

# TD2 : Système de Gestion de Bibliothèque

## Partie 1 : Classe de base (15 min)

Créez une classe Livre avec :

1. Attributs privés : titre, auteur, isbn, disponible (boolean)
2. Constructeur avec validation (ISBN ne peut pas être null/vide)
3. Getters/setters appropriés
4. Méthode emprunter() qui change le statut si disponible
5. Méthode retourner() qui remet le livre disponible
6. toString() bien formaté

## Partie 2 : Héritage (10 min)

Créez une classe LivreNumerique qui hérite de Livre :

1. Attribut supplémentaire : tailleFichier (en Mo)
2. Constructeur approprié
3. Redéfinition de toString()
4. Les livres numériques sont toujours disponibles (redéfinir emprunter())

## Partie 3 : Gestion des exceptions (10 min)

Créez une exception personnalisée **LivreIndisponibleException** et modifiez la méthode emprunter() pour la lancer quand nécessaire.

## Partie 4 : Collection et gestion (15 min)

Créez une classe Bibliotheque avec :

1. Une ArrayList<Livre> privée
2. Méthode ajouterLivre(Livre livre) avec gestion des doublons (même ISBN)
3. Méthode emprunterLivre(String isbn) avec gestion d'exceptions
4. Méthode listerLivresDisponibles() qui retourne une liste filtrée
5. Méthode rechercherParAuteur(String auteur)

## Partie 5 : Tests unitaires (10 min)

Créez une classe BibliothequeTest avec les tests JUnit suivants :

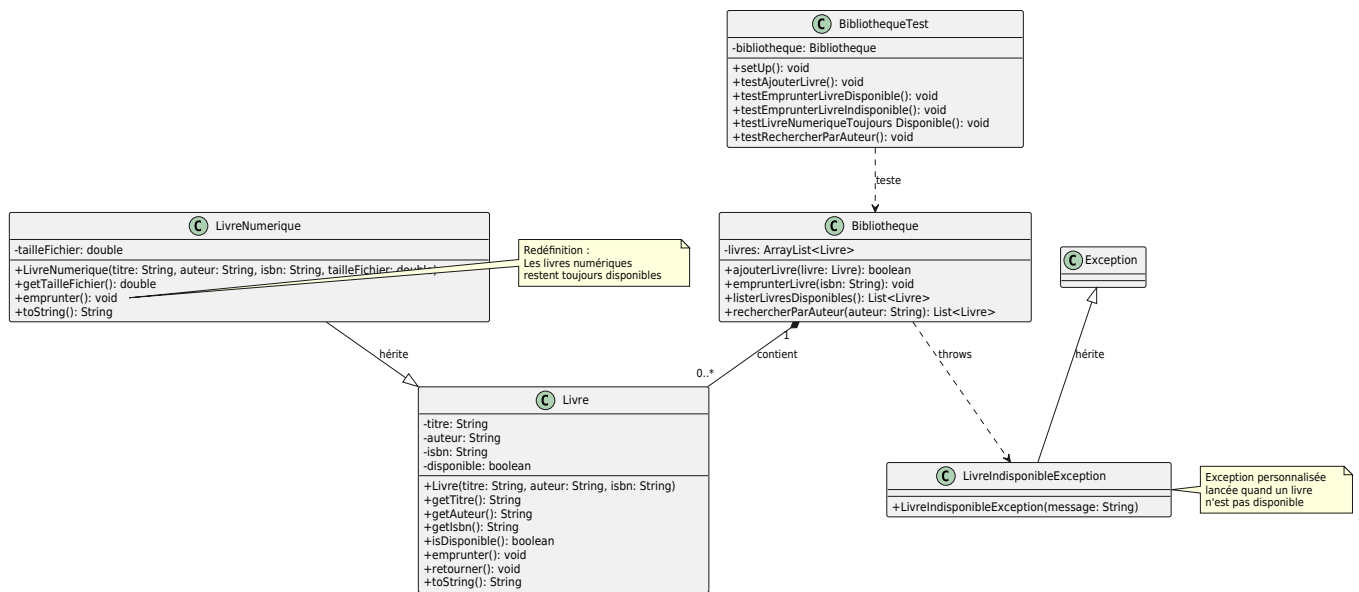
1. testAjouterLivre() : vérifier l'ajout normal et la gestion des doublons
2. testEmprunterLivreDisponible() : vérifier qu'un livre disponible peut être emprunté
3. testEmprunterLivreIndisponible() : vérifier que l'exception est bien lancée
4. testLivreNumeriqueToujours Disponible() : vérifier que les livres numériques restent disponibles
5. testRechercherParAuteur() : vérifier le filtrage par auteur

Annotations à utiliser : `@Test`, `@BeforeEach` (pour initialiser une bibliothèque de test) Assertions à utiliser : assertEquals(), assertTrue(), assertThrows()

### Critères d'évaluation

- Encapsulation respectée
- Héritage correct avec redéfinition
- Constructeurs avec validation
- Exception personnalisée utilisée
- Collections manipulées correctement
- Tests unitaires pertinents et réussis

### Diagramme UML



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

Permanent link: <http://slamwiki2.kobject.net/sio/bloc2/2a/td2?rev=1757897502>

Last update: 2025/09/15 02:51

