Documentation pour la bibliothèque logicielle pour le Concours de déviations dû au TLR

Mise à jour le 13 novembre 2016

Table des matières

IRControl

Cette classe représente les capteurs infrarouges qui vont détecter les lignes.

```
class IRControl{
	/*
	 Constructeur qui cré l'objet "IR sensor"
	 @parm pin: le analogPin que le capteur utilise
	*/
	IRControl(int pin);

	/*
	Sous-programme qui determine si le capteur lit le noir
	@returns: true s'il est noir
	*/
	boolean isBlack();
};
```

UltrasonicControl

Cette classe représente les capteurs utilisés pour la détection d'obstacles. L'utilisateur a seulement besoin de demander pour une distance, car la classe va déclencher le capteur à ultrasons.

```
class UltrasonicControl{
	/*
		Constructeur, cré l'objet pour le capteur
		@param trigPin: le trigger pin sur le capteur
		@param echoPin: le  echo pin sur le capteur
	*/
	UltrasonicControl(int trigPin,int echoPin);

	/*
		Détermine la distance de l'object à lui
		@return: distance en cm, -1 s'il y a une erreur
	*/
	int getDistance();

	/*
		Détermine si un objet est dans le carré adjacent.
		@return: true si un objet est détecté
	*/
	boolean detect();
};
```

MotorControl

Cette classe représente les moteurs. Il communique avec le moteur en utilisant le contrôleur de moteur.

```
class MotorControl
{
	/*
		Constructeur, prend les broches utilisées et règle le pinmode
		@param pwmPin: la broche utilisée pour contrôler la vitesse du moteur
					Pour moteur un pwmPin = 5
					Pour moteur deux pwmPin = 6
		@param dirPin: la broche utilisée pour contrôler la direction
					Pour moteur un dirPin = 4
					Pour moteur deux dirPin = 7
	*/
	MotorControl(int pwmPin,int dirPin);

	/*
		Constructeur qui prenne le moteur qu'il veut utiliser et règle la broche
		Un problème avec ce constructeur est qu'il pourrait utiliser une broche
			occuppée et commend à l'utiliser pour quelque chose d'autre
		@param motorNum: Le moteur dont l'objet représentera. Doit être un
					un ou un deux
	*/
	MotorControl(int motorNum);

	/*
		Bouge le moteur en avant
		@param speed: la vitesse du moteur
	*/
	void forward(int speed);

	/*
		Faire la marche arrière du moteur
		@param speed: la vitesse du moteur
	*/
	void reverse(int speed);

	/*
		Arrête le moteur
	*/
	void halt    ();
};
```