

|什么是设计模式

Q：什么是设计模式？

A：设计模式是指针对软件开发过程中重复发生的问题的解决办法。其中以被称为Gang of Four（GoF）的4人整理出的23种设计模式最为有名。当然，除此之外，还有许多其他的设计模式。请参考以下网页。

·DesignPatterns in Wiki http://c2.com/cgi/wiki？DesignPatterns

◆Patterns Home Page <http://www.hillside.net/patterns/>

|设计模式是万能的吗

Q：设计模式能够解决软件开发中的所有问题吗？

A：不能，每个设计模式都是用于解决软件开发过程中遇到的问题的，但是无论使用什么解决方法，都需要从整体权衡。设计模式并不能解决所有问题。

|如何选择合适的设计

Q：怎样才能选择出合适的设计模式呢？

A：首先必须要明确知道自己的软件中存在什么样的问题。如果问题不够明确，是无法选择出合适的设计模式的。

例如，如果当前面临的问题非常明确，就是“对象太多，浪费了很多内存”，那么我们就会知道“或许Flyweight模式比较合适”。这是因为Flyweight模式是通过共享对象来减少内存使用量的模式。

在学习设计模式时，我们要注意该模式“可以解决什么问题”。

设计模式是理所当然的

Q：所谓设计模式，其解决方法都是理所当然的。我并不认为有值得我们关注和重新学习的价值。为什么设计模式很置要呢？

A：在向经验丰富的开发人员介绍设计模式时，他们会认为这是“理所当然”的。当然是这样的，因为本来设计模式就是开发人员对反复遇到的问题总结出来的解决方法。

设计模式的重要性在于，可以帮助大家很快地掌握那些经验丰富的开发人员才具有的知识和经验。

|设计模式很难背下来

Q:GoF的设计模式一共有23种，很难将它们全部背下来。应该怎么办呢？

A：没有必要全部背下来。因为GoF整理出的23种设计模式并非都是经常使用到的设计模式。

机械地背下这些设计模式没有任何意义。重要的是在自己脑海中理解设计模式是怎样解决问题的。

初级开发人员与设计模式

Q：设计模式对初级开发人员也有密助吗？

A：当然有帮助。

对于刚刚掌握了编程语言，并逐渐开始慢慢地编写一些程序的初级开发人员来说，通过设计模式可以学习到“在进行面向对象编程时，应该注意什么”。例如，通过设计模式，我们可以学到本书中讲解过的可复用性、可替换性、接口（APl）、继承和委托、抽象化等。

此外，设计模式的知识也会对我们自己使用类库有所帮助。这是因为类库中的许多部分都与设计模式有关。

当然，随着自己的技术水平越来越高，开始设计类库时，设计模式的知识对我们的帮助会更大。

|设计模式与模式

Q：除了“设计模式”外，我还常常听到“模式”这个词。两者的意思是相同的吗？

A：严格地说，两者的意思是有区别的。

不论是在什么领域，给“在某种场景下重复发生的问题的解决办法”赋予名字，并整理而成的东西一般都被称为“模式”。

设计模式是适用于软件设计和开发领域的模式，它是模式中的一种。

不过，有时候在软件领域也会将“设计模式”简称为“模式”。

|设计模式与算法

Q:“设计模式”与“算法”是一样的吗？

A：这两者不同，但它们之间有着很深的联系。

算法（algorithm）是指根据输入数据获取输出数据的一系列机械的步骤。算法必须在有限的时间内结束。二分查找算法和快速排序算法都是典型的算法。

我们也可以将算法看作是“解决问题的办法”，将其描述为模式，但是算法并不等于模式。

设计模式不仅与算法有关系，它还与习语（idiom）有关。习语是指编程时经常使用的固定语法（惯用句）。通常，习语具有“高度依赖于编程语言”的特征。与算法一样，习语也可以被看作是

“解决问题的办法”，可以描述为模式，但是习语也并不等于模式。

在本书中，我们使用了具体的示例程序来帮助大家理解设计模式，但设计模式并非具体的实现。这些实现背后的思考方式和解决方法才是设计模式。