

DISCOVERING THE AOP

WHAT IS ASPECT ?

Aspect oriented ဆိုတော့ aspect ဆိုတာ ဘာလဲပေါ့။

Aspect ဆိုတာ method တစ်ခုကို call လိုက်တိုင်းမှာ

အလိုအလျောက် run သွားမယ့် code တွေကိုပြောတာ။

ငါတို့တွေ method တစ်ခု function တစ်ခုကို run လိုက်တယ်

ဆိုရင် business logic တွေ အလုပ်လုပ်သွားတယ်နော်။

ဒါပေမယ့် ငါတို့က business logic နဲ့ မဆိုင်ဘဲနဲ့ run သွားစေခြင်

တဲ့ non-business related code တွေလည်း ရှိသေးတယ်။

logging ရှိမယ် Auditing ရှိမယ်။ Security management

ရှိမယ်။ အဲလိုနေရာတွေမှာ AOP ကိုသုံးတယ်



WHY AOP ?

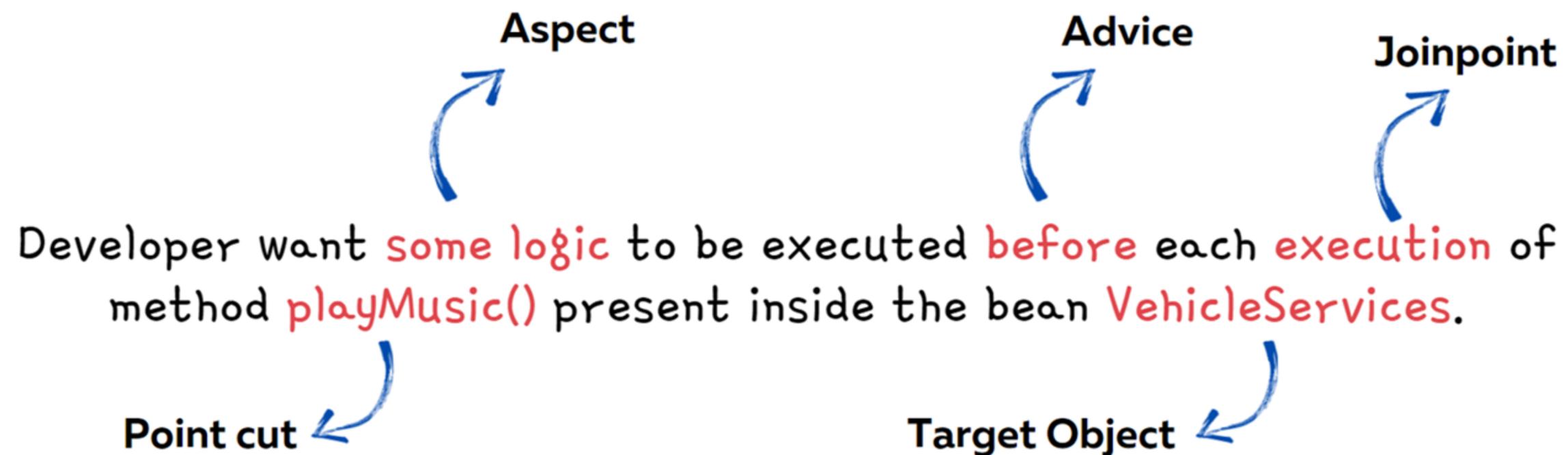
ဘာလို့ AOP လည်းပေါ့။ AOP ကိုသုံးရင် modularity ကိုရစွေတယ်။ ဘာလို့ ရုစေလဲဆိုတော့ separation of cross-cutting concerns ကိုလုပ်လို့ရသွားလို့။ ရှင်းပြမယ်။ ငါတို့ရဲ့ business logic ကဒီလို့ data တစ်ကြောင်းကို ဖျက်တဲ့ function တစ်ခုကို call မယ်ဆိုရင် security အရလည်း authority ရှိလား စစ်မယ် ရှိရင် ဖျက်မယ် ပြီးရင် log လည်း မှတ်မယ်။ အဲမှာတင် မတူညီတဲ့ concerns သုံးခုဟာ ပူးထွေးယှက်တင် (cross cutting) ပြီး အလုပ်လုပ်နေပြီ။ AOP သာ မရှိရင် function တစ်ခုထဲမှာပဲ အဲဒါတွေကို ပေါင်းရေးရတော့မယ်။ AOP ကြောင့်မို့ အဲဒါတွေက ခဲ့ပြီး ရေးလို့ရတယ်။ ပြန်သုံးလို့လည်းရတယ်။ အစီအစဉ်အတိုင်းလည်း run လို့ရပြီ။



AOP IN SPRING

Spring မှပါတဲ့ AOP ကို Implementation လုပ်တော့မယ်ဆိုရင် ဒီ သုံးခုကို သိဖို့လိုတယ်။

- What -> Aspect
- When -> Advice
- Which -> Pointcut



Join point ဆိုတာက event triggers မယ့်အချိန်ကို ပြောတာ။

ADVICE

THERE IS FIVE TYPES OF ADVICE
IN SPRING

@Before

(target method ကို execution မလုပ်ခင် aspect ကို run)

@After

(target method ကို execution လုပ်ပြီးမှ aspect ကို run)

@AfterReturning

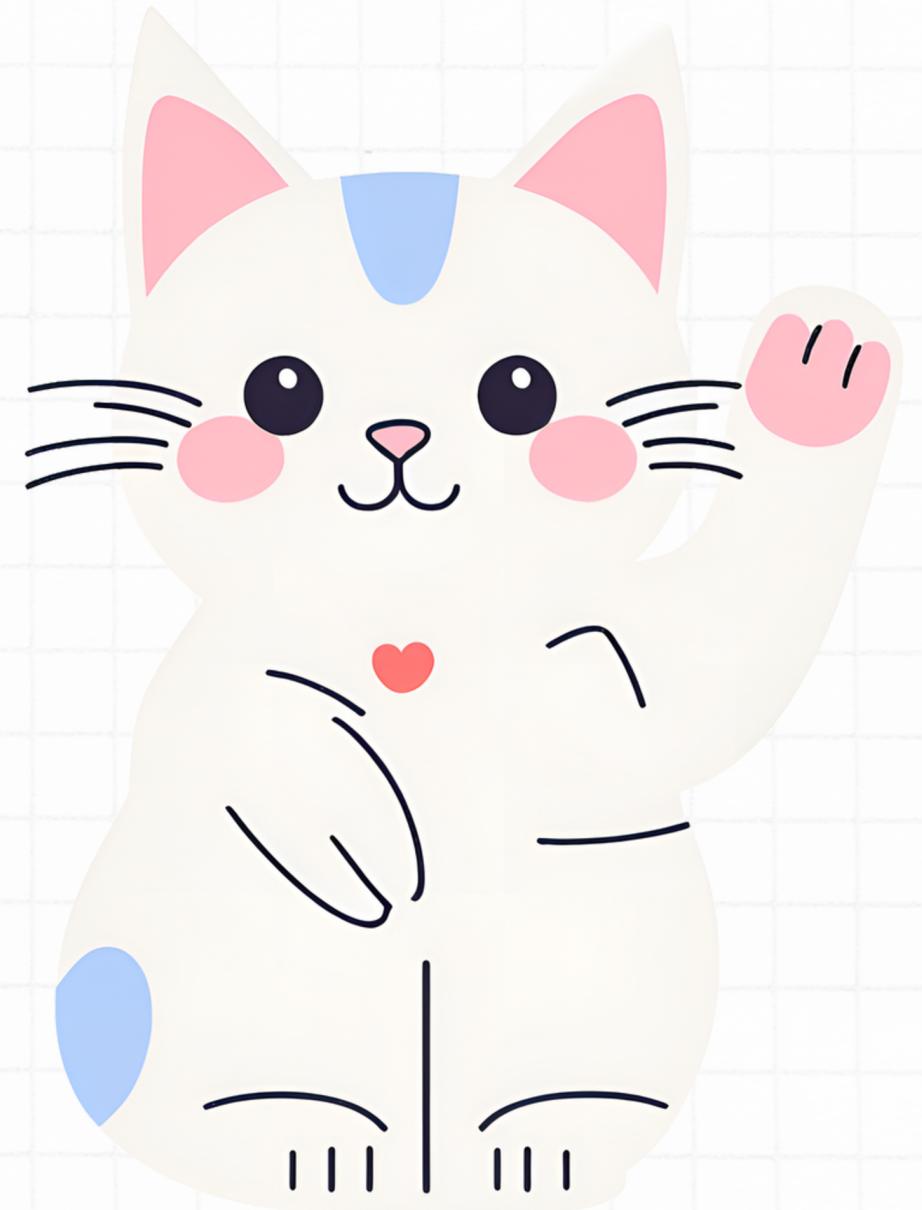
(target method ကို response ပြန်မှ aspect ကို run)

@AfterThrowing

(target method ကို execution မလုပ်ခင် aspect ကို run)

@Around

(ဘယ်အချိန်မှ aspect ကို run မလဲဆိုတာကို define လိုရတယ်။)



အာရိဂတ်း ကိုယ်မော်

