**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 : Enumération de direction. :GAUCHE,DROITE,HAUT,BAS.
* 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)

Ces 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 : on déclare qu’il n’est plus en marche, que ses vitesses de déplacements valent 0 et qu’il est éteint.
* La méthode arrêter. Elle permet si le robot est allumé et en marche 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.. Si la vitesse vaut 0, le robot s’arrête.
* La méthode estPositionValide. Elle permet de retourner la valeur TRUE si le robot n’est à aucune limite de son déplacement FALSE sinon.
* La méthode avancer. Nous augmenterons la vitesse (jusqu’à un maximum de 2).Si le robot est allumé, nous testons si le robot est à sa limite de déplacement. Si oui, nous donnons la valeur false à estEnMarche et nous mettons la vitesse a zéro. Si le robot peut aller à destination(pas hors limite)nous donnons la valeur TRUE à est en marche puisque le robot avance et il se déplace. Puis selon la direction du robot nous le faisons avancer dans cette direction. A chaque appuie sur le bouton, nous pouvons augmenter la vitesse.
* La méthode tournADroite. Elle permet au robot de tourner à droite. Selon son orientation il prendra celle de 90° vers la droite. Oriente le robot à sa droite en fonction de sa direction actuelle. Les directions sont listées dans l’énumération Direction.
* 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 listées dans l’énumération Bouton.

Les méthodes : allumer, éteindre, arrêter,estPositionValide, avancer, tourneADroite et tourneAGauche retourne des booléens.

Le constructeur sans paramètres créé le robot au point 0,0 avec une direction vers le haut et aucune vitesse de déplacement .Il est éteint et pas en déplacement.