

# LESSON 9

SPRING BOOT: LOGGING AND AOP BASICS

MENTOR: ANDREY POMELOV

# SPRING BOOT: LOGGING AND AOP BASICS

---

- What is AOP
- Application of AOP
- Basic concepts of AOP
- AspectJ
- Logging

## WHAT IS AOP

---

**АОП** — аспектно-ориентированное программирование — это парадигма, направленная на повышение модульности различных частей приложения за счет разделения сквозных задач. Для этого к уже существующему коду добавляется дополнительное поведение, без изменений в изначальном коде.

Обычный объектно-ориентированный подход не всегда может эффективно решить те или иные задачи. В такой момент на помощь приходит АОП, дающий нам дополнительные инструменты для постройки приложения. А дополнительные инструменты — это увеличение гибкости при разработке, благодаря которой появляется больше вариантов решения той или иной задачи.

## APPLICATION OF AOP

---

Аспектно-ориентированное программирование предназначено для решения сквозных задач, которые могут представлять собой любой код, многократно повторяющийся разными методами, который нельзя полностью структурировать в отдельный модуль. С его помощью мы можем оставить это за пределами основного кода и определить его по вертикали.

**Пример: безопасность, транзакции, логирование и т.д**

## BASIC CONCEPTS OF AOP

---

**Advice** — это дополнительная логика, код, который вызывается из точки соединения.

Виды советов:

1. Перед (Before)
2. После (After)
3. После возврата (After Returning)
4. После бросания (After Throwing)
5. Вокруг (Around)



## BASIC CONCEPTS OF AOP

---

**Точка соединения (join point)** — точка в выполняемой программе (вызов метода, создание объекта, обращение к переменной), где следует применить совет. Иначе говоря, это некоторое регулярное выражение, с помощью которого и находятся места для внедрения кода (места для применения советов).

**Срез (pointcut)** — набор **точек соединения**. Срез определяет, подходит ли данная точка соединения к данному совету.

**Аспект (aspect)** — модуль или класс, реализующий сквозную функциональность. Аспект изменяет поведение остального кода, применяя совет в точках соединения, определенных некоторым срезом. Иными словами, это комбинация советов и точек соединения.

**Внедрение (introduction)** — изменение структуры класса и/или изменение иерархии наследования для добавления функциональности аспекта в инородный код.

**Цель (target)** — объект, к которому будут применяться советы.

## ASPECTJ

---

**AspectJ** — зрелый AOP фреймворк с огромным комьюнити, множеством публикаций про него, хорошей документацией, достаточно стабильный, интегрированный в разнообразные системы сборки, интеграция в Spring, с хорошей поддержкой в IDE.

# LOGGING

---

Логирование - это процесс записи каких-либо событий, которые происходят в коде.

**Каждая запись лога содержит дату-время, уровень события, сообщение.**

Есть несколько уровней записи. Основные — это info, warn, error.

- **INFO** — обычно это информационные сообщения. Например, записи в базу данных и так далее.
- **WARN** — предупреждения.
- **ERROR** — ошибки. Например, когда мы оборачиваем что-то в try-catch, в блоке catch подставляется e.printStackTrace(). Он выводит запись только в консоль. С помощью логера можно мониторить эту запись.