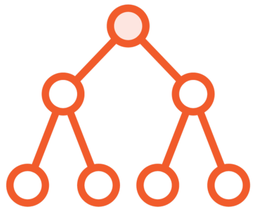
Advanced Classes and Interfaces

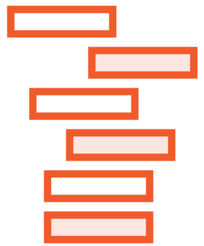
Jesper de Jong
Software Architect

@jesperdj www.jesperdj.com

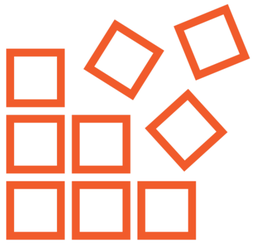
This Module



Nested types



Default, private and static methods in interfaces



Static and instance initializer blocks

Nested types

Types (classes, interfaces, records, enums) that are defined in other places than inside a package.

Overview of Nested Types

Static nested classes

Inner classes

Nested
interfaces
records
enums

Local types

Anonymous classes

Practical examples vs the complete picture

Nested Interfaces, Records and Enums

Nested Classes Versus Other Nested Types

Static nested classes

Inner classes

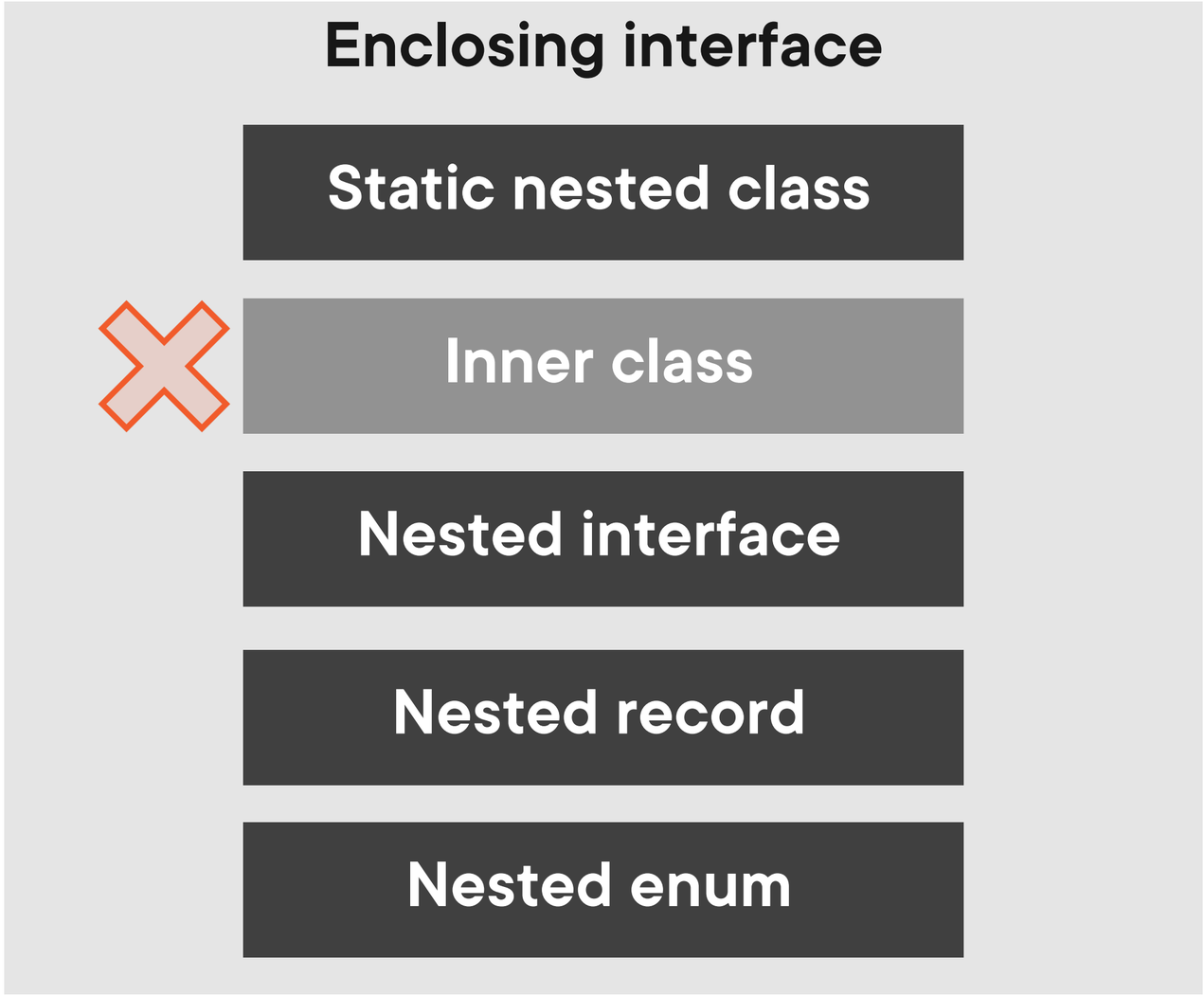
Nested interfaces

Nested records

Nested enums

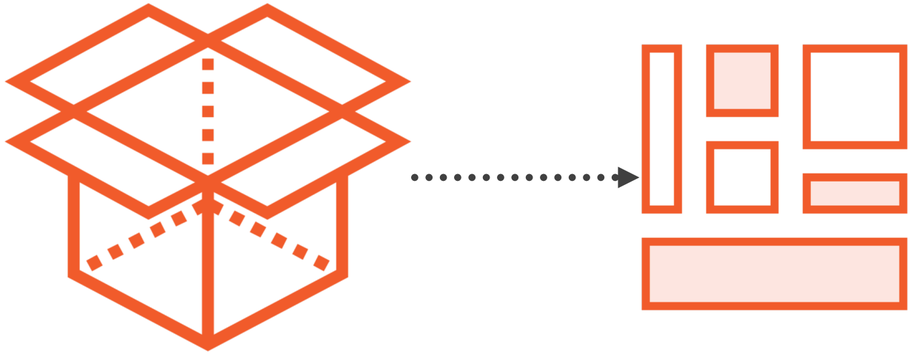
Implicitly static

Nested Types in Interfaces

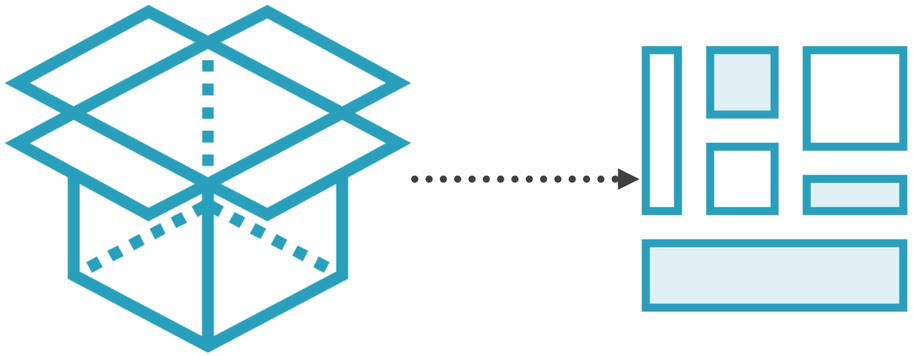


Default, Private and Static Methods in Interfaces

Evolving Interfaces



Version 1.0



Version 2.0

Choose:

1. Break backward compatibility

2. Don’t modify interfaces

Default Methods in Interfaces

```
interface PriceCalculator {

    BigDecimal calculatePrice(Order order);



}
```

Default Methods in Interfaces

```
interface PriceCalculator {

    BigDecimal calculatePrice(Order order);

    default BigDecimal calculateDiscount(Order order)


}
```

Default Methods in Interfaces

```
interface PriceCalculator {

    BigDecimal calculatePrice(Order order);

    default BigDecimal calculateDiscount(Order order) {
        return BigDecimal.ZERO;
    }
}
```

Interfaces Versus Abstract Classes

Interface – Define a common contract for implementing classes

Abstract class – Provide a way to share implementation code

Private and Static Methods in Interfaces

Private methods

Share code between default methods

Static methods

The same as static methods in other types

Summary

- Static nested classes
- Inner classes
- Nested interfaces, records and enums
- Local classes
- Anonymous classes
- Default, private and static methods
- Static and instance initializer blocks

Up Next: Advanced Generics