POO - Design Patterns 1

Design Patterns

Design Patterns - concept

Design pattern-urile sunt soluții tipice pentru probleme care apar în mod frecvent în design-ul software-ului. Acestea sunt o colecție de șabloane care pot fi modificare în funcție de nevoie pentru a rezolva probleme ce țin de design-ul unei soluții.

Un pattern nu reprezintă o bucată de cod ce este introdusă direct în cod, ci mai degrabă un concept pentru a rezolva o problemă anume.

Spre deosebire de algoritme care care rezolvă o problemă urmărind un set de pași exacți, un pattern este doar un concept care poate să difere de la un program la altul.

Design Patterns - clasificare

Design patterns diferă între prin complexitate, implementare, scalabilitatea de a fi aplicat în cât mai multe situații. În funcție de scopul lor, există trei clase de design patterns:

1. Creational patterns: oferă mecanisme de creare a obiectelor care cresc flexibilitatea și au abilitatea de a transforma codul într-unul cât mai reutilizabil
2. Structural patterns: oferă metode de a asambla obiecte și clase în structuri mai mari, pastrându-le cât mai flexibile și eficiente
3. Behavioral patterns: sunt folosiți pentru a realiza o comunicare și o logică de stabilire a responsabilităților între obiecte cât mai eficientă

Creational Patterns

Creational Patterns

Sunt folosiți pentru a crea mecanisme cât mai eficiente pentru crearea de obiecte.

Cei mai folosiți sunt următorii:

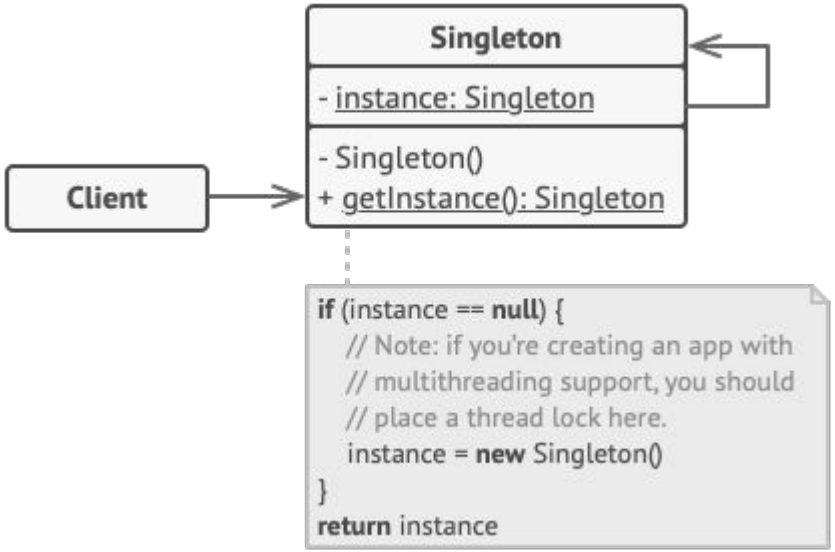
- Singleton
- Factory Method
- Abstract Factory
- Builder
- Prototype

Singleton

Singleton este un Creational Pattern care asigură că există doar un singur obiect pentru clasa respectivă la un moment de timp și oferă o singură cale de accesare a acestuia. Se folosește când avem nevoie să împărțim aceeași resursă în cod (o conexiune la baza de date).

Check list:

- Se definește un atribut private static în clasa singleton care să conțină obiectul
- Se definește un accesor public static care să returneze acel atribut
- Se realizează “inițializarea leneșă” în metoda statică care să creeze un nou obiect la prima apelare
- Constructorii sunt toți private
- Clienții pot folosi doar funcția accesor pentru a manipula Singleton-ul



Singleton Pattern Implementare

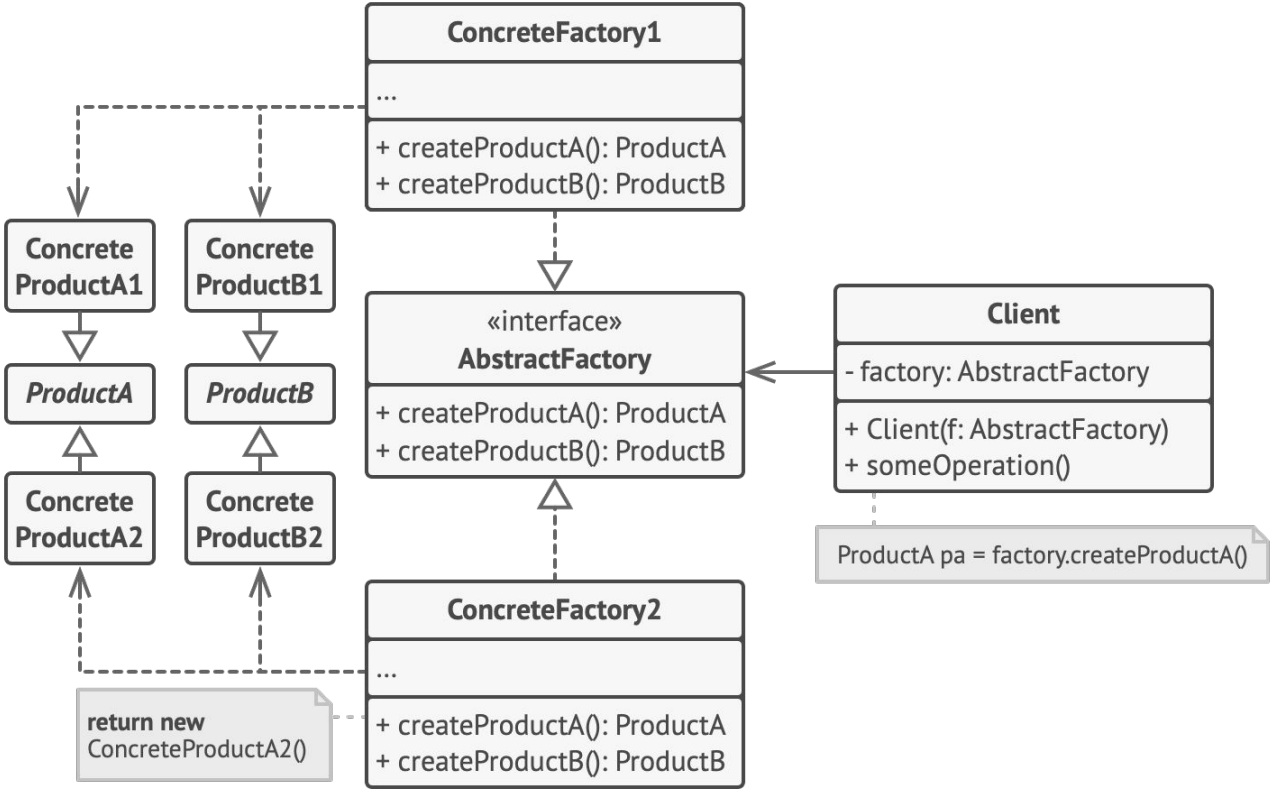
Abstract Factory

Abstract Factory este un pattern cretional care ne lasă să realizăm familii de obiecte care au legături între ele fără să specificăm clasele lor.

Acest pattern este folositor când vrem să creăm obiecte de mai multe tipuri, dar care toate pleacă de la un numitor comun. Astfel se folosește Factory Method care definește o metodă pentru creare de obiecte.

Acest pattern ne ajută ca la run time să ne decidem ce obiecte creăm în funcție de anumiți parametrii.

Astfel creăm o interfață pentru crearea obiectelor, dar lăsăm subclasele să decidă ce clase o să fie instanțiate.

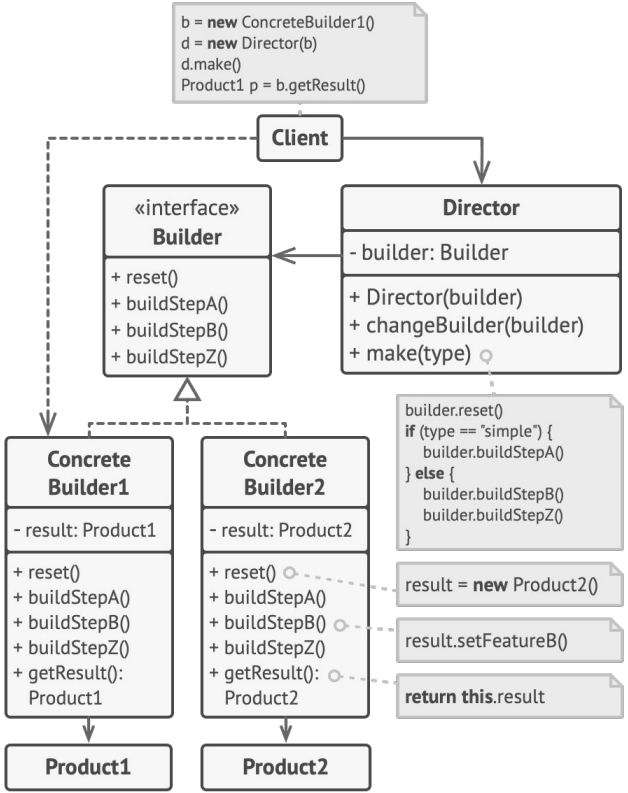


Abstract Factory Pattern

Builder

Builder este un pattern creational folosit pentru a separa construirea obiectelor complexe. Acest pattern este folosit când vrem să construim obiecte complexe dar nu vrem să avem constructori cu foarte mulți parametrii.

Astfel creăm un obiect intermediar a cărui metode definesc obiectul final bucată cu bucată. Acest pattern ne ajută să amânăm crearea unui obiect până ajungem în momentul în care știm toate detaliile despre obiectul respectiv.



Builder Pattern

Summary

De Reținut

Design pattern-urile sunt soluții tipice pentru probleme care apar în mod frecvent în design-ul software-ului.

Există trei tipuri de patterns:

- Creational
- Structural
- Behavioral

Creational patterns: Singleton, Factory Method, Abstract Factory, Builder, Prototype.

Singleton: asigură că există doar un singur obiect pentru clasa respectivă

Abstract Factory: care ne lasă să realizăm familii de obiecte care au legături între ele fără să specificăm clasele lor.

Builder: folosit pentru a separa construirea obiectelor complexe.

Exerciții

Exerciții

Să se realizeze o implementare pentru fiecare din cei 3 patterns prezentați.

Pentru factory să se realizeze pe un exemplu cu mașini, iar pentru builder pentru un exemplu cu burgers.

În cod să aveți o zonă comentată în care să prezentați pros and cons pentru fiecare dintre ei. **3p/pattern**

Proiect POO Discuție

Crăciun fericit!

La mulți ani!

Take care!