RAKTIKUM KONSTRUKSI PERANGKAT LUNAK
TUGAS JURNAL 13

Design Pattern Implementation

Telkom University

disusun Oleh:
Nita Fitrotul Mar’ah
2211104005
SE0601

Dosen Pengampu :
Yudha Islami Sulistya, S.Kom., M.Cs.

S1 REKAYASA PERANGKAT LUNAK
FAKULTAS INFORMATIKA
TELKOM UNIVERSITY PURWOKERTO
2025

1. **MEMBUAT PROJECT GUI BARU**

   Buka IDE misalnya dengan Visual Studio

   a. Misalnya menggunakan Visual Studio, buatlah project baru dengan nama modul113_NIM
   b. Project yang dibuat bisa berupa console atau sejenisnya

2. **MENJELASKAN SALAH SATU DESIGN PATTERN**

   Buka halaman web https://refactoring.guru/design-patterns/catalog kemudian baca design pattern dengan nama "Observer", dan jawab pertanyaan berikut ini (dalam Bahasa Indonesia):

   a. Berikan salah DUA contoh kondisi dimana design pattern "Singleton" dapat digunakan?

      Jawab:

      - Penyimpanan Konfigurasi Aplikasi
        Singleton cocok digunakan untuk menyimpan data konfigurasi yang hanya perlu dimuat satu kali dan dapat diakses di seluruh bagian aplikasi. Contohnya seperti pengaturan koneksi ke database atau pengaturan API yang bersifat global.
      - Sistem Pencatatan Log (Logging)
        Untuk mencatat aktivitas aplikasi secara terpusat, Singleton digunakan agar hanya ada satu objek logger aktif yang menangani semua log. Hal ini mempermudah pelacakan kesalahan dan debugging.

   b. Berikan penjelasan singkat mengenai langkah-langkah dalam mengimplementasikan design pattern "Singleton" .

      Jawab:

      - Buat konstruktor sebagai private
        Agar objek tidak dapat dibuat dari luar class, konstruktor harus bersifat privat.
      - Deklarasikan variabel statis untuk instance
        Variabel ini digunakan untuk menyimpan satu-satunya objek Singleton yang dibuat.
      - Buat metode statis untuk akses instance
        Metode ini akan mengembalikan instance yang sudah ada, atau membuat yang baru jika belum tersedia.
      - (Opsional) Batasi pewarisan class
        Di beberapa bahasa pemrograman, class Singleton bisa ditandai sebagai final/sealed untuk mencegah diturunkan oleh class lain.

   c. Berikan kelebihan dan kekurangan dari design pattern "Singleton"

      Jawab:

      Kelebihan:

- Penggunaan memori lebih hemat, karena hanya ada satu objek yang dibuat.
- Akses global lebih mudah, objek bisa dipanggil dari mana saja dalam sistem.
- Konsistensi data terjaga, karena semua bagian aplikasi mengakses instance yang sama.

Kekurangan:

- Sulit dalam pengujian, karena instance bersifat global dan tidak mudah diganti atau direset dalam pengujian.
- Potensi hambatan kinerja (bottleneck), jika banyak proses mengakses Singleton secara bersamaan.
- Kurang fleksibel, terutama jika suatu saat dibutuhkan lebih dari satu objek dari class tersebut.

3. IMPLEMENTASI DAN PEMAHAMAN DESIGN PATTERN SINGLETON

Buka halaman web berikut https://refactoring.guru/design-patterns/observer dan scroll ke bagian "Code Examples", pilih kode yang akan dilihat misalnya C# dan ikuti langkah-langkah berikut:

a. Dengan contoh yang sudah diberikan, buatlah sebuah class dengan design pattern singleton dengan nama "PusatDataSingleton".
b. Class "PusatDataSingleton" mempunyai dua atribut yaitu "DataTersimpan" yang mempunyai tipe berupa List<string> dan property singleton dengan nama "_instance" dengan tipe data "PusatDataSingleton" itu sendiri.
c. Class tersebut juga memiliki beberapa method yaitu:
   - Konstruktor dari kelas tersebut yang mengisi atribut "DataTersimpan" dengan list kosong.
   - GetDataSingleton() yang mengembalikan "_instance" jika tidak null dan memanggil konstruktor terlebih dahulu apabila nilainya masih null.
   - GetSemuaData() yang mengembalikan list dari property "DataTersimpan".
   - PrintSemuaData() yang melakukan print satu per satu dari string yang ada di list "DataTersimpan".
   - AddSebuahData(string input) yang menambahkan satu data baru "input" ke dalam list "DataTersimpan".
   - HapusSebuahData(int index) yang menghapus sebuah data berdasarkan index tertentu.

4. IMPLEMENTASI PROGRAM UTAMA

Tambahkan beberapa implementasi di program/method utama atau “main”:

a. Buatlah dua variable dengan tipe “PusatDataSingleton” bernama data1 dan data2.
b. Isi kedua variable tersebut dengan hasil keluaran dari GetDataSingleton().
c. Pada data1 lakukan pemanggilan method AddSebuahData() beberapa kali dengan input nama anggota kelompok dan asisten praktikum.
d. Pada data2 panggil method PrintSemuaData(), pastikan keluaran dari hasil print data2 menampilkan nama-nama anggota kelompok dan asisten praktikum.
e. Pada data2 panggil HapusSebuahData() untuk menghapus nama asisten praktikum anda sekarang.
f. Pada data1 panggil PrintSemuaData(), dan seharusnya nama asisten praktikum anda tidak muncul di hasil print tersebut.
g. Langkah terakhir, pada data1 dan data2 panggil GetSemuaData() dan lakukan print dari jumlah “Count” atau elemen yang ada di list pada data1 dan data2.

Source Code:

PusatDataSingleton.cs

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace modul13_2211104005
{
    4 references
    public class PusatDataSingleton
    {
        // Atribut Singleton dan List
        3 references
        private static PusatDataSingleton _instance;
        6 references
        private List<string> DataTersimpan;

        // Konstruktor Private
        1 reference
        private PusatDataSingleton()
        {
            DataTersimpan = new List<string>();
        }

        // Method Singleton
        0 references
        public static PusatDataSingleton GetDataSingleton()
        {
            if (_instance == null)
            {
                _instance = new PusatDataSingleton();
            }
            return _instance;
        }

        // Method untuk mendapatkan semua data
        0 references
        public List<string> GetSemuaData()
        {
            return DataTersimpan;
        }

        // Method untuk mencetak semua data
        0 references
        public void PrintSemuaData()
        {
            Console.WriteLine(" Data Tersimpan:");
            foreach (string data in DataTersimpan)
            {
                Console.WriteLine("- " + data);
            }
        }

        // Method untuk menambahkan data
        0 references
        public void AddSebuahData(string input)
        {
            DataTersimpan.Add(input);
        }

        // Method untuk menghapus data berdasarkan index
        0 references
        public void HapusSebuahData(int index)
        {
            if (index >= 0 && index < DataTersimpan.Count)
            {
                DataTersimpan.RemoveAt(index);
            }
            else
            {
                Console.WriteLine("Index tidak valid.");
            }
        }
    }
}
```

Program.cs

```
using System;

namespace modul13_2211104005
{
    0 references
    class Program
    {
        0 references
        static void Main(string[] args)
        {
            Console.WriteLine(" Nama : Nita Fitrotul Mar'ah ");
            Console.WriteLine(" NIM  : 2211104005 ");
            Console.WriteLine(" Kelas: SE0601 ");

            // Membuat dua variable Singleton
            PusatDataSingleton data1 = PusatDataSingleton.GetDataSingleton();
            PusatDataSingleton data2 = PusatDataSingleton.GetDataSingleton();

            // Menambahkan data (nama anggota kelompok dan asisten)
            data1.AddSebuahData("Nama Anggota 1: Dina");
            data1.AddSebuahData("Nama Anggota 2: Rezky");
            data1.AddSebuahData("Nama Anggota 3: Nadia");
            data1.AddSebuahData("Nama Asisten Praktikum: Imelda");

            // Mencetak semua data melalui data2 (harusnya sama dengan data1)
            Console.WriteLine("\n ======= Data pada data2 ======= ");
            data2.PrintSemuaData();

            // Menghapus nama asisten praktikum
            data2.HapusSebuahData(3); // Hapus asisten praktikum di index ke-3

            // Mencetak kembali data melalui data1 (asisten harus sudah terhapus)
            Console.WriteLine("\n ======= Data pada data1 setelah penghapusan ======= ");
            data1.PrintSemuaData();

            // Mencetak jumlah data pada kedua variabel
            Console.WriteLine("\n Jumlah data pada data1: " + data1.GetSemuaData().Count);
            Console.WriteLine(" Jumlah data pada data2: " + data2.GetSemuaData().Count);
        }
    }
}
```

Hasil:

```
Nama : Nita Fitrotul Mar'ah
NIM : 2211104005
Kelas: SE0601

======= Data pada data2 =======
Data Tersimpan:
- Nama Anggota 1: Dina
- Nama Anggota 2: Rezky
- Nama Anggota 3: Nadia
- Nama Asisten Praktikum: Imelda

======= Data pada data1 setelah penghapusan =======
Data Tersimpan:
- Nama Anggota 1: Dina
- Nama Anggota 2: Rezky
- Nama Anggota 3: Nadia

Jumlah data pada data1: 3
Jumlah data pada data2: 3

D:\KPL\13_Design_Pattern_Implementation\Jurnal\modul13_2211104005
4017.exe (process 7968) exited with code 0 (0x0).
```

Penjelasan

Dalam file PusatDataSingleton.cs, terdapat sebuah class bernama PusatDataSingleton yang menerapkan design pattern Singleton. Tujuan penggunaan pola ini adalah untuk menjamin bahwa hanya satu objek dari class tersebut yang digunakan di seluruh bagian aplikasi. Class ini memiliki atribut DataTersimpan berupa list bertipe string untuk menyimpan data, serta atribut privat _instance yang menyimpan satu-satunya objek Singleton.

Metode GetDataSingleton() digunakan untuk mengakses instance tersebut. Jika instance belum dibuat, maka akan dibuat satu kali; jika sudah ada, maka instance yang sama akan dikembalikan. Class ini juga menyediakan beberapa method seperti AddSebuahData, HapusSebuahData, dan PrintSemuaData untuk menambah, menghapus, dan menampilkan isi data dalam list.

Sementara itu, dalam file Program.cs, dilakukan pengujian terhadap pola Singleton. Dua variabel, yaitu data1 dan data2, dibuat menggunakan GetDataSingleton(). Walaupun terlihat seperti dua objek berbeda, sebenarnya keduanya menunjuk pada objek Singleton yang sama. Saat data ditambahkan melalui data1, data tersebut juga akan terlihat pada data2. Begitu pula saat data dihapus dari data2, perubahannya tercermin di data1. Ini membuktikan bahwa data1 dan data2 merujuk pada instance yang sama. Program kemudian mencetak jumlah data yang sama dari kedua variabel tersebut sebagai bukti kesamaan instansinya.