TUGAS JURNAL
KONSTRUKSI PERANGKAT LUNAK
MODUL XIII
DESIGN PATTERN IMPLEMENTATION

Disusun Oleh:

Maria Nathasya Desfera Pangestu

2211104008

SE0601

Dosen Pengampu:

Yudha Islami Sulistya, S.Kom., M.Cs.

PROGRAM STUDI S1 REKAYASA PERANGKAT LUNAK

FAKULTAS INFORMATIKA

TELKOM UNIVERSITY PURWOKERTO

2025

Tugas Jurnal

1. 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
      - Pengaturan Konfigurasi Aplikasi
        Pola Singleton dapat diterapkan untuk menyimpan pengaturan konfigurasi aplikasi yang hanya perlu dimuat satu kali dan digunakan di seluruh bagian aplikasi. Contohnya termasuk pengaturan koneksi database atau konfigurasi API.
      - Pencatatan Log Aplikasi (Logging)
        Singleton digunakan untuk memastikan bahwa hanya ada satu instance logger yang bertanggung jawab untuk mencatat log dari seluruh aplikasi, sehingga memudahkan proses pemantauan dan debugging.

   B. Berikan penjelasan singkat mengenai mengimplementasikan design pattern “Singleton”
      Jawab:
      Untuk membuat class Singleton, langkah pertama adalah mendefinisikan konstruktor sebagai privat, yang mencegah pembuatan instance dari luar class tersebut. Selanjutnya, perlu dibuat atribut statis yang akan menyimpan satu-satunya instance dari class ini. Kemudian, disediakan method statis yang berfungsi untuk mendapatkan instance; method ini akan memeriksa apakah instance sudah ada. Jika belum ada, maka instance baru akan dibuat dan disimpan dalam atribut statis tersebut. Sebagai langkah tambahan, untuk memastikan bahwa class ini tidak dapat diwarisi, pada beberapa bahasa pemrograman, class Singleton dapat ditandai sebagai sealed (atau final) untuk mencegah pewarisan.

   C. Berikan kelebihan dan kekurangan dari design pattern “Singleton”
      Jawab:
      Kelebihan:
      - Efisiensi Memori
        Dengan hanya ada satu instance yang digunakan, penggunaan memori menjadi lebih hemat.
      - Akses Mudah di Seluruh Aplikasi
        Instance dapat diakses dari mana saja tanpa perlu membuat objek baru.
      - Menghindari Konflik Data
        Karena hanya terdapat satu instance, data yang disimpan tetap konsisten di seluruh aplikasi.

      Kekurangan:

      - Kesulitan dalam Pengujian
        Singleton sulit untuk diuji karena instance bersifat global dan tidak mudah untuk direset.

- Potensi Bottleneck
  Jika terlalu banyak operasi dilakukan pada Singleton, hal ini dapat menyebabkan kemacetan.
- Kurang Fleksibel
  Pola ini tidak cocok digunakan jika aplikasi memerlukan beberapa instance dari class yang sama.

2. IMPLEMENTASI DAN PEMAHAMAN DESIGN PATTERN OBSERVER 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 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

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

Source code Program.cs

```
modul13_2211104008                                  modul13_2211104008.Program
```

```
using System;

namespace modul13_2211104008
{
    0 references
    class Program
    {
        0 references
        static void Main(string[] args)
        {
            Console.WriteLine(" Nama : Maria Nathasya ");
            Console.WriteLine(" NIM  : 2211104008 ");
            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: Lintang");
            data1.AddSebuahData("Nama Anggota 2: Devrin");
            data1.AddSebuahData("Nama Anggota 3: Alfian");
            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);
        }
    }
}
```

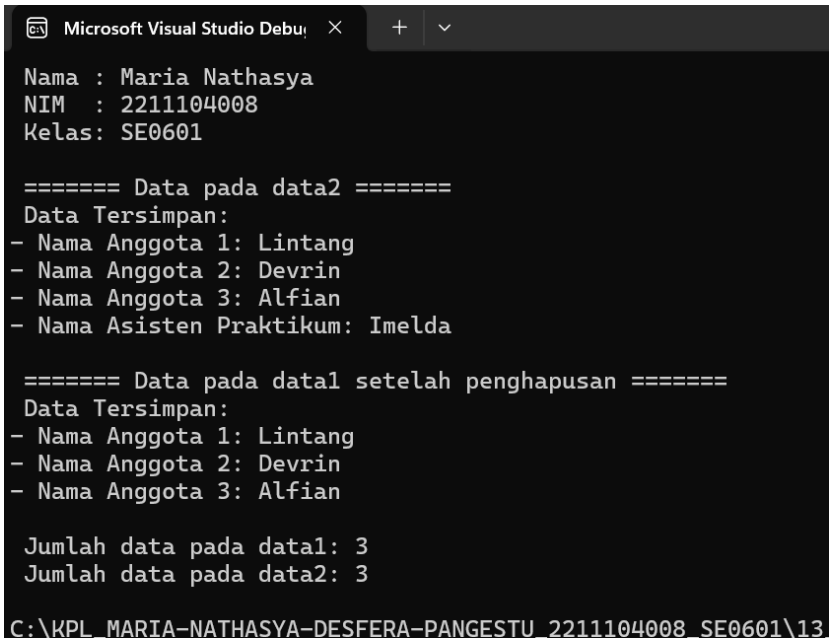
Source code PusatDataSingleton.cs

modul13_2211104008 | modul13_221110

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

namespace modul13_2211104008
{
    8 references
    public class PusatDataSingleton
    {
        // Atribut Singleton dan List
        private static PusatDataSingleton _instance;
        private List<string> DataTersimpan;

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

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

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

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

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

```

```
53          // Method untuk menghapus data berdasarkan index
            1 reference
54          public void HapusSebuahData(int index)
55          {
56              if (index >= 0 && index < DataTersimpan.Count)
57              {
58                  DataTersimpan.RemoveAt(index);
59              }
60              else
61              {
62                  Console.WriteLine("Index tidak valid.");
63              }
64          }
65      }
66  }
```

Output



Pejelasan

Dalam file PusatDataSingleton.cs, terdapat class PusatDataSingleton yang menerapkan pola desain Singleton untuk memastikan hanya ada satu instance yang digunakan di seluruh program. Class ini memiliki atribut DataTersimpan berupa list string untuk menyimpan data dan _instance untuk menyimpan instance tunggal. Metode GetDataSingleton() memastikan instance hanya dibuat sekali, sedangkan metode lain seperti AddSebuahData, HapusSebuahData, dan PrintSemuaData digunakan untuk mengelola data. File Program.cs digunakan untuk menguji pola Singleton dengan membuat dua variabel (data1 dan data2) yang merujuk pada instance yang sama. Data yang ditambahkan melalui data1 juga terlihat di data2, dan perubahan pada data2 akan tercermin di data1, membuktikan bahwa keduanya adalah satu instance. Program ini mencetak jumlah data dari kedua variabel, yang hasilnya selalu sama.