

# 2D-Library

Cyril CHARLON

Henry-Julien DESGRANGES

Anthony GIRARDET

Douha KARIM

Eric TANG

# SOMMAIRE

- ▶ Présentation du projet 2D-Library
- ▶ Fonctionnalités réalisées par itération
- ▶ Réalisation technique
- ▶ Démonstrations
- ▶ Difficultés rencontrées
- ▶ Axes d'amélioration

# Présentation du projet 2D-Library

- ▶ Objectif du projet : Représenter une bibliothèque de livres au format SVG.
- ▶ Idée principale : Avoir une vue d'ensemble de nos livres, potentiellement triée en fonction des préférences de l'utilisateur.
- ▶ Pouvoir personnaliser sa bibliothèque :
  - ▶ Choisir la couleur de nos livres,
  - ▶ Changer la couleur des étagères et de la bibliothèque,
  - ▶ Trier les livres en fonctions de l'auteur, la date de parution ou le titre du livre,
  - ▶ Pouvoir partager sa bibliothèque avec ses amis en l'imprimant au format poster.

# Réalisation technique

| Itération 1   | Itération 2  | Itération 3  | Itération 4   |
|---|--|--|---|
| Mise à jour des dépendances du fichier pom.xml  | Création de l'interface JavaSearcher qui contient une méthode search qui retourne une liste de livre correspondant aux critères de l'utilisateur | Utilisation de la bibliothèque Univocity pour parser le fichier CSV  | Modification de SearchData en fonction des remarques du professeur  |
| Configuration du Travis   | SearchData permet de faire une recherche dans la bibliothèque en fonction des critères de l'utilisateur. Elle implémente JavaSearcher            | Création d'un objet stockant les la recherche de l'utilisateur (nom des auteurs, nom des livres, date de parution) | Modification de WritePrefs, le CSV est maintenant créé à partir d'un objet représentant les critères de l'utilisateur |
| Ajout de la fonctionnalité drawBorder pour entourer chaque livre d'une bordure noire  | Développement d'une classe SimpleSearcher qui implémente l'interface JavaSearcher  | Rendre cliquable la tranche du livre pour effectuer une recherche google sur celui-ci                              | Modification du format du fichier CSV (1 ligne par utilisateur)   |
| Développement des boutons « plus » et « moins » pour augmenter ou diminuer le nombre de livres par étagère dans la bibliothèque | WritePrefs : classe permettant la lecture et l'écriture des préférences d'affichage de l'utilisateur dans un fichier CSV                         | Développement du GUI approprié pour SearchData   | Fonction printSVG permettant d'imprimer au format A4  |

# FONCTIONNALITE : Rechercher un livre dans la bibliothèque

Code : Rechercher un livre en fonction des critères de recherche

```
@Override
public HashSet<Book> search(Collection<Book> collection, String filter, Object value) {
    HashSet<Book> books = new HashSet<>();
    if (filter == "author") {
        collection.stream().filter(b -> b.getAuthor().checkAuthorRegex(value.toString())).forEach(books::add);
    }

    else if (filter == "title") {
        collection.stream().filter(b -> b.getTitle().toLowerCase().contains(value.toString().toLowerCase())
            || isSame(b.getTitle(), value.toString())).forEach(books::add);
    } else {
        @SuppressWarnings("unchecked")
        Range<Integer> p = (Range<Integer>) value;
        collection.stream().filter(b -> p.contains(b.getYear())).forEach(books::add);
        collection.stream().map(b -> p.contains(b.getYear())).forEach(System.out::println);
    }
    return books;
}
```

Rechercher un livre :

# Add Book

Search:

Author

Title

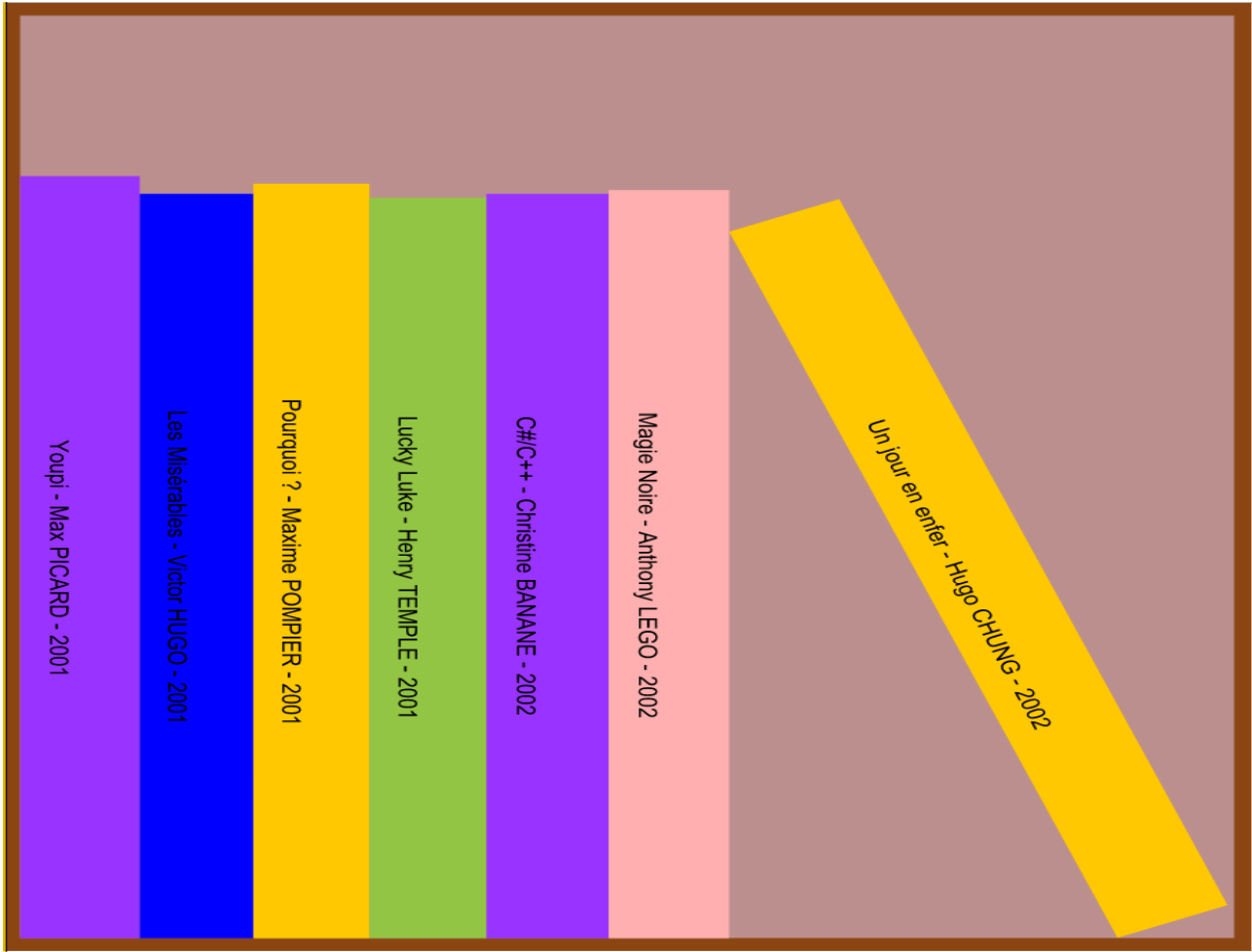
2000

2002

Searching Not limited

Search

Résultat :



# FONCTIONNALITE : Stockage des préférences de l'utilisateur

Code : Parser le fichier CSV contenant les données utilisateurs

```
/**
 *
 * @param fileName the name of the file
 * @return List<String[]> with the value of all
 * @throws URISyntaxException
 * @see <a href=
 *      "https://subscription.packtpub.com/book/big_data_and_business_intelligence/9781787122536/1/ch01lv1sec15/parsing-comma-separated-value-csv-
 */
public static List<SearchPreferences> parseCSVFile(String fileName) throws URISyntaxException {
    CsvParserSettings parserSettings = new CsvParserSettings();
    parserSettings.setLineSeparatorDetectionEnabled(true);
    parserSettings.setHeaderExtractionEnabled(true);
    BeanListProcessor<SearchPreferences> rowProcessor = new BeanListProcessor<>(SearchPreferences.class);
    parserSettings.setProcessor(rowProcessor);
    CsvParser parser = new CsvParser(parserSettings);
    parser.parse(new File(getPathLocation() + fileName));
    return rowProcessor.getBeans();
}
```

Fichier CSV contenant les  
préférences de l'utilisateur :

```
Background_color,Shelves_Color,Books_Color,Position_of_books,Sort_books_by,Books_per_shelf  
LIGHT,LIGHT,DARK,NOT_LEANED,YEAR_MOST_RECENT,19
```

Préférence de  
l'utilisateur sur la GUI :

| <i>Parameters</i>                                    | <i>Choices</i>  |
|--|---|
| <i>Background Color :</i>                            | <input type="radio"/> Auto <input checked="" type="radio"/> Light <input type="radio"/> Dark  |
| <i>Shelves Color :</i>                               | <input type="radio"/> Auto <input checked="" type="radio"/> Light <input type="radio"/> Dark  |
| <i>Books Color :</i>                                 | <input type="radio"/> Auto <input type="radio"/> Light <input checked="" type="radio"/> Dark  |
| <i>Position of books :</i>                           | <input type="radio"/> Leaned <input checked="" type="radio"/> Not leaned  |
| <i>Sort books by :</i>                               | <input type="radio"/> Auto <input type="radio"/> Year <input checked="" type="checkbox"/> Most recent first (sort by year) <input type="radio"/> Author <input type="radio"/> Title |
| <i>Books per shelf:</i>                              | <input type="button" value="Less"/> <input type="text" value="19"/> <input type="button" value="More"/>   |
| <input type="button" value="Reload my library now"/> |   |



# FONCTIONNALITE : Ajouter ou diminuer le nombre de livres par étagère

Code : Bouton permettant de définir le nombre de livres par étagère

```
JButton lessBookPerS = new JButton("Less");
JButton moreBookPerS = new JButton("More");
ActionListener actionListener = new ActionListener() {
    /**
     * For the less and more button we used :
     * https://stackoverflow.com/questions/7300135/how-to-use-an-action-listener-to-check-if-a-certain-button-was-clicked
     */

    @Override
    public void actionPerformed(ActionEvent e) {
        switch (e.getActionCommand()) {
            case "More":
                if (getFieldValue() >= CAPACITY_MAX_PER_SHELF) {
                    JOptionPane.showMessageDialog(optionsJPanel, "Too many books");
                } else {
                    setFieldValue(getFieldValue() + 1);
                }
                break;
            case "Less":
                if (getFieldValue() <= CAPACITY_MIN_PER_SHELF) {
                    JOptionPane.showMessageDialog(optionsJPanel,
                        "Your library can not have less than 5 books because it is not visible");
                } else {
                    setFieldValue(getFieldValue() - 1);
                }
                break;
        }
    }
}
```


Pop up  
d'information  
avertissant  
l'utilisateur qu'on  
ne peut pas avoir  
plus de 20 livres  
par étagère :

*Position of books :* ☒ *Leaned* ☐ *Not leaned*

*Sort books by :* ☒ *Auto* ☐ *Year* ☐ *Most recent first (sort by year)* ☐ *Author* ☐ *Title*

*Books per shelf:*

Message

 Too many books

# FONCTIONNALITE : Recherche d'un livre sur Google en appuyant sur la tranche d'un livre

```
svgCanvas.addLinkActivationListener(new LinkActivationListener() {  
  
    @Override  
    public void linkActivated(LinkActivationEvent e) {  
        try {  
            String urlWithEncodedSpaces = e.getReferencedURI().toString().replaceAll(" ", "%20");  
            Desktop.getDesktop().browse(new URI(urlWithEncodedSpaces).toURL().toURI());  
        } catch (Exception e1) {  
            LOGGER.error("Wrong URL" + e.getReferencedURI().toString());  
            e1.printStackTrace();  
        }  
    }  
});
```

## Bibliothèque générée :

Magie Noire - Anthony LEGO - 2002

Pourquoi ? - Maxime POMPIER - 2001

Infernale - Kevin TIGRE - 1514

Pakistan - Emmanuel LOUP - 2005

Poupi le chien - Manuel PECHE - 2004

Bien coder en JAVA - Christophe CERISE - 2003

C#/C++ - Christine BANANE - 2002

Come on - Citroen ANDRE - 2875

Les animaux - Merlène LEJEUNE - 1996

La Traversée - Elena KOALA - 2018

Youpi - Max PICARD - 2001

Burgers maison - Léo POULET - 2005

Beauté everyday - Jade PARFUM - 2017

Lucky Luke - Henry TEMPLE - 2001

La Magie - Merlin LENCHANTEUR - 1256

La Grande Aventure - Elie EL CHARTOUNI - 2005

I comme Icar - Fanny JAMMES - 2004

Recette pompette - Olympe SUQUET - 2003

Un jour en enfer - Hugo CHUNG - 2002

Les Misérables - Victor HUGO - 2001

## Recherche Google à partir de la tranche du livre :



Les Misérables - Victor HUGO - 2001



Tous

Shopping

Images

Actualités

Vidéos

Plus

Paramètres

Outils

Environ 1 150 000 résultats (0,53 secondes)

### Les Misérables — Wikipédia ✓

[https://fr.wikipedia.org/wiki/Les\\_Misérables](https://fr.wikipedia.org/wiki/Les_Misérables) ✓ ▼

**Les Misérables** est un roman de **Victor Hugo** paru en 1862. Il a donné lieu à de nombreuses .....  
François Cérésa a donné en **2001** une suite controversée aux **Misérables**, avec deux livres intitulés  
Cosette ou le temps des illusions et Marius ou ...

**Genre:** Roman

**Éditeur:** Albert Lacroix et Cie

**Date de parution:** 1862

**Illustrateur:** Émile Bayard

Mini-série, 2000 · Les Misérables (film, 2012) · Les Misérables (film, 1913) · Fantine

### Les Misérables (mini-série, 2000) — Wikipédia ✓

[https://fr.wikipedia.org/wiki/Les\\_Misérables\\_\(mini-série,\\_2000\)](https://fr.wikipedia.org/wiki/Les_Misérables_(mini-série,_2000)) ✓ ▼

... la documentation du modèle. **Les Misérables** est une mini-série en coproduction française-italienne-  
allemande-américaine-espagnole-japonaise-canadienne en 4 épisodes de 90 minutes, réalisée par  
Josée Dayan et écrite par Didier Decoin d'après l'œuvre éponyme de **Victor Hugo** et diffusée ...

Synopsis · Distribution · Lieux de tournage · Commentaires

# Difficultés rencontrées

- ▶ Difficulté à reprendre un projet déjà existant car il faut d'abord lire et comprendre tout le code avant de le modifier,
- ▶ Ayant un emploi du temps très chargé, nous n'avons pas eu assez de temps libre pour travailler sur le projet,
- ▶ Image trop abstraite du rôle du SMPO,
- ▶ Difficulté à se familiariser avec Travis,
- ▶ Certaines personnes n'avaient jamais fait de Java auparavant.

# Axes d'amélioration

- ▶ Classer les livres par genre,
- ▶ Possibilité de créer un compte avec nom d'utilisateur et mot de passe, pour que la bibliothèque soit commune à plusieurs lecteurs :
  - ▶ Possibilité de donner son avis par rapport à un livre et de le noter,
  - ▶ Possibilité de marquer un livre comme lu,
  - ▶ Recommandation de livres similaires (même auteur/genre).
- ▶ Possibilité d'envoyer par email le SVG avec un bouton pour partager sa bibliothèque.

Merci de votre attention !