Understanding the Need for Abstraction

Richard Warburton

@richardwarburto   www.monotonic.co.uk

Abstractions

- Way to arrange code
- Suppresses details
- Simplify interactions

Improves maintenance

Demo

Extending the revenue calculation example

Calculate the total revenue using different pluggable methods

```
List<String> words = new ArrayList<>();
words.add(“Hello”);
words.add(“World”);
```

Abstract Data Types

Using the interface type for collections let’s us modify the implementation

Program to an interface, not an implementation

Abstractions for Humans

People Need Focus
Short term memory only holds 5-7 facts

Readability Matters
Simplify each layer down to the minimum viable concepts

Abstractions Compose
Structure so that abstractions depend upon lower levers

Abstractions Don’t Need Interfaces

Structured Abstraction

Split complex operations into simpler methods

Class Abstraction

Delegate responsibility to other classes

Polymorphism

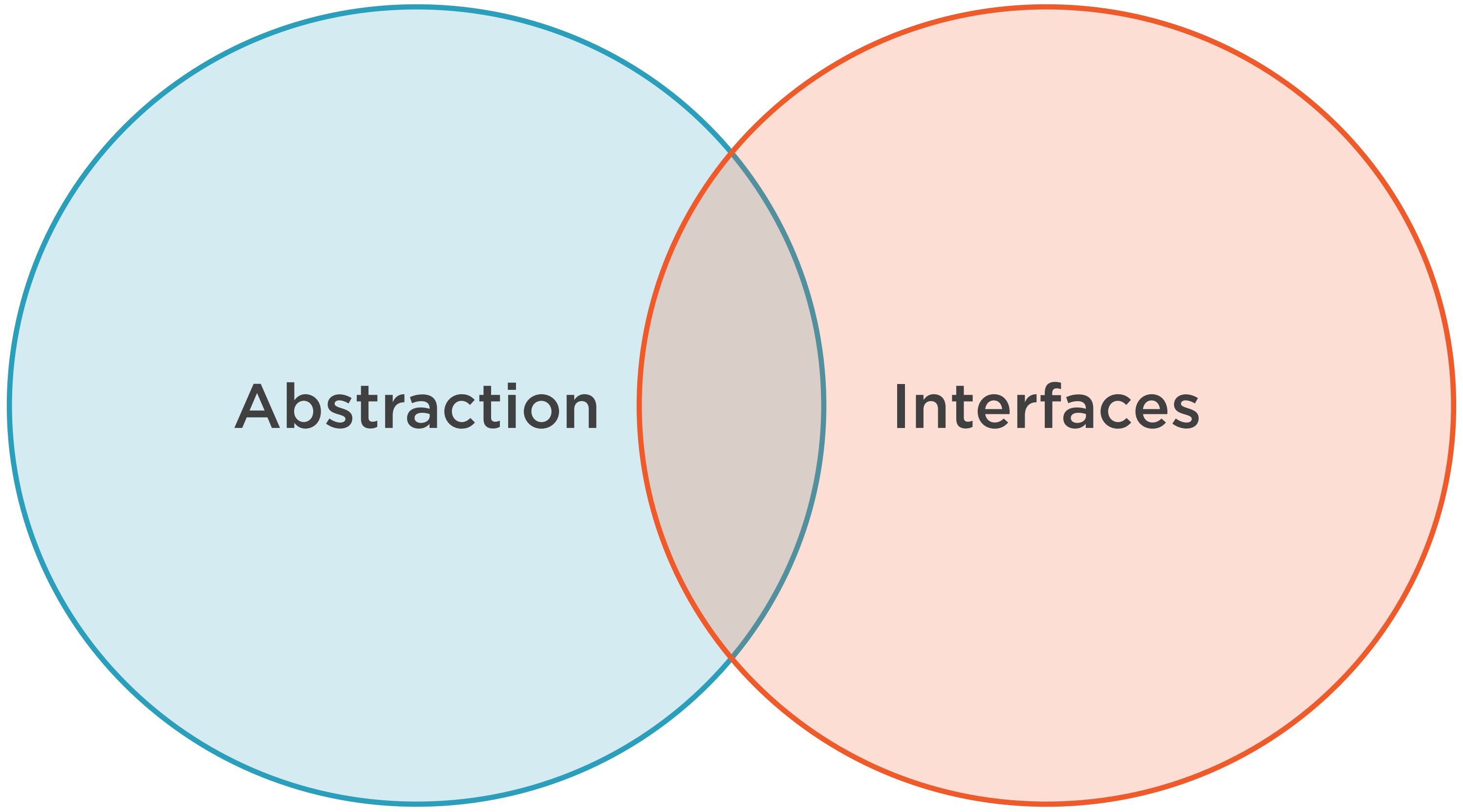
Abstraction layer can have different implementations

Demo

Structured and class based abstraction

Rendering code drawing animals

- An animal
- Facial features
- Arcs, lines, ovals
- Platform specific rendering


Abstraction
Interfaces

Summary

Abstractions are

- Simple
- Composable
- May involve Java `abstract` methods

Enable understandable and maintainable code

Depend on Abstractions, not Details