Design Pattern – Overview

设计模式-概述

Design patterns represent the best practices used by experienced object-oriented software developers. Design patterns are solutions to general problems that software developers faced during software development. These solutions were obtained by trial and error by numerous software developers over quite a substantial period of time.

设计模式体现了经验丰富的面向对象软件开发人员的最佳实践。设计模式是软件开发人员在整个软件开发的过程中面临普遍问题的解决方案。这些问题由众多软件开发人员在一段相当可观的时间里通过实验和错误得到的。

What is Gang of Four (GOF)?

什么是“四人帮”

In 1994, four authors Erich Gamma, Richard Helm, Ralph Johnson and John Vlissides published a book titled **Design Patterns - Elements of Reusable Object-Oriented Software** which initiated the concept of Design Pattern in Software development.

These authors are collectively known as **Gang of Four (GOF)**. According to these authors design patterns are primarily based on the following principles of object orientated design.

* Program to an interface not an implementation
* Favor object composition over inheritance

在1994年四个作者Erich Gamma、Richard Helm、Ralph Johnson和John Vlissides出版了一本标题为 **Design Patterns-Elements of Reusable Object-Oriented Software**（《可复用面向对象软件的基础》）的书，书中提出了设计模式在软件开发中概念。

这些作者被统称为“四人帮”。根据这些作者的观念，设计模式主要是基于一下几种面向对象的设计原则。

* 针对接口编程而非针对实现编程
* 优先使用组合而非继承

Usage of Design Pattern

设计模式的使用方式。

Design Patterns have two main usages in software development.

设计模式在软件开发的过程中主要有两种用途。

Common platform for developers

开发者开发公共平台

Design patterns provide a standard terminology and are specific to particular scenario. For example, a singleton design pattern signifies use of single object so all developers familiar with single design pattern will make use of single object and they can tell each other that program is following a singleton pattern.

设计模式提供一个标准的术语和具体的特定的场景。例如，单例模式表示使用单一对象。所以所有的开发者都熟悉单例模式使用单一的对象，并且他们能告诉对方，这个程序遵循单例模式的原则。

Best Practices

最佳实践

Design patterns have been evolved（演变,发展） over a long period of time and they provide best solutions to certain problems faced during software development. Learning these patterns helps unexperienced developers to learn software design in an easy and faster way.

设计模式已经发展了很长一段时间，他们提供了软件开发过程中面临某些问题的最佳解决方案。学习这些模式能帮助一些没有经验的开发者简单、快速的学习软件设计。

Types of Design Patterns

设计模式的种类

As per the design pattern reference book **Design Patterns - Elements of Reusable Object-Oriented Software** , there are 23 design patterns which can be classified in three categories: Creational, Structural and Behavioral patterns. We'll also discuss another category of design pattern: J2EE design patterns.

每种设计模式我们将参考书**Design Patterns - Elements of Reusable Object-Oriented Software，**这里有23中设计模式，它们被归为3种类别：创建型模式，结构型模式和行为型模式。我们还将讨论另外一个类别的设计模式：J2EE design patterns。

|  |  |
| --- | --- |
| **S.N.** | **模式和描述** |
| 1 | **创建型模式**  这类设计模式提供一种隐藏逻辑的方式来创建对象，而不是直接通过new操作来实例化对象。这使程序可以更灵活的决定创建那些对象给用例。 |
| 2 | **结构型模式**  这里设计模式关注类和对象的组成。采用继承的概念组合接口以及采用定义组和对象的方法来获得新功能。 |
| 3 | **行为型模式**  这种模式特别关注两个对象之间的通信。 |
| 4 | **J2EE模式**  这类设计模式特别关注表示层。  这类设计模式已经被Sun Java Center认同。 |

|  |  |
| --- | --- |
| **S.N.** | **Pattern & Description** |
| 1 | **Creational Patterns** These design patterns provide a way to create objects while hiding the creation logic, rather than instantiating objects directly using new opreator. This gives program more flexibility in deciding which objects need to be created for a given use case. |
| 2 | **Structural Patterns** These design patterns concern class and object composition. Concept of inheritance is used to compose interfaces and define ways to compose objects to obtain new functionalities. |
| 3 | **Behavioral Patterns** These design patterns are specifically concerned with communication between objects. |
| 4 | **J2EE Patterns** These design patterns are specifically concerned with the presentation tier. These patterns are identified by Sun Java Center. |