**Spring Boot Design Patterns - Creational**

## **Pendahuluan**

**Sebelumnya harus sudah paham materi terkait** :

* Java
* Spring Boot

**Agenda**

* Pengenalan Design Pattern
* Pengenalan Creational Design Pattern
* Singleton
* Prototype
* Abstract Factory
* Builder
* Factory Method

## **Pengenalan Design Patterns**

**Pendahuluan**

* Membuat aplikasi berbasis object oriented bukanlah hal yang mudah
* Dan membuat aplikasi berbasis object oriented yang mudah dikembangkan, itu lebih sulit lagi
* Bahkan tak jarang, kadang desain object oriented awal ketika membuat aplikasi, perlu diubah ketika ukuran aplikasi kita semakin kompleks dan besar
* Pengembang aplikasi, biasanya akan mengambil pengalaman dari pengembangan aplikasi sebelumnya, entah dari pengalaman sendiri, atau pengalaman orang lain
* Tujuan course ini adalah belajar membuat desain aplikasi object oriented sebagai Design Patterns

**Apa itu Design Patterns?**

* Design patterns adalah solusi umum yang biasa digunakan ketika menghadapi masalah saat melakukan desain aplikasi
* Design patterns bukanlah solusi yang bisa di transform langsung menjadi kode program
* Design patterns adalah template atau panduan untuk menyelesaikan masalah, dan bisa diimplementasikan berbeda-beda, namun dengan tujuan yang sama

**Buku Design Patterns**

* Design Pattern : Elements of Reusable Object-Oriented Software
* Para penulisnya dikenal dengan panggilan GOF (Gang of Four)

**Kegunaan Design Patterns**

* Design patterns bisa mempercepat proses development dengan menyediakan solusi yang sudah terbukti.
* Menggunakan Design Patterns bisa menolong kita dari kesalahan dimasa depan jika mengimplementasikan solusi sendiri yang belum terbukti baik.
* Design Patterns menyediakan solusi umum, sehingga bisa digunakan di berbagai kasus
* Design Patterns juga sudah umum diadopsi oleh para pengembang perangkat lunak, sehingga menggunakan Design Patterns akan memudahkan perangkat lunak kita mudah dikembangkan

**Jenis-Jenis Design Patterns**

Secara garis besar, Design Patterns dibagi menjadi tiga jenis

* Creational Design Patterns, yaitu Design Patterns yang berhubungan dengan mekanisme pembuatan object
* Structural Design Patterns, yaitu Design Patterns yang berhubungan dengan interaksi antar object dan class ketika membentuk struktur yang lebih besar.
* dan Behavioral Design Patterns, yaitu Design Patterns yang berhubungan dengan penugasan, tanggung jawab, dan hubungan antar object

Pada pembelajaran ini, kita akan fokus membahas Creational Design Patterns

## **Pengenalan Creational Design Patterns**

* Creational Design Patterns adalah Design Patterns untuk membuat object.
* Creational Design Patterns membuat system menjadi lebih independen ketika membuat dan melakukan komposisi object
* Creational Design Patterns kadang menggunakan perawisan untuk membuat object
* Penggunaan Design Patterns tidak diharuskan hanya satu Pattern saja, kita bisa menggabungkan beberapa Design Patterns sekaligus jika memang dibutuhkan

**Jenis-Jenis Creational Design Patterns**

Terdapat banyak Design Pattern yang berjenis Creational, yaitu :

* Singleton,
* Prototype,
* Builder,
* Factory Method,
* dan Abstract Factory

## **Membuat Project**

<https://start.spring.io/>

Dependencies :

* Lombok

## **Singleton Pattern**

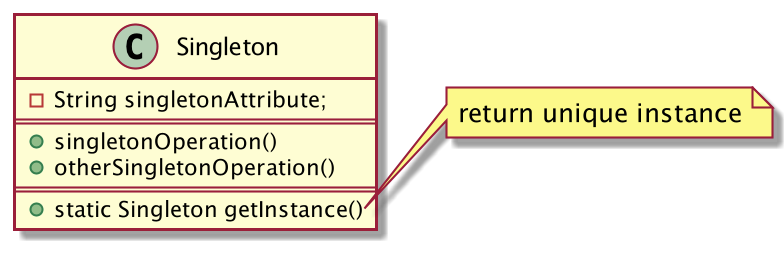
**Singleton**

* GOF : Ensure a class only has one instance, and provide a global point of access to it.
* Memastikan bahwa sebuah class hanya memiliki satu instance object, dan menyediakan cara mengaksesnya secara global (artinya bisa diakses dari mana saja)

**Motivasi**

* Pada beberapa kasus, memastikan sebuah class hanya memiliki satu instant object sangat penting
* Pertanyaannya, bagaimana memastikan sebuah class hanya memiliki satu instance object?
* Solusinya adalah membiarkan class itu sendiri yang bertanggung jawab untuk membuat instance object nya. Class tersebut bisa memastikan tidak ada instance lain yang dibuat, dan akan menyediakan cara mengakses singleton object tersebut
* Ketika terdapat perubahan pada instance object, client yang menggunakan singleton object tersebut tidak perlu melakukan perubahan

**Struktur**



**Keuntungan**

* Class singleton bisa mengatur bagaimana dan kapan client bisa mengakses object singleton tersebut
* Singleton lebih aman dibandingkan global variable. Ini mencegah client secara tidak sengaja mengubah global variable
* Class singleton bisa mengubah implementasi dari instance object nya, tanpa harus melakukan perubahan di client yang mengaksesnya