Architecture du Framework

Java

# Présentation

A la base, le Framework « Slick » est employé pour développer des applications, il a été conçu autour de la librairie LWJGL, afin de rendre cette dernière plus ergonomique, plus simple à utiliser, et permet une meilleure organisation du code.

Lightweight Java Game Library (LWJGL) est une librairie Java open source pour des développeurs de jeu vidéo.

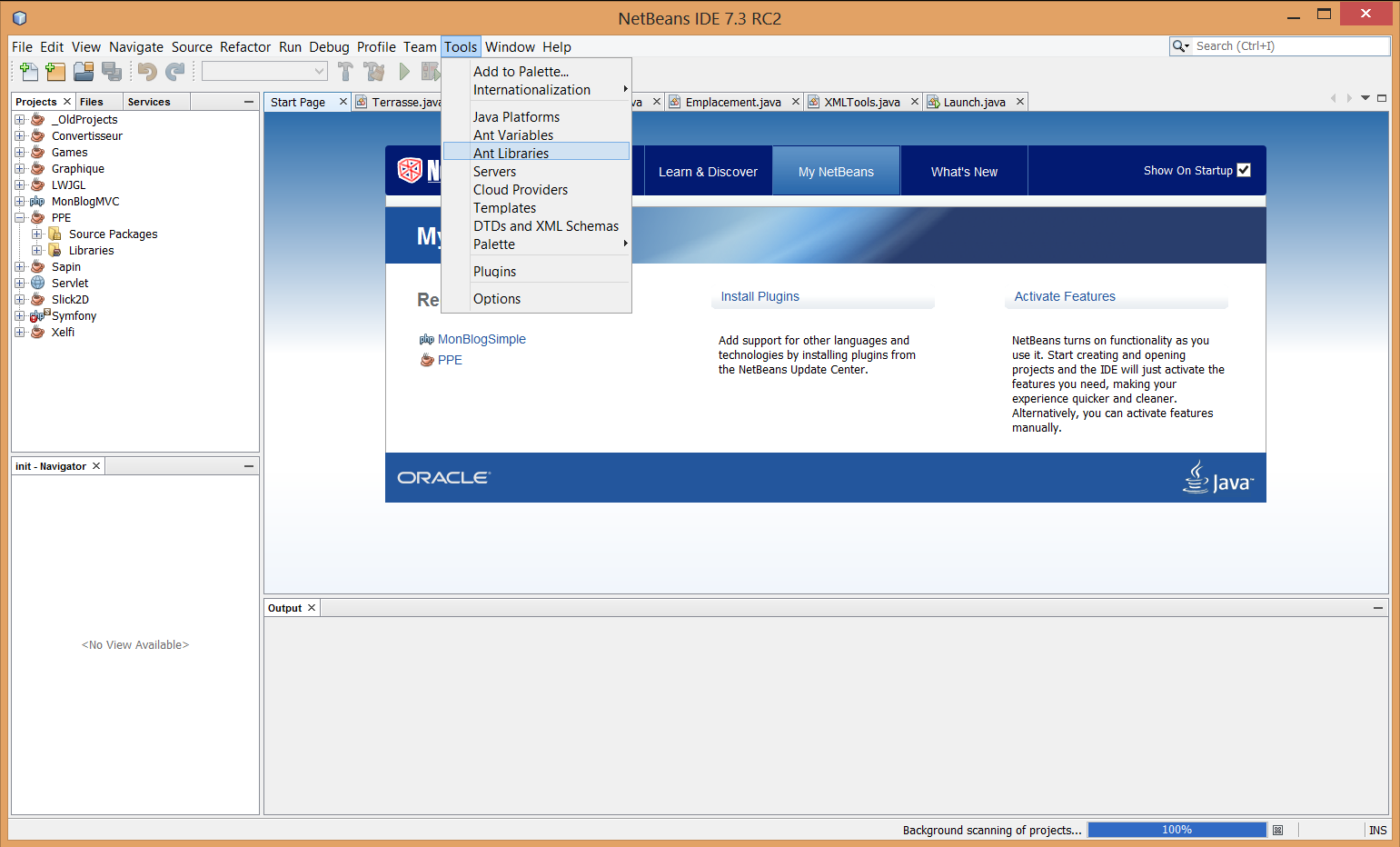
LWJGL expose des bibliothèques multiplate-formes à haute performance généralement utilisées dans le développement de jeux logiciels et des titres multimédia. Il comprend OpenGL (Open Graphics Library), OpenCL (Open Computing Language) et OpenAL (Open Audio Library) et permet l'accès aux contrôleurs comme gamepads et des leviers de commande d'une façon neutre de plate-forme.

Le but principal du projet est de fournir une technologie qui permet aux développeurs Java d'obtenir l'accès aux ressources qui sont autrement indisponibles ou mal mises en œuvre sur la plate-forme de Java existante.

# Installation

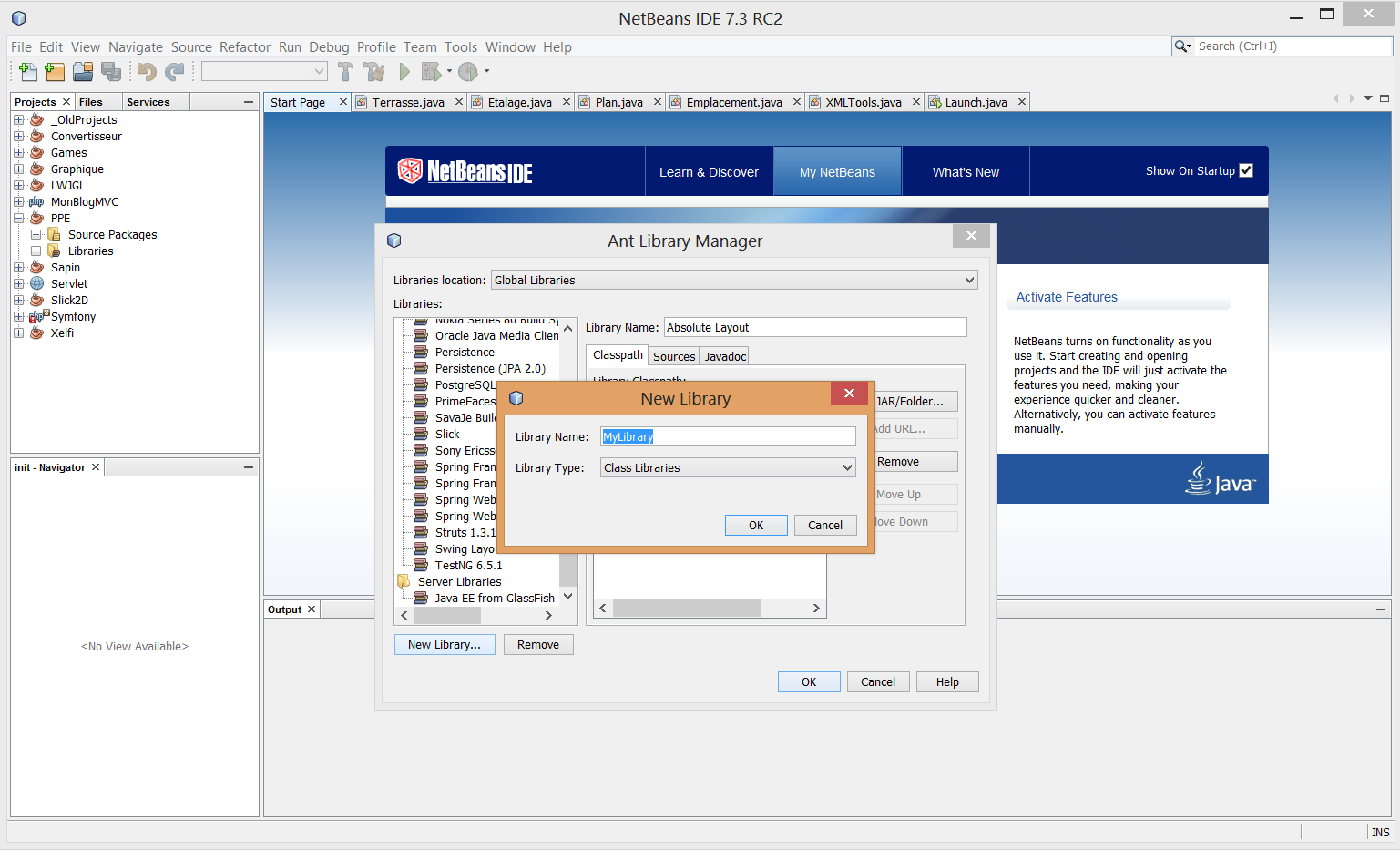
Sous NetBeans, pour utiliser le Framework dans un projet, il faut tout d’abord créer les deux bibliothèques, l’une pour LWJGL, et l’autre pour Slick.

Dans NetBeans, aller dans Tools (Outils) puis Ant Libraries



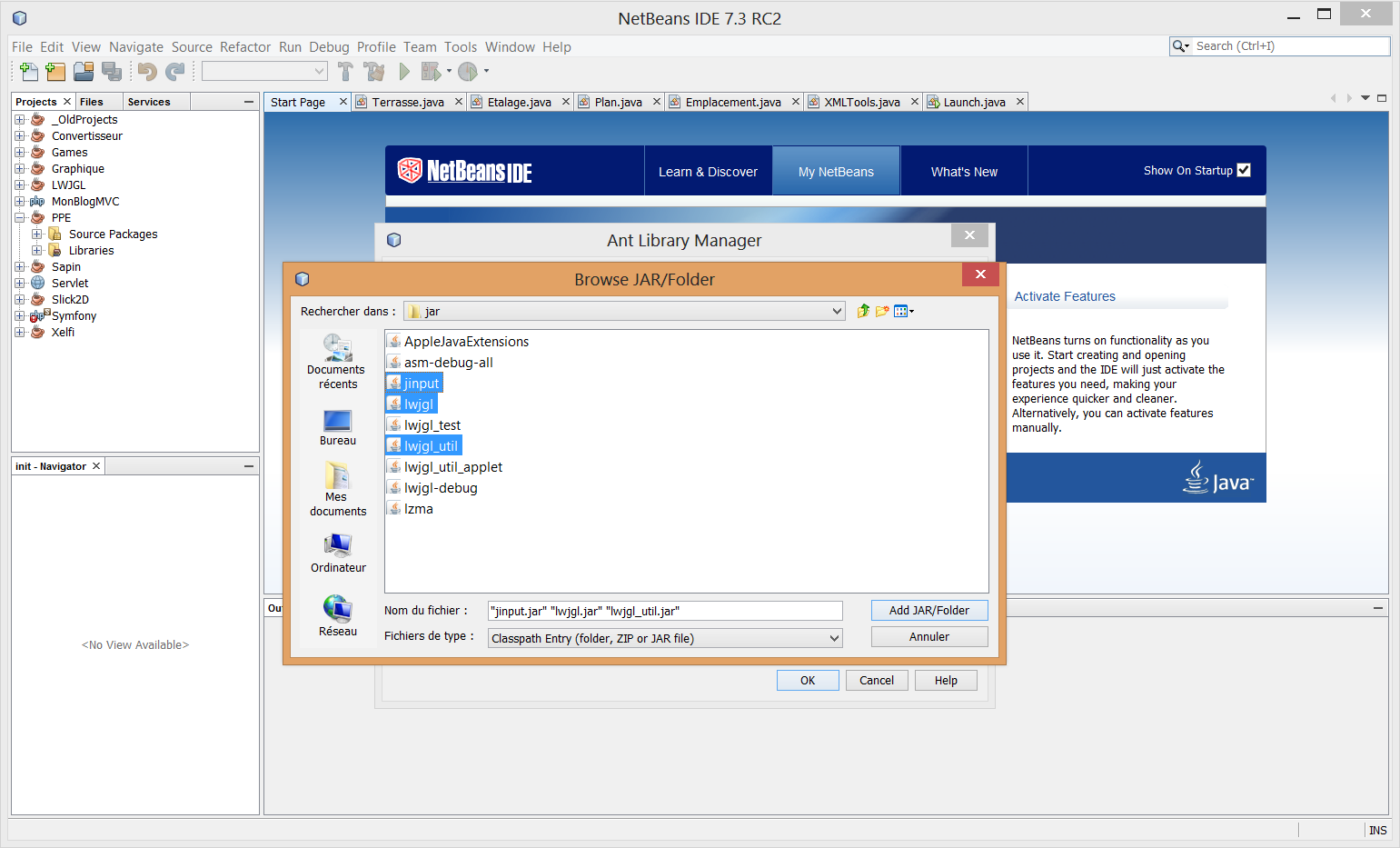
Une liste de toutes les librairies déjà installées sous NetBeans apparait. Pour créer nos deux librairies, il faudra cliquer sur le bouton « new Library… » et lui donner un nom.

Nos deux librairies s’appellent LWJGL et Slick.

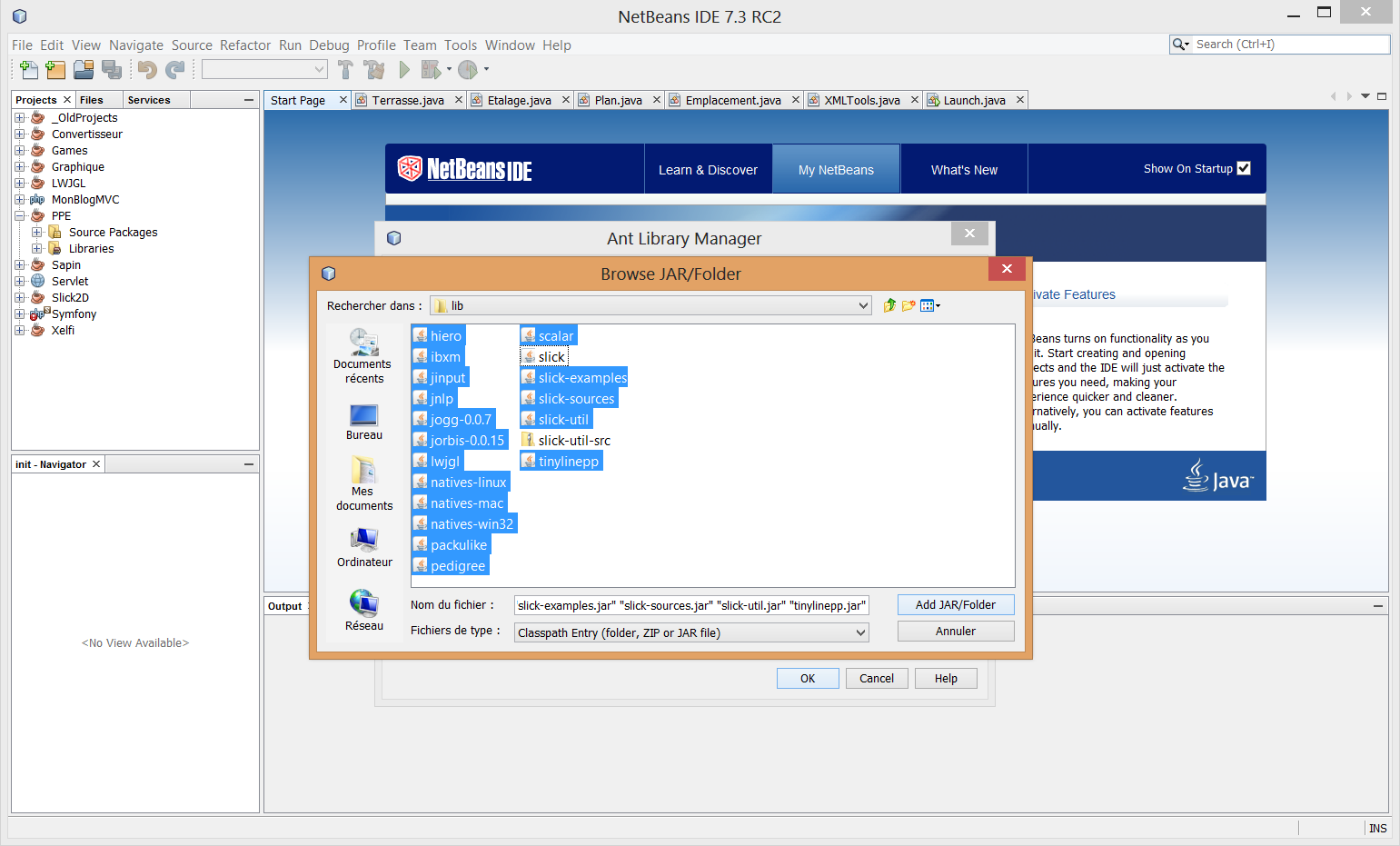


Une fois validée, il faut affecter les fichiers jar au classpath de nos nouvelles librairies, pour se faire, cliquer sur « add jar/folder » et naviguer jusqu’aux dossiers respectifs.

Pour LWJGL, dans le dossier lwjgl/jar



Pour Slick, dans le dossier slick/lib



Lorsque cela est fait, il suffit d’intégrer les deux librairies dans le projet, ainsi que le fichier slick/lib/slick.jar

Note : Dans le projet actuel, les librairies ont déjà été intégrer, leur création résoudra les erreurs liées au fait que NetBeans ne retrouve pas les librairies. Pour le fichier slick.jar, il faut vérifier et modifier le chemin d’accès si celui-ci n’est pas au même emplacement.

De même que dans l’onglet « run » des propriétés du projet, il faut vérifier que VM Option contient le bon chemin :

-Djava.library.path="%chemin absolu du dossier%\lwjgl-2.8.5\native\windows"

# Utilisation

## Class « StateBasedGame »

Pour utiliser le Framework, il faut tout d’abord créer une première classe que l’on appelle usuellement « Launch » ou « App » qui héritera de la classe StateBasedGame.

C’est la base de l’application, le conteneur de tous les états de l’application. C’est la fenêtre où tout va se dérouler.

une méthode est implémentée obligatoirement : initStatesList

dans cette méthode, on y mettra tous les états présent dans l’application (notre application n’en comprend qu’un seul) et au lancement, l’application lira la première en liste.

Pour naviguer entre les états, la méthode « enterState(int ID) » sera utilisée.

Les états sont des classes héritées de la classe « BasicGameState » et doit obligatoirement contenir un attribut public static final int ID qui doit être **unique** entre chaque état !

## Class « «BasicGameState »

C’est la deuxième classe mère de l’application, celle qui va définir un état. Chaque état est indépendant l’un de l’autre, lorsque l’on quitte un état, celui-ci n’existe plus, et ne sera plus mis à jour.

Trois méthodes essentielles sont a implémenter obligatoirement

### public void init(GameContainer container, StateBasedGame game)

Cette méthode permet d’instancier tous les éléments nécessaires au bon fonctionnement de l’état. Un tableau de donnée, l’affichage d’image etc...

### public void render(GameContainer container, StateBasedGame game, Graphics g)

Ici, il sera surtout question de gérer l’affichage de tous les éléments présent dans l’application. Images, tableau, objet…

Pour une meilleure gestion du code, il est donc intéressant, par exemple, d’intégrer une méthode « render » ou « afficher » dans les classes métiers présentes pour n’avoir simplement qu’à les appeler ici.

### public void update(GameContainer container, StateBasedGame game, int delta)

C’est sans doute la méthode la plus importante de l’état, celle qui va mettre à jour toutes informations. Par exemple, lorsque l’on appuie sur la toucher « Entrer » ou « suppr » ou encore sur « Ctrl + C » et « Ctrl + V » etc… là encore, un avantage du Framework est d’intégrer une méthode de mise à jour dans les classes métiers pour mieux gérer son code.

Bien sûr, il est tout à fait possible d’intégrer d’autres méthodes dans l’état, et de les réutiliser dans le code.

L’intérêt majeur de ce Framework est justement la simplicité de gestion de l’affichage (ne serait-ce pour les images et autre objets) et donc le développeur peut se concentrer sur le mécanisme entre ses classes métier sans devoir s’inquiéter de la mise en forme graphique. Basée sur le jeu video, elle permet également de dessiner plus facilement des formes géométriques, telles que des cercles, triangles et autres polynômes.