Clef composite

# Démonstration 6 du module 3

|  |
| --- |
| Les objectifs de cette démonstration :   * Déclaration d’une classe représentant une clef composée de plusieurs attributs * Utilisation de la classe de la clef composite comme identifiant pour un Document |

Déroulement

# Contexte

* Continuer l’application précédente
* Dans cette itération, nous allons ajouter la notion de cours :
  + Un cours a
    - Un identifiant qui est composé de la référence et de la filière
    - Un titre
    - Et une durée (en nombre de jours)
  + Voici le diagramme de classes associées :

Une image contenant texte, capture d’écran, Police, ligne

Description générée automatiquement

* + Voici la structure BSON désirée dans MongoDB :

{

\_id: {

reference: 'M360',

computer\_science\_course: 'Développement'

},

title: 'Java Frameworks - API Web',

duration: 10,

\_class: 'fr.eni.demo.bo.clefcomposite.Cours'

}

* + La collection doit s’appeler computer\_course

# Création de la classe gérant la clef composite :

**package** fr.eni.demo.bo.clefcomposite;

**import** java.io.Serializable;

**import** org.springframework.data.mongodb.core.mapping.Field;

**import** lombok.\*;

@Data

@AllArgsConstructor

@NoArgsConstructor

@Builder

**public** **class** CoursId **implements** Serializable{

**private** **static** **final** **long** ***serialVersionUID*** = 1L;

**private** String reference;

@Field(name = "computer\_science\_course")

**private** String filiere;

}

* Elle déclare les 2 attributs de la clef
* Il faut qu’elle respecte le design pattern POJO
  + Pour cela ajout des annotations de Lombok :
  + @Data
  + @NoArgsConstructor
  + @AllArgsConstructor
  + @Builder
* Elle doit implémenter Serializable car elle est embarquée

# Création de la classe Cours qui est un Document :

**package** fr.eni.demo.bo.clefcomposite;

**import** org.springframework.data.annotation.Id;

**import** org.springframework.data.mongodb.core.mapping.\*;

**import** lombok.\*;

@Data

@AllArgsConstructor

@NoArgsConstructor

@Builder

@Document(collection = "computer\_course")

**public** **class** Cours {

@Id

**private** CoursId id;

@Field(name = "title")

**private** String titre;

@Field(name = "duration")

**private** **int** duree;

}

* Cette classe est très similaire à la classe Stagiaire
* Les différences sont
  + L’annotation @Id est placée sur un attribut de type CoursId
  + C’est le type de l’identifiant qui indique à Spring Data MongoDB qu’il faut gérer une clef composite

# Création de CoursRepository

* Créer l’interface CoursRepository qui hérite de MongoRepository<Cours, CoursId>
* La complexité est de bien faire attention d’utiliser la classe de la clef composite comme classe pour l’identifiant

**package** fr.eni.demo.dal;

**import** org.springframework.data.mongodb.repository.MongoRepository;

**import** fr.eni.demo.bo.clefcomposite.\*;

**public** **interface** CoursRepository **extends** MongoRepository<Cours, CoursId> {

}

# Création d’un test unitaire pour valider la clef composite

**package** fr.eni.demo.bo.clefcomposite;

**import** **static** org.assertj.core.api.Assertions.*assertThat*;

**import** java.util.Optional;

**import** org.junit.jupiter.api.Test;

**import** org.springframework.beans.factory.annotation.Autowired;

**import** org.springframework.boot.test.context.SpringBootTest;

**import** fr.eni.demo.bo.clefcomposite.\*;

**import** fr.eni.demo.dal.CoursRepository;

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

@Slf4j

@SpringBootTest

**class** TestCoursDocument {

@Autowired

CoursRepository coursRepository;

@Test

**void** test\_save\_cours() {

CoursId coursId = CoursId

.*builder*()

.reference("M360")

.filiere("Développement")

.build();

Cours cours = Cours

.*builder*()

.id(coursId)

.titre("Java Frameworks - API Web")

.duree(10)

.build();

coursRepository.save(cours);

//Vérification en base

Optional<Cours> opt = coursRepository.findById(coursId);

*assertThat*(opt).isNotNull();

*assertThat*(opt.isPresent()).isTrue();

Cours coursDB = opt.get();

*assertThat*(coursDB).isNotNull();

*assertThat*(coursDB.getTitre()).isNotNull();

*assertThat*(coursDB.getDuree()).isEqualTo(10);

***log***.info(coursDB.toString());

}

}

## Exécution :

* Spring Data MongoDB a inséré le Document et créé la Collection en 1 fois :
  + Pas de différence significative de ce point de vue

Saving Document containing fields: [\_id, title, duration, \_class]

* + Il faut constater, qu’on ne voit rien de particulier pour les 2 attributs de la clef composite
  + Il y a seulement indiqué \_id comme champs dans la requête
* Lors de la récupération depuis la base du Cours
  + Au niveau des traces :

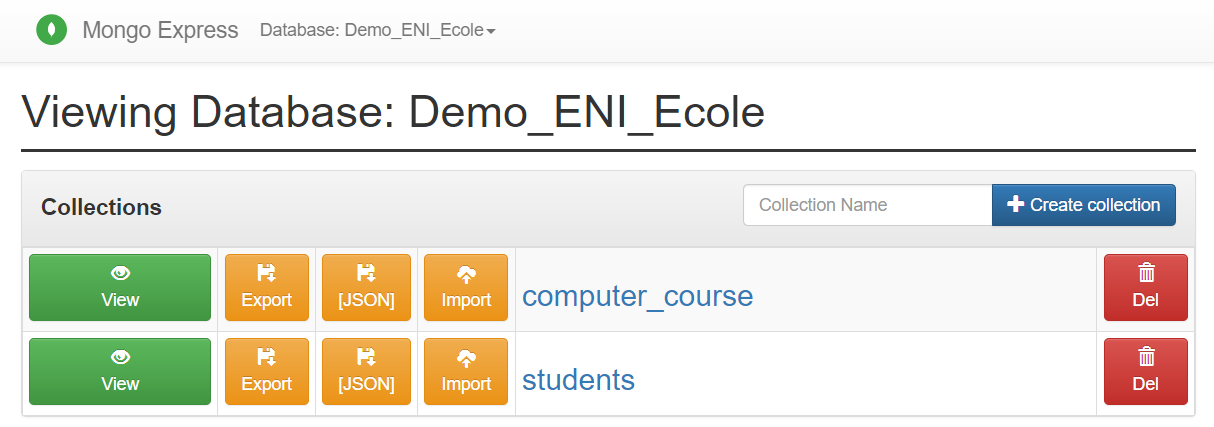
findOne using query: { "id" : { "$java" : CoursId(reference=M360, filiere=Développement) } } fields: Document{{}} for class: class fr.eni.demo.bo.clefcomposite.Cours in collection: computer\_course

* + C’est dans cette requête, que l’on constate la présence des 2 classes et des attributs spécifiques : reference et filiere
  + Le Cours remonté, a bien toutes ses informations :

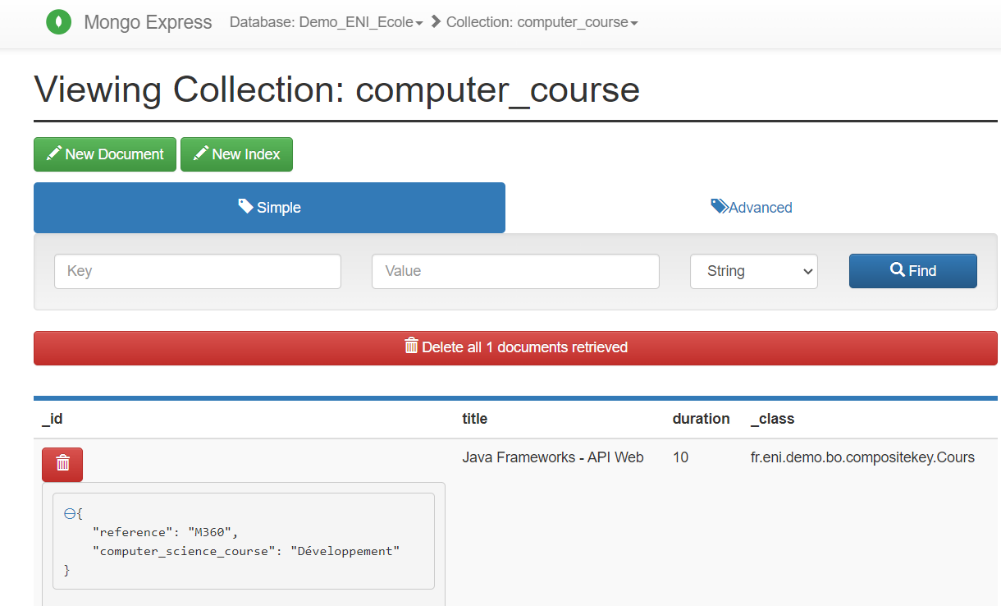
Cours(id=CoursId(reference=M360, filiere=Développement), titre=Java Frameworks - API Web, duree=10)

## En Base :

* La Collection computer\_course est présente



* Regarder dans le détail son contenu :



* Cliquer sur le Document existant :



* + Il y a bien un identifiant \_id composé de 2 propriétés : reference et computer\_science\_course