

Dive Into

# DESIGN PATTERNS

# Table of Contents

<b>Table of Contents .....</b>	<b>4</b>
<b>How to Read This Book.....</b>	<b>6</b>
<b>INTRODUCTION TO OOP .....</b>	<b>7</b>
Basics of OOP .....	8
Pillars of OOP .....	13
Relations Between Objects.....	20
<b>INTRODUCTION TO DESIGN PATTERNS.....</b>	<b>23</b>
What's a Design Pattern?.....	24
Why Should I Learn Patterns? .....	28
<b>SOFTWARE DESIGN PRINCIPLES .....</b>	<b>29</b>
Features of Good Design .....	30
<b>Design Principles.....</b>	<b>34</b>
§ Encapsulate What Varies .....	35
§ Program to an Interface, not an Implementation.	39
§ Favor Composition Over Inheritance .....	44
<b>SOLID Principles .....</b>	<b>48</b>
§ Single Responsibility Principle.....	49
§ Open/Closed Principle.....	51
§ Liskov Substitution Principle .....	54
§ Interface Segregation Principle.....	61
§ Dependency Inversion Principle .....	64

<b>CATALOG OF DESIGN PATTERNS .....</b>	<b>68</b>
<b>    Creational Design Patterns.....</b>	<b>69</b>
§ Factory Method.....	71
§ Abstract Factory.....	87
§ Builder.....	103
§ Prototype.....	122
§ Singleton.....	136
<b>    Structural Design Patterns.....</b>	<b>146</b>
§ Adapter .....	149
§ Bridge .....	162
§ Composite .....	177
§ Decorator.....	191
§ Facade .....	209
§ Flyweight .....	219
§ Proxy .....	233
<b>    Behavioral Design Patterns .....</b>	<b>246</b>
§ Chain of Responsibility .....	250
§ Command.....	268
§ Iterator .....	289
§ Mediator .....	304
§ Memento .....	320
§ Observer .....	336
§ State.....	352
§ Strategy .....	368
§ Template Method .....	381
§ Visitor .....	393
<b>Conclusion .....</b>	<b>409</b>

# How to Read This Book

This book contains the descriptions of 22 classic design patterns formulated by the “Gang of Four” (or simply GoF) in 1994.

Each chapter explores a particular pattern. Therefore, you can read from cover to cover or by picking the patterns you’re interested in.

Many patterns are related, so you can easily jump from topic to topic using numerous anchors. The end of each chapter has a list of links between the current pattern and others. If you see the name of a pattern that you haven’t seen yet, just keep reading—this item will appear in one of the next chapters.

Design patterns are universal. Therefore, all code samples in this book are written in pseudocode that doesn’t constrain the material to a particular programming language.

Prior to studying patterns, you can refresh your memory by going over the **key terms of object-oriented programming**. That chapter also explains the basics of UML diagrams, which is useful because the book has tons of them. Of course, if you already know all of that, you can proceed to **learning patterns** right away.