

UNIVERSITÉ MOHAMMED V
Faculté des Sciences Rabat



Université Mohammed V
Faculté des Sciences
Rabat

**Master Ingénierie de Données et
Développement Logiciel**

Rapport fin de module JAVA

***Professeur* : Mohammed El Haziti**

Réalisé par :

Hala Bahida

Wiam Zellou

Fatima Bouya

Safae Zellou

Fadwa Saoiabi

Année universitaire 2020-2021

Introduction :

Java est un langage de programmation à usage général, évolué et orienté objet dont la syntaxe est proche du C. Ses caractéristiques ainsi que la richesse de son écosystème et de sa communauté lui ont permis d'être très largement utilisé pour le développement d'applications de types très disparates.

Java est notamment largement utilisé pour le développement d'applications d'entreprises et mobiles.

Il existe 2 types de programmes avec la version standard de Java : les applets et les applications.

Dans ce projet on va s'intéresser aux applications

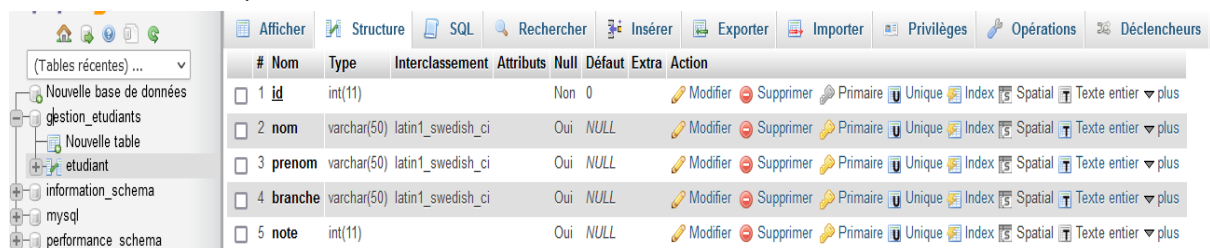
Une application autonome (stand alone program) est une application qui s'exécute sous le contrôle direct du système d'exploitation.

1ème application : Gestion des étudiants

Description

C'est une application qui permet d'ajouter, modifier, supprimer et rechercher un étudiant.

1ère étape: Créer une base de donnée gestion_etudiants qui contient une table etudiant ayant la structure ci-dessous.



#	Nom	Type	Interclassement	Attributs	Null	Défaut	Extra	Action
1	id	int(11)			Non	0		Modifier Supprimer Primaire Unique Index Spatial Texte entier plus
2	nom	varchar(50) latin1_swedish_ci			Oui	NULL		Modifier Supprimer Primaire Unique Index Spatial Texte entier plus
3	prenom	varchar(50) latin1_swedish_ci			Oui	NULL		Modifier Supprimer Primaire Unique Index Spatial Texte entier plus
4	branche	varchar(50) latin1_swedish_ci			Oui	NULL		Modifier Supprimer Primaire Unique Index Spatial Texte entier plus
5	note	int(11)			Oui	NULL		Modifier Supprimer Primaire Unique Index Spatial Texte entier plus

2ème étape: Établir une connexion avec notre base de donnée MySQL

JDBC est l'acronyme de Java DataBase Connectivity et désigne une API pour permettre un accès aux bases de données avec Java.

Les outils nécessaires pour utiliser JDBC :

Les classes de JDBC version 1.0 sont regroupées dans le package java.sql et sont incluses dans le JDK à partir de sa version 1.1. La version 2.0 de cette API est incluse dans la version 1.2 du JDK.

Pour pouvoir utiliser JDBC, il faut un pilote qui est spécifique à la base de données à laquelle on veut accéder. Avec le JDK, Sun fournit un pilote qui permet l'accès aux bases de données via ODBC.

Ce pilote permet de réaliser l'indépendance de JDBC vis à vis des bases de données.

Script de connexion :

```
Source History
1 import java.sql.*;
2
3 public class Connector {
4     Connection con;
5     public Connector(){
6         try{
7             Class.forName("com.mysql.jdbc.Driver");
8         }catch(ClassNotFoundException e){
9             System.err.println(e);
10        }
11        //pour afficher l'erreur
12    }
13    try{
14        con=DriverManager.getConnection("jdbc:mysql://localhost:3306/gestion_etudiants","root","");
15    }catch(SQLException e){System.err.println(e);}
16    }
17    Connection obtenirconnexion(){return con;}
18 }
19
```

3ème étape: Réaliser l'interface graphique de notre application et associer à chaque composant la fonctionnalité souhaitée.

Qu'est-ce qu'un JFrame?

JFrame est une classe qui se trouve dans le package `javax.swing` qui hérite de `java.awt.frame`, il ajoute la prise en charge de l'architecture des composants SWING. Il s'agit d'une fenêtre de niveau supérieur, avec une bordure et une barre de titre. La classe JFrame possède de nombreuses méthodes qui peuvent être utilisées pour la personnaliser.

Voici à quoi ressemble l'interface de notre application:

Fonctionnement de notre application :

Ajouter un étudiant :

Pour ajouter un étudiant il faut fournir les informations suivantes :

- id
- Nom
- Prénom
- Branche
- Note

gestion des notes

id :



Nom :



Prenom:


Branche :

note:


id	nom	prenom	branche	note
1	saouiabi	fadwa	INFO	16
2	wiam	wiam	INFO	16
3	zellou	safae	INFO	16
4	bahida	hala	INFO	16

 **Ajouter**  **actualiser**

 **modifier**  **Supprimer**

 **recherche**

Message

 l'etudiant est bien ajouter

OK

id	nom	prenom	branche	note
3	zellou	safae	INFO	16
4	bahida	hala	INFO	16
5	bouya	fatima	INFO	16
6	zellou	wiam	INFO	16

Supprimer un étudiant :

Pour supprimer un étudiant il suffit de fournir son id et puis cliquer sur supprimer.

gestion des notes

id :

Nom :

Prenom:

Branche :

note:

id	nom	prenom	branche	note
3	zellou	safae	INFO	16
4	bahida	hala	INFO	16
5	bouya	fatima	INFO	16
6	zellou	wiam	INFO	16

supprimer etudiant

attention vous avez supprimer un etudiant,est ce que tu et sure?

id	nom	prenom	branche	note
2	wiam	wiam	INFO	16
3	zellou	safae	INFO	16
4	bahida	hala	INFO	16
5	bouya	fatima	INFO	16

Modifier un étudiant :

Pour modifier un étudiant il faut fournir son id et puis les nouveaux paramètres.

gestion des notes

id :

Nom :

Prenom:

Branche :

note:

id	nom	prenom	branche	note
1	saoiabi	fadwa	INFO	16
2	zellou	zellou	INFO	17
3	zellou	safae	INFO	16
4	bahida	hala	INFO	16

modification

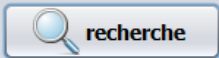
confirmer la modification

id	nom	prenom	branche	note
1	saoiabi	fadwa	INFO	16
2	zellou	wiam	INFO	16
3	zellou	safae	INFO	16
4	bahida	hala	INFO	16

Rechercher un étudiant

Pour rechercher un étudiant il suffit de fournir son id et puis cliquer sur rechercher.

id	nom	prenom	branche	note
4	bahida	hala	INFO	16



Partie d'authentification :

Concernant la partie d'authentification, l'utilisateur doit d'abord entrer son ID ainsi que son mot de passe pour pouvoir se connecter. L'interface d'authentification ressemble à ceci :

Entrez votre ID :

Mot de passe :

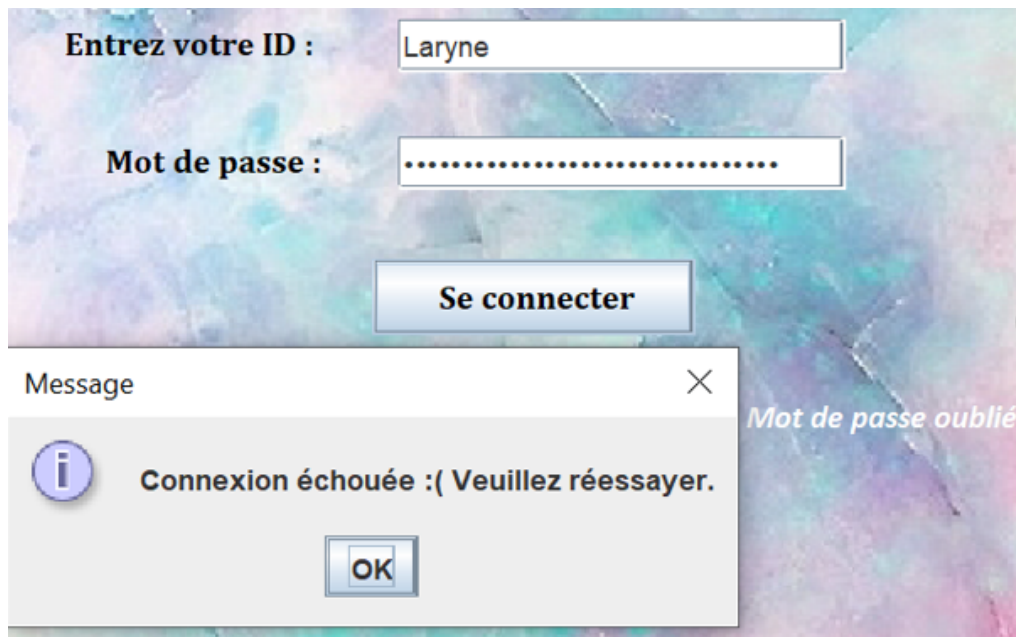
Se connecter

Mot de passe oublié

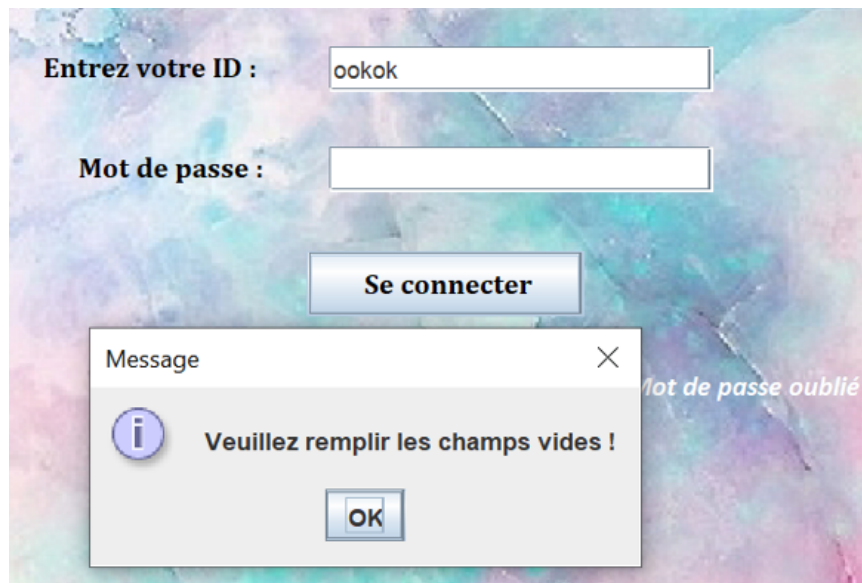
Une fois les informations fournies sont correctes, un message de bienvenue s'affiche :



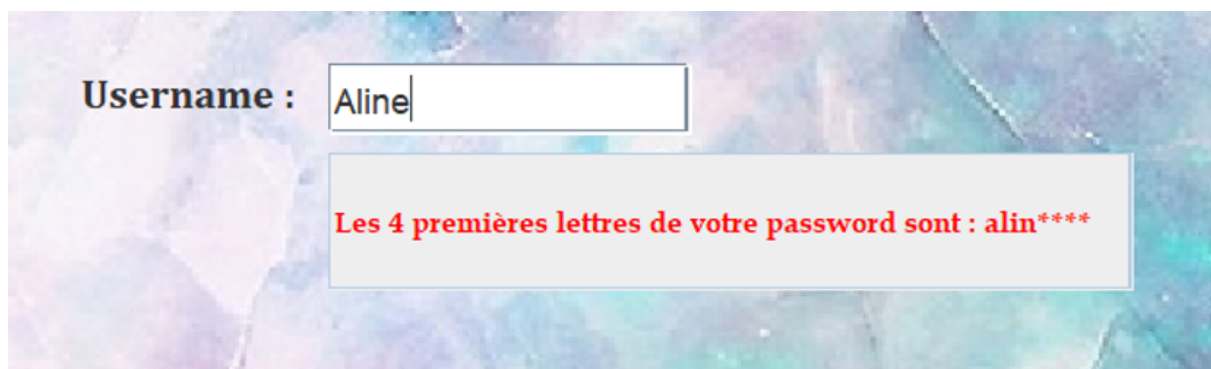
Sinon un message d'erreur fait son apparition :



Si jamais l'utilisateur oublie de fournir certaines informations, un petit message apparaît lui mentionnant comme quoi il doit remplir les champs vides :



Dans le cas d'oubli du mot de passe, ce dernier doit cliquer sur le bouton « Mot de passe oublié » et il sera dirigé par la suite vers une nouvelle interface histoire qu'il puisse récupérer son mot de passe :



Il doit alors entrer son username, et on lui fournira les 4 premières lettres de son password. 😊


```

public static class Thread2 implements Runnable {

    @Override
    public void run() {
        int k= (int)((nbr/4)+1);
        int m=(int)(nbr/2);
        for(int i = k; i<= m;){
            int premier = 1;
            for(int loop = 2; loop <=i; loop++) {
                if((i % loop) == 0 && loop!=i) {
                    premier = 0;
                }
            }
            if (premier != 0){
                System.out.println(i+" est un nombre premier " + Thread.currentThread().getName());
                i++;
                count++;
            }
            else
                i ++;
        }
    }
}

public static class Thread3 implements Runnable{

    @Override
    public void run() {
        int k= (int)((nbr/4)*2 +1);
        int m=(nbr/4)*3;
        for(int i = k; i<= m;){
            int premier = 1;
            for(int loop = 2; loop <=i; loop++) {
                if((i % loop) == 0 && loop!=i) {
                    premier = 0;
                }
            }
            if (premier != 0){
                System.out.println(i+" est un nombre premier " + Thread.currentThread().getName());
            }
        }
    }
}

public static class Thread4 implements Runnable {

    @Override
    public void run() {
        int k= ((nbr/4)*3 +1);
        int m=nbr;
        for(int i = k; i<= m;){
            int premier = 1;
            for(int loop = 2; loop <=i; loop++) {
                if((i % loop) == 0 && loop!=i) {
                    premier = 0;
                }
            }
            if (premier != 0){
                System.out.println(i+" est un nombre premier " + Thread.currentThread().getName());
                i++;
                count++;
            }
            else
                i ++;
        }
    }
}

```

Fonction main() :

```
public static void main(String[] args) throws Exception {
    long t1 = System.currentTimeMillis();
    nbr=100;
    Thread m1 = new Thread(new Thread1(), "m1");
    Thread m2 = new Thread(new Thread2(), "m2");
    Thread m3 = new Thread(new Thread3(), "m3");
    Thread m4 = new Thread(new Thread4(), "m4");
    m1.start();
    m2.start();
    m3.start();
    m4.start();
    m1.join();
    m2.join();
    m3.join();
    m4.join();

    long t2 = System.currentTimeMillis();
    System.out.println("the time is "+(t2-t1));
    System.out.println("le nombre du nombres premiers est "+ count);
}
```

Démarrage de notre programme:

```
Console
<terminated> NbrPremier2 [Java Application] C:\Program Files\Java\jdk-11.0.11\bin\javaw.exe
53 est un nombre premier m3
59 est un nombre premier m3
61 est un nombre premier m3
67 est un nombre premier m3
71 est un nombre premier m3
73 est un nombre premier m3
79 est un nombre premier m4
83 est un nombre premier m4
89 est un nombre premier m4
97 est un nombre premier m4
2 est un nombre premier m1
3 est un nombre premier m1
5 est un nombre premier m1
7 est un nombre premier m1
11 est un nombre premier m1
13 est un nombre premier m1
17 est un nombre premier m1
19 est un nombre premier m1
23 est un nombre premier m1
29 est un nombre premier m2
31 est un nombre premier m2
37 est un nombre premier m2
41 est un nombre premier m2
43 est un nombre premier m2
47 est un nombre premier m2
the time is 16
le nombre des nombres premiers est 25
```

2ème application :

Multiplication de deux matrices utilisant les threads

Description

C'est une application qui permet de multiplier deux matrices A et B de taille 100*100 utilisant 4 threads.

1ère étape: Remplir aléatoirement entre 1 et 99 la matrice A et la matrice B utilisant la fonction random.

- La fonction LoadMatrix() sert à remplir une matrice M, elle prend comme paramètres le nombre des colonnes, le nombre des lignes ainsi que le nom du fichier dans lequel on va stocker la matrice mat.

```
public static int[][] loadMatrix(int nbrrows, int nbrcols, String filename)
```

```
public class MatricesProduit {  
  
    public static int[][] c;  
    public static int[][] a;  
    public static int[][] b;  
  
    public static int[][] loadMatrix(int nbrrows, int nbrcols, String filename) throws Exception {  
        //FileOutputStream sortie = new FileOutputStream(filename);  
  
        PrintWriter pw = new PrintWriter(new FileOutputStream(filename));  
        int mat[][] = new int[nbrrows][nbrcols];  
        for (int i = 0; i < nbrrows; i++) {  
            for (int j = 0; j < nbrcols; j++) {  
                mat[i][j] = (int) (1 + (Math.random() * (100 - 1)));  
                pw.print(mat[i][j] + " ");  
            }  
            pw.println();  
        }  
        pw.close();  
        return mat;  
    }  
}
```

2ème étape: Multiplier les deux matrices utilisant 4 threads.

Au lieu d'utiliser un seul thread pour faire la multiplication de deux matrices A et B de dimension 100*100 qui va prendre un temps d'exécution assez lent , alors on a pensé à utiliser 4 threads pour diminuer le temps d'exécution.

Donc l'idée générale est de diviser la matrice A sur 4 : dans ce cas là, dans chaque thread la matrice A va contenir 25 lignes au lieu de 100 lignes.

Le premier Thread:

La matrice A contiendra 25 lignes : `int k = (a.length) / 4`

```
public static class Thread1 extends Thread {
    @Override
    public void run() {
        int m = a.length;
        int n = b[0].length;
        int k = (a.length) / 4;

        for (int i = 0; i <= k; i++) {
            for (int j = 0; j < n; j++) {
                c[i][j]=0;
                for (int l = 0; l < b.length; l++) {
                    c[i][j] = c[i][j] + a[i][l] * b[l][j];
                }
            }
        }
    }
}
```

Le 2ème Thread:

La matrice A va contenir 25 lignes (de la ligne 26 jusqu'à 50).

`int k = (a.length) / 2+1=51`

Au début k=s avec `int s = ((a.length) /4)+1=26.`

```
public static class Thread2 extends Thread {
    @Override
    public void run() {
        int m = a.length;
        int n = b[0].length;
        int k = (a.length) / 2+1;
        int s = ((a.length) /4)+1;

        for (int i = s ; i < k; i++) {
            for (int j = 0; j < n; j++) {
                c[i][j]=0;
                for (int l = 0; l < b.length; l++) {
                    c[i][j] = c[i][j] + a[i][l] * b[l][j];
                }
            }
        }
    }
}
```

Le 3ème Thread:

La matrice A contiendra 25 lignes (de la ligne 52 jusqu'à 75).

$\text{int } k = ((3 * (a.length)) / 4) + 1 = 76$

Au début $k=s$ avec $\text{int } s = (a.length) / 2 + 1 = 52$.

```
public static class Thread3 extends Thread {  
  
    @Override  
    public void run() {  
        int m = a.length;  
        int n = b[0].length;  
        int k = ((3 * (a.length)) / 4) + 1;  
        int s = (a.length) / 2 + 1;  
  
        for (int i = s; i < k; i++) {  
            for (int j = 0; j < n; j++) {  
                c[i][j] = 0;  
                for (int l = 0; l < b.length; l++) {  
                    c[i][j] = c[i][j] + a[i][l] * b[l][j];  
                }  
            }  
        }  
    }  
}
```

Le 4ème thread:

La matrice A va contenir 25 lignes (de la ligne 76 jusqu'à m=100).

$\text{int } k = ((3 * (a.length)) / 4) + 1 = 76$

```
public static class Thread4 extends Thread {  
  
    @Override  
    public void run() {  
        int m = a.length;  
        int n = b[0].length;  
        int k = ((3 * (a.length)) / 4) + 1;  
  
        for (int i = k; i < m; i++) {  
            for (int j = 0; j < n; j++) {  
                c[i][j] = 0;  
                for (int l = 0; l < b.length; l++) {  
                    c[i][j] = c[i][j] + a[i][l] * b[l][j];  
                }  
            }  
        }  
    }  
}
```

3ème étape: Après la multiplication de la matrice A et la matrice B, maintenant on va afficher le résultat de la multiplication sur le console à l'aide la fonction `displayMatrix()`.

```
public static void displayMatrix(int c[][][]) throws Exception {
    int rows = c.length;
    int cols = c[0].length;
    System.out.println("rows " + rows + " columns " + cols);

    for (int i = 0; i < rows; i++) {
        for (int j = 0; j < cols; j++) {
            System.out.print(c[i][j] + " ");
        }
        System.out.println();
    }
}
```

Dernière étape: La fonction `main()` pour créer des objets de 4 threads, la fonction `start` permet de démarrer ces derniers.

Lorsque l'instruction : `m1.join()` s'exécute, il y'a une mise en attente du thread en cours d'exécution jusqu'à ce que le thread m1 soit terminé. m1 bloque le thread en cours d'exécution, et celui-ci sera débloqué par la fin de ce thread.

`System.out.println("the time is "+(t2-t1))`: va afficher la durée d'exécution de notre programme. T1 représente le temps du début de l'exécution et t2 le temps de la fin d'exécution.

```
public static void main(String[] args) throws Exception {
    long t1 = System.currentTimeMillis();
    a = LoadMatrix(100,100,"a.txt");
    b = LoadMatrix(100,100,"b.txt");
    c = new int [a.length][b[0].length];

    Thread1 m1 = new Thread1();
    Thread2 m2 = new Thread2();
    Thread3 m3 = new Thread3();
    Thread4 m4 = new Thread4();
    m1.start();
    m2.start();
    m3.start();
    m4.start();
    m1.join();
    m2.join();
    m3.join();
    m4.join();

    displayMatrix(c);
    long t2 = System.currentTimeMillis();
    System.out.println("the time is "+(t2-t1));
}
```


Démarrage de notre programme:

```

255405 267250 216511 257256 225795 251224 269650 254152 2
248581 306822 243246 279029 273286 275374 302508 255294 2
247152 291660 225153 248995 259217 255789 282826 221006 2
252454 293726 234590 255169 263050 243900 282181 238335 2
214922 249498 201253 226198 219687 228024 255774 210907 2
238309 275871 215541 239009 239105 237231 253741 222188 2
244836 288711 225323 242173 258601 242944 268069 209584 2
252771 283861 253770 270788 250352 270739 299772 248128 2
242522 291587 225104 254836 242132 247912 267281 233114 2
231083 267062 212618 240616 249888 232660 263917 231914 2
248844 299693 228634 270642 251856 268493 296952 231030 2
235197 298820 233781 262098 246683 248821 285938 226967 2
219427 249344 204950 225788 224426 216442 253658 214749 2
254980 285451 219849 266782 262951 258175 291059 225910 2
245064 289214 211081 238637 250360 235091 271178 214767 2
247236 299605 234339 271449 271616 250342 284141 232530 2
244550 311379 244787 274426 279597 273887 307214 251480 2
240754 253764 207882 241453 243056 223272 263124 222273 2
240733 272651 213846 247523 255200 259054 280058 225115 2
the time is 283

```

MatricesProduit.java	b.txt	a.txt														
1	51	56	20	73	31	69	39	49	53	75	60	86	15	51	30	
2	19	75	36	25	37	70	11	8	69	98	67	3	70	38	17	
3	1	74	2	43	24	81	27	71	74	7	43	19	5	49	76	!
4	56	98	67	23	71	86	5	13	60	98	46	56	88	60	11	
5	5	2	35	67	76	13	44	46	52	88	56	40	71	26	30	
6	6	3	17	16	96	47	20	85	97	74	66	36	47	86	48	
7	51	56	19	3	49	96	37	63	69	47	39	89	40	26	76	
8	80	43	55	4	48	72	44	29	64	17	28	43	38	92	41	
9	97	60	28	21	73	76	4	37	5	48	91	84	60	11	79	
0	27	11	53	84	67	62	8	80	9	20	65	12	6	42	59	
1	54	36	3	32	61	28	69	72	79	41	8	82	8	92	25	
2	41	91	20	52	45	10	93	71	77	66	51	77	74	33	41	
3	51	84	79	59	30	16	8	64	11	39	76	46	72	50	66	
4	30	89	91	50	34	15	72	45	81	24	19	52	82	10	16	
5	84	50	29	5	89	76	98	27	43	52	44	79	49	58	64	
6	88	13	77	40	46	38	70	6	94	98	95	10	4	6	31	
7	36	92	77	19	21	94	12	17	75	16	62	64	43	12	66	
8	7	71	58	85	36	32	91	10	19	97	98	99	24	97	43	
9	60	27	61	62	52	19	67	2	26	19	60	14	41	27	74	
0	17	55	66	77	13	40	27	42	64	6	27	33	13	55	33	
1	4	8	32	99	92	15	21	5	88	4	11	11	43	31	88	!
2	32	74	13	43	37	63	30	38	0	33	54	30	33	30	40	

MatricesProduit.java	b.txt	a.txt
1 12	71	75
2 95	23	60
3 61	75	19
4 88	96	17
5 81	86	32
6 38	44	33
7 56	24	32
8 42	80	69
9 14	81	4
10 26	31	44
11 86	30	85
12 72	91	59
13 33	97	92
14 79	74	78
15 54	89	57
16 98	89	33
17 45	17	63
18 69	76	67
19 55	46	19
20 66	71	39
21 24	84	76
22 18	83	65
23 30	75	13
70 10	32	43
30 85	5	69
62 46	8	79
61 24	43	38
28 94	41	82
3 82	73	87
8 71	2	64
80 51	45	3
81 56	97	52
97 14	21	28
28 97	14	21
46 82	9	48
42 33	23	99
15 80	88	65
83 89	54	40
59 83	95	67
11 34	45	9
25 45	34	18
33 76	66	65
90 88	29	20
4 21	26	3
36 16	77	75
57 83	17	13
66 56	13	56
12 91	24	36
15 44	75	19
17 21	58	61
62 28	94	41
5 3	82	73
12 80	51	45
79 81	56	97
28 97	14	21
4 80	28	97
44 46	6	46
85 40	58	42
59 15	98	15
92 16	86	83
7 14	59	83
57 25	72	11
33 23	72	25
17 50	58	33
67 38	90	88
19 4	21	26
39 36	16	77
76 57	83	17
65 66	56	13
13 12	91	24
75 19	20	63
96 17	95	21
86 32	88	62
44 33	69	5
24 32	70	12
80 69	70	79
81 4	80	28
31 44	46	6
30 85	40	58
91 59	15	98
97 92	16	86
74 78	7	14
89 57	25	72
89 33	23	72
17 63	50	58
76 67	38	90
46 19	4	21
71 39	36	16
84 76	57	83
83 65	66	56
75 13	12	91
12 91	24	36
75 19	20	63
96 17	95	21
86 32	88	62
44 33	69	5
24 32	70	12
80 69	70	79
81 4	80	28
31 44	46	6
30 85	40	58
91 59	15	98
97 92	16	86
74 78	7	14
89 57	25	72
89 33	23	72
17 63	50	58
76 67	38	90
46 19	4	21
71 39	36	16
84 76	57	83
83 65	66	56
75 13	12	91
12 91	24	36
75 19	20	63
96 17	95	21
86 32	88	62
44 33	69	5
24 32	70	12
80 69	70	79
81 4	80	28
31 44	46	6
30 85	40	58
91 59	15	98
97 92	16	86
74 78	7	14
89 57	25	72
89 33	23	72
17 63	50	58
76 67	38	90
46 19	4	21
71 39	36	16
84 76	57	83
83 65	66	56
75 13	12	91
12 91	24	36
75 19	20	63
96 17	95	21
86 32	88	62
44 33	69	5
24 32	70	12
80 69	70	79
81 4	80	28
31 44	46	6
30 85	40	58
91 59	15	98
97 92	16	86
74 78	7	14
89 57	25	72
89 33	23	72
17 63	50	58
76 67	38	90
46 19	4	21
71 39	36	16
84 76	57	83
83 65	66	56
75 13	12	91
12 91	24	36
75 19	20	63
96 17	95	21
86 32	88	62
44 33	69	5
24 32	70	12
80 69	70	79
81 4	80	28
31 44	46	6
30 85	40	58
91 59	15	98
97 92	16	86
74 78	7	14
89 57	25	72
89 33	23	72
17 63	50	58
76 67	38	90
46 19	4	21
71 39	36	16
84 76	57	83
83 65	66	56
75 13	12	91
12 91	24	36
75 19	20	63
96 17	95	21
86 32	88	62
44 33	69	5
24 32	70	12
80 69	70	79
81 4	80	28
31 44	46	6
30 85	40	58
91 59	15	98
97 92	16	86
74 78	7	14
89 57	25	72
89 33	23	72
17 63	50	58
76 67	38	90
46 19	4	21
71 39	36	16
84 76	57	83
83 65	66	56
75 13	12	91
12 91	24	36
75 19	20	63
96 17	95	21
86 32	88	62
44 33	69	5
24 32	70	12
80 69	70	79
81 4	80	28
31 44	46	6
30 85	40	58
91 59	15	98
97 92	16	86
74 78	7	14
89 57	25	72
89 33	23	72
17 63	50	58
76 67	38	90
46 19	4	21
71 39	36	16
84 76	57	83
83 65	66	56
75 13	12	91
12 91	24	36
75 19	20	63
96 17	95	21
86 32	88	62
44 33	69	5
24 32	70	12
80 69	70	79
81 4	80	28
31 44	46	6
30 85	40	58
91 59	15	98
97 92	16	86
74 78	7	14
89 57	25	72
89 33	23	72
17 63	50	58
76 67	38	90
46 19	4	21
71 39	36	16
84 76	57	83
83 65	66	56
75 13	12	91
12 91	24	36
75 19	20	63
96 17	95	21
86 32	88	62
44 33	69	5
24 32	70	12
80 69	70	79
81 4	80	28
31 44	46	6
30 85	40	58
91 59	15	98
97 92	16	86
74 78	7	14
89 57	25	72
89 33	23	72
17 63	50	58
76 67	38	90
46 19	4	21
71 39	36	16
84 76	57	83
83 65	66	56
75 13	12	91
12 91	24	36
75 19	20	63
96 17	95	21
86 32	88	62
44 33	69	5
24 32	70	12
80 69	70	79
81 4	80	28
31 44	46	6
30 85	40	58
91 59	15	98
97 92	16	86
74 78	7	14
89 57	25	72
89 33	23	72
17 63	50	58
76 67	38	90
46 19	4	21
71 39	36	16
84 76	57	83
83 65	66	56
75 13	12	91
12 91	24	36
75 19	20	63
96 17	95	21
86 32	88	62
44 33	69	5
24 32	70	12
80 69	70	79
81 4	80	28
31 44	46	6
30 85	40	58
91 59	15	98
97 92	16	86
74 78	7	14
89 57	25	72
89 33	23	72
17 63	50	58
76 67	38	90
46 19	4	21
71 39	36	16
84 76	57	83
83 65	66	56
75 13	12	91
12 91	24	36
75 19	20	63
96 17	95	21
86 32	88	62
44 33	69	5
24 32	70	12
80 69	70	79
81 4	80	28
31 44	46	6
30 85	40	58
91 59	15	98
97 92	16	86
74 78	7	14
89 57	25	72
89 33	23	72
17 63	50	58
76 67	38	90
46 19	4	21
71 39	36	16
84 76	57	83
83 65	66	56
75 13	12	91
12 91	24	36
75 19	20	63
96 17	95	21
86 32	88	62
44 33	69	5
24 32	70	12
80 69	70	79
81 4	80	28
31 44	46	6
30 85	40	58
91 59	15	98
97 92	16	86
74 78	7	14
89 57	25	72
89 33	23	72
17 63	50	58
76 67	38	90
46 19	4	21
71 39	36	16
84 76	57	83
83 65	66	56
75 13	12	91
12 91	24	36
75 19	20	63
96 17	95	21
86 32	88	62
44 33	69	5
24 32	70	12
80 69	70	79
81 4	80	28
31 44	46	6
30 85	40	58
91 59	15	98
97 92	16	86
74 78	7	14
89 57	25	72
89 33	23	72
17 63	50	58
76 67	38	90
46 19	4	21
71 39	36	16
84 76	57	83
83 65	66	56
75 13	12	91
12 91	24	36
75 19	20	63
96 17	95	21
86 32	88	62
44 33	69	5
24 32	70	12
80 69	70	79
81 4	80	28
31 44	46	6
30 85	40	58
91 59	15	98
97 92	16	86
74 78	7	14
89 57	25	72
89 33	23	72
17 63	50	58
76 67	38	90
46 19	4	21
71 39	36	16
84 76	57	83
83 65	66	56
75 13	12	91
12 91	24	36
75 19	20	63
96 17	95	21
86 32	88	62
44 33	69	5
24 32	70	12
80 69	70	79
81 4	80	28
31 44	46	6
30 85	40	58
91 59	15	98
97 92	16	86
74 78	7	14
89 57	25	72
89 33	23	72
17 63	50	58
76 67	38	90
46 19	4	21
71 39	36	16
84 76	57	83
83 65	66	56
75 13	12	91
12 91	24	36
75 19	20	63
96 17	95	21
86 32	88	62
44 33	69	5
24 32	70	12
80 69	70	79
81 4	80	28

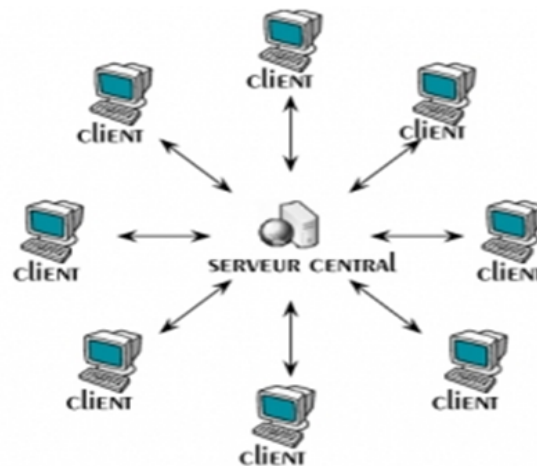
3ème application : Client/Serveur

Socket en java :

Un socket est un point de terminaison d'une communication bidirectionnelle, c'est-à-dire entre un client et un serveur en cours d'exécution sur un réseau donné. Les deux sont liés par un même numéro de port TCP de sorte que la couche puisse identifier la demande de partage de données.

Fonctionnement

Un serveur fonctionne sur une machine bien définie et est lié à un numéro de port spécifique. Le serveur se met simplement à l'écoute d'un client, qui demande une connexion.



1) Le serveur crée un "socket serveur" pour accueillir les clients (associée à un port) et se met en attente.

2) Le client se connecte au socket serveur; Deux sockets sont alors créés : un "socket client", côté client, et un "socket service client" côté serveur.

Ces sockets sont connectées entre eux

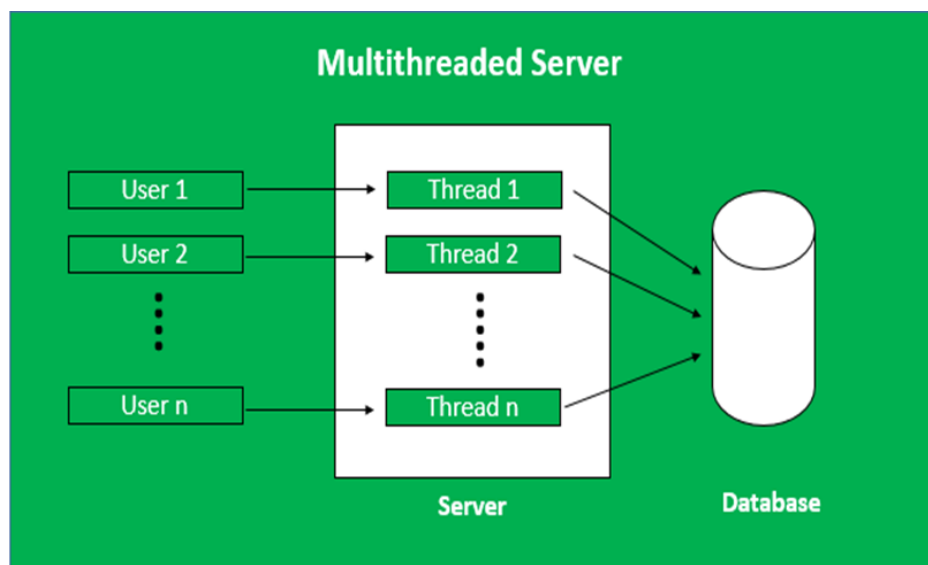
3) Le client et le serveur communiquent par les sockets.

Les classes des sockets en JAVA:

Plusieurs classes interviennent lors de la réalisation d'une communication par sockets.

- La classe `java.net.InetAddress` permet de manipuler des adresses IP.
- La classe `java.net.SocketServer` permet de programmer l'interface côté serveur en mode connecté.
- La classe `java.net.Socket` permet de programmer l'interface côté client et la communication effective par flot via les sockets.
- Les classes `java.net.DatagramSocket` et `java.net.DatagramPacket` permettent de programmer la communication en mode datagramme.

Multithreaded Server :

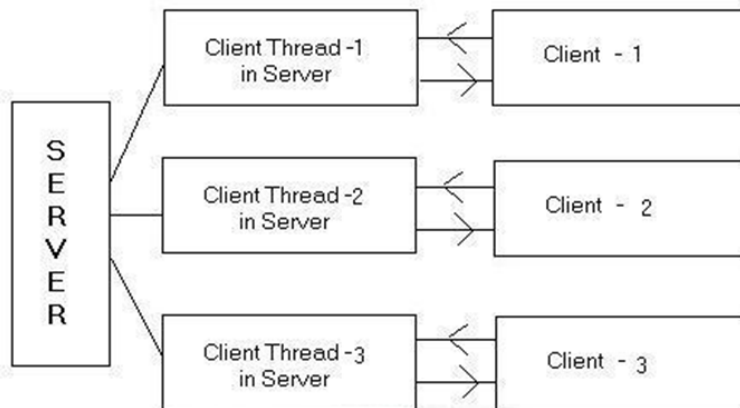


Pour qu'un serveur puisse communiquer avec plusieurs clients en même temps et puisse attendre une connexion à tout moment (pour chaque connexion, il faut créer un nouveau Thread associé à la socket du client connecté puis attendre à nouveau une nouvelle connexion. On introduit alors la notion du Thread et la notion du Multithreaded server)

Un serveur multithread est un serveur ayant plusieurs threads. Lorsqu'un client envoie la demande, un fil est généré à travers lequel un utilisateur peut communiquer avec le serveur. Nous devons

généraliser plusieurs threads pour accepter plusieurs demandes de plusieurs clients en même temps.

Utilisation des Threads



Le principe d'utilisation des Threads est simple. Après avoir créer un objet `ServerSocket` par le serveur, on le place (l'objet) comme paramètre à un constructeur de la classe qui implémente la classe `Runnable` ou étend la classe `Thread`, dès qu'un client souhaite se connecter avec le serveur, un Thread s'occupe de la connexion, il ne sera plus la peine de faire appel au serveur lorsqu'un client souhaite se connecter, tout le boulot sera confié à un Thread.

Exemple d'une application Client/Serveur simple

Ce qu'on veut dire par une application client/serveur simple, c'est une application qui assure la communication entre un serveur et un client.

Pour cela on aura besoin de deux classes :

Une classe `client.java` et une classe `serveur.java` sous un même projet Java

Pour ce sur un IDE , nous on a utilisé eclipse, on crée un projet de type `java application`.

Serveur.java :

```
import java.io.*;
import java.net.*;
public class Serveur {
    public static void main(String[] args) throws IOException
    {
        try (ServerSocket ss = new ServerSocket(1026)) {
            System.out.println("En attente de la connexion d'un client");
            Socket s=ss.accept();
            System.out.println("connexion établie");
            //on récupère la donnée envoyée par le client
            DataInputStream in=new DataInputStream (s.getInputStream());
            String nomClient=in.readUTF();
            //on effectue un traitement
            String s1="bienvenue "+nomClient+", t'es bien connecté";
            //le serveur envoie la donnée au client c à dire s1
            DataOutputStream out= new DataOutputStream(s.getOutputStream());
            out.writeUTF(s1);
        }
    }
}
```

Le serveur reste à l'écoute sur un port pour une éventuelle requête de la part du client

- `ServerSocket ss = new ServerSocket(1026)`
- `Socket s=ss.accept();`

Le client se connecte au serveur et demande un service au serveur

Pour assurer la communication, les machines utilisent des objets sockets (Adresse IP , port)

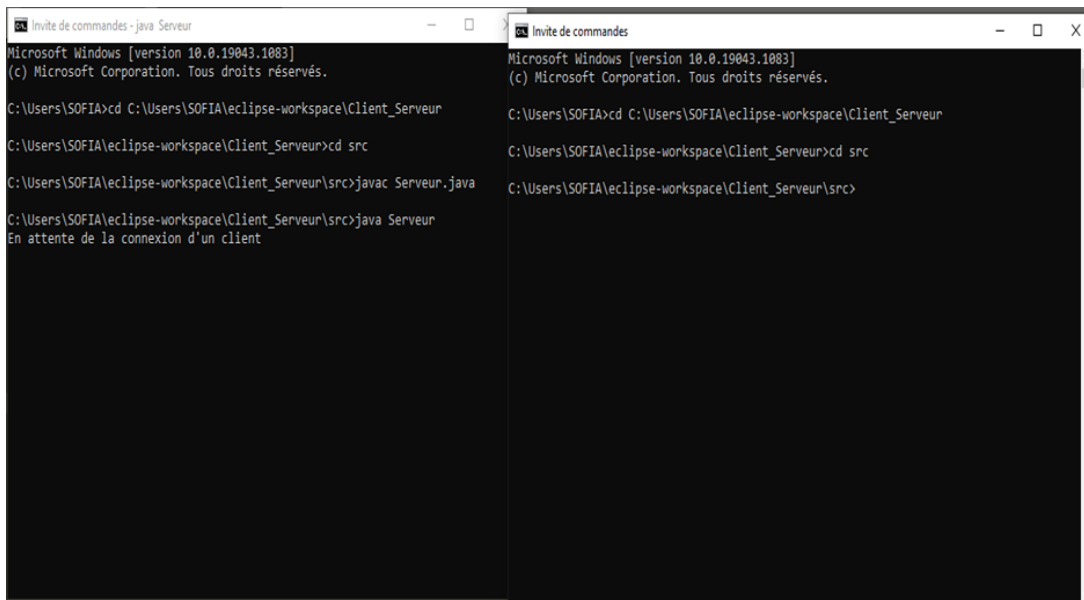
- `Socket client = new Socket ("127.0.0.13",1026)`

Le serveur effectue le traitement et rend le service.

Client.java :

```
import java.io.DataInputStream;
import java.io.DataOutputStream;
import java.io.IOException;
import java.net.Socket;
import java.net.UnknownHostException;
import java.util.Scanner;
public class Client {
    public static void main (String[] args) throws UnknownHostException, IOException {
        try (Socket client = new Socket ("127.0.0.13",1026)) {
            System.out.println("Nom Client");
            try (Scanner sc = new Scanner(System.in)) {
                String nomClient= sc.next();
                //on envoie la donnée au serveur
                DataOutputStream out = new DataOutputStream(client.getOutputStream());
                out.writeUTF(nomClient);
            }
            //on récupère la donnée envoyée par le serveur
            DataInputStream in =new DataInputStream(client.getInputStream());
            String s1= in.readUTF();
            System.out.println(s1);
        }
    }
}
```

Sur 2 invites de commande séparées, on se déplace sur le dossier Src du projet java. On exécute le serveur en premier qui attend la connexion d'un client :



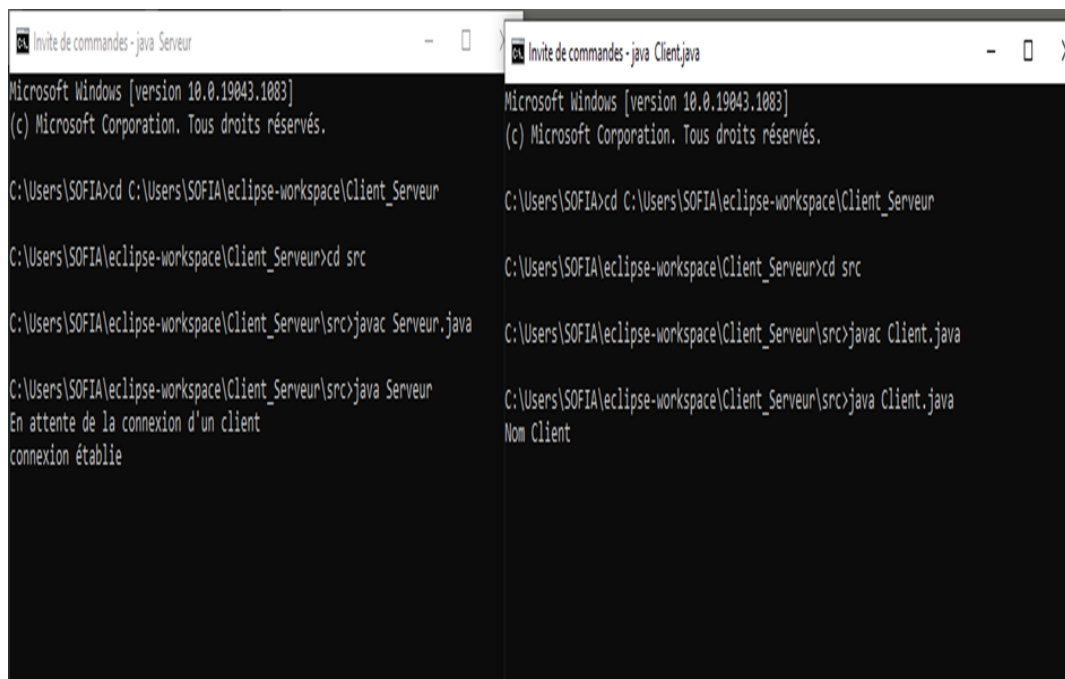
```
Microsoft Windows [version 10.0.19043.1083]
(c) Microsoft Corporation. Tous droits réservés.

C:\Users\SOFIA>cd C:\Users\SOFIA\workspace\Client_Serveur
C:\Users\SOFIA\workspace\Client_Serveur>cd src
C:\Users\SOFIA\workspace\Client_Serveur\src>javac Serveur.java
C:\Users\SOFIA\workspace\Client_Serveur\src>java Serveur
En attente de la connexion d'un client
```

```
Microsoft Windows [version 10.0.19043.1083]
(c) Microsoft Corporation. Tous droits réservés.

C:\Users\SOFIA>cd C:\Users\SOFIA\workspace\Client_Serveur
C:\Users\SOFIA\workspace\Client_Serveur>cd src
C:\Users\SOFIA\workspace\Client_Serveur\src>
```

On exécute après le Client. Le serveur notifie que la connexion est établie :



```
Microsoft Windows [version 10.0.19043.1083]
(c) Microsoft Corporation. Tous droits réservés.

C:\Users\SOFIA>cd C:\Users\SOFIA\workspace\Client_Serveur
C:\Users\SOFIA\workspace\Client_Serveur>cd src
C:\Users\SOFIA\workspace\Client_Serveur\src>javac Serveur.java
C:\Users\SOFIA\workspace\Client_Serveur\src>java Serveur
En attente de la connexion d'un client
connexion établie
```

```
Microsoft Windows [version 10.0.19043.1083]
(c) Microsoft Corporation. Tous droits réservés.

C:\Users\SOFIA>cd C:\Users\SOFIA\workspace\Client_Serveur
C:\Users\SOFIA\workspace\Client_Serveur>cd src
C:\Users\SOFIA\workspace\Client_Serveur\src>javac Client.java
C:\Users\SOFIA\workspace\Client_Serveur\src>java Client.java
Nom Client
```

```
Microsoft Windows [version 10.0.19043.1083]
(c) Microsoft Corporation. Tous droits réservés.

C:\Users\SOFIA>cd C:\Users\SOFIA\workspace\Client_Serveur
C:\Users\SOFIA\workspace\Client_Serveur>cd src
C:\Users\SOFIA\workspace\Client_Serveur\src>javac Serveur.java
C:\Users\SOFIA\workspace\Client_Serveur\src>java Serveur
En attente de la connexion d'un client
connexion établie
C:\Users\SOFIA\workspace\Client_Serveur\src>

Microsoft Windows [version 10.0.19043.1083]
(c) Microsoft Corporation. Tous droits réservés.

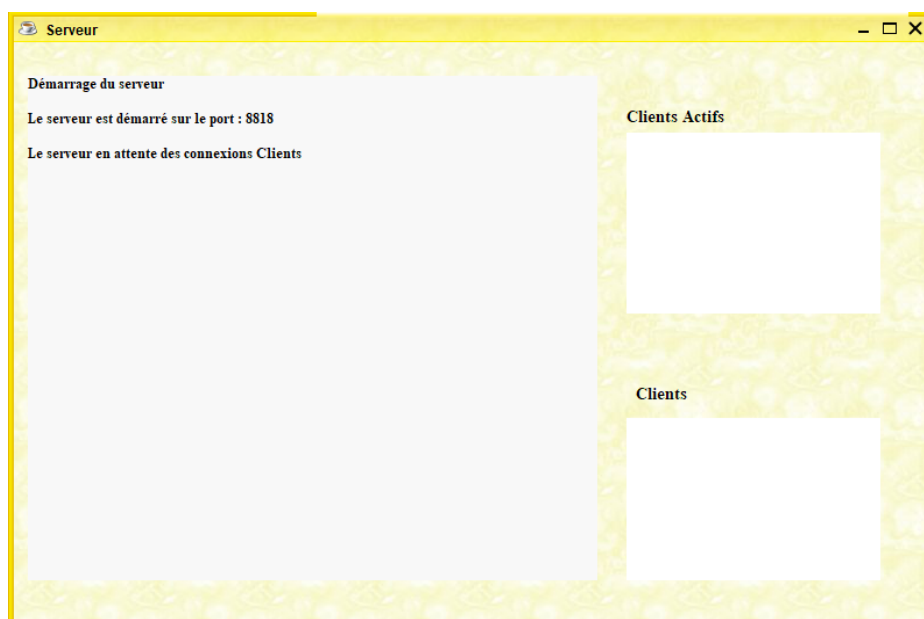
C:\Users\SOFIA>cd C:\Users\SOFIA\workspace\Client_Serveur
C:\Users\SOFIA\workspace\Client_Serveur>cd src
C:\Users\SOFIA\workspace\Client_Serveur\src>javac Client.java
C:\Users\SOFIA\workspace\Client_Serveur\src>java Client.java
Nom Client
safae zellou
bienvenue safae,t'es bien connecté
C:\Users\SOFIA\workspace\Client_Serveur\src>
```

Application de chat Multi-clients

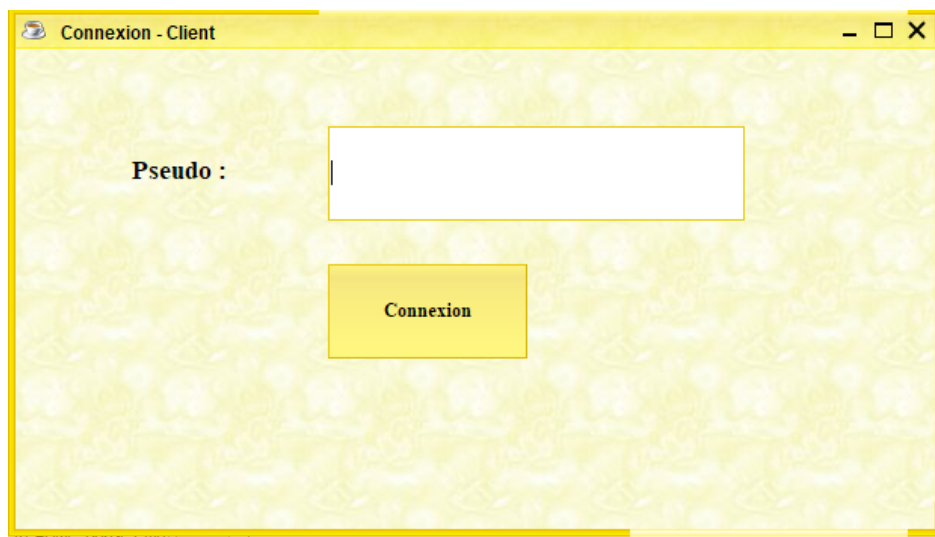
Description :

Notre application contient 3 classes qui implémente JFrame. Le résultat de l'exécution est le suivant :

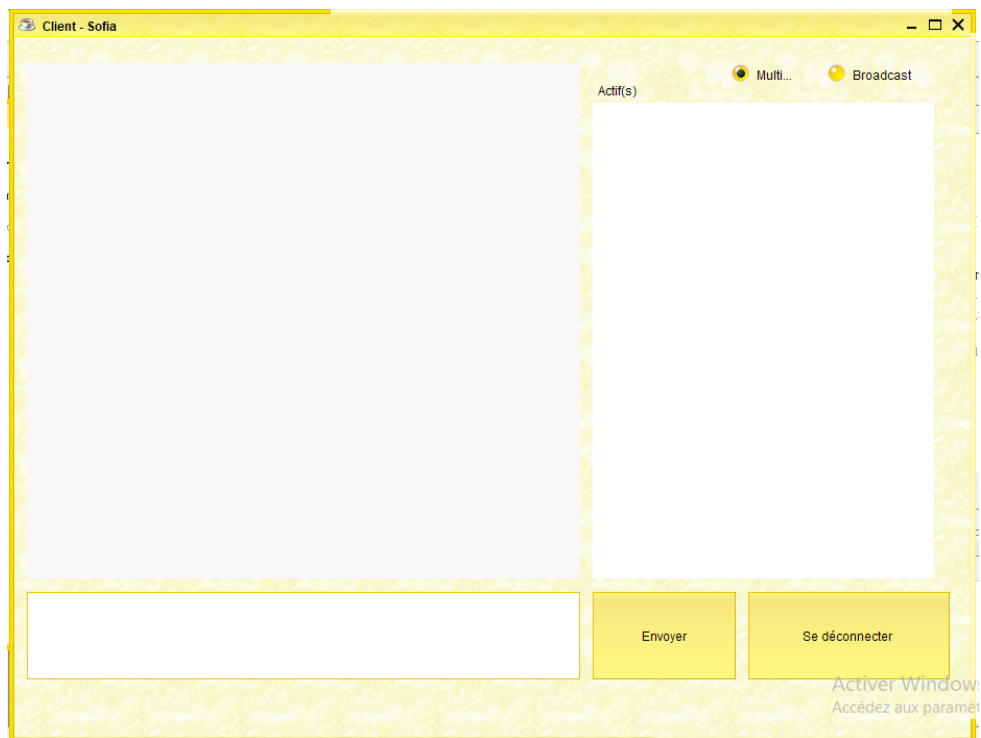
La fenêtre du serveur :



La fenêtre connexion :



La fenêtre client :



La première classe **"ServerView"** contient le code pour le serveur, elle contient la vue de la fenêtre serveur (avec java Swing) et 3 threads :

1. **ClientAccept** : pour accepter les connexions de plusieurs clients à la fois.
2. **MsgRead** : pour lire les messages provenant des clients et faire les traitements appropriés.
3. **PrepareClientList**: pour préparer la liste des clients actifs pour l'afficher sur l'UI (User Interface).

La deuxième classe **"LoginClient"** gère la connexion de chaque client, si un pseudo est déjà utilisé un message d'erreur sera affiché.

La troisième classe de notre application **"ClientView"** contient le code pour les clients, chaque client peut chatter en unicast, multicast ou broadcast.

Exécution de l'application :

Pour exécuter l'application, la première classe à exécuter **"ServerView"** pour démarrer le serveur ceci démarre sur un port spécifique (ex : 8818) et se met à l'attente des connexions client(s).

Une fois que le serveur a bien démarré, on exécute la classe **"LoginClient"** pour connecter un nouveau client. Si la connexion est réussie, une fenêtre Client s'affiche là où le client peut envoyer et recevoir des messages et également se déconnecter.

La connexion et la déconnexion des clients sont notifiées sur la fenêtre du serveur.

Résultats :

Les clientes suivantes (hala bahida, fati bouya , sofia zellou et wiam zellou) sont connectées - notifiées sur le serveur , hala a envoyé un message en broadcast tout le monde l'a reçu ensuite elle a envoyé à chacune un message qu'elle a reçu toute seule.

La cliente sofia se déconnecte et quitte la conversation, cela est notifié sur le serveur.

Sur la liste des clients actifs, sofia n'est plus en ligne mais elle existe toujours sur la liste des clients (all users).

Client - wiam zellou

< fati bouya >Hello tout le monde, je suis Fati Bouya

Actif(s)

Multi...

Broadcast

Client - hala bahida

< fati bouya >Hello tout le monde, je suis Fati Bouya
< fati bouya >Hala comment vas-tu ? ça était les vacances ?

Actif(s)

wiam zellou
sofia zellou

Multi...

Broadcast

Client - sofia zellou

< fati bouya >Hello tout le monde, je suis Fati Bouya
< fati bouya >Hi sofia, comment vas-tu ? tu te rappelles de moi hein ?

Actif(s)

wiam zellou
hala bahida
fati bouya

Multi...

Broadcast

Client - fati bouya

< Vous envoyez un message à tout le monde >Hello tout le monde, je suis Fati Bouya
< Vous envoyez un message à sofia zellou>Hi sofia, comment vas-tu ? tu te rappelles de moi hein ?
< Vous envoyez un message à hala bahida>Hala comment vas-tu ? ça était les vacances ?

Actif(s)

wiam zellou
hala bahida
sofia zellou

Multi...

Broadcast

Client - wiam zellou

< fati bouya >Hello tout le monde, je suis Fati Bouya
sofia zellou est déconnecté...

Actif(s)

hala bahida
fati bouya

Multi...

Broadcast

Client - hala bahida

< fati bouya >Hello tout le monde, je suis Fati Bouya
< fati bouya >Hala comment vas-tu ? ça était les vacances ?
sofia zellou est déconnecté...

Actif(s)

wiam zellou
fati bouya

Multi...

Broadcast

Serveur

Démarrage du serveur

Le serveur est démarré sur le port : 8818

Le serveur en attente des connexions Clients

Client sofia zellou est connecté ...
Client wiam zellou est connecté ...
Client hala bahida est connecté ...
Client fati bouya est connecté ...
sofia zellou est déconnecté...

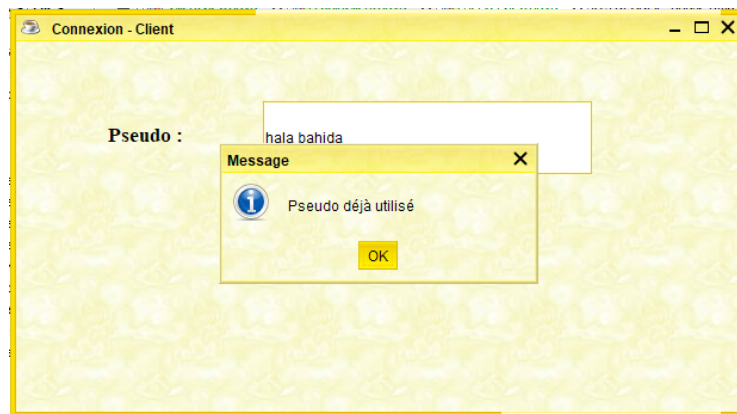
Clients Actifs

wiam zellou
hala bahida
fati bouya

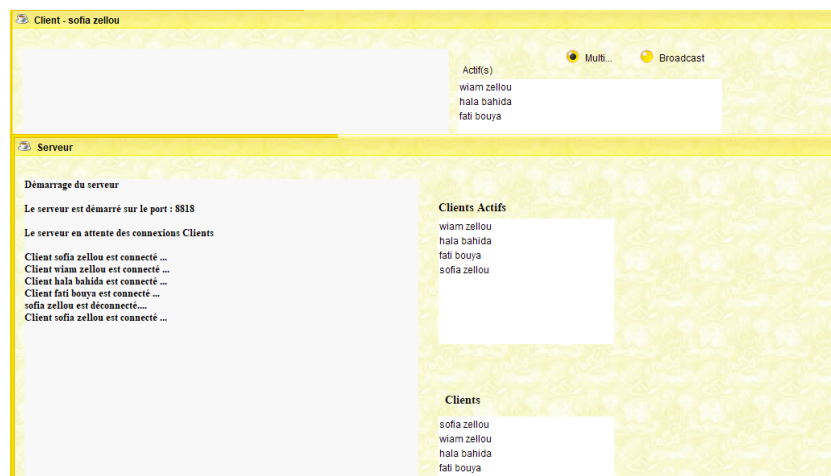
Clients

sofia zellou
wiam zellou
hala bahida
fati bouya

La cliente "hala bahida" est déjà en ligne, un message d'erreur est généré :



La cliente "sofia zellou" est de nouveau connectée. Elle apparaît maintenant sur la liste des utilisateurs actifs mais elle n'est pas dupliquée sur la liste de tous les utilisateurs. Elle n'a plus l'accès aux anciens messages.



Et finalement "fadwa saoiabi" a rejoint la conversation mais elle a envoyé un message personnalisé à uniquement "sofia zellou" et "hala bahida".

