Introduction to the Design Patterns

Get a brief overview of the design patterns, their advantages, and their drawbacks.

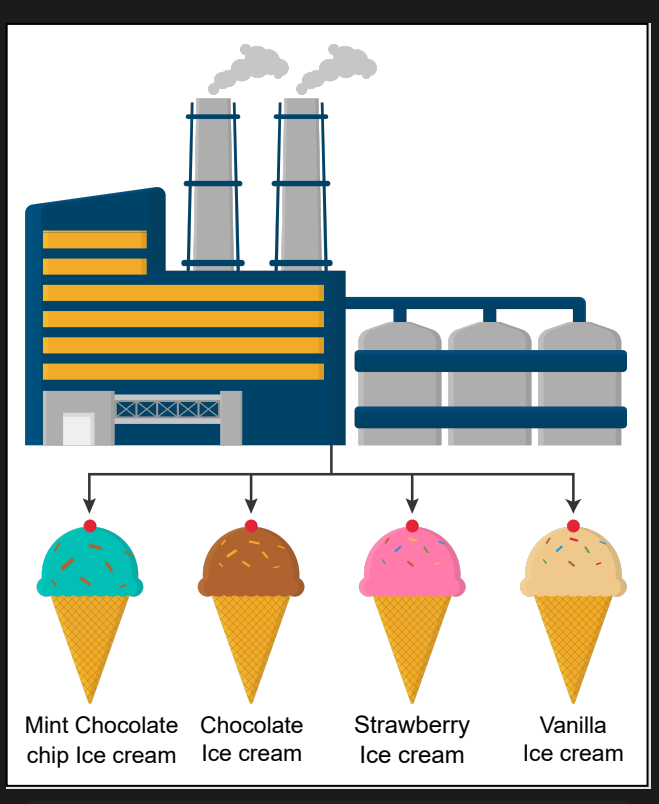
We'll cover the following

- What are design patterns?
- Structure of design patterns
- Advantages of design patterns
- Consequences of unfamiliarity with the design patterns

What are design patterns?

Designing efficient and reusable code is something that all developers strive for. When working on large applications, code structure becomes an integral element where we need to prevent repeating code for similar tasks. However, trying to build upon and adding features to an existing system is quite a challenging task, since a developer needs to know the complete particulars. These include the relationships that each entity possesses or the sort of hierarchy that exists between different entities. Also, developers need to update their code in such a way as to make it easily adaptable to change in the future. In such scenarios, it'll be helpful to have a structure that could be used to solve various common issues. This is where the design patterns come into play.

Design patterns are solutions to frequently occurring real-life problems in software design. They can also be considered customizable templates that can meet the requirements of a particular design problem. Let's take the example of an ice-cream factory that serves as the base of operations to produce various types of ice creams according to the defined requirements. This is similar to how a design pattern can be used as a template in different circumstances.



A real-life example of the design patterns

Structure of design patterns

- **Pattern name:** This is an identification step that describes a design problem.
- **Intent:** This step describes the use case of the particular design pattern.
- **Motivation:** This step illustrates the problem and talks about the inner components in the pattern to solve the problem.
- **Structure:** This step visualizes the structure using a graphical representation of the classes in the pattern.
- **Consequences:** This step describes the trade-offs of a particular pattern.
- **Implementation:** This step illustrates an example of code in any popular programming language.

Advantages of design patterns

The following are some of the advantages of using design patterns:

- They provide correct and efficient solutions since they have been derived and optimized by various experienced programmers over time.
- They are generic templates that can be modified and used for solving different problems.
- They can provide a clean and elegant solution to a large problem by avoiding repetition in the code.
- They provide a template on which the developers can build upon. This allows developers to spend less time on code structure and more on the overall quality of the solution.

Consequences of unfamiliarity with the design patterns

While there are many advantages of using design patterns, it is important to fully understand the depths of design patterns, since unfamiliarity might cause the following issues:

- Design patterns can complicate the architecture of the application if they are managed poorly.
- Developers who are not familiar with them might end up getting confused as to why certain patterns are being used.

Now that we've gotten a flavor of what design patterns are, let's explore some of their different types in the next lesson.

← Back

Quiz: Object-oriented Design Principles

Complete

Next →

Classification of Design Patterns