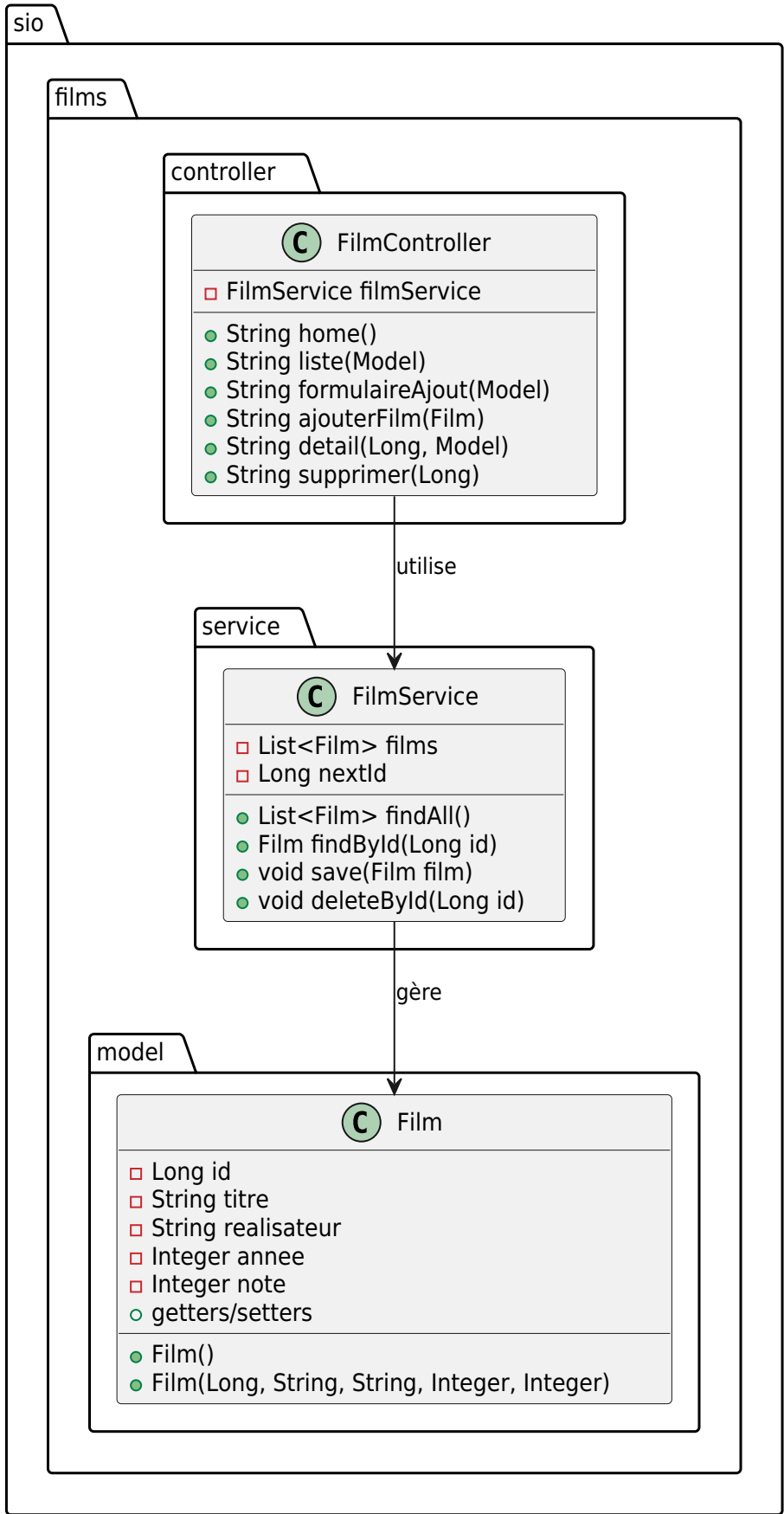


TD3 - Spring Film

Introduction : Routage, controllers, services, entities, views

1. Architecture Générale



2. Fonctionnalités à implémenter

Routes à créer

| Méthode | Route | Action | Vue retournée |
|---------|-----------------------|----------------------------|----------------------|
| GET | / | Page d'accueil | index.html |
| GET | /films | Liste tous les films | liste.html |
| GET | /films/nouveau | Affiche formulaire ajout | formulaire.html |
| POST | /films | Enregistre un nouveau film | Redirection → /films |
| GET | /films/{id} | Détail d'un film | detail.html |
| POST | /films/{id}/supprimer | Supprime un film | Redirection → /films |

3. Modèle de données

Classe Film

Attributs :

- id : Long (identifiant unique)
- titre : String (obligatoire)
- realisateur : String
- annee : Integer (entre 1900 et 2100)
- note : Integer (entre 1 et 5)

Méthodes :

- Constructeur vide
- Constructeur avec tous les paramètres
- Getters et setters pour tous les attributs

4. Service FilmService

Annotation : @Service

Attributs :

- films : List<Film> (stockage en mémoire)
- nextId : Long (compteur pour générer les IDs)

Constructeur :

Initialiser la liste avec 3-4 films de démo

Exemples :

- Inception, Nolan, 2010, note 5
- Matrix, Wachowski, 1999, note 5
- Pulp Fiction, Tarantino, 1994, note 5

Méthodes :

findAll() : List<Film>

- Retourne tous les films

findById(Long id) : Film

- Recherche un film par son ID
- Retourne null si non trouvé

save(Film film) : void

- Si `film.id == null` → nouveau film → assigner `nextId++`
- Sinon → mise à jour → chercher et remplacer le film existant
- Ajouter à la liste

deleteById(Long id) : void

- Supprimer le film de la liste
- Utiliser `films.removeIf(f → f.getId().equals(id))`

5. Contrôleur FilmController

Annotation : @Controller

Injection de dépendance :

```
@Autowired
private FilmService filmService;
```

Méthodes à implémenter :

@GetMapping("/")

```
Méthode : home()
Retour : "index"
```

@GetMapping("/films")

```
Méthode : liste(Model model)
- Récupérer tous les films via le service
- Ajouter au modèle : model.addAttribute("films", ...)
Retour : "liste"
```

@GetMapping("/films/nouveau")

```
Méthode : formulaireAjout(Model model)
- Créer un film vide
- Ajouter au modèle : model.addAttribute("film", new Film())
Retour : "formulaire"
```

@PostMapping("/films")

Méthode : ajouterFilm(@ModelAttribute Film film)
- Sauvegarder via le service
- Redirection : "redirect:/films"

@GetMapping("/films/{id}")

Méthode : detail(@PathVariable Long id, Model model)
- Récupérer le film par ID
- Ajouter au modèle
Retour : "detail"

@PostMapping("/films/{id}/supprimer")

Méthode : supprimer(@PathVariable Long id)
- Supprimer via le service
- Redirection : "redirect:/films"

6. Vues Mustache

index.html

Contenu :

- Titre "Bienvenue sur le gestionnaire de films"
- Lien vers /films ("Voir tous les films")

liste.html

Contenu :

- Titre "Liste des films"
- Tableau avec colonnes : Titre, Réalisateur, Année, Note, Actions
- Boucle `{{#films}}` pour afficher chaque film
- Pour chaque film :
 - Lien "Détail" → /films/{{id}}
 - Formulaire POST "Supprimer" → /films/{{id}}/supprimer
- Bouton "Ajouter un film" → /films/nouveau

formulaire.html

Contenu :

- Titre "Ajouter un film"
- Formulaire POST vers /films
- Champs :
 - titre (text, required)

- réalisateur (text)
- annee (number)
- note (number, min=1, max=5)
- Boutons : Submit "Enregistrer", Lien "Annuler" → /films

detail.html

Contenu :

- Titre du film {{titre}}
- Affichage :
 - Réalisateur : {{realisateur}}
 - Année : {{annee}}
 - Note : {{note}}/5
- Lien "Retour à la liste" → /films
- Formulaire POST "Supprimer ce film" → /films/{{id}}/supprimer

7. Configuration

application.properties

```
# Port du serveur (optionnel)
server.port=8080

#Mustache configuration
spring.mustache.charset=UTF-8
spring.mustache.prefix=classpath:/templates/
spring.mustache.suffix=.html
```

8. Dod

Fonctionnel :

- Affichage de la liste des films
- Ajout d'un nouveau film
- Détail d'un film
- Suppression d'un film
- Navigation fluide entre les pages

Architecture :

- Séparation Contrôleur / Service / Modèle
- Utilisation correcte des annotations Spring
- Templates Mustache bien structurés

Bonus :

- Validation des champs du formulaire
- Messages de confirmation
- Gestion d'erreur si film introuvable

From:

<http://slamwiki2.kobject.net/> - **SlamWiki 2.1**

Permanent link:

<http://slamwiki2.kobject.net/sio/bloc2/2a/td3?rev=1761093315>

Last update: **2025/10/22 02:35**

