Object-Oriented Design Patterns

Outline

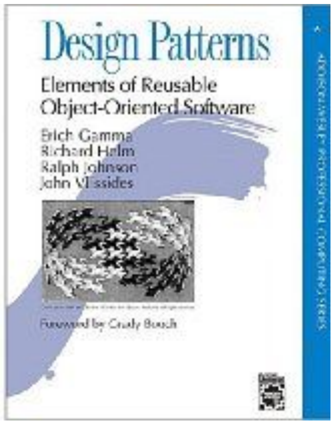
- Overview of Design Patterns
- Four Design Patterns
  - Iterator
  - Decorator
  - Strategy
  - Observer

The Beginning of Patterns

- Christopher Alexander, architect
  - A Pattern Language--Towns, Buildings, Construction
  - “Each pattern describes a *problem* which occurs over and over again in our environment, and then describes the core of the *solution* to that problem”
  - In software engineering, a **design pattern** is a general repeatable solution to a commonly occurring problem in software design
- Other patterns: novels (tragic, romantic, crime), movies genres (drama, comedy, documentary)

“Gang of Four” (GoF) Book

- Design Patterns: Elements of Reusable Object-Oriented Software, Addison-Wesley Publishing Company, 1994
- Written by this "gang of four"
  - Dr. Erich Gamma, then Software Engineer, Taligent, Inc.
  - Dr. Richard Helm, then Senior Technology Consultant, DMR Group
  - Dr. Ralph Johnson, then and now at University of Illinois, Computer Science Department
  - Dr. John Vlissides, then a researcher at IBM
    - Thomas J. Watson Research Center
    - See John's WikiWiki tribute page http://c2.com/cgi/wiki?JohnVlissides



Design Patterns
Elements of Reusable Object-Oriented Software
Erich Gamma
Richard Helm
Ralph Johnson
John Vlissides
Foreword by Grady Booch

Object-Oriented Design Patterns

- This book defined 23 patterns in three categories
  - *Creational patterns* deal with the process of object creation
  - *Structural patterns*, deal primarily with the static composition and structure of classes and objects
  - *Behavioral patterns*, which deal primarily with dynamic interaction among classes and objects

Documenting Discovered Patterns

- Many other patterns have been introduced documented
  - For example, the book **Data Access Patterns** by Clifton Nock introduces 4 decoupling patterns, 5 resource patterns, 5 I/O patterns, 7 cache patterns, and 4 concurrency patterns.
  - Other pattern languages include telecommunications patterns, analysis patterns
  - Patterns are mined at places like Patterns Conferences

ChiliPLoP

- Recent patterns books work shopped at ChiliPLoP, Wickenburg and Carefree Arizona
  - Patterns of Enterprise Application Arhitecture Martin Fowler
  - Patterns of Fault Tolerant Software, Bob Hamner
  - Patterns in XMLPatterns in XML Fabio Arciniegas
  - Patterns of Adopting Agile Development Practices Amr Elssamadisy
  - 2010: Patterns of Parallel Programming, Ralph Johnson
    - 16 patterns and one Pattern Language work shopped

GoF Patterns

- *Creational Patterns*
  - Abstract Factory
  - Builder
  - Factory Method
  - Prototype
  - Singleton
- *Structural Patterns*
  - Adapter
  - Bridge
  - Composite
  - Decorator
  - Façade
  - Flyweight
  - Proxy
- *Behavioral Patterns*
  - Chain of Responsibility
  - Command
  - Interpreter
  - **Iterator**
  - Mediator
  - Memento
  - Observer
  - State
  - **Strategy**
  - Template Method
  - Visitor

Why Study Patterns?

- Reuse tried, proven solutions
  - Provides a head start
  - Avoids gotchas later (unanticipated things)
  - No need to reinvent the wheel
- Establish common terminology
  - Design patterns provide a common point of reference
  - Easier to say, “We could use Strategy here.”
- Provide a higher level prospective
  - Frees us from dealing with the details too early

Other advantages

- Most design patterns make software more modifiable, less brittle
  - we are using time tested solutions
- Using design patterns makes software systems easier to change—more maintainable
- Helps increase the understanding of basic object-oriented design principles
  - encapsulation, inheritance, interfaces, polymorphism

Style for Describing Patterns

- We will use this structure:
  - *Pattern name*
  - *Recurring problem:* what problem the pattern addresses
  - *Solution:* the general approach of the pattern
  - Advantage / Disadvantage
  - *Use Example(s):*
    - 2 examples of this pattern, in Java