Design Patterns

Denis Ponizovkin

April 11, 2019

Table of contents

Design Pattern

Definition

In software engineering, a software design pattern is a general, reusable solution to a commonly occurring problem within a given context in software design.

Roots

The idea was introduced by the architect Christopher Alexander and has been adapted for various other disciplines, most notably computer science.

> The elements of this language are entities called patterns. Each pattern describes a problem that occurs over and over again in our environment, and then describes the core of the solution to that problem, in such a way that you can use this solution a million times over, without ever doing it the same way twice. — Christopher Alexander

From architecture to software

GoF

Christopher Alexander is the architect who first studied patterns in buildings and communications and developed "pattern language" for generating them. His work has inspired us time and again. There are many ways in which our work is like Alexander's. Both are based on observing existing systems and looking for patterns in them. Both have templates for describing patterns. Both rely on natural language and lots of examples to described patterns rather than formal languages, and both give rationales for each pattern. — Design Patterns.

Why do I need to study patterns

- Reusability;
- The use of common terminology;
- Design patterns provide us with an abstract, high-level view of both the problem and the whole process of object-oriented development.
- design patterns allow a developer or a group of developers to find design solutions for complex problems without creating a cumbersome class inheritance hierarchy.

Other advantages

- Efficiency improvement of single developers and whole group of developers;
- The use of many design patterns also allows you to create more modifiable and flexible software;
- Properly studied design patterns greatly assist in a common understanding of the basic principles of object-oriented design.