Nama : Muhammad Fariz Nur Hidayat

Kelas : SE063

NIM : 2211104069

1. MENJELASKAN DESIGN PATTERN SINGLETON

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

A. Berikan salah dua contoh kondisi dimana design pattern “Singleton” dapat digunakan.

Jawab :

- Koneksi database — Dalam sebuah aplikasi, hanya dibutuhkan satu koneksi database yang dapat diakses dari mana saja tanpa membuat koneksi baru berkali-kali.
- Logger (Pencatat log) — Sistem pencatatan log biasanya hanya memerlukan satu instance logger untuk mengelola dan menyimpan catatan aktivitas aplikasi secara terpusat.

B. Berikan penjelasan singkat mengenai langkah-langkah dalam mengimplementasikan design pattern “Singleton”.

Jawab :

- Buat konstruktor kelas menjadi private agar objek dari luar kelas tidak bisa dibuat langsung.
- Buat sebuah atribut statis private yang menyimpan instance tunggal dari kelas tersebut.
- Buat method statis public, misal GetInstance(), yang berfungsi untuk mengembalikan instance tunggal, jika instance belum ada maka method ini akan membuatnya terlebih dahulu (lazy initialization).
- Semua akses ke instance Singleton harus melalui method ini sehingga memastikan hanya satu instance yang dibuat dan digunakan di seluruh aplikasi.

C. Berikan tiga kelebihan dan kekurangan dari design pattern “Singleton”.

Jawab :

Kelebihan:

- Menjamin hanya ada satu instance objek yang dibuat di seluruh aplikasi.
- Memberikan titik akses global ke instance tersebut.
- Menghemat sumber daya karena instance dapat digunakan kembali tanpa membuat objek baru.

Kekurangan:

- Melanggar prinsip tanggung jawab tunggal (single responsibility principle) karena mengelola instance sekaligus logika bisnis.
- Bisa menyebabkan kesulitan pada pengujian unit (unit testing) karena ketergantungan global.
- Dalam lingkungan multithread, perlu penanganan khusus agar instance tidak dibuat ganda (konkurensi/locking).

2. IMPLEMENTASI DAN PEMAHAMAN DESIGN PATTERN SINGLETON

Buka halaman web berikut https://refactoring.guru/design-patterns/singleton 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:

i. Konstruktor dari kelas tersebut yang mengisi atribut "DataTersimpan" dengan list kosong.

ii. GetDataSingleton() yang mengembalikan "_instance" jika tidak null dan memanggil konstruktor terlebih dahulu apabila nilainya masih null.

iii. GetSemuaData() yang mengembalikan list dari property "DataTersimpan".

iv. PrintSemuaData() yang melakukan print satu per satu dari string yang ada di list "DataTersimpan".

v. AddSebuahData(string input) yang menambahkan satu data baru "input" ke dalam list "DataTersimpan".

vi. HapusSebuahData(int index) yang menghapus sebuah data berdasarkan index tertentu.

Jawab :

Program.cs

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

namespace modul13_2211104069
{
   public class PusatDataSingleton
   {
      // Property singleton: instance tunggal
      private static PusatDataSingleton _instance;

      // Atribut data tersimpan berupa List<string>
      private List<string> DataTersimpan;

      // Konstruktor private agar tidak bisa diakses luar kelas
      private PusatDataSingleton()
      {
         DataTersimpan = new List<string>();
      }

      // Method untuk mendapatkan instance singleton
      public static PusatDataSingleton GetDataSingleton()
      {
         if (_instance == null)
         {
            _instance = new PusatDataSingleton();
         }
         return _instance;
      }

      // Mengembalikan list data tersimpan
      public List<string> GetSemuaData()
```

```
        {
            return DataTersimpan;
        }

        // Mencetak semua data yang tersimpan
        public void PrintSemuaData()
        {
            Console.WriteLine("Data Tersimpan:");
            for (int i = 0; i < DataTersimpan.Count; i++)
            {
                Console.WriteLine($"{i}: {DataTersimpan[i]}");
            }
        }

        // Menambahkan sebuah data baru ke list
        public void AddSebuahData(string input)
        {
            DataTersimpan.Add(input);
        }

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

    internal class Program
    {
        static void Main(string[] args)
        {
            // Kosong sesuai struktur awal
        }
    }
}
```

Penjelasan kode

Kode di atas mengimplementasikan pola desain **Singleton** melalui kelas PusatDataSingleton. Kelas ini memastikan hanya ada satu instance tunggal yang dapat diakses secara global dengan menggunakan metode statis GetDataSingleton(). Atribut DataTersimpan menyimpan data dalam bentuk list string, dan hanya bisa diakses melalui instance singleton tersebut. Konstruktor kelas dibuat private agar tidak dapat dibuat instance baru dari luar kelas. Kelas ini menyediakan berbagai metode untuk mengelola data, seperti menambahkan data (AddSebuahData), menghapus data berdasarkan indeks (HapusSebuahData), mendapatkan seluruh data (GetSemuaData), dan mencetak semua data yang tersimpan (PrintSemuaData). Dengan demikian, pola Singleton pada kelas ini menjamin bahwa semua bagian program yang menggunakan PusatDataSingleton akan bekerja dengan satu objek data yang sama dan konsisten.

3. 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.

Jawab:

Program.cs

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

```
using System.Threading.Tasks;

namespace modul13_2211104069
{
   public class PusatDataSingleton
   {
      // Property singleton: instance tunggal
      private static PusatDataSingleton _instance;

      // Atribut data tersimpan berupa List<string>
      private List<string> DataTersimpan;

      // Konstruktor private agar tidak bisa diakses luar kelas
      private PusatDataSingleton()
      {
         DataTersimpan = new List<string>();
      }

      // Method untuk mendapatkan instance singleton
      public static PusatDataSingleton GetDataSingleton()
      {
         if (_instance == null)
         {
            _instance = new PusatDataSingleton();
         }
         return _instance;
      }

      // Mengembalikan list data tersimpan
      public List<string> GetSemuaData()
      {
         return DataTersimpan;
      }

      // Mencetak semua data yang tersimpan
      public void PrintSemuaData()
      {
         Console.WriteLine("Data Tersimpan:");
         for (int i = 0; i < DataTersimpan.Count; i++)
         {
            Console.WriteLine($"{i}: {DataTersimpan[i]}");
         }
      }

      // Menambahkan sebuah data baru ke list
      public void AddSebuahData(string input)
```

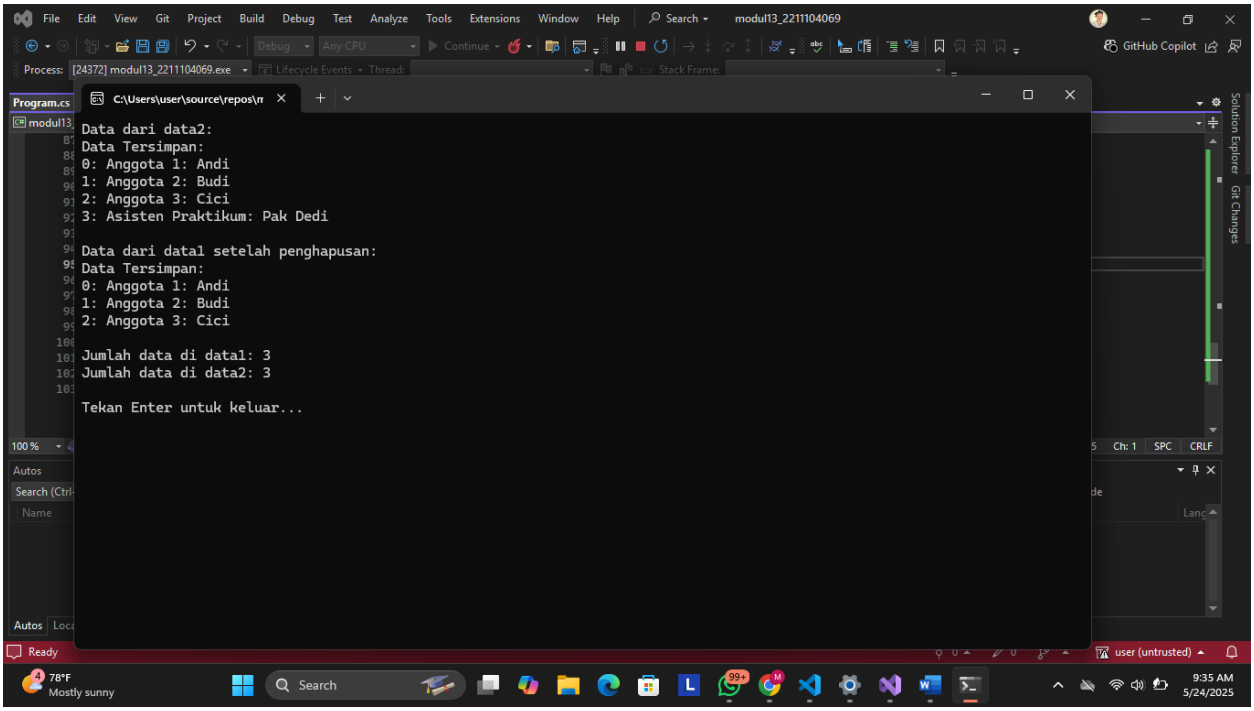
```
    {
        DataTersimpan.Add(input);
    }

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

internal class Program
{
    static void Main(string[] args)
    {
        // A. Membuat dua variabel dengan tipe PusatDataSingleton
        var data1 = PusatDataSingleton.GetDataSingleton();
        var data2 = PusatDataSingleton.GetDataSingleton();

        // B. Mengisi data1 dengan beberapa data nama anggota dan asisten praktikum
        data1.AddSebuahData("Anggota 1: Andi");
        data1.AddSebuahData("Anggota 2: Budi");
        data1.AddSebuahData("Anggota 3: Cici");
        data1.AddSebuahData("Asisten Praktikum: Pak Dedi");

        // C. Memanggil PrintSemuaData() melalui data2 untuk menampilkan semua data
        Console.WriteLine("Data dari data2:");
        data2.PrintSemuaData();

        // D. Menghapus nama asisten praktikum melalui data2 (misal index 3)
        data2.HapusSebuahData(3);

        // E. Memanggil PrintSemuaData() melalui data1, asisten seharusnya sudah hilang
        Console.WriteLine("\nData dari data1 setelah penghapusan:");
        data1.PrintSemuaData();

        // F. Menampilkan jumlah data di data1 dan data2
        Console.WriteLine($"\nJumlah data di data1: {data1.GetSemuaData().Count}");
        Console.WriteLine($"Jumlah data di data2: {data2.GetSemuaData().Count}");
```

```
            Console.WriteLine("\nTekan Enter untuk keluar...");
            Console.ReadLine();
        }
    }
}
```

Penjelasan Kode

Kode di atas mengimplementasikan pola desain Singleton melalui kelas PusatDataSingleton, yang memastikan hanya ada satu instance tunggal dari kelas ini selama program berjalan. Kelas tersebut menyimpan data berupa list string dan menyediakan berbagai metode untuk menambah, menghapus, menampilkan, dan mengambil data yang tersimpan. Di dalam method Main, dua variabel (data1 dan data2) dibuat dengan memanggil method GetDataSingleton(), sehingga keduanya merujuk ke instance yang sama. Data ditambahkan melalui data1, kemudian data tersebut ditampilkan menggunakan data2, menunjukkan bahwa kedua variabel berbagi data yang sama. Selanjutnya, data asisten praktikum dihapus menggunakan data2, dan hasilnya dapat terlihat kembali melalui data1, membuktikan konsistensi instance tunggal. Program kemudian menampilkan jumlah data yang tersimpan dari kedua variabel, yang pasti sama, memperkuat konsep Singleton yang memastikan penggunaan satu objek bersama di seluruh aplikasi.

Output


File Edit View Git Project Build Debug Test Analyze Tools Extensions Window Help
modul13_2211104069
GitHub Copilot
Program.cs
C:\Users\user\source\repos\r
Data dari data2:
Data Tersimpan:
0: Anggota 1: Andi
1: Anggota 2: Budi
2: Anggota 3: Cici
3: Asisten Praktikum: Pak Dedi
Data dari data1 setelah penghapusan:
Data Tersimpan:
0: Anggota 1: Andi
1: Anggota 2: Budi
2: Anggota 3: Cici
Jumlah data di data1: 3
Jumlah data di data2: 3
Tekan Enter untuk keluar...
Solution Explorer
Git Changes
Ready
78°F
Mostly sunny
Search
9:35 AM
5/24/2025