

**TUGAS PENDAHULUAN
KONSTRUKSI PERANGKAT LUNAK**

**MODUL XIV
CLEAN CODE**



Disusun Oleh :

Atika Aji Hadiyani

2211104003

SE0601

Dosen Pengampu :

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

**PROGRAM STUDI S1 SOFTWARE ENGINEERING
TELKOM UNIVERSITY PURWOKERTO**

TUGAS PENDAHULUAN XIV

1. MEMBUAT PROJECT MODUL

Buka IDE misalnya dengan Visual Studio

- A. Copy salah satu folder tugas pendahuluan yang dimiliki sebelumnya (dari modul 2 sampai modul 10), kemudian rename folder hasil copy-paste tersebut dengan tpmodul14_NIM (coba pilih tugas pendahuluan yang paling sederhana)
- B. Misalnya menggunakan Visual Studio, bukalah project/folder yang di-copy sebelumnya.

2. REFACTORING DENGAN STANDAR CODE

Dengan mengikuti standard code yang digunakan (misal C# dengan standar dari .NET), pastikan kode yang dikumpulkan memenuhi faktor-faktor berikut:

- A. Naming convention
 - Variable / Property / Attribute
 - Method / Function / Procedure
- B. White space dan indentation
- C. Variable / attribute declarations
- D. Comments

Jawab:

Saya menyalin tugas pendahuluan modul lima tentang “Generics” dan rename dengan nama folder tpmodul14_2211104003.

a. Source Code sebelum di refactor

```
using System;

public class HaloGeneric
{
    public static void SapaUser<T>(T user)
    {
        Console.WriteLine($"Halo user {user}");
    }
}

public class DataGeneric<T>
{
    private T data;

    public DataGeneric(T data)
    {
        this.data = data;
    }

    public void PrintData()
    {
        Console.WriteLine($"Data yang tersimpan adalah: {data}");
    }
}

class Program
{
    static void Main()
    {
        HaloGeneric.SapaUser("Atika Aji");

        // Membuat objek DataGeneric dengan NIM
        DataGeneric<string> data = new DataGeneric<string>("2211104003");
        data.PrintData();
    }
}
```

b. Source Code sesudah di refactor

```
using System;

// Kelas generik untuk menampilkan sapaan kepada user
1 reference
public class GreetingUtility
{
    1 reference
    public static void DisplayGreeting<T>(T userName)
    {
        Console.WriteLine($"Halo user {userName}");
    }
}

// Kelas generik untuk menyimpan dan menampilkan data
3 references
public class GenericData<T>
{
    private T _data;

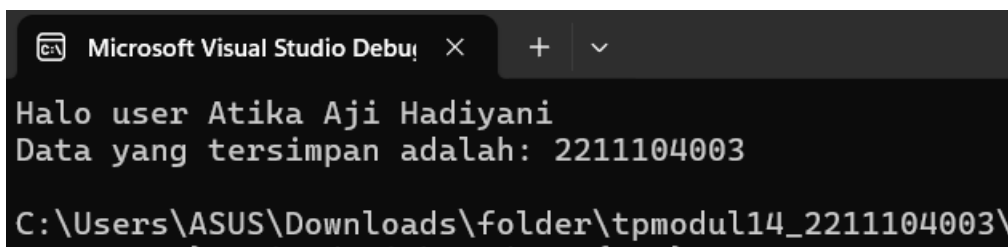
    // Konstruktor untuk menyimpan data
    1 reference
    public GenericData(T data)
    {
        _data = data;
    }

    // Method untuk mencetak data yang tersimpan
    1 reference
    public void DisplayData()
    {
        Console.WriteLine($"Data yang tersimpan adalah: {_data}");
    }
}

// Kelas utama Program
0 references
class Program
{
    0 references
    static void Main()
    {
        // Menampilkan sapaan untuk user
        GreetingUtility.DisplayGreeting("Atika Aji Hadiyani");

        // Membuat objek GenericData dengan NIM dan menampilkan data
        GenericData<string> studentData = new GenericData<string>("2211104003");
        studentData.DisplayData();
    }
}
```

c. Hasil



The screenshot shows the Microsoft Visual Studio Debug Console window. The title bar indicates it is the 'Debug Console' for the 'Microsoft Visual Studio Debu' project. The console output displays the following text: 'Halo user Atika Aji Hadiyani' followed by 'Data yang tersimpan adalah: 2211104003'. The full path of the file being executed is visible at the bottom: 'C:\Users\ASUS\Downloads\folder\tpmodul14_2211104003\...'.

d. Penjelasan

Berikut merupakan penjabaran mengenai clean code yang telah melalui proses refactoring:

1. Naming Convention

a. Class Names:

- HaloGeneric diubah menjadi GreetingUtility agar lebih deskriptif dan jelas.
- DataGeneric diubah menjadi GenericData agar lebih deskriptif

b. Method Names:

- SapaUser diubah menjadi DisplayGreeting agar sesuai standar PascalCase dan lebih jelas).
- PrintData diubah menjadi DisplayData agar lebih deskriptif

c. Variable Names:

- Variabel `_data` diberi prefiks `_` untuk menunjukkan bahwa itu adalah variabel private sesuai standar .NET.

2. Variable / Property / Attribute

- Atribut `_data` sudah menggunakan konvensi penamaan dengan prefiks `_` untuk variabel private.
- Variabel lokal `studentData` sudah jelas dan menggunakan camelCase agar sesuai standar untuk variabel lokal.

3. Method / Function / Procedure

- Semua method menggunakan PascalCase, sesuai standar .NET.
- Metode memiliki nama yang deskriptif: `DisplayGreeting` dan `DisplayData` yang langsung menunjukkan fungsinya.

4. White Space dan Indentation

- Indentasi konsisten menggunakan 4 spasi agar sesuai standar Visual Studio.
- Pemisahan antar class memiliki whitespace yang jelas.
- Setiap method memiliki spacing yang cukup untuk keterbacaan.

5. Variable / Attribute Declarations

- Atribut `_data` dideklarasikan secara private dengan tipe generik.

- b. Variable `studentData` diinisialisasi dengan tipe generik `GenericData<string>`.

6. Comments

Comment ditambahkan di bagian:

- a. Setiap class untuk menjelaskan fungsinya.
- b. Method untuk menjelaskan fungsi metode tersebut.
- c. Bagian kode utama (Main) untuk menjelaskan proses yang dilakukan.