Java SE: Design pattern

- Cos’è un design pattern
- Definizione formale
- Classificazione
- Esempi d’uso
  - Nelle librerie standard Java
  - Codice di esempio
- Progetto di riferimento
  - https://github.com/egalli64/mpjp *(modulo 6)*

Design pattern

- È una soluzione verificata a un problema comune
- Progettazione più flessibile e modulare
- Documentazione del codice più intuitiva
- Testi storici
  - A pattern language – Christopher Alexander – 1977
  - Using Pattern Languages for Object-Oriented Programs – Kent Beck, Ward Cunningham – 1987
  - Design Patterns – Erich Gamma et al. (GoF: Gang of Four) – 1994

Definizione

- Nome
  - Descrive il pattern e la sua soluzione in un paio di parole
- Problema
  - Contesto e ragioni per applicare il pattern
- Soluzione
  - Elementi del design, relazioni, responsabilità e collaborazioni
- Conseguenze
  - Risultato, costi e benefici, impatto sulla flessibilità, estensibilità, portabilità del sistema
  - Possibili alternative

Classificazione

- Scopo
  - Creazionali
    - Creazione di oggetti
  - Strutturali
    - Composizione di classi e oggetti
  - Comportamentali
    - Interazione tra oggetti o classi
    - Flusso di controllo
- Raggio d’azione
  - Classe (statico)
    - Ereditarietà
  - Oggetto (dinamico)
    - Associazione
    - Interfacce

Creazionali

- Singleton
  - Assicura che esista una sola istanza di una data classe
- Factory method – costruttore virtuale
  - Delega la decisione sul tipo di un oggetto da creare alle sottoclassi
- Abstract factory
  - Interfaccia per la creazione di oggetti correlati in diversi contesti
- Builder
  - Creazione di un oggetto complesso separata dalla sua rappresentazione
- Prototype
  - Creazione di un oggetto per clonazione
- ...

Strutturali

- Façade
  - Fornisce una interfaccia unica ad un sistema complesso
- Composite
  - Albero di oggetti semplici e composti, trattati allo stesso modo dal client
- Decorator
  - Aggiunge metodi a un oggetto dinamicamente
- Adapter
  - Adatta l’interfaccia di una classe a una diversa richiesta
- Proxy
  - Fornisce un oggetto semplificato che ne rappresenta un altro
- …

Comportamentali

- Observer – Pub-Sub
  - notifica i cambiamenti dello stato di un oggetto a molti sottoscrittori
- Memento
  - Persistenza dello stato di un oggetto
- Iterator
  - Accesso sequenziale alle componenti interne di un oggetto senza esporne la struttura
- Strategy
  - Permette di incapsulare algoritmi e delegare al client la scelta d’uso
- Mediator
  - Gestione dell’interazione del client con oggetti diversi
- ...

Altri pattern

- Funzionali
  - Generator
- Concorrenza
  - Future e promise
- Architetturali
  - Model View Controller (MVC)
  - Client/Server
  - Data Access Object (DAO)

Singleton

- È necessario che esista un'unica istanza di una classe
- Le soluzioni in Java normalmente richiedono che il Ctor sia privato (o protetto), e che un metodo statico (factory) sia responsabile dell'istanziazione e dell'accesso
- Una semplice implementazione eager: una proprietà e un getter, entrambi statici
  - L'inizializzazione di static member è thread safe
  - Static initializer se la creazione dell'oggetto è complessa
- Ancora più semplice: usando una classe enum
- Per evitare l'istanziazione dell'oggetto fin quando non sia necessaria: lazy
  - Getter sincronizzato: ogni accesso paga il costo della sicurezza multithreading
  - Getter con doppio check sull'istanza: blocco sincronizzato solo quando necessario
  - Inner class statica: usa il meccanismo standard Java per garantire la sincronizzazione

Static factory method

- Uso di un metodo statico al posto di un ctor → Factory method
  - classe dell'oggetto creato determinato da parametri passati
- Controllo nella creazione → Singleton
- il nome del metodo dà maggior chiarezza al suo scopo
  - Creazione via conversione: from(), valueOf()
  - Accesso a una istanza regolamentata: getInstance()
  - Creazione di una nuova istanza: newInstance(), create()
- Molto usato in Java
  - String.valueOf()
  - LocalDate.of()
  - …

Decorator

- Modifica il comportamento di un oggetto a runtime
- Viene passato al ctor di una classe derivata che implementa una interfaccia più ricca
- Visto in Java I/O
  - File → FileWriter → PrintWriter
  - File → FileReader → BufferedReader

Iterator

- Forte accoppiamento tra iterato e iteratore
- In Java è disponibile l’interfaccia Iterator pensata per il Java Collections Framework
- Gli iteratori in Java sono monouso
- Due metodi fondamentali
  - Esiste un successore? hasNext()
  - Ritorna il successore: next()
    - NoSuchElementException in caso di uso scorretto

Strategy

- La strategia da utilizzare in un dato algoritmo viene passata dall'utente come parametro
- Esempio Java Collections Framework
  - List.sort() richiede come parametro un Comparator
    - Definisce la strategia da applicare nel sorting
    - Metodo che accetta in input due parametri (elementi della lista) e ritorna un valore negativo, zero, o positivo, per indicare che il primo parametro è minore, uguale o maggiore del secondo
    - Se viene passato null, si applica l'ordinamento naturale per il tipo corrente