PRACTICAL PATTERNS FOR THE PROGRESSING PROGRAMMER

Brent Stewart

Twitter: @brentestewart

about.me/brentstewart

Platinum Sponsors

SyllogisTeks

OAKWOOD
Powering Transformation. Together.

POLARIS SOLUTIONS

Maritz CX

EMERSON
Climate Technologies

Gold Sponsors

World Wide Technology
asynchrony labs

CONCERO TECHNOLOGY GROUP

Sovereign Technologies

VantageLinks

Daugherty BUSINESS SOLUTIONS

KEYHOLE SOFTWARE

stackify

ADAPTIVE SOLUTIONS GROUP

cosentry

UNISYS

WORKFORCE CENTER
SAINT LOUIS UNIVERSITY

Robert Half Technology

Perficient

USI Ungerboeck Software International

signature CONSULTANTS

SWANK MOTION PICTURES, INC.
Tim Swank, Chairman

Microsoft

TEKsystems
Our people make IT possible

twilio

mortgage re:turns

pluralsight
hardcore dev and IT training

Covenant TECHNOLOGY PARTNERS

Byrne Software TECHNOLOGIES, INC.

EQUIFAX

ArchitectNow
Designing Tomorrow's Solutions Today

Silver Sponsors

Amitech Solutions

LogicNP Software
The power of Components

STL TECH TALK

jacobson staffing

WITSINC

redgate
ingeniously simple

ansira

- Co-Founder
  Alien Arc Technologies, LLC
- Co-Organizer of the Kansas City .NET User Group
- Wrote my first application when I was 10 years old and have been hooked ever since.
- Developing professionally since 1996
- Bought my GoF patterns book in 2004 (and it still looks like new)

WHAT IS A DESIGN PATTERN?

"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 such a way that you can use this solution a million times over, without ever doing it the same way twice." – Christopher Alexander

Alexander, et al, A Pattern Language. Oxford University Press, 1977

DESIGN PATTERNS

What they are

- They deal with application and system design
- They are abstractions on top of code
- They deal with relationships
- They deal with problems that have already been solved

What they are not

- They are not language or environment specific
- They are not algorithms
- They are not a specific implementation

FOUR ESSENTIAL ELEMENTS

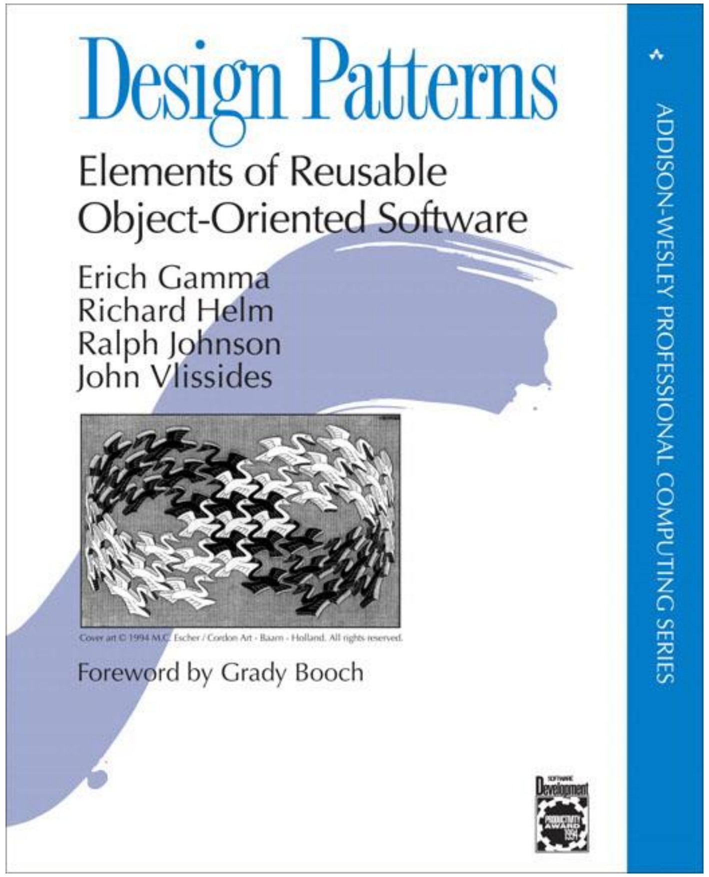
- Pattern Name
- Problem it tries to solve
- Solution
- Consequences

WHY SHOULD I CARE ?

- Common Vocabulary
- Keep from re-inventing the wheel
- Think about design and not get caught up in implementation details
- Provide a starting point for a solution
- Can speed up development in a team

WHO ARE THE GANG OF FOUR?

- Erich Gamma
- Richard Helm
- Ralph Johnson
- John Vlissides
- Written in 1994



Design Patterns

Elements of Reusable Object-Oriented Software

Erich Gamma
Richard Helm
Ralph Johnson
John Vlissides

Cover art © 1994 M.C. Escher / Cordon Art - Baarn - Holland. All rights reserved.

Foreword by Grady Booch

ADDISON-WESLEY PROFESSIONAL COMPUTING SERIES

CREATIONAL PATTERNS

- Abstract Factory
- Builder
- Factory Method
- Prototype
- Singleton

STRUCTURAL PATTERNS

- Adapter
- Bridge
- Composite
- Decorator
- Façade
- Flyweight
- Proxy
- Repository

BEHAVIORAL PATTERNS

- Chain of Responsibility
- Command
- Event Aggregator
- Interpreter
- Iterator
- Media
- Memento
- Null Object
- Observer
- State
- Strategy
- Template Method
- Visitor

DEMO TIME

REPOSITORY PATTERN

Mediates between the domain and data mapping layers using a collection-like interface for accessing domain objects.

Fowler, et al, Patterns of Enterprise Application Architecture, Addison Wesley 2003

DECORATOR PATTERN

Attach additional responsibilities to an object dynamically. Decorators provide a flexible alternative to subclassing for extending functionality.

Gamma, et al, *Design Patterns*. Addison-Wesley, 1994

FACTORY METHOD PATTERN

Define an interface for creating an object, but let subclasses decide which class to instantiate. Factory method lets a class defer instantiation to subclasses

Gamma, et al, *Design Patterns*. Addison-Wesley, 1994

OBSERVER PATTERN

Define a one-to-many dependency between objects so that when one object changes state, all its dependents are notified and updated automatically.

Gamma, et al, *Design Patterns*. Addison-Wesley, 1994

ANTI-PATTERNS

An **anti-pattern** (or **antipattern**) is a common response to a recurring problem that is usually ineffective and risks being highly counterproductive.

Common Anti-Patterns

- Architect Astronaut
- Cargo Cult Programming
- Big ball of mud
- God Object
- Poltergeists
- Magic Numbers/Strings
- Sequential Coupling

RESOURCES

Books

- Design Patterns (Gang of Four)
- Patterns of Enterprise Application Architecture – Martin Fowler
- Head First Design Patterns – Eric Freeman & Elisabeth Freeman

Blogs

- Martin Fowler – www.martinfowler.com
- Uncle Bob Martin – blog.cleancoder.com
- Dino Esposito – software2cents.wordpress.com

Pluralsight courses

- Design Patterns Library – Multiple authors
- Design Patterns On-Ramp – Jeremy Clark

THANK YOU

Brent Stewart

Alien Arc Technologies, LLC

@brentestewart

About.me/brentstewart

Slides are available on my github site