AIDE ROBOCODE

Table des matières

Bienvenue à Robocode!

Le jeu Robocode a été développé initialement, par Mathew A. Nelson (2001-2005), il a été repris, en 2006 par Flemming N. Larsen.

Robocode est un jeu qui permet d'apprendre à programmer en java en faisant affronter des robots tanks virtuels.

C'est une application Java qui permet :
de programmer ses robots
de les tester
de lancer des matchs et des tournois

Voici le message d'accueil de Mathew A. Nelson :

« J'ai travaillé dur pour mettre ce jeu à votre disposition, et j'espère que vous prendrez autant de plaisir à y jouer que j'en ai pris à le développer. Depuis que je suis enfant, j'ai été très attiré par les jeux de ce type, et j'adore la programmation en Java, j'ai donc décidé de combiner les deux.

Les batailles de Robocode ont lieu dans une arène, dans laquelle des robots à 6 roues combattent jusqu'à ce qu'un seul d'entre eux reste « en vie ». Je souhaite préciser que Robocode ne contient pas de sang, de personnalités publiques, ni de politiques. Les combats ont pour seul but de fournir l'excitation de la compétition que nous aimons tant. Il y a cependant des explosions. ☺

Vous verrez qu'il est assez évident de démarrer dans ce jeu, et ces pages vont vous y aider.

Avant tout, prenez votre pied! Vous pouvez créer un robot en un jour, mais cela pourrait vous prendre des mois. Je ne dirai pas qu'il est facile d'apprendre à programmer, mais vous allez prendre du bon temps en le faisant.

Merci ! » –Mat

Suite : Installation de Robocode_Démarrer

Retour : Table des matières_Démarrer

Installation du Kit de Développement Java

Robocode est un programme Java. Il fonctionne sous Windows, Linux, Mac, etc.
Avant de commencer à installer et utiliser Robocode, vous devez disposer d'un « Kit de Développement Java ». Celui-ci est disponible gratuitement sur le site de la société SUN.

Pour télécharger le « kit de développement Java 5.0 », cliquez ici.

Une fois le JDK 5.0 installé sur votre machine, vous pourrez poursuivre en installant Robocode.

Suite : Installation de Robocode
Retour : Table des matières

Installation de Robocode

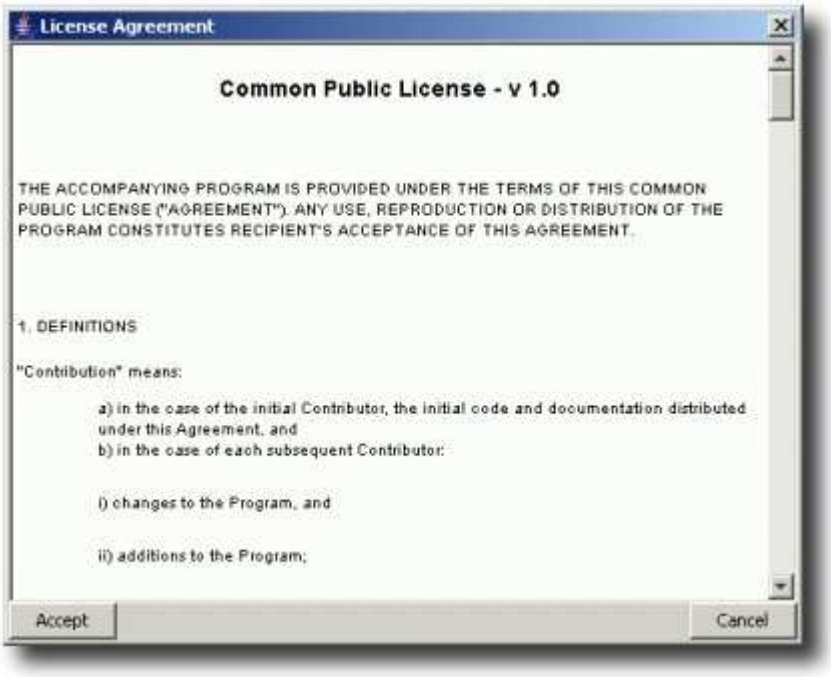
Avant de démarrer l'installation de Robocode, vous devez disposer d'une version du kit de développement Java 5.0. Si ce n'est pas le cas, référez vous à : Installation du Kit de Développement Java.

Ensuite, téléchargez Robocode en cliquant ici.

Le fichier que vous avez téléchargé s'appelle robocode-setup.jar.

Double cliquez sur ce fichier et l'installation commencera automatiquement.

Vous verrez d'abord apparaître une fenêtre qui vous demandera d'accepter la licence (gratuite) du logiciel:



Cliquez sur "Accept"

Ensuite il vous proposera d'installer le logiciel dans le répertoire c:\robocode (pour Windows):

Acceptez tout, et l'installation est terminée!

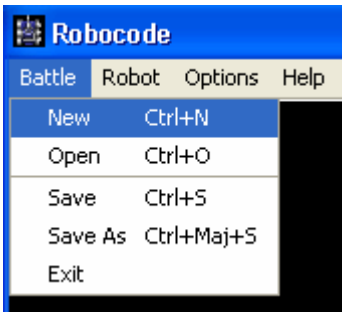
Suite : Démarrer

Retour : Table des matières

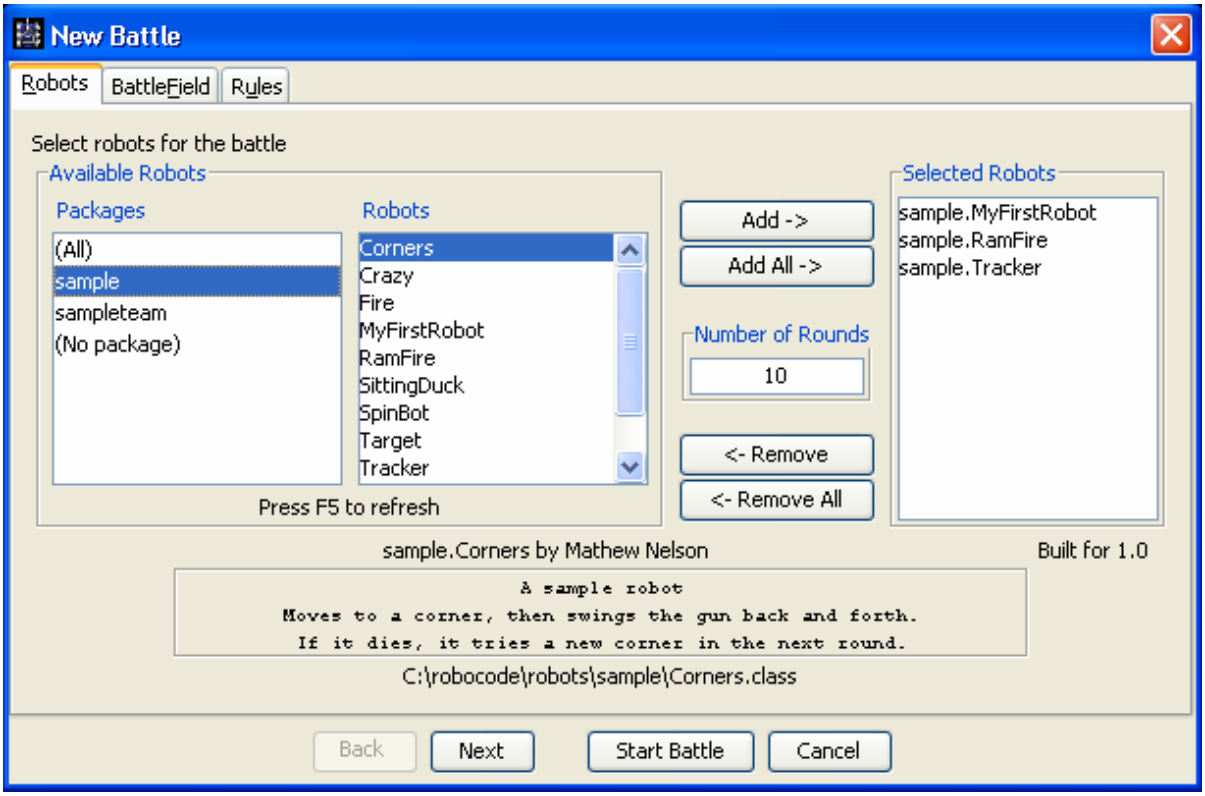
Démarrer

Pour lancer Robocode, double-cliquez sur le raccourci. Si vous êtes sous Windows, une fenêtre DOS s'ouvrira avant le lancement du programme.

Tout d'abord, lançons une bataille afin de voir à quoi le jeu ressemble. Cliquez simplement le bouton "**Battle**" du menu principal, ensuite, sélectionnez "**New**" :



L'écran "**New Battle**" s'affiche, dans lequel vous pouvez sélectionner les robots et les options pour une bataille. Pour cette bataille, nous essaierons MyFirstRobot, RamFire, et Tracker. Ajoutez-les en double cliquant sur leurs noms (ou en sélectionnant chacun d'eux et en cliquant « Add »). L'écran devrait maintenant ressembler à ceci :



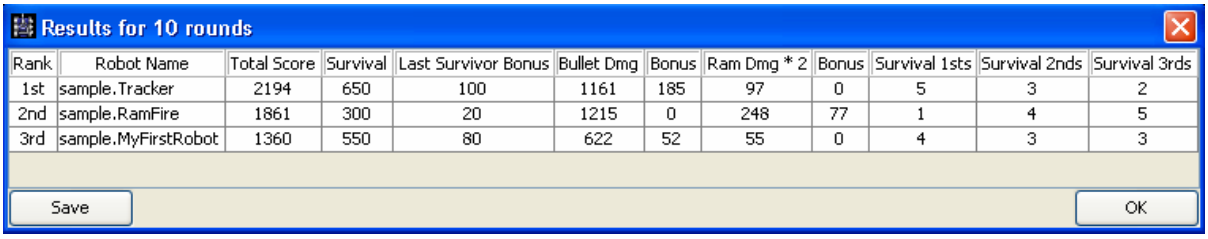
Observez le "**Number of Rounds**", il indique le nombre de combats qui auront lieu durant cette bataille. Pour l'instant, nous le laisserons à 10.
Vous pouvez lancer le combat en enfonçant le bouton « Start Battle ».

Suite : Scores

Retour : Table des matières

Scores

Quand une bataille se termine, vous verrez apparaître une fenêtre présentant un ensemble de résultats ressemblant à ceci :



Rank	Robot Name	Total Score	Survival	Last Survivor Bonus	Bullet Dmg	Bonus	Ram Dmg * 2	Bonus	Survival 1sts	Survival 2nds	Survival 3rds
1st	sample.Tracker	2194	650	100	1161	185	97	0	5	3	2
2nd	sample.RamFire	1861	300	20	1215	0	248	77	1	4	5
3rd	sample.MyFirstRobot	1360	550	80	622	52	55	0	4	3	3

Voici le détail des résultats :

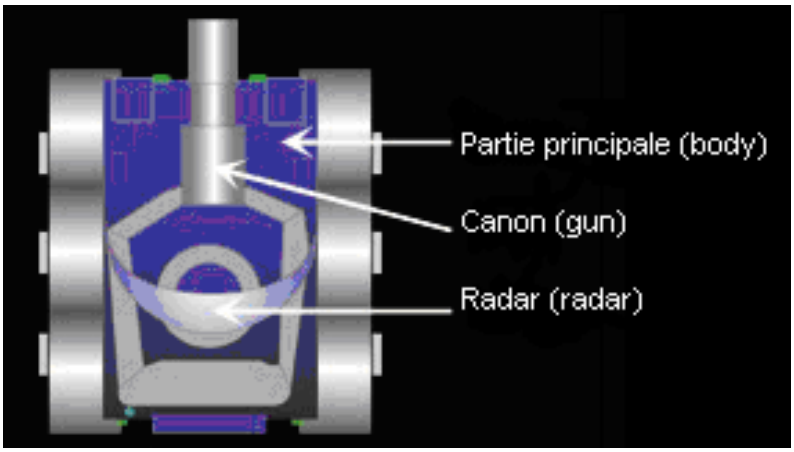
- Total Score – C'est le total général, la somme de tous les autres. Il détermine le score de chaque robot dans cette bataille ;
- Survival Score – Chaque robot en vie reçoit 50 points lorsqu'un autre robot est détruit ;
- Last Survivor Bonus – Le dernier robot en vie reçoit 10 points de bonus pour chaque robot détruit avant lui ;
- Bullet Damage – Un robot reçoit 1 point pour chaque point de dommage causé à l'ennemi;
- Bullet Damage Bonus – Quand un robot détruit un ennemi, il reçoit un bonus de 20% de tous les dommages qu'il a causés à cet ennemi ;
- Ram Damage - Un robot reçoit 1 point pour chaque point de dommage causé à l'ennemi en le tamponnant ;
- Ram Damage Bonus - Quand un robot détruit un ennemi en le tamponnant, il reçoit un bonus de 30% de tous les dommages qu'il a causés à cet ennemi ;
- Survival 1sts, 2nds, 3rds, ne contribuent pas au score total, mais donne une indication de la durée de survie d'un robot.

Suite : Anatomie d'un robot

Retour : Table des matières

Anatomie d’un robot

Chaque robot est constitué de trois parties qui peuvent chacune bouger indépendamment :



- La partie principale du véhicule peut :
  - avancer (ahead)
  - reculer (back)
  - tourner à gauche (turnLeft)
  - tourner à droite (turnRight)
- Le canon (gun) est installé sur la partie principale du véhicule, il peut :
  - tourner à gauche (turnGunLeft)
  - tourner à droite (turnGunRight)
  - tirer des obus (fire)
- Le radar est installé sur le canon. Il peut :
  - tourner à gauche (turnRadarLeft)
  - tourner à droite (turnRadarRight)

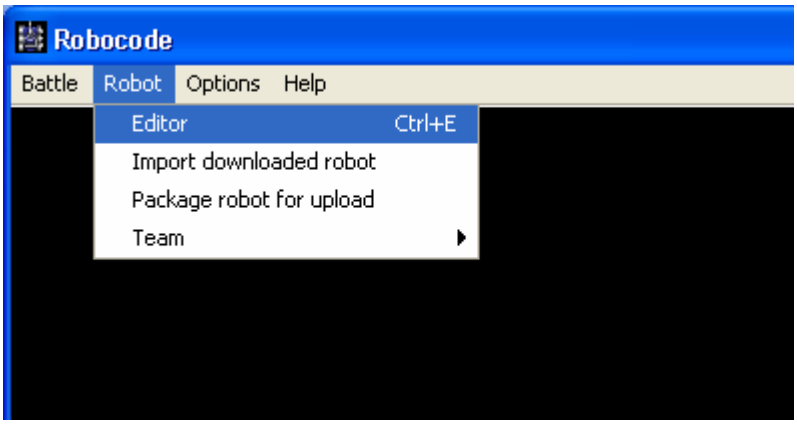
Suite : Création d’un Robot

Retour : Table des matières

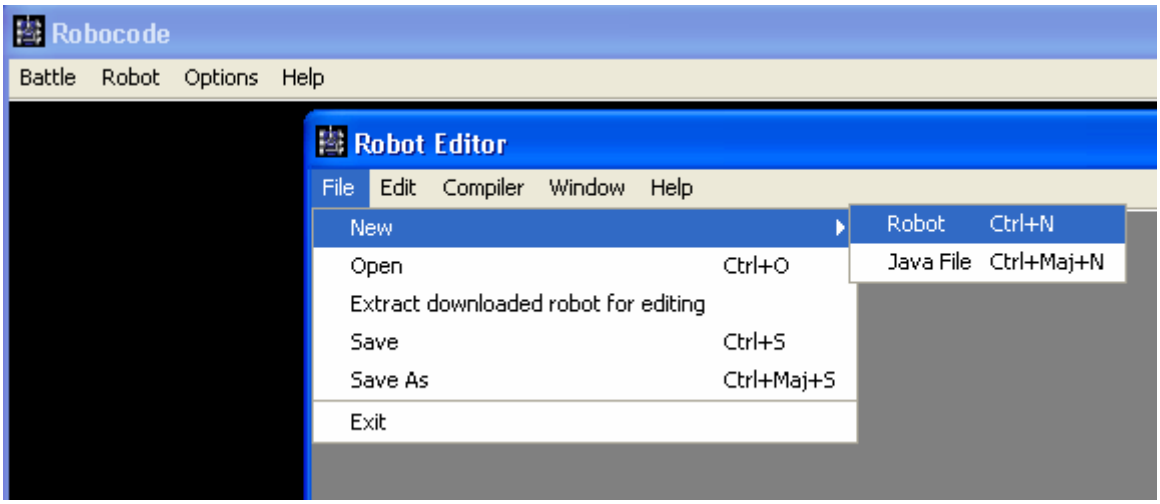
Création d'un Robot

Créer un robot est facile, mais le rendre invincible ne l'est pas.
Sa création peut vous prendre quelques minutes, ou alors des mois et des mois.
La mise au point d'un robot peut devenir une drogue! Les améliorations apportées à votre robot ne se feront pas sans peine, vous ferez des erreurs et il y aura plus d'un tir manqué. Mais au fur et à mesure de l'expérience acquise, vous apprendrez à votre robot où aller, comment réagir, qui éviter, et dans quelle direction tirer. Devra-t-il se cacher dans un coin ou se lancer dans la bagarre?

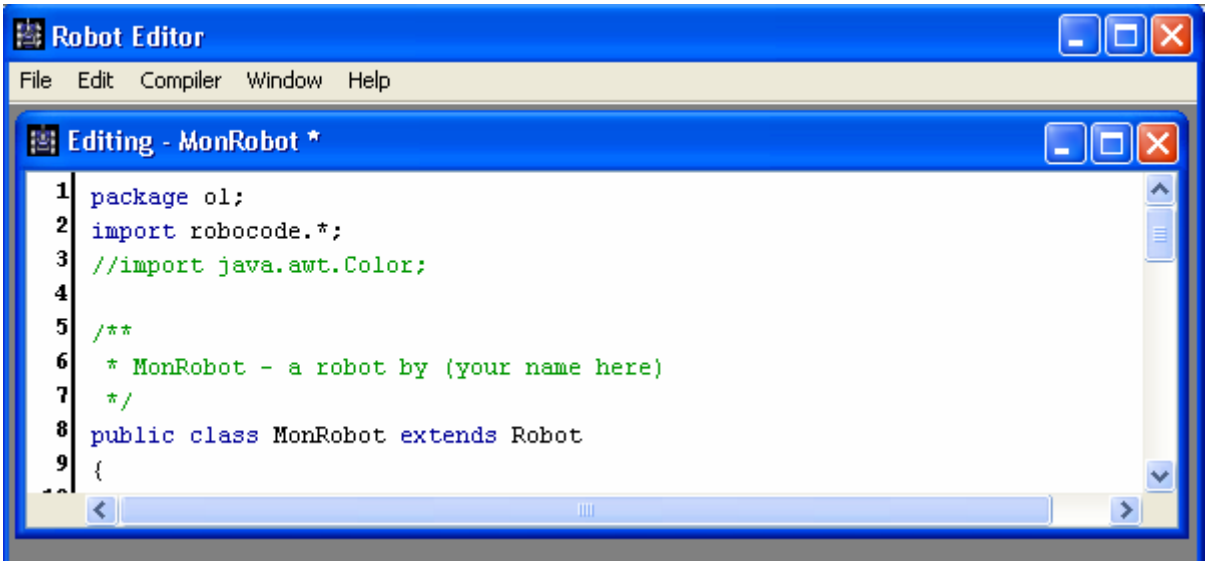
Pour créer un robot, lancez le programme et sélectionnez « Robot\Editor » :



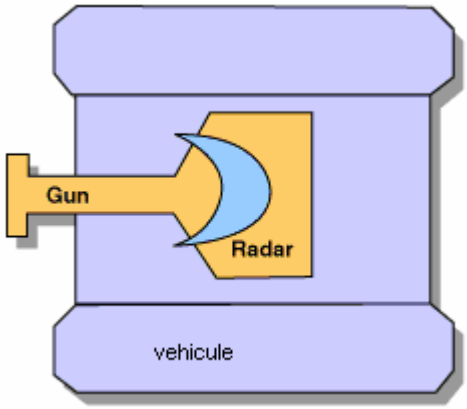
Ensuite, sélectionnez « File\New\Robot » :



Encodez le nom de votre robot (dans l'exemple : « MonRobot », placé dans le répertoire « \ol »).



Le robot est composé de trois éléments : la partie principale (vehicule ou body), le canon (gun) et le radar. Chaque élément peut tourner sur 360° et indépendamment des autres parties.



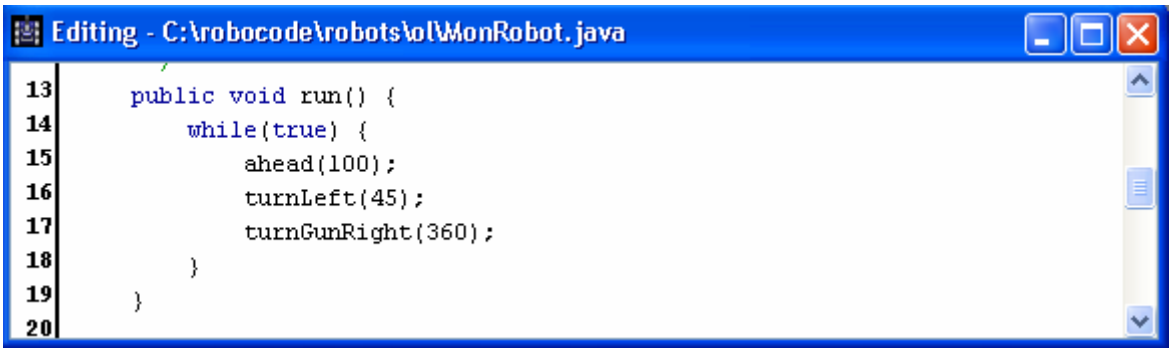
- La partie principale du véhicule est contrôlée par les instructions suivantes:
  - avancer : *void* ***ahead****(double distance)*
  - reculer : *void* ***back****(double distance)*
  - tourner à gauche d'un certains nombre de degrés : *void* ***turnLeft****(double degrees)*
  - tourner à droite d'un certains nombre de degrés : *void* ***turnRight****(double degrees)*

- Le canon (gun) est installé sur la partie principale du véhicule, il peut :
  - tourner à gauche : *void* ***turnGunLeft****(double degrees)*
  - tourner à droite : *void* ***turnGunRight****(double degrees)*
  - tirer des obus : *void* ***fire****(double puissance)*. Plus la puissance (1, 2 ou 3) sera importante, plus les dommages occasionnés le seront. L'énergie dépensée par le robot qui tire est proportionnelle à la puissance de celui-ci.

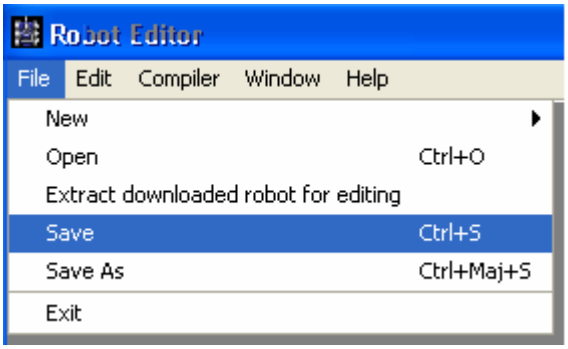
- Le radar est installé sur le canon. Il peut :
  - tourner à gauche : *void* ***turnRadarLeft****(double degrees)*
  - tourner à droite : *void* ***turnRadarRight****(double degrees)*

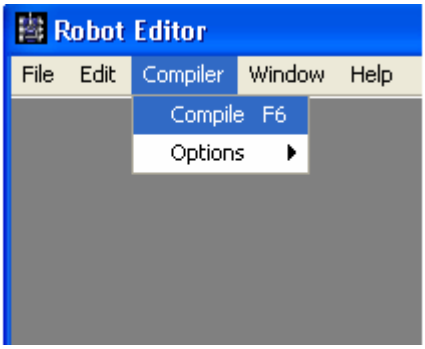
Pour déterminer le comportement du robot, ces ordres (appels de méthodes) peuvent être entrés dans la méthode « run() » qui sera appelée à intervalles réguliers lors d'un combat. Dans l'exemple ci-dessous, à chaque cycle, le robot :

- avancera de 100 pixels ;
- ensuite, tournera à gauche de 45° ;
- et fera tourner son canon sur 360°.

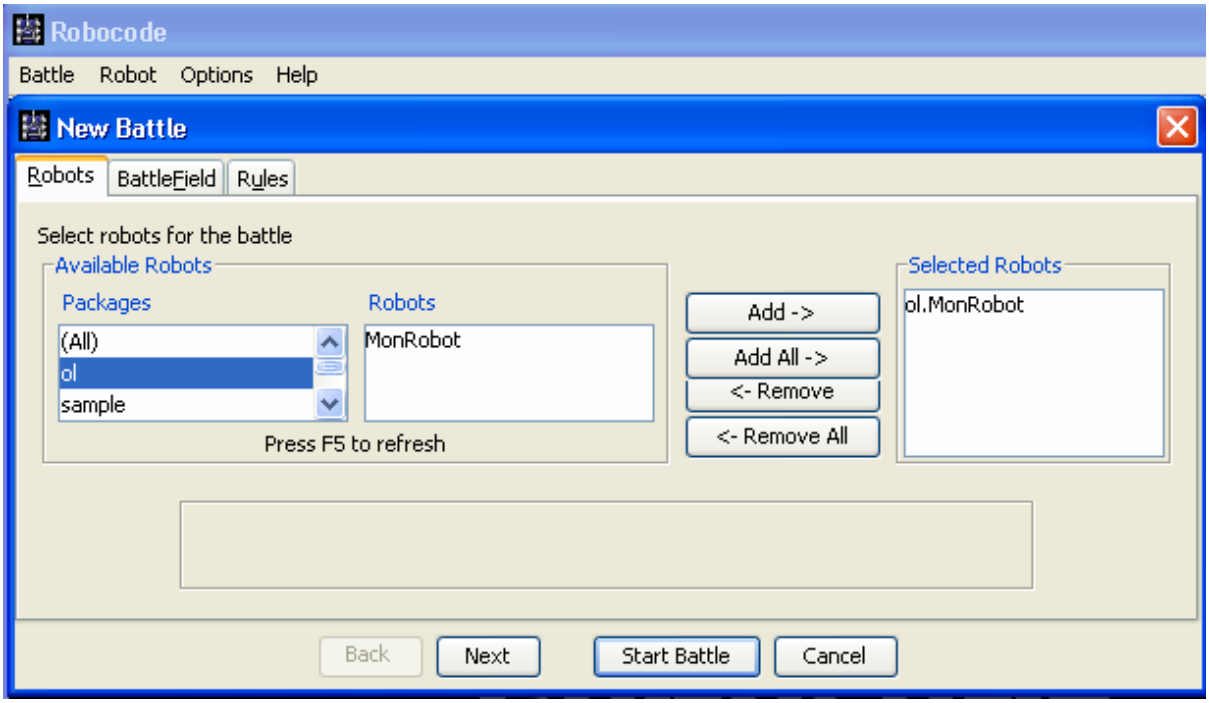


Avant de tester le robot ainsi créé, sauvez et compilez le fichier (compiler un robot le transforme en code exécutable):





Pour tester votre robot, créez une nouvelle bataille :

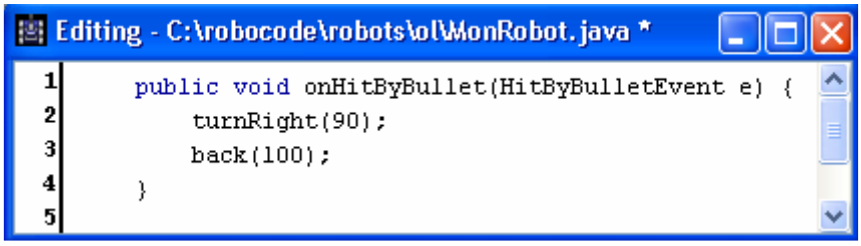


Suite : Améliorations d'un Robot
Retour : Table des matières

Améliorations d'un Robot

Vous pouvez améliorer le comportement de votre robot en le faisant réagir à des événements. Par exemple, lorsqu'il est touché par un obus, vous pouvez lui ordonner de se déplacer afin de ne pas rester dans le champ de tir.

L'exemple suivant ordonne au robot de tourner vers la droite et de reculer lorsqu'il constate qu'il a été touché par un obus :



Voici les méthodes qui peuvent être complétées afin d'améliorer le comportement du robot lorsqu'un événement survient :

onBulletHit : méthode appelée quand le robot a tiré et touché un autre robot ;
onBulletHitBullet : méthode appelée quand un de ses obus a rencontré un autre obus ;
onBulletMissed : méthode appelée quand un de ses obus s'écrase sur un mur, aucun robot touché;
onDeath : méthode appelée quand le robot « agonise » ;
onHitByBullet : méthode appelée quand le robot est touché par un obus;
onHitRobot : méthode appelée quand il heurte un autre robot;
onHitWall : méthode appelée quand il heurte un mur ;
onRobotDeath : méthode appelée quand un autre robot est détruit;
onScannedRobot : méthode appelée quand le radar du robot détecte un autre robot;
onWin : méthode appelée quand le robot a gagné le combat.

Pour plus d'informations sur ces méthodes et les événements qui y sont associés, consultez la documentation de la classe Robot (voir Documentation « javadoc »).

Remarque : lorsque le robot tourne, le canon tourne également dans la même direction avec la même amplitude (comme dans la réalité). Il est cependant possible, de configurer le robot pour que le canon continue à pointer dans une certaine direction malgré la rotation du robot. Ceci peut être fait, à en appelant la méthode suivante :

```
setAdjustGunForRobotTurn( true )
```

Le radar étant monté sur le canon, il peut aussi être configuré, soit par rapport aux rotations du robot, soit par rapport aux rotations du canon.

```
setAdjustRadarForRobotTurn( boolean flag )
```

setAdjustRadarForGunTurn(boolean flag)

Suite: Télécharger d’autres Robots

Retour : Table des matières

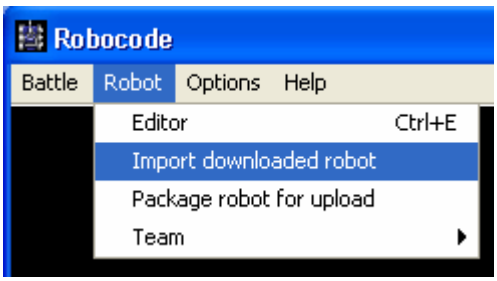
Télécharger d’autres robots

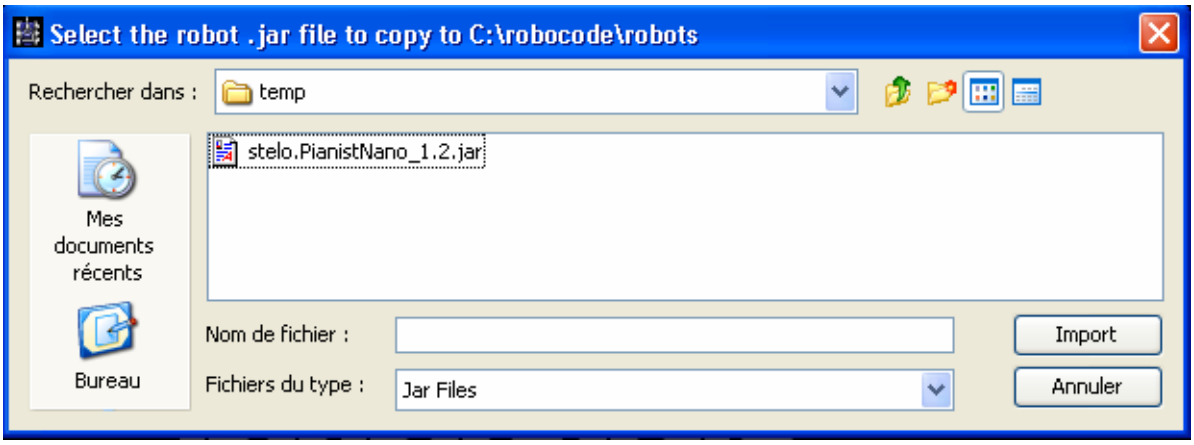
Vous n’êtes pas le seul à écrire des robots. Il existe des centaines de robots écrits par d’autres personnes que vous pouvez télécharger et tester dans une bataille. Attention, certains sont extrêmement difficiles à battre. Il y a une grande variété de robots disponibles fonctionnant suivant différents types de stratégies. Vous pouvez les télécharger, les essayer et apprendre en les observant afin d’améliorer vos propres robots.

Ces robots sont disponibles sur le site suivant : [Robocode Repository]

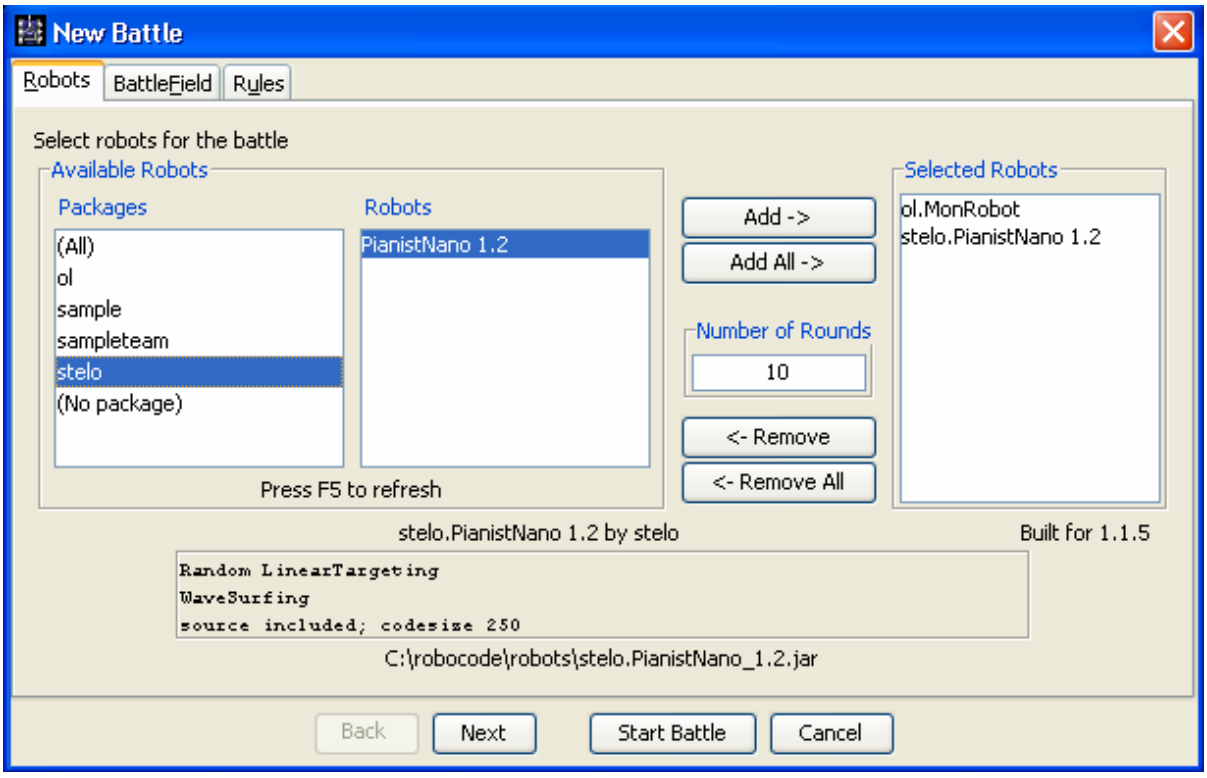
Pour les utiliser, il suffit de télécharger leur fichier « .jar » (tous les robots sont inclus dans un fichier archive ayant une extension « .jar » qui signifie "Java ARchive"), et de les sauver dans votre répertoire **\robots**.

Remarque : La façon la plus facile d’utiliser les robots téléchargés est de les sauver directement dans le répertoire **\robots**. Cependant, vous pouvez les sauvez à un autre endroit et par la suite utiliser "Import downloaded robot" pour les y copier.





Une fois que le nouveau fichier « .jar » du robot est placé dans votre répertoire **\robots**, enfoncez la touche **F5** afin de rafraîchir l'écran « New Battle ». Le robot apparaîtra dans la liste :



Essayez! Votre robot a-t-il pu vaincre ce nouveau robot ?

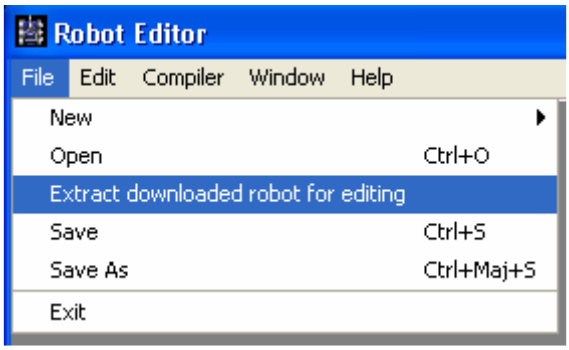
Suite : Apprendre des autres Robots

Retour : Table des matières

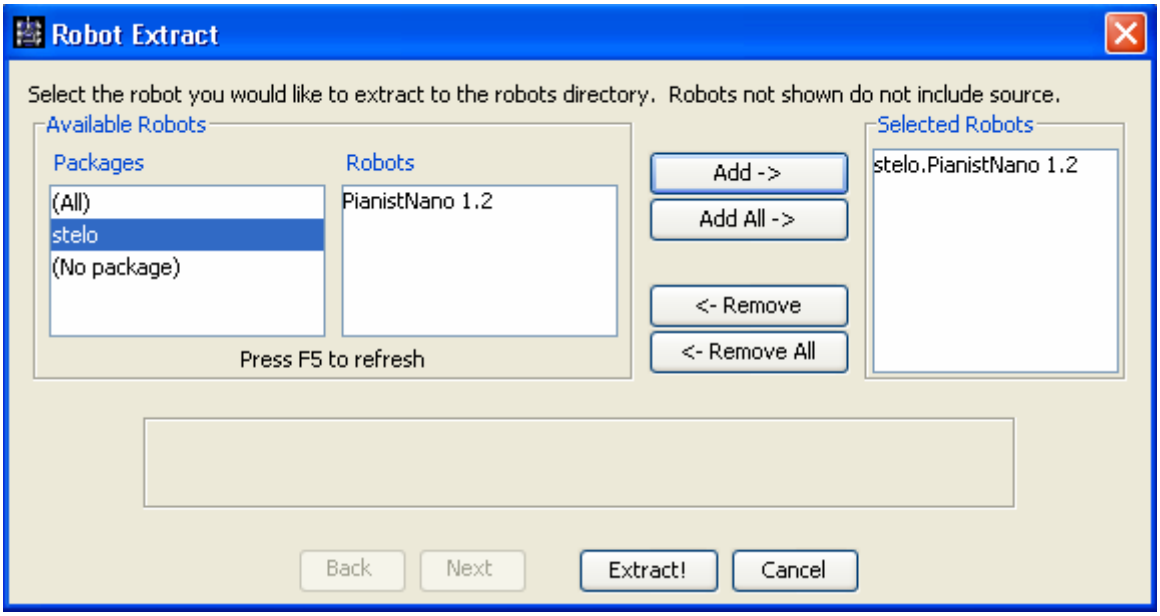
Apprendre des autres Robots

A présent vous devriez savoir comment utiliser l'éditeur pour créer des robots. Si ce n'est pas le cas, commencez par : Création d'un Robot.

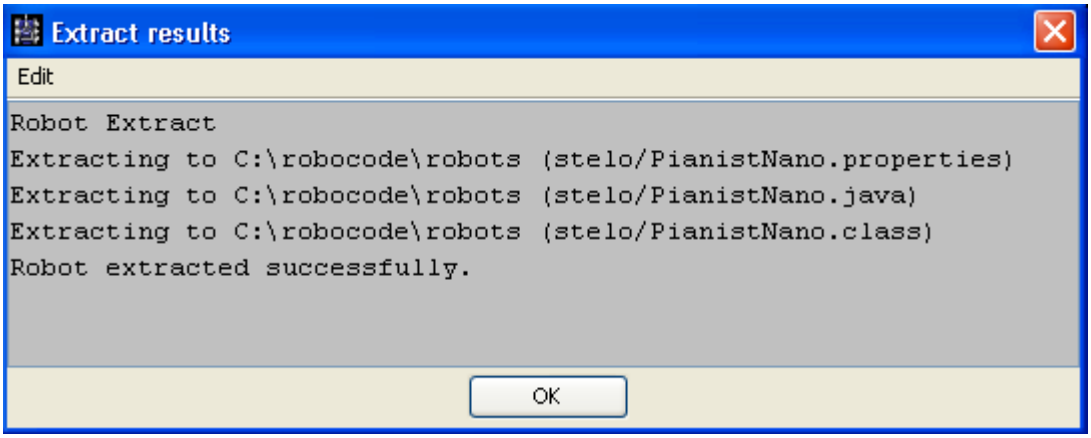
Vous pouvez apprendre en examinant le code de robots écrits par d'autres. Certains auteurs incluent le code source de leur robot dans le fichier « .jar », vous pouvez le consulter en cliquant sur "File\Extract downloaded robot for editing" de l'éditeur de robots.



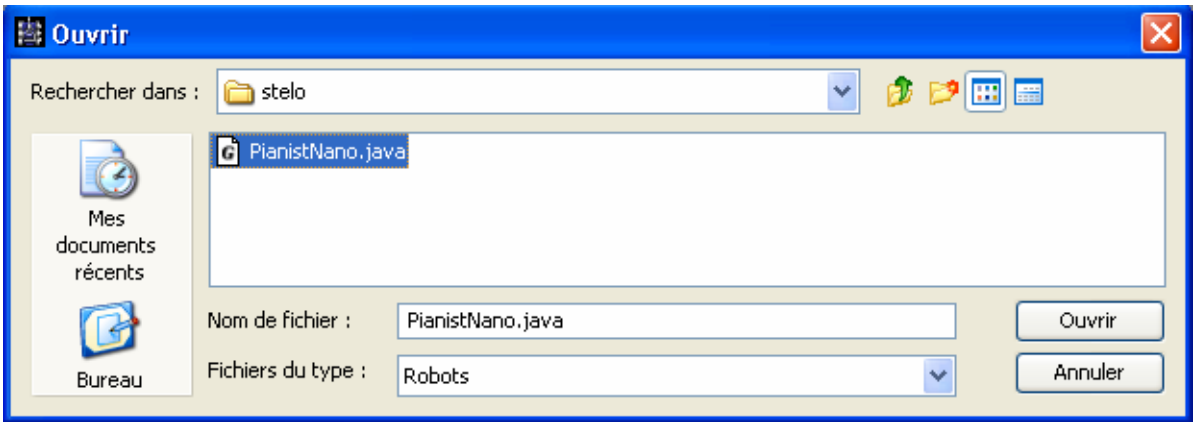
Sélectionnez simplement le robot que vous voulez extraire…



Ensuite, cliquez sur « Extract ! » pour extraire le fichier « .jar » dans le répertoire de robots :



Comme vous le voyez dans cet exemple, les fichiers ont été extraits dans C:\robocode\robots\stelo. Vous pouvez, à présent, les ouvrir et les consulter à l'aide de l'éditeur de robots.



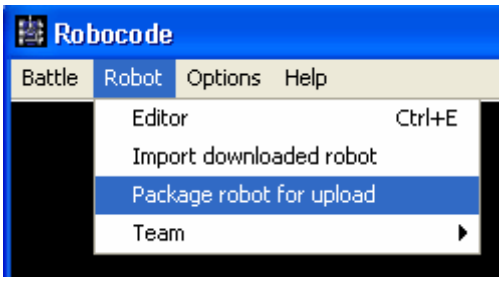
Suite : Télépartager un robot (upload)

Retour : Table des matières

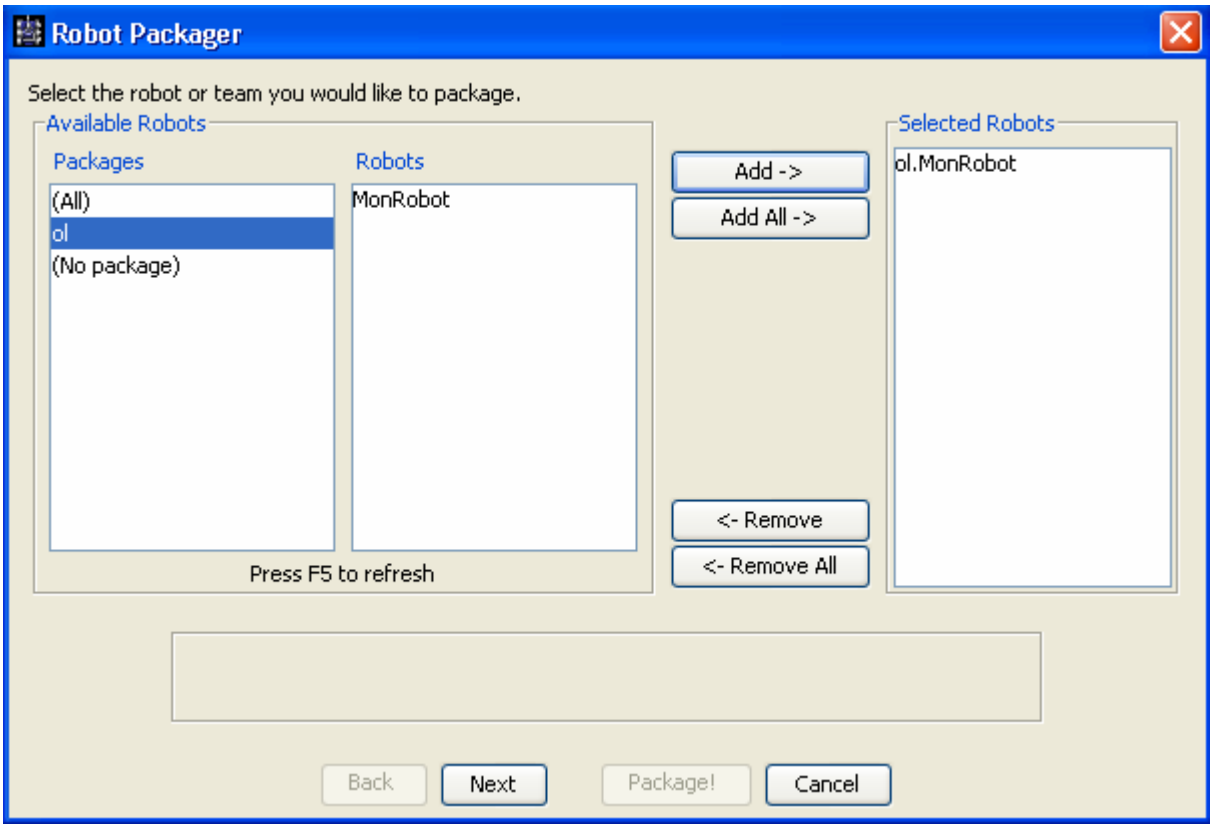
Télépartager un robot (upload)

Une fois que vous avez créé un robot, vous pouvez le partager avec d'autres personnes via internet. Voici comment :

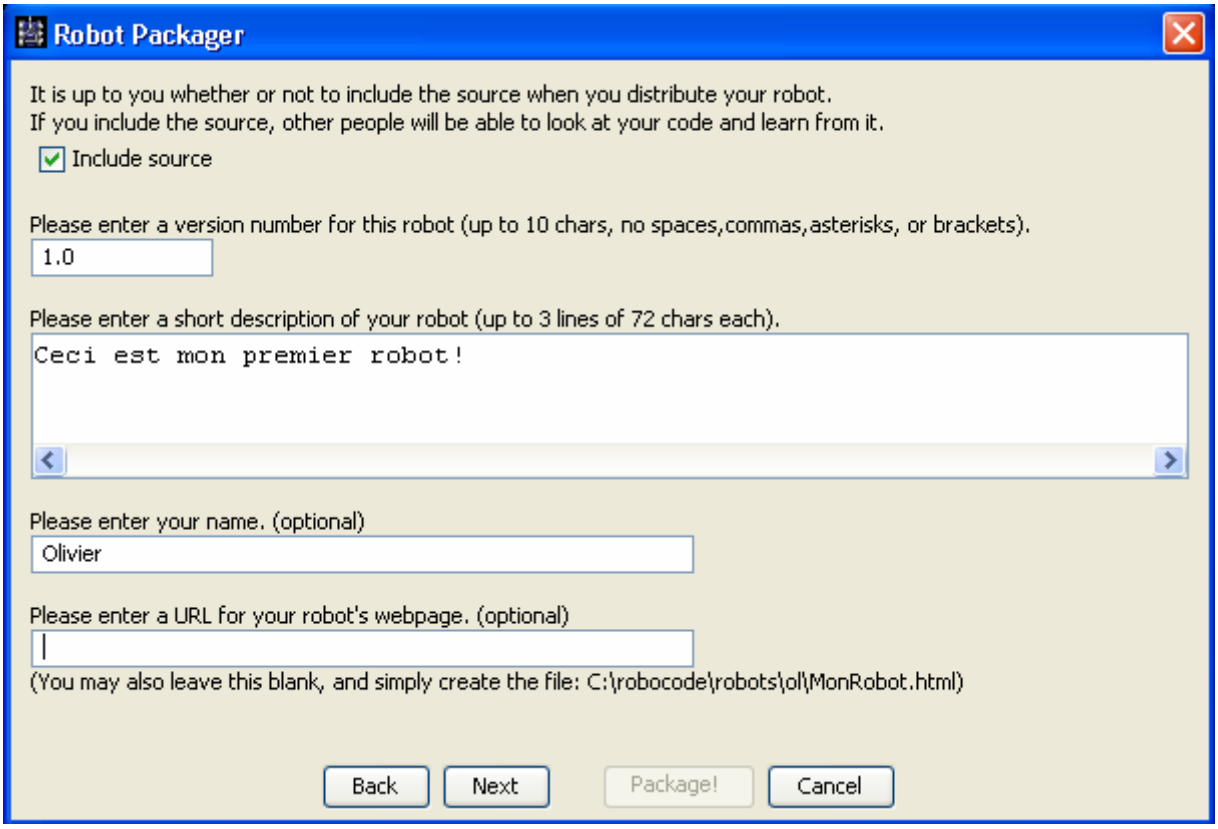
Pour commencer : sélectionnez "Robot\Package robot for upload" :



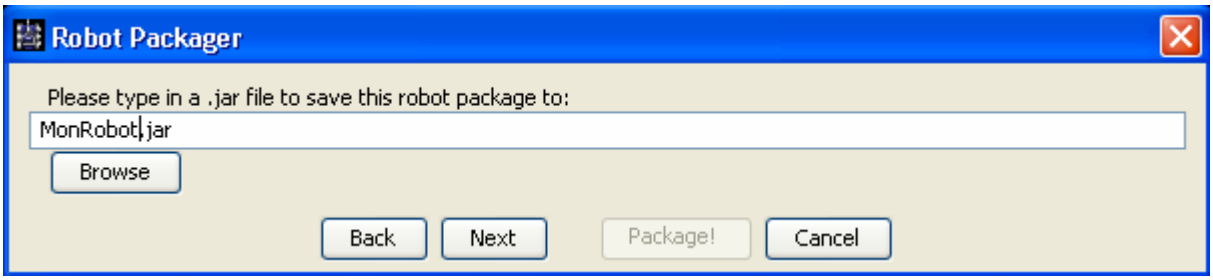
Ensuite, sélectionnez le robot que vous voulez partager. Dans l'exemple, "MonRobot" a été choisi :



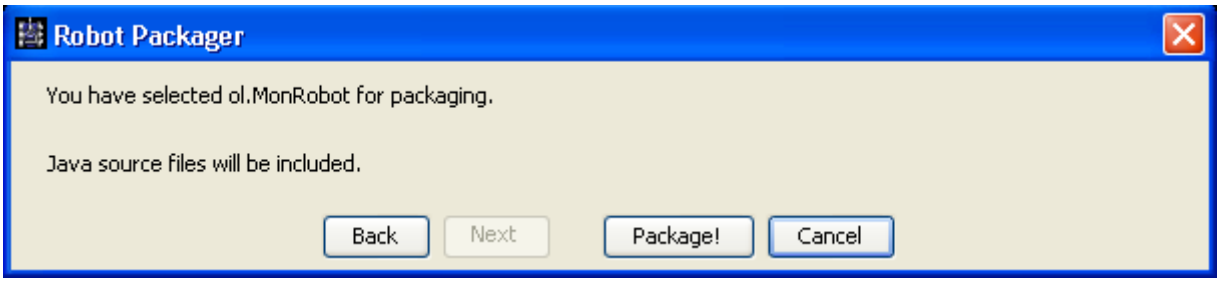
Cliquez sur le bouton « Next » et complétez la fenêtre suivante :



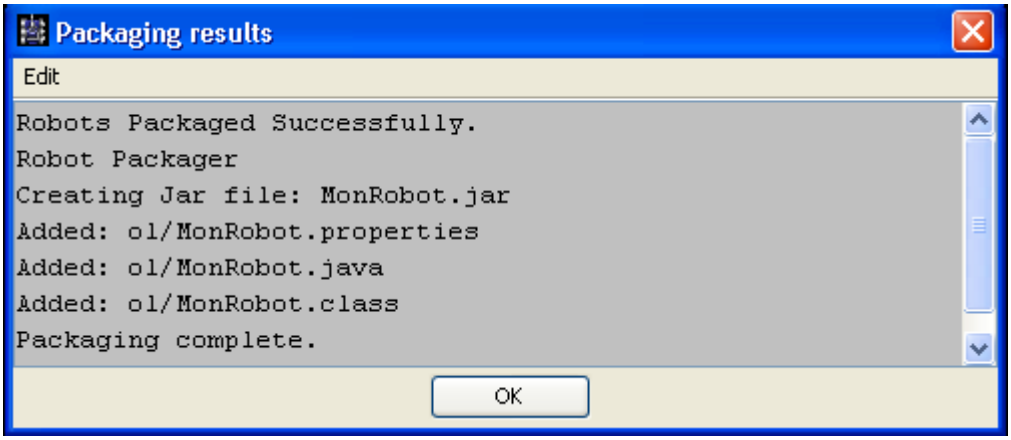
Cliquez à nouveau sur le bouton « Next », et choisissez un nom de fichier :



Cliquez sur le bouton "Package!" dans le dernier écran et Robocode créera un fichier « .jar » contenant votre robot :



Robocode affichera l’écran suivant :



Vous pouvez maintenant vous connectez à Robocode Repository afin de télépartager (uploader) votre nouveau fichier, dans l'exemple : C:\robocode\MonRobot.jar

Suite : Utiliser un IDE

Retour : Table des matières

Utiliser un IDE

Si vous êtes « fatigué » d'utiliser l'éditeur de Robocode, il est temps d'utiliser un outil de développement plus agréable, un IDE (Integrated Development Environment).
Ce tutorial va vous montrer comment utiliser Eclipse, disponible à l'adresse suivante : www.eclipse.org.

Eclipse a été utilisé pour écrire Robocode. Qu'attendez-vous ? Chargez-le maintenant !

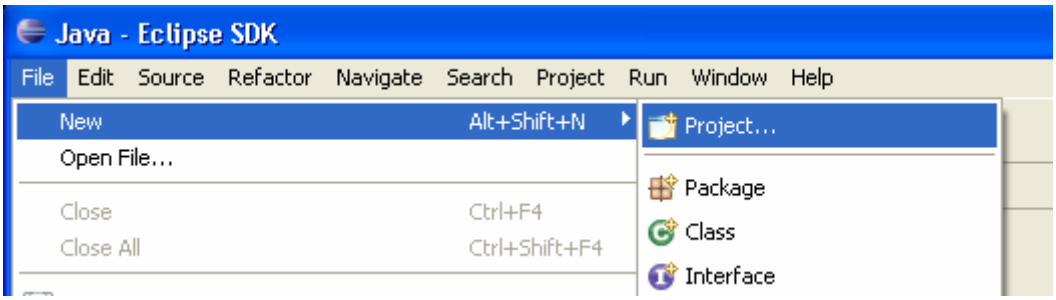
Une fois que vous avez installé et lancé Eclipse, vous poursuivrez en créant un projet pour vos robots.

Suite : Création d'un projet pour vos Robots
Retour : Table des matières

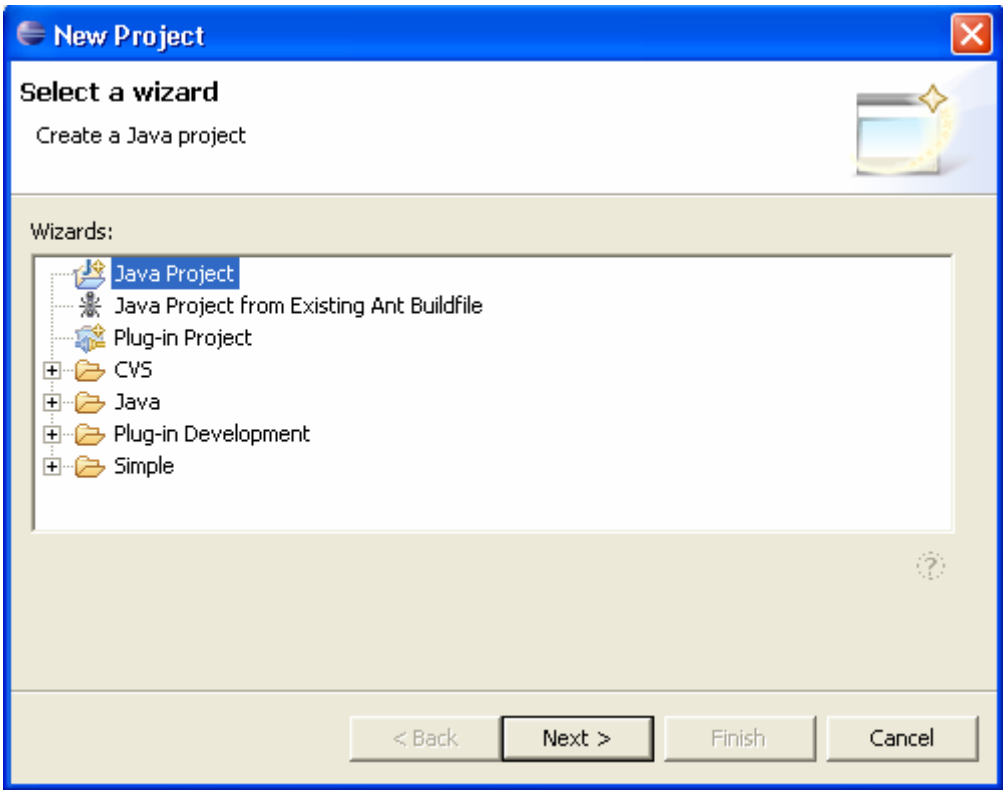
Création d'un projet pour vos Robots

Si vous voulez créer un robot en utilisant Eclipse, il faut d'abord créer un « Project » afin de les y stocker.

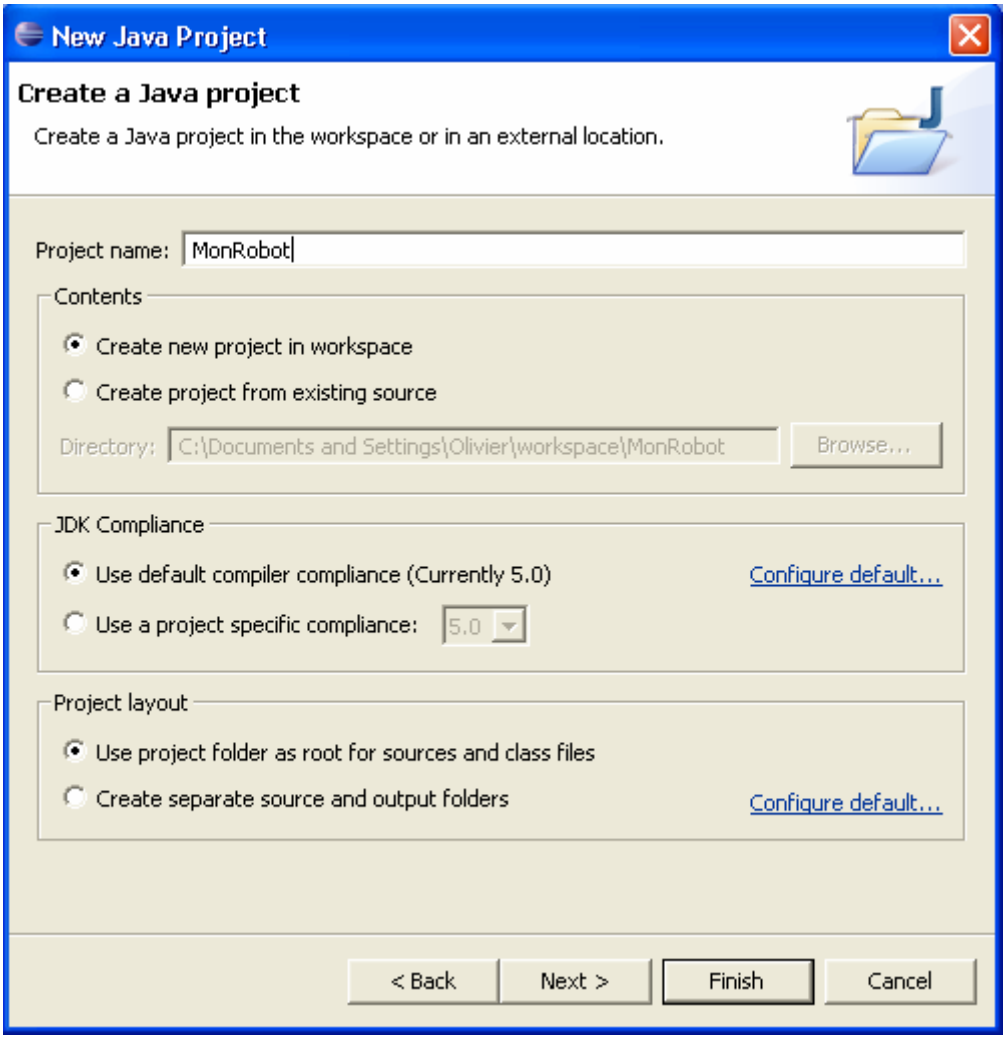
Commencez par sélectionner : « File\New\Project » :



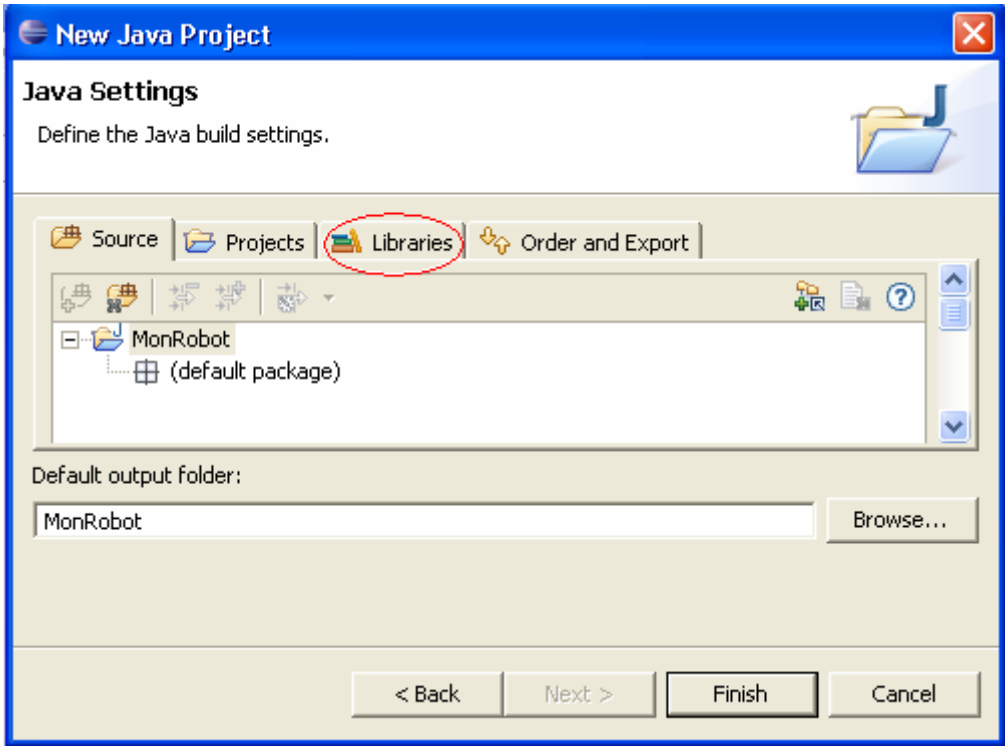
Ensuite, dans la fenêtre de dialogue « New Project », sélectionnez « Java Project » et cliquez sur « Next » :

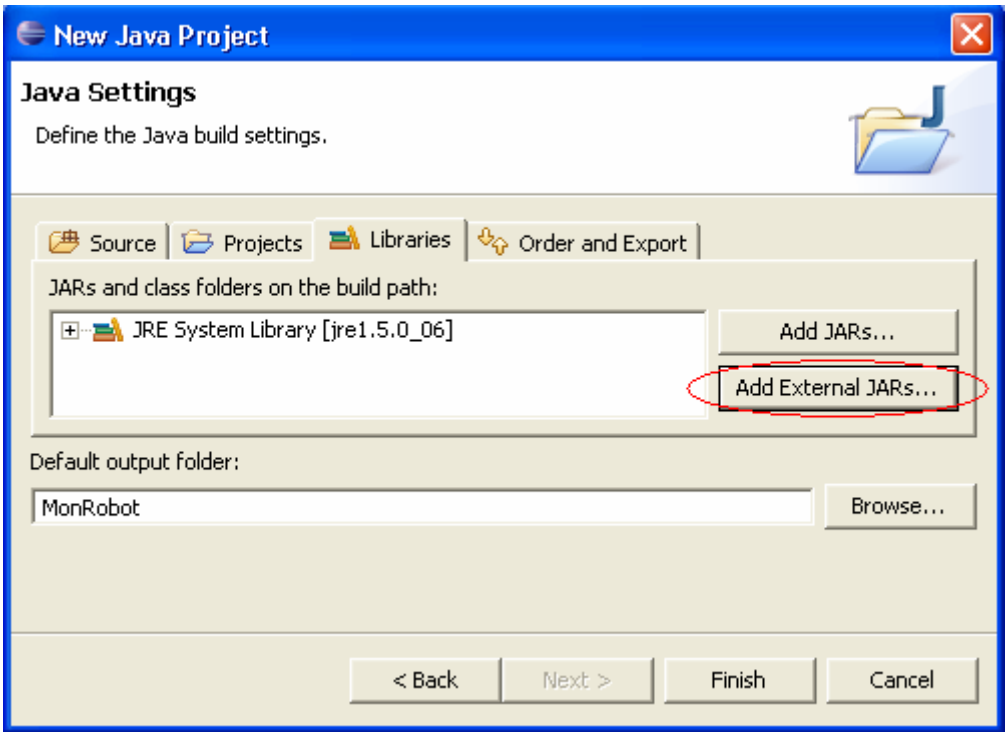


Introduisez un nom pour le projet, ensuite cliquez « Next ». **Ne** cliquez **pas** sur « Finish ».

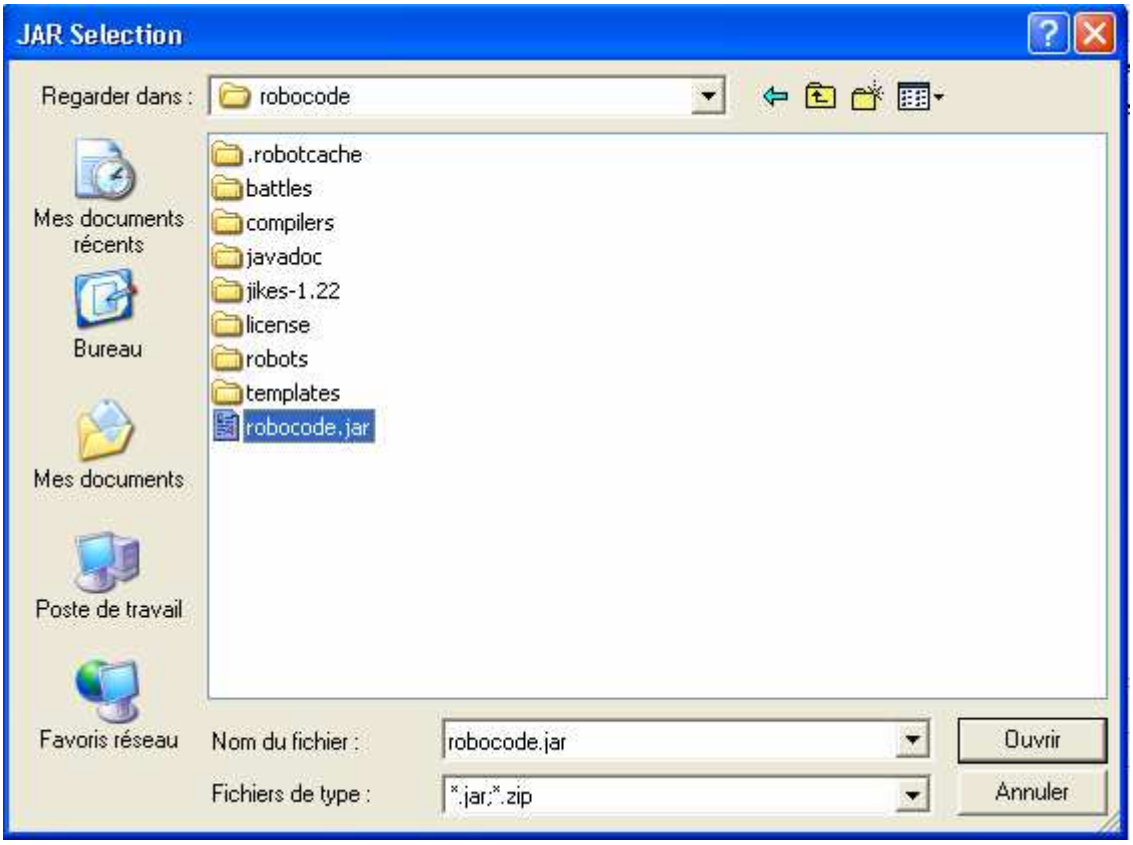


Sélectionnez l’onglet « Libraries », ensuite, cliquez sur "Add External JARs" :

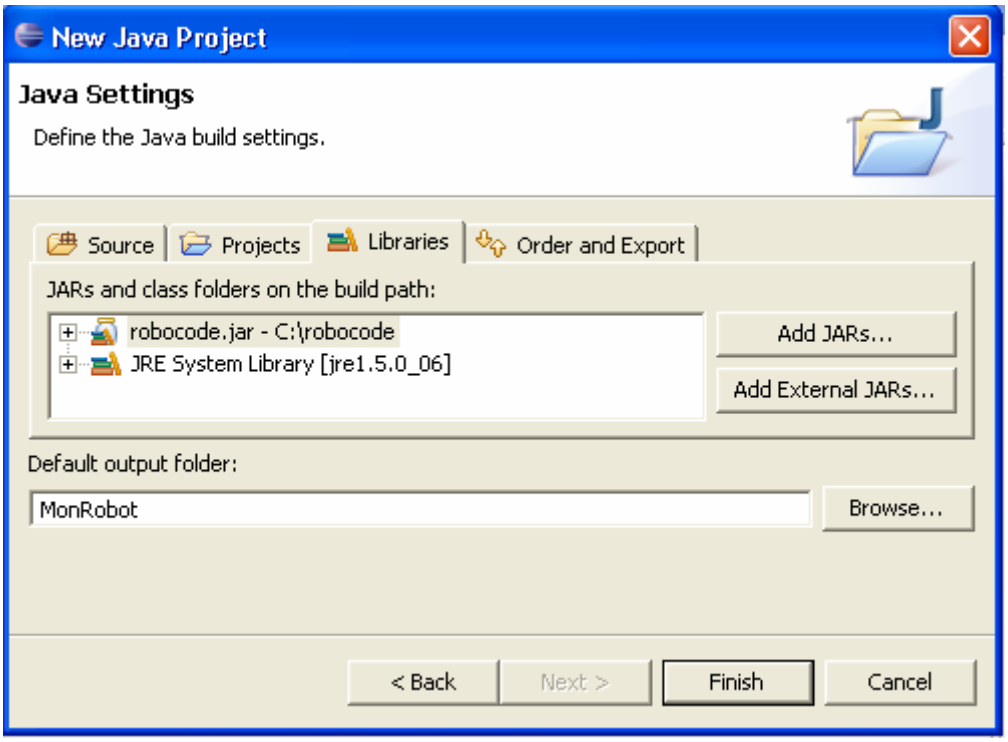




Parcourez le répertoire robocode, sélectionnez "robocode.jar", ensuite, cliquez sur "Ouvrir" :



La fenêtre affichée devrait ressembler à celle-ci :



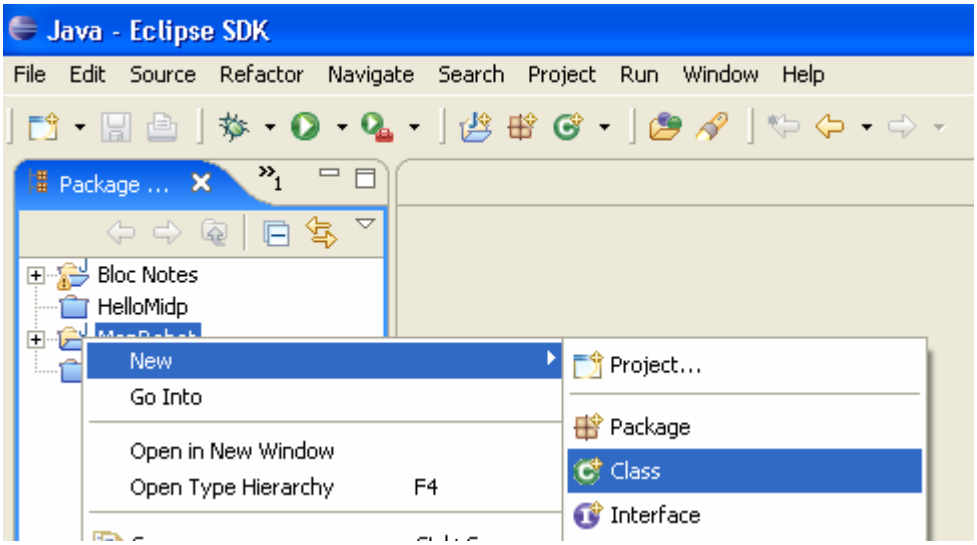
Terminez la création du projet en cliquant sur « Finish ». Voilà, le projet a été créé, dans l'exemple, il porte le nom : « MonRobot ».

Suite : Création d’un Robot avec Eclipse
Retour : Table des matières

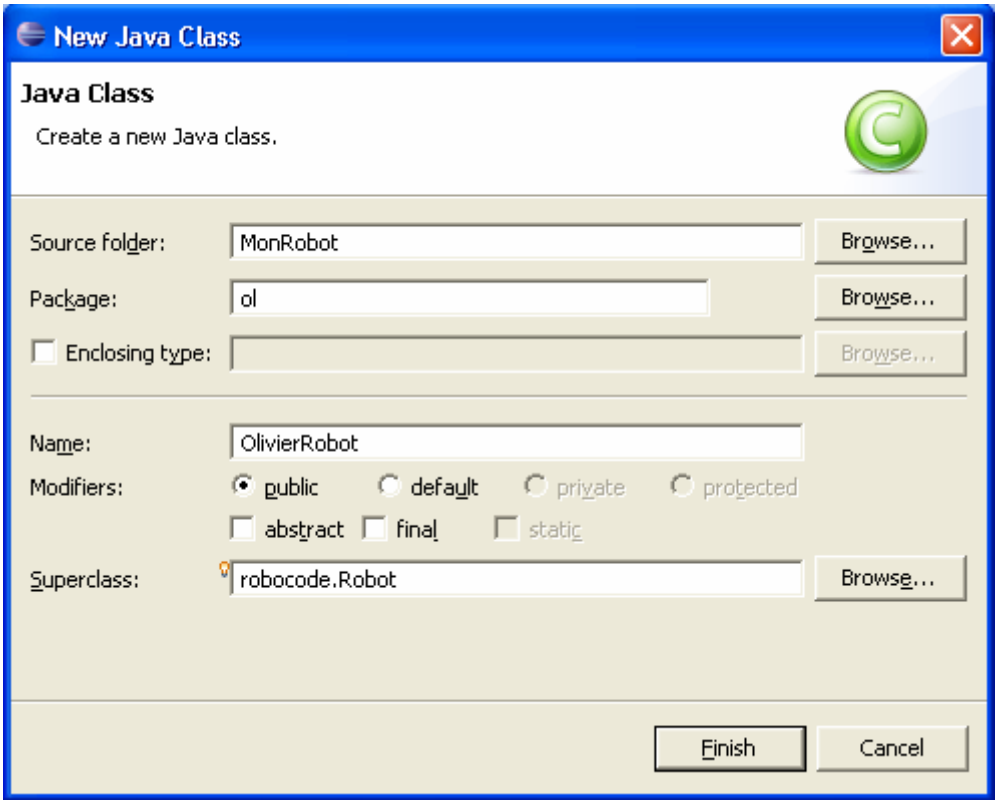
Création d’un Robot avec Eclipse

Avant de créer un robot avec Eclipse, assurez-vous d’avoir créer d’abord un projet : Création d’un projet pour vos Robots.

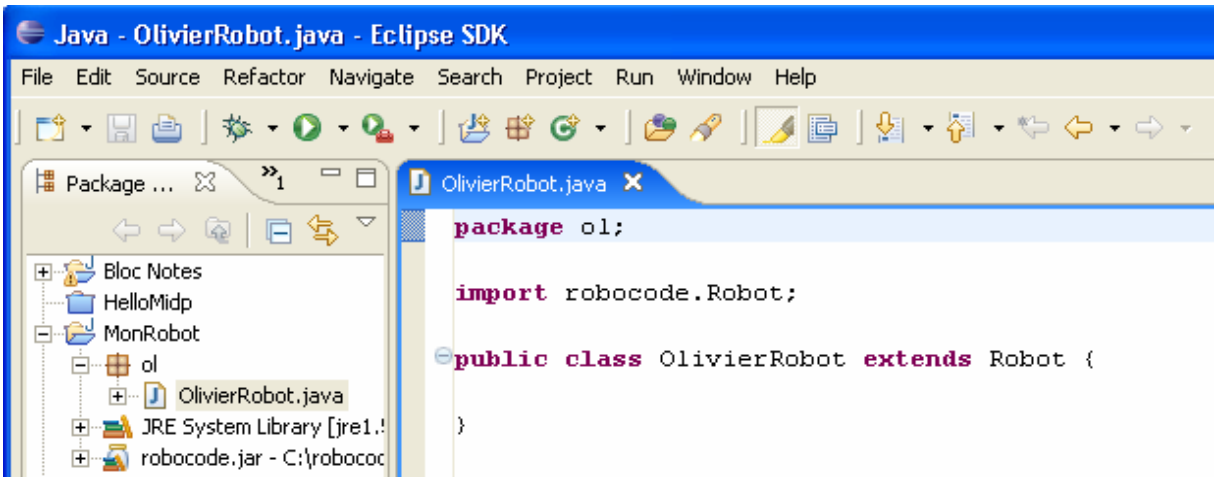
Ensuite, faites un clic droit de la souris sur le projet « MonRobot » et sélectionnez « New\ class » :



Ensuite introduisez le nom de « package », nous vous suggérons de taper vos initiales, suivi du nom de la classe, ici « OlivierRobot », et de changer Superclass en "robocode.Robot":



Cliquez sur « Finish » et vous verrez apparaître la classe de votre robot, comme ceci :



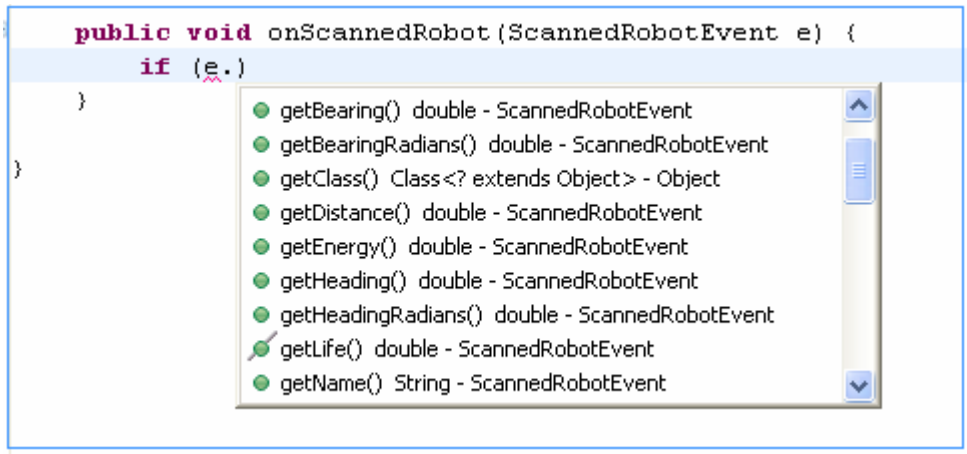
Il vous reste à compléter la classe en y ajoutant au minimum la méthode « run() » contenant quelques appels de méthodes et à sauver votre travail (CTRL-S) ou en cliquant sur la disquette :

OlivierRobot.java

```
package ol;

import robocode.*;

public class OlivierRobot extends Robot {

    public void run() {

        while(true) {
            ahead(100);
            turnGunRight(360);
        }
    }

    public void onScannedRobot(ScannedRobotEvent e) {
        fire(1);
    }

}
```

Eclipse est un outil de développement très puissant. Nous vous conseillons de suivre son tutorial :

Voici un petit truc bien pratique : lorsque vous ne vous souvenez plus exactement du nom d'une méthode, tapez « CTRL-ESPACE », Eclipse vous proposera les différentes méthodes disponibles…



Afin de tester votre nouveau robot dans l'arène de Robocode, il faut l'ajouter et lancer une bataille.

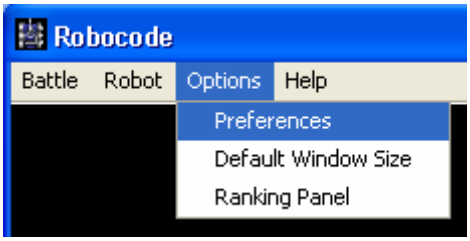
Suite : Ajout de vos robots à Robocode

Retour : Table des matières

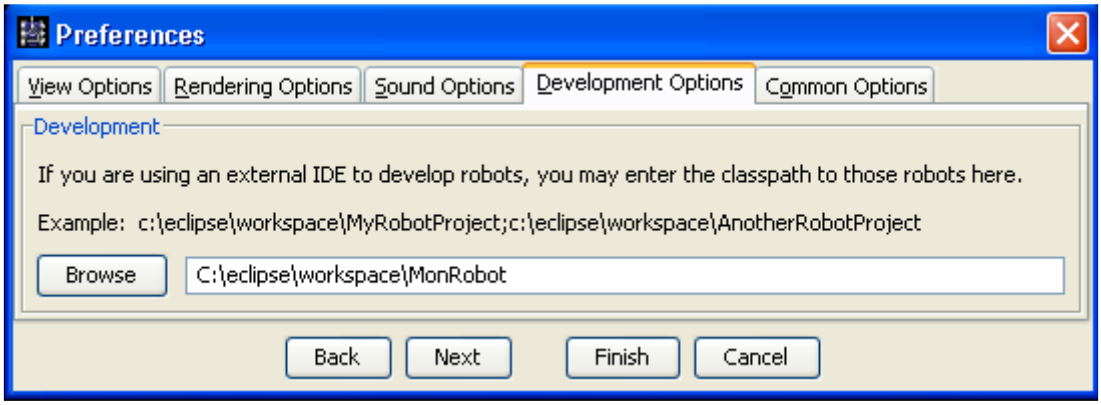
Ajout de vos robots à Robocode

Après avoir créé un nouveau robot avec Eclipse (voir Création d'un Robot avec Eclipse), vous devez configurer Eclipse afin de lui signaler où se situe votre robot, ensuite, vous pourrez le tester en lançant une nouvelle bataille.

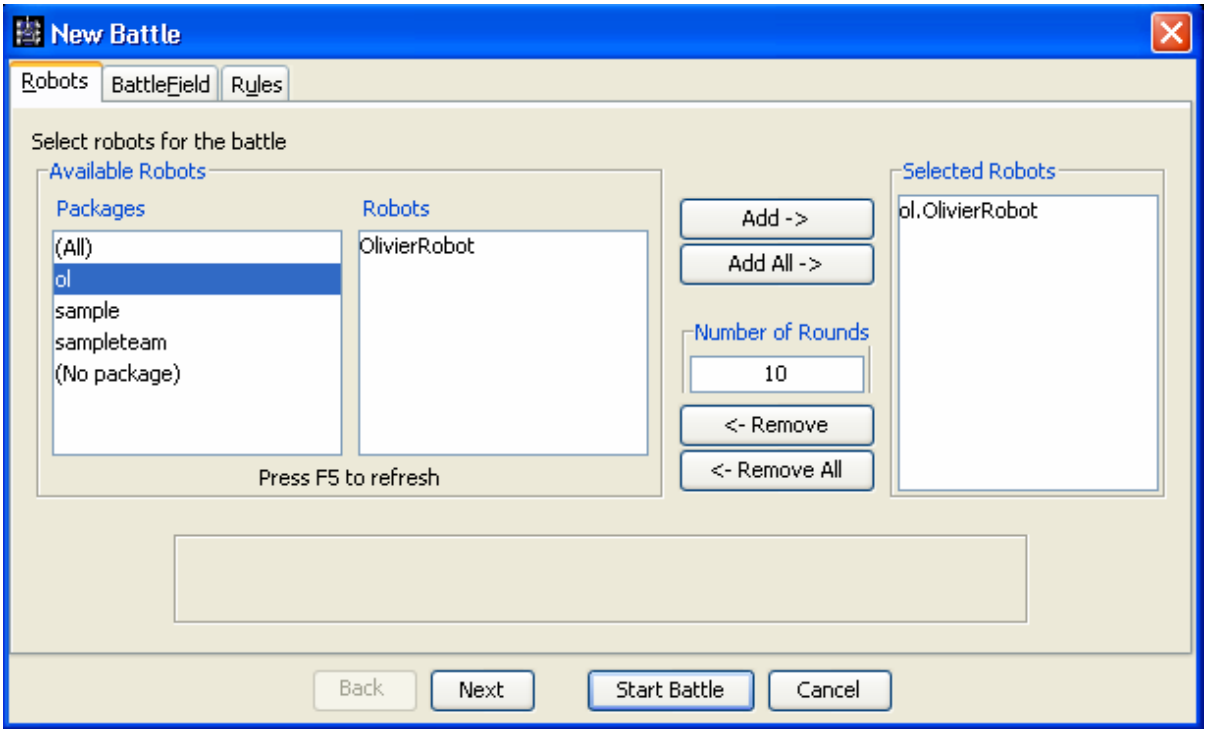
Commencez par sélectionner « Options\Preferences » :



Ensuite, sélectionnez l'onglet « Development Options ». Renseignez le chemin d'accès à l'espace de travail défini sous Eclipse suivi du nom de projet :



Enfin, sélectionnez et ajoutez votre robot à une nouvelle bataille :

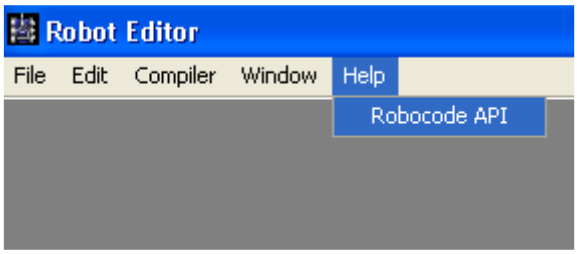

New Battle
Robots
BattleField
Rules
Select robots for the battle
Available Robots
Packages
Robots
(All)
ol
sample
sampleteam
(No package)
OlivierRobot
Press F5 to refresh
Add ->
Add All ->
Number of Rounds
10
<- Remove
<- Remove All
Selected Robots
ol.OlivierRobot
Back
Next
Start Battle
Cancel

Documentation « javadoc »

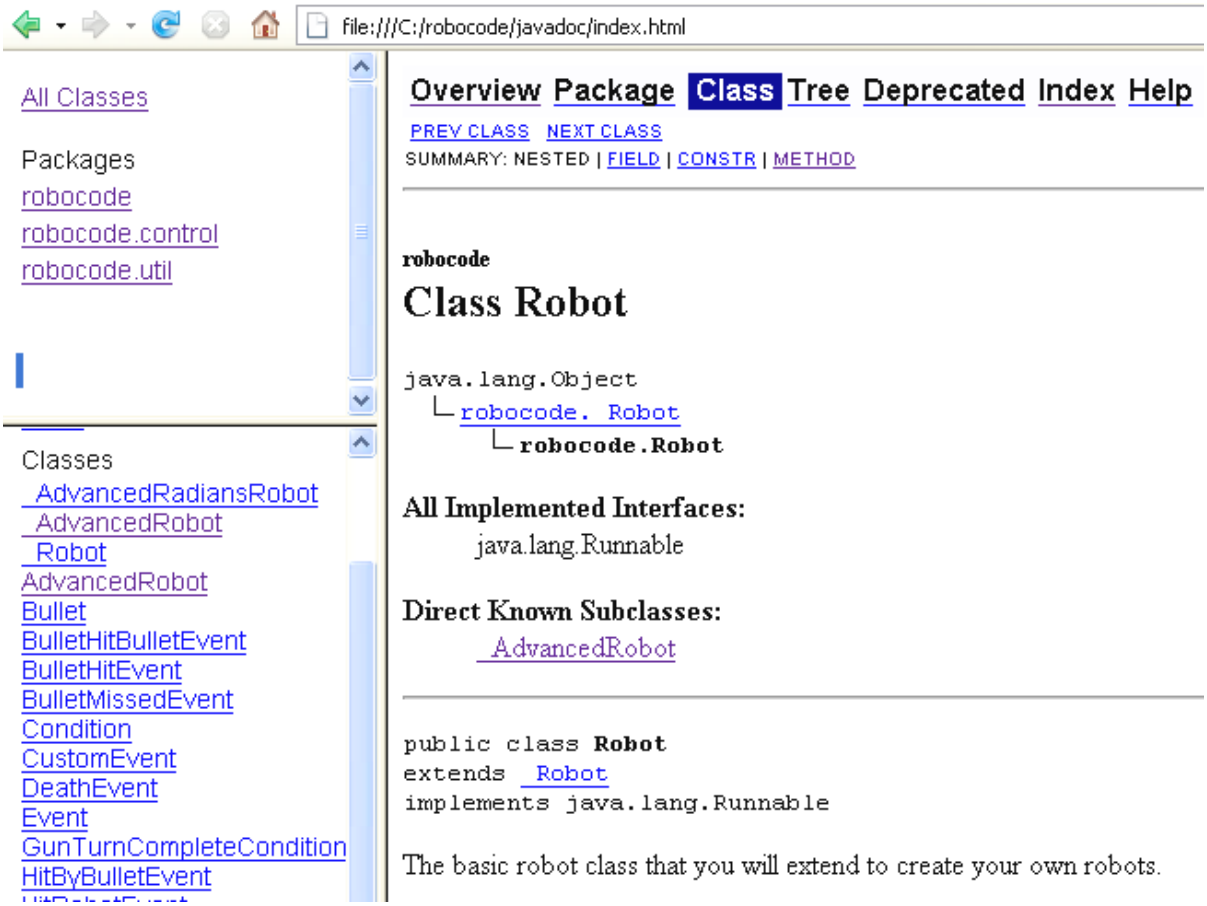
Les différentes classes de Robocode sont documentées en anglais. Cette documentation de type « javadoc » peut être consultée à l'aide d'un navigateur :

- soit en sélectionnant le fichier « index.html » disponible dans le répertoire d'installation de Robocode. Exemple : « C:\robocode\javadoc\ index.html » ;

- soit en sélectionnant dans l'éditeur de Robocode l'item suivant : « Help\Robocode API ».



Voici, à titre d'exemple, une copie d'écran de la documentation de la classe Robot :



Glossaire

A

- **ahead** : avancer le robot.

B

- **back** : reculer le robot.
- **bearing** : direction, orientation. (voir getBearing)
- **body** : partie principale du robot.
- **bullet damage** : point obtenu pour chaque dommage causé à l'ennemi.
- **bullet damage bonus** : points supplémentaires obtenus quand un robot détruit un ennemi. Il reçoit un bonus de 20% de tous les dommages qu'il lui a causés.

C

D

E

- **eclipse** : outil de développement d'applications. Il permet de créer, modifier et tester des classes Java.

F

- **fire** : tirer des obus.

G

- **getBearing** : demande d'une orientation. Il obtient un angle (exprimé en degrés) qui lui permet de situer l'ennemi, un mur ou un obus par rapport à sa propre trajectoire.
- **getDistance** : demande de la distance séparant le robot d'un autre objet (robot ennemi, mur, etc.
- **gun** : canon.

H

I

- **IDE** : « Integrated Development Environment. ». ou « Environnement de développement intégré ». Ce sont des outils de développement d'applications qui offrent de nombreuses fonctionnalités facilitant l'encodage, les modifications, les tests et le déploiement de programmes. jGrasp, Eclipse sont des exemple d'IDE.

J

- **javadoc** : ensemble de pages html documentant une ou plusieurs classes Java. Ces pages html sont générées en lançant la commande « javadoc ». Exemple : javadoc MonRobot.java
- **jGrasp** : outil de développement d'applications. Il permet de créer, modifier et tester des classes Java.

K

L

- **last survivor bonus** : score obtenu par le dernier robot en vie. Il reçoit 10 points de bonus pour chaque robot détruit avant lui.

M

N

O

- **onBulletHit** : que faire quand le robot a tiré et touché un autre robot.
- **onBulletHitBullet** : que faire quand un de ses obus a rencontré un autre obus :
- **onBulletMissed** : que faire quand un de ses obus s'écrase sur un mur (aucun robot touché);
- **onDeath** : que faire quand le robot « agonise ».
- **onHitByBullet** : que faire quand le robot est touché par un obus.

- **onHitRobot** : que faire quand il heurte un autre robot.
- **onHitWall** : que faire quand il heurte un mur.
- **onRobotDeath** : que faire quand un autre robot est détruit.
- **onScannedRobot** : que faire quand le radar du robot détecte un autre robot.
- **onWin** : que faire quand le robot a gagné le combat.

P
Q
R

- **ram damage** : point obtenu pour chaque dommage causé à l'ennemi en le tamponnant.
- **ram damage bonus** : points supplémentaires obtenus quand un robot détruit un ennemi en le tamponnant. Il reçoit un bonus de 30% de tous les dommages qu'il lui a causés.

S

- **setAdjustGunForRobotTurn(true)** : configurer le robot pour que le canon continue à pointer dans la direction initiale malgré la rotation du robot. Par défaut, lorsque le robot tourne, le canon tourne également dans la même direction avec la même amplitude (comme dans la réalité).
- **setAdjustRadarForRobotTurn(true)** : configurer le robot pour que le radar continue à pointer dans la direction initiale malgré la rotation du robot. Par défaut, lorsque le robot tourne, le radar et le canon tourne également dans la même direction avec la même amplitude (comme dans la réalité).
- **setAdjustRadarForGunTurn(true)** : configurer le robot pour que le radar continue à pointer dans la direction initiale malgré la rotation du canon. Par défaut, lorsque le robot tourne, le radar et le canon tourne également dans la même direction avec la même amplitude (comme dans la réalité).
- **setColors(…)** : colorier la partie principale (body), le canon et le radar dans différentes couleurs.
- **survival score** : score obtenu par un robot en vie. Il reçoit 50 points lorsqu'un autre robot est détruit.
- **survival 1sts, 2nds, 3rds :** indication de la durée de survie d'un robot.

T

- **total score** : total général, somme de tous les autres scores. Il détermine le score de chaque robot dans la bataille.
- **turnGunLeft** : tourner le canon vers la gauche.
- **turnGunRight** : tourner le canon vers la droite.

- **turnLeft** : tourner le robot vers la gauche.
- **turnRadarLeft** : tourner le radar vers la gauche.
- **turnRadarRight** : tourner le radar vers la droite.
- **turnRight** : tourner le robot vers la droite.

U
V
W
X
Y
Z

Retour : Table des matières