

Professional Skills
Agile Fundamentals
Jira
Git
Databases Introduction
Java Beginner
Maven
Testing (Foundation)
Java Intermediate
<div><div></div>Optionals</div>
<div><div></div>JDBC CRUD</div>
<div><div></div>Exceptions</div>
<div><div></div>SOLID Principles</div>
<div><div></div>Single Responsibility</div>
<div><div></div>Open/Closed</div>
<div><div></div>Liskov Substituiton</div>
<div><div></div>Interface Segregation</div>
<div><div></div>Dependency Inversion</div>
<div><div></div>Best Practice</div>
<div><div></div>Design Patterns</div>
<div><div></div>Creational Design Patterns</div>
<div><div></div>Structural Design Patterns</div>
<div><div></div>Behavioural Design Patterns</div>
<div><div></div>Collection & Map</div>
<div><div></div>HashSets</div>
<div><div></div>HashMaps</div>
<div><div></div>Enums</div>
<div><div></div>Logging</div>
<div><div></div>Generics</div>
<div><div></div>Lambda Expressions</div>
<div><div></div>Streams</div>
<div><div></div>Complexity</div>
<div><div></div>Input and Output</div>
<div><div></div>Local Type Inference</div>
HTML
CSS

Design Patterns

Contents

- [Overview](#)
- [Tutorial](#)
- [Exercises](#)

Overview

Design Patterns represent best practices developed and used by experienced software developers.

Design patterns can generally be found with object-oriented languages. In general, they are solutions to common problems that developers faced during development. Think of them as "templates" that you can use to make your code more efficient or effective.

There are currently 23 well-known Design patterns, [which are all from this 1994 textbook](#).

The usage of design patterns focus in two areas:

- Standardisation** – design patterns offer a standard of terms in programming.
- Best practice** – tested methods of programming that have been refined throughout the years.

Design patterns can be classified into 3 categories:

- [Creational](#) – focuses on creating objects, whilst hiding the logic that creates them.
- [Structural](#) – focuses on the structure of classes and objects, and emphasises the use of inheritance and interfaces to define the composition of objects.
- [Behavioural](#) – focused on the communication between objects.

Tutorial

Not applicable.

Exercises

Not applicable.

Javascript
Spring Boot
Selenium
Sonarqube
Advanced Testing (Theory)
Cucumber
MongoDB
Express
NodeJS
React
Express-Testing
Networking
Security
Cloud Fundamentals
AWS Foundations
AWS Intermediate
Linux
DevOps
Jenkins Introduction
Jenkins Pipeline
Markdown
IDE Cheatsheet