

# TD - BIBLIOTHÈQUE

## Langage TypeScript - Bibliothèque

### *Annexe à utiliser : Annexe 1*

On désire réaliser un programme permettant de gérer une petite bibliothèque municipale.

Pour cela on a analysé qu'on avait besoin d'une classe Bibliothèque, d'une classe Adherent et d'un ensemble de classes de Ouvrage qui présentent les caractéristiques suivantes (voir diagramme de classe UML Annexe 1) :

- La bibliothèque comprend un ensemble d'ouvrages et un ensemble d'adhérents
- Les ouvrages :
  - Ils sont caractérisés par un identifiant, un titre, une date de parution.
  - Ils n'ont pas d'existence réelle.
- Les volumes sont soit des livres, soit des bandes dessinées.
  - Les volumes ont un auteur.
  - Les volumes n'ont pas d'existence réelle.
  - La méthode « Emprunter » permet d'emprunter un livre en fonction du nombre d'exemplaires disponibles (voir plus bas).
- Les BD ont un nom de dessinateur (qui correspond à l'auteur du volume) et un nom de scénariste.
- Les livres ont un code ISBN
- Les adhérents ont des caractéristiques (nom, prénom).
- Les adhérents peuvent emprunter des livres ou des BD. On doit pouvoir savoir à tout moment quels sont les livres ou BD empruntés par un adhérent.
  - La classe « Adherent » a donc une liste d'objet « Emprunt ».
  - La méthode « Emprunte(**Volume** volume) » permet d'ajouter l'emprunt d'un volume pour cet adhérent.
  - La méthode « AfficherEmprunts » affiche les emprunts de cet adhérent.
- La classe « Emprunt » contient une référence sur un « Volume » et sur un « Adhérent ».
  - La méthode « Description » permet d'afficher le volume emprunté et l'emprunteur (adhérent)

Un livre ou BD peut être empruntable si et seulement si le nombre total empruntés est inférieur au nombre d'exemplaire.

#### Exemple :

Le livre « *Systèmes multi-agents* » existe en 2 exemplaires.

Il est emprunté 1 fois. La classe Volume permet de déterminer qu'il reste 1 livre à emprunter et donc que le livre est empruntable.

Le développeur a écrit un jeu de test (annexe 2) qui a produit une sortie Console (annexe 3).

**Remarque** : Vous devez ajouter les données, propriétés et/ou méthodes que vous jugez nécessaires dans les classes utilisées en le précisant.

Exemple de flux d'un emprunt :

- La classe « Bibliothèque » fait l'emprunt : `bibliotheque.Emprunte(adherent, volume)`
  - Elle vérifie si le « volume » existe
  - Puis elle demande à la classe Adherent de faire l'emprunt
- La classe Adherent demande au « volume » de l'emprunter
- La classe Volume avec la méthode « `bool Emprunter()` » vérifie si le volume est disponible ; si c'est le cas le nombre d'exemplaire est incrémenté et la méthode retourne vrai ; sinon retourne faux.
- Si l'emprunt est réalisable alors la classe « Adherent » ajoute ce livre à sa liste d'emprunts

Travail à faire	
<b>1.1</b>	Compléter les classes suivantes (avec propriétés, constructeur et méthodes) : <ul style="list-style-type: none"><li>• Ouvrage</li><li>• Volume</li><li>• Livre</li><li>• BandeDessine</li><li>• Emprunt</li></ul> Ajouter la propriété « <code>NombreExemplaireDisponible</code> » dans la classe « Volume » : cette méthode affiche le nombre d'exemplaire disponible
<b>1.2</b>	Compléter la classe « Bibliothèque » permettant de faire une mini-gestion d'ouvrages : <ul style="list-style-type: none"><li>• Gestion d'adhérents : Ajout uniquement</li><li>• Gestion des ouvrages : Ajout, Suppression, Recherche par numéro et par titre</li><li>• Attention : vous ne pouvez supprimer un adhérent que s'il n'a aucun d'emprunts en cours</li></ul>
<b>1.3</b>	Compléter la classe « Bibliothèque » : <ul style="list-style-type: none"><li>• Compléter la méthode « <code>Emprunte</code> ».</li><li>• Compléter la méthode <code>AfficherEmprunteursOuvrage</code> permettant de lister les emprunts par adhérent.</li></ul> Cette liste doit afficher : <ul style="list-style-type: none"><li>◦ Le nom et prénom adhérent</li><li>◦ Le nombre d'emprunts total</li></ul> La liste des emprunts : titre de l'ouvrage et date d'emprunt

Annexe 1 : Diagramme de classe (style Java)

