***Pointcut expression***

Để tìm kiếm xem điểm nào sẽ thực hiện advice

- cú pháp expression để match với method

execution ( kiểu-modifier? ***kiểu-trả-về*** declaring-type-pattern?

***tên hàm(param-pattern)*** throws-pattern? )

- ? là ko bắt buộc

vd @Before("execution(void addCustomer())")

- dùng \* để match vs mọi cái mình muốn

vd @Before("execution(\* \* add\*())")

- ***param-pattern:***

***() method ko có param***

***(\*) matches method có 1 param ko qtram kiểu dl***

***(..) matches method có 0 or n param ko qtram kiểu dl***

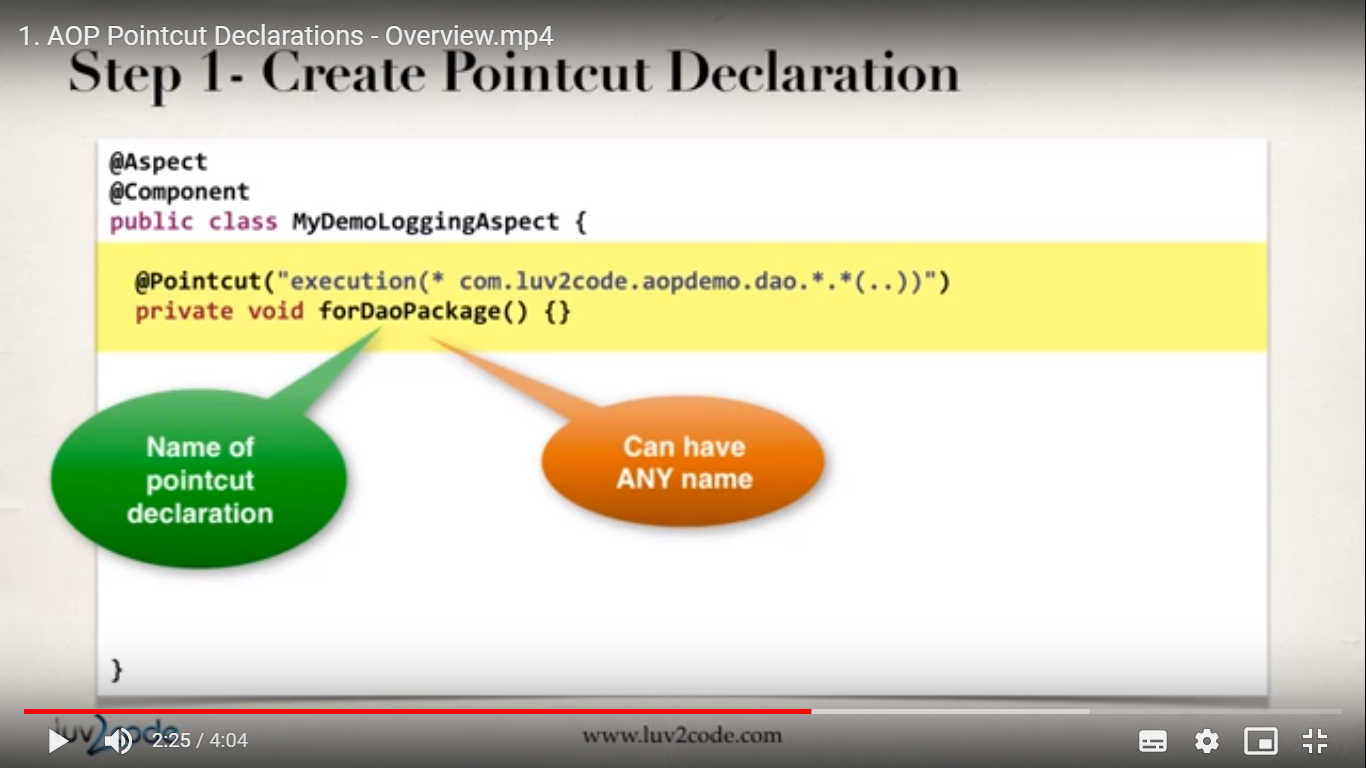
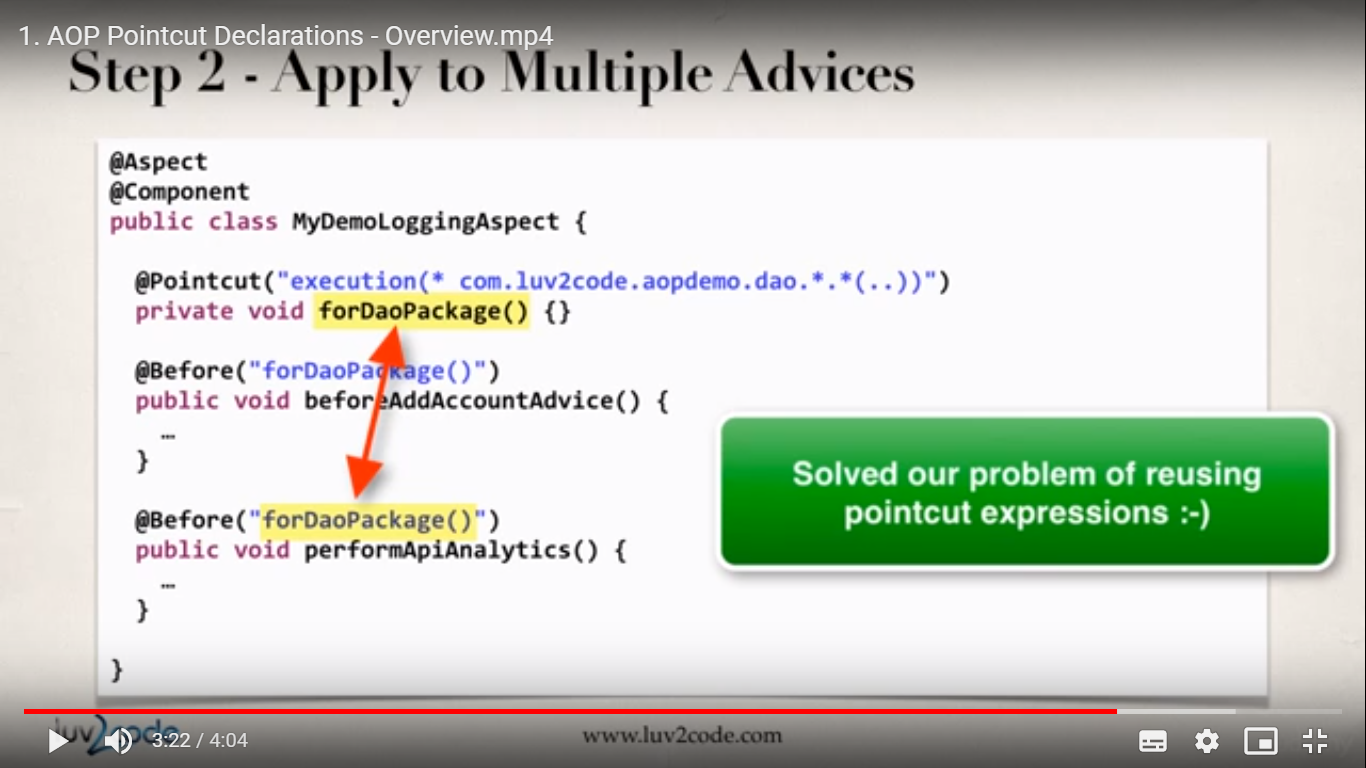
*vd @Before("execution(\* \* add\*(com.my\_project.model.Khoa))")*

*@Before("execution(\* \* com.my\_project.dao.\*.\*(..))")*  modifier return-type package.class.method(param type)

\*\*\* Nếu param là kiểu dl Obj thì phải full tên vd: com.my\_project.model.Khoa chứ kp Khoa không :>

***PointCut Declarations***

Làm thế nào để tạo hoặc dùng chung 1 expression: ta sẽ sd point cut declaration :D

1. Tạo pointcut declaration
2. Apply vs nhiều advice 

Lợi ích :

* Reuse pointcut expression
* Update pointcut expression duy nhất 1 nơi thôi =))
* Có thể share và combine-kết hợp pointcut expression (coming up later)

***Combine expression***

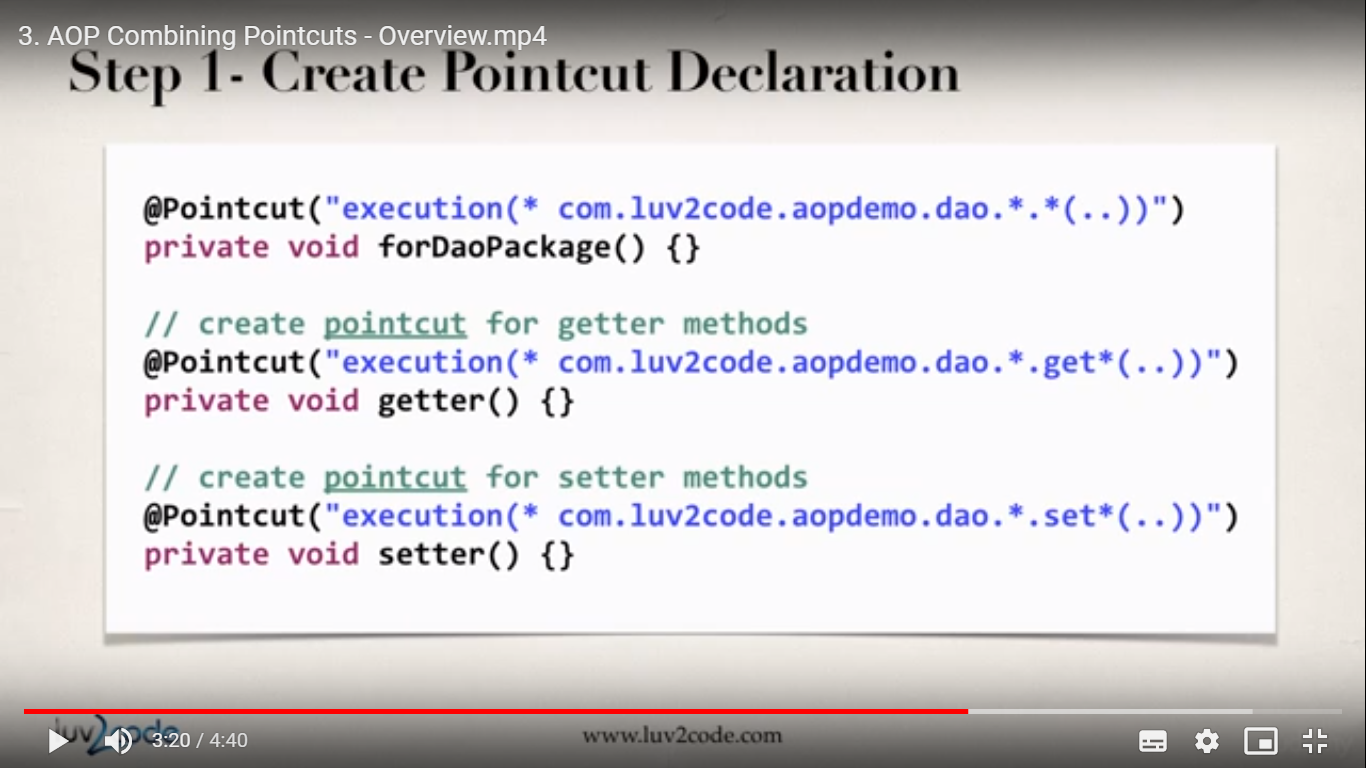
Sử dụng bằng cách sdung logic như and, or, not, …

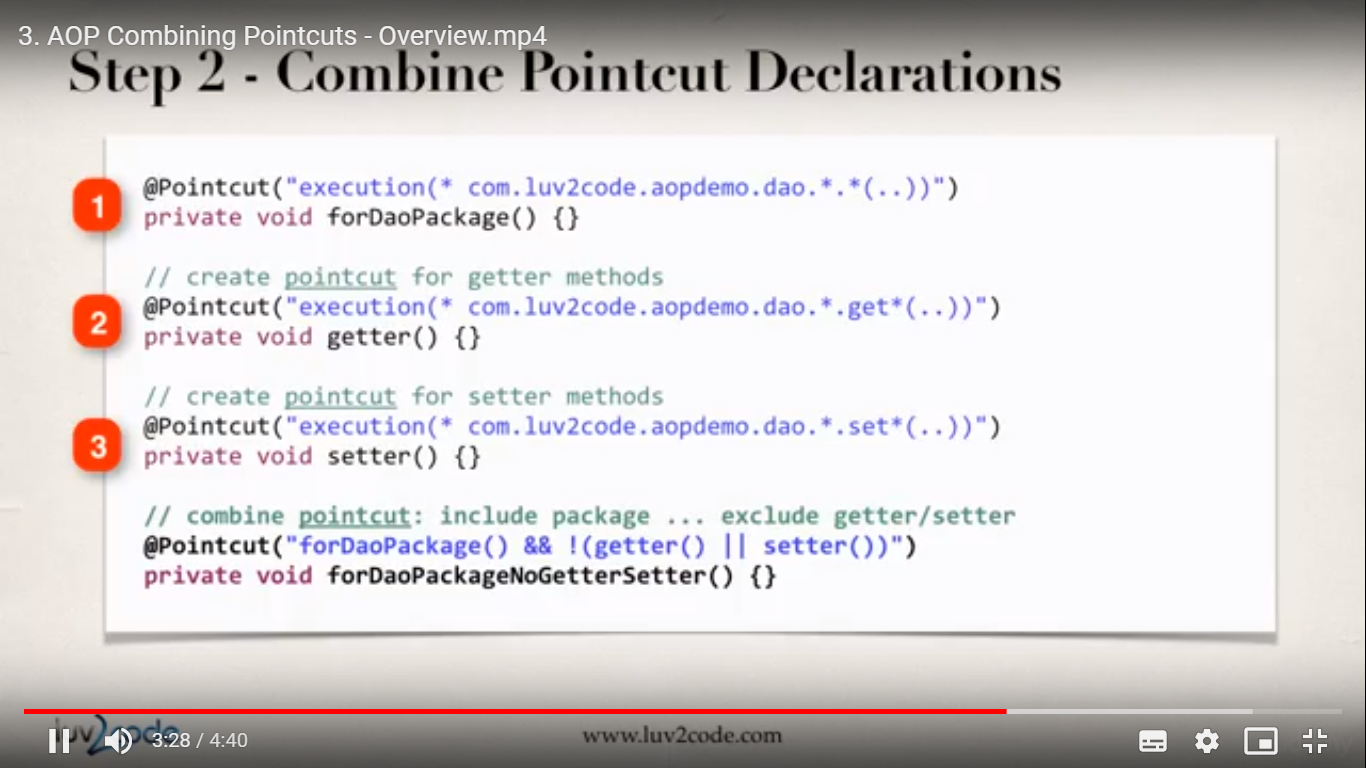
*Vd*: muốn sử dụng pointcut expression cho các hàm trong lớp ***TRỪ*** hàm getter – setter

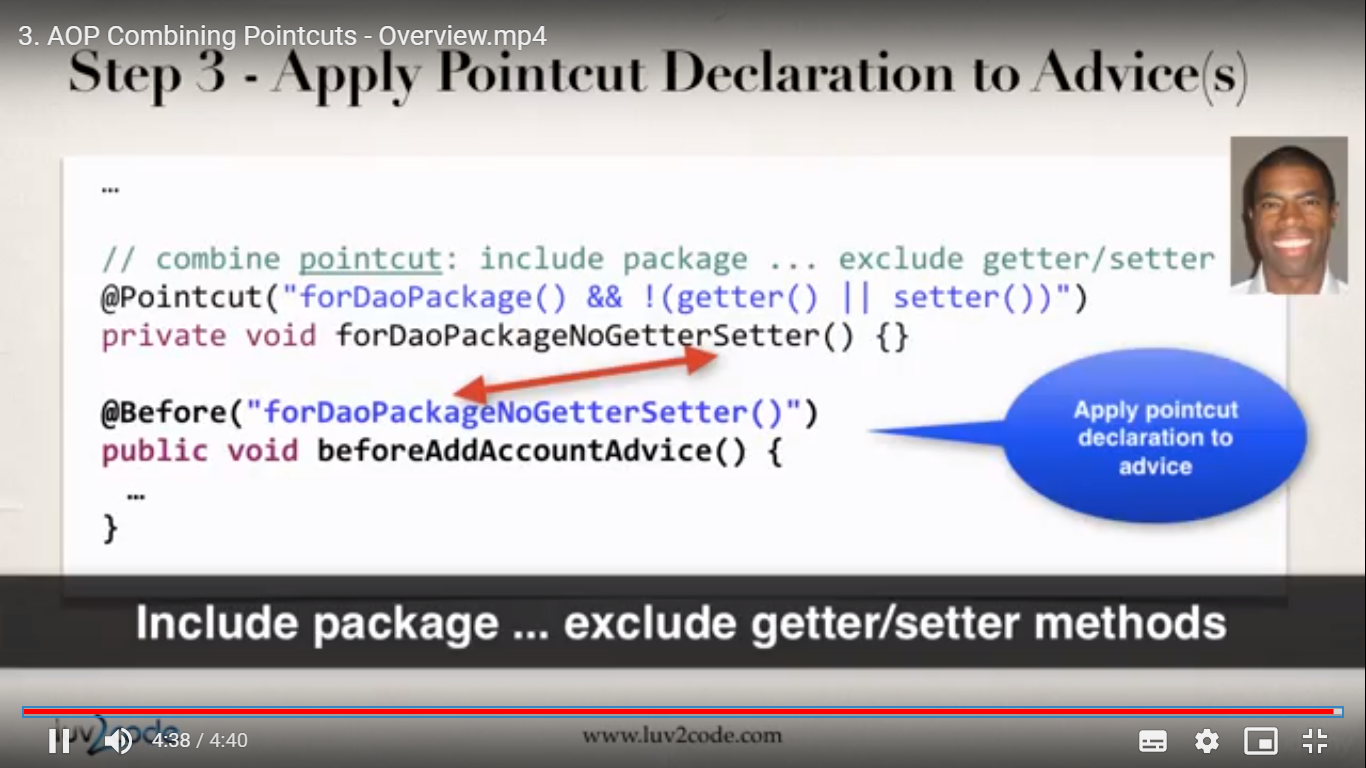
Xử lí chỉ khi nó true( làm như if )

Vd: @Before(“epressionOne() || expressionTwo()”)

@Before(“epressionOne() && !expressionTwo()”)

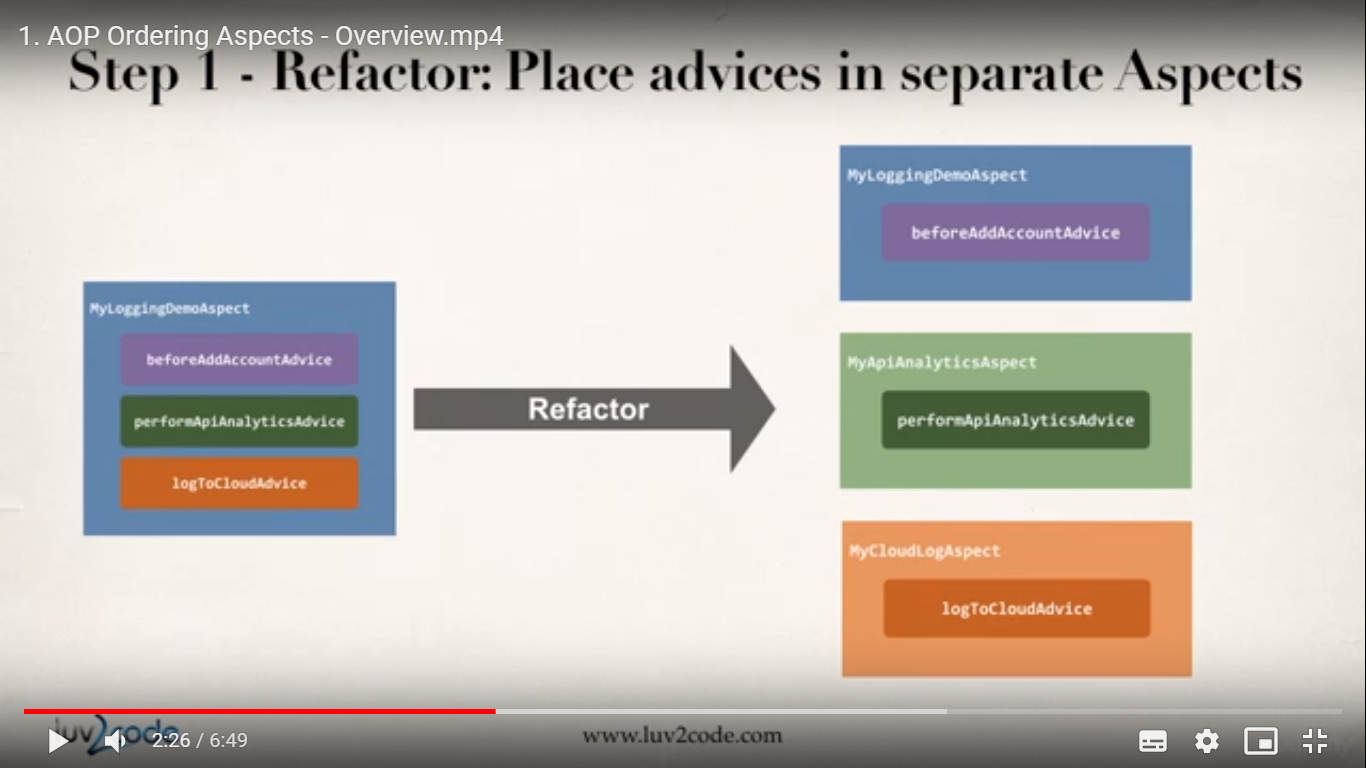
1. B1
2. B2



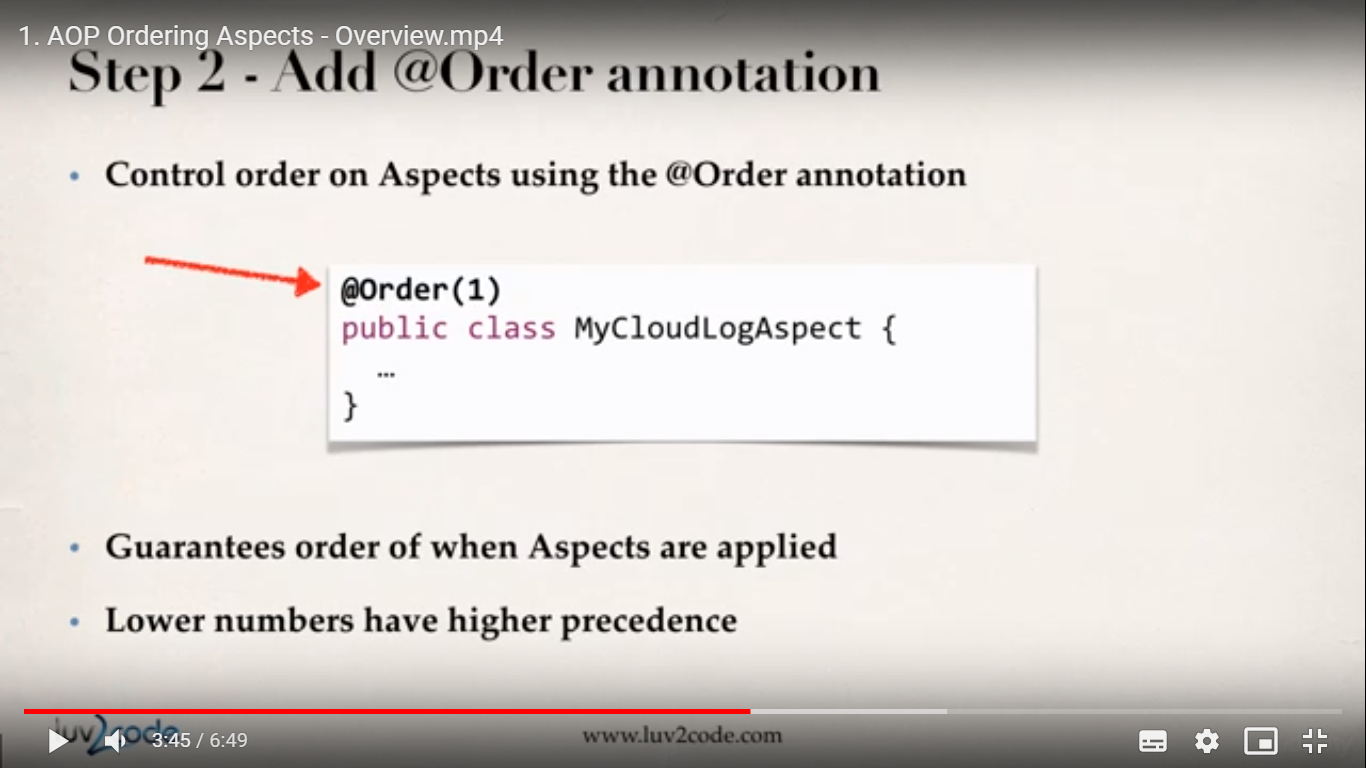
1. B3

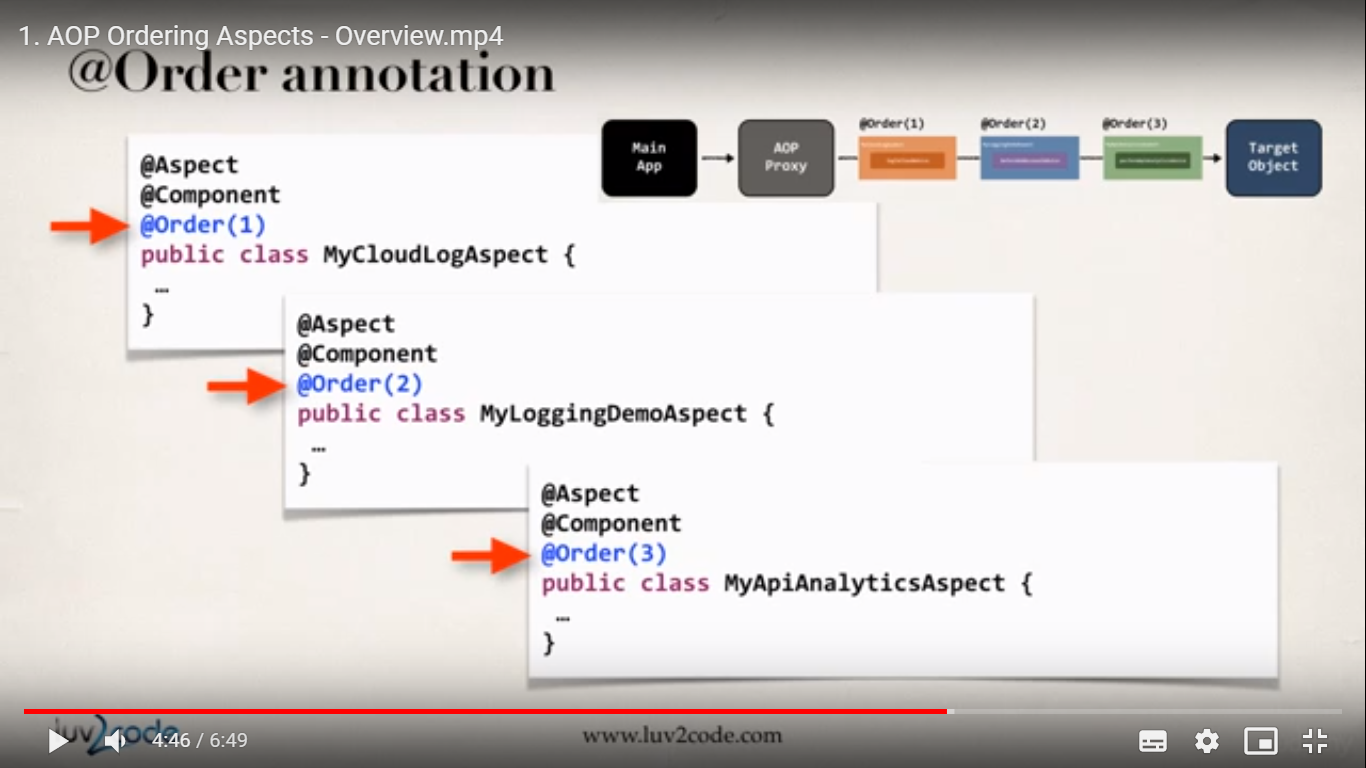
***Sắp xếp các Class ASPECT ADVICE***

1. Tách các ASPECT riêng ra 1 class



2. Thêm annotation @ORDER vào sau @ASPECT





\*\*\*\* Số không cần phải ltuc. Chấp nhận số âm vd Order(-43) -> Order(6) -> Order(17). Trùng số thì không chạy ASPECT trùng

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

***READ ARGUMENT WITH JOINPOINT***

Đọc value từ các hàm được sd (chắc là sd ứng dụng cho @AfterReturning và @AfterThrowing)

<https://drive.google.com/drive/folders/1M-WzpRAQdSnzlx1_-SCEjr_5PG0yTkd9>