Pattern-Based Software Development in Java
A Hands-on Introduction to Patterns with Models and Code

The essence and basic structure of a software design solution may be repeated many times, even though the realisation is different in each case. Patterns offer a technique for capturing such recurrence, allowing design experience to be understood, distilled and shared.

The Pattern-Based Software Development in Java course introduces patterns from the ground up, presenting principles as well as concrete examples. It develops understanding through lectures, discussion and hands-on labs, which reinforce the concepts by putting them into practice.

Objectives

- Understand what does and does not go to make up a pattern
- Understand the beneficial role of patterns in all aspects of development
- Learn and use common patterns for object-oriented and large-scale design
- Appreciate patterns from the strategic level down to idiomatic examples in Java

Audience

The course is suitable for software developers familiar with object-oriented principles and practices. Programming experience in Java is assumed, and familiarity with UML is beneficial.

Content

**Software Architecture**
- Defining architecture
- Dependencies
- Stability and change
- Patterns

**Core Pattern Concepts**
- Patterns in software architecture
- Pattern anatomy
- Role of patterns
- Essential pattern form elements
- Common pattern resources

**Introductory Pattern Examples**
- General design patterns in OO
- The Composite pattern
- The Proxy pattern
- Patterns beyond objects

**Combining Patterns**
- Pattern catalogues
- Pattern communities
- Classes and patterns in JUnit
- From individual to multiple patterns
- The Visitor pattern
- Pattern stories and languages

**Pattern Context Dependency**
- Context sensitivity
- The Client Proxy pattern
- Strategic and tactical patterns
- Idioms
- The Immutable Value pattern
- The Combined Method pattern
- The Data Transfer Object (DTO) pattern

**Patterns for Decoupling**
- The Layers pattern and variations
- The Fragile Base Class problem
- The Explicit Interface pattern
- The Separated Interface pattern
- The Bridge pattern

**Patterns for Adaptation**
- The Object Adapter pattern
- The Class Adapter pattern
- The Wrapped Adapter pattern
- The Decorator pattern
- The Template Method pattern
- The Facade pattern

**Patterns for Object Management**
- The Factory Method pattern
- The Disposal Method pattern
- The Manager pattern
- The Leasing pattern
- The Evictor pattern

**Patterns for Pluggability**
- The Strategy pattern
- The Interceptor pattern
- The Null Object pattern
- The Context Object pattern
- The Mock Object pattern
- The Command pattern
- The Command Processor pattern
- The Block pattern

**Patterns for Iteration**
- The Iterator pattern
- The Enumeration Method pattern
- The Collecting Parameter pattern
- The Batch Method pattern
- The Batch Iterator pattern

**Patterns for Object Lifecycles**
- Modal Behaviour
- The Objects for States (State) pattern
- The Methods for States pattern
- The Collections for States pattern

**Patterns for Notification**
- Event flow
- The Observer pattern
- The Model–View–Controller (MVC) pattern
- The Event Channel pattern
- The Pipes and Filters pattern

**Pattern Pitfalls**
- Common pitfalls
- Pattern applicability and quality
- Dysfunctional patterns and applications
- The Getters and Setters ’pattern’
- The Singleton pattern (and avoiding it)

Additional Details

**Duration**
- 4 days (can also be run in 3 days with fewer and shorter lab sessions)

**Setup**
- Projection facilities for a laptop
- Whiteboards
- Flip charts
- Reference cards

**Contact**
- Kevlin Henney
- kevin@curbralan.com
- Curbralan Limited
- +44 117 942 2990