



Javora

Rapport de projet



Jérémy VARANGE, Yohan BORDES, Noah LACASTE



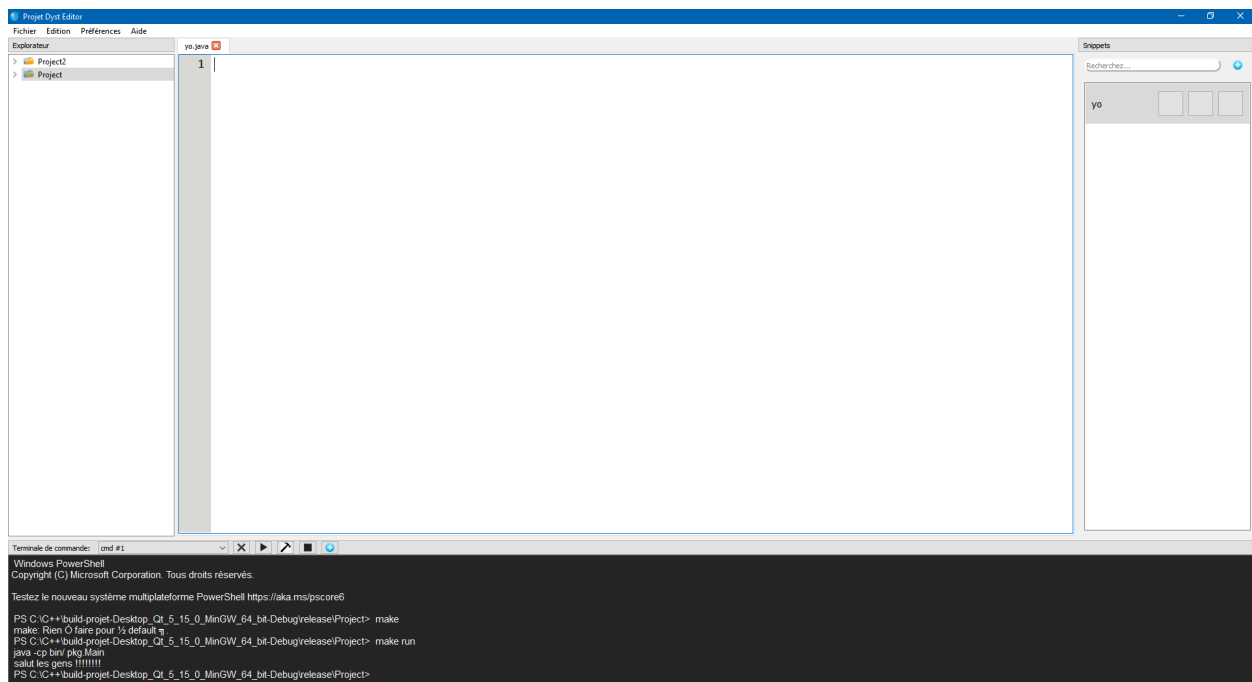
Remerciements

Merci à Pierre VALARCHER pour avoir accepté d'être notre tuteur, ainsi qu'à toute l'équipe enseignante, pour leur suivi durant ces deux années.

Jérémy VARANGE, Yohan BORDES, Noah LACASTE

Synthèse

Ce rapport présente Javora, un environnement de développement (IDE), un logiciel permettant d'éditer du texte, compiler du code et un ensemble d'outils qui permettent d'augmenter la productivité des programmeurs.



(IDE - Thème clair)

L'éditeur de texte met à disposition des options telles que de la colorisation syntaxique, des double parenthèses/accolades/guillemets. Des options de sélection et raccourcis clavier comme le copier coller, sélectionner tout,... La possibilité de chercher un mot et de le remplacer. Enfin, il y a la possibilité d'utiliser des snippets

La console permet de naviguer dans ses fichiers et de compiler son code avec notamment quelques boutons qui facilitent le processus.

Enfin l'ide permet de créer des projets, packages et fichiers ainsi que de générer le makefile d'un projet. Aussi l'on peut modifier, sauvegarder et supprimer ces mêmes entités depuis l'IDE.

Jérémy VARANGE, Yohan BORDES, Noah LACASTE



Code source

Le code source de l'application est disponible à l'adresse suivante :

Exécution

Pour lancer le programme il faut avoir à disposition Qt 5.

| | |
|----------------------------------------------------|-----------|
| Remerciements | 1 |
| Synthèse | 2 |
| Code source | 3 |
| Exécution | 3 |
| 1. Présentation du projet | 5 |
| 1 - 1 Point de départ | 5 |
| 1 - 2 Membres | 5 |
| 1 - 3 Réalisation | 6 |
| 1 - 4 Fonctionnalités supplémentaires | 7 |
| 1 - 5 Fonctionnalités à venir | 7 |
| 2. Présentation de l'éditeur de texte | 8 |
| 2 - 1 Coloration syntaxique | 8 |
| 2 - 2 Chercher un mot | 9 |
| 2 - 3 Remplacer un mot | 11 |
| 2 - 4 Onglets | 11 |
| 2 - 5 Double caractère | 12 |
| 2 - 6 Surbrillance double caractères | 12 |
| 2 - 7 Indentation | 13 |
| 2 - 8 Option/Raccourcis clavier | 13 |
| 3. Présentation de l'explorateur de fichier | 15 |
| 3 - 1 Outils | 15 |
| 4. Présentation des options de l'IDE | 17 |
| 4 - 1 Fichier | 17 |
| 4 - 2 Edition | 22 |
| 4 - 3 Préférences | 22 |
| 5. Présentation des Snippets | 25 |
| 5 - 1 Barre des snippets | 25 |
| 6. Présentation de la Console | 26 |
| 6 - 1 Barre de la console | 26 |
| 6 - 2 Utilisation de la console | 27 |



1. Présentation du projet

1 - 1 Point de départ

L'idée de départ était de développer une application suffisamment complète et sérieuse afin que ce ne soit pas un projet quelconque mais une réelle démonstration de nos compétences, y compris pour notre avenir.

De plus, on voulait apprendre quelque chose de nouveau, comme un autre langage de programmation.

Après une période de réflexion le concept de faire un IDE a rapidement convaincu tout le groupe puisque l'on était tous familiarisé avec le concept et l'on savait qu'il y avait des choses intéressantes à apprendre de cela.

1 - 2 Membres

Le groupe est composé de trois personnes :

- Jérémy VARANGE
- Yohan BORDES
- Noah LACASTE

Le tuteur du projet est l'enseignant Mr. Pierre Valarcher de l'IUT Sénart-Fontainebleau (77 - Seine et Marne).

Ce projet a été réalisé dans le cadre des Projets tutorés de 2ème année de DUT Informatique, commencé le 03 octobre 2020. Le responsable des projets tutorés est Mr. Pierre Valarcher.

Jérémy VARANGE, Yohan BORDES, Noah LACASTE

1 - 3 Réalisation

Une bonne partie des demandes du cahier des charges initial a été faite. Beaucoup de tâches ont été ajoutées pendant le développement.

L'éditeur est en grande partie finie, mais quelques fonctionnalités sont encore absentes.

Les spécifications ont une priorité selon le fonctionnement MoSCoW (ordre de priorités (Must Should Could Would) d'une tâche, de M les tâches à haute priorité jusqu'à W les tâches optionnelles).

| | | |
|---|---------------------------------------------|--------------------|
| M | Lecture de fichier | Fait |
| M | Ecriture de fichier | Fait |
| M | Compilation | Fait |
| S | Onglet de gestion de projet (explorateur) | Fait |
| S | Snippets | Fait |
| S | Coloration syntaxique | Fait |
| S | Auto complétion | Snippet uniquement |
| C | Raccourcis clavier | Fait |
| C | Génération de makefile | Fait |
| W | Transformation code en diagramme | Non fait |
| W | Synchronisation avec git | Non fait |

1 - 4 Fonctionnalités supplémentaires

Des spécifications ont été ajoutées au fur et à mesure du développement, selon les retours de notre tuteur, nos envies, les besoins des futurs utilisateurs et les retours de nos tests.

| | | |
|---|------------------------------|----------|
| S | Auto complétion des Snippets | Fait |
| S | Thème sombre | Fait |
| S | Bouton stop (console) | Non Fait |

1 - 5 Fonctionnalités à venir

| | |
|---|---------------------------------|
| S | Synchroniser avec git un projet |
| S | Auto complétion |
| W | Documentation |

2. Présentation de l'éditeur de texte

Cette partie présente tous les éléments et les fonctionnalités de l'éditeur.

2 - 1 Coloration syntaxique

L'éditeur de texte permet d'avoir une coloration syntaxique en fonction des expressions saisies dans l'éditeur. Ainsi certains les nombres, chaînes de caractères, fonctions, type et d'autres mots clé ont tous des colorations différentes.

A screenshot of a code editor window with a dark theme. The editor shows Java code for a binary tree. The code is syntax-highlighted: keywords like 'public', 'void', 'if', 'while', 'return', and 'static' are in red; identifiers like 'Node', 'focusNode', 'key', 'args', 'theTree', and 'Boss' are in green; strings like 'Boss', 'Vice President', 'Office Manager', 'Secretary', 'Sales Manager', and 'Salesman 1' are in yellow; and numbers like 50, 25, 15, 30, 75, and 85 are in blue. Comments are in grey. The code is as follows:

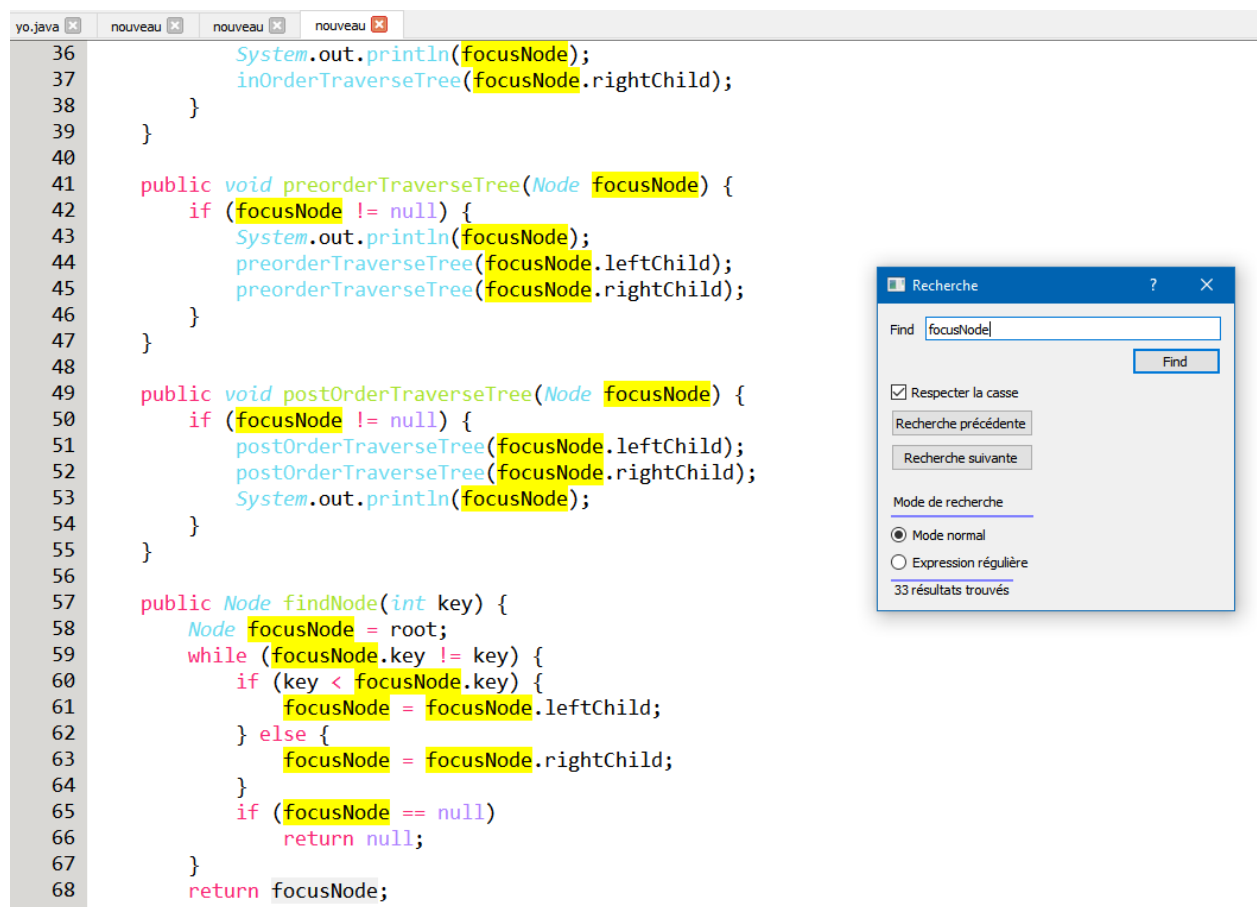
```
52
53 public void postOrderTraverseTree(Node focusNode) {
54     if (focusNode != null) {
55         postOrderTraverseTree(focusNode.leftChild);
56         postOrderTraverseTree(focusNode.rightChild);
57         System.out.println(focusNode);
58     }
59     this.cc = 20;
60 }
61
62 /*
63 ceci
64 est /*
65 un //
66 commentaire)
67 */
68
69 public Node findNode(int key) {
70     Node focusNode = root;
71     while (focusNode.key != key) {
72         if (key < focusNode.key) {
73             focusNode = focusNode.leftChild;
74         } else {
75             focusNode = focusNode.rightChild;
76         }
77         if (focusNode == null)
78             return null;
79     }
80     return focusNode;
81 }
82
83 public static void main(String[] args) {
84     BinaryTree theTree = new BinaryTree();
85     theTree.addNode(50, "Boss");
86     theTree.addNode(25, "Vice President");
87     theTree.addNode(15, "Office Manager");
88     theTree.addNode(30, "Secretary");
89     theTree.addNode(75, "Sales Manager");
90     theTree.addNode(85, "Salesman 1");
```

De plus cette coloration syntaxique n'a pas de problème à gérer certains cas compliqués comme des commentaires dans des commentaires ou des barres obliques dans des chaînes de caractères.

2 - 2 Chercher un mot

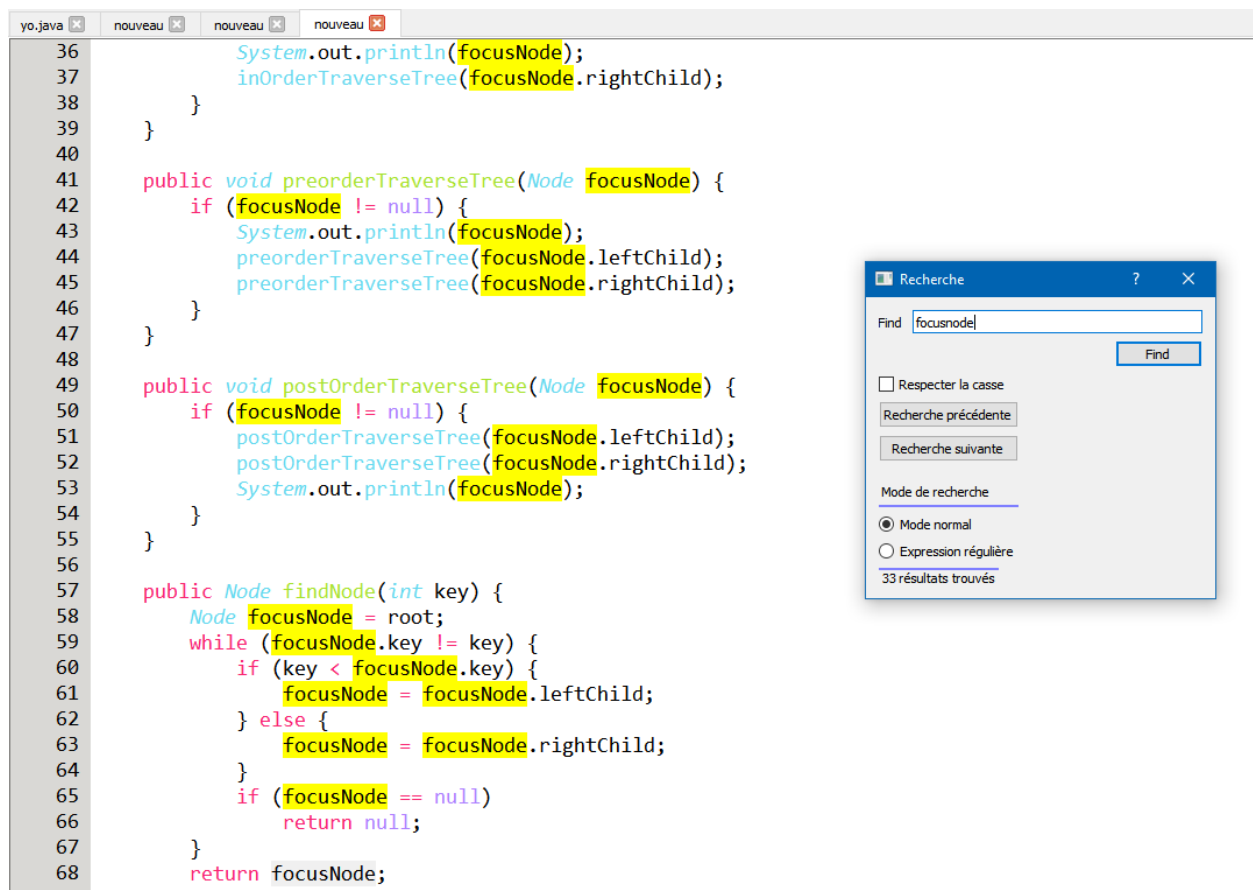
On peut chercher un mot ou une expression régulière dans n'importe quel fichier.

Une fenêtre apparaîtra alors :



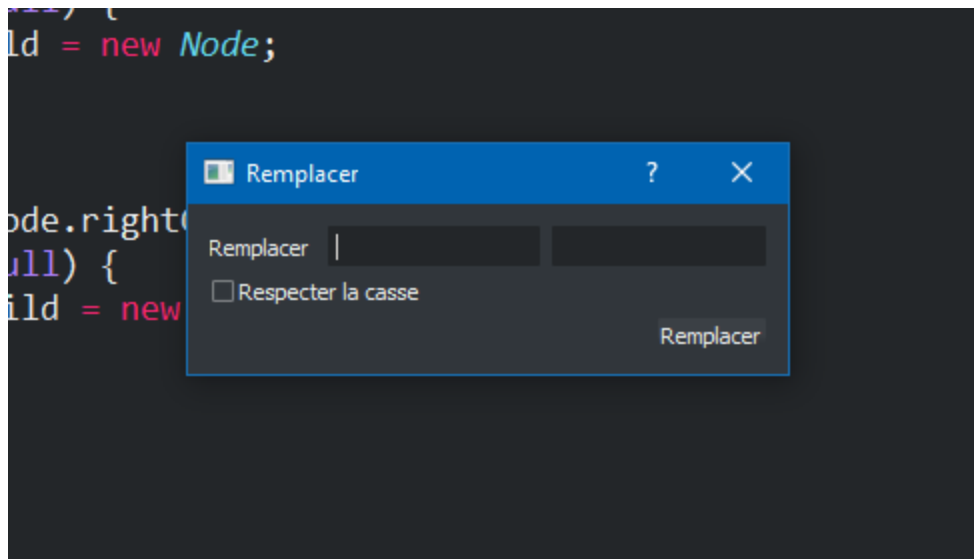
Comme on peut le voir, le mot recherché peut être sensible à la casse ou pas. Additionnellement, il y a aussi la possibilité de rechercher une expression régulière à la place. Si un pattern est trouvé il apparaîtra en surbrillance comme ci-dessus.

Ci-dessous l'on peut voir la même recherche sans respecter la casse.



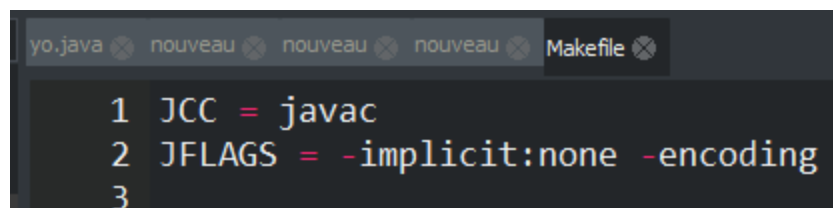
2 - 3 Remplacer un mot

On peut aussi remplacer un mot et ceci en précisant si l'on veut respecter la casse ou pas encore une fois



2 - 4 Onglets

Si plusieurs fichiers sont ouverts, l'éditeur grise les onglets qui ne sont pas actifs.



Grâce à Qt, les onglets sont aussi interchangeables de place.

2 - 5 Double caractère

Pour certains caractères comme les guillemets, les accolades, les parenthèses, les apostrophes ou encore les crochets, l'éditeur mettra deux des caractères tapés puis mettra le curseur entre les caractères pour faciliter l'écriture de fonctions, tableaux ou autres.

```
public class Main {}
```

2 - 6 Surbrillance double caractères

Les accolades, parenthèses et crochets ont tous le droit de voir en surbrillance le caractère avec lequel il sont associé pour rendre plus lisible le code.

```
public void postOrderTraverseTree(Node focusNode) {  
    if (focusNode != null) {  
        postOrderTraverseTree(focusNode.leftChild);  
        postOrderTraverseTree(focusNode.rightChild);  
        System.out.println(focusNode);  
    }  
    this.cc = 20;  
}
```

```
this.cc = 20;  
tab[[20],[30]] = 20;
```

```
public void postOrderTravers
    if (focusNode != null) {
        postOrderTraverseTre
        postOrderTraverseTre
```

2 - 7 Indentation

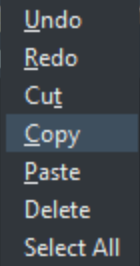
Il y a aussi une indentation automatique comme sur tout éditeur de texte. Cela rend plus simple l'écriture et la lisibilité du code.


```
1 public class Main {
2     |
3 }
4
```

2 - 8 Option/Raccourcis clavier

Sont mis à disposition à l'utilisateur les options suivantes : undo, redo, couper, copier, coller, supprimer et tout sélectionner.

```
19         replace = replace.rightChild;
20         if (replace == null) {
21             replace = new Node();
22             replace.rightChild = new Node();
23         }
24     }
25 }
26 }
27 }
```





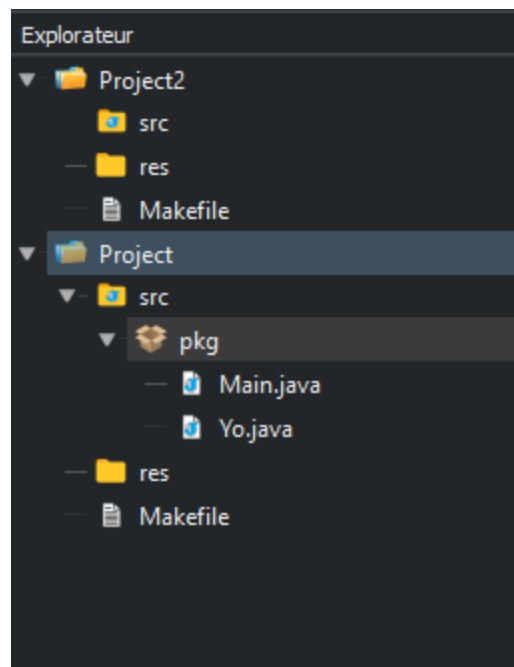
On peut accéder à ces outils en sélectionnant une partie de texte et en faisant un clic droit ou bien en utilisant directement les raccourcis clavier associés c'est-à-dire Ctrl+Z, Ctrl+Y, Ctrl+X, Ctrl+C, Ctrl+V, Del et Ctrl+A respectivement.

3. Présentation de l'explorateur de fichier

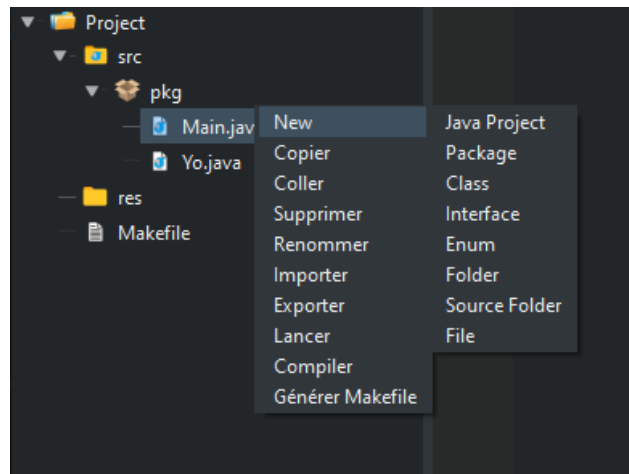
Cette partie présente tous les éléments et les fonctionnalités de l'explorateur de fichier.

3 - 1 Outils

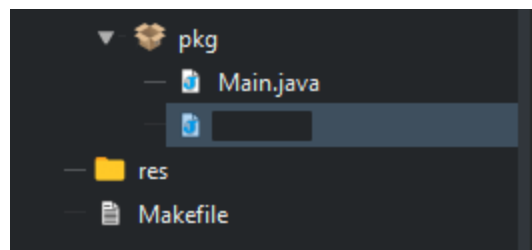
L'explorateur de fichier permet de voir l'arborescence de ses projets ainsi que de manipuler les fichiers qui s'y trouvent.



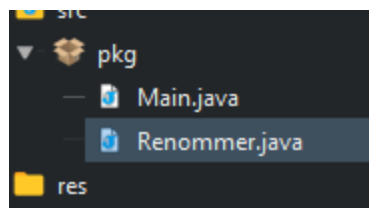
Ceci est particulièrement utile et surtout plus rapide que de passer par la console ou même la création de fichier de l'IDE.



On peut donc depuis cette arborescence faire les actions comme ci-dessus c'est-à-dire créer un fichier, le renommer, le supprimer ou générer un makefile.



(exemple de renommage)



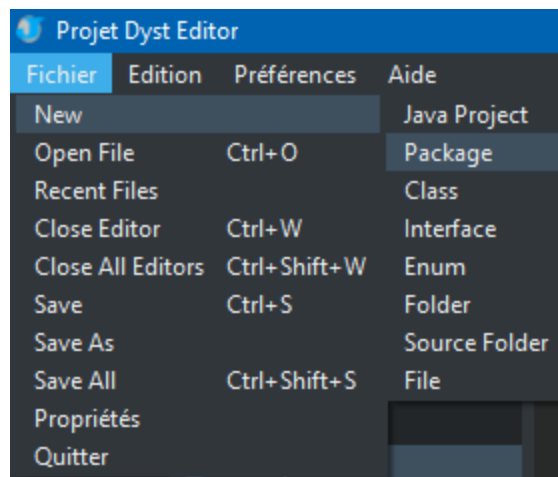
La création comporte un ensemble de type de fichier qui peuvent être utiles pour du développement sous Java. On peut aussi créer un projet ou un dossier depuis ici.

4. Présentation des options de l'IDE

Cette partie présente tous les éléments et les fonctionnalités de l'IDE.

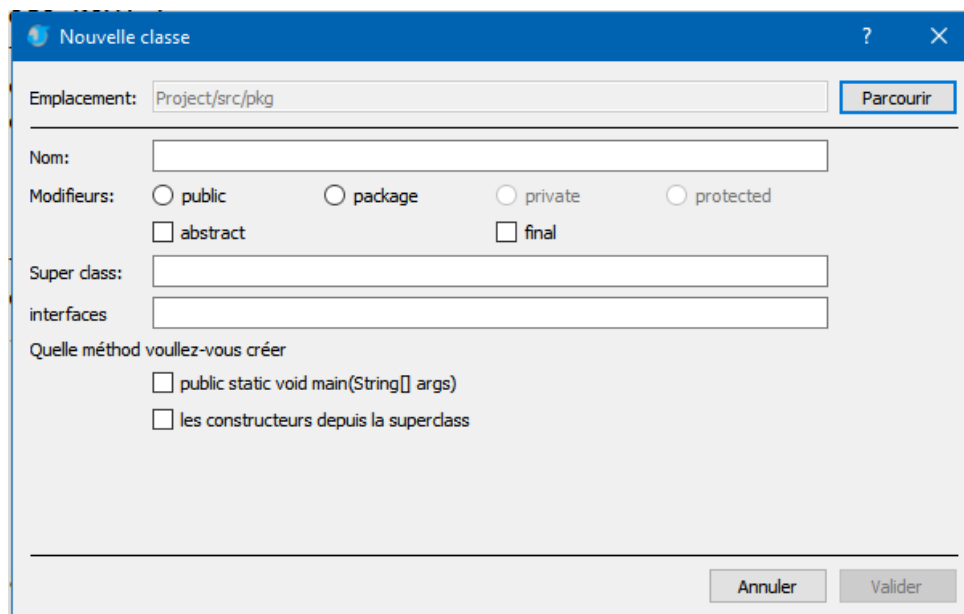
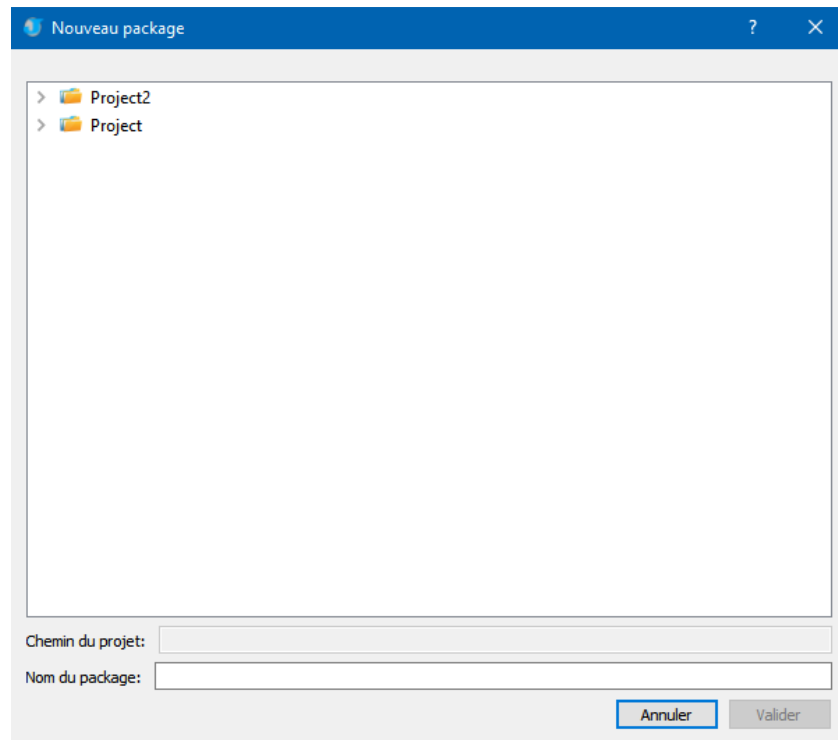
4 - 1 Fichier


On retrouve ici les mêmes options que celles trouvées dans l'explorateur pour la création de fichiers ou autres.



On peut aussi évidemment ouvrir un fichier depuis ce menu, sauvegarder un fichier ou encore quitter l'application.

Ci-dessous se trouve les fenêtres de dialogue de création de ces fichiers.




 Nouvelle énumération ? X

Emplacement:

Nom:

Modifieurs: ☐ public ☐ package ☐ private ☐ protected

interfaces

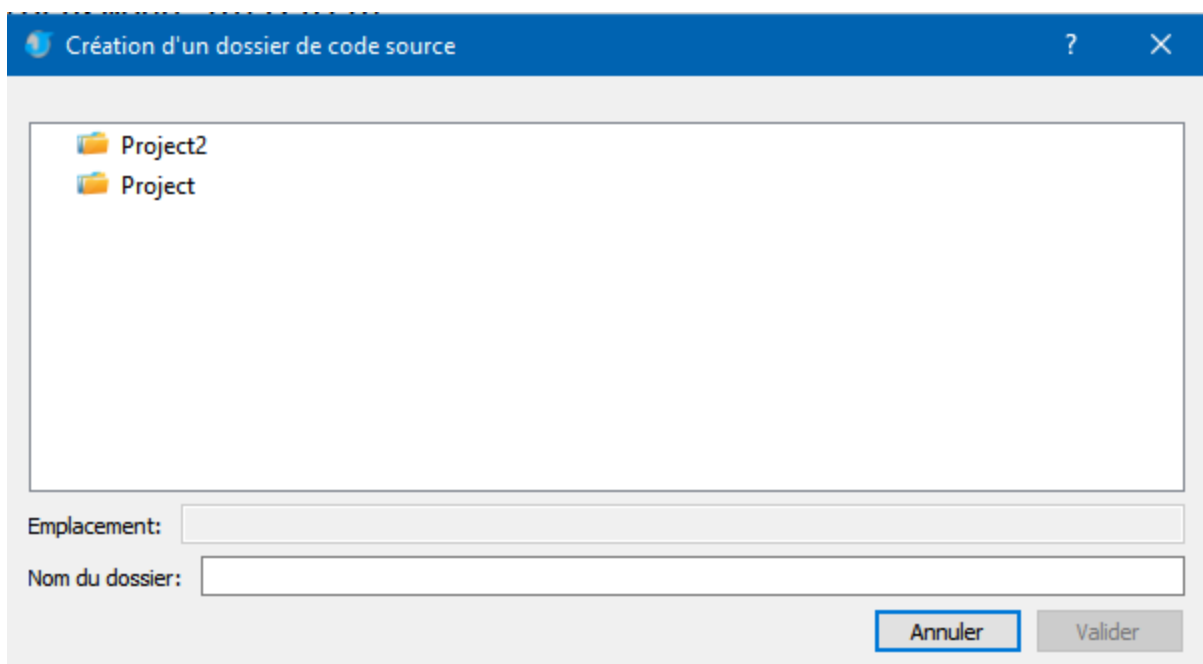
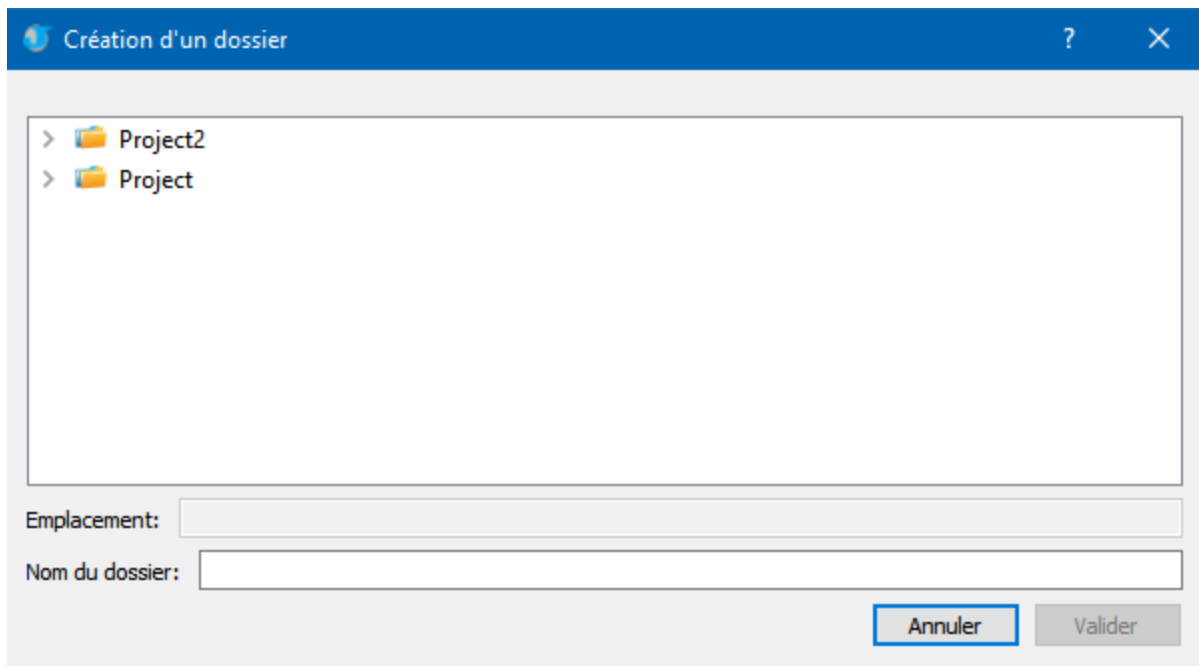
 Nouvelle interface ? X

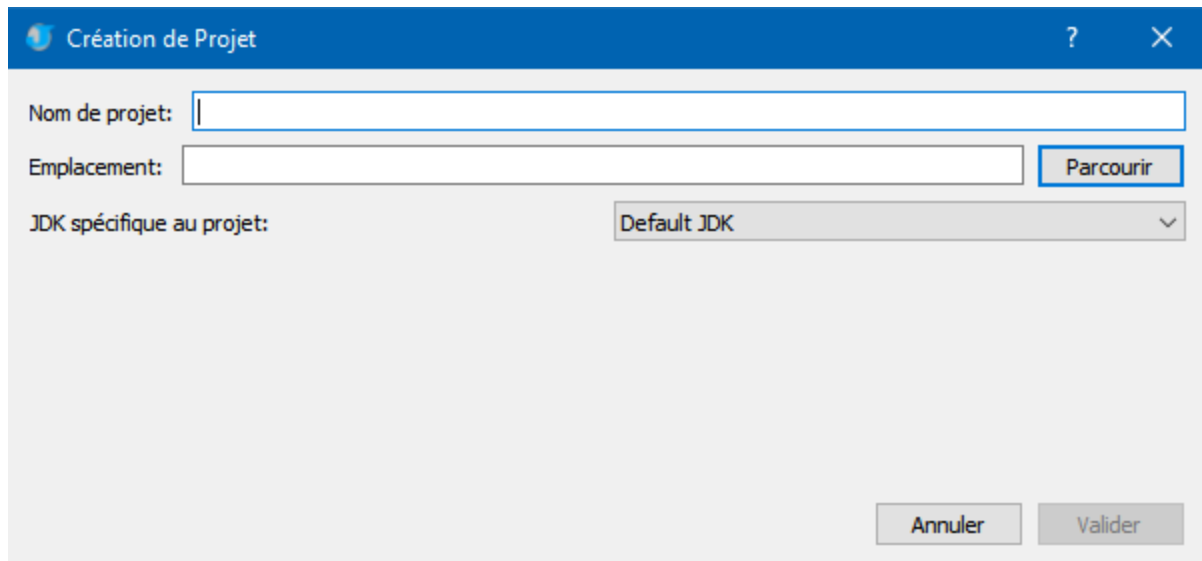
Emplacement:

Nom:

Modifieurs: ☐ public ☐ package ☐ private ☐ protected

interfaces





A screenshot of a 'Création de Projet' (Project Creation) dialog box. The dialog has a blue title bar with the text 'Création de Projet' and standard window controls (help, close). The main area contains three input fields: 'Nom de projet:' (Project Name), 'Emplacement:' (Location), and 'JDK spécifique au projet:' (JDK specific to the project). The 'Emplacement:' field has a 'Parcourir' (Browse) button next to it. The 'JDK spécifique au projet:' field is a dropdown menu currently showing 'Default JDK'. At the bottom right, there are two buttons: 'Annuler' (Cancel) and 'Valider' (Validate).

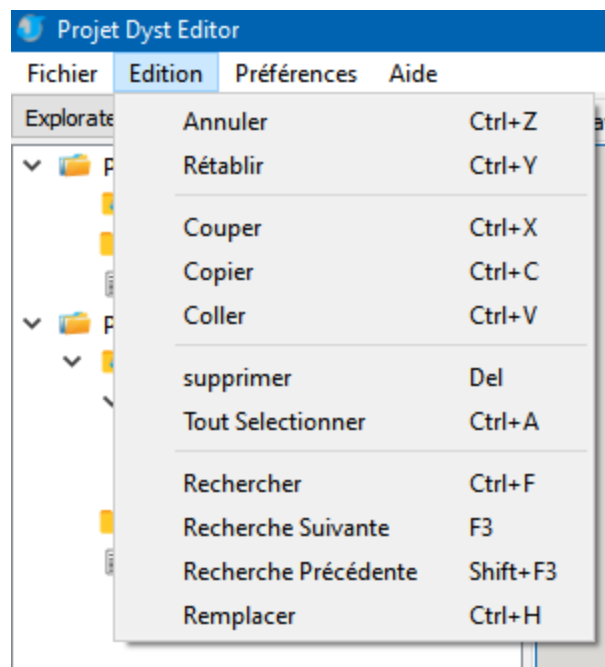
Nom de projet:

Emplacement:

JDK spécifique au projet:

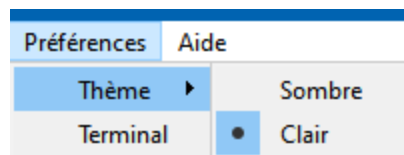
4 - 2 Edition

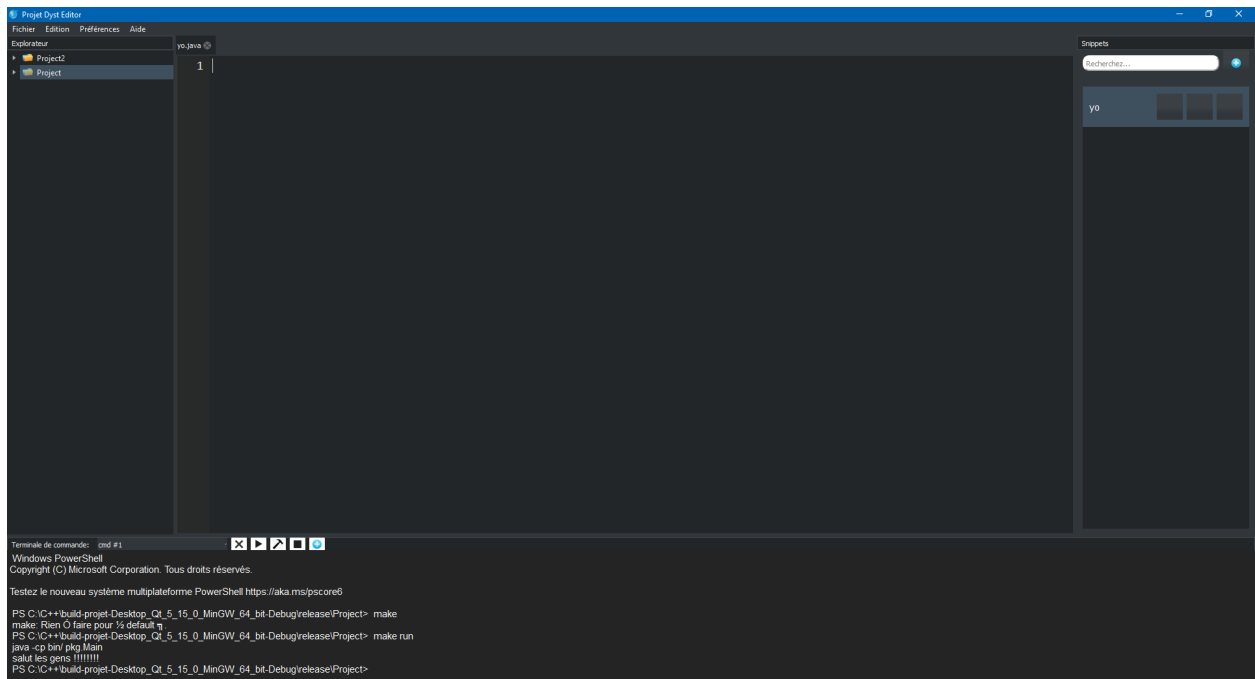
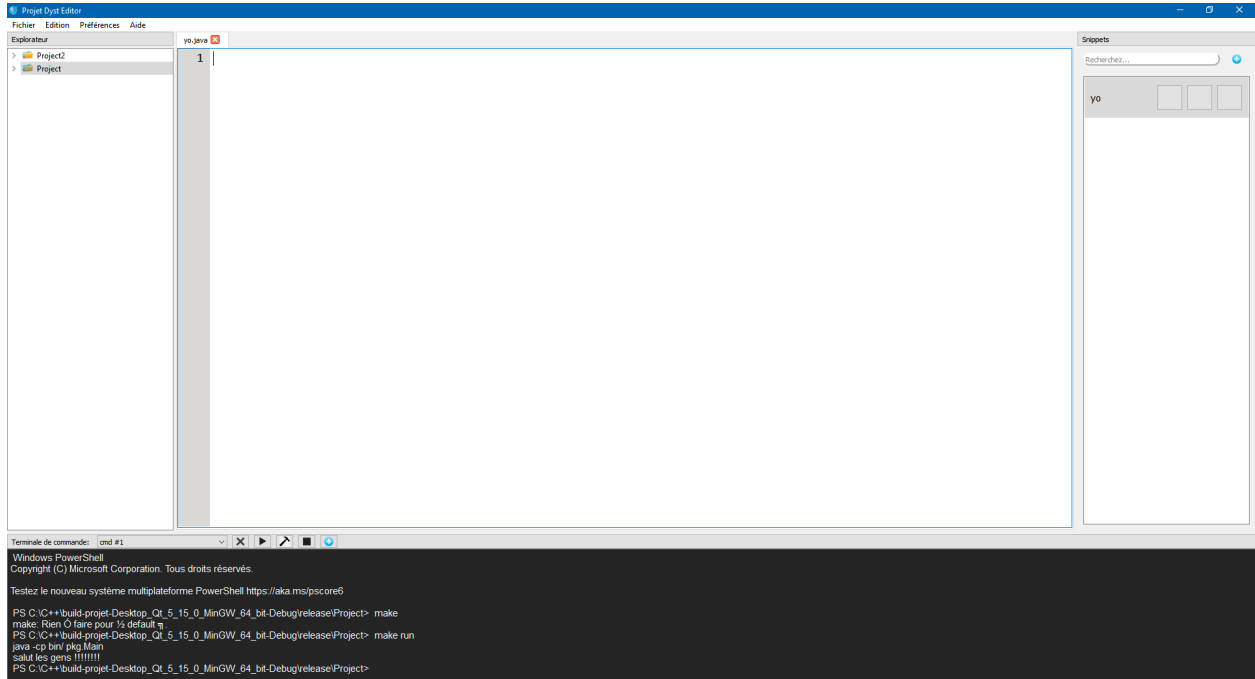
Depuis ce menu on retrouve tous les raccourcis clavier que l'on peut effectuer sur la fenêtre d'édition de texte.



4 - 3 Préférences

Le menu des préférences permet de changer l'apparence de l'IDE, d'un thème sombre à un thème clair comme ci-dessous.

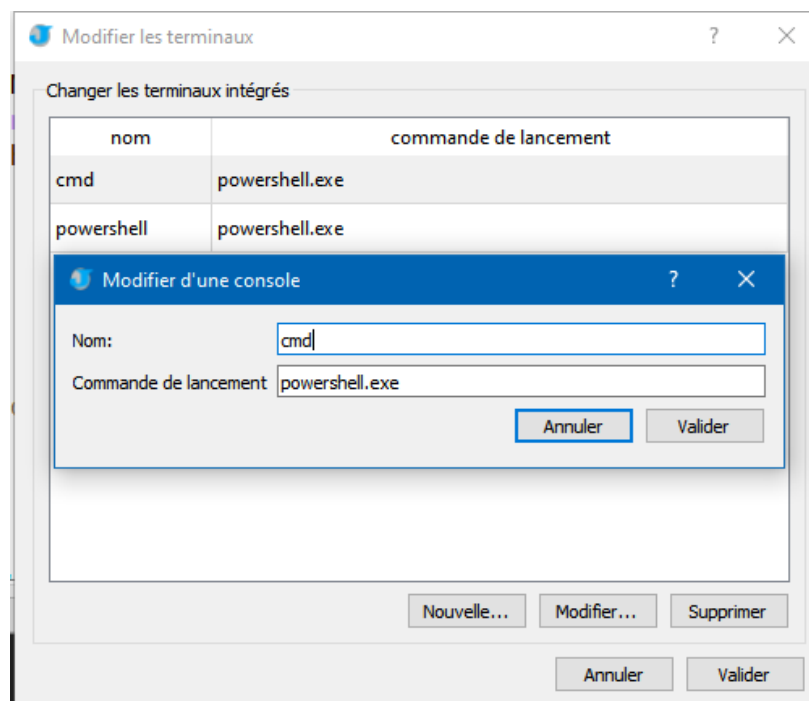
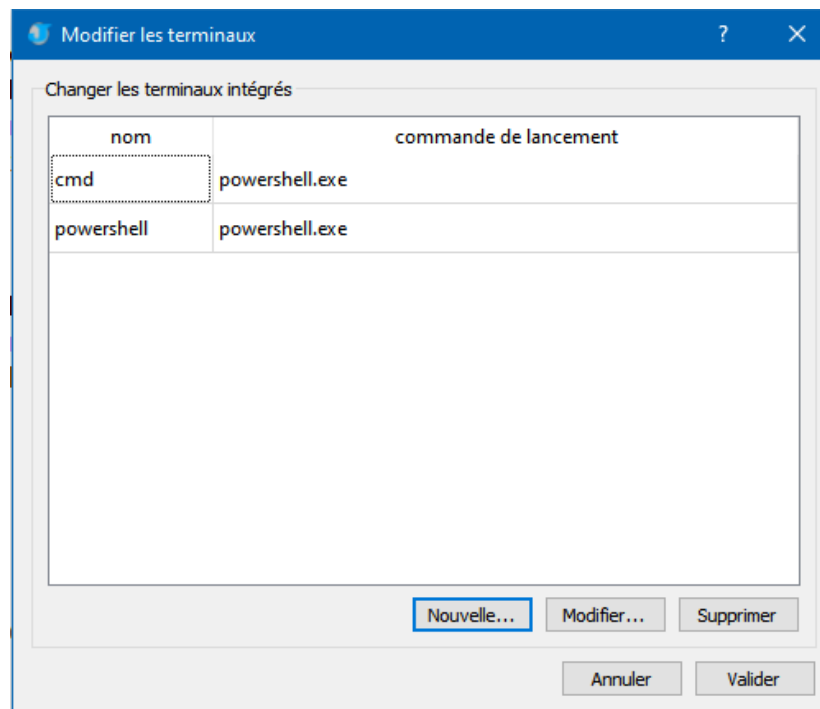




Jérémy VARANGE, Yohan BORDES, Noah LACASTE

L'onglet préférence permet aussi d'apporter des modifications au terminal sélectionné.

D'ici l'utilisateur peut décider quel logiciel sera lancé pour servir de console ainsi que d'un nom pour pouvoir différencier. Il peut aussi modifier les terminaux qu'il a créés.

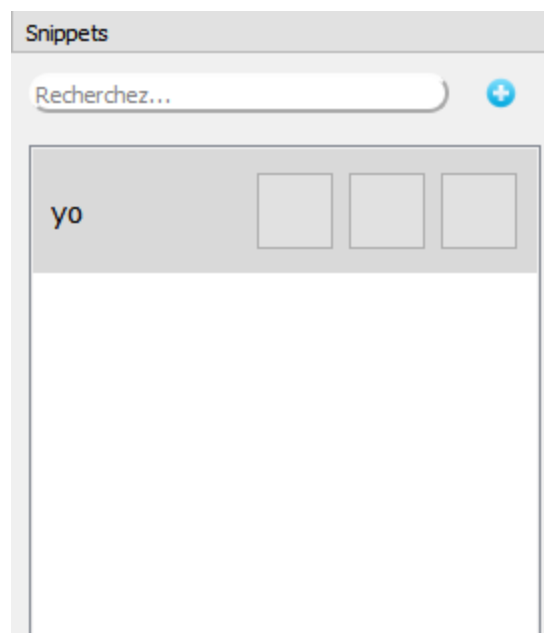


5. Présentation des Snippets

Cette partie présente tous les éléments et les fonctionnalités de la partie snippet.

5 - 1 Barre des snippets

Un snippet est une petite portion réutilisable de code source ou de texte. Ordinairement, ce sont des unités formellement définies à incorporer dans des modules plus larges. Ils sont donc très utiles pour ne pas perdre de temps à recopier le même code, c'est pour cela que c'était un Should.



Depuis ce menu on peut donc chercher un snippet, en ajouter un grâce au bouton bleu, en supprimer un avec le carré le plus à droite, copier un snippet avec le carré du milieu et ouvrir un snippet avec le carré le plus à gauche.



6. Présentation de la Console

Cette partie présente tous les éléments et les fonctionnalités de la partie console.

6 - 1 Barre de la console

La barre de la console comporte 5 boutons :



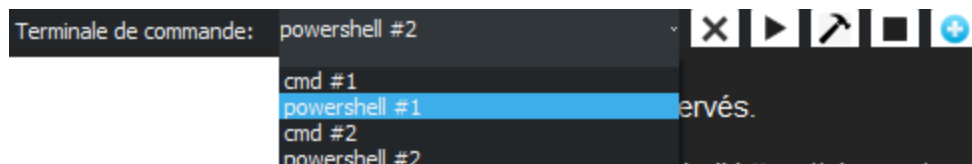
La croix permet de supprimer le terminal sélectionné.

Le bouton play permet de compiler et d'exécuter le makefile (make run).

Le bouton marteau ne fait que la compilation (make).

Le bouton avec un carré devrait faire s'arrêter la console.

Enfin le bouton plus permet d'ajouter d'autres consoles comme ci-dessous par exemple.



6 - 2 Utilisation de la console

Puisque la console est directement intégrée à l'application, toutes les commandes classiques marchent.

```
Terminale de commande: cmd #1
PS C:\C++\build-projet-Desktop_Qt_5_15_0_MinGW_64_bit-Debug\release\Project> make run
java -cp bin/ pkg.Main
salut les gens !!!!!!!
PS C:\C++\build-projet-Desktop_Qt_5_15_0_MinGW_64_bit-Debug\release\Project> ls

Répertoire : C:\C++\build-projet-Desktop_Qt_5_15_0_MinGW_64_bit-Debug\release\Project

Mode                LastWriteTime         Length Name
----                -
d-----          19/03/2021   22:06             bin
d-----          19/03/2021   18:45             res
d-----          19/03/2021   18:46             src
-a----          20/03/2021   16:02         516 .javora.jpml
-a----          20/03/2021   17:34         424 Makefile
-a----          19/03/2021   23:37        1253 Project.jar

PS C:\C++\build-projet-Desktop_Qt_5_15_0_MinGW_64_bit-Debug\release\Project> |
```