikasi):
    def kirim(self, pesan):
        return f"[Email] Mengirim : {pesan}"


class SMS(Notifikasi):
    def kirim(self, pesan):
        return f"[SMS] Mengirim : {pesan}"


class PushNotification(Notifikasi):
    def kirim(self, pesan):
        return f"[Push Notification] Mengirim : {pesan}"
```

3. Di bagian utama program Anda:

a. Buat sebuah list yang berisi setidaknya satu objek dari setiap class anak (Email, SMS, PushNotif).

```python
daftar_pesan = [Email(), SMS(), PushNotification()]
```

b. Tulis sebuah loop for yang mengiterasi list tersebut.
c. Di dalam loop, panggil method kirim() pada setiap objek notifikasi dengan pesan yang sama, misalnya "Diskon Spesial! Hanya untuk Anda!".

```
pesan = "Diskon spesial! hanya untuk Anda."

for i in daftar_pesan:
    print(i.kirim(pesan))
```

d. Amati bagaimana satu pemanggilan method menghasilkan tiga output yang berbeda, menunjukkan polymorphism bekerja dengan sempurna.

```
[Running] python -u "c:\Users\itsma\Documents\Tugas Kuliah\Informatika\PRATIKUM\Semester 3\OOP\P5\latihan_5.py"
[Email] Mengirim : Diskon spesial! hanya untuk Anda.
[SMS] Mengirim : Diskon spesial! hanya untuk Anda.
[Push Notification] Mengirim : Diskon spesial! hanya untuk Anda.

[Done] exited with code=0 in 0.118 seconds
```

Kesimpulan

Desain polimorfik ini membuat kode jauh lebih fleksibel dan mudah dikelola karena kita dapat menambahkan tipe notifikasi baru (seperti SMS,Email,dan PushNotif.) tanpa mengubah satu baris pun kode yang sudah ada untuk mengirim pesan. Sebaliknya, menggunakan percabangan if-elif-else akan memaksa kita untuk terus-menerus memodifikasi blok kondisional setiap kali ada tipe notifikasi baru, sehingga kode menjadi kaku dan sulit dipelihara seiring waktu.