

Compte rendu TD6 SLAM 5

Dans cette partie, en tant que stagiaire dans une entreprise, votre but sera de concevoir une application java client lourd en SWING qui permet de récupérer une image dans un dossier local et de l'insérer dans une base de donnée.

Pour cela dans un premier temps, nous allons créer une base de donnée qui s'intitule "db_images" et qui contiendra la table "monimage".

Création base de donnée :

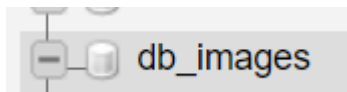
Bases de données



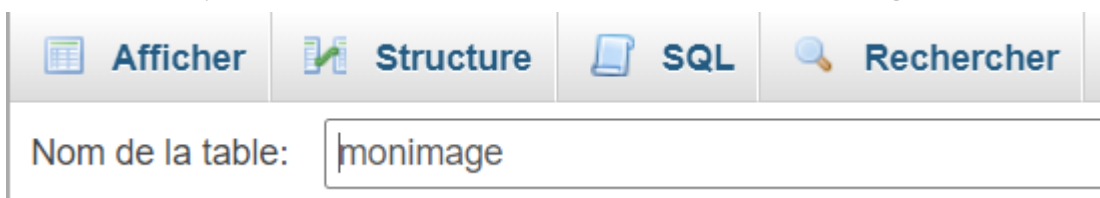
Créer une base de données

db_images utf8_bin Créer

et on appuie sur "créer" :



Maintenant que ceci a été fait, créons la table "monimage" :



Afficher Structure SQL Rechercher

Nom de la table: monimage

Et ainsi que ces colonnes :

#	Nom	Type	Interclassement	Attributs	Null	Valeur par défaut	Extra	Action
<input type="checkbox"/>	1 id	varchar(10)	utf8_general_ci		Non	Aucune		Modifier Supprimer Primaire Unique Index Spatial Texte entier plus
<input type="checkbox"/>	2 nom	varchar(50)	utf8_general_ci		Non	Aucune		Modifier Supprimer Primaire Unique Index Spatial Texte entier plus
<input type="checkbox"/>	3 description	varchar(500)	utf8_general_ci		Non	Aucune		Modifier Supprimer Primaire Unique Index Spatial Texte entier plus
<input type="checkbox"/>	4 image	longblob			Non	Aucune		Modifier Supprimer Primaire Unique Index Spatial Texte entier plus

Une fois que ceci a été fait, passons maintenant à la partie code. Pour cela nous allons créer une classe qui va se nommer "PhotoDB" comme ci-dessous :

New Java Class

Java Class

The use of the default package is discouraged.

Source folder:

Package:

Enclosing type:

Name:

Modifiers: public package private protected
 abstract final static

Superclass:

Interfaces:

Which method stubs would you like to create?

public static void main(String[] args)
 Constructors from superclass
 Inherited abstract methods

Do you want to add comments? (Configure templates and default value [here](#))

Generate comments

Voici le programme :

```

13 public class PhotoDB extends JFrame{
14     JButton btnAjouter;
15     JButton btnParcourir;
16     JButton btnSupprimer;
17     JLabel label;
18     JTextField txtId;
19     JTextField textNom;
20     JTextArea area;
21     String s;
22
23     public PhotoDB() {
24         super ("Insérer une photo dans une base de donnée");
25         this.setDefaultCloseOperation(DISPOSE_ON_CLOSE);
26         btnAjouter= new JButton("Ajouter");
27         btnAjouter.setBounds(200, 300, 100, 40);
28
29         btnParcourir = new JButton("Parcourir");
30         btnParcourir.setBounds(80, 300, 100, 40);
31
32         btnSupprimer= new JButton("Supprimer");
33         btnSupprimer.setBounds(80, 300, 100, 40);
34
35         txtId= new JTextField("ID");
36         txtId.setBounds(320, 290, 100, 20);
37
38         textNom= new JTextField("Nom");
39         textNom.setBounds(320, 330, 100, 20);
40
41         area= new JTextArea("Description",100,100);
42
43         JScrollPane pane= new JScrollPane(area);
44         pane.setBounds(450, 270, 200, 100);
45
46         label = new JLabel();
47         label.setBounds(10, 10, 670, 250);
48
49
50
51         btnParcourir.addActionListener(new ActionListener() {
52             @Override
53             public void actionPerformed(ActionEvent e) {
54                 JFileChooser fileChooser = new JFileChooser();
55                 fileChooser.setCurrentDirectory(new File (System.getProperty("user.dir")));
56                 FileNameExtensionFilter filter = new FileNameExtensionFilter("*.IMAGE", "jpg", "gif", "png");
57                 fileChooser.addChoosableFileFilter(filter);
58                 int result = fileChooser.showSaveDialog(null);
59                 if(result==fileChooser.APPROVE_OPTION) {
60                     File selectedFile = fileChooser.getSelectedFile();
61                     String path = selectedFile.getAbsolutePath();
62                     label.setIcon(ResizeImage(path));
63                     s = path;
64                 }else if(result==JFileChooser.CANCEL_OPTION) {
65                     System.out.println("Pas de données");
66                 }
67             }
68         });
69     }
70 }

```

Dans cette première partie du code, nous avons défini les variables des boutons, des champs de texte, etc...

Par la suite, nous avons commencé à créer des boutons avec leurs noms (par exemple : ajouter, supprimer, parcourir) et leurs attribuer des tailles. Et pareil pour les champs de texte.

Après, nous avons géré l'événement pour le bouton "parcourir" avec "ActionListener". Puis nous avons défini les extensions des images.

```

77     }
78     });
79
80
81     btnAjouter.addActionListener(new ActionListener() {
82     @Override
83     public void actionPerformed(ActionEvent e) {
84         try {
85             Connection con = DriverManager.getConnection("jdbc:mysql://localhost/db_images","root","root");
86             PreparedStatement ps= con.prepareStatement("insert into monimage(id,nom,description,image)values(?,?,?,?)");
87             InputStream is= new FileInputStream(new File(s));
88
89             ps.setString(1,txtId.getText());
90             ps.setString(2,textNom.getText());
91             ps.setString(3,area.getText());
92             ps.setBlob(4,is);
93             ps.executeUpdate();
94             JOptionPane.showMessageDialog(null, "Engistrement effectué");
95         }catch (Exception ex) {
96             ex.printStackTrace();
97         }
98     }
99
100     }
101     });
102
103     this.add(label);
104     this.add(txtId);
105     this.add(textNom);
106     this.add(pane);
107     this.add(btnAjouter);
108     this.add(btnParcourir);
109     this.add(btnSupprimer);
110     this.setLayout(null);
111     this.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
112     this.setSize(700, 420);
113     this.setVisible(true);
114 }
115
116 public ImageIcon ResizeImage(String imgPath) {
117     ImageIcon MyImage = new ImageIcon(imgPath);
118     Image img = MyImage.getImage();
119     Image newImage = img.getScaledInstance(label.getWidth(),label.getHeight(), Image.SCALE_SMOOTH);
120     ImageIcon image = new ImageIcon(newImage);
121     return image;
122 }
123
124
125 public static void main(String[] args) {
126     new PhotoDB();
127 }
128 }
129

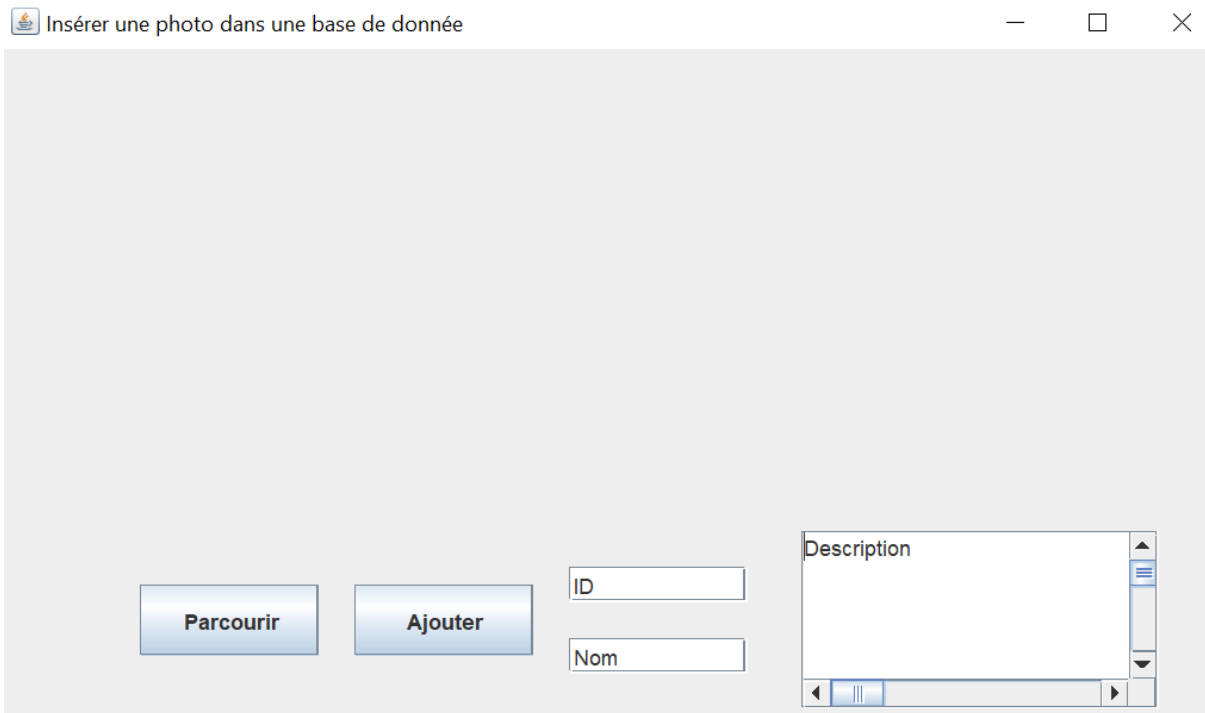
```

Dans cette deuxième partie du code, nous avons géré l'événement pour le bouton "Ajouter".

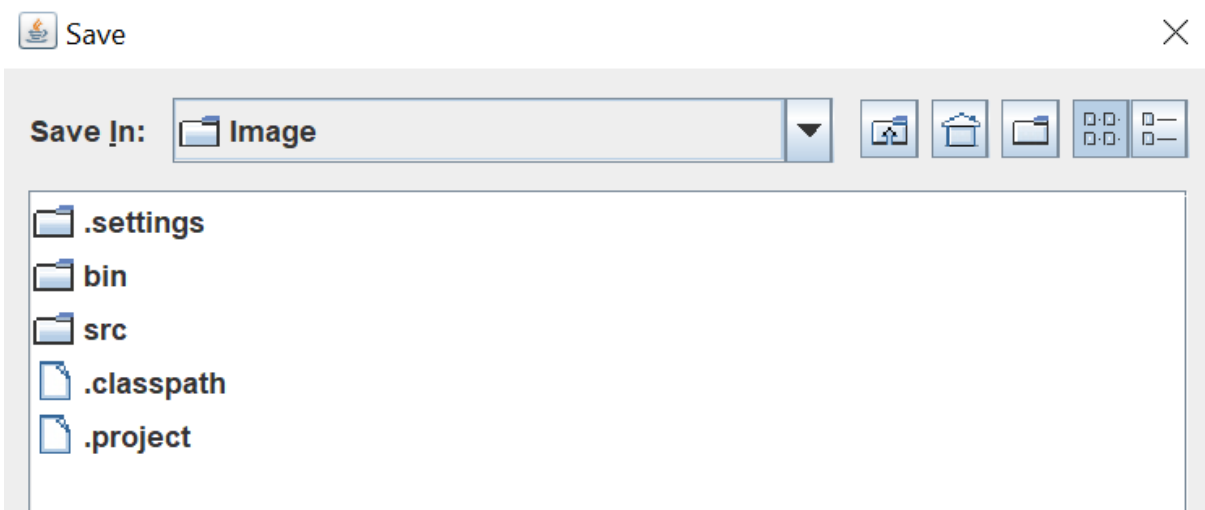
Par la suite, nous avons fait la connexion avec la base de données que nous avons créé précédemment notamment avec la requête "INSERT" qui va introduire les descriptions et les id de nos futures images dans la base de données.

Ensuite nous avons défini la taille de nos images.

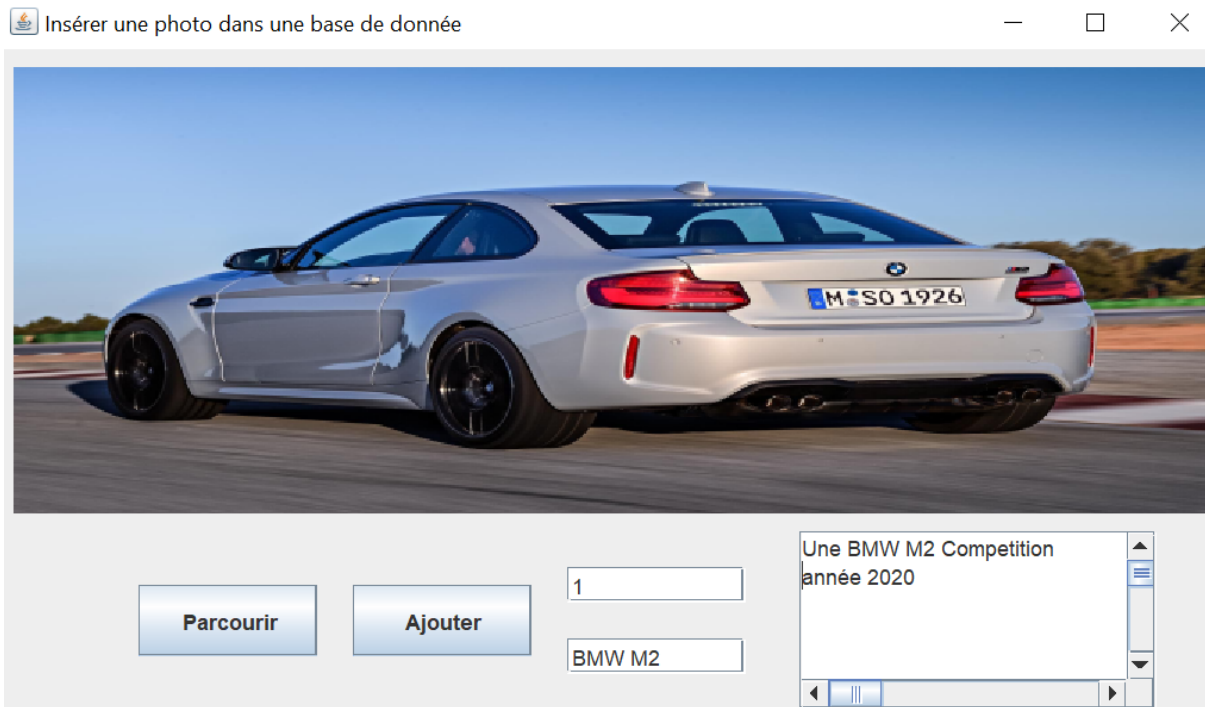
Voici le résultat :



Appuyons sur "Parcourir" :



Allons chercher nos images :



Maintenant que nous avons l'image, nous allons l'ajouter dans la base de donnée :

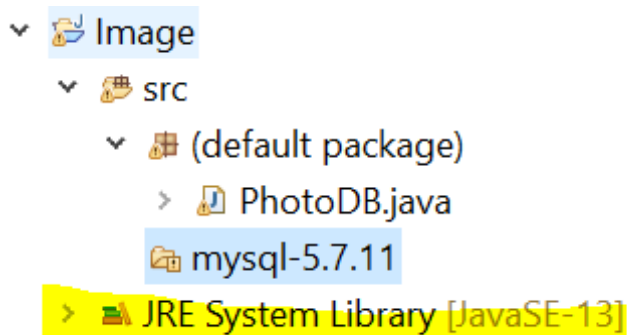
Oups... la BMW ne veut pas s'ajouter quand j'appuie sur le bouton !

Voyons ce qu'on peut faire !

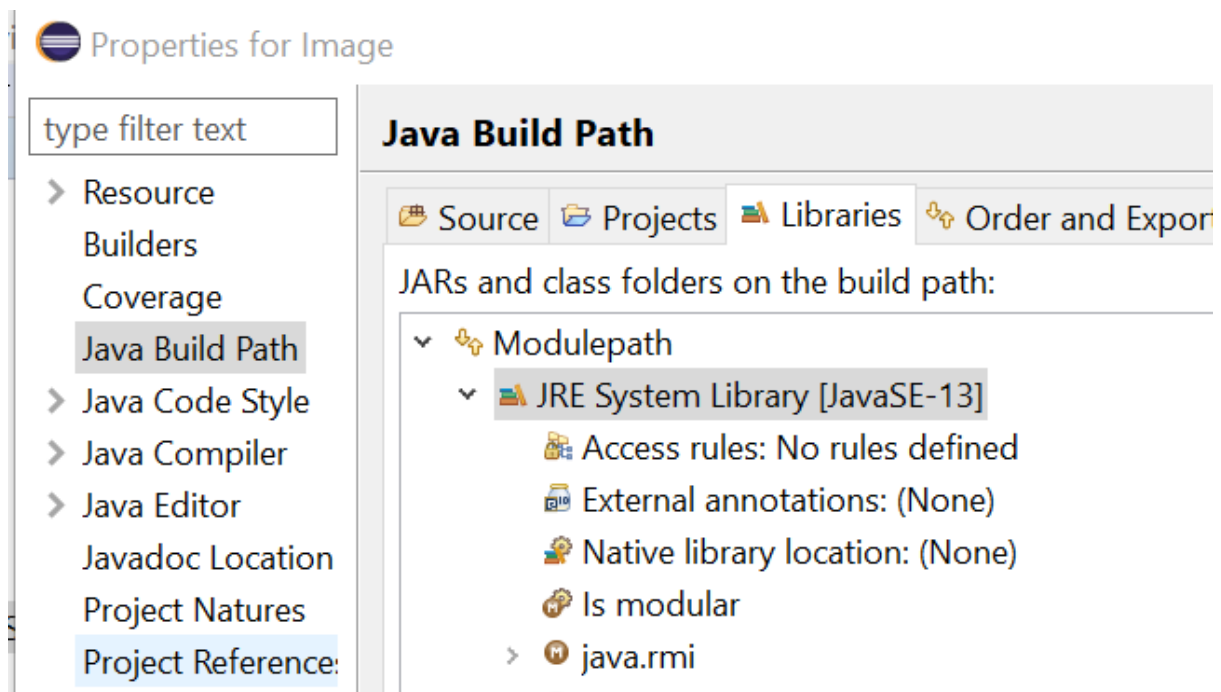
```
Console
PhotoDB [Java Application] C:\Program Files\Java\jdk-13.0.2\bin\javaw.exe (10 déc. 2020 à 21:15:15)
at java.desktop/java.awt.EventQueue$5.run(EventQueue.java:745)
at java.desktop/java.awt.EventQueue$5.run(EventQueue.java:743)
at java.base/java.security.AccessController.doPrivileged(AccessController.java:391)
at java.base/java.security.ProtectionDomain$JavaSecurityAccessImpl.doIntersectionPrivilege
at java.desktop/java.awt.EventQueue.dispatchEvent(EventQueue.java:742)
at java.desktop/java.awt.EventDispatchThread.pumpOneEventForFilters(EventDispatchThread.j
at java.desktop/java.awt.EventDispatchThread.pumpEventsForFilter(EventDispatchThread.java
at java.desktop/java.awt.EventDispatchThread.pumpEventsForHierarchy(EventDispatchThread.j
at java.desktop/java.awt.EventDispatchThread.pumpEvents(EventDispatchThread.java:109)
at java.desktop/java.awt.EventDispatchThread.pumpEvents(EventDispatchThread.java:101)
at java.desktop/java.awt.EventDispatchThread.run(EventDispatchThread.java:90)
java.sql.SQLException: No suitable driver found for jdbc:mysql://localhost/db_images
at java.sql/java.sql.DriverManager.getConnection(DriverManager.java:702)
at java.sql/java.sql.DriverManager.getConnection(DriverManager.java:228)
at PhotoDB$2.actionPerformed(PhotoDB.java:85)
at java.desktop/javaw.swing.AbstractButton.fireActionPerformed(AbstractButton.java:1967)
at java.desktop/javaw.swing.AbstractButton$Handler.actionPerformed(AbstractButton.java:23
at java.desktop/javaw.swing.DefaultButtonModel.fireActionPerformed(DefaultButtonModel.jav
at java.desktop/javaw.swing.DefaultButtonModel.setPressed(DefaultButtonModel.java:262)
```

Nous avons toute une liste d'erreurs ! Pour cela nous devons ajouter le connecteur dans la librairie comme ci-dessous :

Nous devons dans un premier temps aller à la librairie comme ci-dessous :



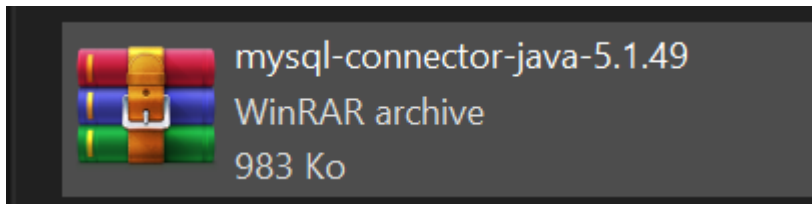
Ensuite nous allons sur "Built path" en faisant clic droit comme ci-dessous :



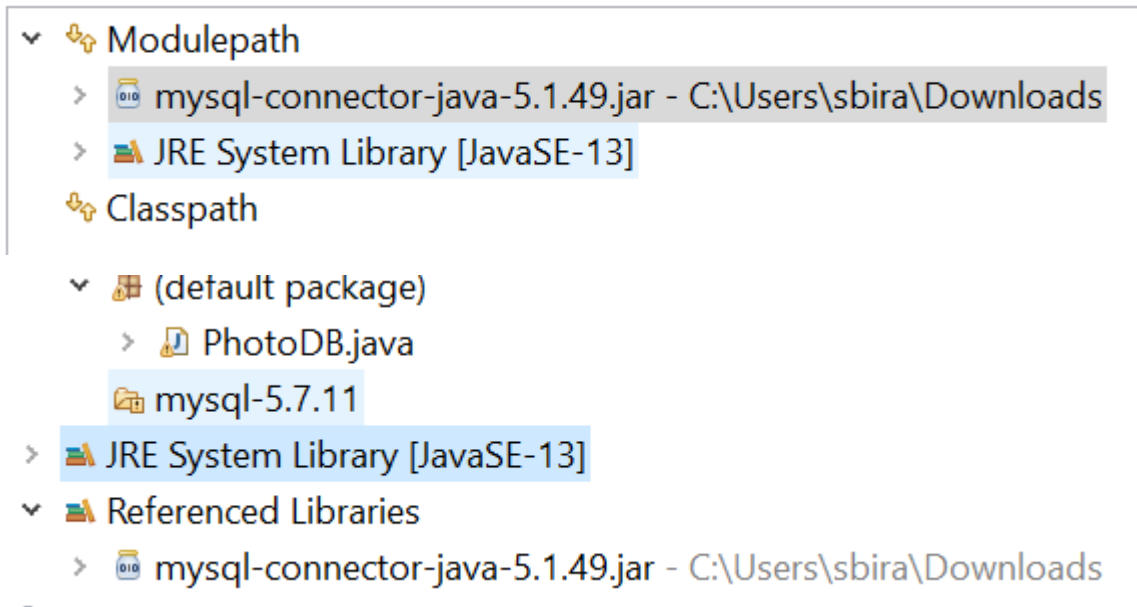
On appuis sur "module" et on clique sur "add External JAR's" comme ci-dessous :



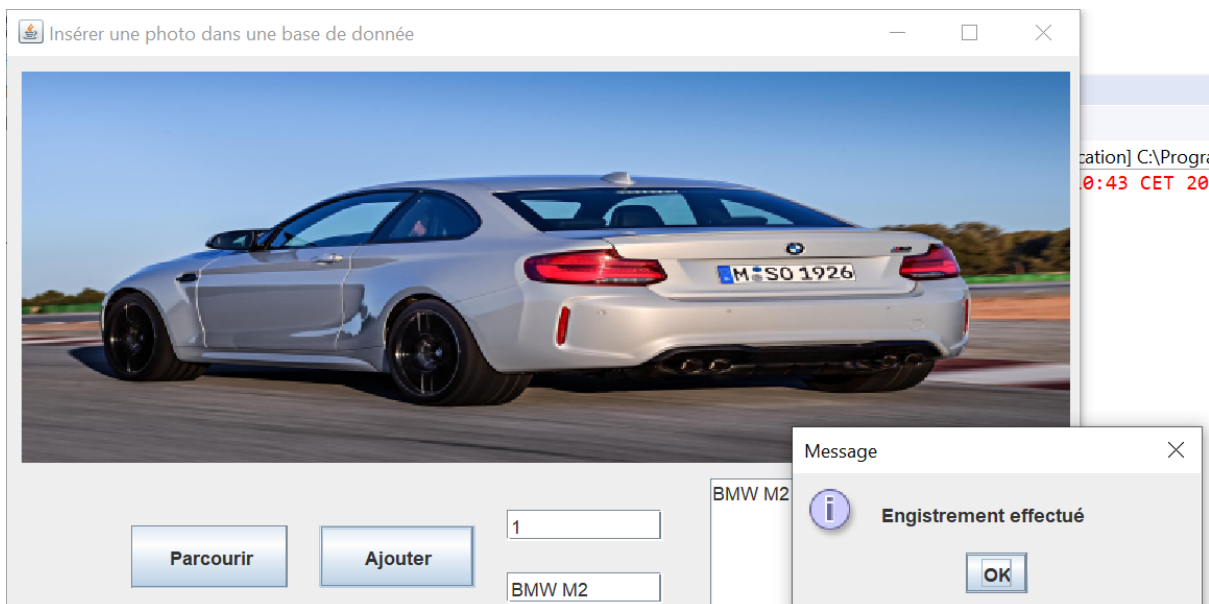
Et ensuite on met le connecteur :



Voici une fois ajouté :






Maintenant cliquons "apply and close" et allons tester à nouveau si ça fonctionne :



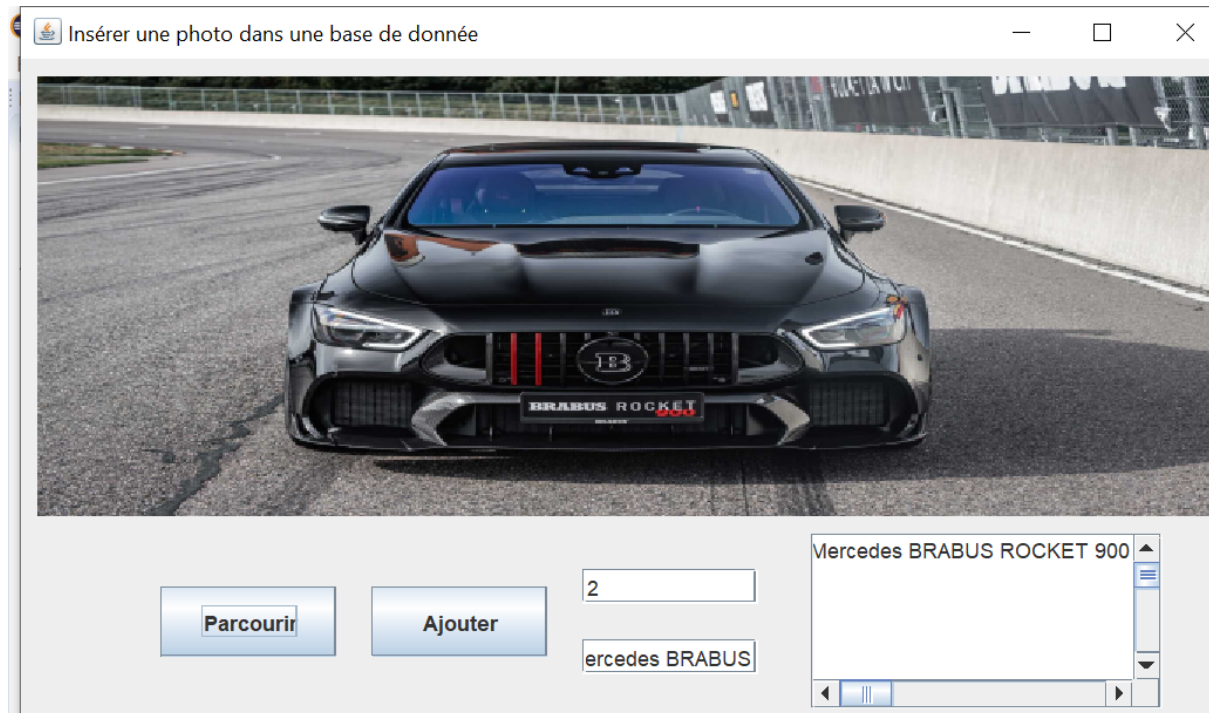
Ca fonctionne la M2 a repris son chemin vers la base de donnée avec le message de confirmation. Mais voyons si elle est bien arrivée à destination (base de donnée "db_images") :

+ Options

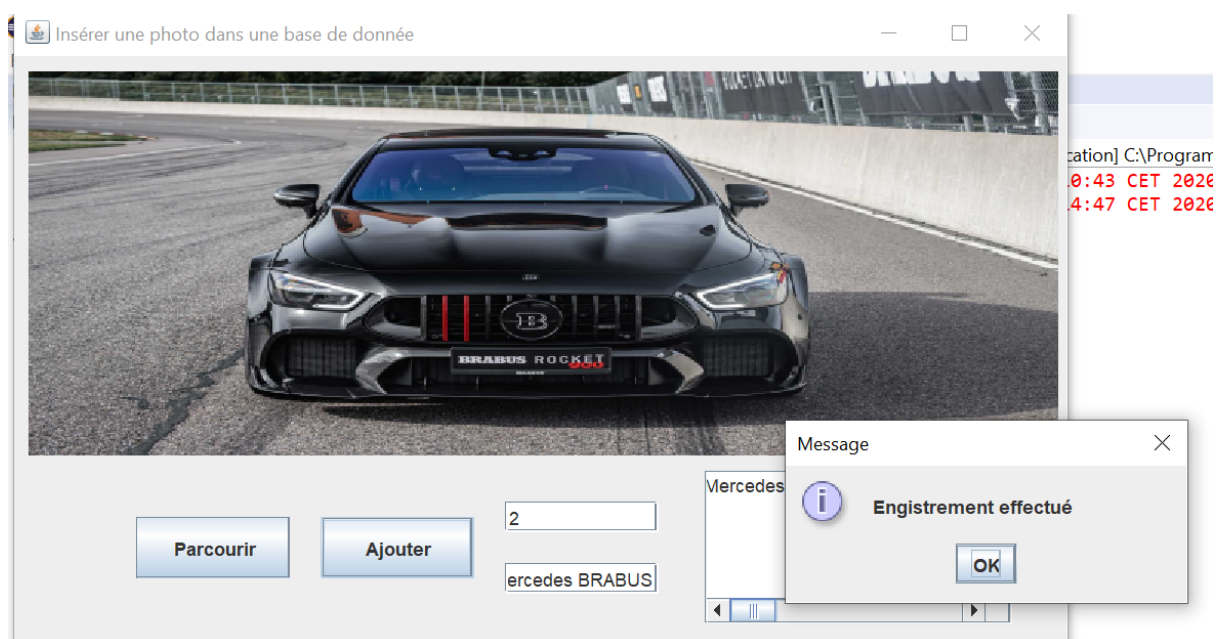
	id	nom	description	image
<input type="checkbox"/>    1	BMW M2	BMW M2 Competition	[BLOB - 169,3 Kio]	

La voiture est arrivée à destination.







Testons pour une autre image...



Testons !



Regardons la base de donnée :

<input type="checkbox"/>	 Modifier	 Copier	 Effacer	1	BMW M2	BMW M2 Competition	[BLOB - 169,3 Kio]
<input type="checkbox"/>	 Modifier	 Copier	 Effacer	2	Mercedes BRABUS	Mercedes BRABUS ROCKET 900	[BLOB - 924,8 Kio]

Elle aussi est arrivée à destination !

Les éléments ont été ajoutés correctement à la base de donnée avec le id, le nom et la description.