

Nama : Reva Doni Aprilio

NIM : 1302204051

Kelas : SE-44-01

Jurnal Modul 11

1. Menjelaskan Design Pattern

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

- Saat dimana semua instance membutuhkan data yang sama, misal data perpustakaan dan terdapat lebih dari 1 admin dan banyak buku. Semua admin dapat mengakses data buku yang sama sehingga tidak akan terjadi perbedaan data.
- Umumnya digunakan untuk mengecek variable static didalam sebuah class, karena semua objek pasti akan mendapatkan hasil yg sama.

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

- Membuat sebuah class Singleton
- Membuat property _instance dengan tipe data class nya dan dibuat static.
- Membuat method static untuk mengakses instance, dicek dahulu apakah sudah ada atau belum. Jika belum ada akan membuat instance baru, lalu return instance.
- Coba class di main program, membuat minimal 2 objek dari class singleton. Mekanisme 2 objek tersebut akan dapat berbagi data yang sama karena menerapkan design pattern singleton.

C. Berikan tiga kelebihan dan kekurangan dari design pattern "Singleton".

Kelebihan :

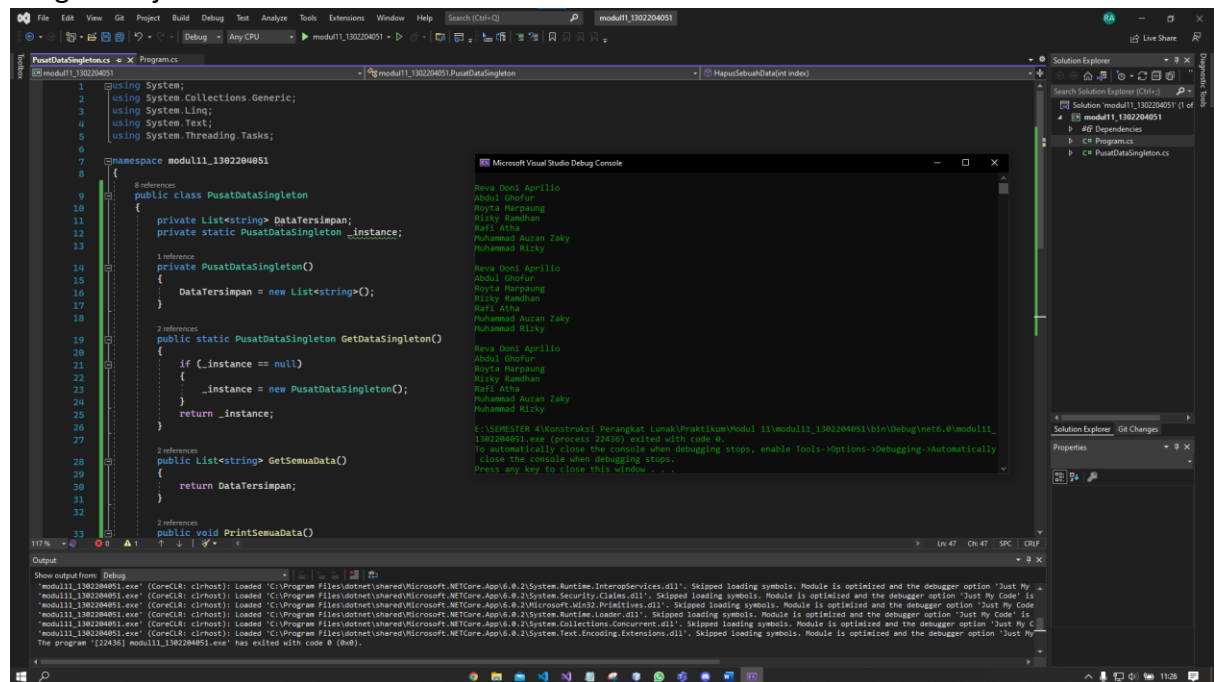
1. Dapat mengakses data secara global
2. Singleton di inisialisasi hanya sekali saat pertama kali dibutuhkan
3. Pastikan dalam 1 class hanya terdapat 1 instance

Kekurangan :

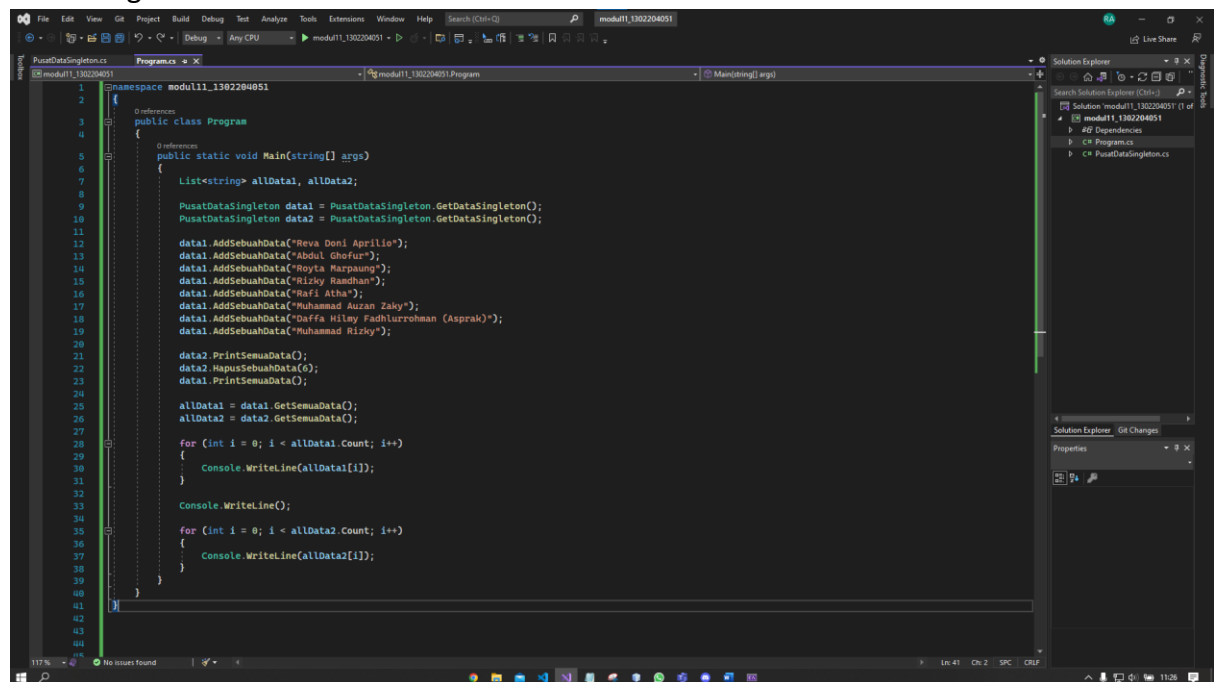
1. Akan menjadi kurang bagus jika punya banyak component dan masing2 komponen mengetahui terlalu banyak data tentang satu sama lain.
2. Mungkin akan sulit saat menjalankan unit test.
3. Pattern ini membutuhkan treatment khusus

D. Screenshot

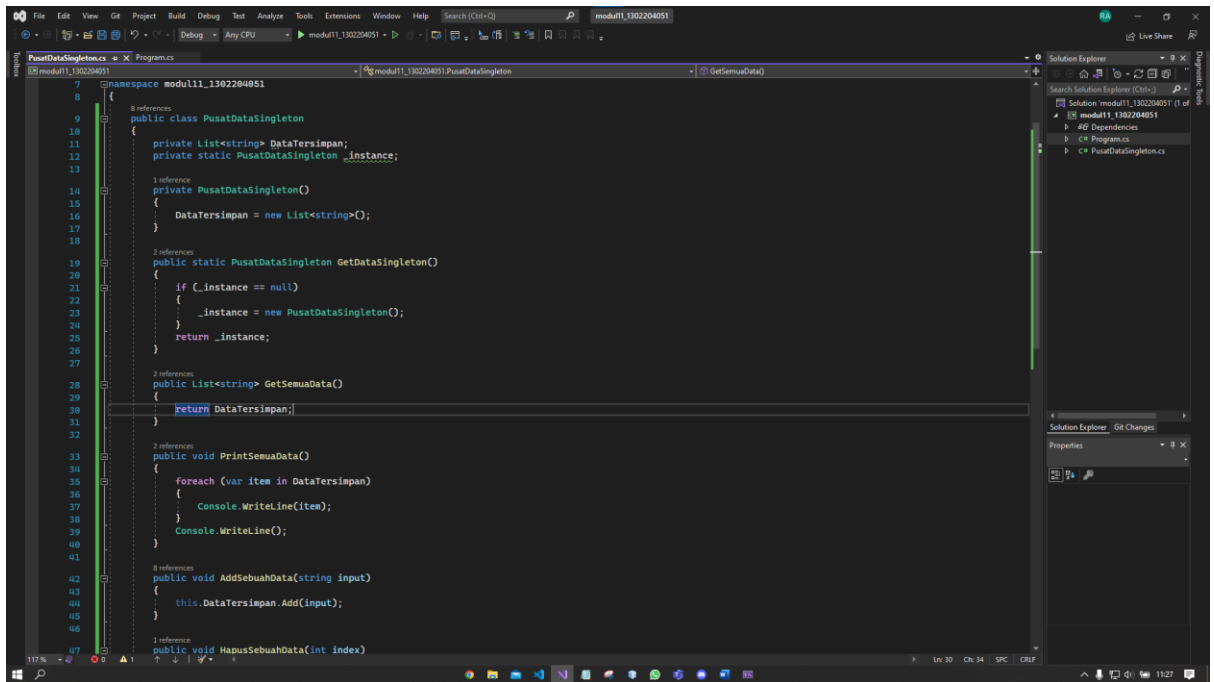
Program dijalankan :



Class Program :

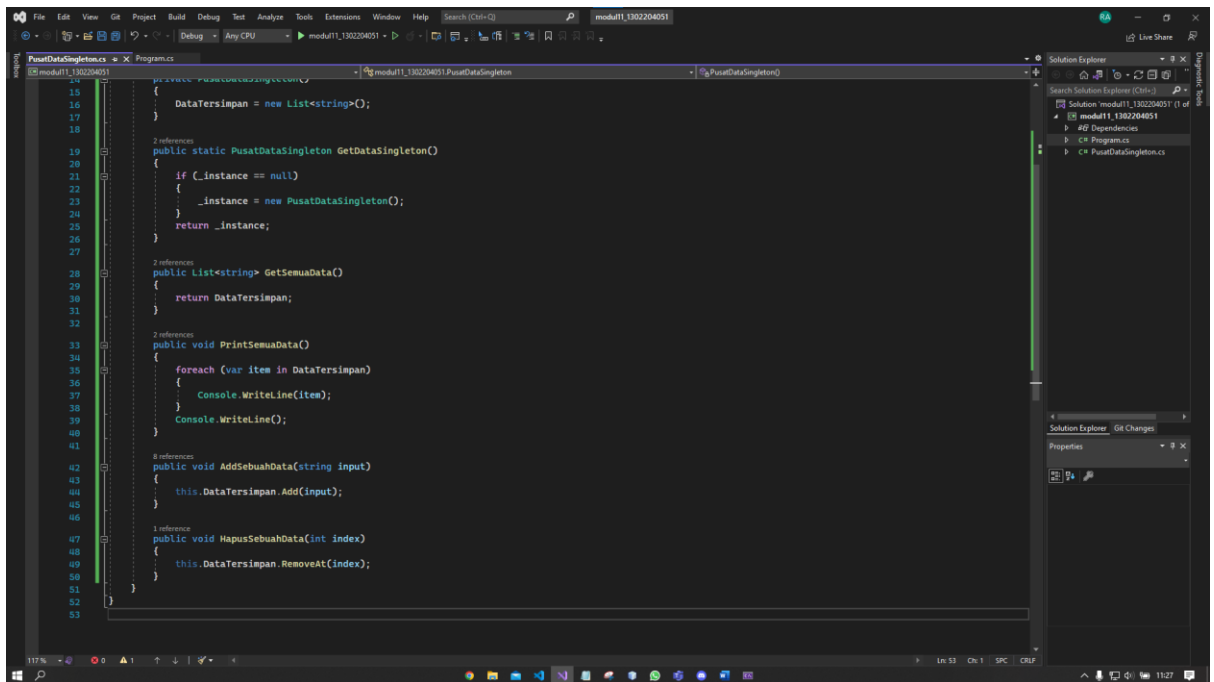


Class PusatDataSingleton:



This screenshot shows the implementation of the `PusatDataSingleton` class in a C# project. The class is defined within the `modul11_1302204051` namespace. It includes a private list `DataTersimpan` and a static instance `_instance`. The `GetDataSingleton()` method ensures only one instance exists. Other methods include `GetSemuaData()`, `PrintSemuaData()`, `AddSebuahData()`, and `HapusSebuahData()`.

```
7 namespace modul11_1302204051
8 {
9     8 references:
10     public class PusatDataSingleton
11     {
12         private List<string> DataTersimpan;
13         private static PusatDataSingleton _instance;
14
15         1 reference:
16         private PusatDataSingleton()
17         {
18             DataTersimpan = new List<string>();
19
20         2 references:
21         public static PusatDataSingleton GetDataSingleton()
22         {
23             if (_instance == null)
24             {
25                 _instance = new PusatDataSingleton();
26             }
27             return _instance;
28
29         2 references:
30         public List<string> GetSemuaData()
31         {
32             return DataTersimpan;
33
34         2 references:
35         public void PrintSemuaData()
36         {
37             foreach (var item in DataTersimpan)
38             {
39                 Console.WriteLine(item);
40             }
41             Console.WriteLine();
42
43         8 references:
44         public void AddSebuahData(string input)
45         {
46             this.DataTersimpan.Add(input);
47
48         1 reference:
49         public void HapusSebuahData(int index)
50         {
51             this.DataTersimpan.RemoveAt(index);
52         }
53     }
```



This screenshot shows the `HapusSebuahData` method in the `PusatDataSingleton` class. The method takes an integer index and removes the corresponding element from the `DataTersimpan` list using `RemoveAt`.

```
15 private PusatDataSingleton()
16 {
17     DataTersimpan = new List<string>();
18 }
19
20 2 references:
21 public static PusatDataSingleton GetDataSingleton()
22 {
23     if (_instance == null)
24     {
25         _instance = new PusatDataSingleton();
26     }
27     return _instance;
28
29 2 references:
30 public List<string> GetSemuaData()
31 {
32     return DataTersimpan;
33
34 2 references:
35 public void PrintSemuaData()
36 {
37     foreach (var item in DataTersimpan)
38     {
39         Console.WriteLine(item);
40     }
41     Console.WriteLine();
42
43 8 references:
44 public void AddSebuahData(string input)
45 {
46     this.DataTersimpan.Add(input);
47
48 1 reference:
49 public void HapusSebuahData(int index)
50 {
51     this.DataTersimpan.RemoveAt(index);
52 }
53 }
```