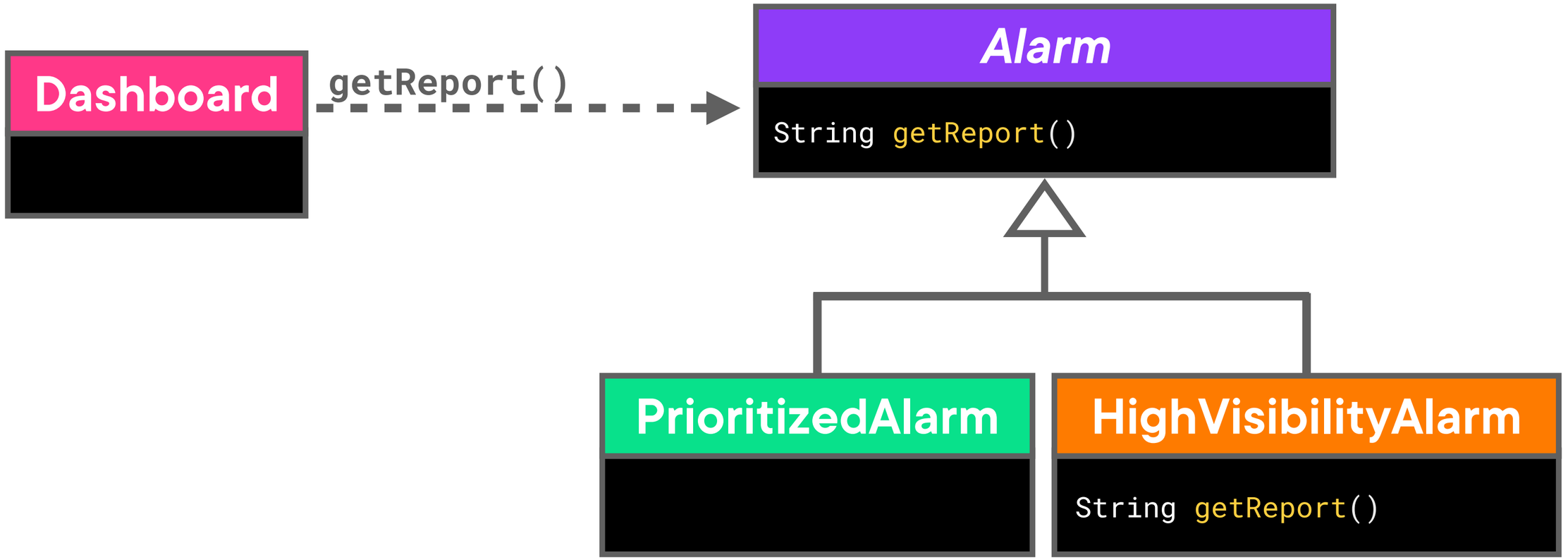
Designing with Inheritance and Polymorphism

Paolo Perrotta

Developer, Author

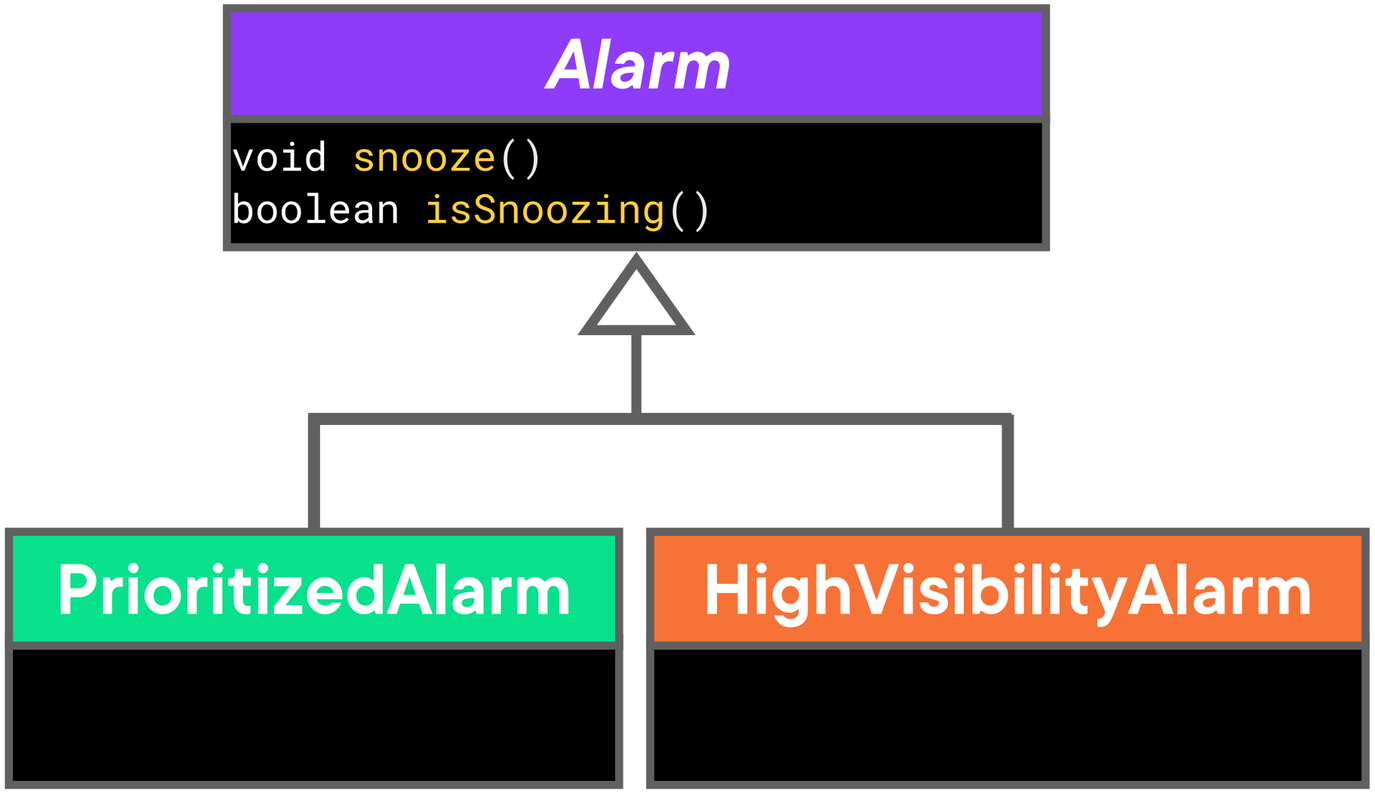
@nusco | www.paoloperrotta.com

What Inheritance Is About

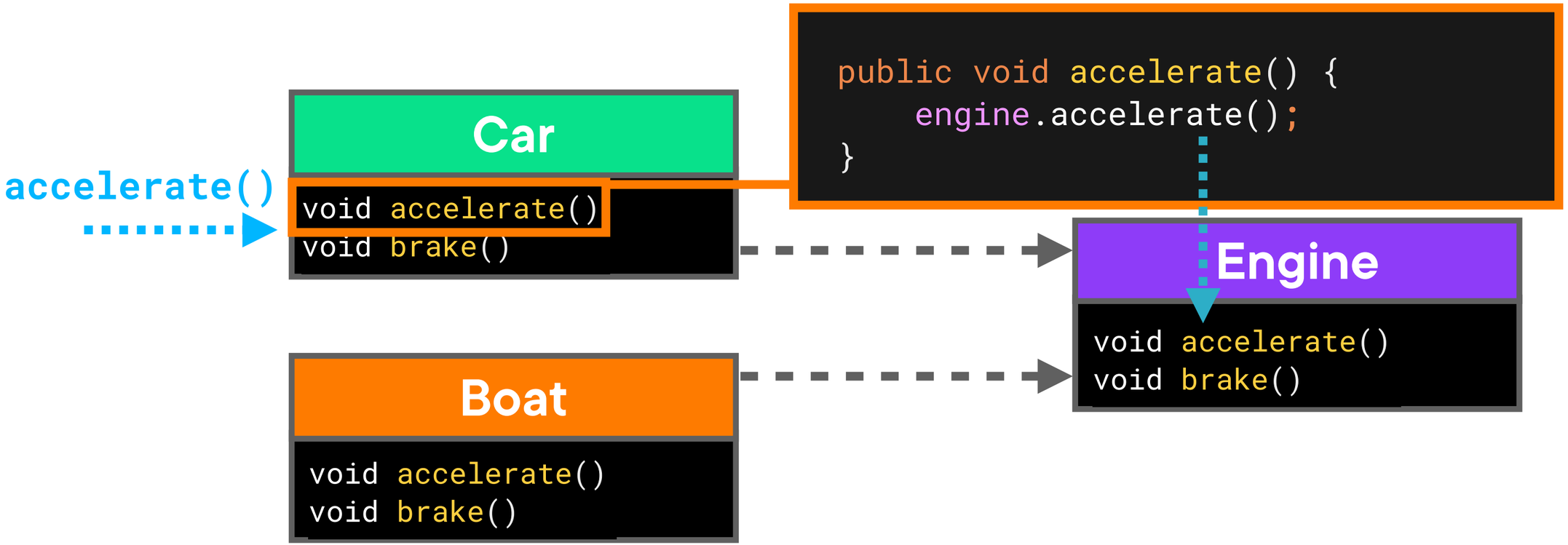


Inheritance is (mostly) about upcasting.

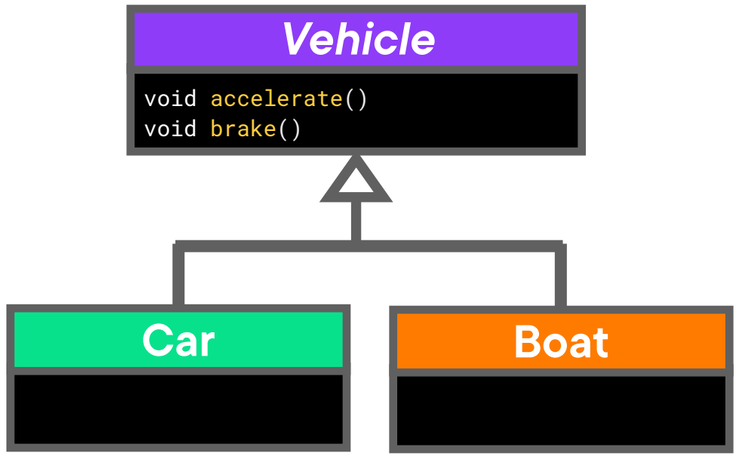
Using Inheritance to Share Code

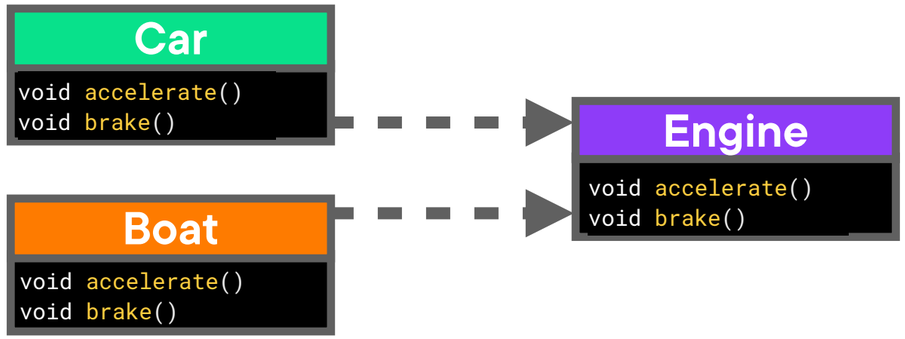


Another Way to Share Code: Delegation



Inheritance vs. Delegation

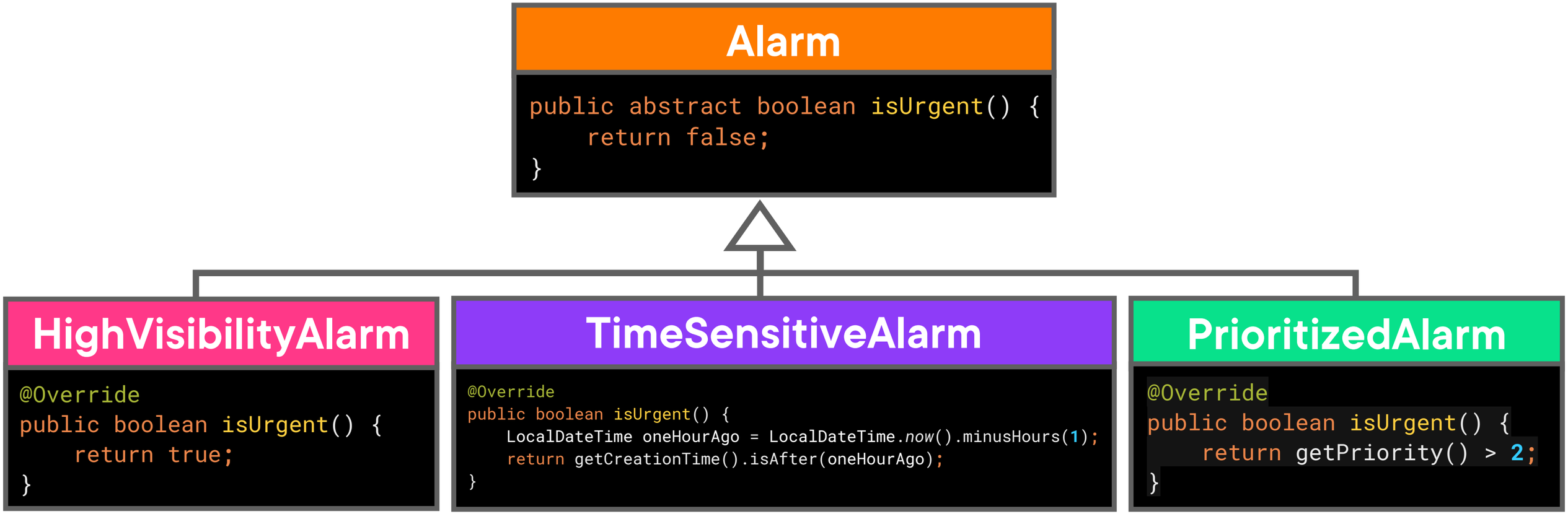




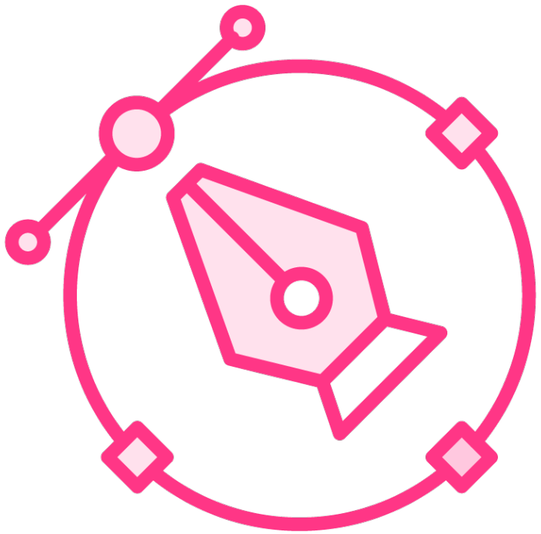
Switching on Type

```
public static boolean isAlarmUrgent(Alarm alarm) {
    if (alarm instanceof PrioritizedAlarm) {
        PrioritizedAlarm prioritizedAlarm = (PrioritizedAlarm)alarm;
        return prioritizedAlarm.getPriority() > 2;
    } else if (alarm instanceof HighVisibilityAlarm) {
        return true;
    } else if (alarm instanceof TimeSensitiveAlarm) {
        TimeSensitiveAlarm timeSensitiveAlarm = (TimeSensitiveAlarm)alarm;
        LocalDateTime oneHourAgo = LocalDateTime.now().minusHours(1);
        return timeSensitiveAlarm.getCreationTime().isAfter(oneHourAgo);
    } else
        return false;
}
```

Replacing Switches on Type with Polymorphism



Two Design Guidelines



Use inheritance when you want to upcast

- That's what inheritance is about
- If you want to share code, consider delegation

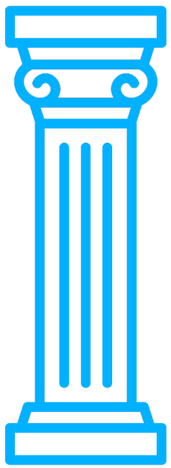
Don't switch on type

- Avoid chains of *instanceof* and downcasts
- Use polymorphism instead

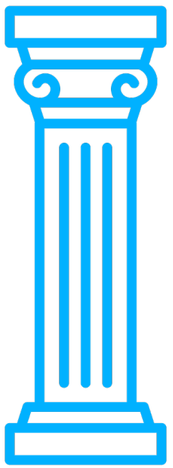
The Pillars of OOP

Abstraction

Encapsulation



Inheritance



Polymorphism

Up Next:

Using the *static* Keyword