Bridge Pattern

Bryan Hansen

twitter: bh5k | http://www.linkedin.com/in/hansenbryan

Concepts

- Decouple Abstraction and implementation
- Encapsulation, Composition, Inheritance
- Changes in Abstraction won't affect client
- Details won't be right
- Examples:
  - Driver
  - JDBC

Design

Interfaces and Abstract classes

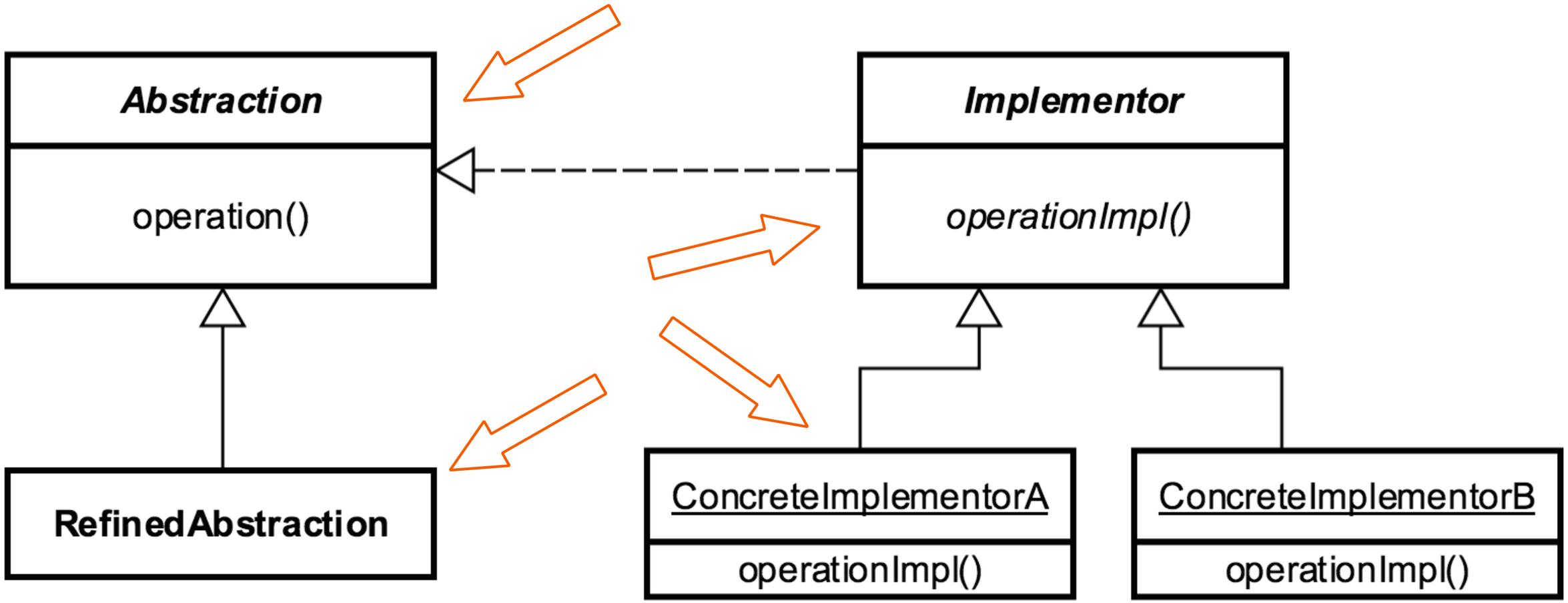
Composition over Inheritance

More than Composition

Expect change from both sides

Abstraction, Implementor, Refined Abstraction, Concrete Implementor

UML


Abstraction
operation()
Implementor
operationImpl()
RefinedAbstraction
ConcreteImplementorA
operationImpl()
ConcreteImplementorB
operationImpl()

Everyday Example - JDBC

```
DriverManager.registerDriver(new org.apache.derby.jdbc.EmbeddedDriver());

String dbUrl = "jdbc:derby:memory:codejava/webdb;create=true";

Connection conn = DriverManager.getConnection(dbUrl);

Statement sta = conn.createStatement();
```

Exercise Adapter

Color and Shape

Color and Shape Bridge

Create Bridge

Another Bridge

Pitfalls

- Increases complexity
- Conceptually difficult to plan
- More than just OO
- What goes where

Contrast

Bridge

- Designed upfront
- Abstraction and implementation vary
- Built in advance
- Complex

Adapter

- Works after code is designed
- Legacy
- Retrofitted
- Provides different interface

Bridge Summary

- Design for uncertainty
- Can be complex
- Provides flexibility
- More than composition