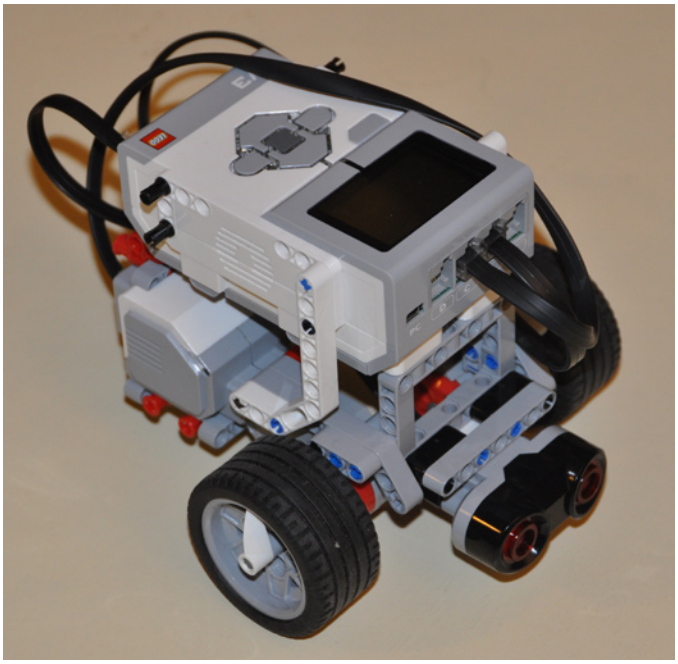
Iniciación a la robótica con LEGO® MINDSTORMS, 18ª entrega

Lenguajes alternativos de programación para EV3: RobotC

Por Koldo Olaskoaga

Gracias a las aportaciones de las comunidades de usuarios y las iniciativas comerciales, LEGO® MINDSTORMS RCX y NXT pueden ser programados por medio de entornos de programación muy diversos, tanto gráficos como textuales. Si bien el ecosistema de LMS EV3 todavía no está a la par de los anteriores, se dispone de diversas opciones además del entorno oficial EV3-G. En este artículo voy a escribir sobre RobotC y en próximos números de HispaBrick Magazine® lo haré sobre otros.

Para los ejemplos utilizo el modelo básico de la versión educativa de EV3, pero otros robots con sistema de dirección diferencial ofrecerían un comportamiento similar.



RobotC

RobotC es un entorno de programación basado en C pensado para ser utilizado en educación y competiciones con carácter educativo, ofreciendo un entorno amigable para aquellas personas que se acercan por primera vez a C.

Tras una serie de versiones beta puestas a disposición de los usuarios para su testeo, a finales de agosto se publicó la primera versión oficial del nuevo RobotC 4.x para LEGO MINDSTORMS EV3, una versión que había llegado antes a otras plataformas.

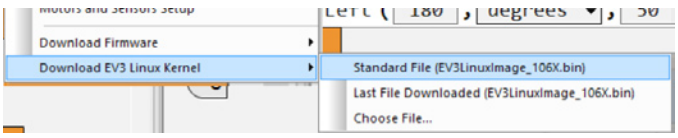
Esta nueva versión, además de dar soporte tanto al NXT como al EV3, ofrece una novedad importante: un sistema de programación gráfico que puede facilitar la transición de otros lenguajes gráficos a un lenguaje textual como C.

El firmware de RobotC

El EV3 para poder ejecutar los programas que se descargan en él, necesita tener instalado un pequeño programa llamado firmware. De este modo el EV3 puede comprender los programas que se descargan en él y ejecutarlos. Periódicamente pueden publicarse actualizaciones para mejorar sus prestaciones.

RobotC necesita que en el EV3 esté instalado su propio firmware, ya que no es compatible con el firmware original de LEGO. Sin embargo, esta incompatibilidad no se produce en el sentido contrario, lo que quiere decir que con el firmware de RobotC se pueden ejecutar programas creados tanto con RobotC como con EV3-G.

Por eso, lo primero que habrá que hacer será instalar el firmware propio de RobotC. Para ello se empieza descargando desde el menú Robot el **EV3 Linux Kernel** y, a continuación, se descarga el firmware. De este modo ya tendremos preparado el EV3 para RobotC.



El entorno gráfico de programación

Este nuevo entorno de programación recuerda de cierto modo a Scratch, básicamente debido a que las estructuras de control, de color naranja, se abren para contener en su interior las instrucciones que haya que ejecutar. Las

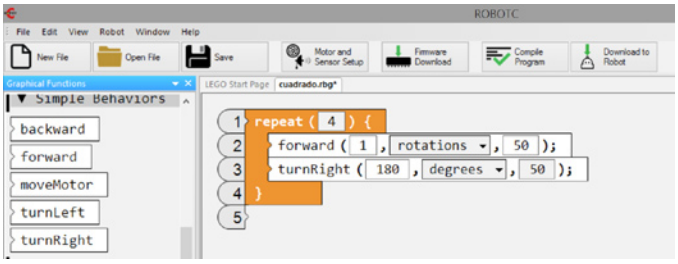
instrucciones se colocan donde se desea con un simple arrastrar y soltar.

Este modo gráfico utiliza una paleta de funciones basada en lo que denomina lenguaje natural. Se trata de un lenguaje de alto nivel, es decir, un lenguaje similar al que las personas utilizamos para comunicarnos. Entre los grupos de funciones puede destacarse el de comportamientos simples en el que encontramos las funciones avanzar, retroceder, girar a la izquierda, girar a la derecha y mover un motor. Estas funciones facilitan el aprendizaje y requieren una configuración específica de los motores y sensores para que funcione del modo esperado, motor izquierdo en el puerto B, derecho en el puerto C y sensor de ultrasonidos en el puerto 4.

Mi primer programa con RobotC

Vamos a empezar con una tarea sencilla de programar, hacer que el robot se mueva describiendo un cuadrado para detenerse en el punto de inicio. Los pasos a dar para ello se pueden resumir en los siguientes: repetir 4 veces avanzar y girar a la derecha.

Veamos cómo crear este programa en el entorno gráfico de RobotC. Recordemos que hemos de utilizar la configuración de motores que trae por defecto, que es el motor izquierdo en el puerto B y el derecho en el C.



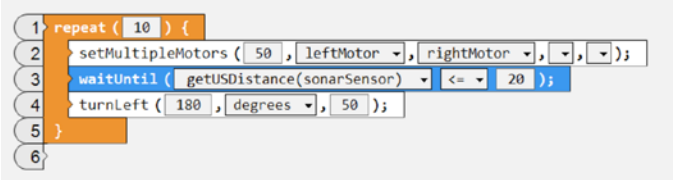
En la imagen podemos ver el programa, con la paleta de funciones a la izquierda. Aquí se utilizan además de la estructura repetir dos bloques de programación: **forward** (adelante) y **turnRight** (girar a la derecha). El tercer parámetro en cada una de estas funciones es la potencia, en este caso 50. Poder utilizar las funciones del grupo comportamientos simples, hace que este programa sea muy sencillo.

Si el robot lo hubiésemos montado con LEGO® WeDo, lo podríamos controlar desde Scratch con el siguiente programa u otro similar. Puede apreciarse la similitud en el diseño gráfico, aunque en este caso el programa es más sencillo en RobotC.

Dadas las características de los motores que se utilizan con WeDo, los motores Technic PF, cuando se montan simétricos uno a cada lado, deberán girar en sentido contrario para que el vehículo se mueva hacia adelante.

Robot con sensor ultrasonidos que evita obstáculos

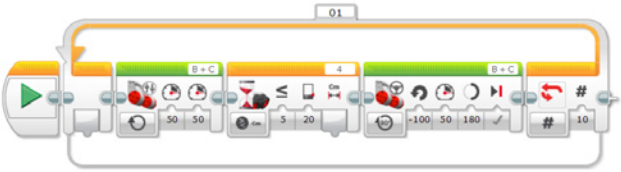
En este nuevo programa vamos a combinar el uso de motores y un sensor. El objetivo es que el robot avance en línea recta hasta que la distancia a un obstáculo sea menor o igual a 20 cm. En ese momento ha de girar y repetir lo anterior para que una vez que detecte obstáculos por 10 veces se detenga. La secuencia de pasos a no tiene mayor misterio, repetir 10 veces poner en marcha el robot para que se desplace en línea recta, esperar hasta encontrar obstáculo y girar.



El programa es sencillo y fácil de interpretar. En este caso no se puede utilizar el bloque **forward** utilizado en el caso anterior, ya que dicho bloque solo puede ser utilizado cuando se quiere que el robot avance por una distancia o tiempo determinado. Aquí lo que necesitamos es un bloque que ponga los motores en marcha y permita que el programa continúe su ejecución y pase al siguiente paso. Para ello se utiliza el bloque **setMultipleMotors**, que permite poner en marcha todos los motores que se desee simultáneamente.

Tras poner en marcha el robot la siguiente instrucción, en azul, establece que se espera hasta que la distancia que mide el sensor de ultrasonidos sea menor o igual a 20 cm. A continuación gira y vuelve a empezar.

El programa equivalente en EV3G sería el siguiente:



Puede apreciarse, que salvando el diferente diseño gráfico, es posible establecer un paralelismo entre los dos programas.

Convertir el programa en uno textual

Una de las características interesantes del nuevo entorno gráfico de RobotC es que ayuda en el paso de un entorno gráfico a uno textual. Para ello dispone de una herramienta que permite convertir el programa en formato gráfico al correspondiente en el formato textual de RobotC. Por ejemplo, utilizando la herramienta Convert Graphical File to Text del menú View el

programa anterior lo convertiría en el siguiente:

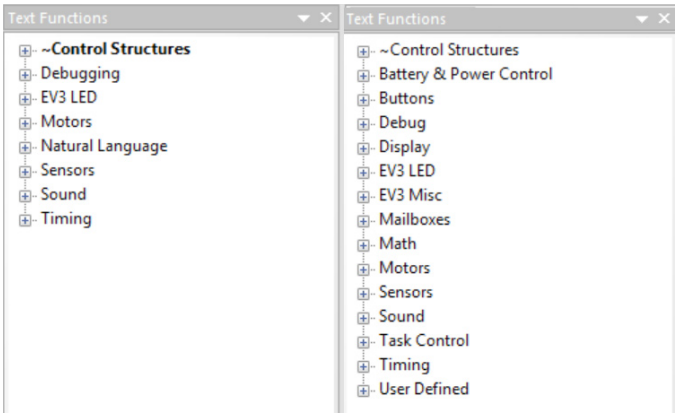
```
1  #pragma config(Sensor, S1,     touchSensor,    sensorEV3_Touch)
2  #pragma config(Sensor, S2,     gyroSensor,     sensorEV3_Gyro)
3  #pragma config(Sensor, S3,     colorSensor,    sensorEV3_Color)
4  #pragma config(Sensor, S4,     sonarSensor,    sensorEV3_Ultrasonic)
5  #pragma config(Motor,  motorA,          armMotor,      tmotorEV3_Large, PIDControl, encoder)
6  #pragma config(Motor,  motorB,          leftMotor,     tmotorEV3_Large, PIDControl, driveLeft, encoder)
7  #pragma config(Motor,  motorC,          rightMotor,    tmotorEV3_Large, PIDControl, driveRight, encoder)
8  //*!!Code automatically generated by 'ROBOTC' configuration wizard               !!*//
9
10
11 task main()
12 {
13   repeat (10) {
14     setMultipleMotors(50, leftMotor, rightMotor, , );
15     waitUntil (getUSDistance(sonarSensor) <= 20);
16     turnLeft(180, degrees, 50);
17   }
18 }
19
```

La parte superior del código la genera de modo automático RobotC a partir del modo en el que se hayan configurado los motores y sensores. El programa propiamente dicho es el fragmento comprendido entre las líneas 11 y 18. En este caso, dada la sencillez del programa y el uso del lenguaje natural, los programas son muy parecidos, pero cuando se programa en modo textual la sintaxis es muy importante y olvidar cerrar una línea con un “;” genera errores.

Entorno textual

Una vez que se está familiarizado con el entorno gráfico toca pasar a la programación textual. Aquí también hay dos modos de programación diferentes, el lenguaje natural, que es el que también se utiliza en la programación gráfica, y el RobotC basado en texto. En este segundo modo, hay tres niveles, el básico, el experto y el super usuario. En el modo básico se limitan las funciones disponibles para la programación así como opciones disponibles en preferencias. En el experto se puede acceder a la paleta de funciones completa mientras que en el super usuario abrirá todas las puertas de RobotC.

En la siguiente imagen pueden verse los conjuntos de funciones que ofrece cada uno de ellos, el lenguaje natural a la izquierda y el de la derecha el basado en texto del nivel experto.

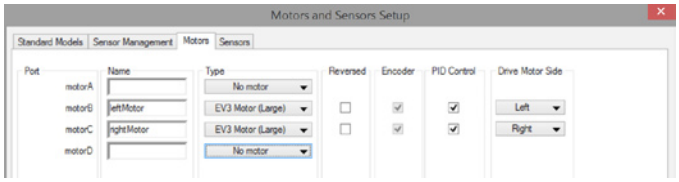


Puede observarse la diferencia entre las dos paletas de funciones, si se opta por el lenguaje natural va a resultar más sencillo desarrollar un programa, pero se podrán crear programas más sofisticados cuando se puede acceder a todas las funciones de RobotC.

A continuación se presenta el ejemplo anterior utilizando el modo textual de RobotC. Puede apreciarse que los motores se controlan de modo individual, y que en lugar de la estructura repetir un número determinado de veces se utiliza la estructura For, habitual en los diferentes lenguajes de programación.

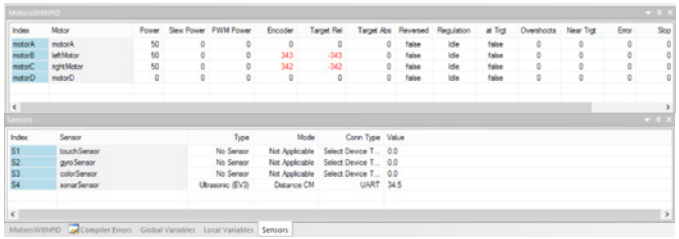
```
1  #pragma config(Sensor, S4,     Sonar,          sensorEV3_Ultrasonic)
2  #pragma config(Motor,  motorB,          leftMotor,     tmotorEV3_Large, PIDControl, driveLeft, encoder)
3  #pragma config(Motor,  motorC,          rightMotor,    tmotorEV3_Large, PIDControl, driveRight, encoder)
4  //*!!Code automatically generated by 'ROBOTC' configuration wizard               !!*//
5
6  task main()
7  {
8  for(int i = 0; i <= 9; i++)/* inicializa la variable i con el valor para incrementar su valor de 1 en 1
9                                hasta que llegue a 9 (es decir, 10 bucles)*/
10 {
11     motor[rightMotor] = 50;   // el motor derecho a potencia 50
12     motor[leftMotor]  = 50;   // el motor izquierdo a potencia 50
13     while (getUSDistance(Sonar) > 20){}; // el robot en marcha mientras la distancia medida sea > 20
14
15     motor[rightMotor] = -50;  //el motor derecho a potencia -50
16     moveMotorTarget(leftMotor, 180, 50); //el motor izquierdo avanza 180 grados
17     waitUntilMotorStop(leftMotor); //el flujo del programa espera hasta que el motor izquierdo avance 180 grados
18 }
19 motor[rightMotor] = 0;  // detener el motor derecho
20 motor[leftMotor]  = 0;  // detener el motor izquierdo
21 }
```

Cuando se programa en el modo texto, los motores y sensores hay que configurarlos previamente.
En la siguiente imagen puede verse la ventana de configuración de los motores con sus diversas opciones.



El depurador

Para cuando las cosas no van como se desee, RobotC dispone de un depurador que permite observar en tiempo real cómo cambian los valores de los motores, sensores, temporizadores, variables… de tal modo que se facilite la identificación de los problemas. A esto hay que sumar la posibilidad de ejecutar un programa paso a paso, todo esto con el robot conectado al ordenador por cable o de modo inalámbrico. En la imagen siguiente puede verse el estado de los motores y sensores durante la ejecución del programa anterior.



En resumen, RobotC es un entorno de programación comercial con un interface gráfico que facilita la iniciación en la programación textual y permite aprovechar el potencial del EV3. Se utiliza en varias competiciones de robots tales como FTC y VEX Robotics. Dispone de un activo foro oficial que da soporte a los usuarios, principalmente en inglés. Periódicamente publican actualizaciones.
#