Making Your Java Code More Object-oriented

ATTAINING EXTENSIBILITY WITH OBJECT-ORIENTED CODE

Zoran Horvat
CEO AT CODING HELMET

@zoranh http://codinghelmet.com

What Makes the Code Object-oriented?

Polymorphism

Encapsulation

Inheritance

What Makes the Code Object-oriented?

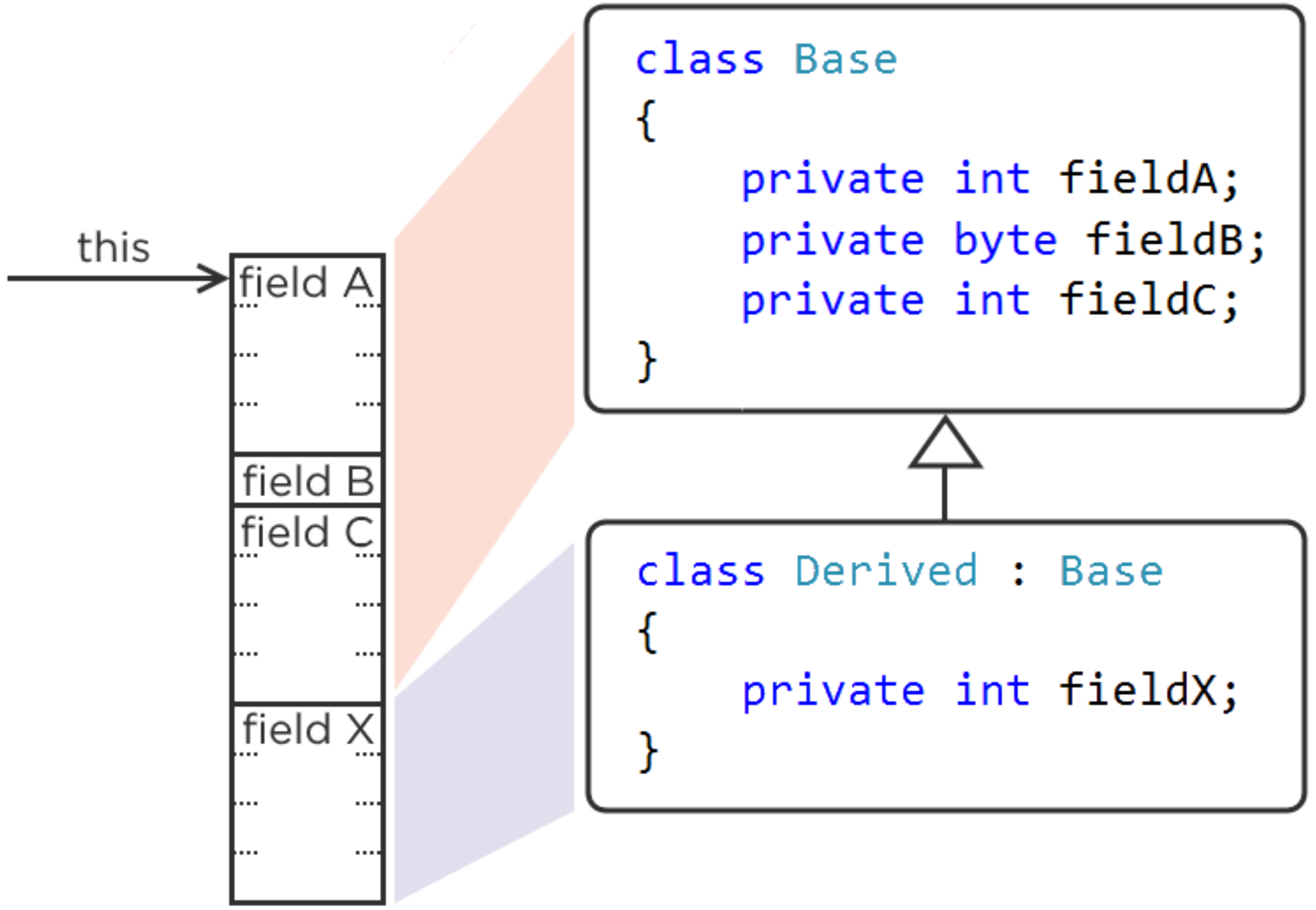
Dynamic dispatch

Implicit this reference

Inheritance

Polymorphism

Encapsulation

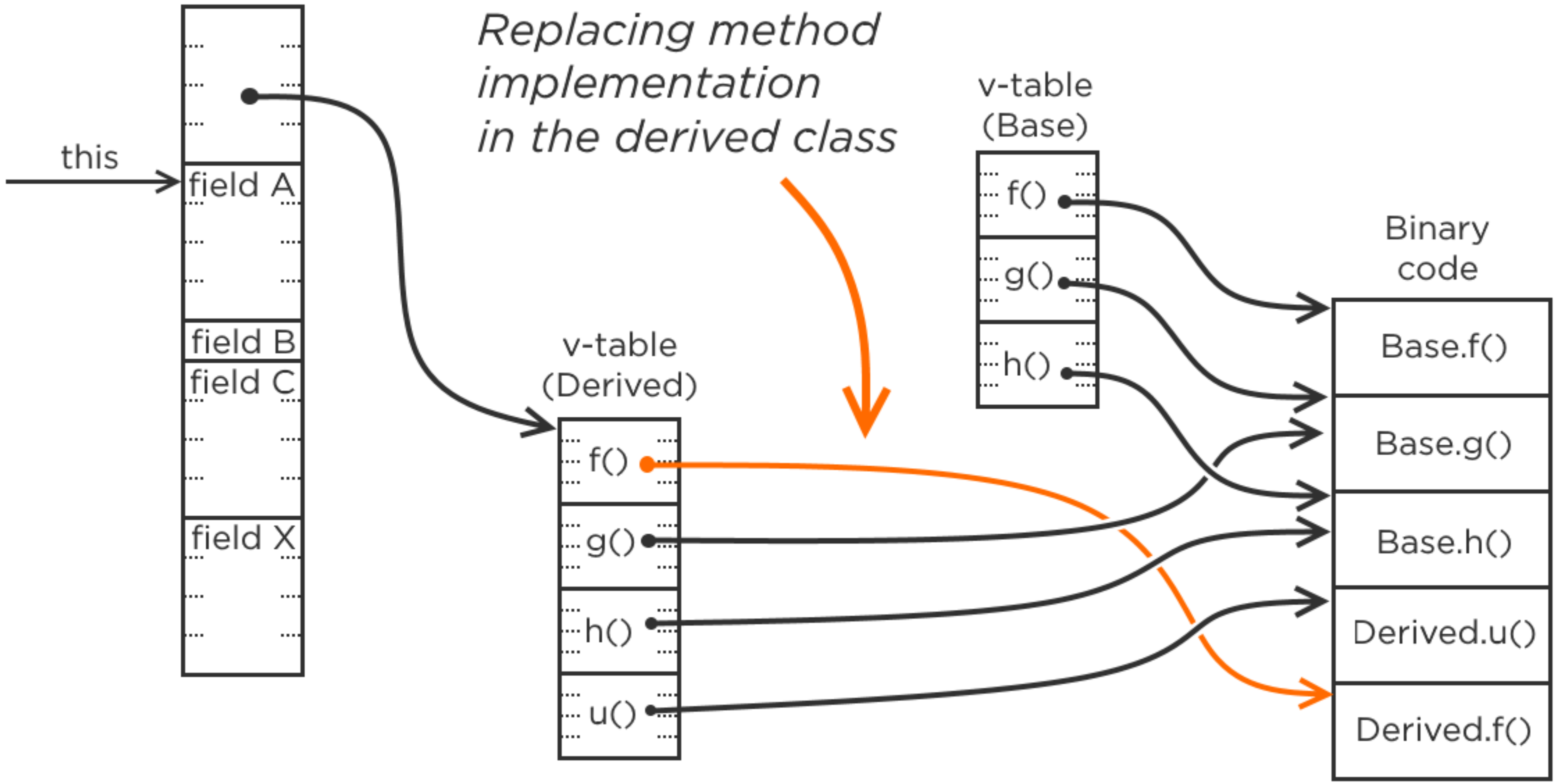


```
class Base
{
    private int fieldA;
    private byte fieldB;
    private int fieldC;
}
```

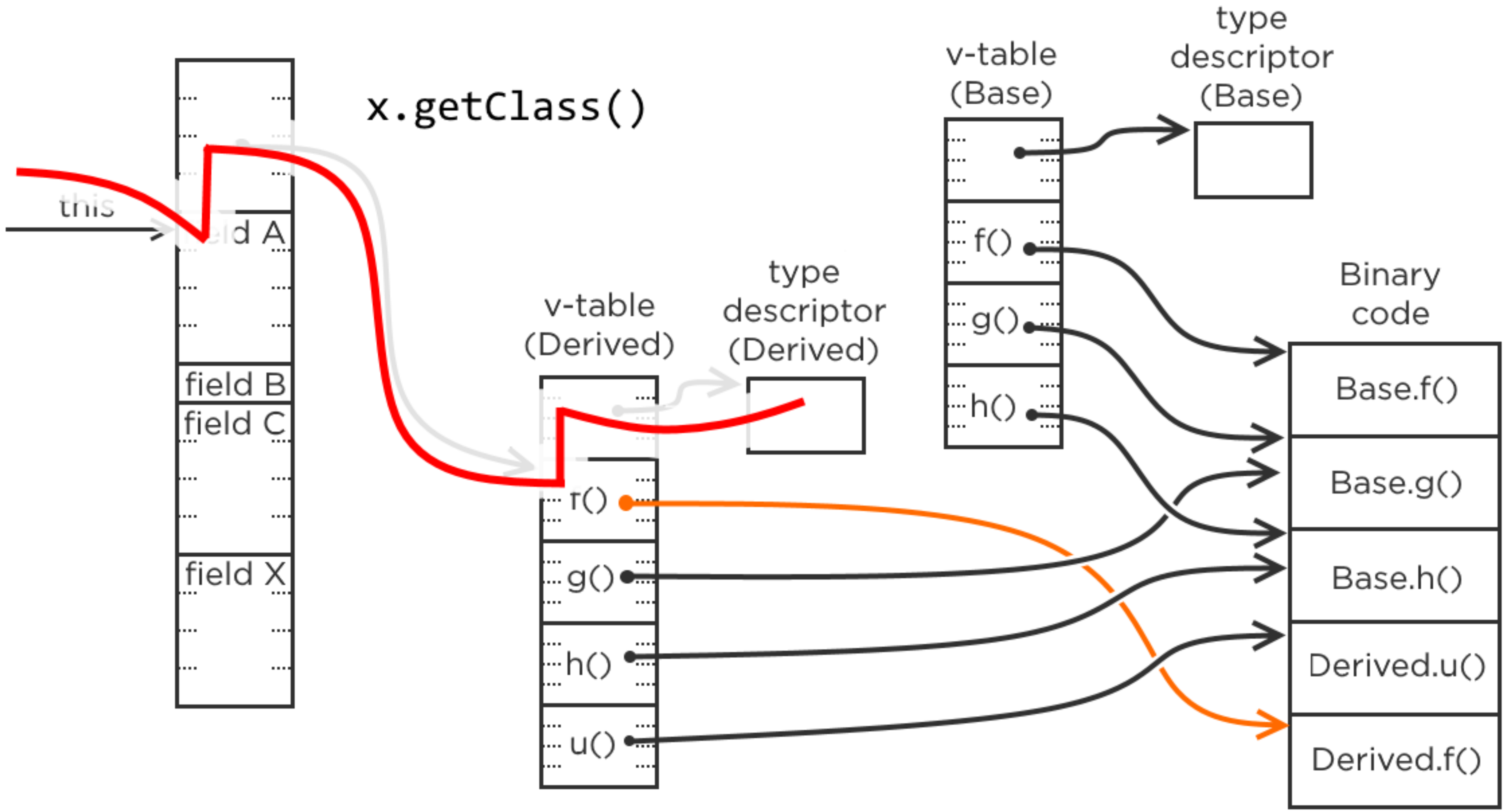
```
class Derived : Base
{
    private int fieldX;
}
```

Method override:

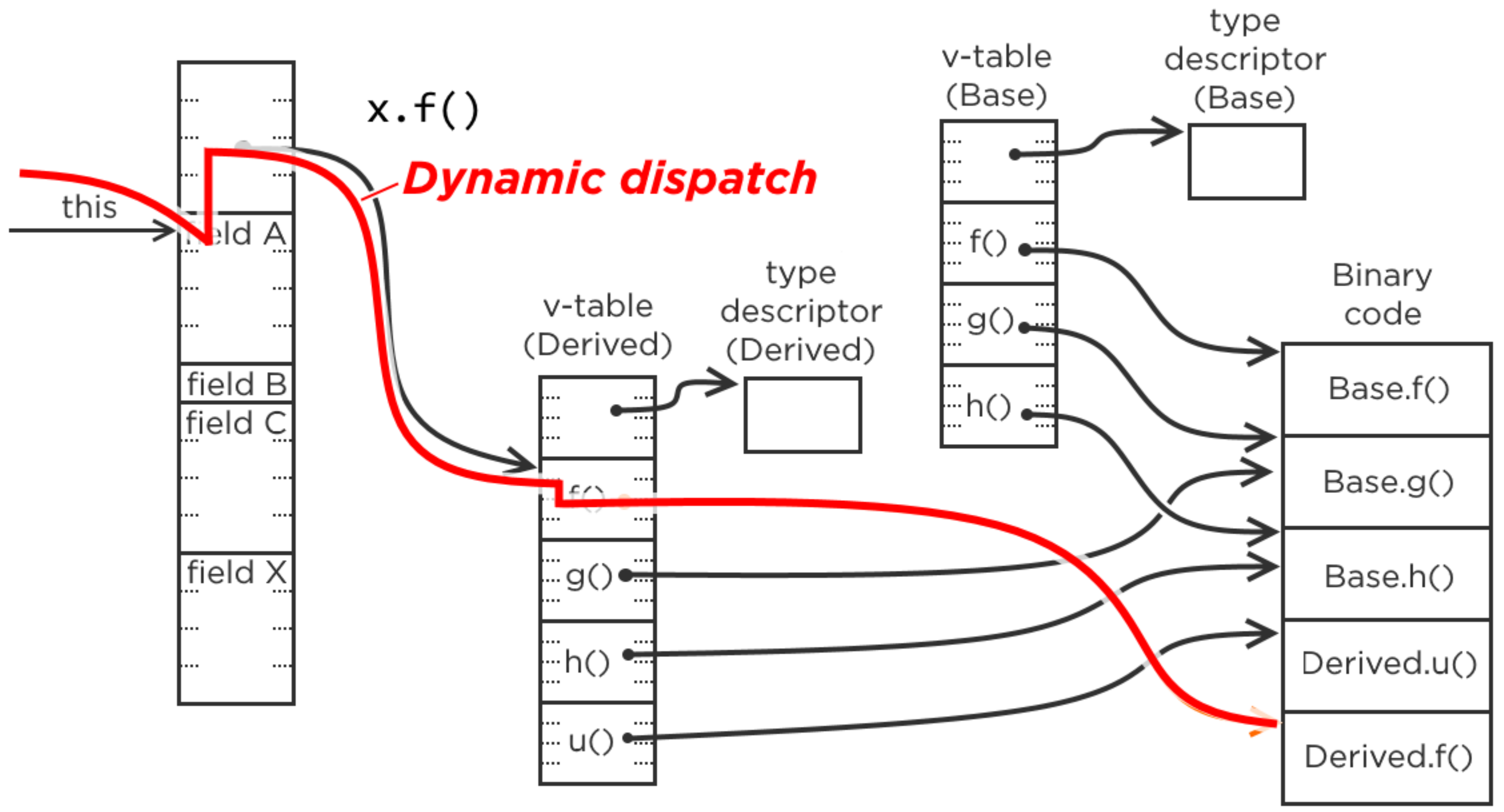
Replacing method implementation in the derived class



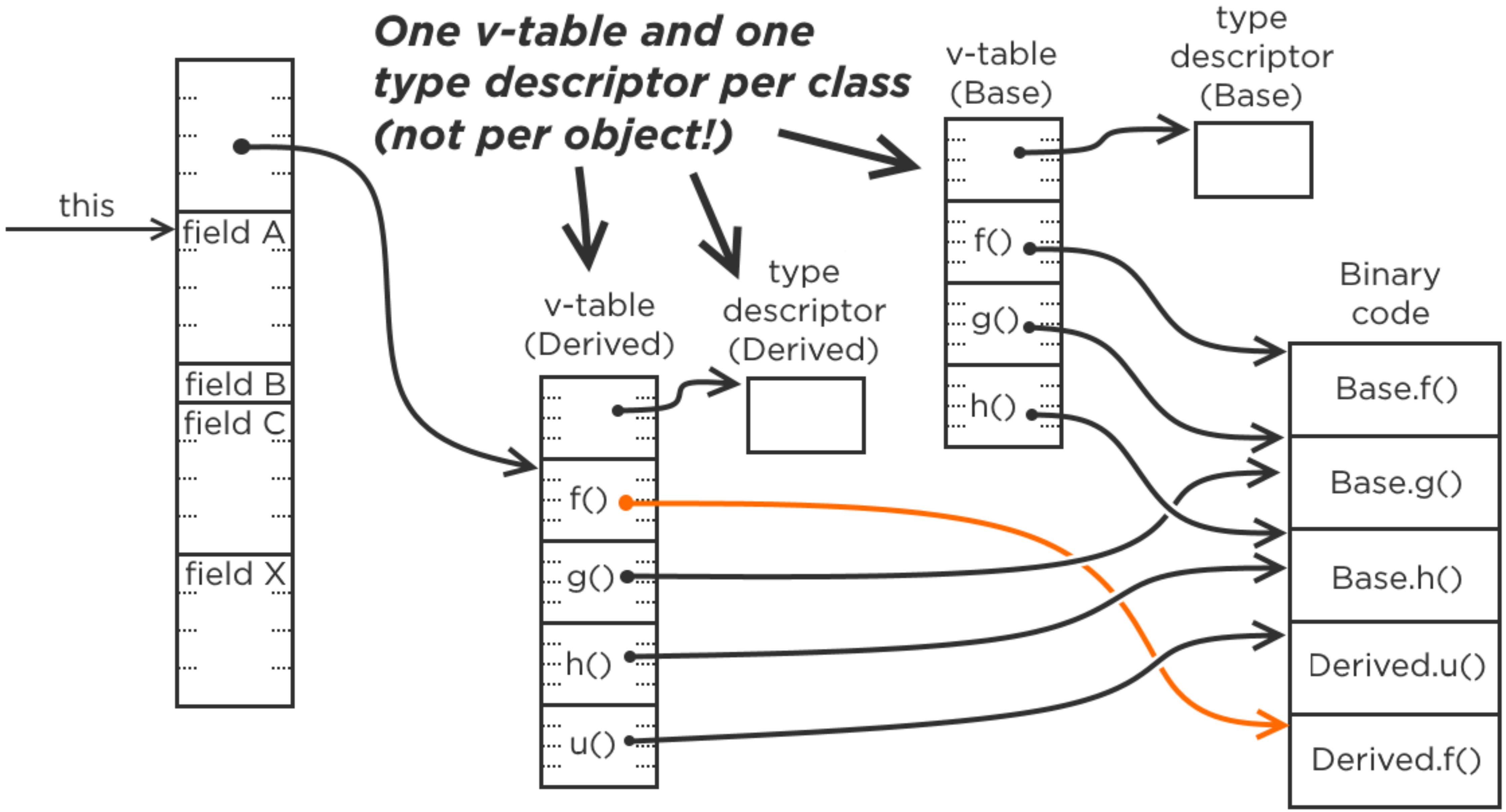
```
x.getClass()
```



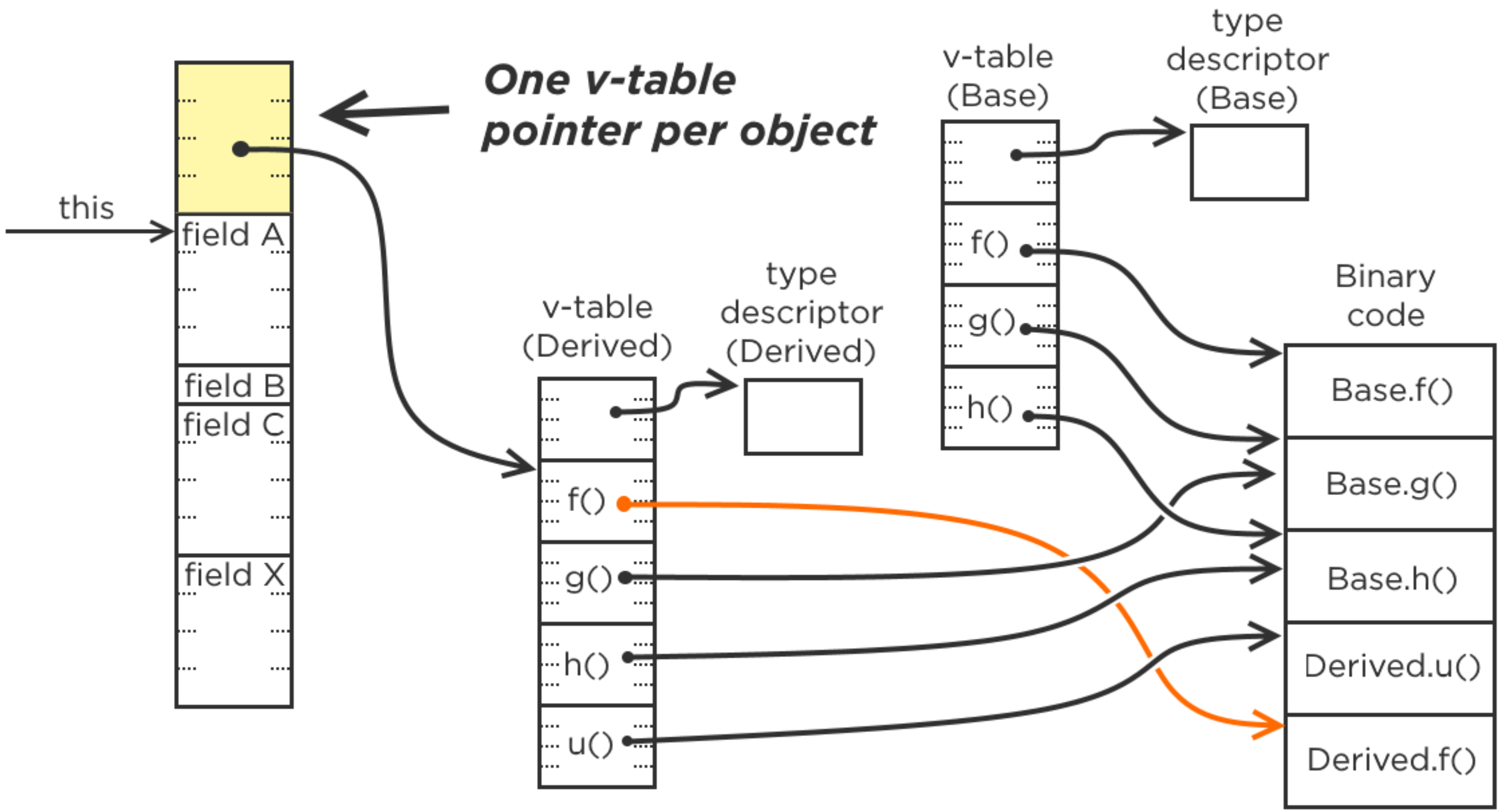
this
field A
field B
field C
field X
v-table (Derived)
type descriptor (Derived)
f()
g()
h()
u()
v-table (Base)
type descriptor (Base)
f()
g()
h()
Binary code
Base.f()
Base.g()
Base.h()
Derived.u()
Derived.f()


x.f()
Dynamic dispatch
this
field A
field B
field C
field X
v-table (Derived)
f()
g()
h()
u()
type descriptor (Derived)
v-table (Base)
f()
g()
h()
type descriptor (Base)
Binary code
Base.f()
Base.g()
Base.h()
Derived.u()
Derived.f()

One v-table and one type descriptor per class (not per object!)



One v-table pointer per object



What Makes the Code Object-oriented?

Object-oriented *C* code?	✓	
Object-oriented *assembly* code?	✓	
Early *C++* and *Objective-C*	✓	structured + *impression* of objects
Mature *C++* and *Objective-C*	✓	fully object-oriented
Java* and *C#	✓	fully object-oriented + *functional*

Then, why so much structured/procedural code today?

What Makes the Code Object-oriented?

Object-oriented C code? ✓	
Object-oriented *assembly* code? ✓	
Early *C++* and *Objective-C* ✓	structured + *impression* of objects
Mature *C++* and *Objective-C* ✓	fully object-oriented
Java* and *C# ✓	fully object-oriented + *functional*

Then, why so much structured/procedural code today?

Why do books still teach programming that way?

What Makes the Code Object-oriented?

Object-oriented C code?	✓	
Object-oriented *assembly* code?	✓	
Early *C++* and *Objective-C*	✓	structured + *impression* of objects
Mature *C++* and *Objective-C*	✓	fully object-oriented
Java* and *C#	✓	fully object-oriented + *functional*

Then, why so much structured/procedural code today?

Why do books still teach programming that way?

It takes time to understand object-oriented programming

What Makes the Code Object-oriented?

Dynamic dispatch

Implicit this reference

Inheritance

Polymorphism

Encapsulation

What Follows in This Course

Introduction

A collection is an object

A missing object is also an object

Branching on Booleans

Replace branching with polymorphic calls

Immutable objects

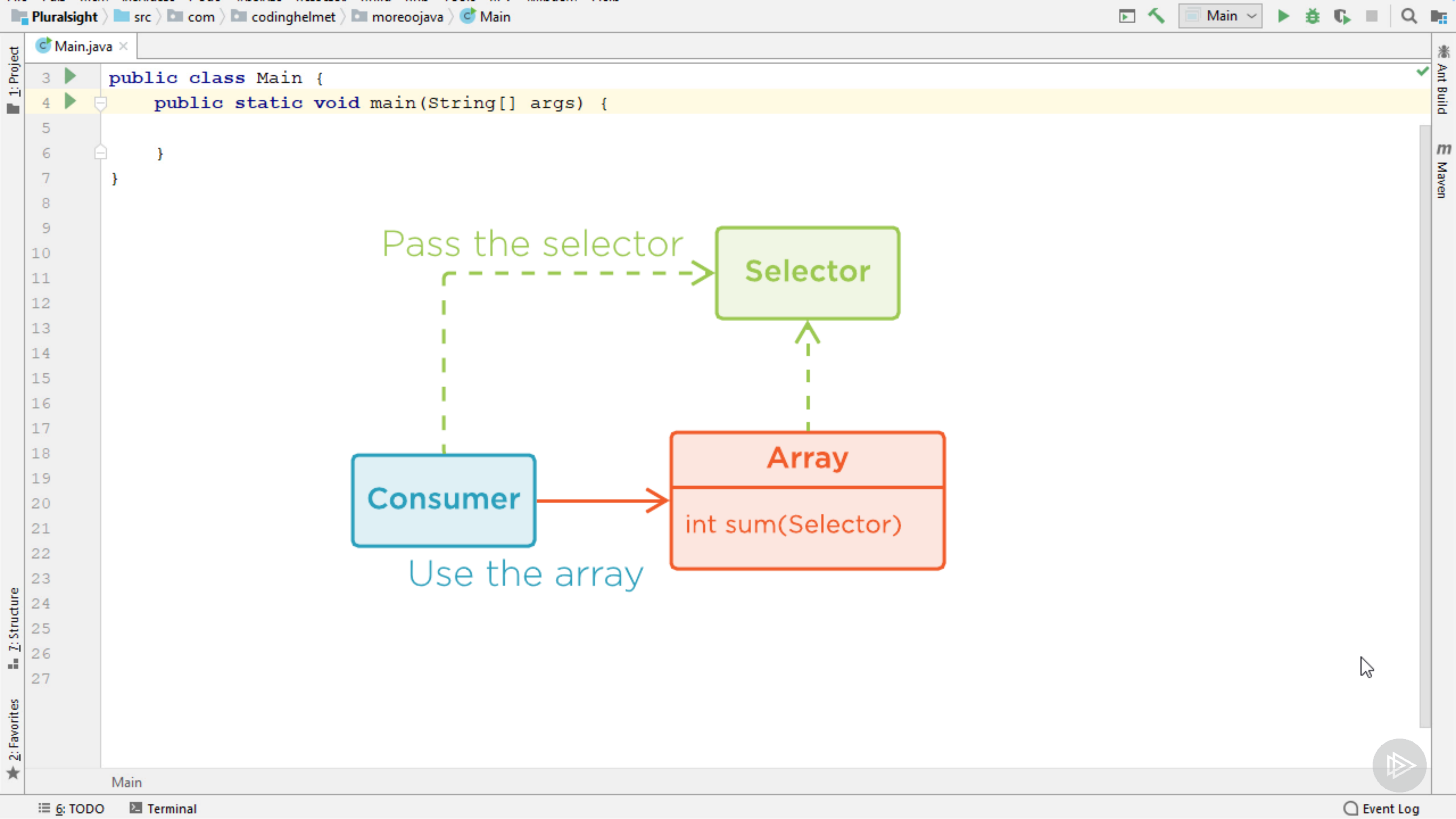
How to avoid bugs due to mutability

Avoiding nulls

Null is not an object

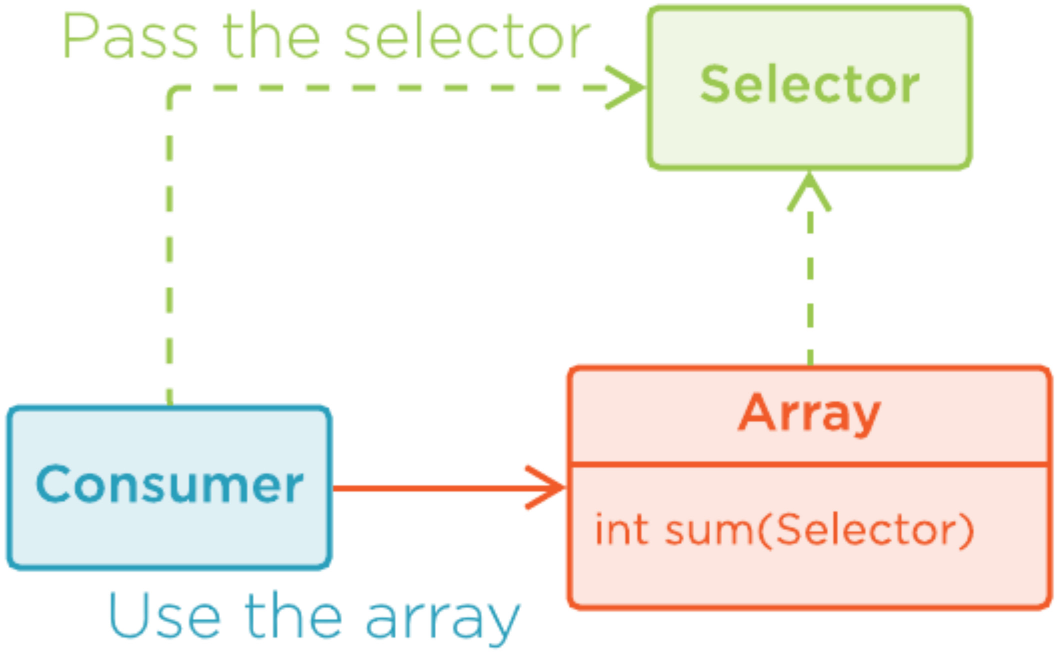
Optional<T> type

No more nulls in business applications



```java
public class Main {
    public static void main(String[] args) {

    }
}
```



Summary

Motivation to write object-oriented code

- Business applications are hard to make right
- Makes software design easier

Summary

In this course you will learn to:

- Detect where objects are missing
- Avoid branching around Booleans
- Remove null references
- Apply principles of object-oriented programming

Summary

Next module:
Rendering Branching over Boolean Flags Obsolete