# Mise en place de Spring AOP

# Démonstration 1 du module 9

|  |
| --- |
| Les objectifs de cette démonstration sont   * Utilisation et configuration des starters de Spring AOP * Création d’une classe annotée @Aspect et @Component * Créer 5 types d’advices |

Contexte

* Nous continuons notre application précédente : demo-nosql.
  + Si votre application n’est pas complète, vous pouvez reprendre le code Java dans les ressources
  + Attention, le paramétrage du projet Spring Boot n’est pas inclus, il vous faut le compléter
  + Pensez aux 2 propriétés de GitHub pour OAuth 2.0 dans application.yml
* Dans cette itération, nous voulons créer nos propres aspects

Déroulement

# Dépendances Spring Boot :

* Dans build.gradle ; ajouter le starter de Spring AOP

// Starter AOP

**implementation** 'org.springframework.boot:spring-boot-starter-aop:3.1.3'

* Faire un Gradle Refresh Project
* Votre projet permet de créer des aspects avec Spring AOP

# Configuration des logs

* Nous allons manipuler plus de logs qu’au paravent.
  + Pour éviter de saturer les ressources, nous allons changer le niveau de trace du projet
* Dans le fichier de configuration de Spring Boot : application.yml :
  + Nous avions un niveau de log pour MongoTemplate
  + Nous ajoutons un niveau pour root : WARN

#Log

logging:

level:

root: WARN

org:

springframework:

data:

mongodb:

core:

MongoTemplate: DEBUG

# Classe des aspects :

Nous voulons tracer plus précisément les informations liées à nos méthodes de requêtes spécifiques.

* Créer une classe pour contenir les aspects
  + Poser l’annotation @Aspect, @Component et @Slf4j
  + Ainsi, cette classe peut définir des aspects. C’est un bean pour Spring. Et elle peut manipuler les logs

**package** fr.eni.demo.aop;

**import** java.util.List;

**import** org.aspectj.lang.JoinPoint;

**import** org.aspectj.lang.ProceedingJoinPoint;

**import** org.aspectj.lang.annotation.After;

**import** org.aspectj.lang.annotation.AfterReturning;

**import** org.aspectj.lang.annotation.AfterThrowing;

**import** org.aspectj.lang.annotation.Around;

**import** org.aspectj.lang.annotation.Aspect;

**import** org.aspectj.lang.annotation.Before;

**import** org.springframework.stereotype.Component;

**import** fr.eni.demo.bo.Avis;

**import** lombok.extern.slf4j.Slf4j;

@Aspect

@Component

@Slf4j

**public** **class** LogAspect {

…

}

# Pointcut

Nous voulons tracer plus précisément les informations liées à nos méthodes de requêtes spécifiques.

* Nous voulons tracer toutes les méthodes de requêtes de l’interface AvisRepository ; pour la recherche des Avis selon leur note
* Dans cette interface, il y a 3 méthodes à gérer :

**public** **interface** AvisRepository **extends** MongoRepository<Avis, String> {

List<Avis> findByNoteCours(@Param("noteCours") **int** noteCours);

List<Avis> findByNoteCoursGreaterThan(@Param("noteCours") **int** noteCours);

List<Avis> findByNoteCoursLessThan(@Param("noteCours") **int** noteCours);

* Un pointcut permettant de répondre à cette demande est :

execution(public \* fr.eni.demo.dal.AvisRepository.findByNote\*(int))

* + Nous ne gérons pas le type de retour (\*), il n’est pas discriminant
  + Nous précisons le package, le nom de l’interface
  + Nous créons un début de nom de méthode : findByNote\* pour englober les trois
  + Les trois méthodes prennent un int en paramètre ; pour l’exemple
* Un autre pointcut possible est :

execution(public \* \*..AvisRepository.findByNote\*(..))

* + Nous ne gérons pas le type de retour (\*), il n’est pas discriminant
  + Nous ne précisons pas le package, nous lui demandons de descendre dans les packages jusqu’à trouver le nom de l’interface
  + Nous créons un début de nom de méthode : findByNote\* pour englober les trois
  + Nous ne précisons pas le type des paramètres (..) ; il n’est pas discriminant

Nous allons utiliser ces 2 pointcuts pour montrer qu’ils sont valides

# Advice

Nous voulons montrer les 5 types d’advices possibles :

* Advice before
* Advice around
* Advice after (FINALLY)
* Advice returning (TRY avec valeur de retour)
* Advice throwing (CATCH – levée d’une exception)

## Advice before

* Ajouter une méthode pour capturer le nom de la méthode exécutée
* Utiliser @Before
* Et notre premier pointcut :

execution(public \* fr.eni.demo.dal.AvisRepository.findByNote\*(int))

@Before("execution(public \* fr.eni.demo.dal.AvisRepository.findByNote\*(int))")

**public** **void** before(JoinPoint jp) {

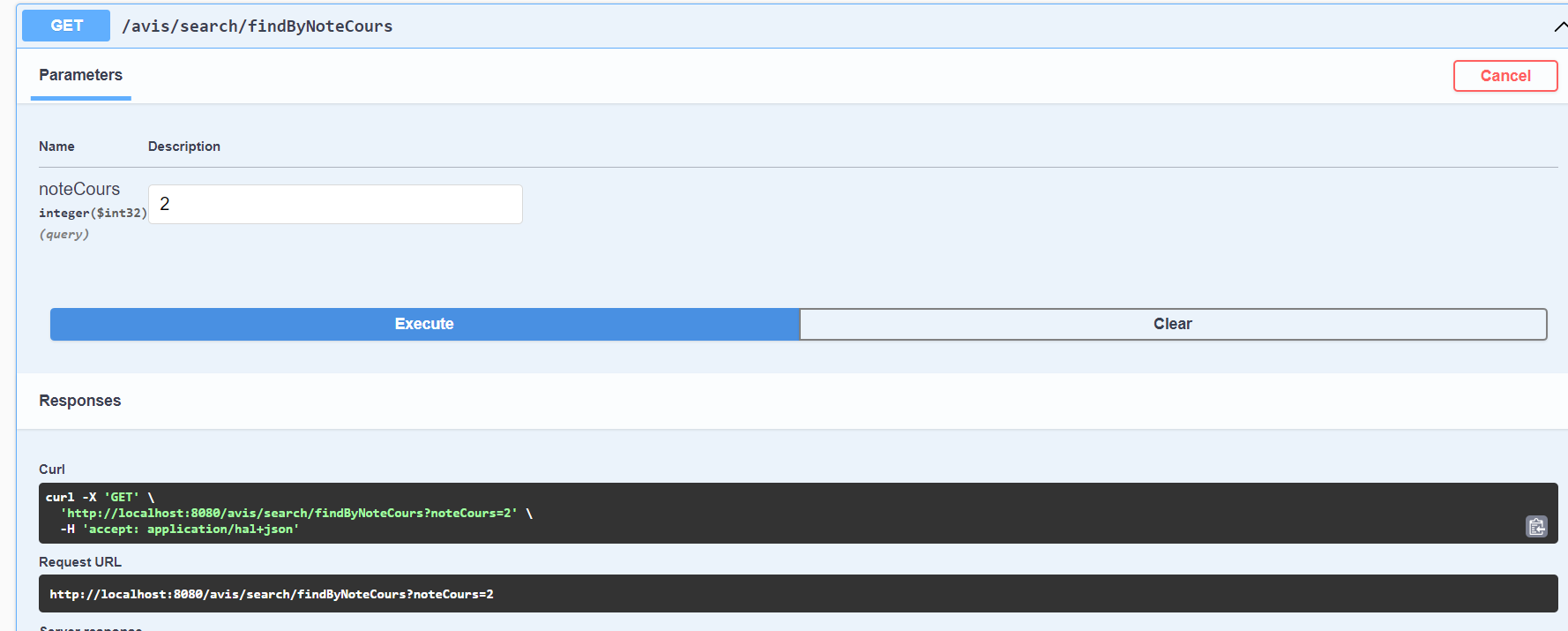
***log***.warn("ENI AOP - @Before - "+ jp.getSignature().getName());

}

* Elle prend en paramètre JoinPoint pour récupérer les informations de la méthode exécutée

### Exécution :

* Lancer l’application
* Pensez à vous connecter à GitHub pour avoir le droit d’utiliser Swagger-UI
* Tester findByNoteCours



* Remarquez dans les logs de l’IDE
  + La trace du @Before avant la trace de MongoTemplate

fr.eni.demo.aop.LogAspect ENI AOP - @Before - findByNoteCours

o.s.data.mongodb.core.MongoTemplate find using query: { "course\_note" : 2} fields: Document{{}} for class: class fr.eni.demo.bo.Avis in collection: reviews

## Advice arounds

* Ajouter une méthode pour tracer le temps d’exécution de la méthode appelée
* Utiliser @Around
* Et notre second pointcut :

execution(public \* \*..AvisRepository.findByNote\*(..))

* Elle prend en paramètre ProceedingJoinPoint
  + Pour appeler la méthode d’origine
* Elle propage les exceptions : Throwable

@Around(value="execution(public \* \*..AvisRepository.findByNote\*(..))")

**public** Object around(ProceedingJoinPoint call) **throws** Throwable {

Object result = **null**;

**long** before = System.*currentTimeMillis*();

result = call.proceed();

**long** after = System.*currentTimeMillis*();

**long** duration = after - before;

String nomMethode = call.getSignature().getDeclaringTypeName() + "/" + call.getSignature().getName();

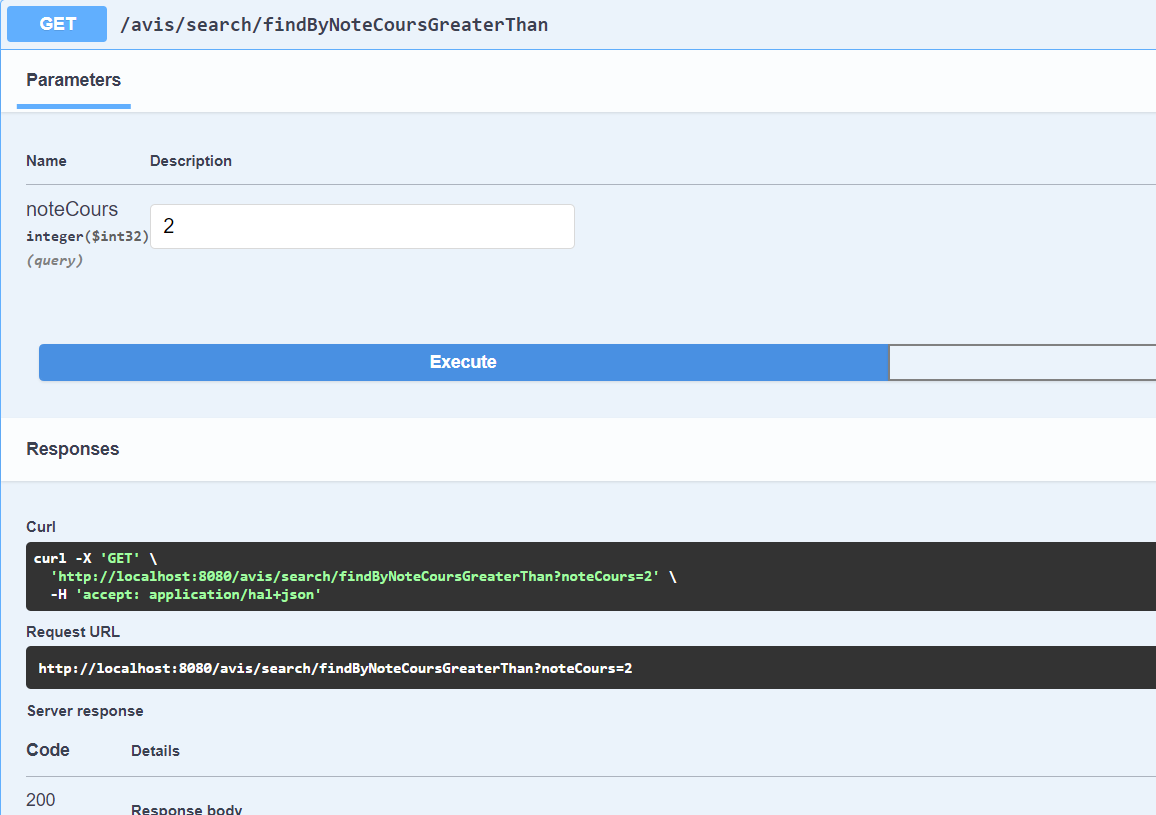
***log***.warn("ENI AOP - @Around - " + nomMethode + ":" + duration + " ms");

**return** result;

}

### Exécution :

* Tester findByNoteCoursGreaterThan



* Remarquez dans les logs de l’IDE
  + Vous avez les traces de @Before et @Around

fr.eni.demo.aop.LogAspect ENI AOP - @Before - findByNoteCoursGreaterThan

o.s.data.mongodb.core.MongoTemplate find using query: { "course\_note" : { "$gt" : 2}} fields: Document{{}} for class: class fr.eni.demo.bo.Avis in collection: reviews

fr.eni.demo.aop.LogAspect ENI AOP - @Around - fr.eni.demo.dal.AvisRepository/findByNoteCoursGreaterThan:58 ms

## Advice after

* Il s’exécute que l’exécution de la méthode se passe bien ; ou qu’elle échoue
* C’est équivalent au FINALLY dans les blocs TRY-CATCH-FINALLY
* Ajouter une méthode, elle sert uniquement à montrer l’ordre d’appel des aspects
* Utiliser @After

@After("execution(public \* \*..AvisRepository.findByNote\*(..))")

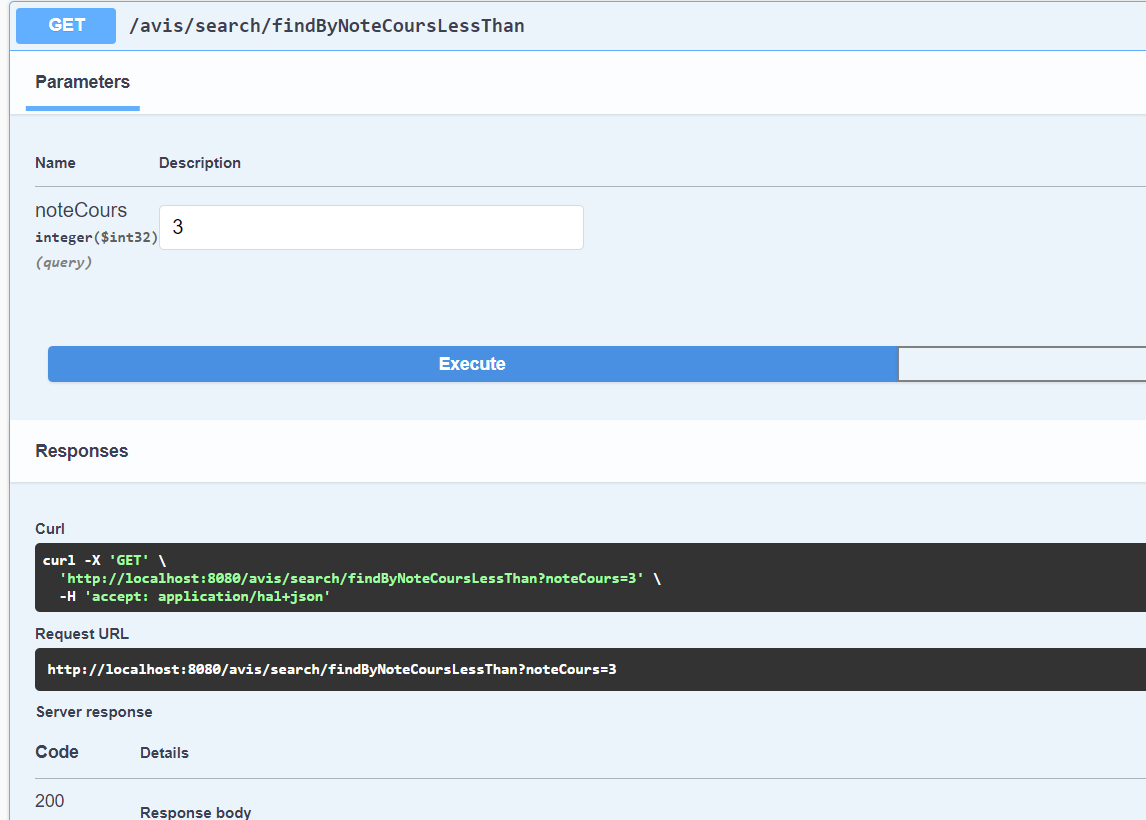
**public** **void** after() {

***log***.warn("ENI AOP - @After - FINALLY");

}

### Exécution :

* Tester findByNoteCoursLessThan



* Remarquez dans les logs de l’IDE
  + Vous avez les traces de @Before, @After et @Around

fr.eni.demo.aop.LogAspect ENI AOP - @Before - findByNoteCoursLessThan

o.s.data.mongodb.core.MongoTemplate find using query: { "course\_note" : { "$lt" : 3}} fields: Document{{}} for class: class fr.eni.demo.bo.Avis in collection: reviews

fr.eni.demo.aop.LogAspect ENI AOP - @After - FINALLY

fr.eni.demo.aop.LogAspect ENI AOP - @Around - fr.eni.demo.dal.AvisRepository/findByNoteCoursLessThan:32 ms

## Advice after returning

* Il s’exécute si la méthode arrive à retourner son résultat
* C’est équivalent à un return dans le bloc TRY de TRY-CATCH-FINALLY
* Ajouter une méthode, elle prend en paramètre JointPoint et la liste des Avis retournée
* Utiliser @AfterReturning et paramétré « returning »

@AfterReturning(pointcut = "execution(public \* \*..AvisRepository.findByNote\*(..))", returning = "result")

**public** **void** afterReturning(JoinPoint jp, List<Avis> result) {

***log***.warn("ENI AOP - @AfterReturning - TRY - RETURN [" + jp.getSignature().getName() + "] = " + result);

}

### Exécution :

* Tester findByNoteCours
* Remarquez dans les logs de l’IDE
  + Vous avez les traces de @Before, @AfterReturning, @After et @Around

fr.eni.demo.aop.LogAspect ENI AOP - @Before - findByNoteCours

o.s.data.mongodb.core.MongoTemplate find using query: { "course\_note" : 3} fields: Document{{}} for class: class fr.eni.demo.bo.Avis in collection: reviews

fr.eni.demo.aop.LogAspect ENI AOP - @AfterReturning - TRY - RETURN [findByNoteCours] = [Avis(id=64f86d869071fe1c40dbde2e, notePedagogie=4, commentairePedagogie=commentaire sur la pédagogie, noteCours=3, commentaireCours=null, stagiaire=null, formateur=null, cours=null), Avis(id=64f86d869071fe1c40dbde2f, notePedagogie=4, commentairePedagogie=commentaire sur la pédagogie, noteCours=3, commentaireCours=commentaire sur le cours, stagiaire=null, formateur=null, cours=null), Avis(id=64f86d869071fe1c40dbde31, notePedagogie=3, commentairePedagogie=Commentaire sur la pédagogie (3), noteCours=3, commentaireCours=Commentaire du cours (3), stagiaire=Stagiaire(immatriculation=ENI\_12531, promotion=CDA12341), formateur=Formateur(email=abaille@campus-eni.fr, nom=BAILLE, prenom=Anne-Lise), cours=Cours(reference=M360, filiere=Développement, titre=Java Frameworks - API Web, duree=10)), Avis(id=64f86d869071fe1c40dbde34, notePedagogie=3, commentairePedagogie=Commentaire sur la pédagogie (3), noteCours=3, commentaireCours=Commentaire du cours (3), stagiaire=Stagiaire(immatriculation=ENI\_12531, promotion=CDA12341), formateur=Formateur(email=pmontembault@campus-eni.fr, nom=MONTEMBAULT, prenom=Philippe), cours=Cours(reference=M030, filiere=Développement, titre=Web Client, duree=5)), Avis(id=64f86d869071fe1c40dbde37, notePedagogie=3, commentairePedagogie=Commentaire sur la pédagogie (3), noteCours=3, commentaireCours=Commentaire du cours (3), stagiaire=Stagiaire(immatriculation=ENI\_12531, promotion=CDA12341), formateur=Formateur(email=fdelachesnais@campus-eni.fr, nom=DELACHESNAIS, prenom=Frédéric), cours=Cours(reference=M070, filiere=Développement, titre=POO, duree=10)), Avis(id=64f86d879071fe1c40dbde3a, notePedagogie=4, commentairePedagogie=commentaire sur la pédagogie, noteCours=3, commentaireCours=commentaire sur le cours, stagiaire=Stagiaire(immatriculation=ENI\_CAMPUS\_202311987, promotion=EDMW0001), formateur=null, cours=null), Avis(id=64f86d879071fe1c40dbde3b, notePedagogie=4, commentairePedagogie=commentaire sur la pédagogie, noteCours=3, commentaireCours=commentaire sur le cours, stagiaire=Stagiaire(immatriculation=ENI\_CAMPUS\_202312456, promotion=EDMW0002), formateur=Formateur(email=abaille@campus-eni.fr, nom=BAILLE, prenom=Anne-Lise), cours=null)]

fr.eni.demo.aop.LogAspect ENI AOP - @After – FINALLY

fr.eni.demo.aop.LogAspect ENI AOP - @Around - fr.eni.demo.dal.AvisRepository/findByNoteCours:139 ms

* Advice after returning permet de récupérer la liste des Avis retournée

## Advice after throwing

* Il s’exécute si la méthode lance une exception
* C’est équivalent à CATCH des blocs TRY-CATCH-FINALLY
* Ajouter une méthode, elle prend en paramètre Throwable la classe mère de toutes les exceptions de Java. Nous pourrons ainsi tracer les erreurs
* Utiliser @AfterThrowing et paramétré  « throwing »

@AfterThrowing(pointcut = "execution(public \* fr.eni.demo.dal.AvisRepository.findByNote\*(..))", throwing = "re")

**public** **void** afterThrow(Throwable re) {

***log***.warn("ENI AOP - @AfterThrowing - CATCH - " + re.getMessage() + "[" + re.getCause() + "]");

}

### Exécution :

* Tester findByNoteCours. Pour avoir une exception au niveau Repository ; le plus simple est d’éteindre le serveur de base de données
* Dans Docker, stopper votre serveur
* Remarquez dans les logs de l’IDE
  + Il faut attendre le timeout de connexion

fr.eni.demo.aop.LogAspect ENI AOP - @Before - findByNoteCours

o.s.data.mongodb.core.MongoTemplate find using query: { "course\_note" : 3} fields: Document{{}} for class: class fr.eni.demo.bo.Avis in collection: reviews

fr.eni.demo.aop.LogAspect ENI AOP - @AfterThrowing - CATCH - Timed out after 30000 ms while waiting to connect. Client view of cluster state is {type=UNKNOWN, servers=[{address=localhost:27017, type=UNKNOWN, state=CONNECTING, exception={com.mongodb.MongoSocketOpenException: Exception opening socket}, caused by {java.net.ConnectException: Connection refused: no further information}}][com.mongodb.MongoTimeoutException: Timed out after 30000 ms while waiting to connect. Client view of cluster state is {type=UNKNOWN, servers=[{address=localhost:27017, type=UNKNOWN, state=CONNECTING, exception={com.mongodb.MongoSocketOpenException: Exception opening socket}, caused by {java.net.ConnectException: Connection refused: no further information}}]]

fr.eni.demo.aop.LogAspect ENI AOP - @After - FINALLY

* Vous avez les traces de @Before, @AfterThrowing et @After
  + Pas de trace de @Around 🡪 il est en échec. Il ne peut être retourné que si tout va bien