LE GOUVERNEMENT
DU GRAND-DUCHÉ DE LUXEMBOURG
Ministère de l'Éducation nationale,
de l'Enfance et de la Jeunesse

EXAMEN DE FIN D'ÉTUDES SECONDAIRES CLASSIQUES
2021

BRANCHE	SECTION(S)	ÉPREUVE ÉCRITE
SCIPR	**CI, CLI**	*Durée de l'épreuve :* *180 minutes* *Date de l'épreuve :* 31/05/2021

Dans votre répertoire de travail (à définir par chaque lycée), vous trouverez un dossier nommé **EXAMEN_CI**. Renommez ce dossier en remplaçant le nom actuel par votre code de l'examen (exemple de notation : **LXY_CI2_05**). Tous vos fichiers devront être sauvegardés à l'intérieur de ce dossier, qui sera appelé **votre dossier** par la suite.

CENTRE DE DOCUMENTATION ET D'INFORMATION

Partie modélisation (10 points)

- La partie modélisation doit être rendue avant que la partie implémentation puisse être débutée. Cette partie est à réaliser sur papier.
- La partie modélisation est à rendre après 35 minutes au plus tard, mais vous être libre de la rendre plus tôt.

Le CDI (Centre de Documentation et d'Information) vous contacte pour développer une application permettant de faciliter les emprunts et retours des livres de leur collection. En effet, bien que chaque livre dispose d'un numéro d'identification unique, les emprunts se faisaient jusqu'à maintenant manuellement sur papier ce qui était très lent et laborieux.

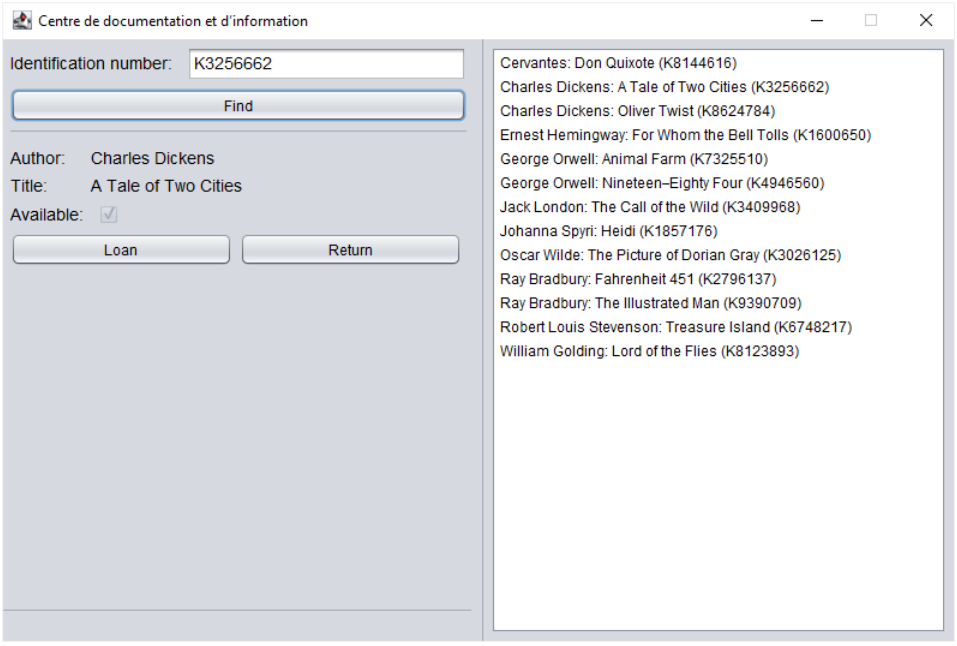
Principe de l'application

Le CDI dispose d'un fichier JSON contenant la collection entière de leurs livres. Un livre a un numéro d'identification (qui commence par la lettre majuscule « K » suivi d'un nombre de sept chiffres), un titre ainsi qu'un auteur. Chaque livre n'est disponible qu'une seule fois.

De plus, le CDI a un fichier texte qui contient les emprunts des livres, un emprunt par ligne. Les emprunts sont regroupés par livre et se trouvent en ordre chronologique. Une ligne de ce fichier texte a le format suivant : *« numéro d'identification »;« date début de l'emprunt »;« date fin de l'emprunt »*. Si le livre est actuellement emprunté, alors la ligne contient la valeur « ? » comme date fin.

Le CDI souhaite une application avec les fonctionnalités suivantes :

1. Au lancement, l'application lit tous les livres et tous les emprunts depuis les fichiers respectifs. La collection des livres est affichée triée dans une liste, qui ne permet aucune interaction.



2. Les livres et les emprunts doivent être stockés dans des structures de données adéquates pendant l'exécution de l'application. Ceci permettra à l'avenir l'ajout de fonctionnalités supplémentaires comme des statistiques.
3. Le responsable du CDI peut entrer dans le champ texte le numéro d'identification d'un livre pour le rechercher et en afficher les informations pertinentes. Grâce aux données contenues dans la structure de données des emprunts, l'application peut afficher si le livre en question est actuellement disponible pour être emprunté. Cette recherche d'un livre avec son numéro d'identification doit être très rapide (peu importe le nombre de livres chargés) car c'est la principale interaction avec le logiciel et se fait plusieurs fois par minute aux heures de pointe.
4. Les boutons « Loan » et « Return » permettent d'emprunter ou de retourner le livre dont les informations sont actuellement affichées dans la partie gauche de l'interface (si cette opération est possible), ce qui met aussi à jour le fichier texte contenant les emprunts.

Travail à réaliser

Réaliser un modèle de classe du **modèle** de l'application sur papier en respectant les conventions UML usuelles.

1. Limitez-vous uniquement aux attributs des classes.
2. Rajoutez dans votre modèle toutes les méthodes nécessaires pour afficher la collection triée des livres dans la liste.
3. Vous pouvez compléter votre modèle de classe par des explications supplémentaires.

LE GOUVERNEMENT
DU GRAND-DUCHÉ DE LUXEMBOURG
Ministère de l'Éducation nationale,
de l'Enfance et de la Jeunesse

EXAMEN DE FIN D'ÉTUDES SECONDAIRES CLASSIQUES
2021

BRANCHE	SECTION(S)	ÉPREUVE ÉCRITE
SCIPR	**CI, CLI**	*Durée de l'épreuve :* *180 minutes* *Date de l'épreuve :* *08/06/2020*

Partie implémentation (50 points)

Ouvrez dans votre dossier le projet **CDI** mis à disposition. Complétez l'application en vous basant sur la version exécutable, le diagramme UML et les instructions données par la suite.

Classe « Loan » (2 points)

La classe **Loan** représente un emprunt d'un livre. Les dates sont de type String et se trouvent dans le format ISO 8601 : YYYY-MM-DD. (Exemple : 2021-02-17)

Attributs, constructeur et méthodes : (2 points)

- **startDate** représente la date début de l'emprunt.
- **endDate** représente la date fin de l'emprunt. La valeur de cet attribut est « ? » si le livre n'a pas encore été retourné.
- Le **constructeur** initialise la valeur des attributs **startDate** et **endDate**. La valeur initiale de **endDate** est toujours « ? ».
- **getStartDate(), getEndDate()** et **setEndDate()** sont des accesseurs/modificateurs des attributs respectifs.
- La méthode **hasBeenReturned()** retourne *true* si le livre a été retourné et que cet emprunt est donc terminé.

Classe « Node » (4 points)

La classe **Node** représente un nœud d'une liste chaînée permettant de stocker un emprunt. Implémentez la classe **Node** selon le schéma UML. La classe comprend un constructeur, des accesseurs et manipulateurs ainsi que les méthodes suivantes :

- **hasNext()** retourne *true* si le nœud possède un successeur et *false* s'il est le dernier nœud de la liste.
- **getLast()** est une méthode récursive qui retourne le dernier nœud de la liste.

Classe « Book » (18 points)

La classe **Book** représente un livre avec un numéro d'identification, un titre ainsi qu'un auteur. Cette classe sauvegarde aussi les emprunts du livre en question dans une liste chainée implémentée manuellement. L'attribut **root** représente le premier nœud (la tête) de cette liste. Si la liste est vide, alors ce livre n'a pas encore été emprunté. Les nouveaux emprunts sont toujours rajoutés à la fin de la liste. L'emprunt le plus récent se trouve donc à la fin de la liste. (1 point)

La classe comprend un constructeur ainsi que les accesseurs **getTitle()** et **getAuthor()**. La méthode **toString()** retourne une représentation textuelle du livre (cf. exécutable). (1 point)

La méthode privée **getLast()** retourne le dernier nœud de la liste en utilisant la méthode récursive **getLast()** de la classe Node. Elle retourne *null* si la liste est vide. (2 points)

La méthode **isAvailable()** retourne *true* si le livre n'est actuellement pas emprunté. Rappel : la liste d'un livre qui n'a jamais été emprunté est vide et l'emprunt le plus récent se trouve toujours à la fin de la liste. (2 points)

La méthode **loanBook()** prend en paramètre la date début d'un emprunt, crée un nouveau emprunt et le rajoute à la fin de la liste. Si la méthode est appelée alors que le livre n'est actuellement pas disponible pour un emprunt, la méthode lance une exception du type *InvalidLoanOperation*[1] avec comme message « Book <*book identifier*> is not available for a loan! ». (3.5 points)

La méthode **returnBook()** prend en paramètre la date fin d'un emprunt et termine l'emprunt le plus récent avec cette date. Si la méthode est appelée alors que le livre n'est actuellement pas emprunté, la méthode lance une exception du type *InvalidLoanOperation* avec comme message « Book <*book identifier*> has not been loaned! ». (2.5 points)

La méthode **saveLoansToFile()** prend en paramètre un flux de sortie texte de type **PrintWriter** et écrit dans ce flux tous les emprunts de ce livre, un emprunt par ligne. Une ligne a le format suivant : *« numéro d'identification »;« date début de l'emprunt »;« date fin de l'emprunt »*. Si le livre est actuellement emprunté, alors la date fin est la valeur par défaut, c.-à-d. la valeur « ? ». (2 points)

La classe **Book** doit implémenter l'interface **Comparable** pour que deux livres puissent se comparer entre eux. La comparaison entre deux livres se fait d'abord sur base des noms des auteurs et, si les noms des auteurs sont égaux, sur base des titres des livres (cf. exécutable). (4 points)

Classe « CDI » (10 points)

La classe **CDI** représente la collection entière de tous les livres. Ces livres sont sauvegardés dans une table associative **hmBooks** qui associe au numéro d'identification du livre (la clé) le livre en question (la valeur). La classe comprend aussi la méthode d'accès à la table associative **get()**. (1 point)

La méthode **readLoansFromFile()** lit les emprunts des livres depuis le fichier texte « loans.txt ». Le format d'un tel emprunt est celui produit par la méthode **saveLoansToFile()** de la classe **Book**. Les emprunts se trouvent dans l'ordre chronologique avec l'emprunt le plus récent pour un certain livre se trouvant dans la dernière ligne de ses emprunts (cf. exemple ci-dessous).

Si la liste des emprunts contient un numéro d'identification qui ne correspond à aucun livre de la collection, alors la méthode lance une exception du type *IllegalArgumentException* avec comme message « Unknown identification number <book identifier> in line <line number> » en indiquant

[1] La classe **InvalidLoanOperation** vous est mise à disposition. Il s'agit d'une *checked exception*.

dans le message le numéro de la ligne dans laquelle l'erreur s'est produite. La première ligne a le numéro 1. De plus, toutes les exceptions doivent être transmises à la méthode appelante. (5 points)

Exemple :

```
K7325510;2021-02-03;2021-02-13
K7325510;2021-02-15;2021-02-18
K8123893;2020-01-01;2020-01-02
K8123893;2020-01-03;2021-02-06
K8123893;2021-03-10;?
```

1. Le livre *K7325510* a été emprunté deux fois et est actuellement disponible.
2. Le livre *K8123893* a été emprunté trois fois et l'emprunt actuel n'est pas terminé. Il n'est donc actuellement pas disponible.

La méthode **saveLoansToFile()** écrit les emprunts des livres dans le fichier texte « loans.txt » en utilisant la méthode **saveLoansToFile()** de la classe **Book**. L'ordre des livres ne joue aucune importance aussi longtemps que les emprunts sont regroupés par livre et se trouvent en ordre chronologique. Toutes les exceptions doivent être transmises à la méthode appelante. (2 points)

La méthode **toArray()** retourne un tableau d'objets qui correspondent aux livres de la table associative. Ce tableau doit être trié. (2 points)

Classe « MainFrame » (16 points)

Implémentez la classe **MainFrame** en vous basant sur le schéma UML et l'exécutable fourni. L'attribut **cdi** représente le modèle de l'application. L'attribut **book** représente le livre recherché et trouvé dont les informations sont affichées dans l'interface. (0.5 points)

La méthode **updateView()** actualise les valeurs affichées en se basant entre autres sur la valeur de l'attribut **book**. (2.5 points)

La méthode **showMessage()** prend en paramètre un message d'erreur et affiche ce message pour exactement 3 secondes dans le libellé *outputLabel*. Attention à la situation où deux messages sont affichés rapidement l'un après l'autre. (3 points)

La méthode **loadBooksFromJSON()** lit les livres depuis le fichier JSON « books.json » grâce à la librairie « gson » mise à disposition. Le fichier contient une seule ligne de texte qui correspond à la sérialisation d'un objet de type **CDI** sous format JSON. Toutes les exceptions doivent être transmises à la méthode appelante. (2 points)

Au lancement de l'application, celle-ci lance la méthode **initBooks()** qui lit d'abord les livres et ensuite les emprunts. Finalement, l'interface est mise à jour. Si une exception se produit alors le message d'erreur de l'exception est affiché. (2.5 points)

Le bouton « Find » recherche le livre dont le numéro d'identification se trouve dans le champ texte et met à jour l'interface. Si le champ texte est vide, alors le message « Please enter a valid book identification number » s'affiche. Si le numéro d'identification ne correspond à aucun livre, alors le message « No book found with given identification number » s'affiche. (1.5 points)

Le bouton « Loan » emprunte le livre trouvé et met à jour le fichier texte des emprunts. Si le livre n'était pas disponible pour un emprunt, alors le message d'erreur y relatif (cf. **loanBook()** dans **Book**) est affiché. (2.5 points)

Astuce : utiliser l'instruction *java.time.LocalDate.now().toString()* pour avoir la date actuelle.

Le bouton « Return » retourne le livre trouvé et met à jour le fichier texte des emprunts. Si le livre n'était actuellement pas emprunté, alors le message d'erreur y relatif (cf. **returnBook()** dans **Book**) est affiché. (1.5 points)

Schéma UML

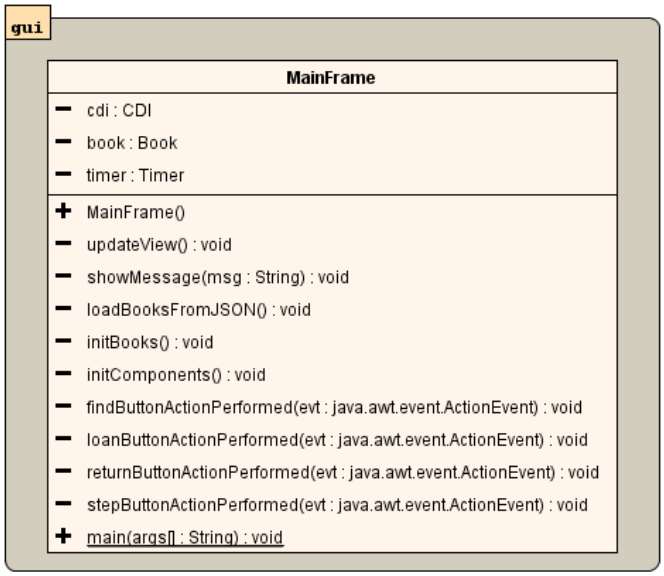


Figure 1: Schéma UML (simplifié !) du paquet gui.

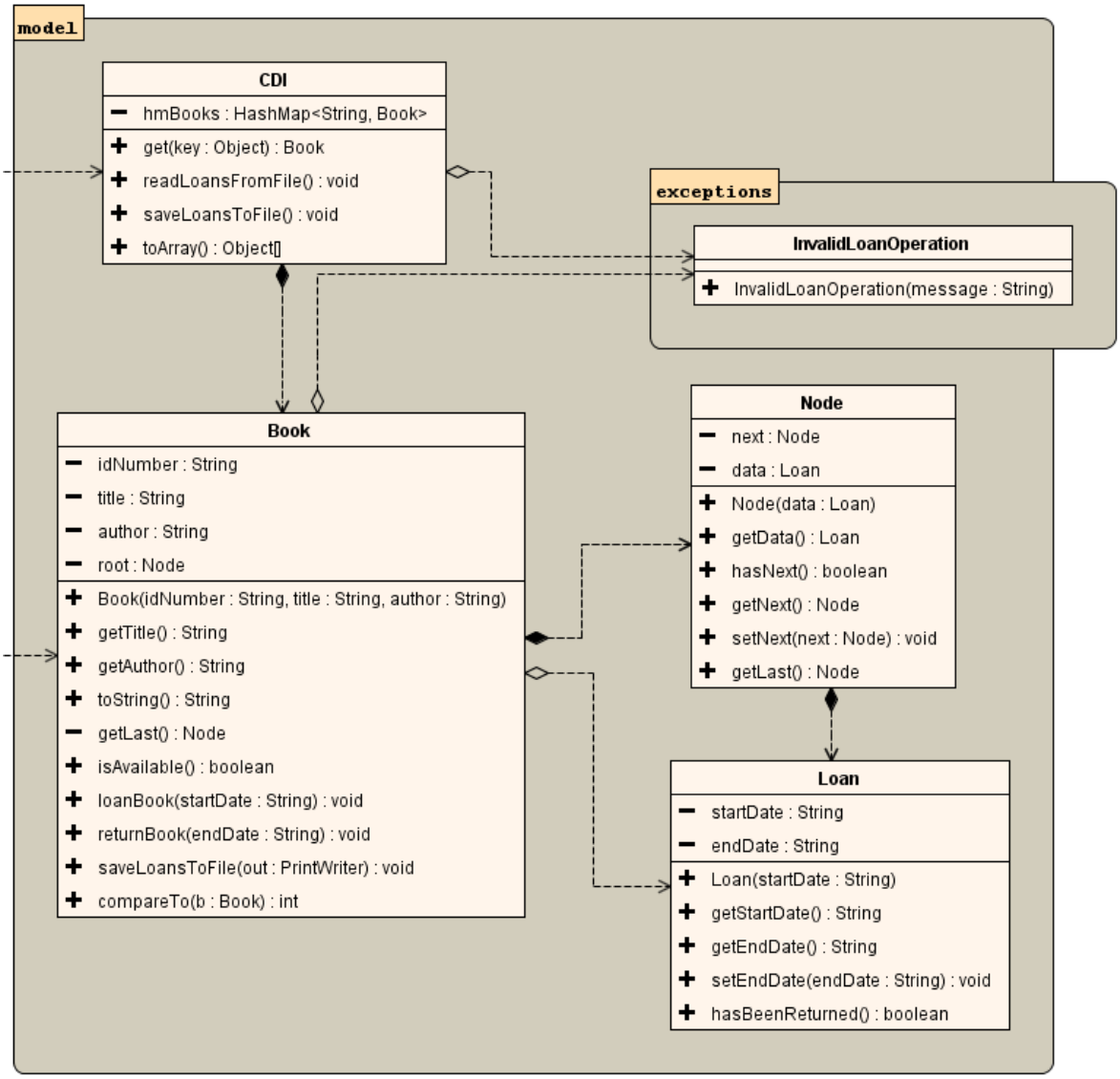


Figure 2: Schéma UML des paquets model et exceptions.