Robot

**La classe Point :** Elle sert à pouvoir utiliser un point qui a pour attribut une abscisse et une ordonnée.

**La classe Quadrillage :** Elle sert à créer un quadrillage qui représente les délimitations de déplacement qu’aura le robot. Il ne pourra se déplacer que sur ce quadrillage.

Je suis parti du principe qu’un quadrillage est composé de plusieurs points. Qu’il a pour maximum et minium d’abscisses et d’ordonnées les limites jusqu’aux quelles le robot pourra se déplacer.

**La classe Robot :** Cette classe nous permettra de manipuler des robots instanciés depuis celle-ci.

Ils prennent pour attributs :

* estAllume : booléen, il sert à pouvoir se référer à une valeur pour savoir si le robot est allumé ou non.
* estEnMarche : booléen, il sert à pouvoir se référer à une valeur pour savoir si le robot est en marche ou non.
* position : Point, cet attribut est de type Point .Il représente la position du robot à chaque instant. Il a une abscisse et une ordonnée.
* direction : double, il sert à pouvoir orienter le robot en fonction de sa valeur.
* vitesse : entier, il sert à pouvoir donner une vitesse de déplacement à notre robot. Elle aura pour valeur max par la suite 2 ( le nombre de cran qu’elle parcours par pas).
* quadrillage : Quadrillage, cet attribut est de type Quadrillage. Il permet au robot de se repérer en fonction de sa position et a des limites (abscisse et ordonnées max/min)

Ce robots ont les comportements suivants :

* La méthode allumer. Elle permet au robot, s’il est éteint, de s’allumer. En utilisant l’attribut estAllume.
* La méthode éteindre. Elle permet au robot, s’il est allumé, de s’éteindre. En utilisant l’attribut estAllume.
* La méthode arrêter. Elle permet si le robot est allumé et en marche(voir attributs correspondants), de réduire la vitesse jusqu’à ce qu’elle vaille zéro. A chaque appuie sur le bouton : diminution de la vitesse, ceci jusqu’à l’arrêt du robot.
* La méthode avancer. On y déclare un booléen auquel on affecte la condition où on ne touche pas les limites du quadrillage. Puis si le robot est allumé, et que le robot ne touche aucun bord, alors nous augmenterons la vitesse (jusqu’à un maximum de 2). Nous donnons la valeur TRUE à est en marche puisque le robot avance. Puis selon la direction du robot nous le faisons avancer, du nombre de cran(s) correpondant(s) à sa vitesse, dans sa direction. A chaque appuie sur le bouton, nous pouvons augmenter la vitesse.
* La méthode reculer est la même méthode qu’avancer mais le robot recule. C’est à dire que nous soustrayons la valeur de la vitesse(nombre de cran(s)) a la position au lieu de l’ajouter.
* La méthode tournADroite. Elle permet au robot de tourner à droite. Selon son orientation il prendra celle de 90° vers la droite. ( à savoir PI/2).
* La méthode tourneAGauche est la même que celle vers la droite. Cependant vers la gauche.
* Enfin la méthode action. Elle est faite d’un switch case. Selon la valeur que prend l’appuie sur le bouton nous exécutons une des méthodes ci-dessus.Les valeurs des boutons sont listés dans l’énumération Enum Bouton déclarée en haut de la classe robot.

Les méthodes : allumer, éteindre, arrêter, avancer, reculer, tourneADroite et tourneAGauche retourne des booléens. Les valeurs que prendrons ces booléens renverrons si l’action à été effectuée ou non.