



(Oracle,  
2023-source ref.[1])

# Projet 4: Débuggez une application Java

Début du projet : Vendredi 4 novembre 2022

Date estimée de la soutenance: **Date à venir**

Présenté par Hilde JACOBI -Formation Développeur Salesforce 2022-2023



# Plan de soutenance

## 1/ Introduction

## 2/ Localiser les problèmes du code d'Alex

**Problème: La liste de symptômes est lue mais le décompte est incorrect**

- ISymptomReader.java Interface créé par Alex  
( L'interface n'a pas les méthodes de class SymptomReader + il n'y a pas de fichier pour la class SymptomReader )
- ReadSymptomDataFromFile.java  
( Class ReadSymptomDataFromFile implement l'interface ISymptomReader )
- AnalyticsCounter.java  
(Class AnalyticsCounter avec la main method )

## 3/ Création du code Java

**3.1/ Solution : creation dans GitHub un repository Fork**

**3.2/ Solution : creation du code from scratch**

- ◆ Mettre symptoms.txt dans un buffer
- ◆ Utiliser method bufferedReader pour lire symptômes dans le Buffer
- ◆ Créer une List of string avec la variable "symptoms"
- ◆ Créer une méthode printFile(List<String> symptoms) pour mettre ligne par ligne la liste de "symptoms" dans FileWriter("result.out")
- ◆ Créer une méthode TreeMap pour avoir les symptômes dans un ordre alphabétique et les décomptes

## 4/Conclusion

# Definition

## Développeur Salesforce

“est spécialisé dans l'utilisation des langages de programmation Lightning et Apex, pour concevoir et implémenter des solutions sur une plateforme Salesforce.”

(OpenClassrooms - source ref. [2])



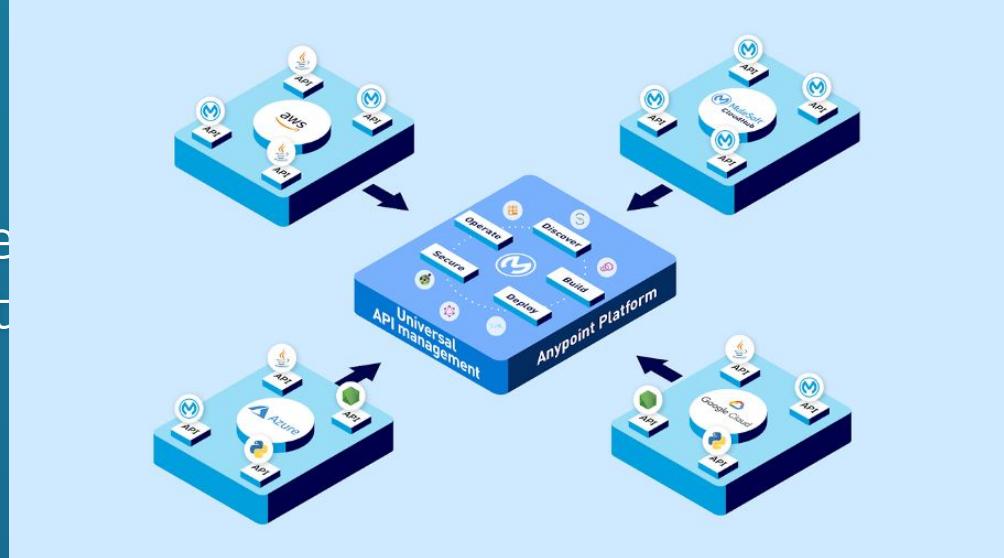
(123rf - source ref. [3])

# Definition

## API

“L'API (Application Programming Interface) est une interface qui connecte des logiciels, des services et des applications aux environnements différents afin qu'ils puissent connecter leurs données..”

(Salesforce - source ref. [4])



(Salesforce- source ref. [4])



OpenClassrooms

Hilde JACOBI - Formation Développeur Salesforce 2022-2023

1

# INTRODUCTION

## Présentation du projet 4



OpenClassrooms

Hilde JACOBI - Formation Développeur Salesforce  
2022-2023

“

L'objectif de ce quatrième  
projet est de modifier le code  
Java créé par Alex en utilisant  
la programmation orientée  
objet.

(OpenClassrooms, 2023-source ref. [5])



OpenClassrooms

Hilde JACOBI - Formation Développeur Salesforce  
2022-2023

2

## Localiser les problèmes du code d'Alex

La liste de symptômes est lue mais le décompte est incorrect

# Explications des problèmes du code d'Alex

**Problème : La liste de symptômes est lue mais le décompte est incorrect.**

- ISymptomReader.java Interface créé par Alex  
( L'interface n'a pas les méthodes de class SymptomReader + il n'y a pas de fichier pour la class SymptomReader )
- ReadSymptomDataFromFile.java  
( Class ReadSymptomDataFromFile implement l'interface ISymptomReader )
- AnalyticsCounter.java  
(Class AnalyticsCounter avec la main method )

**Solution: Crédit à l'origine : Main.java + SymtomReader.java**

*Créer les fichiers : Main.java + SymtomReader.java*

*Modifier le fichier : ISymptomReader.java*

*Mettre symptoms.txt dans un buffer*

*Utiliser method BufferedReader pour lire symptômes dans le Buffer*

*Créer une List of string avec la variable symptoms*

*Créer une méthode printFile(List<String> symptoms) pour mettre ligne par ligne la liste de symptoms dans  
FileWriter("result.out")*

*Créer une méthode TreeMap pour avoir les symptômes dans un ordre alphabétique et les décomptes*



OpenClassrooms

Hilde JACOBI - Formation Développeur Salesforce 2022-2023

# 3

# Création du code Java

3.1/ Solution : creation dans GitHub mon repository Fork+  
importer repository dans Eclipse

Creation dans GitHub

*Project\_DA\_Java\_EN\_Come\_to\_the\_Rescue\_of\_a\_Java\_Application*

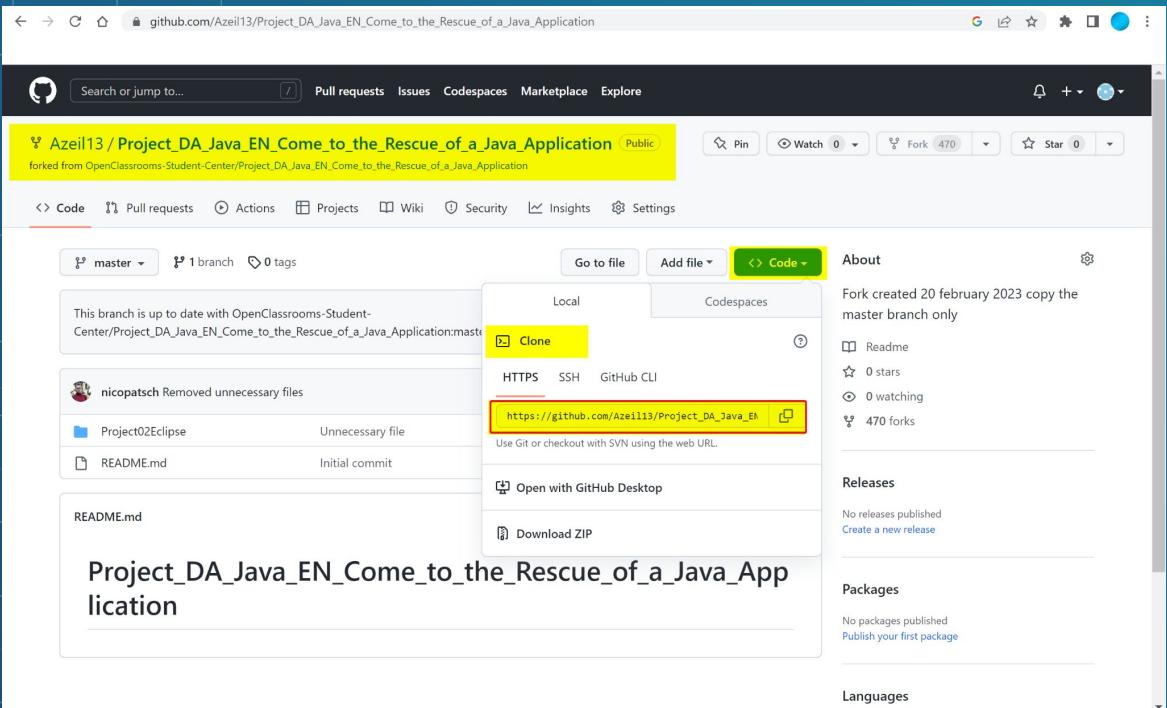
*forked from*

*OpenClassrooms-Student-Center/Project\_DA\_Java\_EN\_Come\_to\_the\_Rescue\_of\_a\_Java\_Application*

- [https://github.com/Azeil13/Project\\_DA\\_Java\\_EN\\_Come\\_to\\_the\\_Rescue\\_of\\_a\\_Java\\_Application](https://github.com/Azeil13/Project_DA_Java_EN_Come_to_the_Rescue_of_a_Java_Application)

# Solution : creation dans GitHub repository Fork+ importer repository dans Eclipse

Azeil13/Project\_DA\_Java\_EN\_Come\_to\_the\_Rescue\_of\_a\_Java\_Application  
forked from OpenClassrooms-Student-Center/Project\_DA\_Java\_EN\_Come\_to\_the\_Rescue\_of\_a\_Java\_Application  
[https://github.com/Azeil13/Project\\_DA\\_Java\\_EN\\_Come\\_to\\_the\\_Rescue\\_of\\_a\\_Java\\_Application](https://github.com/Azeil13/Project_DA_Java_EN_Come_to_the_Rescue_of_a_Java_Application)



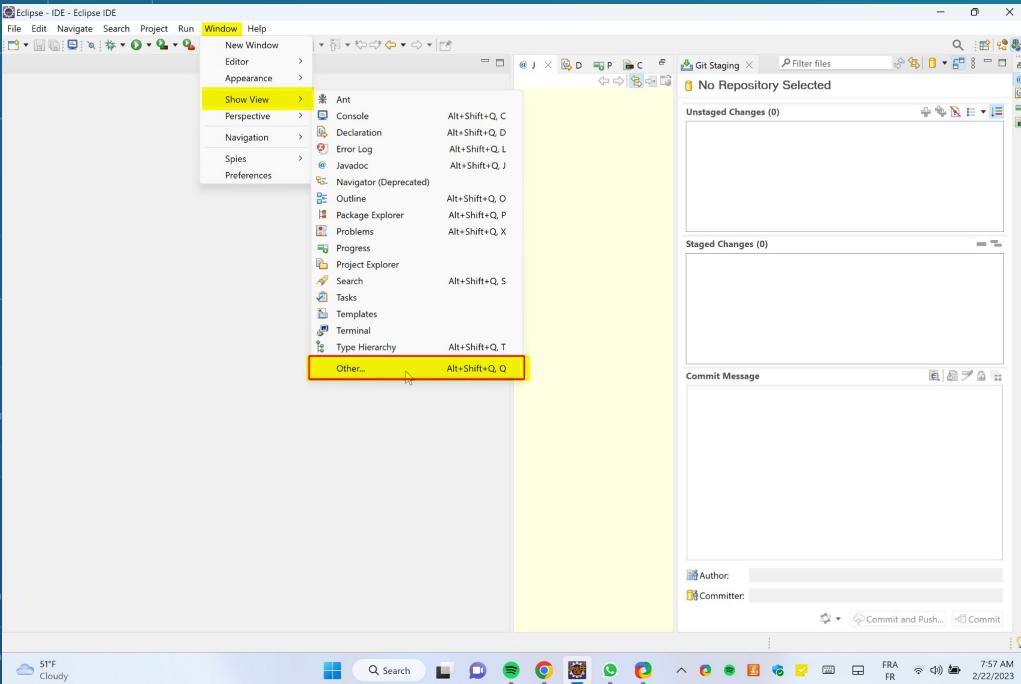
The screenshot shows a GitHub repository page for 'Azeil13/Project\_DA\_Java\_EN\_Come\_to\_the\_Rescue\_of\_a\_Java\_Application'. The 'Code' tab is selected. A 'Clone' modal is open, displaying the URL [https://github.com/Azeil13/Project\\_DA\\_Java\\_EN\\_Come\\_to\\_the\\_Rescue\\_of\\_a\\_Java\\_Application.git](https://github.com/Azeil13/Project_DA_Java_EN_Come_to_the_Rescue_of_a_Java_Application.git), which is highlighted with a red box.

## ◆ Step 1: in Github Copy URL

Étape 1:  
dans Github  
Copier le lien  
[https://github.com/Azeil13/Project\\_DA\\_Java\\_EN\\_Come\\_to\\_the\\_Rescue\\_of\\_a\\_Java\\_Application.git](https://github.com/Azeil13/Project_DA_Java_EN_Come_to_the_Rescue_of_a_Java_Application.git)

# Solution : creation dans GitHub repository Fork+ importer repository dans Eclipse

Azeil13/Project\_DA\_Java\_EN\_Come\_to\_the\_Rescue\_of\_a\_Java\_ApplicationPublic  
forked from OpenClassrooms-Student-Center/Project\_DA\_Java\_EN\_Come\_to\_the\_Rescue\_of\_a\_Java\_Application  
[https://github.com/Azeil13/Project\\_DA\\_Java\\_EN\\_Come\\_to\\_the\\_Rescue\\_of\\_a\\_Java\\_Application](https://github.com/Azeil13/Project_DA_Java_EN_Come_to_the_Rescue_of_a_Java_Application)

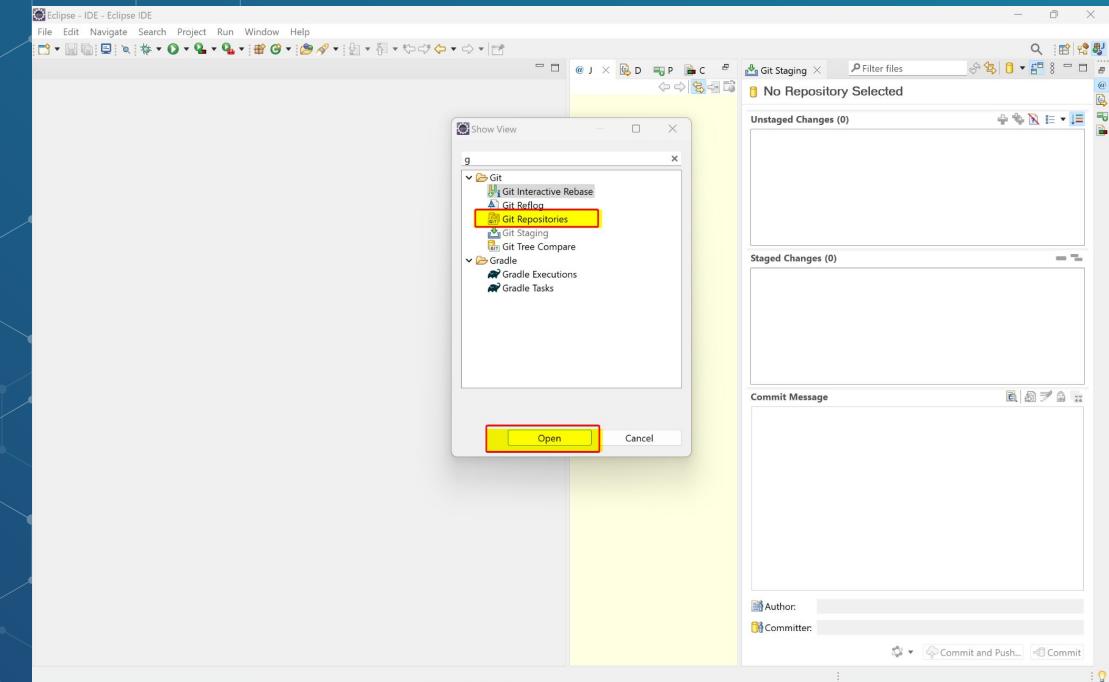


❖ Step 2: Open Eclipse and go to Window > Show views > Other > Git > Git Repositories for making git repositories visible in eclipse as shown in the image.

Étape 2 : Ouvrez Eclipse et accédez à Fenêtre > Afficher les vues > Autre > Git > Repositories Git pour rendre les repositories git visibles dans Eclipse, comme indiqué dans l'image.

# Solution : creation dans GitHub repository Fork+ importer repository dans Eclipse

Azeil13/Project\_DA\_Java\_EN\_Come\_to\_the\_Rescue\_of\_a\_Java\_ApplicationPublic  
forked from OpenClassrooms-Student-Center/Project\_DA\_Java\_EN\_Come\_to\_the\_Rescue\_of\_a\_Java\_Application  
[https://github.com/Azeil13/Project\\_DA\\_Java\\_EN\\_Come\\_to\\_the\\_Rescue\\_of\\_a\\_Java\\_Application](https://github.com/Azeil13/Project_DA_Java_EN_Come_to_the_Rescue_of_a_Java_Application)



❖ Select Git Repositories + click open

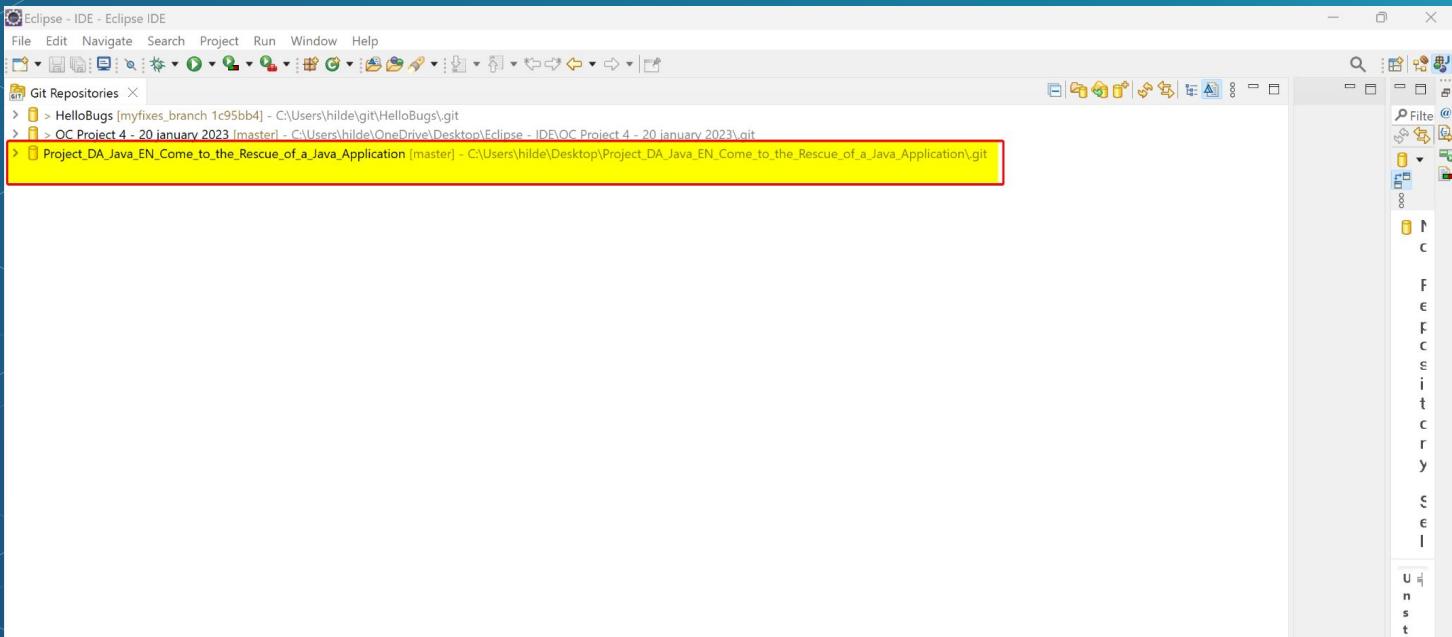
Sélectionnez Git Repositories + cliquez sur Ouvrir

# Solution : creation dans GitHub repository Fork+ importer repository dans Eclipse

Azeil13/Project\_DA\_Java\_EN\_Come\_to\_the\_Rescue\_of\_a\_Java\_ApplicationPublic

forked from OpenClassrooms-Student-Center/Project\_DA\_Java\_EN\_Come\_to\_the\_Rescue\_of\_a\_Java\_Application

[https://github.com/Azeil13/Project\\_DA\\_Java\\_EN\\_Come\\_to\\_the\\_Rescue\\_of\\_a\\_Java\\_Application](https://github.com/Azeil13/Project_DA_Java_EN_Come_to_the_Rescue_of_a_Java_Application)

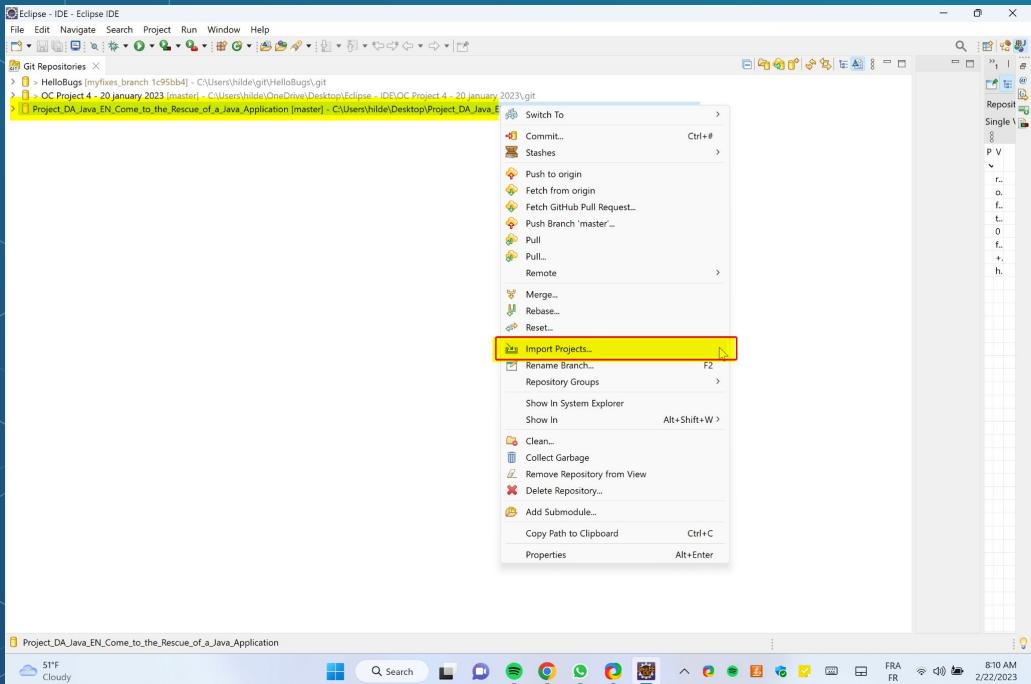


# Solution : creation dans GitHub repository Fork+ importer repository dans Eclipse

Azeil13/Project\_DA\_Java\_EN\_Come\_to\_the\_Rescue\_of\_a\_Java\_ApplicationPublic

forked from OpenClassrooms-Student-Center/Project\_DA\_Java\_EN\_Come\_to\_the\_Rescue\_of\_a\_Java\_Application

[https://github.com/Azeil13/Project\\_DA\\_Java\\_EN\\_Come\\_to\\_the\\_Rescue\\_of\\_a\\_Java\\_Application](https://github.com/Azeil13/Project_DA_Java_EN_Come_to_the_Rescue_of_a_Java_Application)

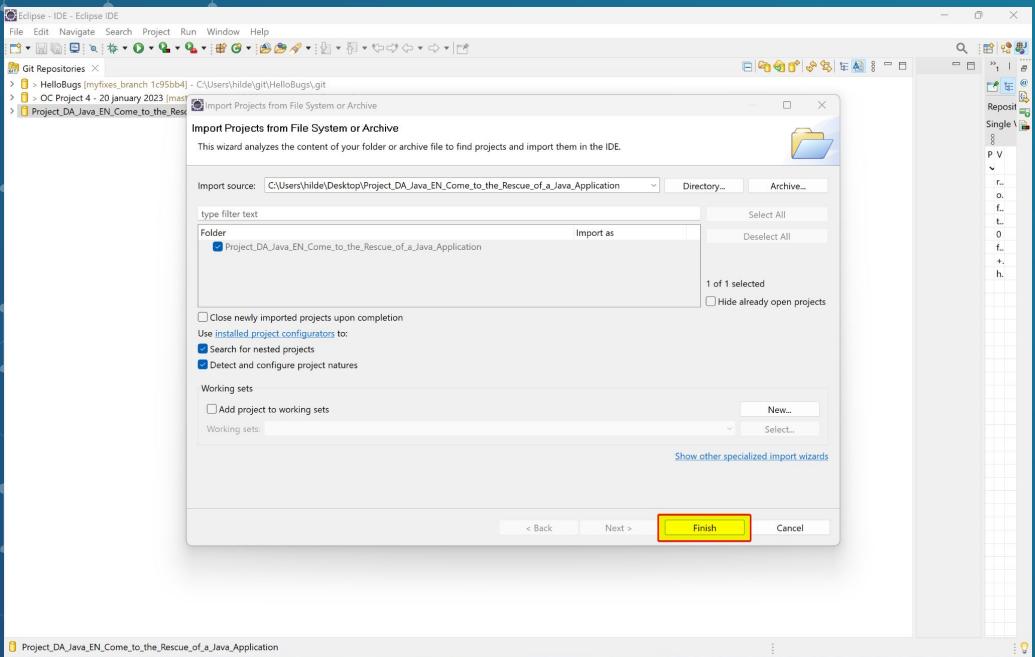


❖ Import project into IDE  
(Integrated Development Environment)  
Eclipse

Importer le projet dans IDE  
(environnement de développement intégré)  
Eclipse

# Solution : creation dans GitHub repository Fork+ importer repository dans Eclipse

Azeil13/Project\_DA\_Java\_EN\_Come\_to\_the\_Rescue\_of\_a\_Java\_Application  
forked from OpenClassrooms-Student-Center/Project\_DA\_Java\_EN\_Come\_to\_the\_Rescue\_of\_a\_Java\_Application  
[https://github.com/Azeil13/Project\\_DA\\_Java\\_EN\\_Come\\_to\\_the\\_Rescue\\_of\\_a\\_Java\\_Application](https://github.com/Azeil13/Project_DA_Java_EN_Come_to_the_Rescue_of_a_Java_Application)

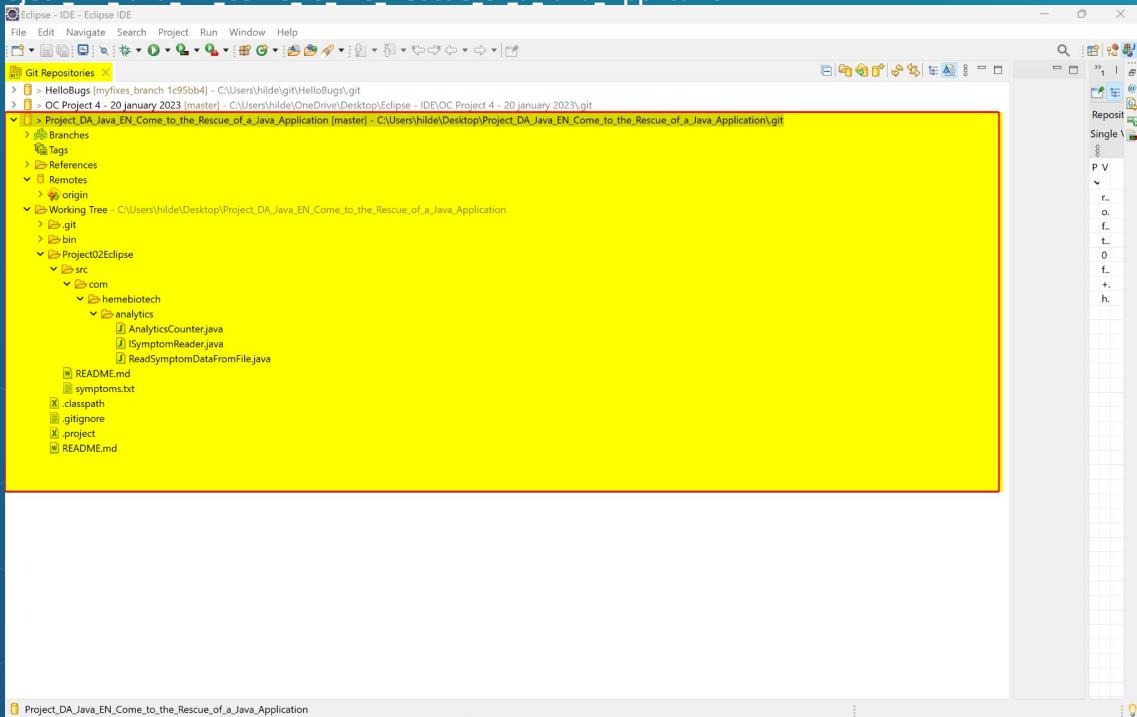


 Click finish

Cliquez sur terminer

# Solution : creation dans GitHub repository Fork+ importer repository dans Eclipse

Azeil13/Project\_DA\_Java\_EN\_Come\_to\_the\_Rescue\_of\_a\_Java\_Application  
Public  
forked from OpenClassrooms-Student-Center/Project\_DA\_Java\_EN\_Come\_to\_the\_Rescue\_of\_a\_Java\_Application  
[https://github.com/Azeil13/Project\\_DA\\_Java\\_EN\\_Come\\_to\\_the\\_Rescue\\_of\\_a\\_Java\\_Application](https://github.com/Azeil13/Project_DA_Java_EN_Come_to_the_Rescue_of_a_Java_Application)





OpenClassrooms

Hilde JACOBI - Formation Développeur Salesforce 2022-2023

# 3

# Création du code Java

## 3.2/ Solution: creation du code from scratch

Creation du code from scratch *créer public class Main*

- Mettre symptoms.txt dans un buffer
- Utiliser method bufferedReader pour lire symptômes dans le Buffer
- Créer une List of string avec la variable symptoms
- Créer une méthode printFile(List<String> symptoms) pour mettre ligne par ligne la liste de symptoms dans FileWriter("result.out")
- Créer une méthode TreeMap pour avoir les symptômes dans un ordre alphabétique et les décomptes

# Explications du code mis en place

- ❖ **Créer Interface ISymptomReader**

```
public interface ISymptomReader {  
    public List<String> readFile(String filePath);  
    public void printFile(Map<String, Integer> sortedSymptoms, String outputPath) throws IOException;  
}
```

- ❖ **Créer public class Main qui est la classe principale du fichier Main.java**

```
public class Main {  
}
```

- ❖ **Créer class SymptomReader**

```
public class SymptomReader implements ISymptomReader {  
}
```

- ❖ **Créer javadoc**

```
/** Documentation */  
Commentaires sur la documentation pour générer la documentation API (application programming interfaces)
```

- ❖ **Commit code**



## interface ISymptomReader

```
public interface ISymptomReader {  
    public List<String> readFile(String filePath);  
    public void printFile(Map<String, Integer> sortedSymptoms, String outputPath) throws IOException;  
}
```

La classe SymptomReader implémente l'interface ISymptomReader car elle définit les méthodes de l'interface.

En java, on déclare la classe implémente l'interface avec le mot clé **implements**.

Les méthodes d'interface n'ont pas de corps - le corps est fourni par la classe "implémenter" qui est SymptomReader

Utilisation de l'interface pour assurer la sécurité - masquer certains détails et afficher uniquement les détails importants.

class SymptomReader

```
public class SymptomReader  
implements ISymptomReader{  
}
```

class Main

```
public class Main {  
}
```

# Explications du code mis en place

- ❖ Créer public class Main qui est la classe principale du fichier Main.java

```
package com.hemebiotech.analytics;

import java.util.List;
import java.util.Map;
import java.util.TreeMap;

/**
 * This is the main class Main
 *
 * @author hilde Jacobi
 * @version 1 Build February 23, 2023.
 */
```

```

public class Main {
    // Inside main, call the methods on the br object
    /**
     * 
     */
    * @param args
    */
    public static void main(String[ ] args) { //Java main() method
        ISymptomReader reader = new SymptomReader(); // Create a br object
        List<String> symptoms = reader.readFile("C:\\\\Users\\\\hilde\\\\OneDrive\\\\Desktop\\\\OpenClassroom-
Project4\\\\Project\\\\Project_DA\\\\Java_EN_Come_to_the_Rescue_of_a_Java_Application-master\\\\Project02Eclipse\\\\symptoms.txt"); // Call the readFile(BufferedReader reader) method
        // method openFile of the class is hidden by the use of the interface
        // read.openFile("XXXX"); -> does not compile and it is done on purpose

        System.out.println(symptoms.size());
        Map<String, Integer> sortedSymptoms = new TreeMap<String, Integer>();
        for(String symptom : symptoms){
            if(!sortedSymptoms.containsKey(symptom)){
                // I initialize my counter for the symptom if it is the first time that I see it
                sortedSymptoms.put(symptom, 0);
            }

            // I am looking for the number of symptoms that I have already found
            int nbSymptomFound = sortedSymptoms.get(symptom);
            // I add 1 to the number of symptoms
            sortedSymptoms.put(symptom, nbSymptomFound+1);

        }
        try{
            reader.printFile(sortedSymptoms, "C:\\\\Users\\\\hilde\\\\OneDrive\\\\Desktop\\\\OpenClassroom- Project
4\\\\Project\\\\Project_DA\\\\Java_EN_Come_to_the_Rescue_of_a_Java_Application-master\\\\Project02Eclipse\\\\symptomsSorted.txt");
        } catch (Exception e){
            e.printStackTrace();
        }
    }
}

```

## Créer public class Main qui est la classe principale du fichier Main.java

TreeMap method pour avoir les symptômes dans un ordre alphabétique et les décomptes

Initialise mon compteur pour le symptôme si c'est la première fois que je le vois

Je cherche le nombre de symptômes que j'ai déjà trouvé + J'ajoute 1 au nombre de symptômes

# Explications du code mis en place

- ❖ Créer Interface ISymptomReader qui est l'interface du fichier ISymptomReader.java

```
/**  
 * This is the Interface ISymptomReader  
 * @author hilde Jacobi  
 * @version 1 Build February 23, 2023.  
 */  
  
public interface ISymptomReader {  
    public List<String> readFile(String filePath);  
    public void printFile(Map<String, Integer> sortedSymptoms, String outputPath) throws IOException;  
}
```

# Explications du code mis en place

## ❖ Créer class SymptomReader qui est la classe du fichier SymptomReader.java.

```
package com.hemebiotech.analytics;  
import java.io.BufferedReader;  
import java.io.FileNotFoundException;  
import java.io.FileWriter;  
import java.io.IOException;  
import java.util.ArrayList;  
import java.util.List;  
import java.util.Map;  
import java.util.TreeMap;  
import java.io.FileReader;  
  
/**  
 * This is the class SymptomsReader which implements Interface ISymtomsReader  
 *  
 * @author hilde Jacobi  
 * @version 1 Build February 23, 2023.  
 */
```

```
public class SymptomReader implements ISymptomReader{  
  
    //Create a openFile(String filename) Method  
    /**  
     * open a file and return BufferedReader  
     * @param filename it is the path of the file to open in the buffer  
     * @return return a BufferedReader of all the lines of the file  
     */  
    public BufferedReader openFile(String filename) {  
        System.out.println("Open the file:" + filename);  
        BufferedReader reader;  
        try {  
            reader = new BufferedReader (new FileReader(filename));  
            return reader;  
        } catch (FileNotFoundException e) {  
            // TODO Auto-generated catch block  
            e.printStackTrace();  
        }  
        return null;  
    }  
}
```

**Créer class SymptomReader qui est la classe du fichier SymptomReader.java**

Mettre symptoms.txt dans un buffer + Utiliser method bufferReader pour lire symptômes dans le Buffer

```
//Create a readfile(BufferedReader reader) Method
/**
 * 
 * @param reader create read File Method
 * @return list of string File of symptoms
 */
public List<String> readfile(String filePath) { //method BufferedReader native of Java
    BufferedReader reader =openFile(filePath);

    List<String> symptoms = new ArrayList<>(); // create a List of string with variable symptoms

    try {
        while (reader.ready()) {
            symptoms.add(reader.readLine()); // add in the List what it is reading each sentence
        }
    } catch (IOException e) {
        // TODO Auto-generated catch block
        e.printStackTrace();
    }
    return symptoms;
}
```

## Créer class SymptomReader qui est la classe du fichier SymptomReader.java

Créer une List of string avec la variable symptoms + ajouter dans la List chaque phrase qui est lue et retourne les symptômes

```
/*
 * Write printFile method
 * and create the file result.out which contain the line read line by line of the symptoms.txt
 */

/**
 * @param Map<String,Integer> sortedSymptoms : map of the symptoms
 * @throws IOException
 */
public void printFile(Map<String,Integer> sortedSymptoms ,String outputPath) throws IOException {      //method printFile no return
    FileWriter resultsDocument = new FileWriter(outputPath); // create a new File

    for (String symptomKey: sortedSymptoms.keySet()) //create a new variable loop for line by line symptom
        try {
            System.out.println(symptomKey+":"+sortedSymptoms.get(symptomKey));
            resultsDocument.write(symptomKey+":"+sortedSymptoms.get(symptomKey)); // put in the file the newline
        } catch (IOException e) {
            e.printStackTrace();
        }
    resultsDocument.close();
}
```

## Créer class SymptomReader qui est la classe du fichier SymptomReader.java

Créer une méthode printFile (*Map<String,Integer> sortedSymptoms ,String outputPath*)  
pour mettre ligne par ligne la liste de symptoms dans *FileWriter("result.out")*



# Exécution du code java-cliquer sur le bouton Run

```
Open the file:C:\Users\hilde\OneDrive\Desktop\OpenClassroom- Project  
4\Project\Project_DA_Java_EN_Come_to_the_Rescue_of_a_Java_Application-master\Project02Eclipse\symptoms.txt  
100  
anxiety:5  
arrhythmias:3  
blindness:1  
blurred vision:5  
constricted pupils:3  
cough:6  
dilated pupils:4  
dizziness:5  
dry mouth:8  
fever:7  
headache:3  
high blood pressure:10  
inflammation:7  
insomnia:4  
low blood pressure:4  
nausea:5  
rapid heart rate:1  
rash:4  
shortness of breath:3  
stiff neck:4  
stomach pain:3  
tremor:4  
water retention:1
```



Eclipse - IDE - Project\_DA\_Java\_EN\_Come\_to\_the\_Rescue\_of\_a\_Java\_Application/Project02Eclipse/src/com/hemebiotech/analytics/SymptomReader.java - Eclipse IDE

File Edit Source Refactor Navigate Search Project Run Window Help

Project Explorer × ISymptomReader.java Main.java SymptomReader.java

cliquer sur Run

```
1 package com.hemebiotech.analytics;
2
3@ import java.io.BufferedReader;
4
5@ /**
6 * This is the class SymptomsReader which implements Interface ISymtomsReader
7 *
8 * @author hilde Jacobi
9 * @version 1 Build February 23, 2023.
10 */
11
12 public class SymptomReader implements ISymptomReader{
13
14     //Create a openFile(String filename) Method
15     /**
16      * open a file and return BufferReader
17      * @param filename it is the path of the file to open in the buffer
18      * @return return a BufferReader of all the lines of the file
19     */
20     public BufferedReader openfile(String filename) {
21         System.out.println("Open the file:"+ filename);
22         BufferedReader reader;
23         try {
24             reader = new BufferedReader (new FileReader(filename));
25             return reader;
26         } catch (FileNotFoundException e) {
27             // TODO Auto-generated catch block
28             e.printStackTrace();
29         }
30         return null;
31     }
32
33     //Create a readFile(BufferedReader reader) Method
34     /**
35      * @param reader create read File Method
36      * @return list of string File of symptoms
37     */
38     public List<String> readFile(String filePath) { //method BufferedReader native of Java
39         BufferedReader reader =openfile(filePath);
40
41         List<String> symptoms = new ArrayList<String>(); // create a List of strings with variable symptoms
42
43         try {
44             String line;
45             while((line = reader.readLine()) != null) {
46                 symptoms.add(line);
47             }
48         } catch (IOException e) {
49             // TODO Auto-generated catch block
50             e.printStackTrace();
51         }
52
53         return symptoms;
54     }
55 }
```

SymptomReader.java - Project\_DA\_Java\_EN\_Come\_to\_the\_Rescue\_of\_a\_Java\_Application/Project02Eclipse/src/com/hemebiotech/analytics



Eclipse - IDE - Project\_DA\_Java\_EN\_Come\_to\_the\_Rescue\_of\_a\_Java\_Application/Project02Eclipse/src/com/hemebiotech/analytics/SymptomReader.java - Eclipse IDE

File Edit Source Refactor Navigate Search Project Run Window Help

Project Explorer Sym... 2

@ Javadoc Declaration Console Progress Coverage

<terminated> Main (2) [Java Application] C:\Program Files\Java\jdk-19\bin\javaw.exe (Feb 23, 2023, 3:28:00 PM – 3:28:00 PM) [pid: 26652]

Open the file:C:\Users\hilde\OneDrive\Desktop\OpenClassroom- Project 4\Project\Project\_DA\_Java\_EN\_Come\_to\_the\_Rescue\_o...

100

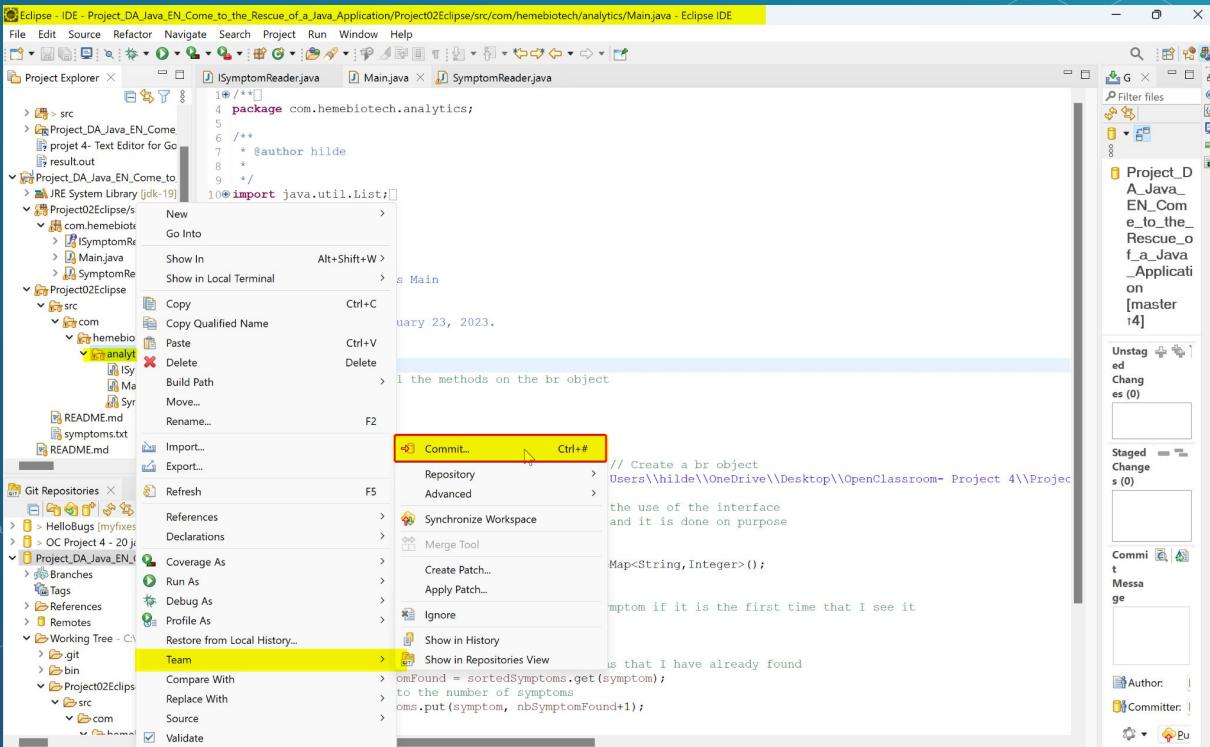
anxiety:5  
arrhythmias:3  
blindness:1  
blurred vision:5  
constricted pupils:3  
cough:6  
dilated pupils:4  
dizziness:5  
dry mouth:8  
fever:7  
headache:3  
high blood pressure:10  
inflammation:7  
insomnia:4  
low blood pressure:4  
nausea:5  
rapid heart rate:1  
rash:4  
shortness of breath:3  
stiff neck:4  
stomach pain:3  
tremor:4  
water retention:1

README.md  
result.out  
symptoms.txt  
README.md

Les symptômes dans l'ordre alphabétique et les décomptes



# Dans Eclipse Commit Code





# Dans GitHub Code commit par Hilde JACOBI

[https://github.com/Azeil13/Project\\_DA\\_Java\\_EN\\_Come\\_to\\_the\\_Rescue\\_of\\_a\\_Java\\_Application/pulse](https://github.com/Azeil13/Project_DA_Java_EN_Come_to_the_Rescue_of_a_Java_Application/pulse)

The screenshot shows a GitHub Pulse page for the repository `Azeil13 / Project_DA_Java_EN_Come_to_the_Rescue_of_a_Java_Application`. The page displays activity from February 16, 2023, to February 23, 2023. Key statistics include:

- 0 Active pull requests
- 0 Active issues
- 0 Merged pull requests
- 0 Open pull requests
- 0 Closed issues
- 0 New issues

A callout box highlights the following commit information:  
Excluding merges, 1 author has pushed 2 commits to master and 2 commits to all branches. On master, 4 files have changed and there have been 24 additions and 0 deletions.

The page also features a bar chart showing file changes across different branches.



OpenClassrooms

Hilde JACOBI - Formation Développeur Salesforce  
2022-2023

# 4

# Conclusion



# CONCLUSION

Java est un langage de programmation pour coder des applications utiles au quotidien (smartphones, électroménagers: lave-linges, ...)  
Java est utilisé pour sa multiplateforme permettant au codeur d'écrire son code java une seule fois sans avoir à le réécrire pour l'adapter aux applications PC, Mac, Linux, ...

Java est une programmation orientée objet qui facilite un travail de coopération grâce à la bibliothèque java des fonctions (package/class/method...)

Que réserve l'avenir de Java ? Quelles sont les nouveautés qui pourront permettre à Java de rester #1 dans le futur ?



OpenClassrooms

Hilde JACOBI - Formation Développeur Salesforce 2022-2023

# 5

# REFERENCES



- [1] Oracle, 2023,Produits et technologies Java,La plate-forme de développement innovante la plus populaire au monde , image, date de consultation 15 février 2023, <<https://www.oracle.com/fr/java/>>
- [2] OpenClassrooms 2022, Développeur Salesforce, date de consultation 15 février 2023, <<https://openclassrooms.com/fr/paths/512-developpeur-salesforce> >
- [3] Banque d'images ,Automatisation ,123rf, image, date de consultation 15 février 2023 , <[https://fr.123rf.com/photo\\_45130232\\_automatisation.html?vti=lvlucte2068c0g6e9w-1-9](https://fr.123rf.com/photo_45130232_automatisation.html?vti=lvlucte2068c0g6e9w-1-9)>
- [4] Salesforce 2023, Qu'est-ce qu'un API : définition, enjeux,date de consultation 15 février 2023 <<https://www.salesforce.com/fr/resources/articles/definition-api/>>
- [5] OpenClassrooms 2023,Débuggez une application Java,date de consultation 15 février 2023 <<https://openclassrooms.com/fr/paths/512/projects/610/assignment>>