Five Minute Design Patterns

Doug Marttila
Forest and the Trees

May 30, 2009

Template

Factory

Singleton

Iterator

Adapter

Façade

Observer

Command

Strategy

Decorator

Design Patterns

- WTF
- Not a framework
- 23 - not all applicable to Flash and Flex
- You've used them

Why Patterns Matter

- Common language
- CS is good for you
- Speed up your development
- Preexisting documentation
- Refine your solutions
- Explains the why

Best Practices?

- Not really in this presentation
- Beware patterns overload

Best Practices

- Encapsulate what varies
- Program to an interface (or super class)
- Favor composition over inheritance
- Inversion of Control
- ★ Hollywood principle

Categories of Patterns

(Are these useful at all? Not so much.)

- Creational
- Structural
- Behavioral

Template

- Define the skeleton (or invariant parts) of an algorithm in a method, while deferring some steps to subclasses.
- Setup the big picture and leave the details to the person doing the work.

Factory

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

Singleton

- Singleton Pattern ensures a class has only one instance, and provides a global point of access to it.

Iterator

- **The Iterator Pattern provides a way to access the elements of an aggregate object sequentially without exposing its underlying representation**

Adapter

- The Adapter Pattern converts the interface of a class into another interface the clients expect.
- Adapter lets classes work together that couldn'totherwise because of incompatible interfaces.

Façade

- The Façade Pattern provides a unified interface to a set of interfaces in a subsystem. Facade defines a higher level interface that makes the subsystem easier to use.

- Principle of Least Knowledge - talk only to your immediate friends

Observer

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

Command

- The Command Pattern encapsulates a request as an object, thereby letting you parameterize other objects with different requests, queue or log requests, and support undoable operations.

Strategy

- The Strategy Pattern defines a family of algorithms, encapsulates each one, and makes them interchangeable. Strategy lets the algorithm vary independently from the clients that use it.

Decorator - last one!

- The Decorator Pattern attaches additional responsibilities to an object dynamically. Decorators provide a flexible alternative to subclassing for extending functionality.

More Patterns!!!

- State
- Composite
- Bridge
- Proxy
- Interpreter
- Builder
- Abstract Factory
- Memento
- Prototype
- Visitor
- Mediator
- Chain of Responsibility
- Flyweight
- MVC

Resources

- Beerfug.com
- http://www.forestandthetrees.com/designPatterns/
- Head First Design Patterns (the best) Java
- GoF (dry, but what started it all)
- ActionScript 3 with Design Patterns (Lott, Patterson)
- ActionScript 3.0 Design Patterns (Sanders, Cumaranatunge)

Thank You!

Doug Marttila

forestandthetrees.com

doug@forestandthetrees.com

beerfug.com
(first wednesday of every month, beer + code)

FoT!