



Prise en main

Du logiciel Eclipse Java

Sommaire

<u>Création d'un projet sous Eclipse</u>	Page N°3
<u>Création de classe et debugage de votre programme</u>	Page N°7
<u>Mise en œuvre du Plugin windows builder</u>	Page N°12
<u>Création d'une archive JAR exécutable</u>	Page N°22
<u>Installation du pilote RXTXCOM</u>	Page N°25
<u>Installation du pilote RXTXCOM sur Raspberry</u>	Page N°29
<u>Intégration de la librairie JFreechart</u>	Page N°30
<u>Réalisation d'un Menu sous WINDOWSBUILDER</u>	Page N°37
<u>Installation de PI4J sous Raspberry PI</u>	Page N°42
<u>Installation du driver JDBC MYSQL</u>	Page N°46
<u>Création d'une librairie JAR</u>	Page N°48
<u>Comment lancer un script au démarrage de la PI</u>	Page N°54
<u>Quelques composant graphiques JAVA</u>	Page N°55

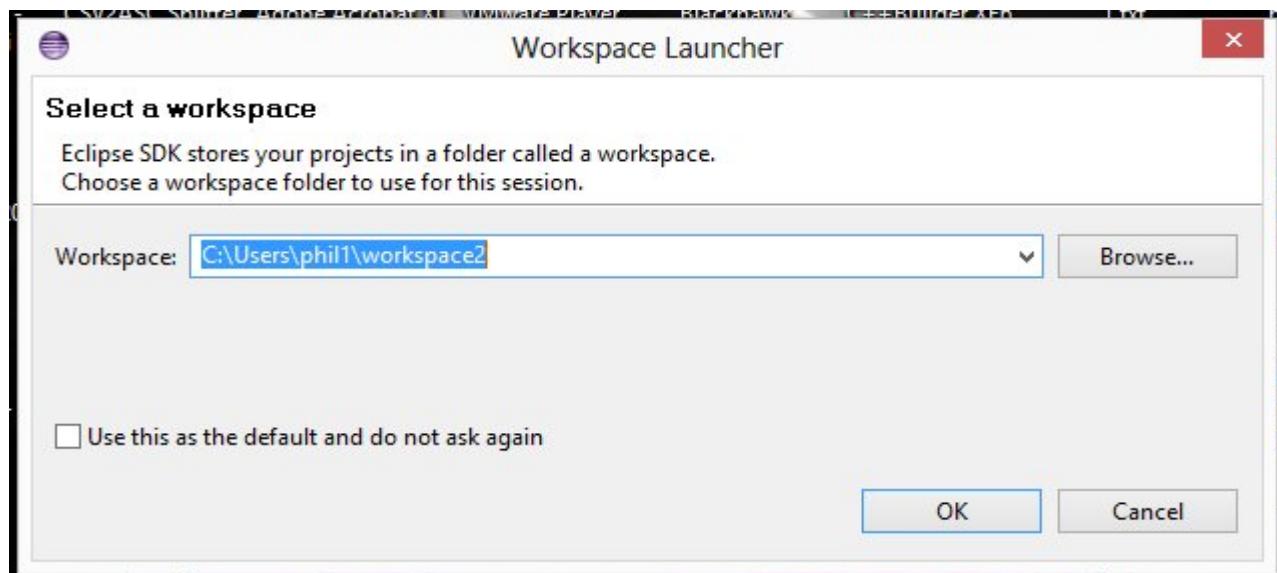
Eclipse est un logiciel qui intègre un environnement de développement pour le langage JAVA. Il permet aussi de développer avec d'autres types de langage PHP , C , C++

1) Création d'un projet sous eclipse

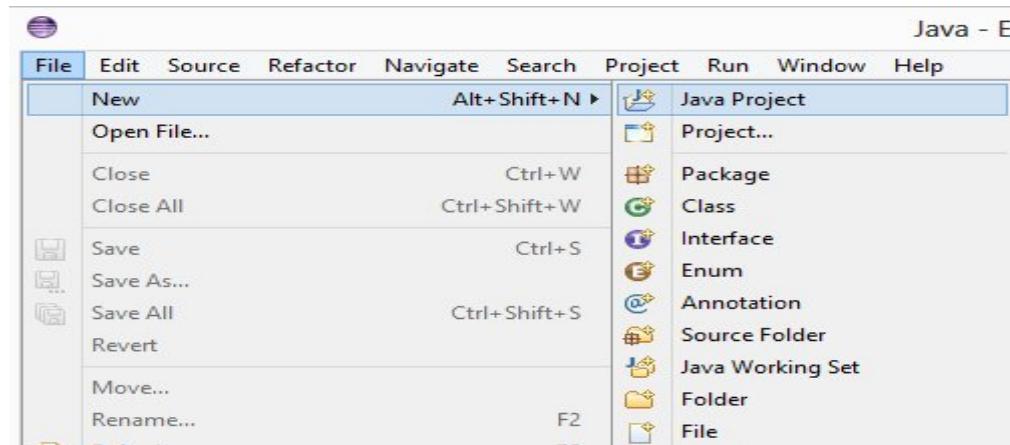
Lancez le programme eclipse java



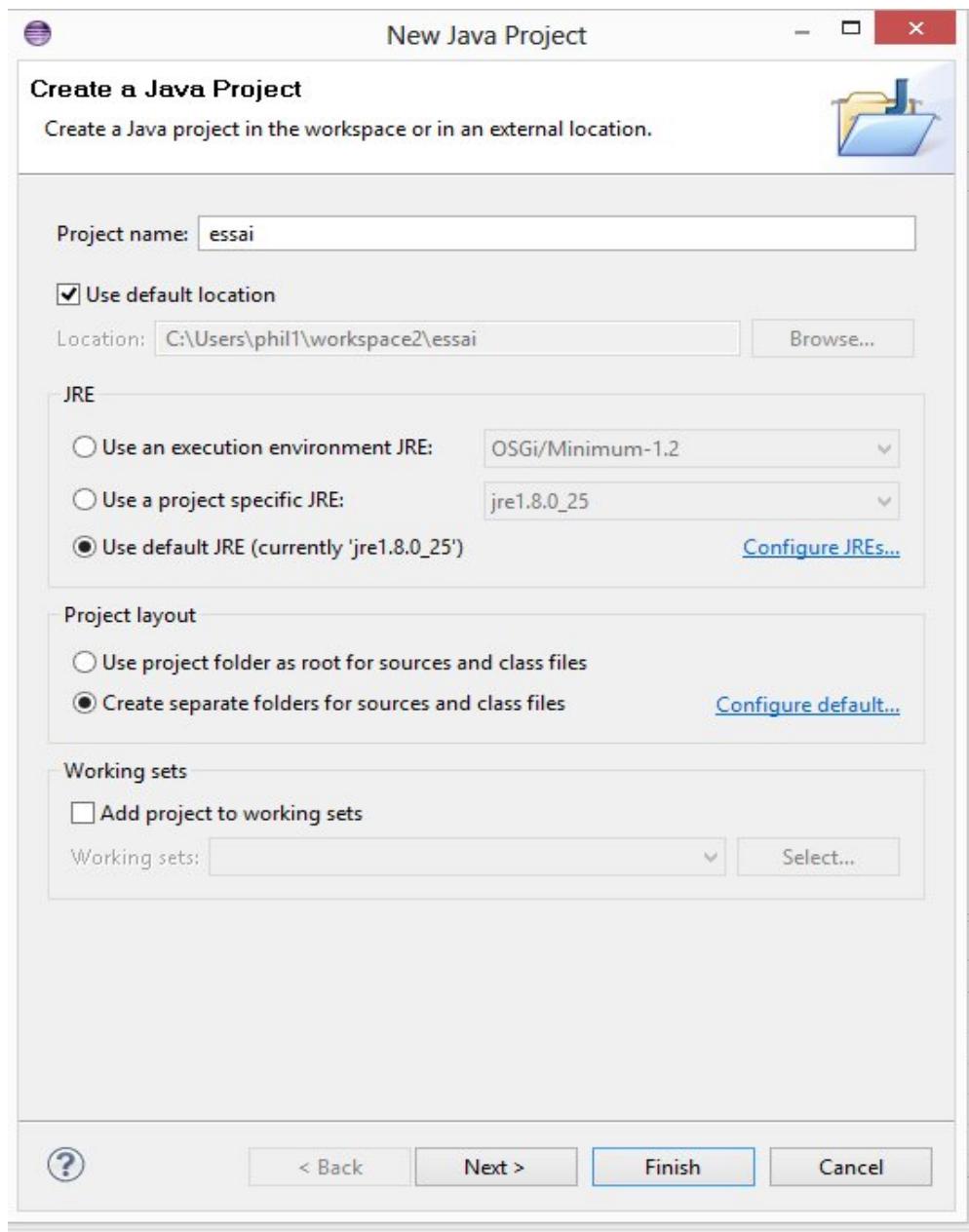
Choisir un espace de travail ou les différents projets seront enregistrés



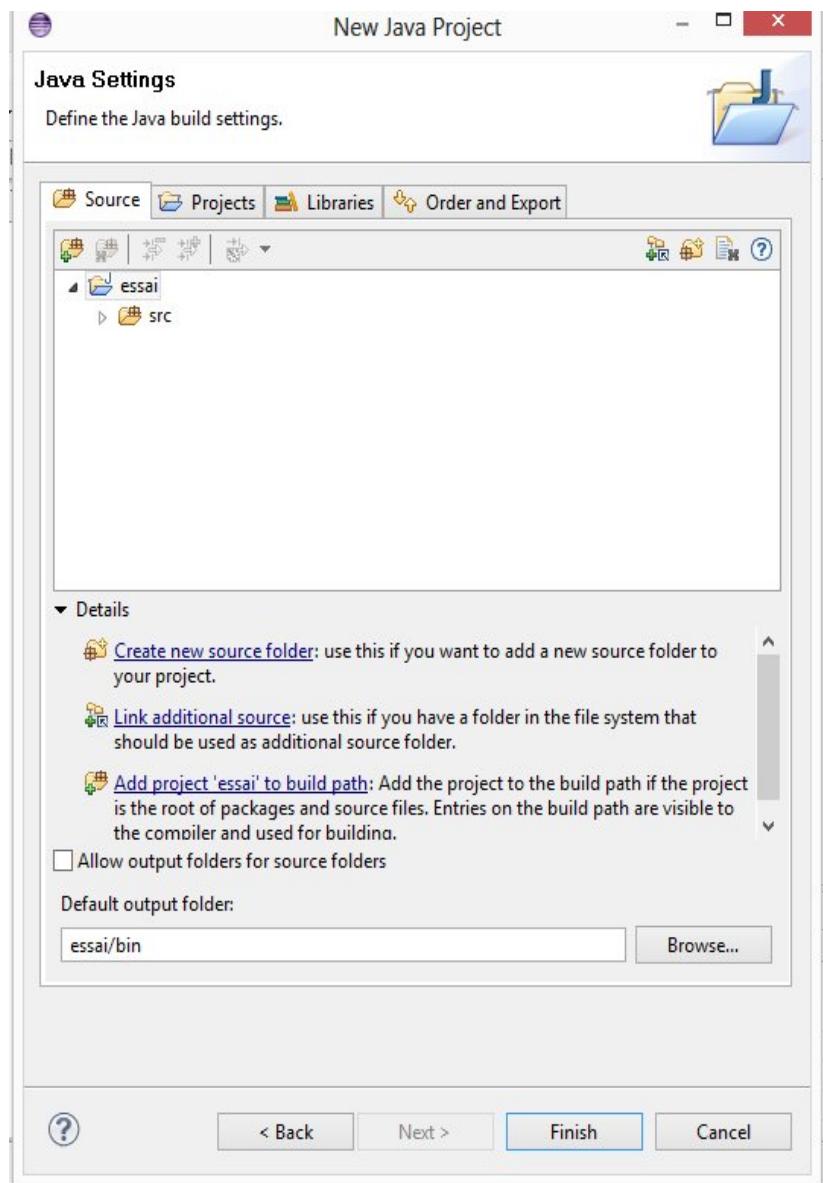
Créez un nouveau projet



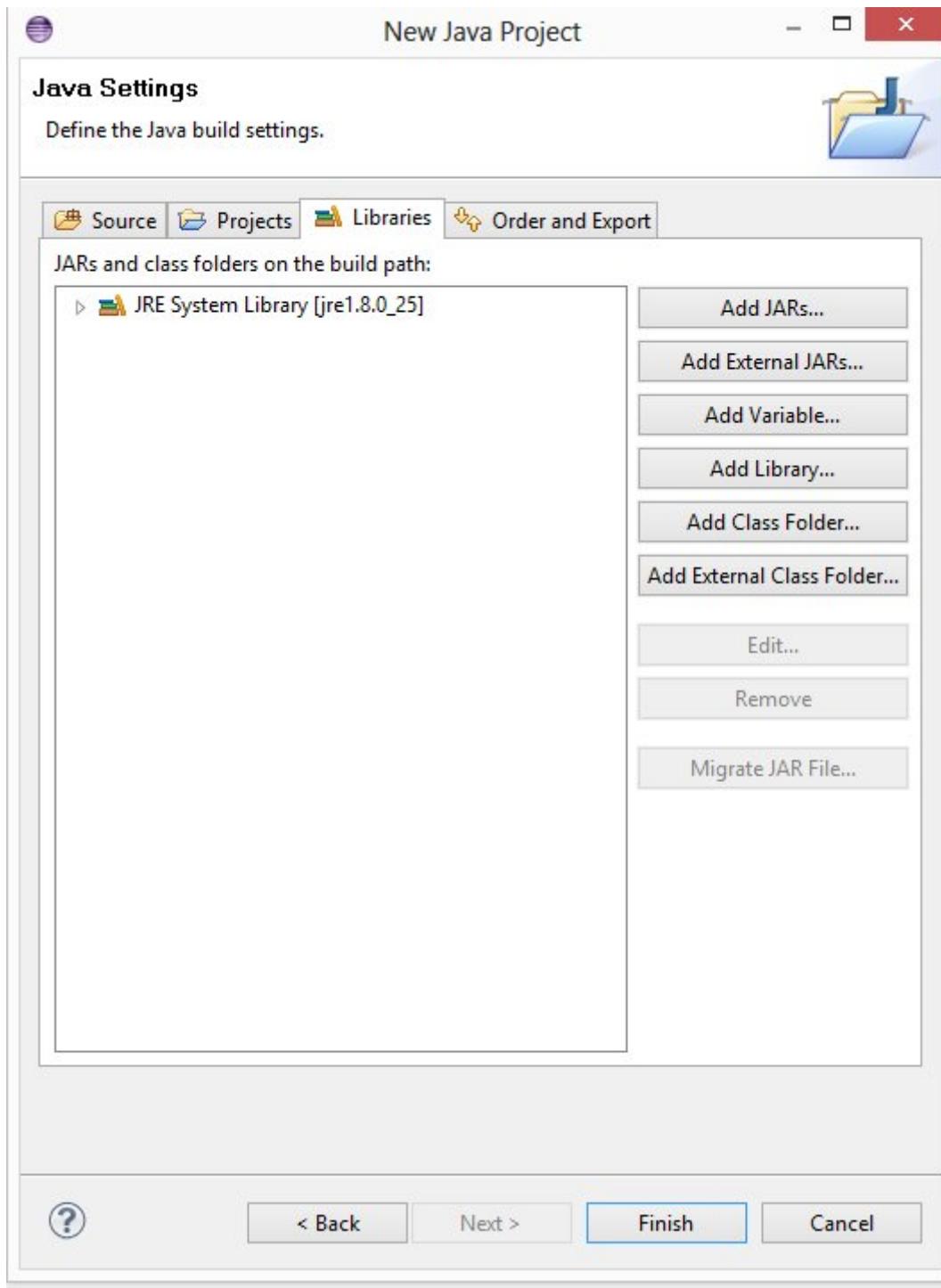
Donnez un nom au projet et choisir la JRE implantée sur la plateforme windows



L'arborescence du projet apparaît :

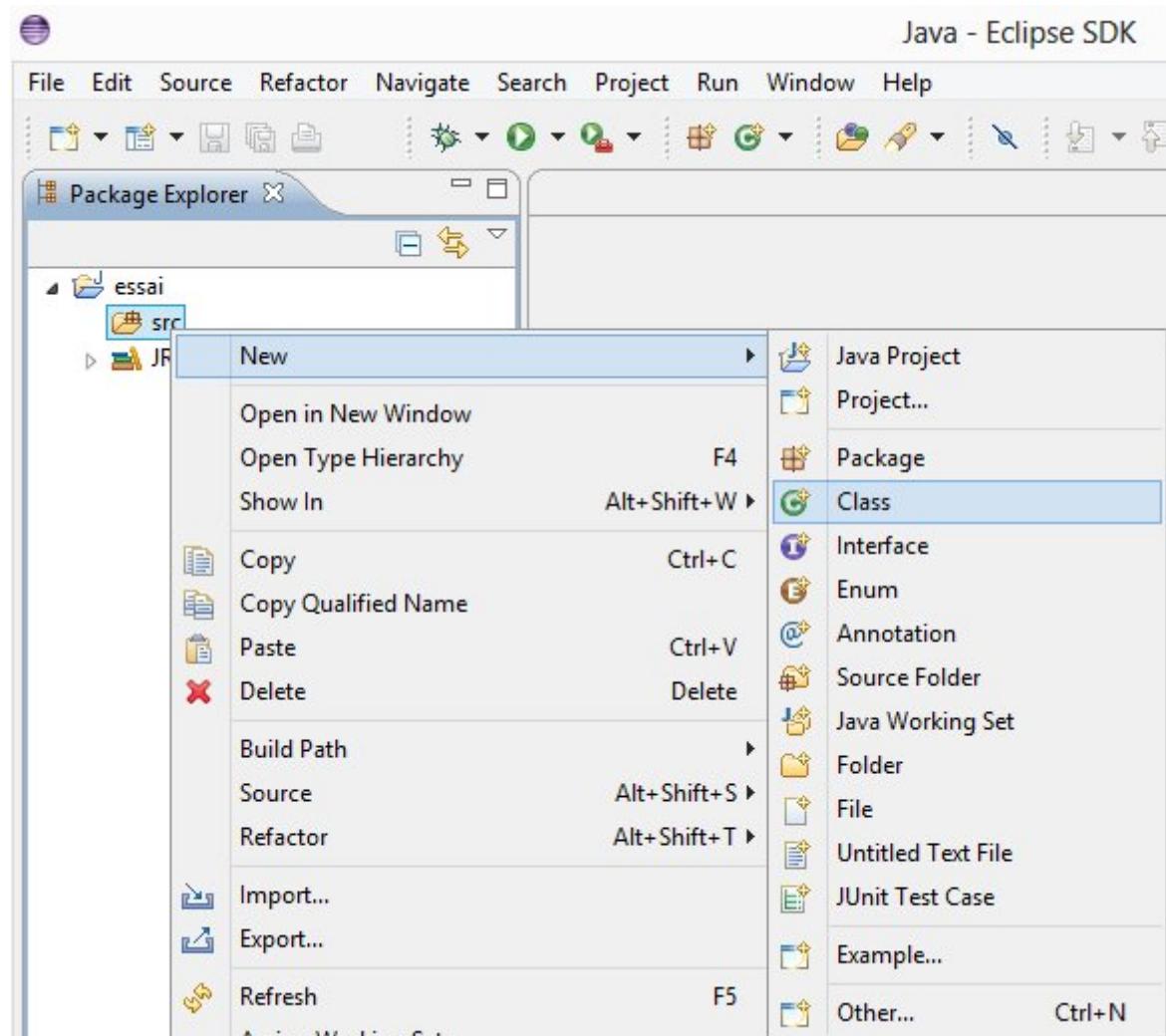


Si des librairies ou des paquetages sont à importer cliquez sur l'onglet libraries

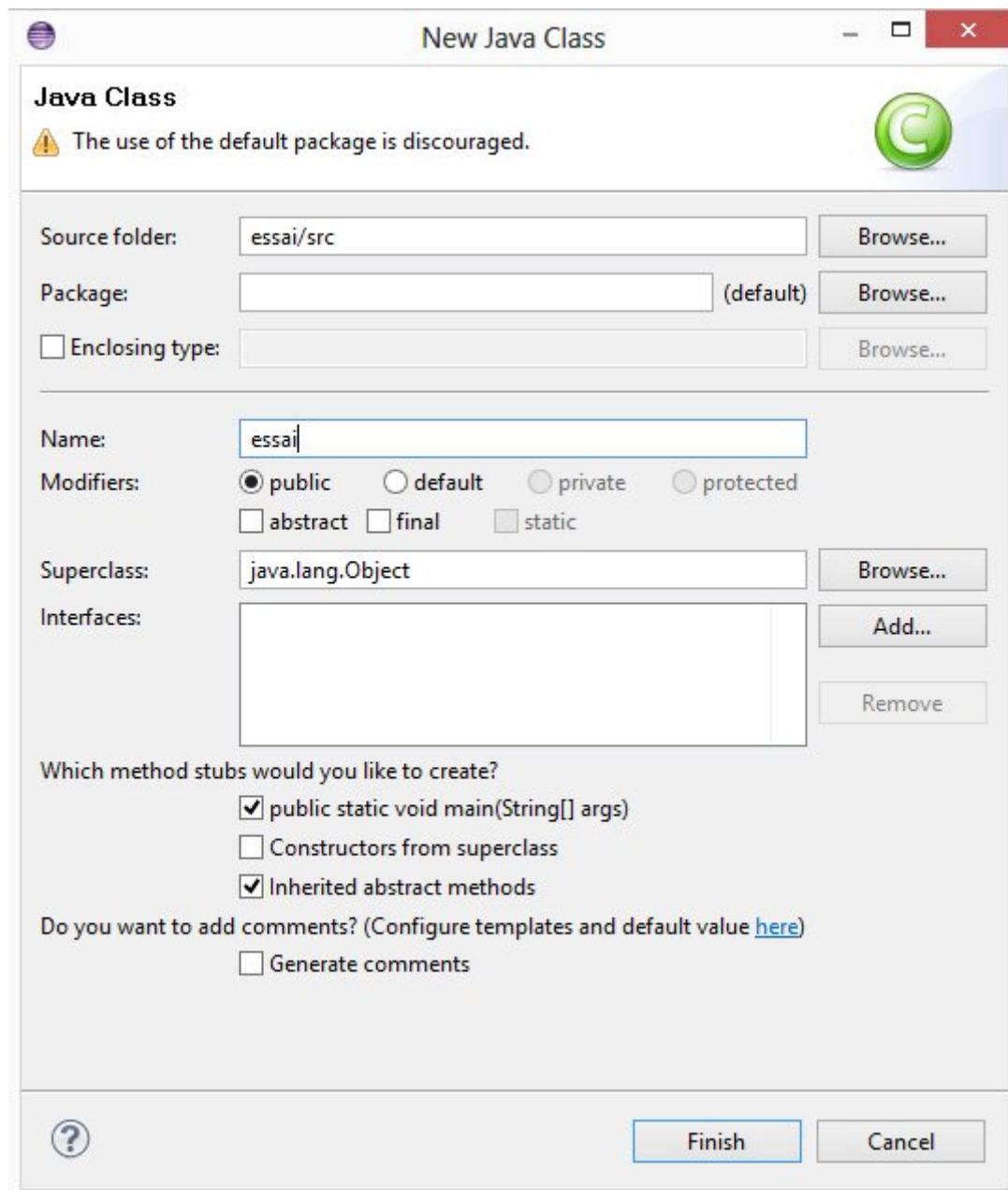


2) Création de classe et débogage de votre programme

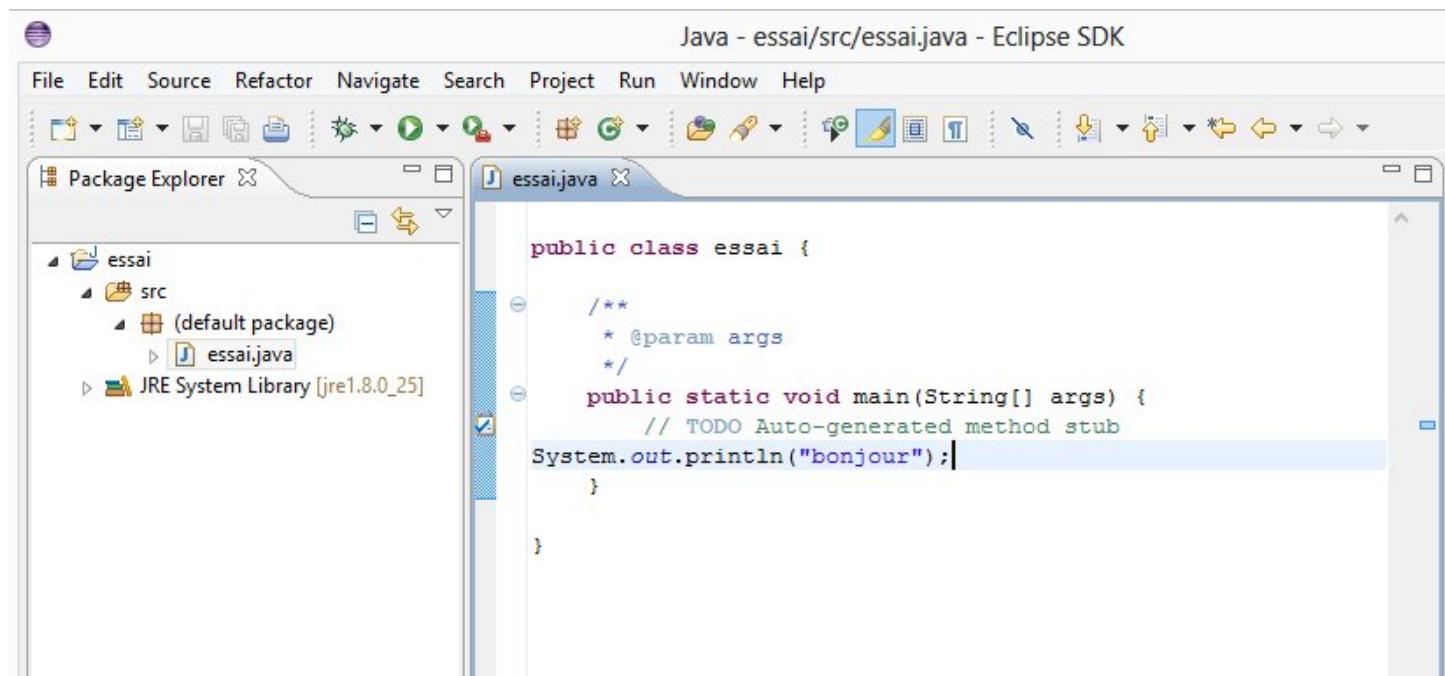
Pour créer une nouvelle classe cliquez droit sur src puis new → Class



Définir le nom et la structure de la nouvelle classe



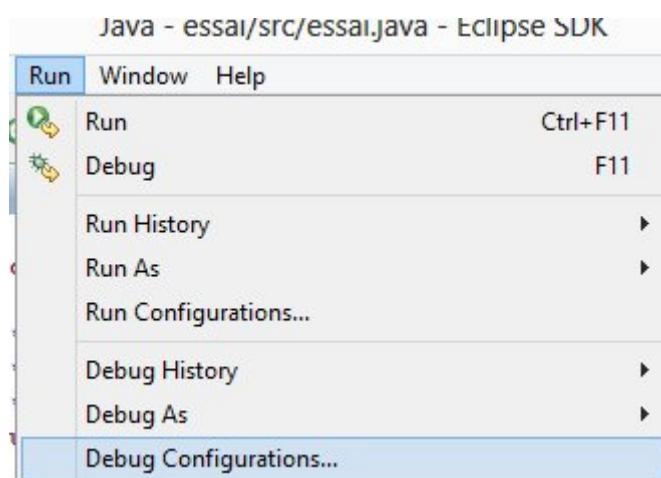
Complétez le code de cette nouvelle classe



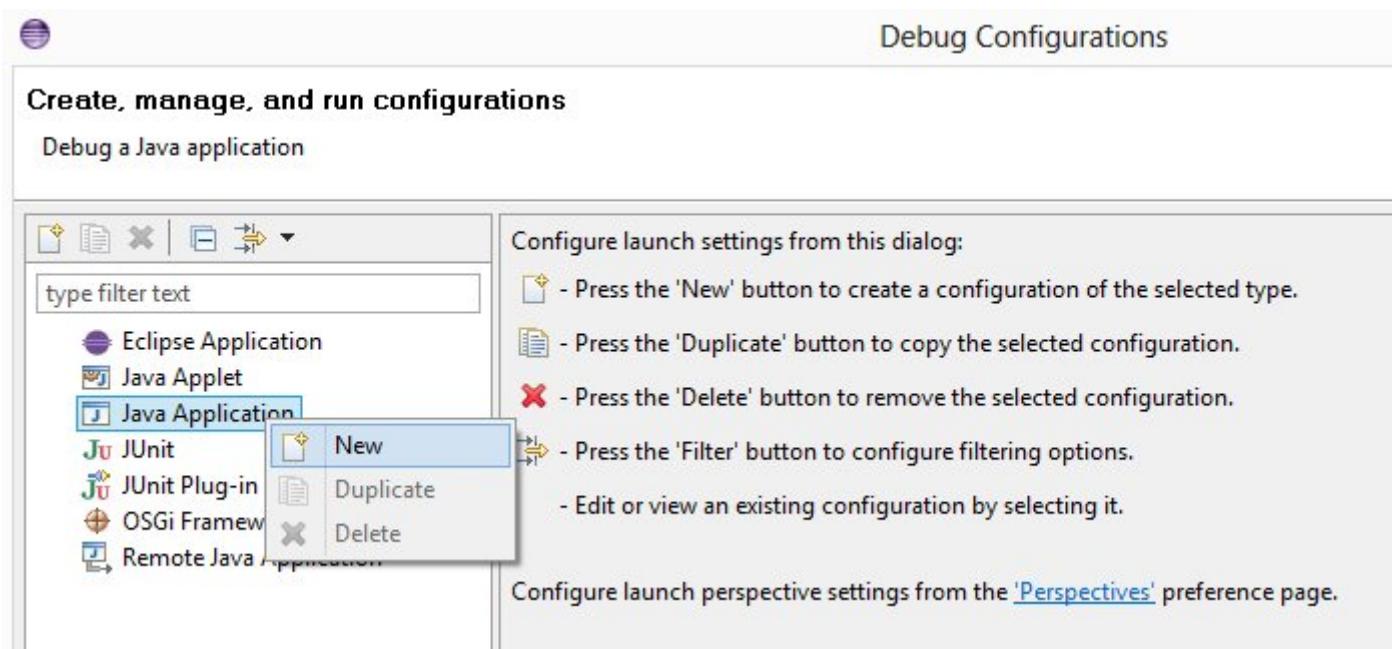
The screenshot shows the Eclipse IDE interface. The title bar reads "Java - essai/src/essai.java - Eclipse SDK". The menu bar includes File, Edit, Source, Refactor, Navigate, Search, Project, Run, Window, and Help. The toolbar has various icons for file operations like Open, Save, and Print. On the left is the "Package Explorer" view showing a project named "essai" with a "src" folder containing a file named "essai.java". The main editor window displays the following Java code:

```
public class essai {  
    /**  
     * @param args  
     */  
    public static void main(String[] args) {  
        // TODO Auto-generated method stub  
        System.out.println("bonjour");  
    }  
}
```

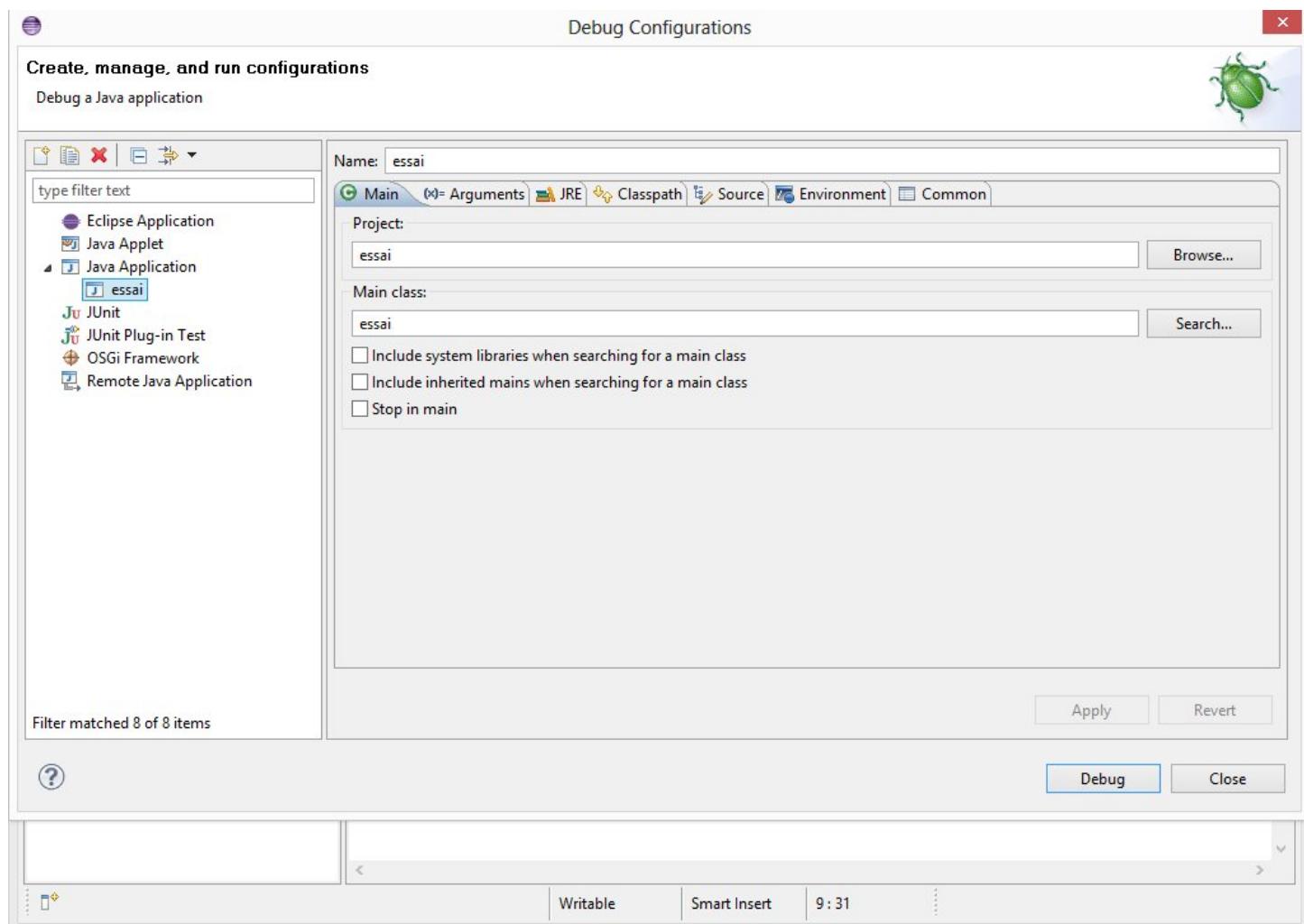
Pour debugger notre programme faire run → Debug configurations



Cliquez droit sur Java Application et faire new

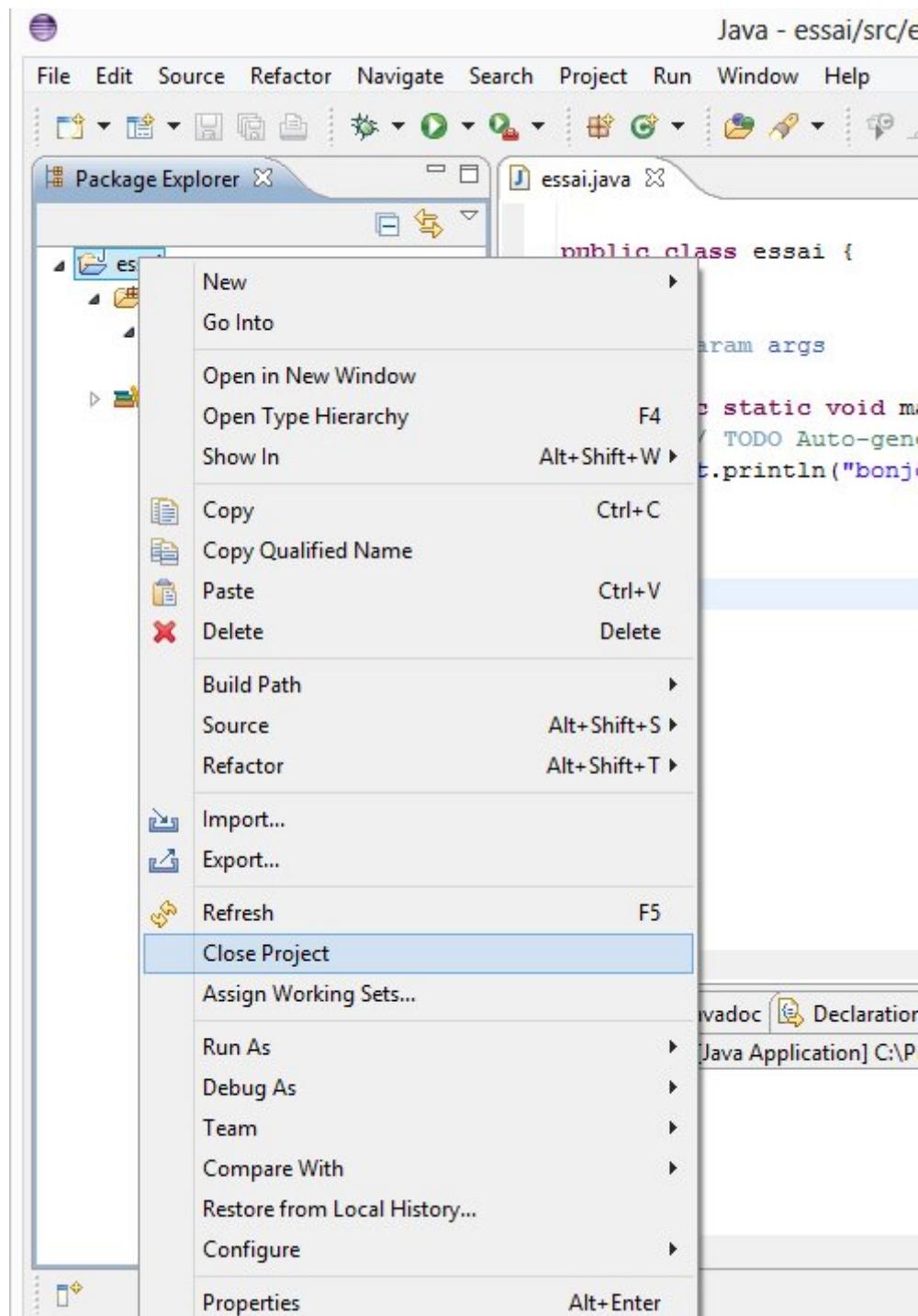


Choisir notre projet et la classe où se situe la méthode statique main (point d'entrée de notre programme)



Cliquez sur le bouton debug pour lancez l'interface de débogage :

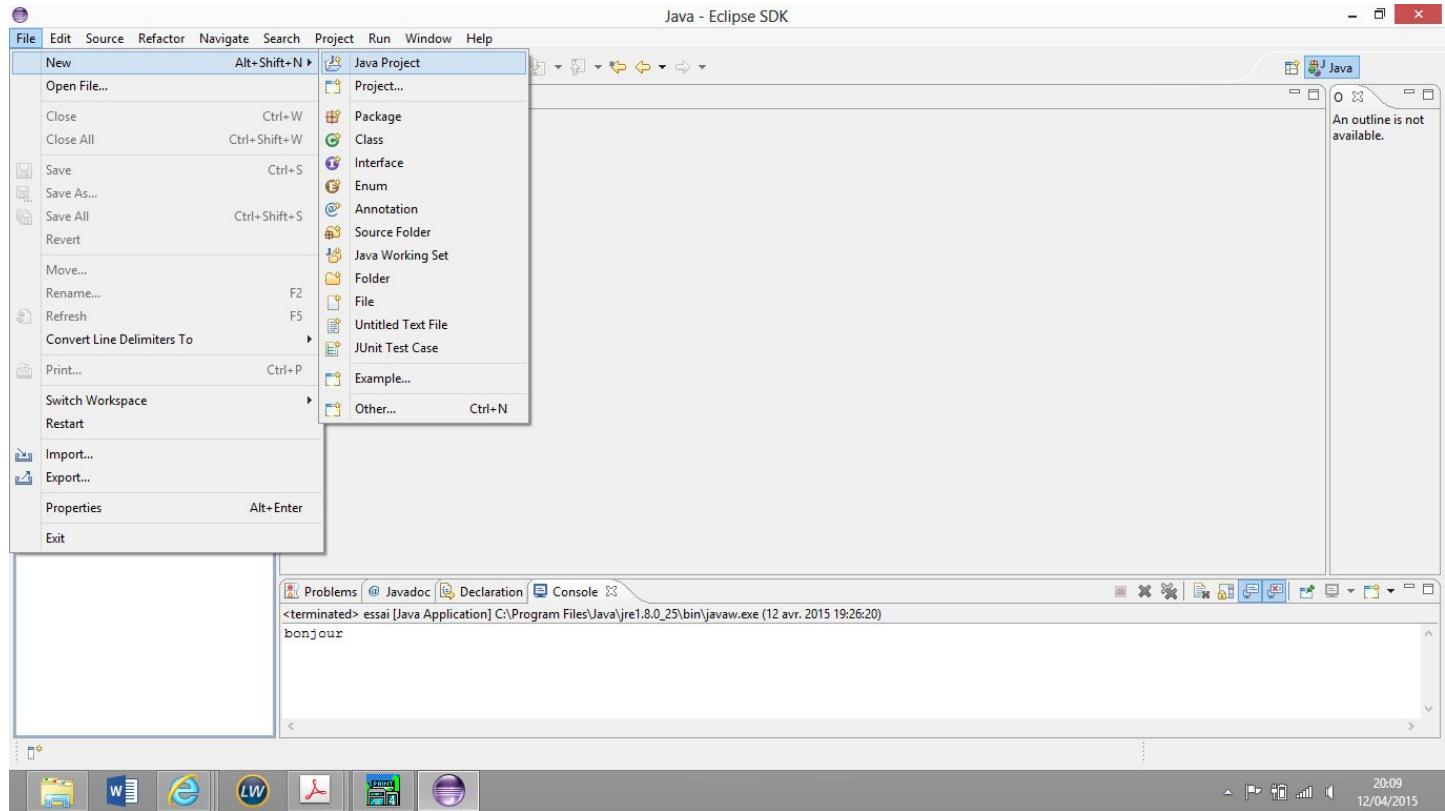
Lorsque l'on souhaite fermer le projet courant, cliquez droit sur le nom du projet et close project



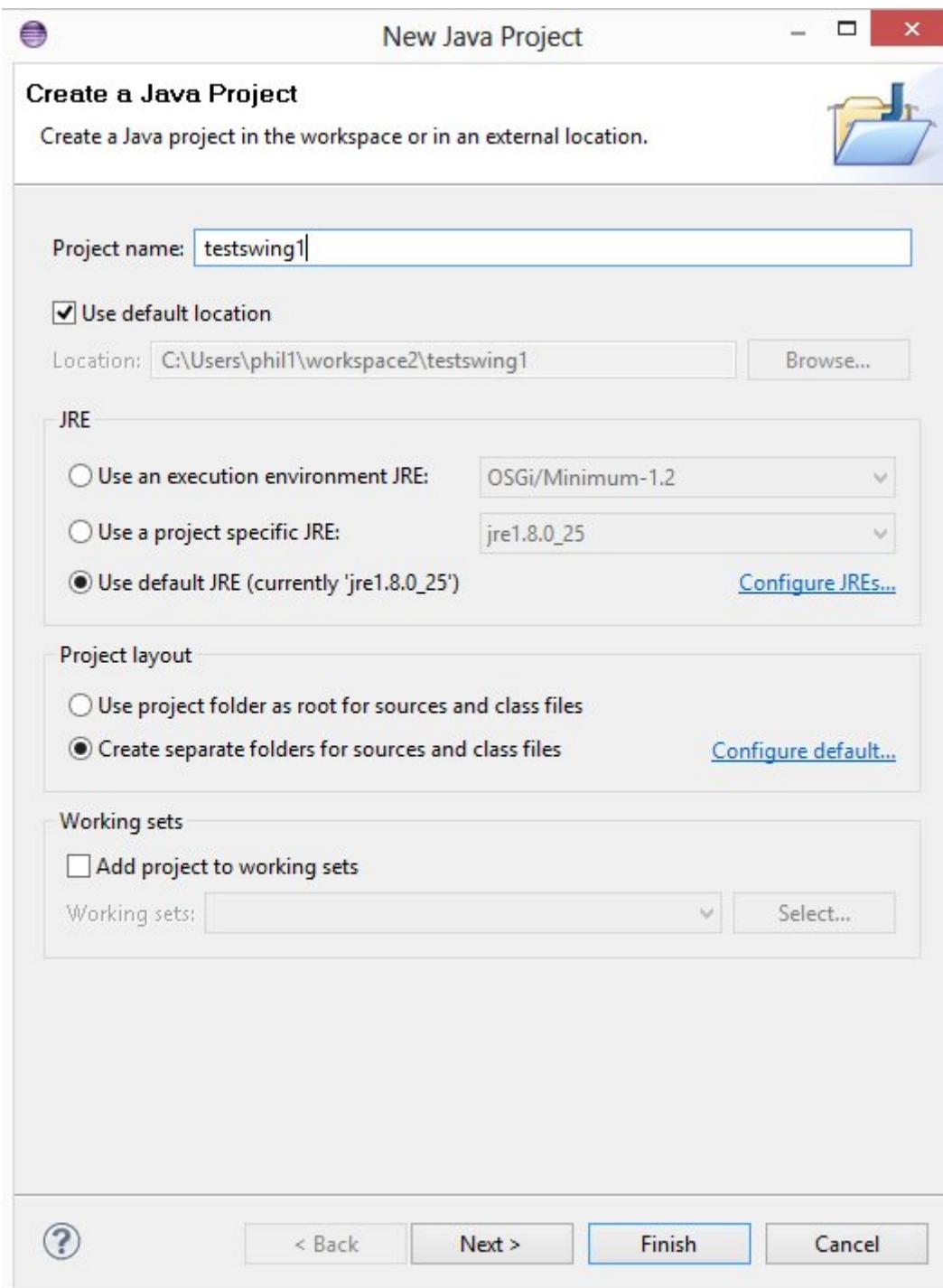
3) Mise en œuvre du Plugin windows builder pour le développement d'applications rapides

Eclipse dispose d'un Plugin qui lui permet de disposer l'un logiciel d'application rapide RAD. Ce plugin a comme nom windows builder que nous allons utiliser par la suite. On suppose au préalable que ce plugin est installé sur eclipse.

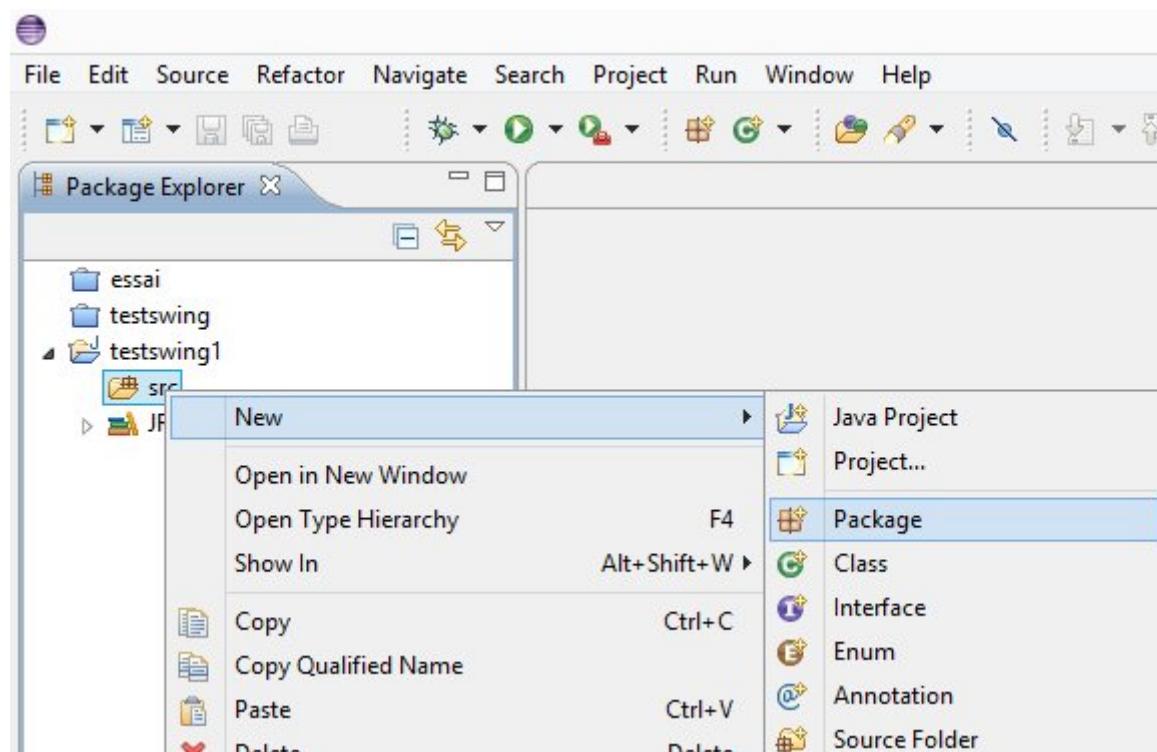
Il faut créer un nouveau projet Java



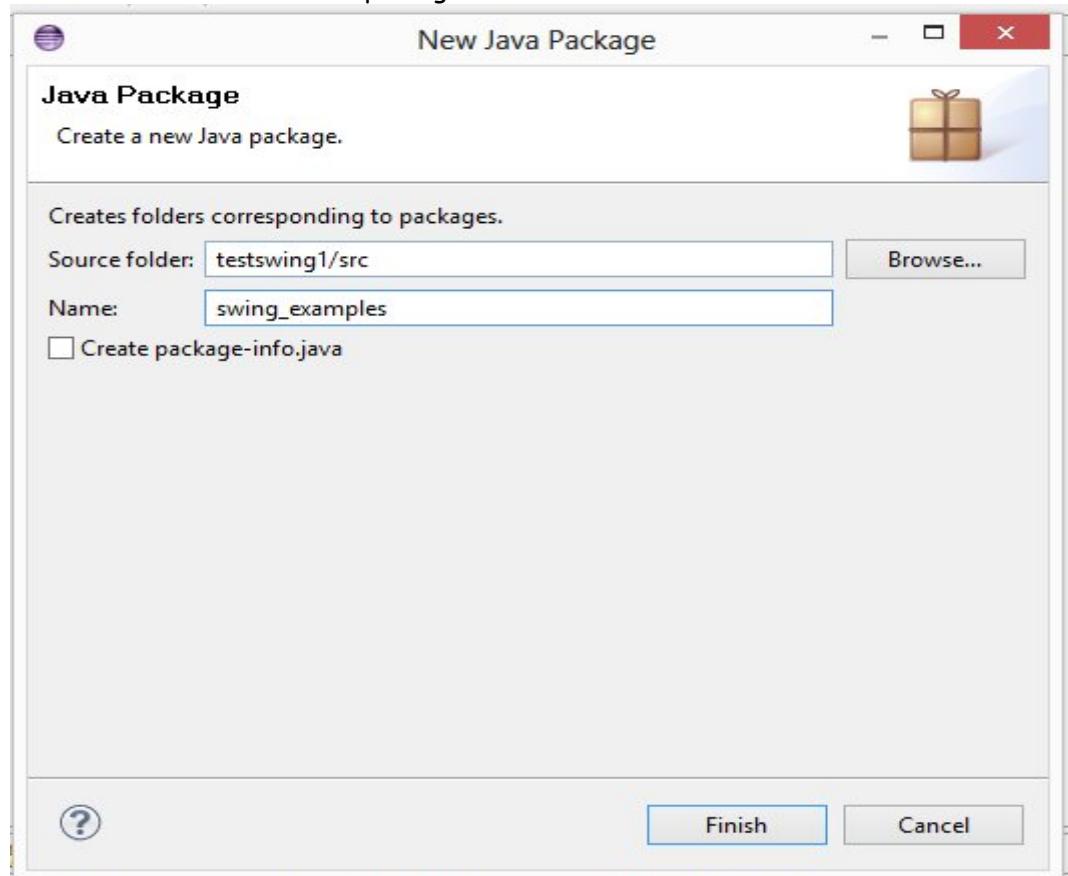
Donnez un nom à ce nouveau projet :



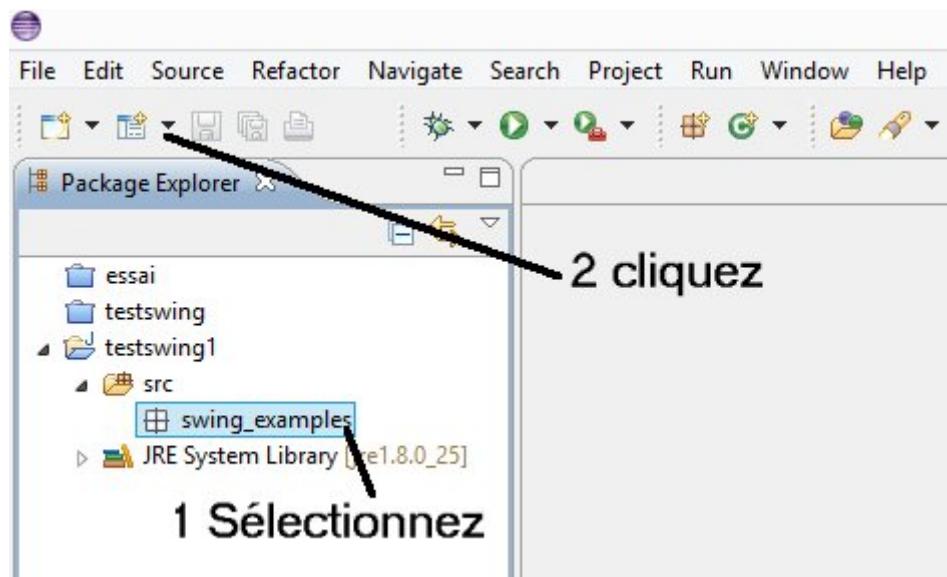
Puis définir un nouveau package



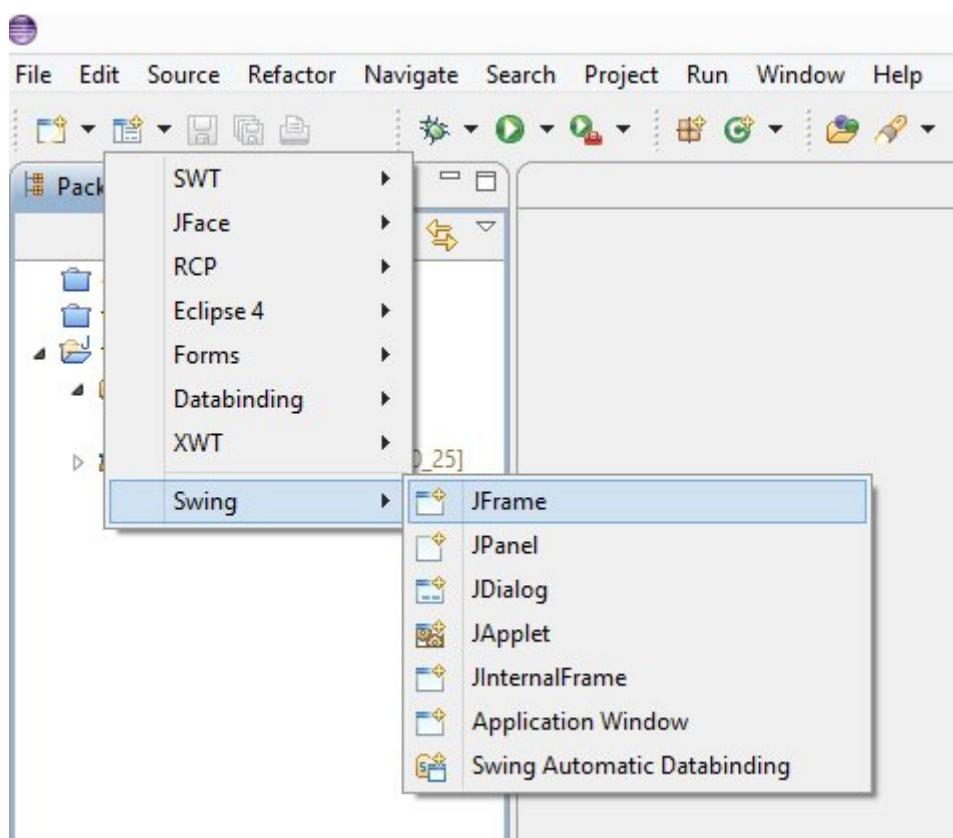
Donnez un nom à ce nouveau package



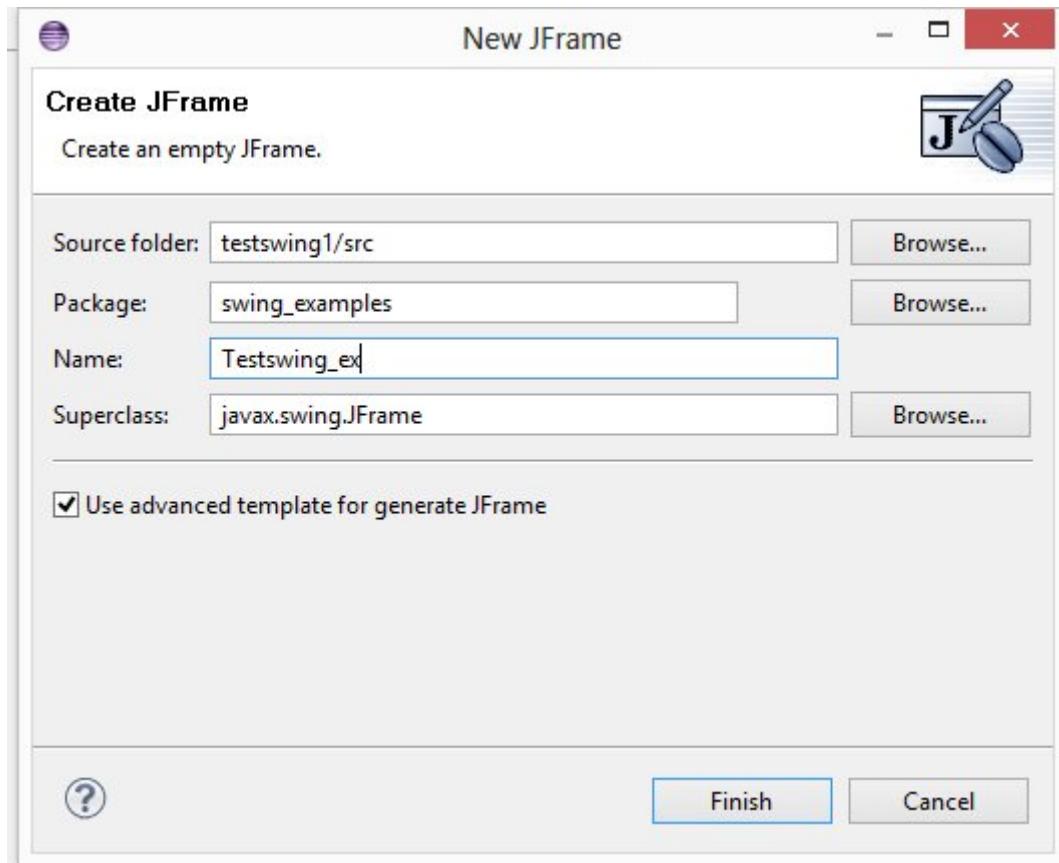
Sélectionnez et cliquez



Choisir un JFrame dans la bibliothèque graphique Swing



Donnez un nom à la classe qui héritera de la classe JFrame



Automatiquement Eclipse génère le code de cette nouvelle Classe

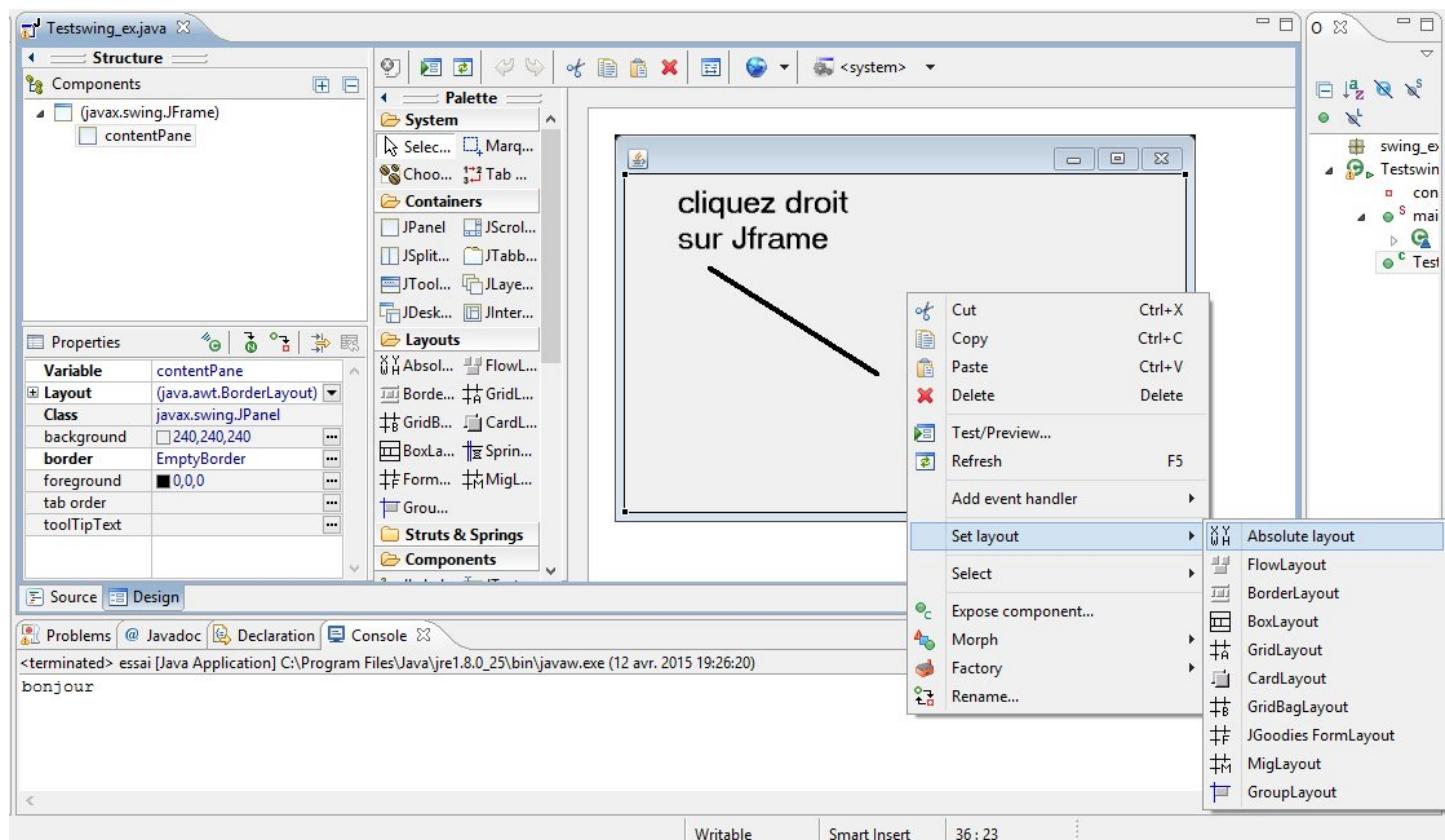
```
Testswing_ex.java X
public static void main(String[] args) {
    EventQueue.invokeLater(new Runnable() {
        public void run() {
            try {
                Testswing_ex frame = new Testswing_ex();
                frame.setVisible(true);
            } catch (Exception e) {
                e.printStackTrace();
            }
        }
    });
}

/**
 * Create the frame.
 */
public Testswing_ex() {
    setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
    setBounds(100, 100, 450, 300);
    contentPane = new JPanel();
    contentPane.setBorder(new EmptyBorder(5, 5, 5, 5));
    contentPane.setLayout(new BorderLayout(0, 0));
    setContentPane(contentPane);
}
```

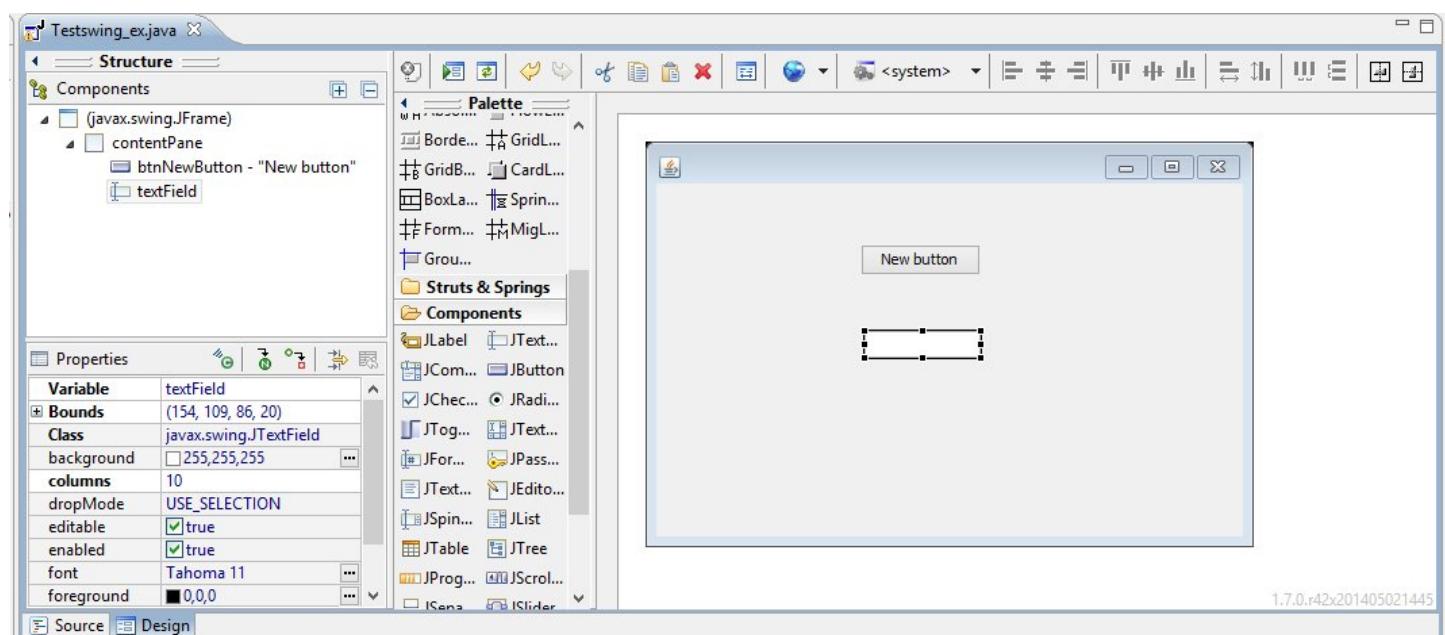
cliquez

Source Design

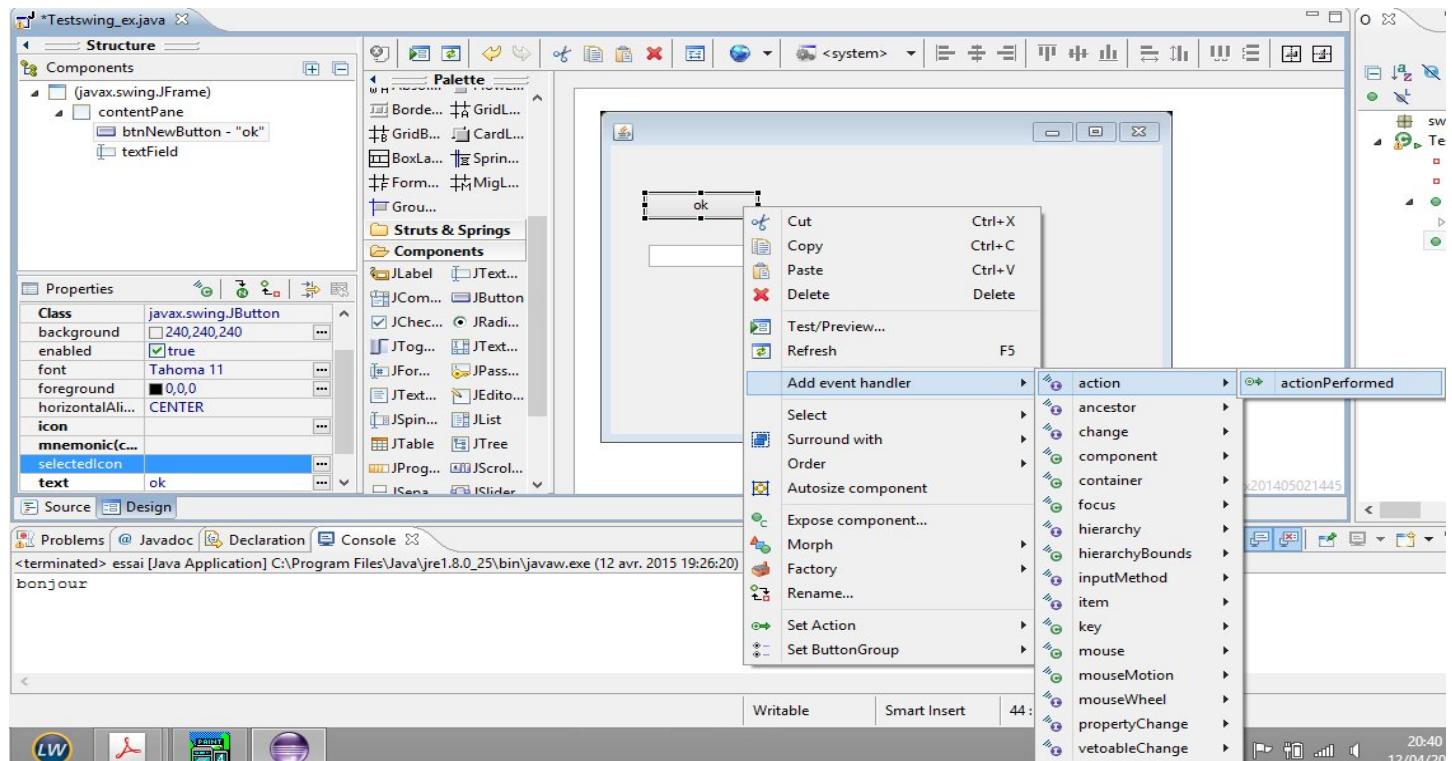
Pour basculer sur l'interface graphique cliquez sur l'onglet design, l'interface de windows builder apparait



Puis placez nos objets graphiques sur l'interface



Pour gérer un événement sur un objet, sélectionnez l'objet cliquez droit puis sélectionnez le type d'événement attendu



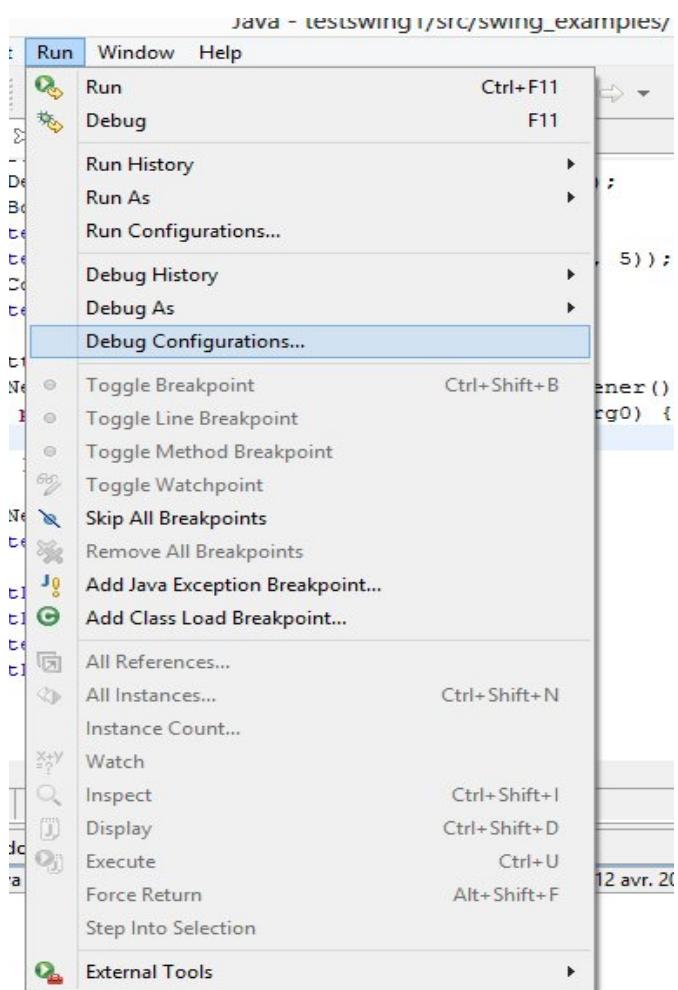
Complétez alors le code lié à l'événement

```
*Testswing_ex.java
    setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
    setBounds(100, 100, 450, 300);
    contentPane = new JPanel();
    contentPane.setBorder(new EmptyBorder(5, 5, 5, 5));
    setContentPane(contentPane);
    contentPane.setLayout(null);

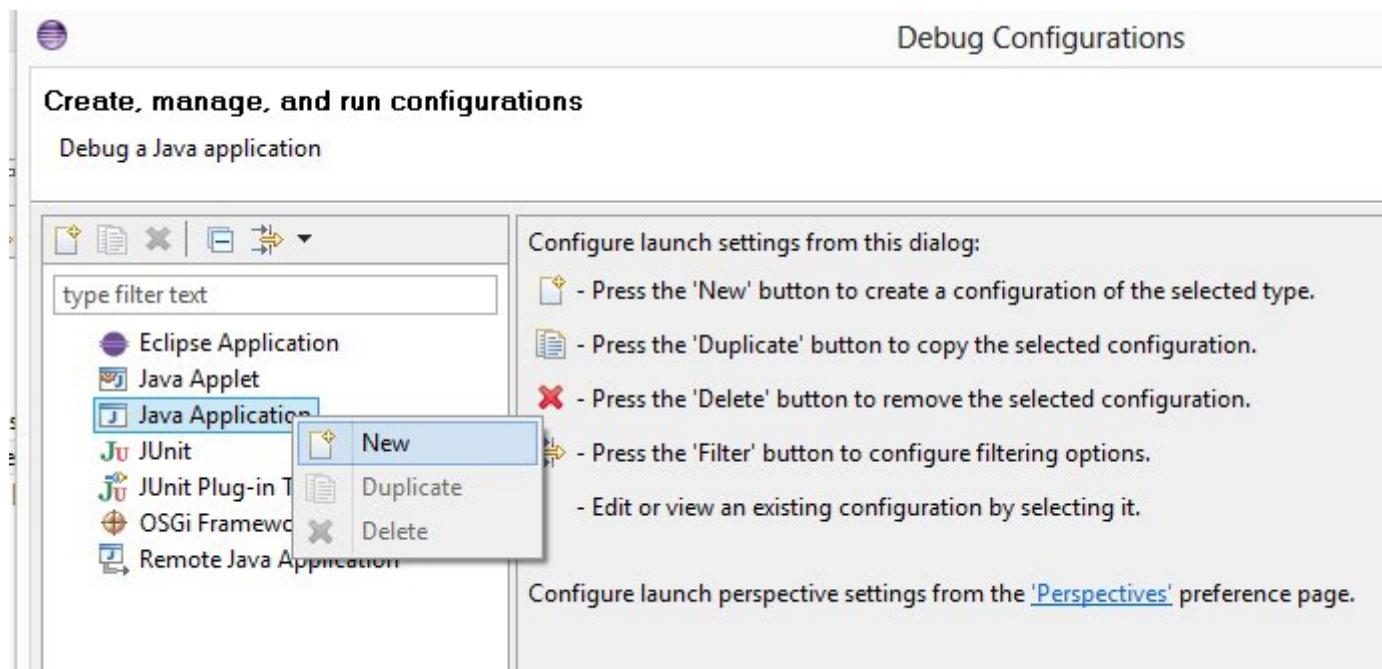
    JButton btnNewButton = new JButton("ok");
    btnNewButton.addActionListener(new ActionListener() {
        public void actionPerformed(ActionEvent arg0) {
            textField.setText("hello");
        }
    });
    btnNewButton.setBounds(27, 43, 89, 23);
    contentPane.add(btnNewButton);

    textField = new JTextField();
    textField.setBounds(30, 90, 86, 20);
    contentPane.add(textField);
    textField.setColumns(10);
}
```

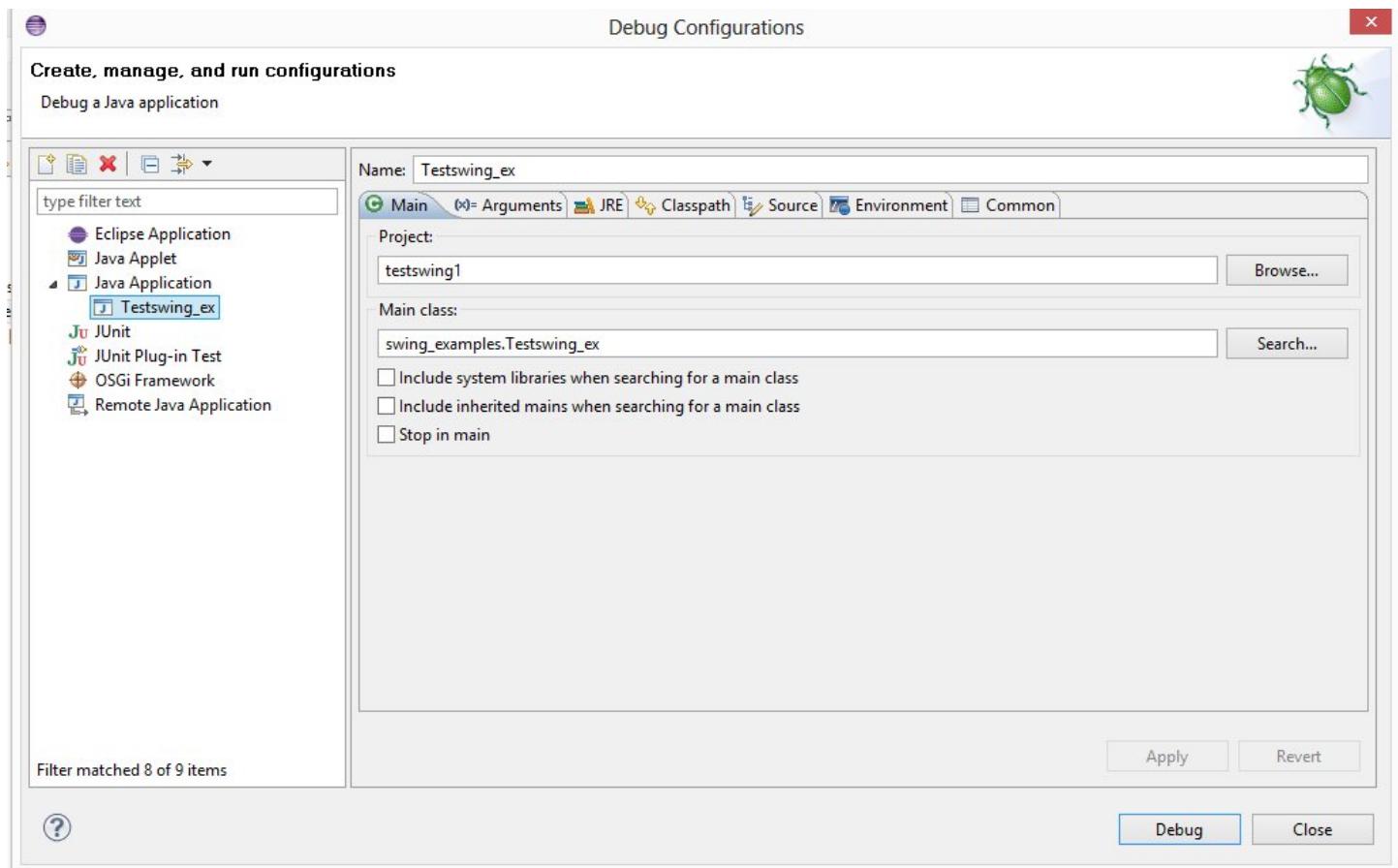
Pour tester notre programme en mode debug suivre la procédure page N°9



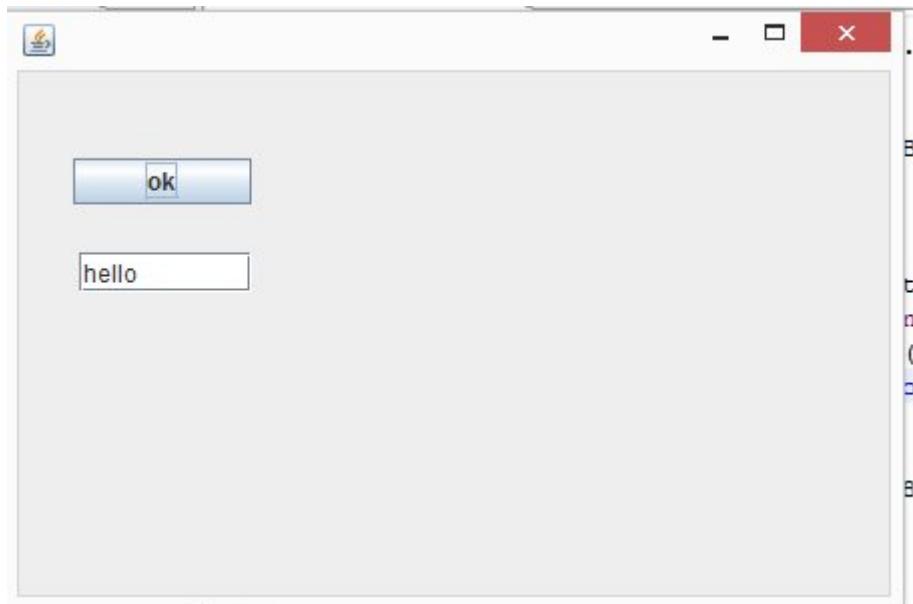
Cliquez droit



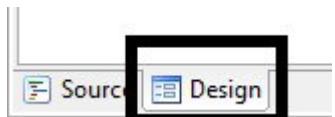
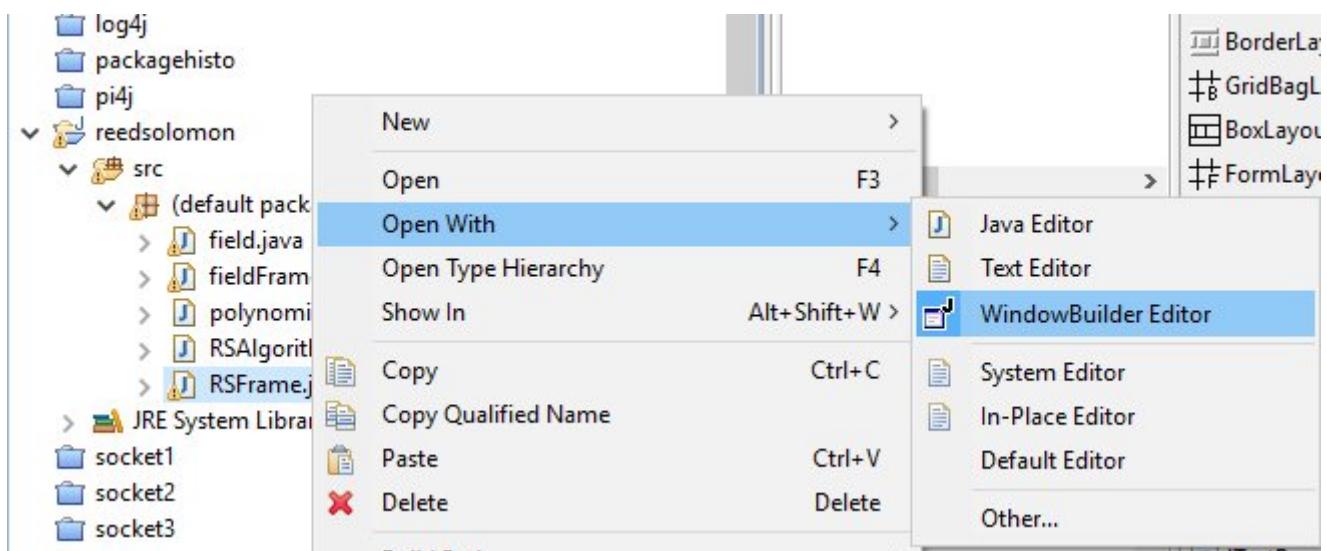
Sélectionnez les bons éléments pour le projet et la classe de la méthode main



Puis en mode debug testez notre programme en cliquant sur le bouton :

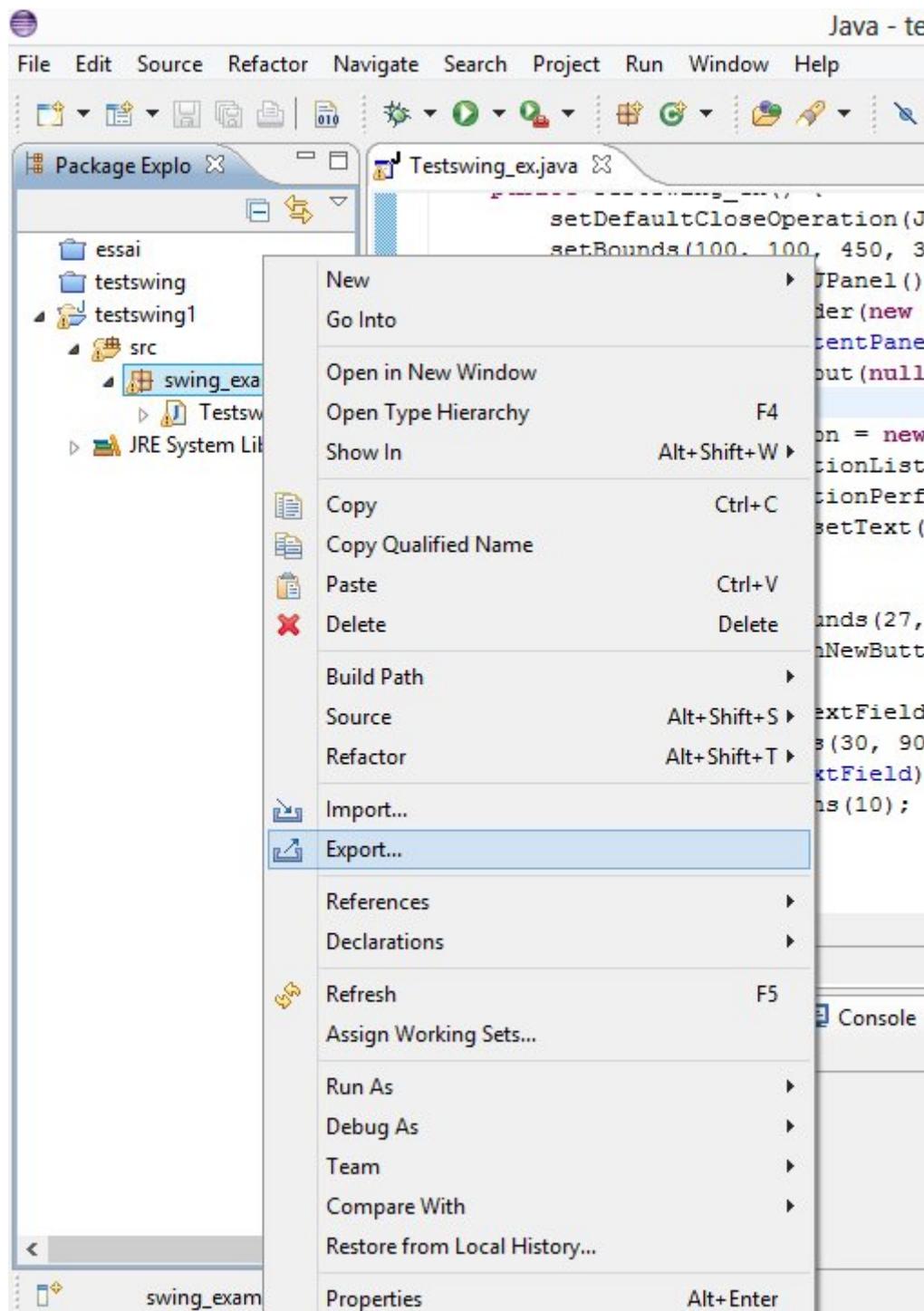


Si l'onglet design n'apparaît pas cliquez droit sur le fichier source

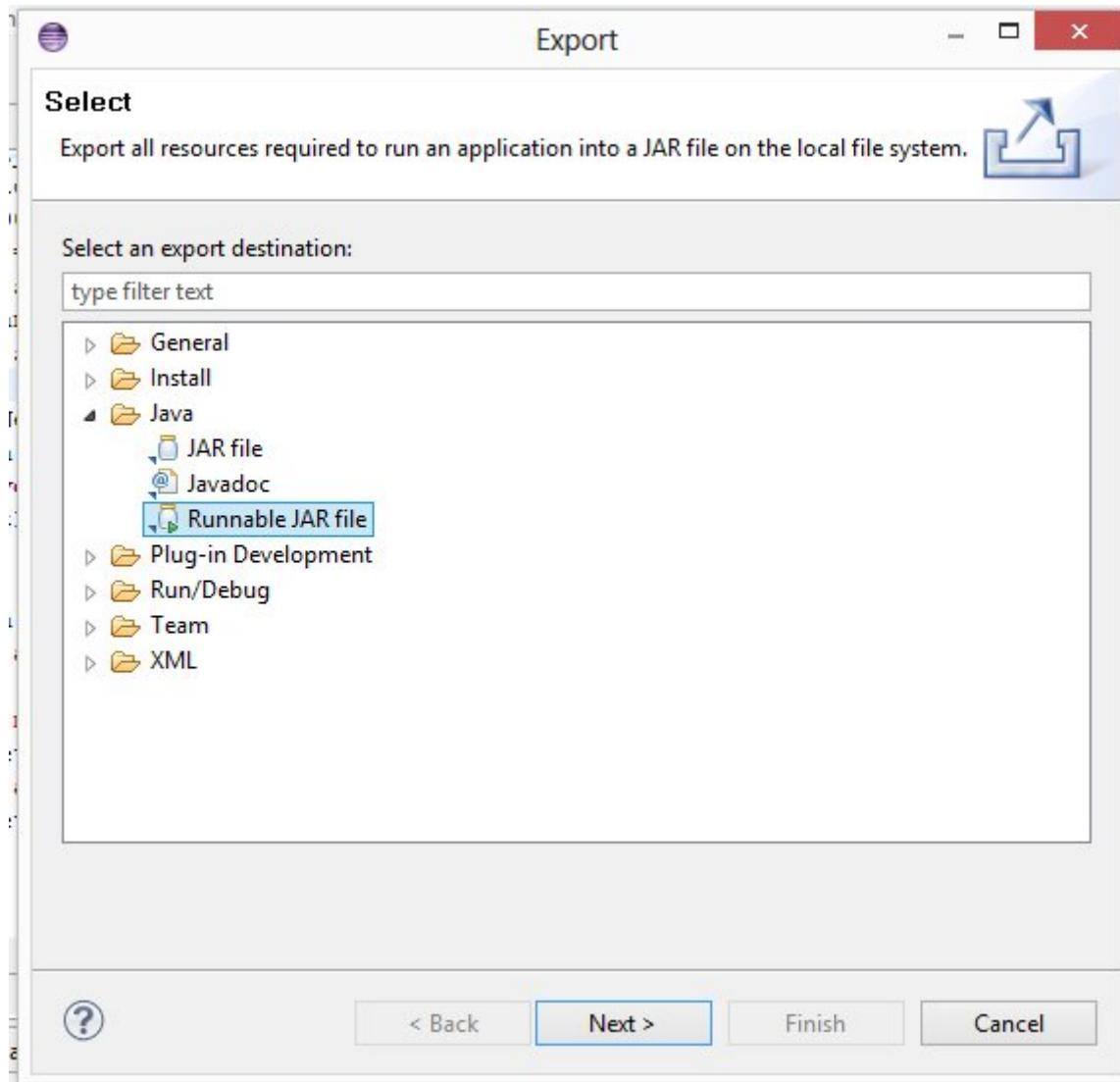


4) Création d'une archive JAR exécutable

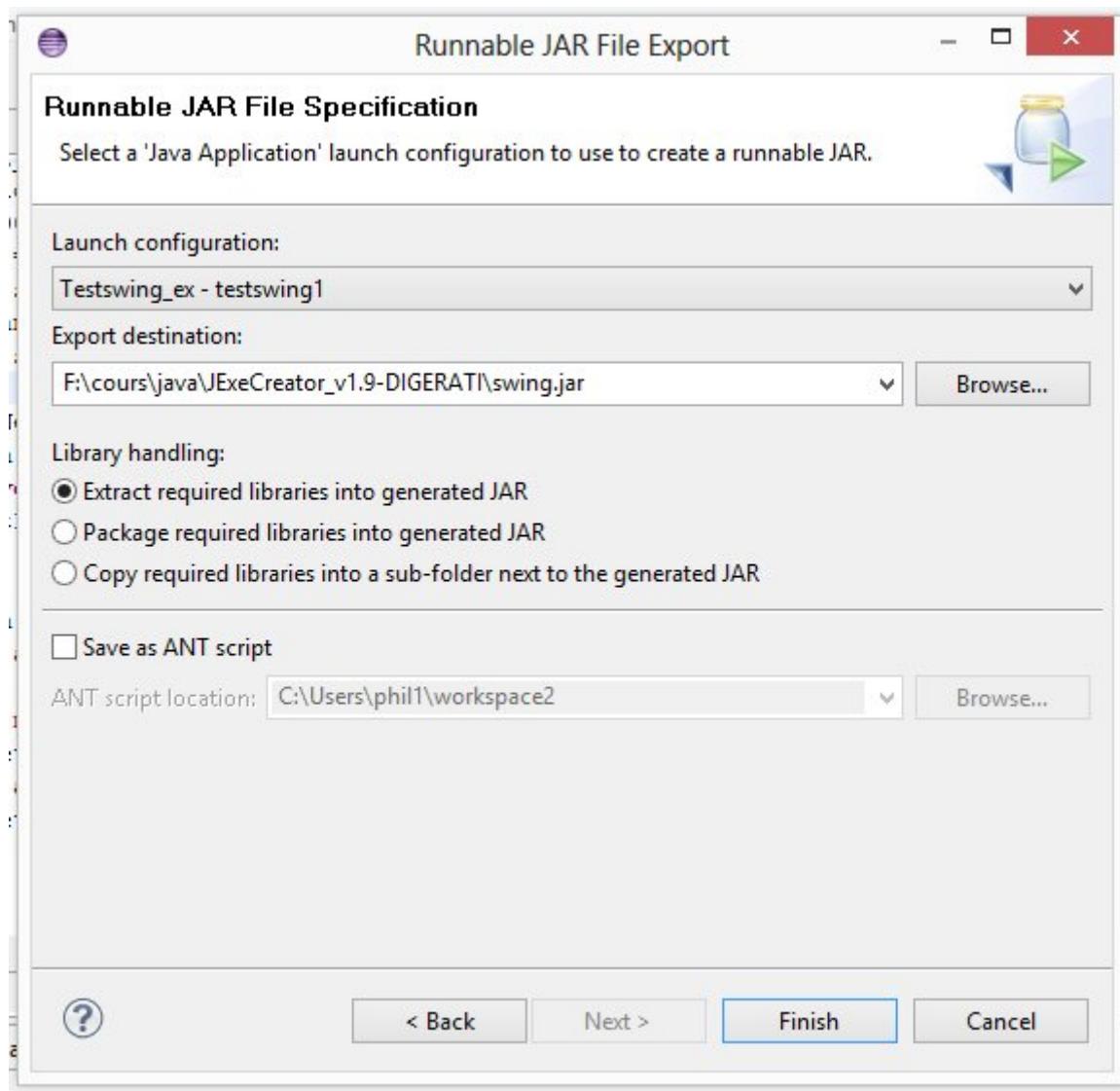
On souhaite créer une archive jar qui intègre notre projet JAVA. Sélectionnez notre paquetage, cliquez droit et choisir export



Choisir un exécutable du type JAR runnable



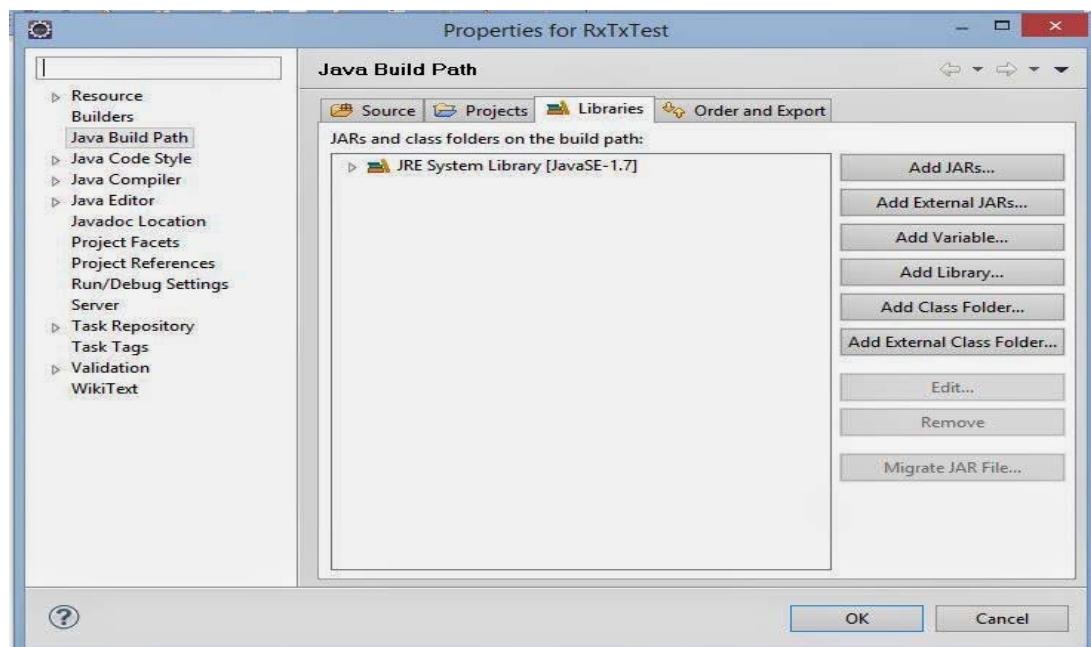
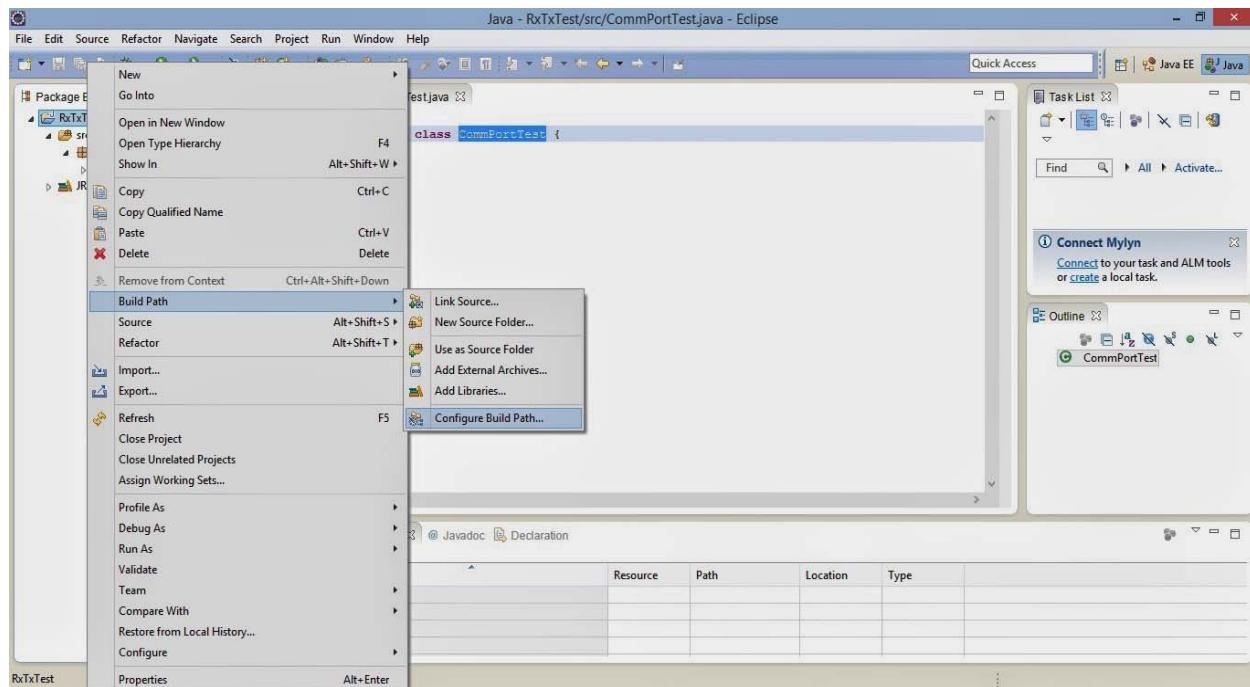
Sélectionnez la classe ou se situe la méthode main ainsi que le répertoire où sera créé l'archive .jar



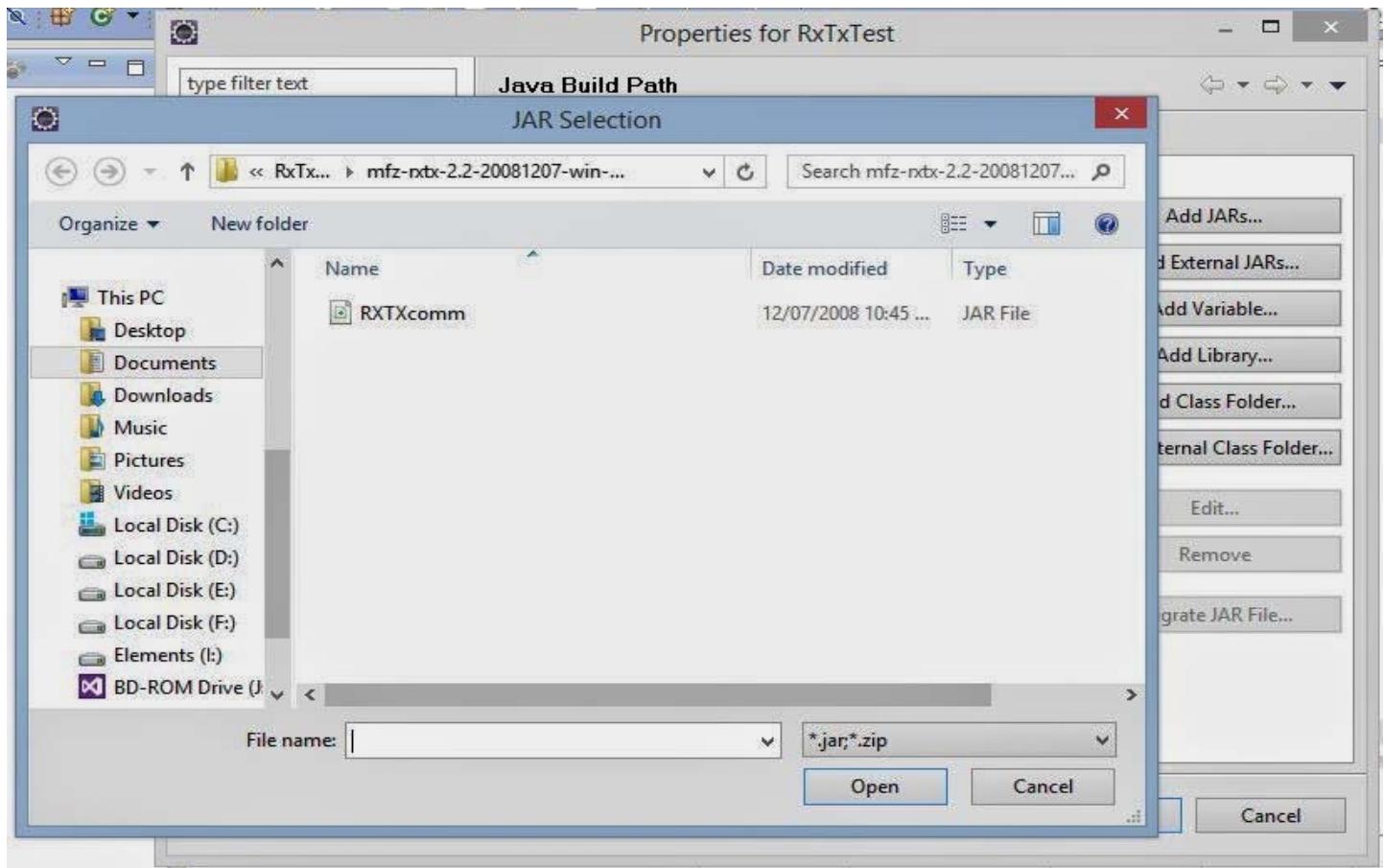
Installation du pilote RXTXCOM

Présentation

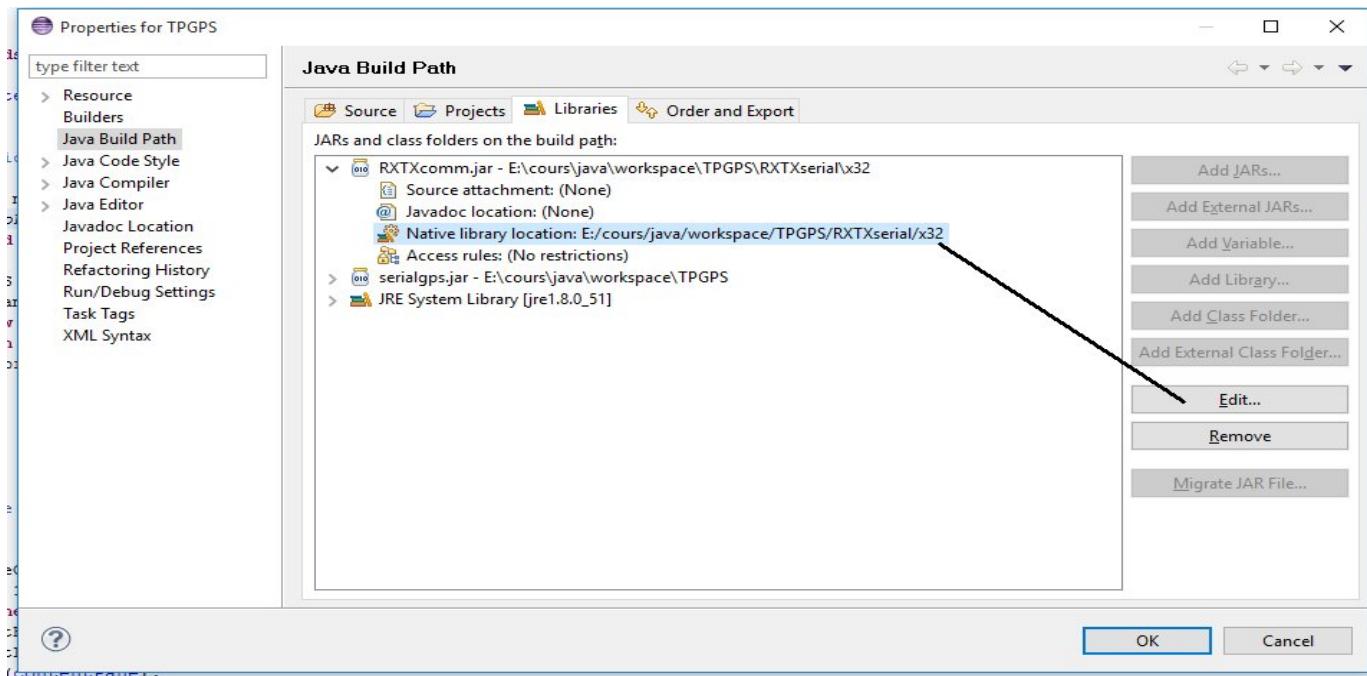
Cliquez droit sur le projet puis BuildPath configurer build path



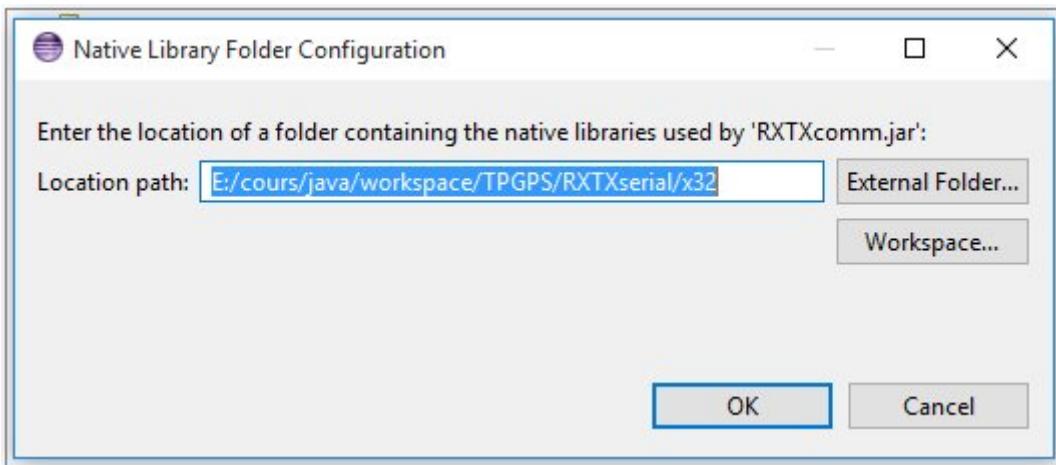
Dans le répertoire où se situe le fichier RXTXcomm.jar, sélectionnez celui-ci



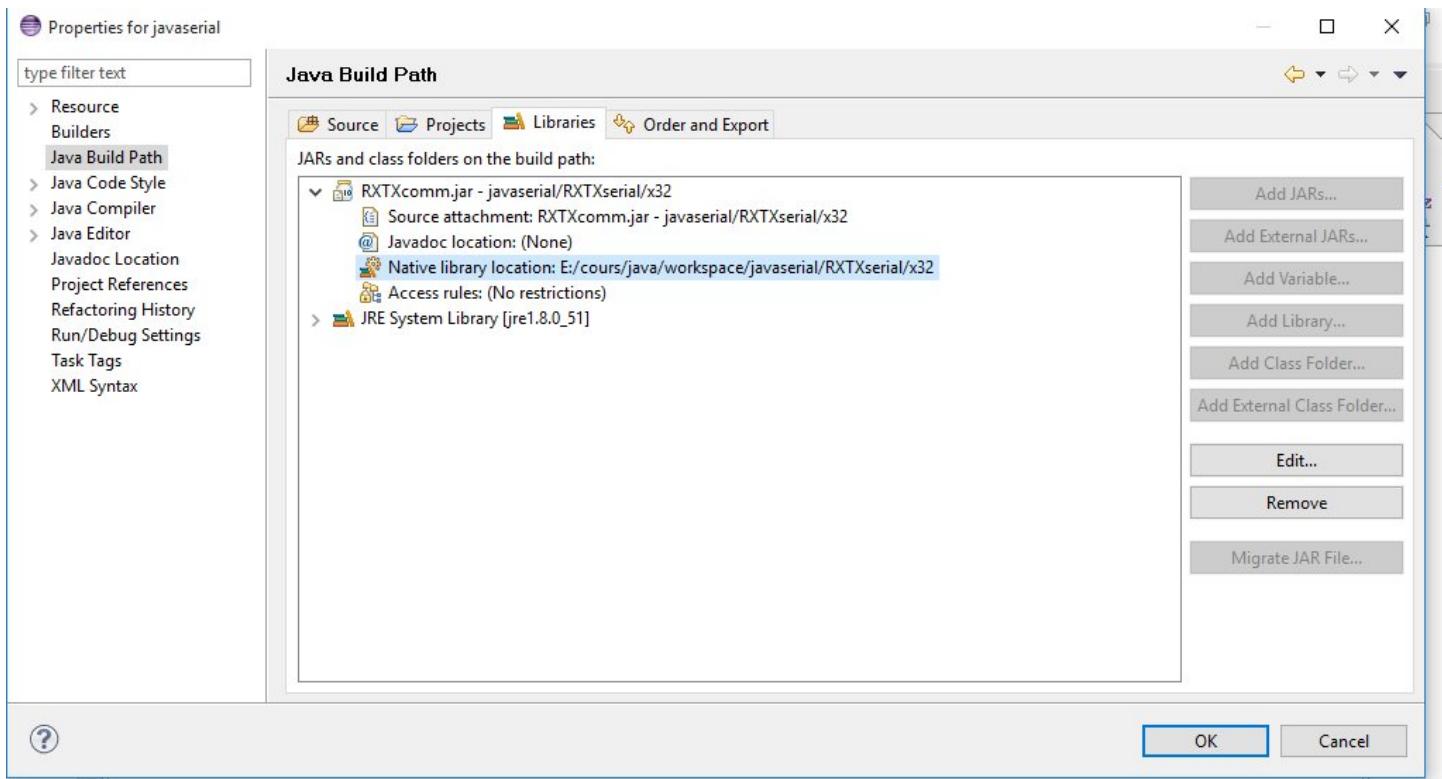
Sélectionnez le lien Native library location et cliquez sur edit



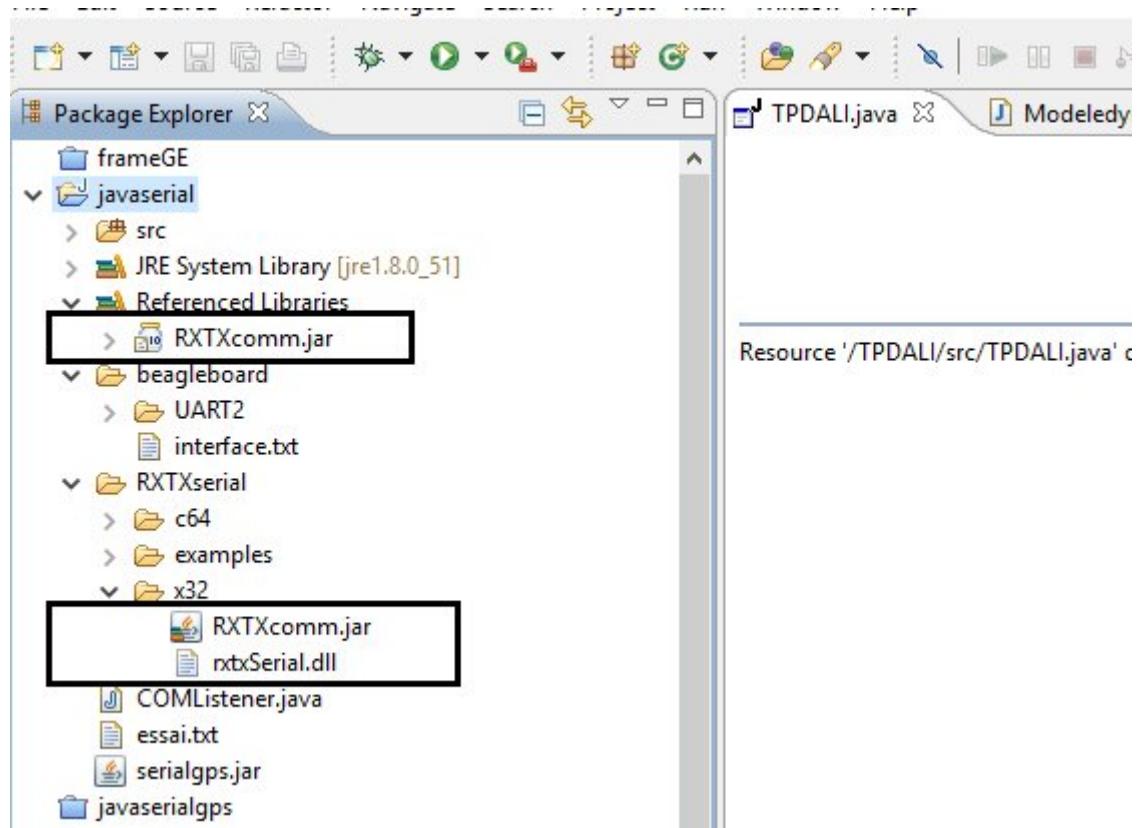
Sélectionnez le répertoire où se situe la DLL rxtxSerial.dll



La configuration est prête



Notre projet est prêt



Ne pas oubliez d'importer les librairies suivantes :

```
import gnu.io.CommPortIdentifier;
import gnu.io.SerialPort;
```

Attention à désinstaller les ports virtuels du bluetooth car cela pose des problèmes

Installation du pilote RXTXCOM

Sur Raspberry

Sur La raspberry pi 2 on peut installer la package en disposant d'une connexion internet de la manière suivante

```
sudo apt-get install librxtx-java
```

Si le package n'est pas disponible Téléchargez le fichier

```
librxtx-java_2.2pre2-11_armhf.deb
```

puis installez le package debian

```
sudo dpkg -i librxtx-java_2.2pre2-11_armhf.deb
```

Pour lancez la compilation du fichier source

```
javac -cp /usr/share/java/RXTXcomm.jar *.java
```

Pour lancez l'application java nav.class

```
java -Djava.library.path=/usr/lib/jni/ -cp /usr/share/java/RXTXcomm.jar:. nav
```

Dans notre fichier source ne pas oublier de déclarer

```
import gnu.io.*;
import java.io.*;
import java.util.*;
```

Intégration de la librairie JFreechart

Présentation

Jfreechart fait partie d'un ensemble d'API permettant de générer des graphiques du type courbes, histogramme Ces graphiques s'intègrent sous l'environnement swing .

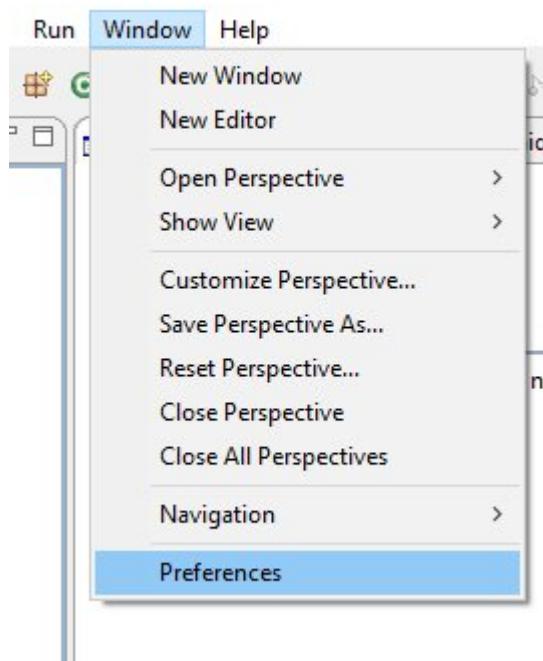
Téléchargez les deux packages suivants :

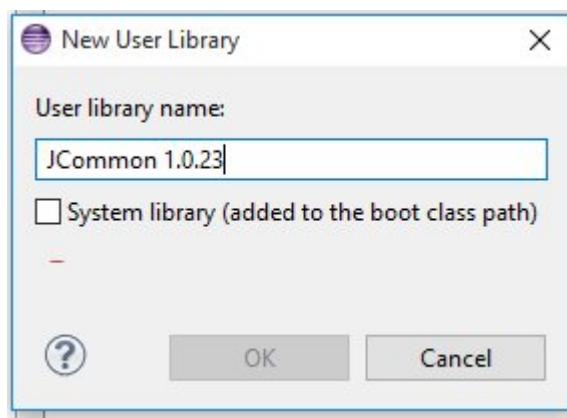
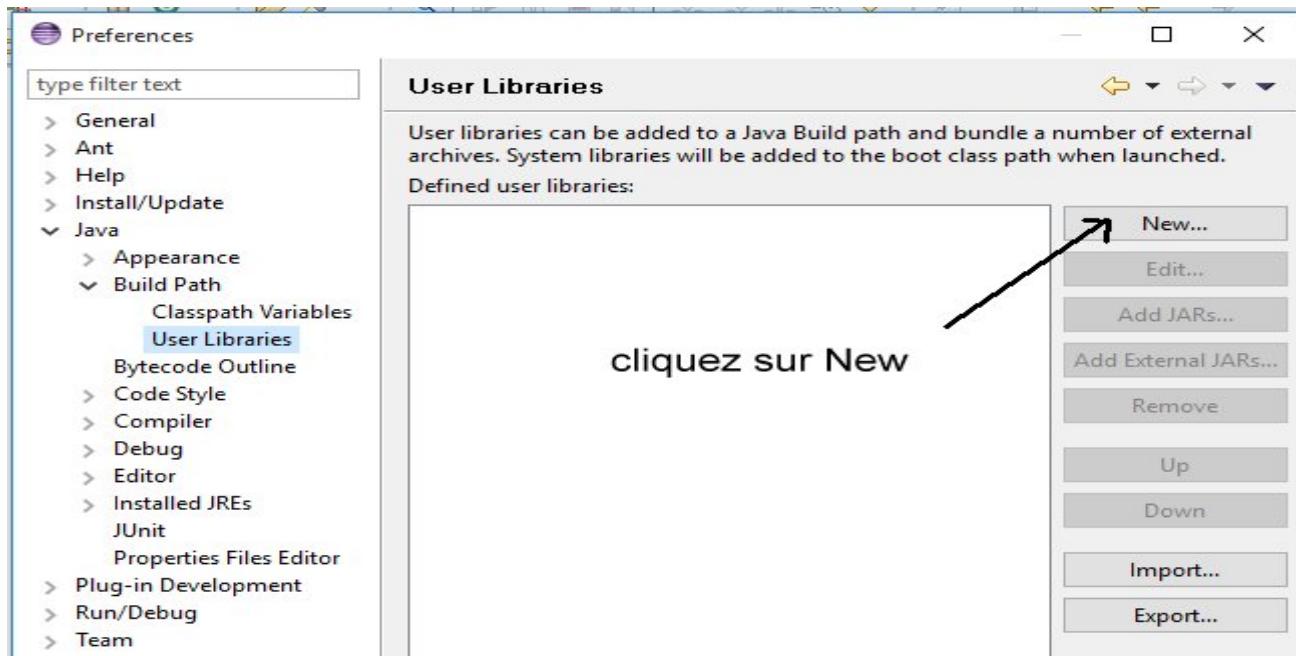
jcommon-1.0.23.zip
jfreechart-1.0.19

Décompactez les deux archives

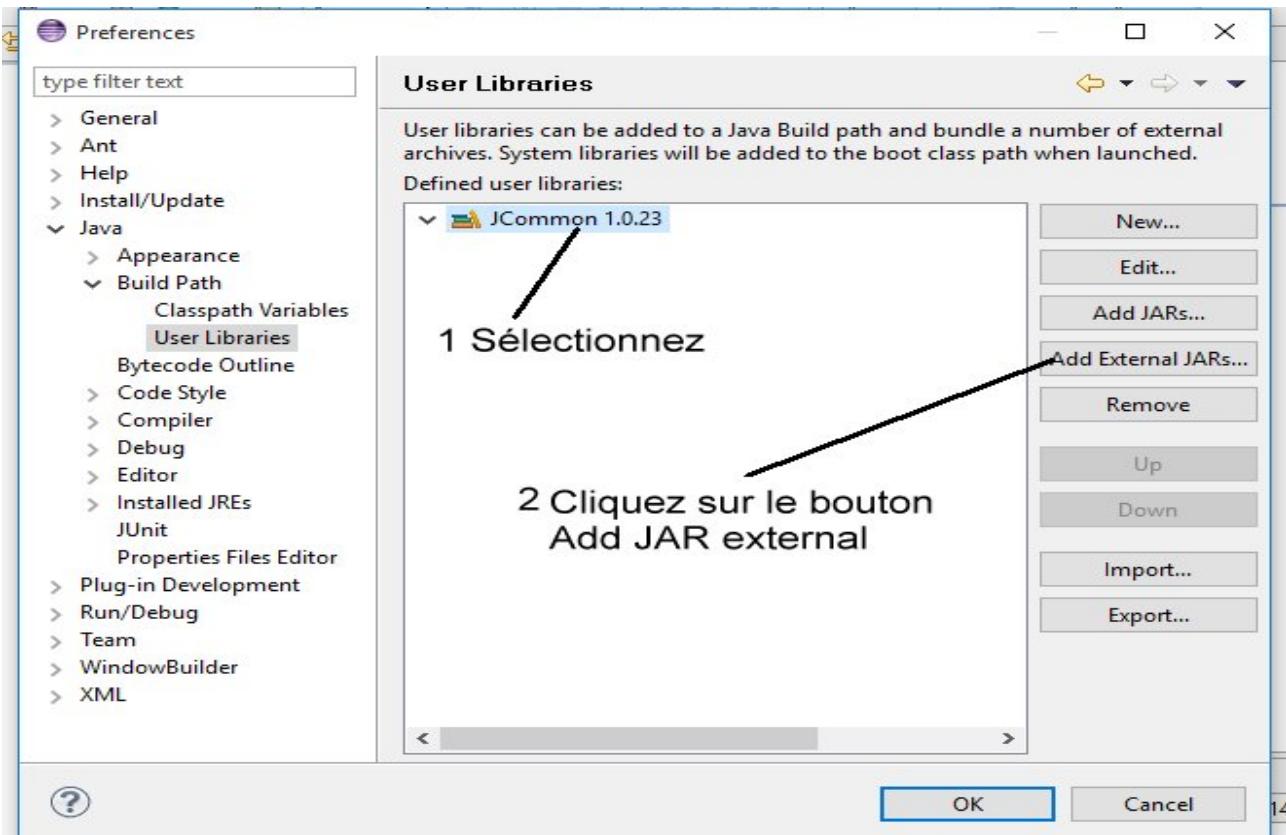
Afin d'intégrer de manière permanente la librairie on va utiliser la procédure suivante :

Allez dans windows → préférences

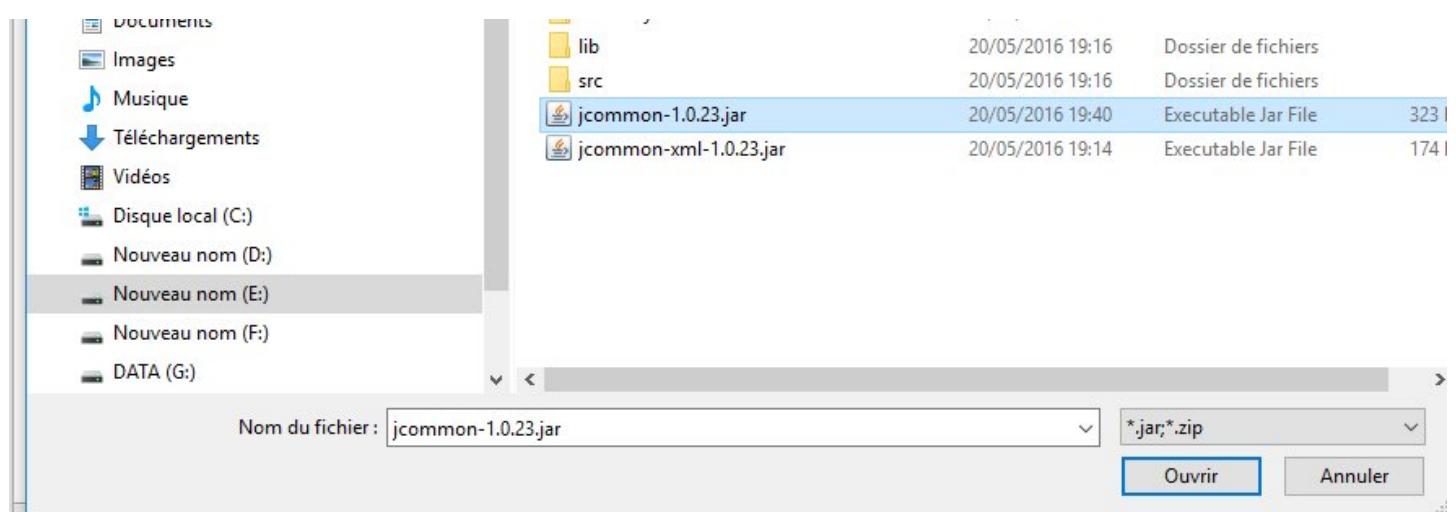




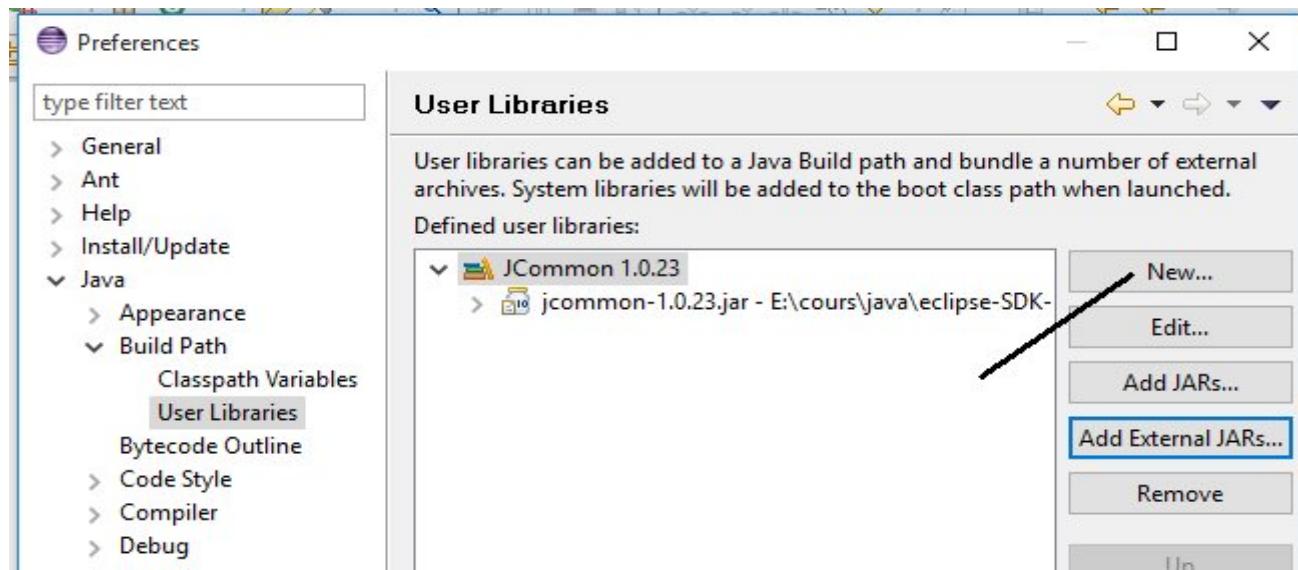
Sélectionnez JCommon 1.0.23



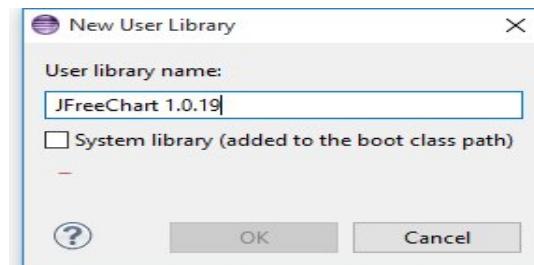
Choisir le fichier JCommon-1.0.23.jar



Cliquez sur new

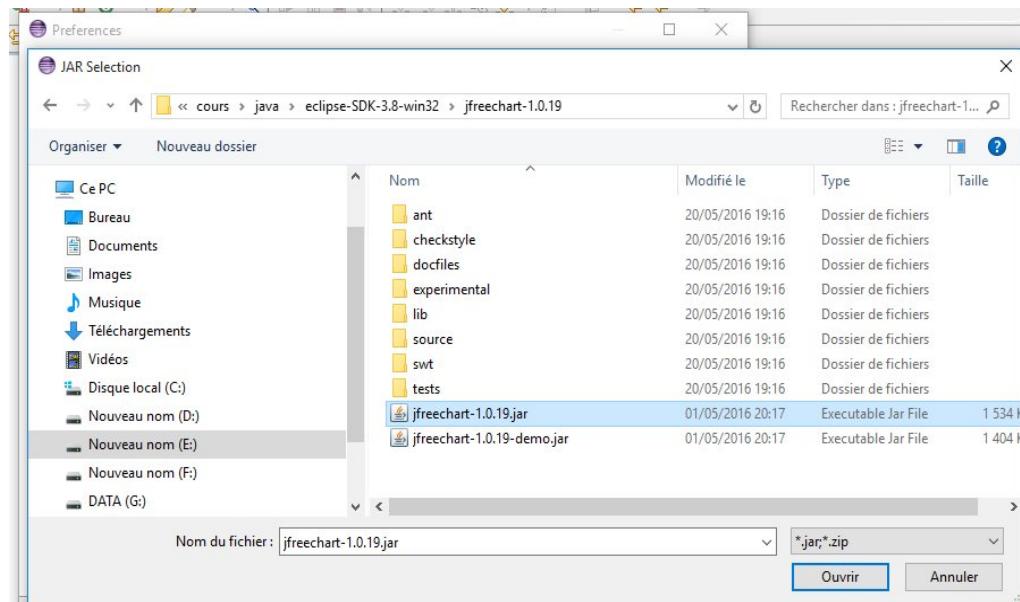


Créez un nouveau paquetage

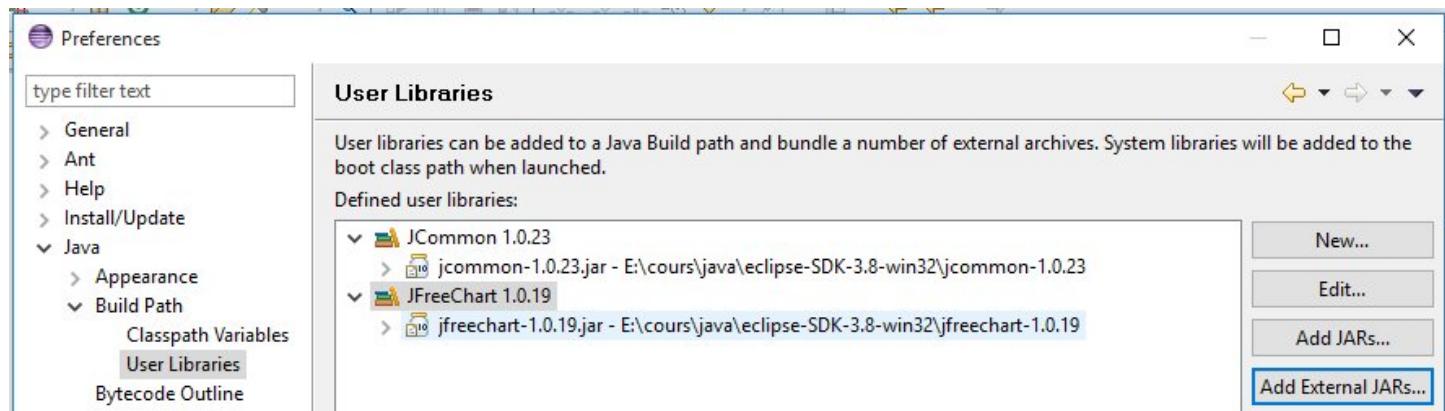


Cliquez sur Add external Jar

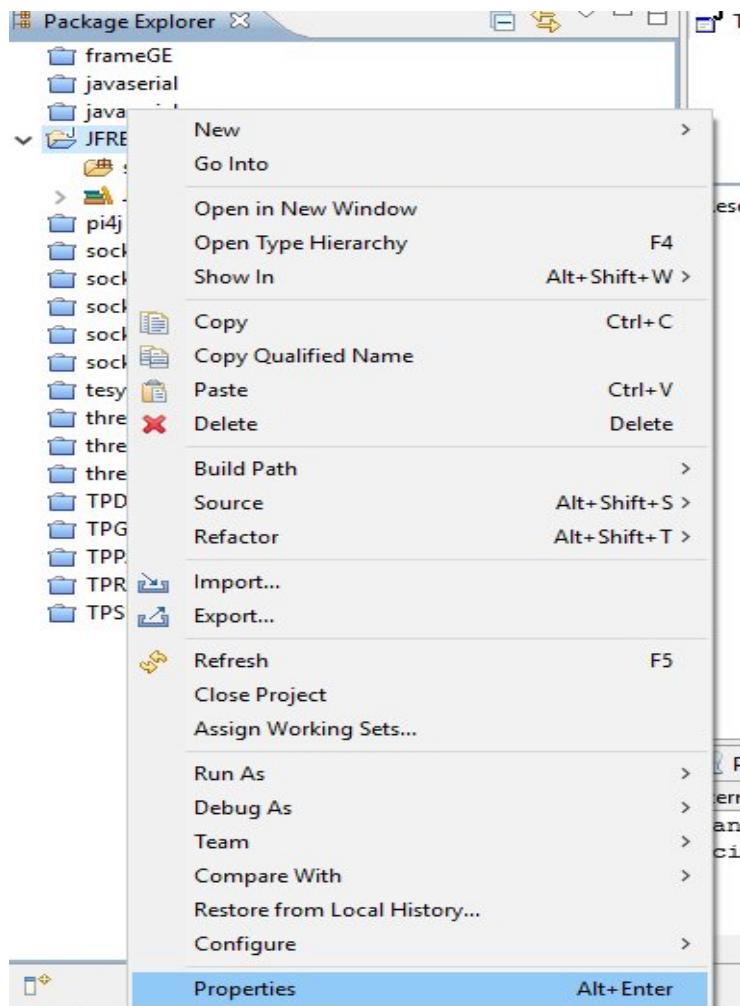
Et sélectionner le fichier

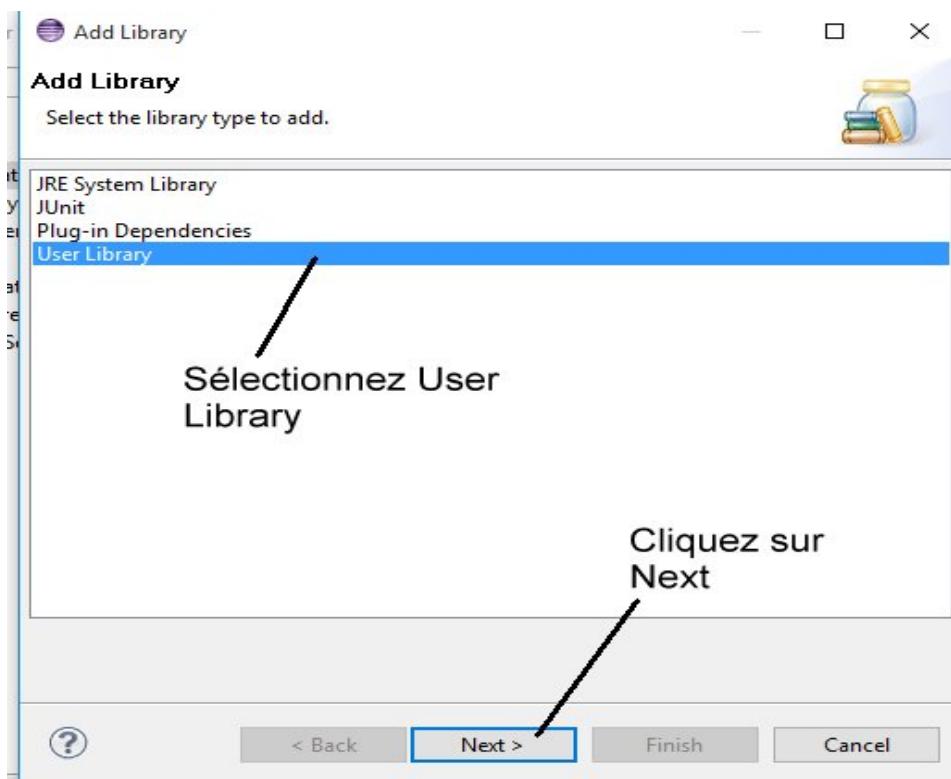
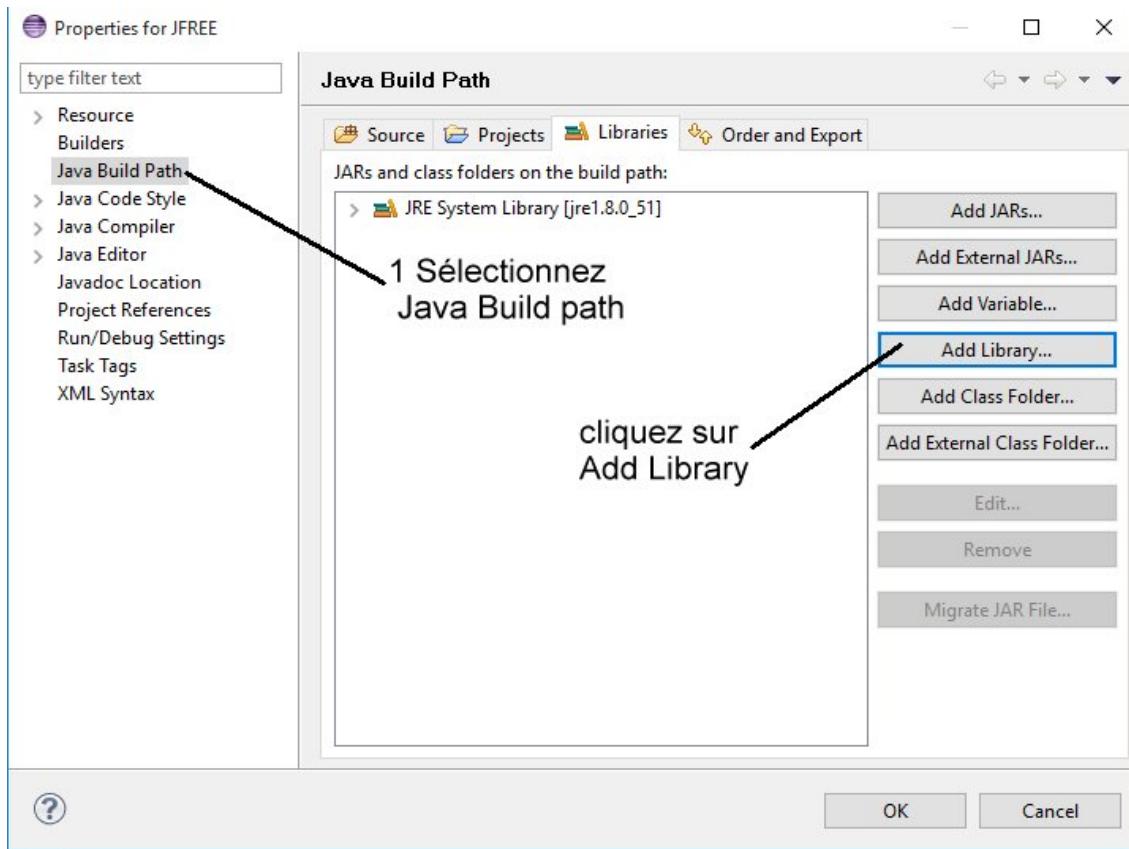


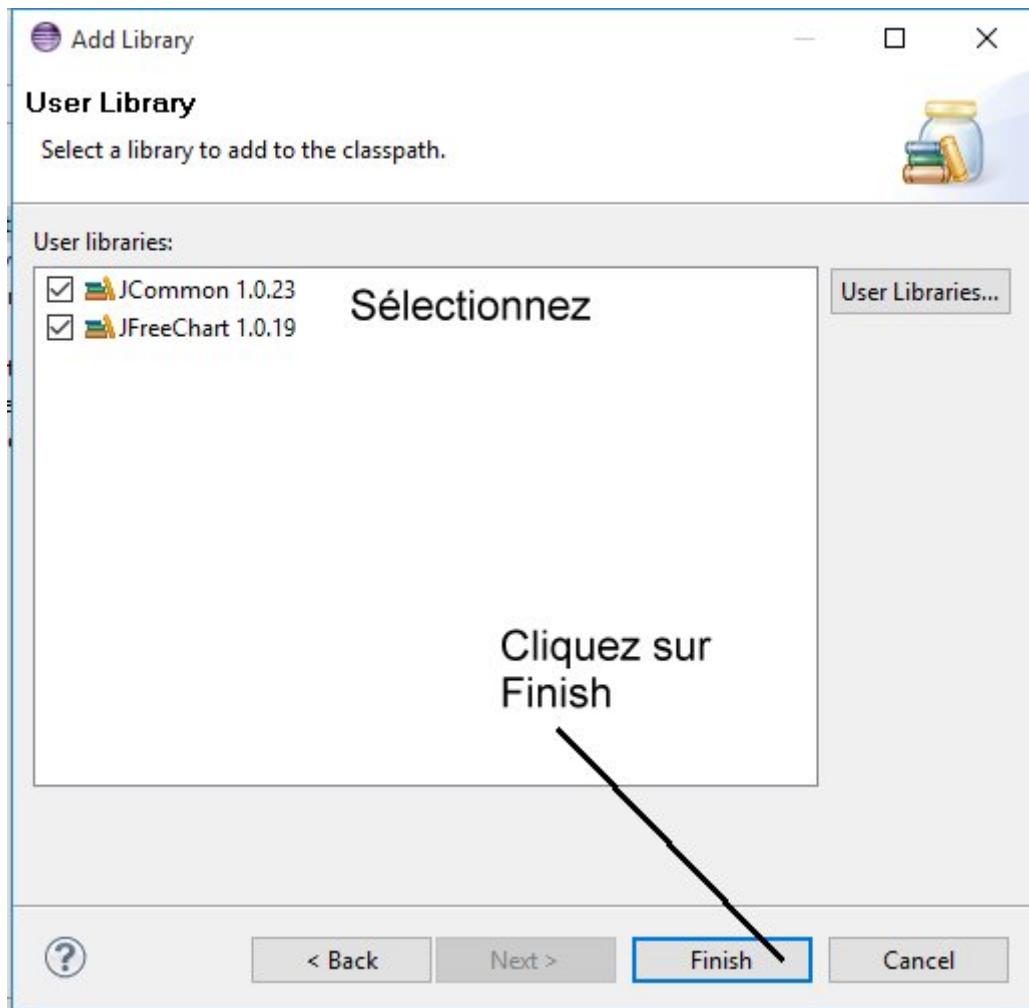
Les librairies sont les suivantes :



Dans notre projet cliquez droit sur le nom du projet







Pour compilez Jfreechart et la librairie PI4J ensemble sur la raspberry PI 2

```
//projet température :  
sudo javac –classpath  
/home/pi/rasp/JAVA/jfreechart.jar:/home/pi/rasp/JAVA/jcommon.jar:/opt/pi4j/lib/* *.* //POUR  
COMPILE  
sudo java -Djava.library.path=/usr/lib/jni -classpath  
/home/pi/rasp/JAVA/jfreechart.jar:/home/pi/rasp/JAVA/jcommon.jar:/opt/pi4j/lib/*:. capteur_I2C  
//POUR EXECUTER
```

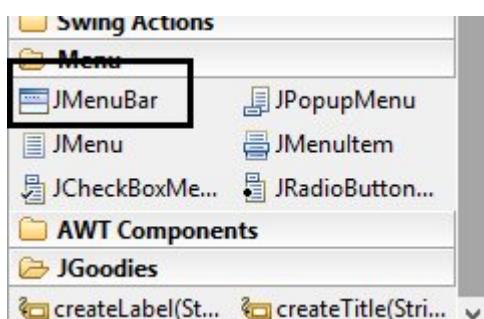
Réalisation d'un Menu sous WINDOWSBUILDER

Présentation

La première chose est de créer un projet sous Windowsbuilder du type swing et de cliquer sur l'onglet design



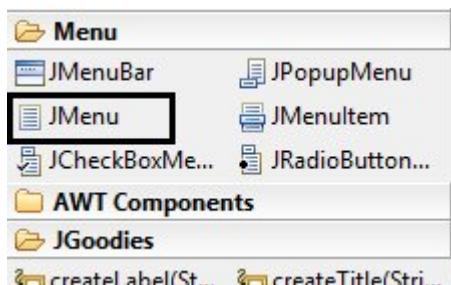
On va d'abord créer la bar de menu cliquez sur l'objet JMenuBar



Déposez et glissez l'objet en haut de la feuille



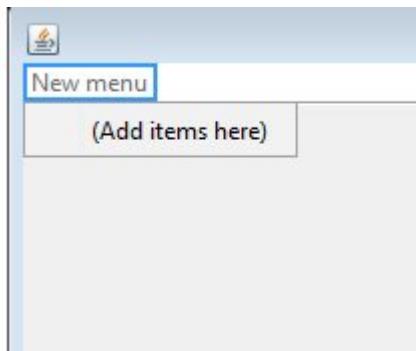
On va créer un premier menu sur la barre des menus



Glissez l'objet JMenu dans la barre des menus

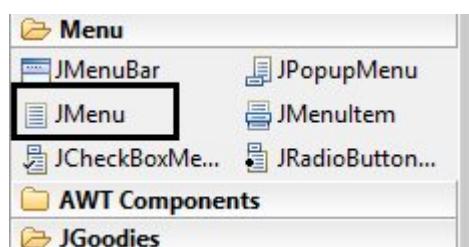


Le nouveau menu s'appelle par défaut New Menu

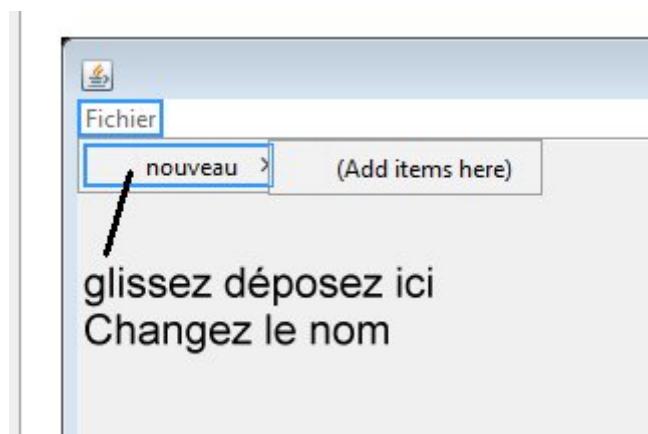


Changez son nom en fichier par exemple

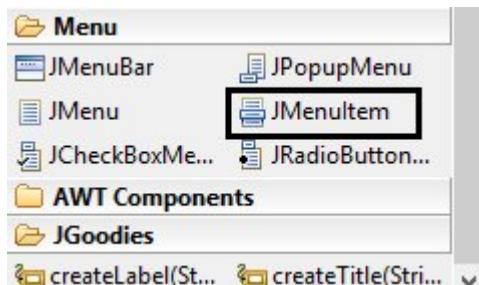
On peut rajouter un sous menu au premier menu en utilisant l'objet JMenu



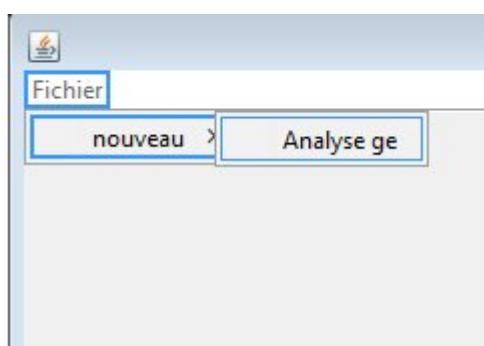
Déposez ce sous menu sous le menu fichier



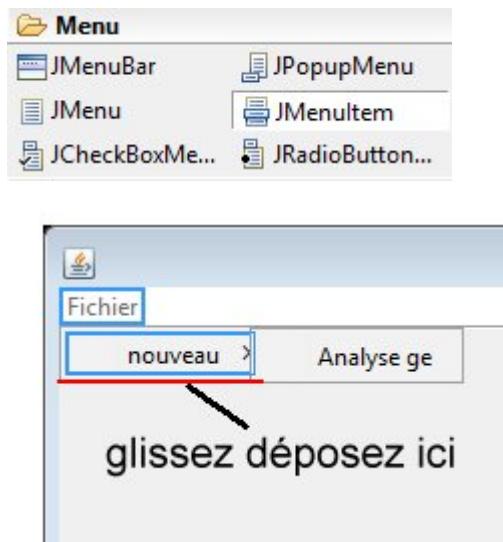
Pa la suite dans le sous menu on peut disposer d'un Item afin de lancer un événement



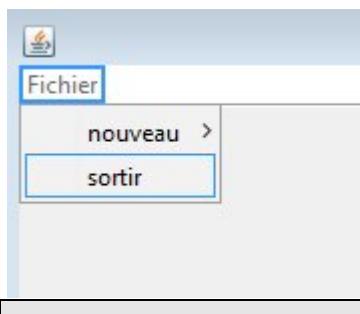
Ce nouvel item s'appellera Analyse ge



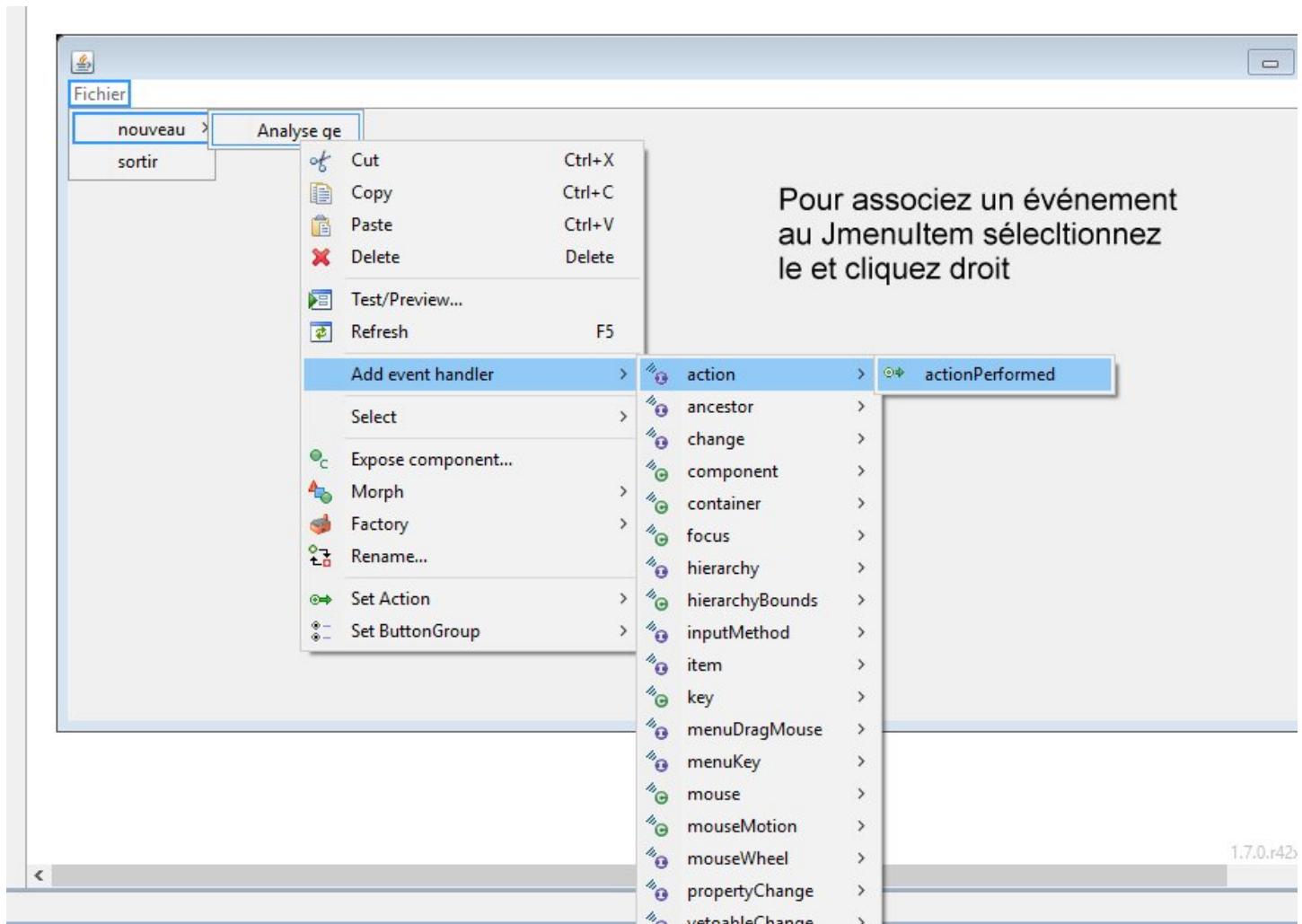
Dans le menu fichier on souhaite créer un item sortir, afin de sortir du programme



Changez le nom de l'item en sortir



Pour lancer un événement lorsqu'on cliquera sur l'item, cliquez droit sur l'item → Add event handler → action → action performed



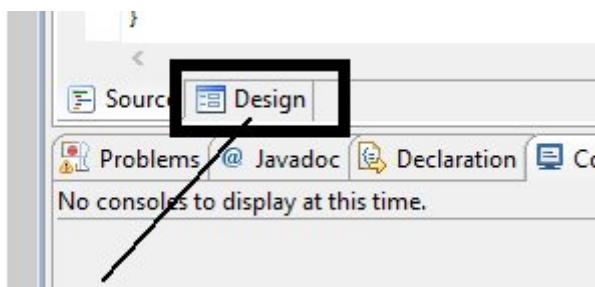
Un objet est créé dans le code afin de gérer cet événement :

```
JMenuItem mntmSortir = new JMenuItem("sortir");
mntmSortir.addActionListener(new ActionListener() {
    public void actionPerformed(ActionEvent arg0) {
        System.exit(JFrame.EXIT_ON_CLOSE);
    }
});
```

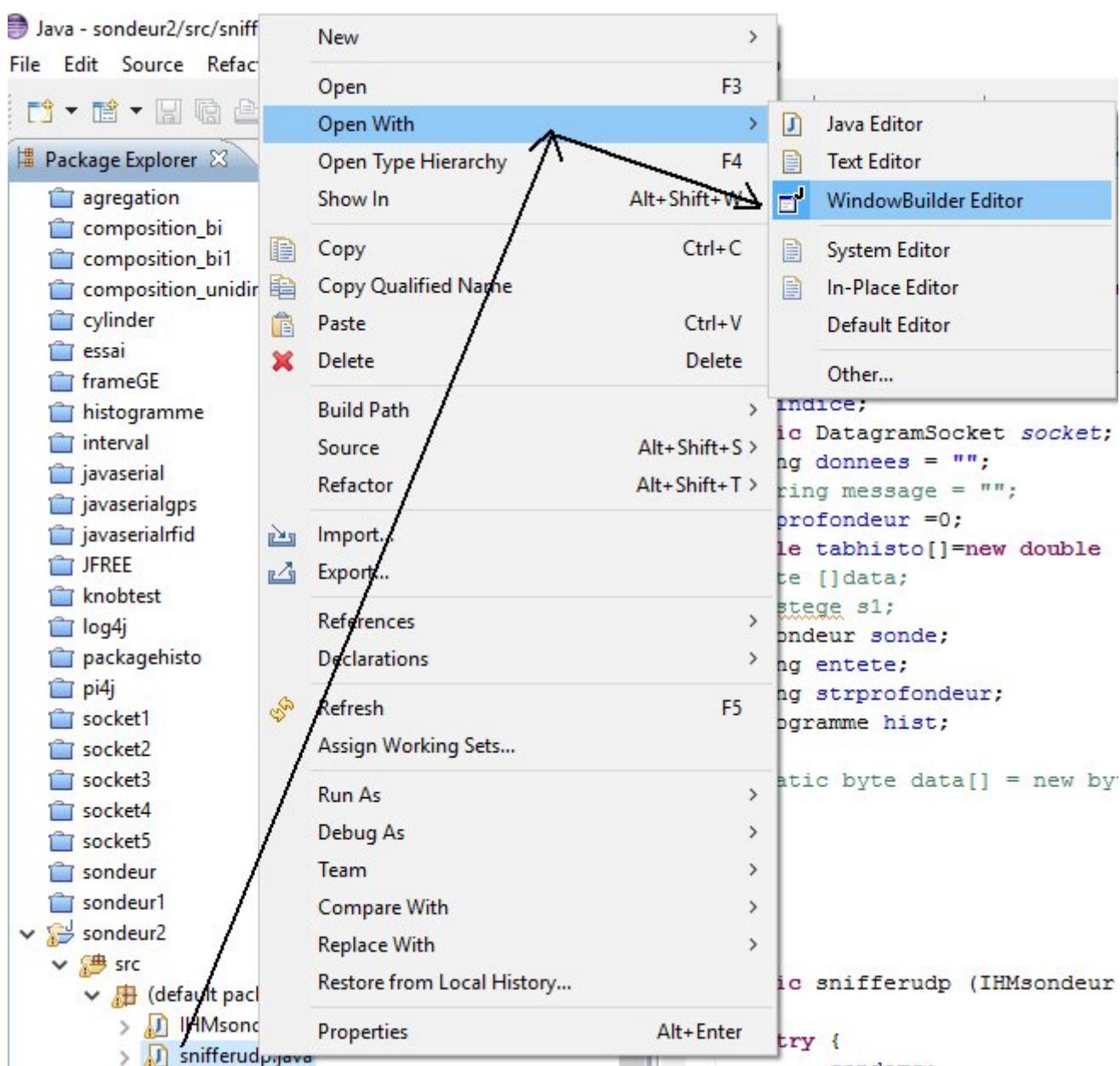
Pour le sous menu sortir

```
System.exit(JFrame.EXIT_ON_CLOSE);
```

L'ongle design peut disparaître



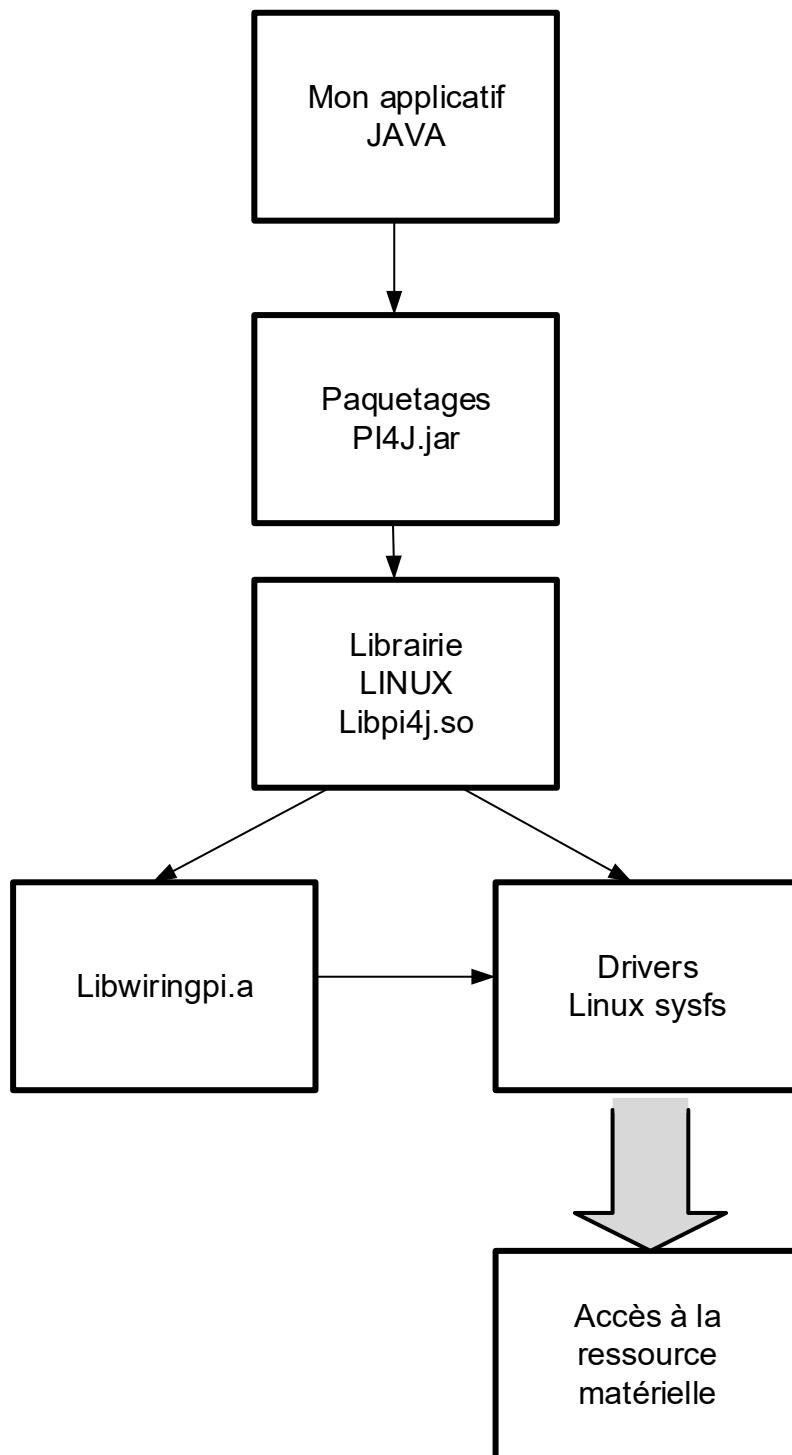
Pour faire apparaître l'ongle design cliquez droit sur la classe ou se trouve la fonction main puis → open With - → windowBuilder Editor



Installation de PI4J dans l'environnement LINUX RASPIAN de la RASPBERRY

1) Présentation

PI4J est un ensemble de paquetage PI4J.jar permettant en langage JAVA de piloter le port d'extension de la carte RASPBERRY PI2.



2) Installation

Téléchargez sur Internet le package debian suivant :

<http://get.pi4j.com/download/pi4j-1.0.deb>

ou par un apt-get

sudo apt-get install pi4j

Installez le sur la carte RASPBERRY PI 2 par la commande :

sudo dpkg -i pi4j-1.0.deb

Les fichiers sources seront installés dans :

/opt/pi4j/lib
/opt/pi4j/examples

Notre programme doit importer les packages suivants:

```
import com.pi4j.io.gpio.GpioController;
import com.pi4j.io.gpio.GpioFactory;
import com.pi4j.io.gpio.GpioPinDigitalInput;
import com.pi4j.io.gpio.GpioPinDigitalOutput;
import com.pi4j.io.gpio.PinPullResistance;
import com.pi4j.io.gpio.RaspiPin;
import com.pi4j.io.gpio.PinState;
import com.pi4j.io.gpio.event.GpioPinDigitalStateChangeEvent;
import com.pi4j.io.gpio.event.GpioPinListenerDigital;
import com.pi4j.io.*;
import com.pi4j.wiringpi.Gpio;
```

3) Compilation et exécution

Pour compiler notre programme deux méthodes existent :

```
sudo pi4j --compile *.java
```

Ou javac -classpath .:classes:/opt/pi4j/lib/* ...

Pour exécuter notre applicatif :

```
sudo pi4j --run nom
```

Ou

```
sudo java -classpath .:classes:/opt/pi4j/lib/* ...
```

Attention à décharger le module gpiosen avant d'utiliser pi4j

4) Le pilotage des ports d'entrées sorties

PI4J permet de piloter en entrées ou en sortie les broches du connecteur d'extension de la RASPBERRY PI2.

L'applicatif suivant permet de programmer en sortie la broche *GPIO_12* du connecteur d'extension et de programmer des niveaux logiques sur cette broche

```
// crée un contrôleur GPIO
Final GpioController gpio = GpioFactory.getInstance();
// Crée une broche de sortie sur le GPIO
Final GpioPinDigitalOutput output = gpio.provisionDigitalOutputPin( RaspiPin.GPIO_12,
PinState.LOW);
// Pilote la broche de sortie du GPIO
output.high(); // sortie à l'état haut
output.low(); // sortie à l'état bas
output.toggle(); // Change l'état de la sortie
```

L'applicatif suivant permet de programmer en entrée la broche *GPIO_02* du connecteur d'extension et de lire un niveau logique sur cette broche.

```
// crée un contrôleur GPIO
Final GpioController gpio = GpioFactory.getInstance();

final GpioPinDigitalInput input = gpio.provisionDigitalInputPin( RaspiPin.GPIO_02,
PinPullResistance.PULL_DOWN);

boolean etat;
if (input.isHigh()==true) etat=true; // La broche d'entrée est à un
else etat=false ;// la broche d'entrée est à zéro
```

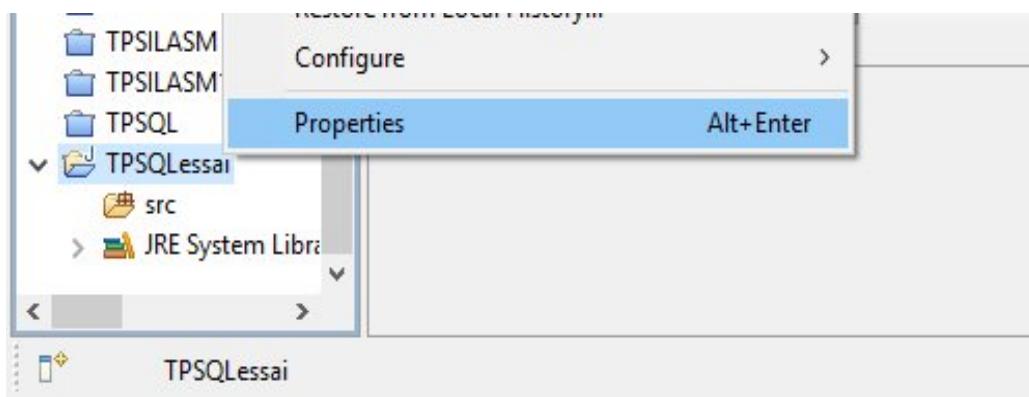
Sous X11VNC il n'est pas possible de lancer un programme réalisé avec swing. Pour résoudre ce problème lancez dans une fenêtre terminale

```
xhost +
sudo pi4j --run nom&
```

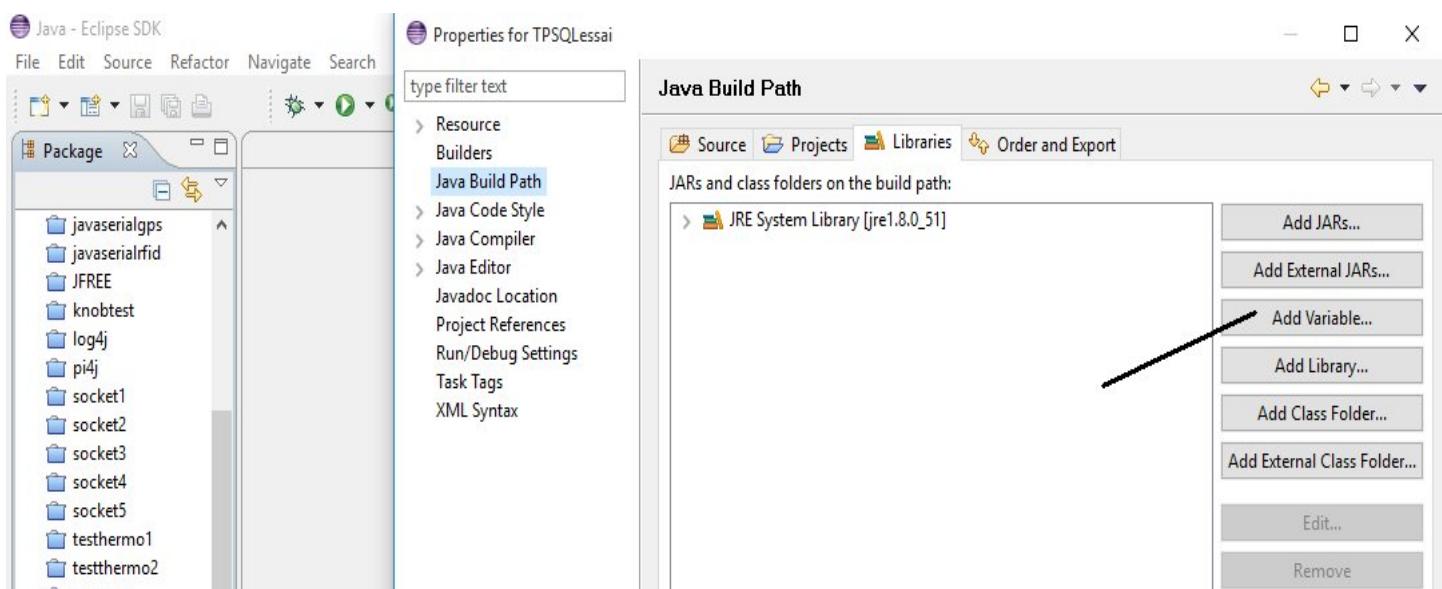
Installation du driver JDBC pour MySQL sous Eclipse

Créez un nouveau projet JAVA sous eclipse.

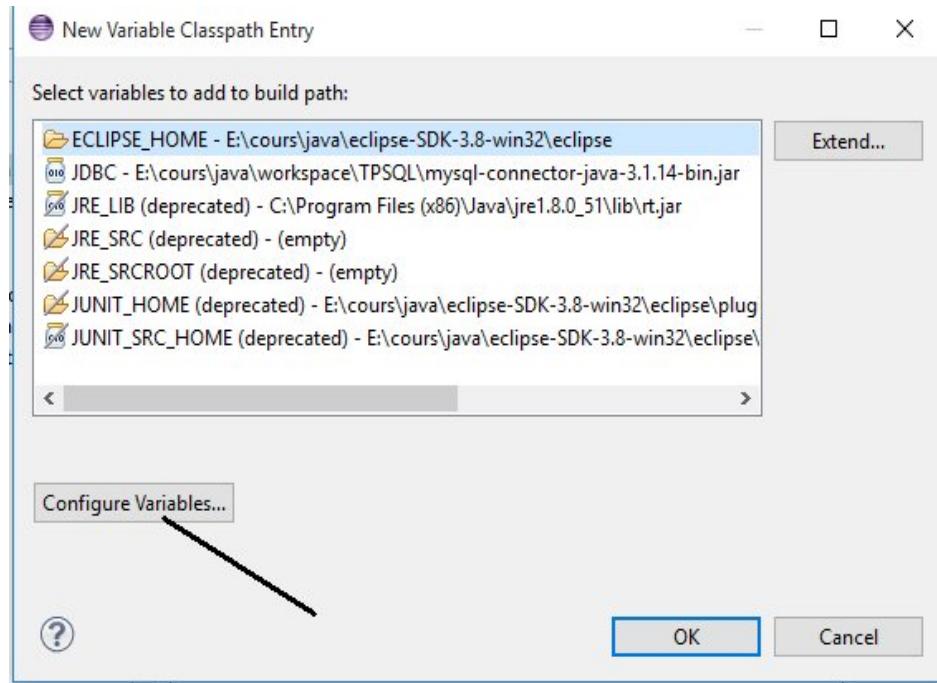
Par la suite on va inclure notre driver JDBC au Classpath de notre projet. Cliquez droit sur le nom du projet et properties



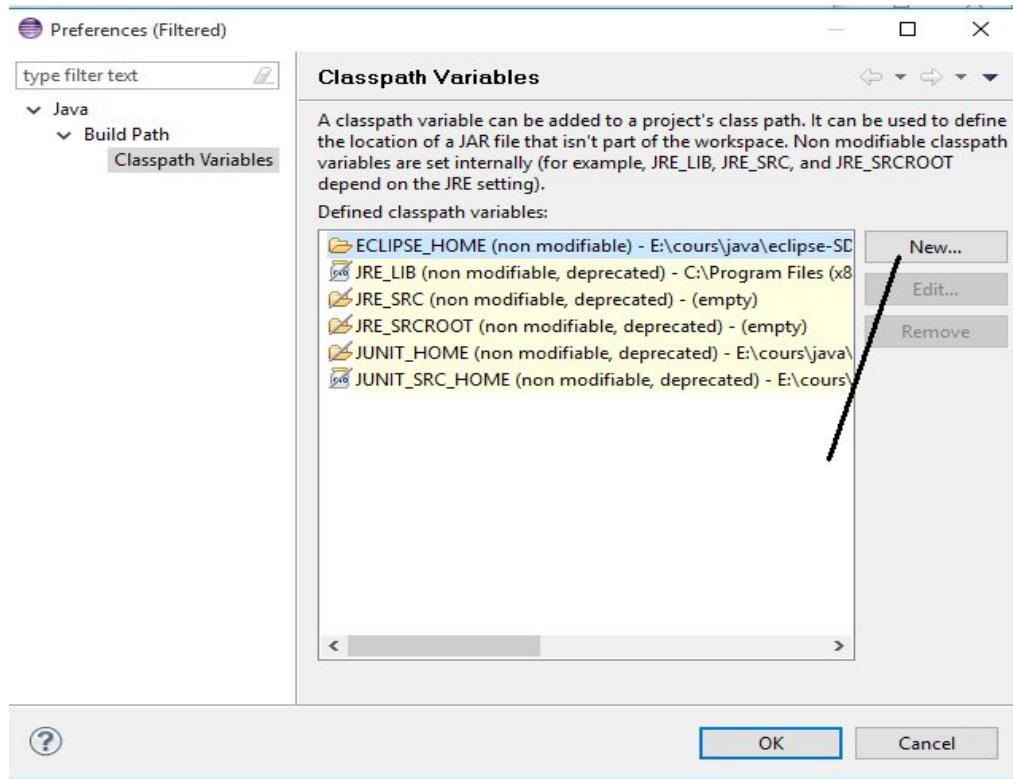
Puis java build path et cliquez sur add Variable



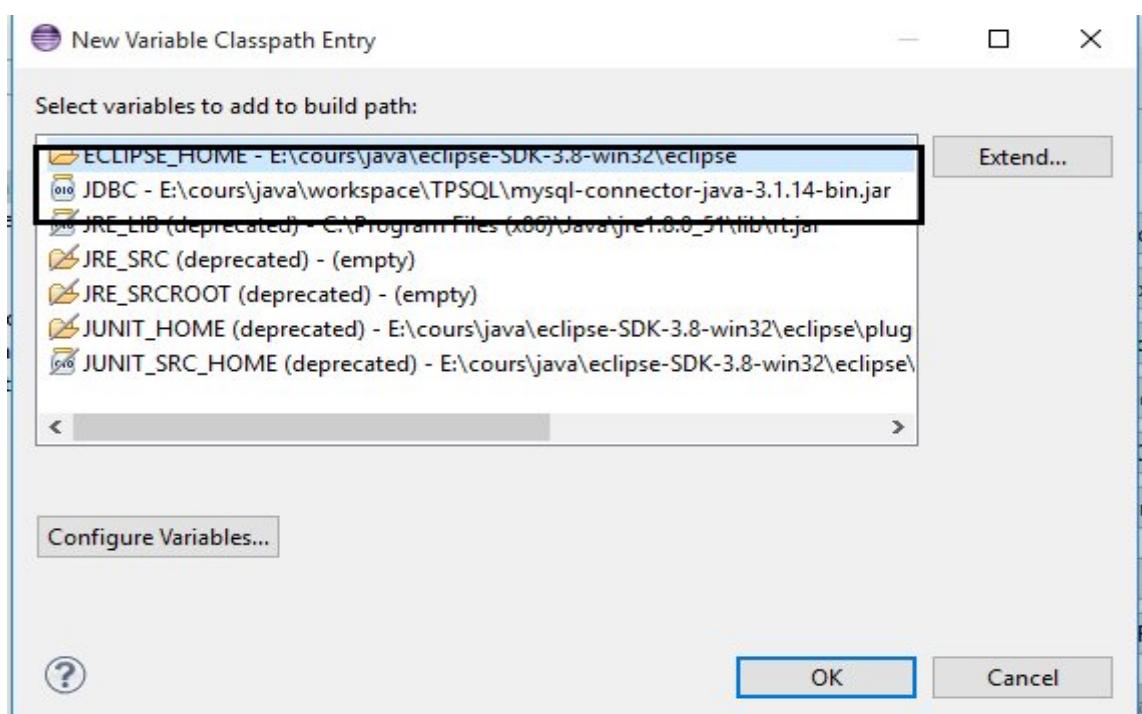
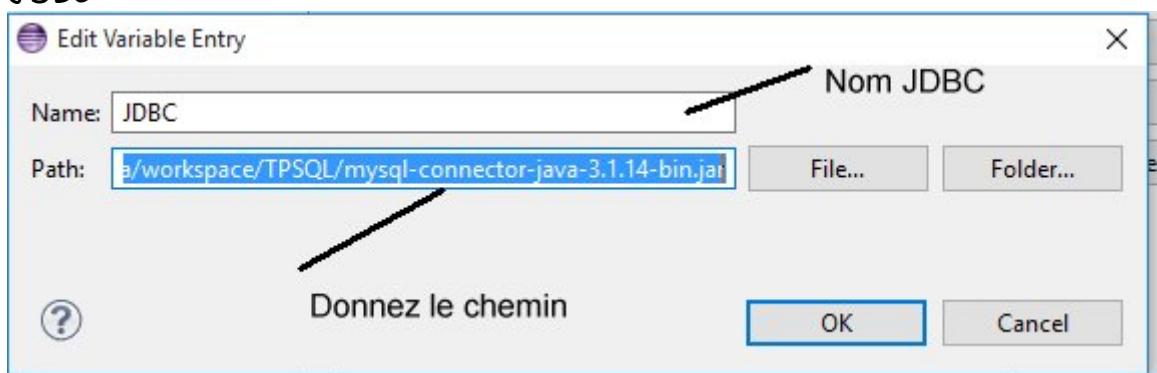
Cliquez sur Configure Variables



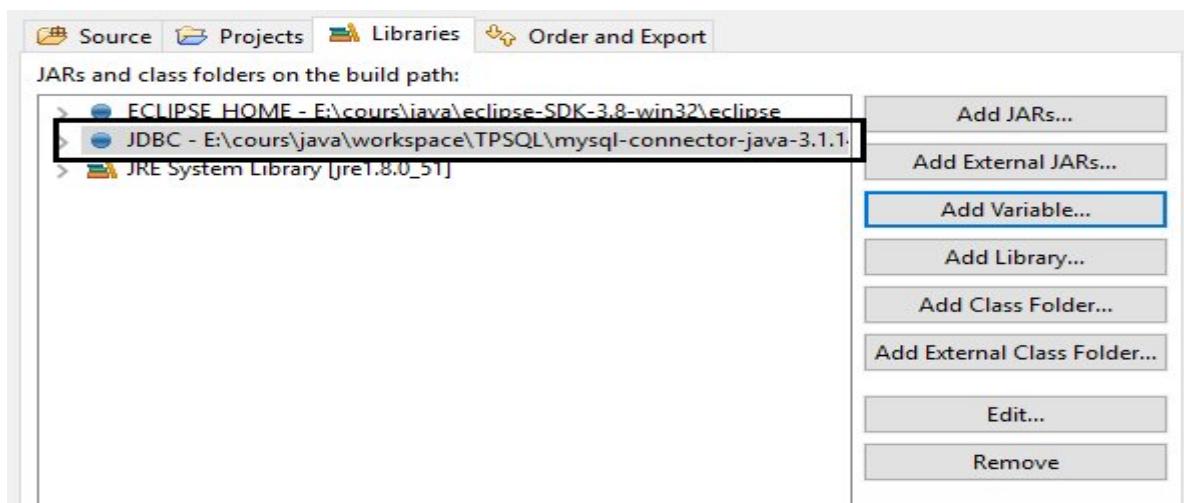
Cliquez sur new



Donnez le nom que vous voulez pour désigner la nouvelle variable , fournir le chemin du driver JDBC



Double cliquez sur notre variable pour l'intégrer à notre projet :

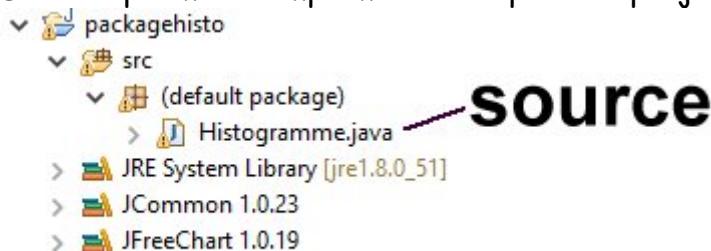


Création d'une librairie JAR

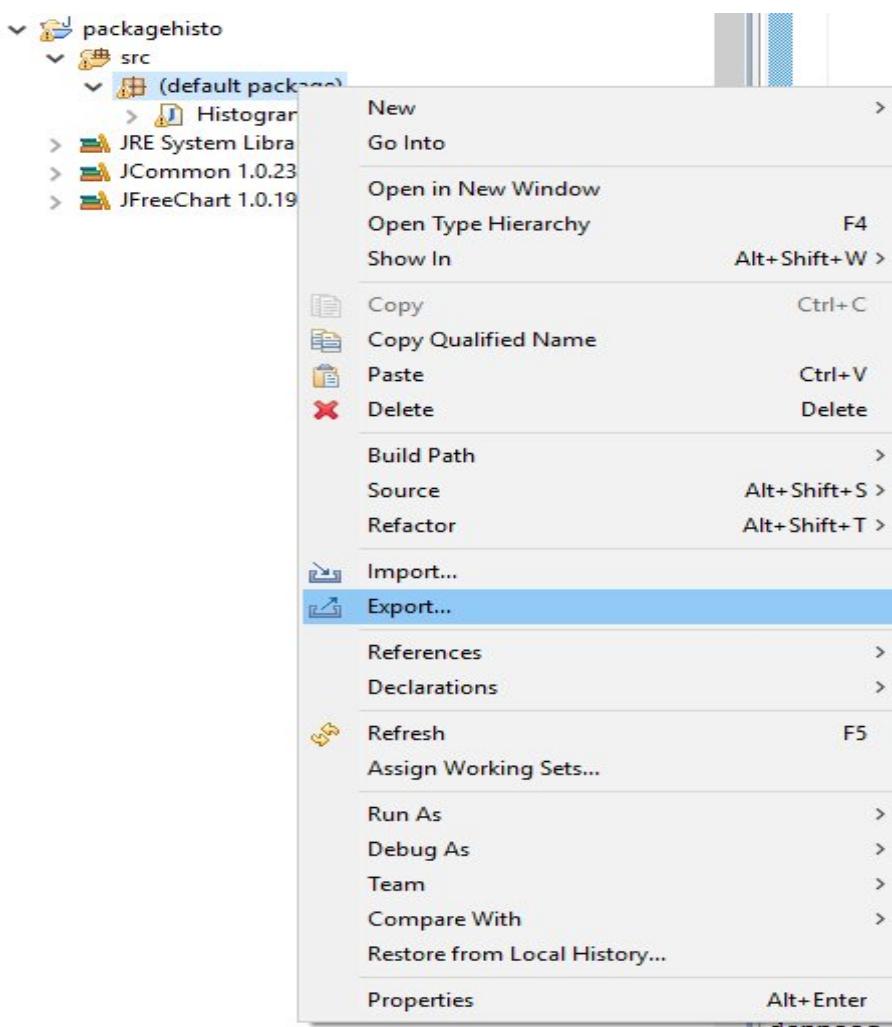
Il est parfois intéressant dans un projet de réutiliser des classes réalisées précédemment sans être obligé de fournir le code source. On peut alors compresser cet ensemble de classe dans une librairie sous la forme d'un fichier Jar non exécutable.

1) Création du fichier JAR

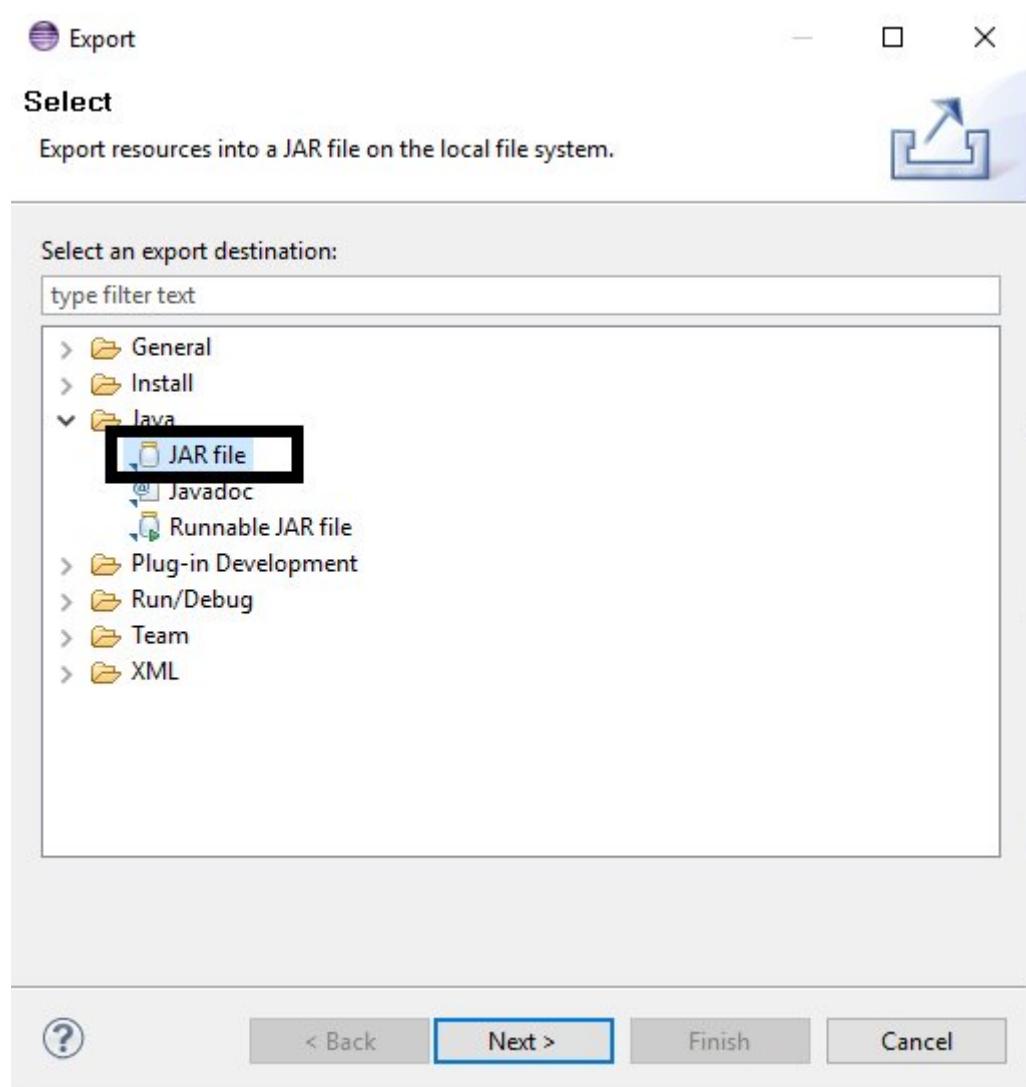
Dans un premier temps mettre en place le projet sous eclipse avec le fichier source :



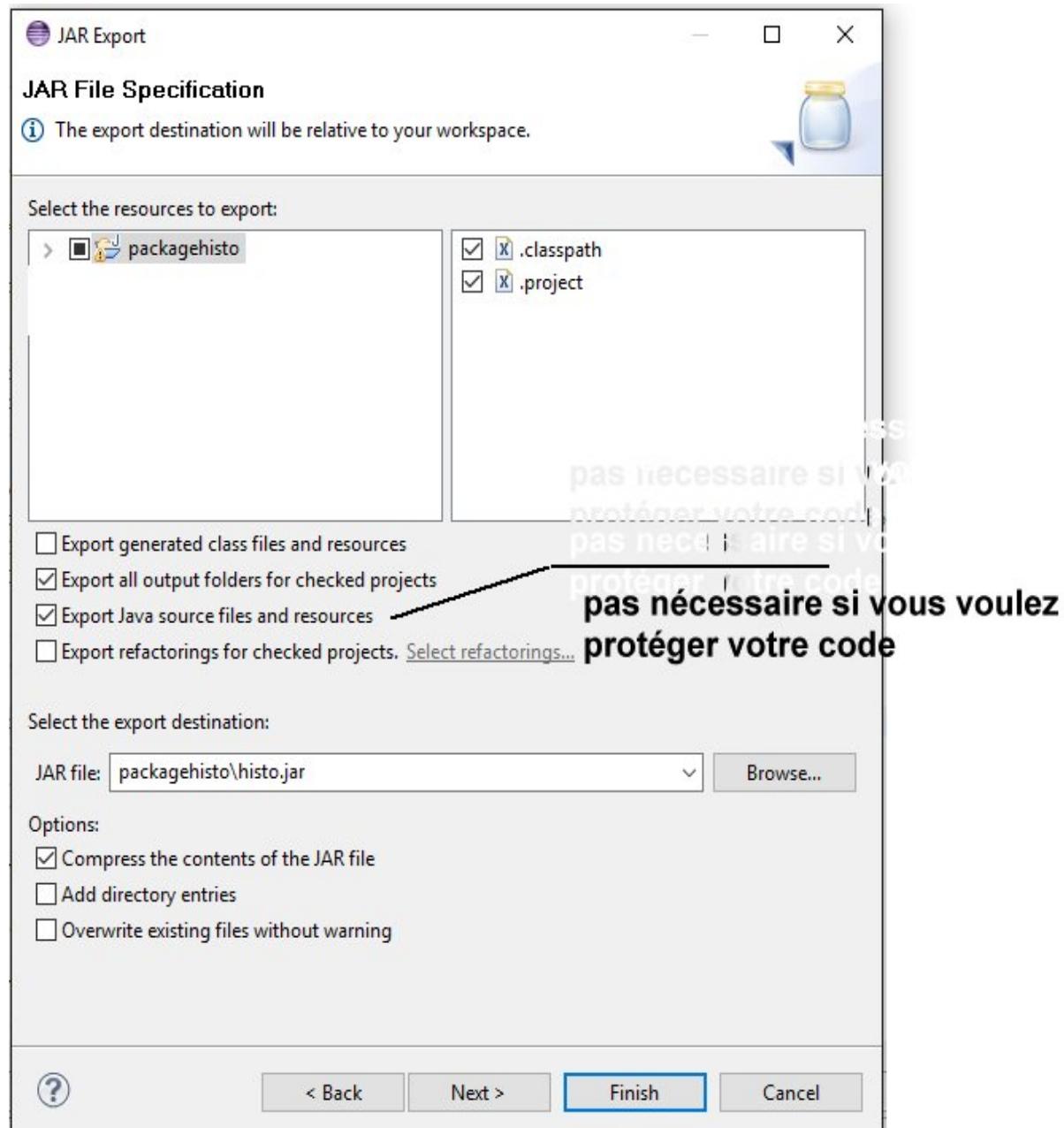
Cliquez droit sur le paquetage puis export :



Choisir un fichier JAR

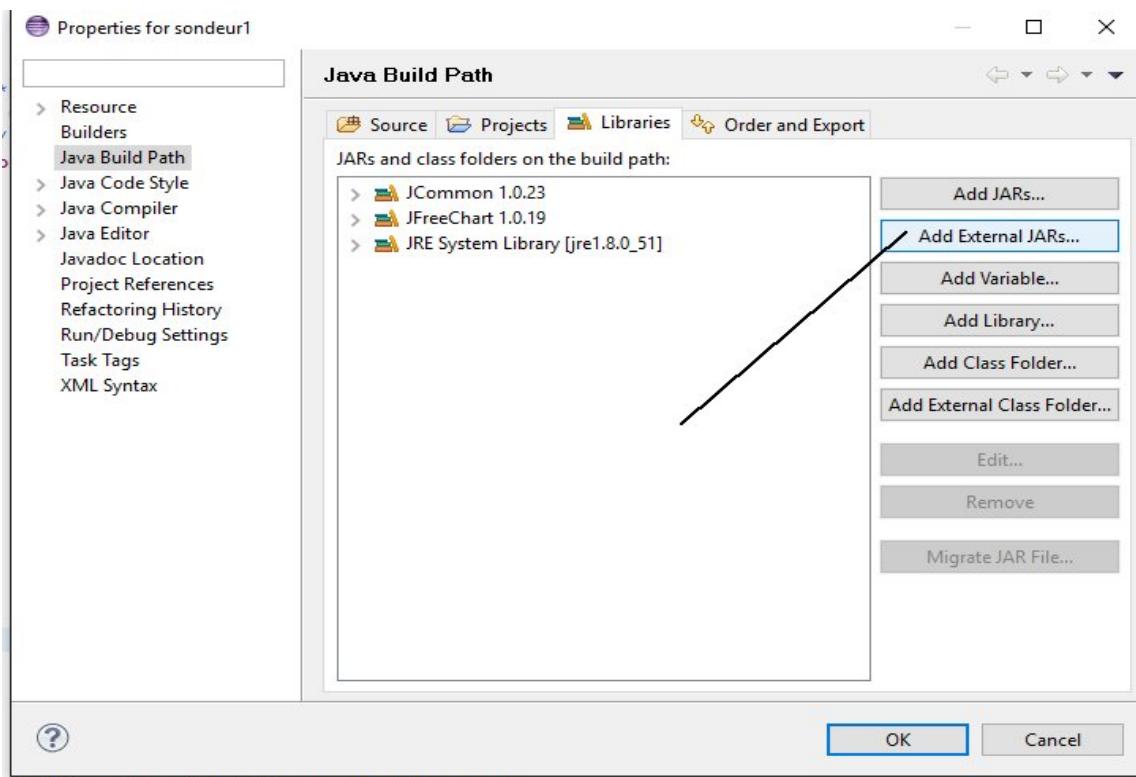


Sélectionner les éléments suivants pour la création du fichier JAR :

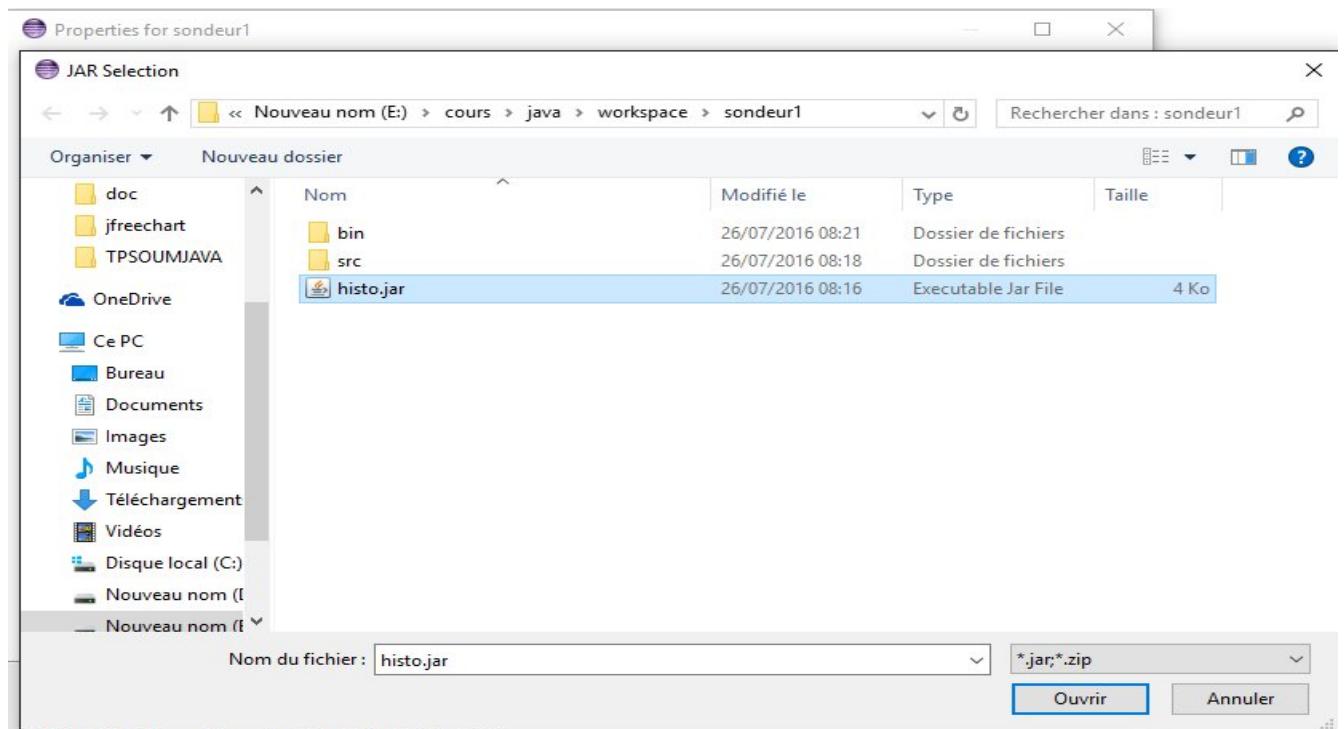


2) Intégration du fichier JAR dans notre projet

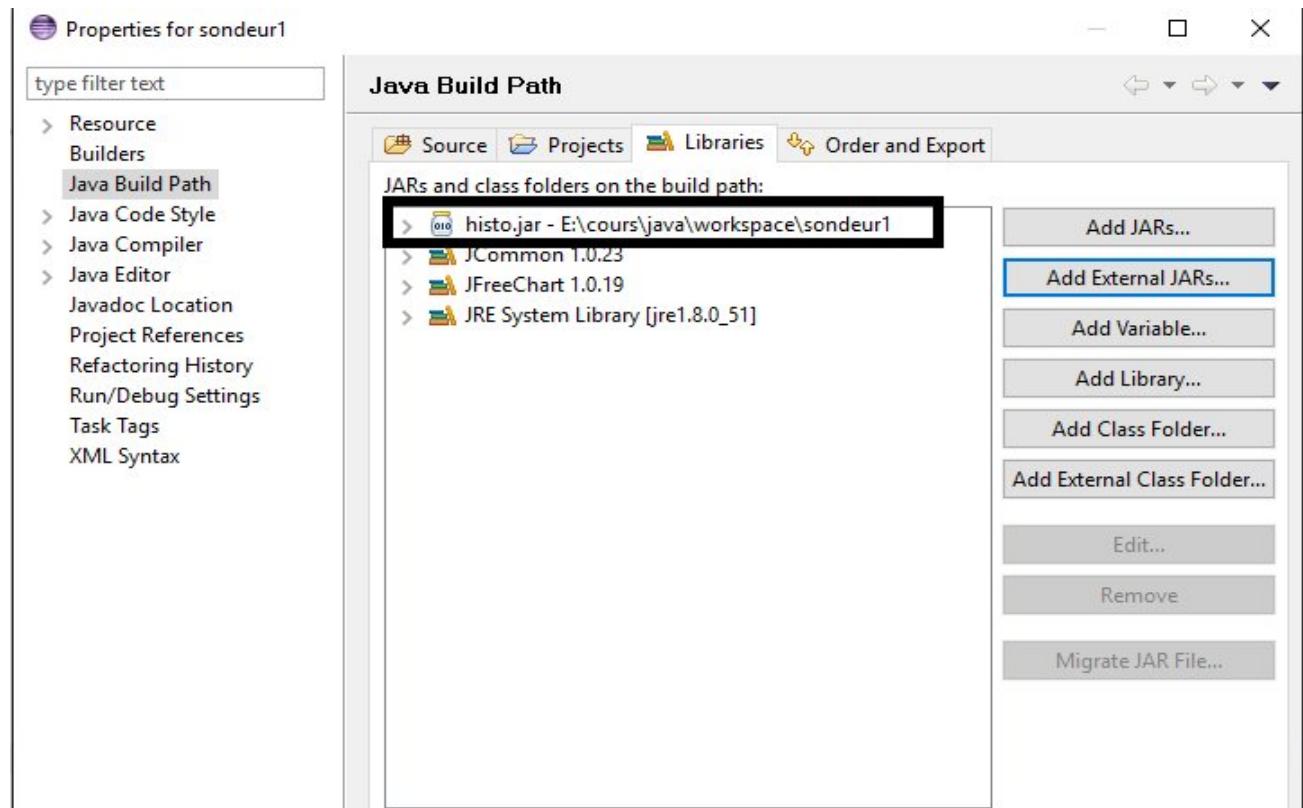
Sélectionner un fichier Jar externe



Fournir le chemin où il se situe



Notre librairie est intégrée dans notre projet



Comment lancer un script au démarrage de la PI

Mettre en place votre script dans le répertoire de votre choix. Ce script permettra de lancer un programme au démarrage

`sudo nano essai.sh`

`#!/bin/bash`

```
sudo java -Djava.library.path=/usr/lib/jni/ -cp  
/usr/share/java/RXTXcomm.jar:/home/pi/servermlv:. ServerTIM
```

Enregistrez votre fichier script

Donnez tous les droits au scripts :

`Chmod 777 essai .sh`

Editez le fichier /etc/rc.local

`sudo nano essai.sh et rajoutez la ligne suivante avant le exit 0 :`

```
fi  
su - pi -c "/home/pi/servermlv/TIM.sh &"  
exit 0
```

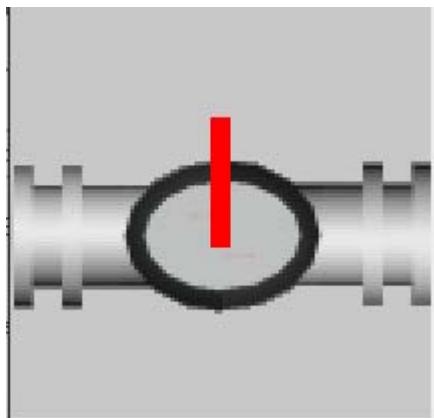


Quelques composants personnels

En JAVA

Les composants

Projet vanne



Constructeur

```
public Panneauvanne(boolean etat1)
```

etat1=true ouvert

etat1=false fermé

méthodes

```
public void setniveau(boolean etat1)
```

Dimension d'origine

(0, 0, 210, 210)

Projet cadran



Constructeur

```
public Panneaucadran(int angle, String chaine)
```

int angle : angle en degré de 0 à 90

String chaine : chaîne affichée en bas du cadran

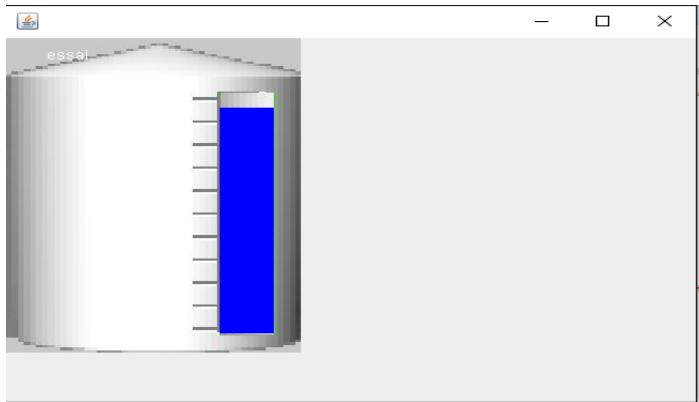
méthodes

```
public void setniveau(int angle, String chaine)
```

Dimension d'origine

(0, 0, 200, 117)

Projet cuve



Constructeur

```
Panneaucuve(int nivcuve, int nivcuvemax, int seuill, int seuil2)
Int nivcure : niveau courant de la cuve
Int nivcuvemax : niveau maximum de la cuve
Int seuil : seuil en % à partir duquel la cuve est presque vide (couleur jaune)
Int seuil : seuil en % à partir duquel le niveau de la cuve est critique (couleur rouge)
```

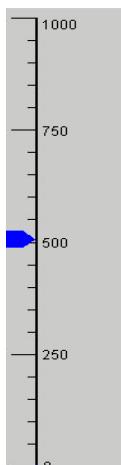
méthodes

```
public void setniveau(int nivcuve, int nivcuvemax, int seuill, int seuil2)
```

Dimension d'origine

```
(0, 0, 218, 305);
```

Projet jauge



Constructeur

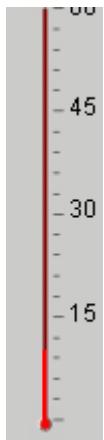
```
public Panneaujauge(int hauteurmax, int hauteur)
```

méthodes

```
public void setniveau(int hauteurmax, int hauteur)
```

Dimension d'origine (0, 0, 100, 441)

Projet jauge température



Constructeur

```
Panneaujaugetemp(int hauteurmax, int hauteur)
```

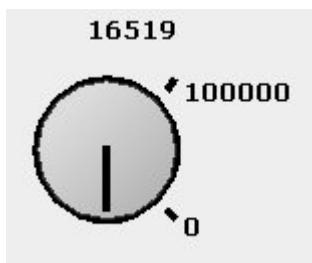
méthodes

```
public void setniveau(int hauteurmax, int hauteur)
```

Dimension d'origine

```
(0, 0, 60, 225)
```

Projet potentiomètre ou knobtest



Constructeur

```
Potentiometre(double limite) ;
```

```
Double limite : valeur maximale de la gamme ;
```

méthodes

```
public void setvaleur(double value) ;
```

```
double value : position du curseur du potentiomètre
```

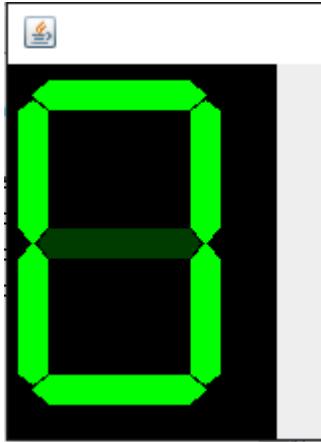
```
public double getvaleur() ;
```

```
double : position du curseur du potentiomètre
```

Dimension d'origine

```
(0,0,170,170)
```

Projet LCD



Constructeur

```
public SevenSegmentDisplay()
```

méthodes

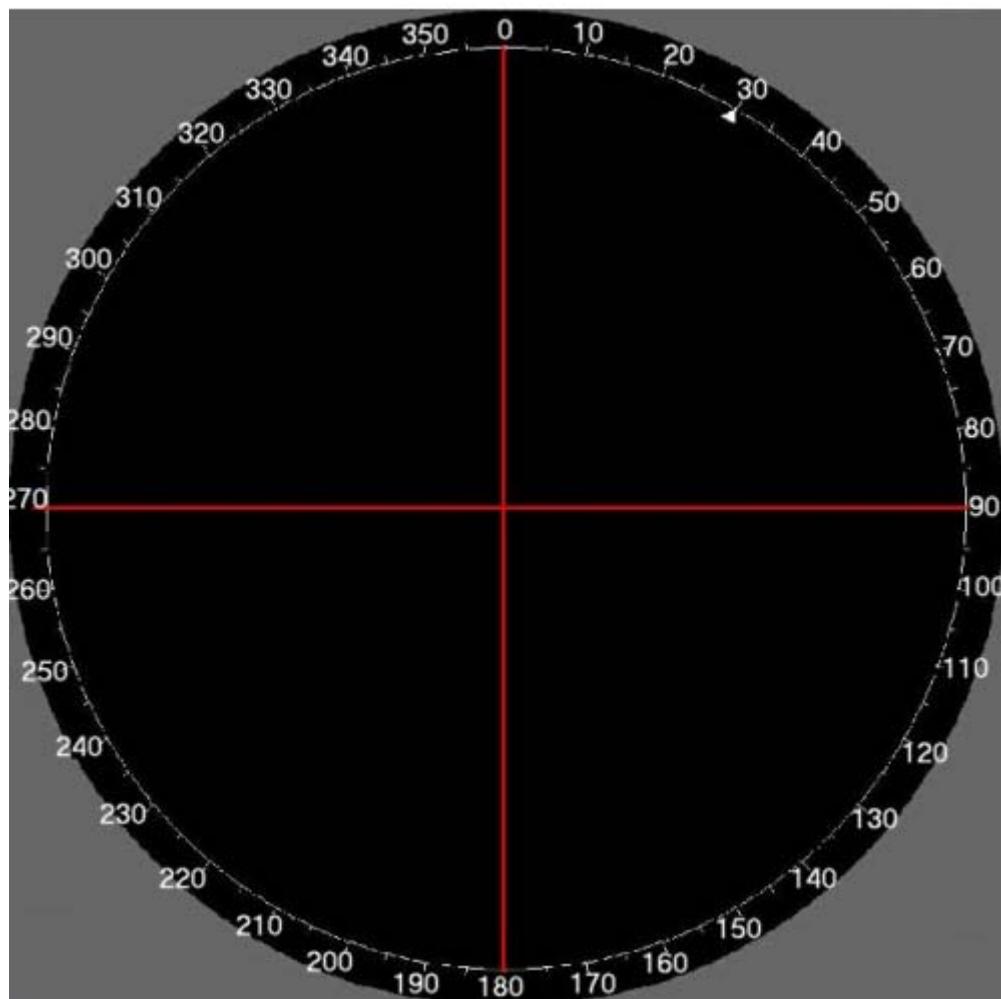
```
public void writeNumber(int n)
```

```
public void affiche(int a)
```

Dimension d'origine

(0, 0, 60, 225)

Projet scope pour AIS



Constructeur

```
public Panneauscope(IHMAIS I2)// IHMAIS2 fenêtre principale
méthodes
public void setrayon(float rayon)// valeur du rayon en mètre de 3000m à 30000m
```

Dimension d'origine

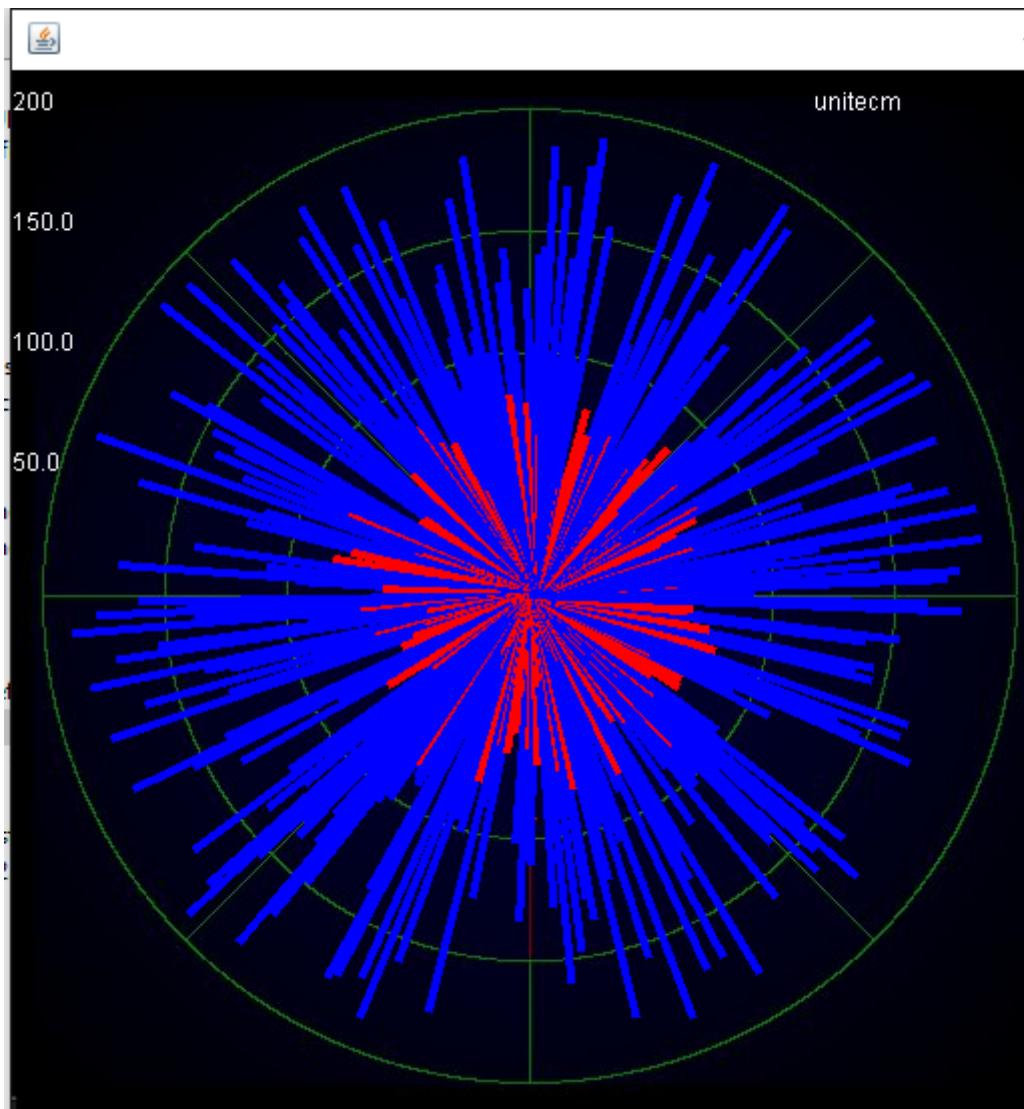
0, 0, 500, 500

Pour alimenter le scope on doit fournir un tableau de piste dynamique.

```
public ArrayList<PisteAIS> tabpiste;// tableau dynamique du type piste
tabpiste = new ArrayList<PisteAIS>();// création du tableau dynamique
```

Chaque piste a le format suivant :

```
public class PisteAIS extends Cartesien{  
protected float speed;  
protected String MMSI;  
protected long MMSIint;  
protected float longitude;  
protected float latitude;  
protected int cap;  
protected int type;  
  
public PisteAIS(long MMSIint, float vitesse, float longitude, float lat,int cap, int  
type)  
}  
  
pistel= new PisteAIS(518908000,0.0f,5.9f,43.043250f,12,3);  
tabpiste.add(pistel);
```

Projet scope1 pour détection environnement sonarConstructeur

```
public Panneauscope1(String unite, faisceau[] tabfaisceau1 ,int hauteur, boolean bord,  
int seuil)
```

```
String unite // légende
```

```
int hauteur // rayon de détection
```

```
boolean bord // non utilisé true par défaut
```

```
Int seuil // seuil de détection minimal à partir duquel le faisceau devient rouge
```

```
faisceau[] tabfaisceau1// tableau statique d'objet du type faisceau
```

méthodes

```
setniveau(String unite, faisceau[] tabfaisceau1 , int hauteur)
```

création de l'objet Panneauscope

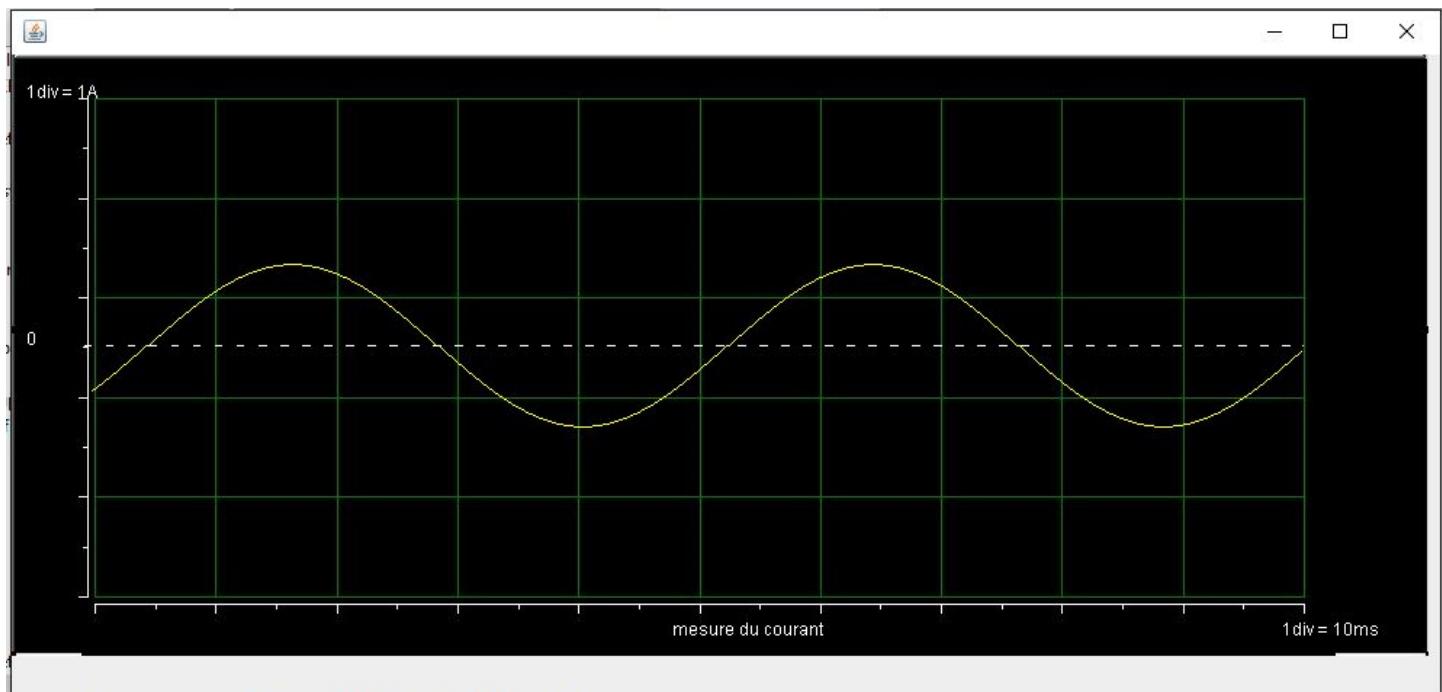
```
pan = new Panneauscope1("unite cm",tabfaisceau,200, true,100);
```

Dimension d'origine

0 , 0 , 514 , 522

```
public class faisceau {  
    private int angle;  
    private double distance;  
    public faisceau() {  
        // TODO Auto-generated constructor stub  
    }  
    public int getangle(){  
        return this.angle;  
  
    }  
    public double getdistance(){  
        return this.distance;  
  
    }  
    public void setdistance(double distance){  
        this.distance=distance;  
  
    }  
  
    public void setangle(int angle){  
        this.angle=angle;  
  
    }  
}
```

Projet oscilloscope



Constructeur

```
public Panneauoscillo( int size, String abscisse, String ordonnees, String title)

int size // dimension du tableau de données à afficher
String abscisse, // texte à afficher sur l'axe des abcisses
String ordonnees // texte à afficher sur l'axe des ordonnées
String title // texte du titre de la courbe
```

Méthodes

```
public void envoidonnée(int tableau[])

int tableau[] // donnée à afficher La dimension doit correspondre à celle du constructeur

création de l'objet Panneauscope

pan = new Panneauoscillo(1,"1div = 10ms","1div = 1A","mesure du courant");
```

Dimension d'origine

0, 0, 976, 414)