UNIVERSITÉ CHEIKH ANTA DIOP DE DAKAR

ECOLE SUPÉRIEURE POLYTECHNIQUE

DÉPARTEMENT GÉNIE INFORMATIQUE

PATRONS DE CONCEPTION

COURS INTRODUCTIF

Formateur

Dr Mouhamed DIOP

mouhamed.diop@esp.sn

Objectifs

- A l'issue du cours, l'étudiant devra :
  - Comprendre la notion de patrons de conception
  - Connaitre les principaux patrons de conception
  - Pouvoir implémenter les patterns de base
  - Connaitre le contexte d'utilisation de chaque pattern

Contenu

- Introduction
- Définition d’un patron de conception
- Classification des différents patrons de conception
- Comment choisir un patron de conception ?
- Comment utiliser les patrons de conception ?

Contexte

- Lors du développement de grosses applications, des problèmes de « conception » risquent d'apparaitre
  - Il faudra trouver des réponses à un certains nombres de questions
    - Comment choisir les bonnes classes ?
    - Comment faire intéragir les classes de manière optimale ?
      - Comment les faire coopérer tout en les gardant suffisamment autonomes pour être réutilisables ?
    - Comment rédiger le code de sorte qu'il soit facilement maintenable ?
      - Comment faire évoluer le code sans remettre en cause l'existant ?
  - Les patrons de conception aident le développeur à mieux gérer ces problèmes

Définition

- Un patron de conception est une manière de résoudre, de façon indépendante d'un langage, à l'aide d'une organisation appropriée de classes, un problème qui se pose de façon récurrente
  - Il consiste en un schéma d'objets (ou de classes) qui forme une solution à un problème connu et fréquent en conception OO
    - Solution dont la conception provient de l'expérience de programmeurs
    - Basée sur les bonnes pratiques de la programmation orientée objets (héritage, encapsulation, polymorphisme, composition, etc.)
  - Il indique comment structurer les classes et leurs objets pour résoudre certains problèmes de conception OO

Un peu d'histoire

- Bien qu'on associe assez souvent la notion de patrons de conception à la programmation, ce concept provient du monde de l'architecture
  - Lors de la parution en 1977 de l'ouvrage « A pattern Language »
    - Il a été rédigé par plusieurs auteurs dont Christopher Alexander
- Il a été repris par les développeurs donnant lieu à la publication de plusieurs ouvrages
  - Le premier en date et parmi les plus célèbres a été rédigé par quatre auteurs surnommés Gang of Four (abrégé souvent en GoF)
    - Il a été publié en 1995 dans le livre intitulé « Design Pattern – Elements of Reusable Object-Oriented Software »

Avantages liés à l'utilisation des patterns

- Partage d'un vocabulaire puissant
- Expression de plus de choses en moins de mots
- Rester longtemps dans la conception sans devoir plonger dans les détails de l'implémentation
- Avancement rapide de l'équipe grâce à l'élimination des malentendus
- Vocabulaire commun incitant les développeurs juniors à apprendre vite en conception

Classification des patterns (1/2)

- L'ouvrage de référence du GoF décrit 23 patterns d'intérêt général
  - D'autres patterns ont été proposés par la suite dont certains qui sont spécifiques à des domaines bien précis
    - C'est l'exemple des patterns J2EE de Sun ou des patterns JSP
- Les patterns sont généralement classés en catégories
  - La classification la plus utilisée étant celle du GoF
  - Certains auteurs utilisent des classifications différentes
    - Suivant l'auteur, un même pattern peut se trouver dans deux catégories différentes

Classification des patterns (2/2)

- Le GoF classe les patterns en trois grandes catégories
  - Les patterns de construction (ou de création)
    - Permettent d'isoler la création des objets de leur utilisation
  - Les patterns de structuration (ou de structure)
    - Permettent d'assembler des classes ou des objets individuels en des structures plus complexes
  - Les patterns de comportement
    - Portent sur l'utilisation d'algorithmes et sur l'affectation de responsabilités aux objets

Les patterns de construction (1/2)

- Ils ont pour but d'organiser la création des objets
  - Abstraient les mécanismes d'instanciation des objets
    - Permet à un client d'utiliser un objet sans vraiment connaitre son type effectif
    - Permet d'écrire du code plus souple, puisque indépendant des classes concrètes
  - Rend les systèmes indépendants de la manière dont les objets sont crées en leur offrant des mécanismes d'instanciation des classes concrètes
  - Encapsule l'utilisation de classes concrètes en favorisant l'utilisation des interfaces

Les patterns de construction (2/2)

- On en distingue cinq dans la classification du GoF
  - Abstract Factory
  - Builder
  - Factory Method
  - Prototype
  - Singleton

Les patterns de structuration (1/2)

- Facilitent l'organisation de la hiérarchie des classes et leurs relations
  - Permettent de combiner des classes ou des objets pour créer de nouvelles structures
- Leur mise en oeuvre peut s'appuyer sur l'héritage ou sur la composition
  - Si l'héritage est utilisée, la structure est définie à la compilation
  - Si la composition est utilisée, on obtient une structure dynamique susceptible d'être modifiée durant l'exécution

Les patterns de structuration (2/2)

- Ils sont sept dans la classification du GoF
  - Adapter
  - Bridge
  - Composite
  - Decorator
  - Façade
  - Flyweight
  - Proxy

Les patterns de comportement (1/2)

- Ils permettent entre autre de :
  - Organiser les interactions entre les objets
  - Répartir les traitements entre les objets
- Ils sont onze dans la classification du GoF
  - Chain of responsability
  - Command
  - Interpreter
  - Iterator
  - Mediator

Les patterns de comportement (2/2)

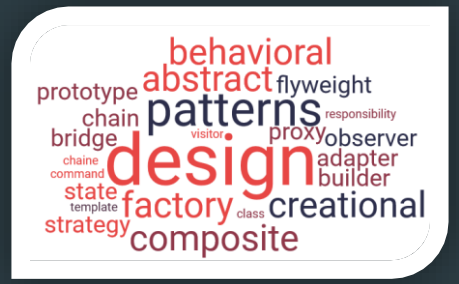
- Ils sont onze dans la classification du GoF (suite)
  - Memento
  - Observer
  - State
  - Strategy
  - Template Method
  - Visitor

Comment choisir les patterns de conception ?

- Bien maitriser chaque pattern
  - Connaitre leur schéma générique UML, en dehors de tout contexte particulier
  - Connaitre la liste de chacun de leurs participants / composants
  - Connaitre les collaborations existant entre leurs composants
- Déterminer si le pattern se rapproche de façon pertinente du problème rencontré
- Adapter la structure générique du pattern au contexte du système
  - Renommer les méthodes et les classes
  - Raffiner les relations entre les classes

Comment utiliser les patterns de conception ?

- On les utilise comme on le fait avec les API :
  - On peut les intégrer dans un programme pour tirer profit des fonctionnalités qu'ils offrent
    - Gain de temps dans la construction du modèle de développement
    - Gain de qualité dans la structuration du code
    - Ils sont souples, faciles à comprendre et à mettre en oeuvre
  - Ils ne permettent pas de structurer le code à eux seuls
    - Ils ne s'intègrent pas directement dans le code
    - Il faudra adapter leur structure générique au contexte du système



Le Pattern Singleton