Design Patterns C

Software design pattern

Reusing design patterns can help to prevent such issues, and enhance code readability for those familiar with the patterns. Software design techniques

In software engineering, a software design pattern or design pattern is a general, reusable solution to a commonly occurring problem in many contexts in software design. A design pattern is not a rigid structure to be transplanted directly into source code. Rather, it is a description or a template for solving a particular type of problem that can be deployed in many different situations. Design patterns can be viewed as formalized best practices that the programmer may use to solve common problems when designing a software application or system.

Object-oriented design patterns typically show relationships and interactions between classes or objects, without specifying the final application classes or objects that are involved. Patterns that imply mutable state may be unsuited for functional...

Design Patterns

Design Patterns: Elements of Reusable Object-Oriented Software (1994) is a software engineering book describing software design patterns. The book was

Design Patterns: Elements of Reusable Object-Oriented Software (1994) is a software engineering book describing software design patterns. The book was written by Erich Gamma, Richard Helm, Ralph Johnson, and John Vlissides, with a foreword by Grady Booch. The book is divided into two parts, with the first two chapters exploring the capabilities and pitfalls of object-oriented programming, and the remaining chapters describing 23 classic software design patterns. The book includes examples in C++ and Smalltalk.

It has been influential to the field of software engineering and is regarded as an important source for object-oriented design theory and practice. More than 500,000 copies have been sold in English and in 13 other languages. The authors are often referred to as the Gang of Four (GoF...

Design pattern

interaction design / human–computer interaction Pedagogical patterns, in teaching Pattern gardening, in gardening Business models also have design patterns. See

A design pattern is the re-usable form of a solution to a design problem. The idea was introduced by the architect Christopher Alexander and has been adapted for various other disciplines, particularly software engineering.

Creational pattern

Creational design patterns are further categorized into object-creational patterns and class-creational patterns, where object-creational patterns deal with

In software engineering, creational design patterns are design patterns that deal with object creation mechanisms, trying to create objects in a manner suitable to the situation. The basic form of object creation could result in design problems or in added complexity to the design due to inflexibility in the creation procedures. Creational design patterns solve this problem by somehow controlling this object creation.

Visitor pattern

Visitor design pattern is one of the twenty-three well-known Gang of Four design patterns that describe how to solve recurring design problems to design flexible

A visitor pattern is a software design pattern that separates the algorithm from the object structure. Because of this separation, new operations can be added to existing object structures without modifying the structures. It is one way to follow the open/closed principle in object-oriented programming and software engineering.

In essence, the visitor allows adding new virtual functions to a family of classes, without modifying the classes. Instead, a visitor class is created that implements all of the appropriate specializations of the virtual function. The visitor takes the instance reference as input, and implements the goal through double dispatch.

Programming languages with sum types and pattern matching obviate many of the benefits of the visitor pattern, as the visitor class is able...

Proxy pattern

Proxy design pattern is one of the twenty-three well-known GoF design patterns that describe how to solve recurring design problems to design flexible

In computer programming, the proxy pattern is a software design pattern. A proxy, in its most general form, is a class functioning as an interface to something else. The proxy could interface to anything: a network connection, a large object in memory, a file, or some other resource that is expensive or impossible to duplicate. In short, a proxy is a wrapper or agent object that is being called by the client to access the real serving object behind the scenes. Use of the proxy can simply be forwarding to the real object, or can provide additional logic. In the proxy, extra functionality can be provided, for example caching when operations on the real object are resource intensive, or checking preconditions before operations on the real object are invoked. For the client, usage of a proxy object...

Modern C++ Design

Modern C++ Design: Generic Programming and Design Patterns Applied is a book written by Andrei Alexandrescu, published in 2001 by Addison-Wesley. It has

Modern C++ Design: Generic Programming and Design Patterns Applied is a book written by Andrei Alexandrescu, published in 2001 by Addison-Wesley. It has been regarded as "one of the most important C++ books" by Scott Meyers.

The book makes use of and explores a C++ programming technique called template metaprogramming. While Alexandrescu didn't invent the technique, he has popularized it among programmers. His book contains solutions to practical problems which C++ programmers may face. Several phrases from the book are now used within the C++ community as generic terms: modern C++ (in contrast to C/C++ style), policy-based design, and typelist.

All of the code described in the book is freely available in his library Loki. The book has been republished and translated into several languages...

Decorator pattern

decorator design pattern is one of the twenty-three well-known design patterns; these describe how to solve recurring design problems and design flexible

In object-oriented programming, the decorator pattern is a design pattern that allows behavior to be added to an individual object, dynamically, without affecting the behavior of other instances of the same class. The decorator pattern is often useful for adhering to the Single Responsibility Principle, as it allows functionality to be divided between classes with unique areas of concern as well as to the Open-Closed Principle, by allowing the functionality of a class to be extended without being modified. Decorator use can be more efficient than subclassing, because an object's behavior can be augmented without defining an entirely new object.

Factory method pattern

overridden by subclasses. It is one of the 23 classic design patterns described in the book Design Patterns (often referred to as the "Gang of Four" or simply

In object-oriented programming, the factory method pattern is a design pattern that uses factory methods to deal with the problem of creating objects without having to specify their exact classes. Rather than by calling a constructor, this is accomplished by invoking a factory method to create an object. Factory methods can be specified in an interface and implemented by subclasses or implemented in a base class and optionally overridden by subclasses. It is one of the 23 classic design patterns described in the book Design Patterns (often referred to as the "Gang of Four" or simply "GoF") and is subcategorized as a creational pattern.

Adapter pattern

adapter design pattern is one of the twenty-three well-known Gang of Four design patterns that describe how to solve recurring design problems to design flexible

In software engineering, the adapter pattern is a software design pattern (also known as wrapper, an alternative naming shared with the decorator pattern) that allows the interface of an existing class to be used as another interface. It is often used to make existing classes work with others without modifying their source code.

An example is an adapter that converts the interface of a Document Object Model of an XML document into a tree structure that can be displayed.