



# GRADO EN INGENIERÍA EN TECNOLOGÍA DE LAS TELECOMUNICACIONES

Curso Académico 2021/2022

Trabajo Fin de Grado

## ROBÓTICA EDUCATIVA CON PYTHON Y MBOT

Autor : Eva García Domingo  
Tutor : Jose María Cañas Plaza



# Índice general

|   |           |
|---|-----------|
| <b>1. Introducción</b>  | <b>1</b>  |
| <b>2. Objetivos</b>   | <b>2</b>  |
| 2.1. Objetivos  | 2         |
| 2.1.1. Programación en Python de un robot basado en Arduino                                   | 2         |
| 2.1.2. Propuesta educativa completa, basada en Scratch y Python y escalonada según dificultad | 2         |
| 2.2. Requisitos   | 3         |
| 2.3. Metodología  | 3         |
| <b>3. Infraestructura</b>   | <b>5</b>  |
| 3.1. Entorno  | 5         |
| 3.2. Hardware   | 6         |
| 3.2.1. Placa Arduino  | 6         |
| 3.2.2. MBot   | 6         |
| 3.2.2.1. Sensores   | 8         |
| 3.2.2.2. Actuadores   | 9         |
| 3.3. Software   | 11        |
| 3.3.1. Scratch y mBlock   | 11        |
| 3.3.2. Arduino  | 14        |
| 3.3.3. Python   | 15        |
| <b>4. Plataforma PyBo-Kids</b>  | <b>17</b> |
| 4.1. Programa residente   | 17        |
| 4.2. Programa PC  | 17        |
| 4.3. Diseño   | 17        |
| 4.4. Resultado: programa PC y programa residente  | 17        |
| <b>5. Aplicación educativa</b>  | <b>18</b> |
| 5.1. Contexto   | 18        |
| 5.2. Primera etapa: Scratch   | 18        |
| 5.2.1. Objetivos docentes: metodología  | 18        |
| 5.2.2. Prácticas  | 18        |
| 5.3. Segunda etapa: PyBo-Kids en Python   | 18        |
| 5.3.1. Objetivos docentes: metodología  | 18        |
| 5.3.2. Introducción a Python  | 18        |
| 5.3.3. Prácticas  | 18        |

|                        |           |
|------------------------|-----------|
| <i>ÍNDICE GENERAL</i>  | 1         |
| <b>6. Conclusiones</b> | <b>19</b> |
| <b>Bibliografía</b>    | <b>20</b> |



# Capítulo 1

## Introducción

Esto es un ejemplo. Ver en [1] [2]

# Capítulo 2

## Objetivos

En este explicaremos los objetivos de este Trabajo Fin de Grado, así como la intencionalidad de ellos y la metodología para llevarlos a cabo.

### 2.1. Objetivos

Como se ha adelantado en la introducción, el carácter, y por tanto el objetivo, principal de este Trabajo Fin de Grado es educacional. Siguiendo este carácter, abordaremos dos caminos complementarios, explicados a continuación.

#### 2.1.1. Programación en Python de un robot basado en Arduino

El objetivo técnico será proporcionar una biblioteca en python que poder utilizar para programar un robot concreto basado en una placa base de Arduino, con el fin de dar una opción de lenguaje más sencilla. Para que esto sea posible, también se trabajará en un programa residente, en Arduino, que grabar en la placa base, que ofrezca una comunicación con la biblioteca de python. Así, se podrán programar los sensores y actuadores del robot con funciones en python, cuya lógica estará en esta biblioteca y de la cual no tendrán que preocuparse los alumnos.

#### 2.1.2. Propuesta educativa completa, basada en Scratch y Python y escalonada según dificultad

Ofreceremos una propuesta educativa, para un curso escolar, orientada según niveles de dificultad y con objetivos docentes. Para ellos, crearemos diferentes ejercicios, o prácticas, de robótica -con el robot educacional Mbot, describiendo los objetivos conceptuales que se persiguen y ordenándolas con la finalidad de un aprendizaje gradual de programación. Este curso estará orientado principalmente a alumnos de Educación Primaria o Secundaria (alumnos sin conocimientos previos de programación). Para esta propuesta, utilizaremos tanto el lenguaje de programación por bloques Scratch, proporcionado por el fabricante, como nuestro *middleware* en Python, más complejo y, por tanto, como segunda parte avanzada.

## 2.2. Requisitos

Respecto a estos objetivos, hay ciertos requisitos que se han marcado para cumplirlos:

- La plataforma desarrollada, llamada PyBo-Kids-2.0, deberá ejecutarse en el lenguaje de programación Python, y el robot educativo podrá programarse en este lenguaje.
- Esta plataforma contendrá los métodos necesarios, en una biblioteca, para empaquetar la lógica de los sensores y actuadores del robot, haciendo esta lógica invisible al usuario (alumno).
- PyBo-Kids tendrá asociada una biblioteca de Arduino, desarrollada para entenderse con la biblioteca de Python y establecer una comunicación exitosa entre el robot (lado residente) y el PC.
- Esta biblioteca Arduino estará pre-cargada en el robot, para no tener que cambiar el programa del lado residente cada vez que se quiera programar algo nuevo en el Mbot.
- Los ejercicios y prácticas educativas se diseñarán para utilizar el robot Mbot y sus periféricos, así como la plataforma del fabricante y la desarrollada en este Trabajo Fin de Grado (PyBo-Kids-2.0).
- Los contenidos educativos tendrán serán una guía de programación y de robótica, tendrán unos objetivos a corto plazo (por ejercicio), a medio plazo (por bloque de lenguaje) y a largo plazo (por curso). Sin embargo, por edad de los alumnos, podría darse el caso de no poder completar el segundo bloque; en este caso se orientará el curso para avanzar todo lo posible.

## 2.3. Metodología

Este Trabajo Fin de Grado tiene como premisa un trabajo ya realizado, durante un curso escolar, de clases de robótica a alumnos de Educación Primaria. Con esta base, y este conocimiento adquirido, se crearon los objetivos y sus requisitos, para ampliar la propuesta educativa. Para cumplir estos objetivos, se ha trabajado manteniendo un ciclo semanal de reuniones telemáticas con el tutor, donde se comentaban: avances realizados con respecto a los hitos marcados la semana anterior, problemas encontrados, ideas de trabajo, y nuevos hitos para trabajar durante la semana. La progresión necesaria para el cumplimiento ha sido la siguiente:

1. Familiarización con el entorno robótico de Arduino: uso del robot Mbot con el lenguaje nativo de la placa base, para entender la comunicación *Serial* y el funcionamiento de los actuadores y sensores (valores de entrada y salida), además de familiarización con el lenguaje en sí.
2. Comunicación "robótica" básica entre Python y Arduino, a la que ir añadiendo los periféricos del robot.
3. Diseño de un sistema de mensajes estandarizados para el desarrollo de ambas bibliotecas Python y Arduino, con el que poder establecer una comunicación exitosa entre PC y robot.



4. Desarrollo de ambas bibliotecas con el diseño anterior, y de ejercicios utilizando éstas.
5. Adaptación de los ejercicios realizados durante el curso escolar a la nueva plataforma PyBo-Kids.

# Capítulo 3

## Infraestructura

En este capítulo describiremos la infraestructura utilizada, tanto software como hardware, detallando los pasos a seguir si se desea emular. En caso de que algún componente haya sido elegido entre otros de igual aplicación, expondremos las razones de la elección.

### 3.1. Entorno

Teniendo en cuenta el carácter educativo de este Trabajo, y su pretendida aplicación en estudiantes de Educación Primaria, se ha elegido el sistema operativo Windows, un entorno conocido, amigable y fácilmente accesible, para instalar y utilizar las diferentes herramientas software.

Tradicionalmente, cuando se hablaba de programación y especialmente de programación robótica, siempre se ha utilizado el sistema operativo Linux. Éste daba la posibilidad de instalar todos los paquetes de lenguajes de programación y entornos, mientras que Windows era especialmente cerrado en cuanto a lenguajes no nativos (fuera de *bash* o de *visual basic*, se hacía complicado utilizar un lenguaje de programación sin acabar recurriendo a virtualizar una máquina Linux), y los entornos software (aplicaciones) de terceros diseñados para programar habitualmente no tenían una versión instalable para Windows.

Sin embargo, los últimos años el Sistema Operativo se ha abierto a esta operativa, ya que la política popular demandaba poder utilizarlo, siendo el sistema operativo más utilizado por usuarios, como herramienta de desarrollo también a nivel usuario. La nueva línea de comandos de Windows, **Powershell** (también lenguaje de scripting), añadió a la original *Bash* características nativas de Linux, siendo una herramienta de programación además de administración. De hecho, en la última versión, está disponible para el propio SO de Linux (en algunas distribuciones).

Al principio de este Trabajo, se consideró a Linux un entorno menos amigable para usuarios sin experiencia en programación, y de la edad comentada anteriormente, por ser considerado "no de usuario": en el poco probable caso de que un estudiante conociera el sistema, lo consideraba algo muy complejo y no accesible para su nivel. Por tanto, elegir Windows eliminaba ese prejuicio y contaba con la ventaja de predisponer positivamente al alumno y de facilitarle el acceso al entorno.

## 3.2. Hardware

En esta sección describiremos los componentes hardware utilizados. La razón de la elección de éstos responde a la misma filosofía que en la sección anterior, la facilidad de acceso a los componentes y la facilidad con que se complementa en el entorno.

### 3.2.1. Placa Arduino

Las placas Arduino son las más extendidas en cuanto a robótica. Arduino nació como una solución barata con el principal objetivo de utilizarlo en Educación. Además, al ser un proyecto liberado al público, su uso está extendido a toda una comunidad, que amplía y comparte sus propios desarrollos.

Estas placas son hardware libre (uno de las principales razones de su bajo coste), y contienen un procesador re-programable y una serie de pines hembra, donde se conectarán los periféricos de entrada/salida necesarios para controlar un robot. Hay diferentes modelos de placas Arduinos, cada una fabricada con un propósito diferente; en este caso, hemos utilizado el modelo mCore, basado en [Arduino Uno](#), ya que es la que lleva por defecto el robot educativo Mbot (que comentaremos en la sección 3.2.2). Además de los pines hembra, o puertos, contiene una serie de actuadores y sensores integrados en la placa.

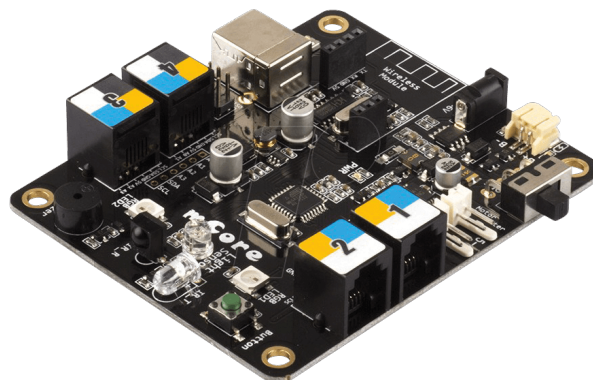


Figura 3.1: Placa mCore

### 3.2.2. MBot

Actualmente hay una gran variedad de robots educativos, orientados principalmente a los alumnos más jóvenes. El objetivo de la robótica educativa es ofrecer un entorno amigable y divertido, además de *más realista* que la programación tradicional. Es más fácil para un niño o niña sentirse interesado por algo que tiene una reacción *visible*, en algo que puede tocar y manejar, que en la programación normal que, aunque produce una reacción, es mucho más abstracta. Esto convierte la robótica en la herramienta perfecta para enseñar lenguajes, conceptos básicos de programación como funciones, uso de variables, y avanzar hasta la algoritmia se hace un proceso natural al que los propios alumnos llegan ellos mismos.

El principal problema al que nos enfrentamos para enseñar programación a los más jóvenes son los lenguajes de programación. Son poco intuitivos y legibles, sumado a la dificultad de mantener la atención de un alumno o alumna tan joven escribiendo en un ordenador en estos lenguajes. Era necesaria, por tanto, una forma de enseñar estos conceptos de programación

comentados anteriormente sin tener que utilizar lenguajes clásicos de programación, al menos para los alumnos más jóvenes. La solución fue la programación por bloques de *pseudocódigo*. El pseudocódigo es un *framework*, una carcasa, que recubre el verdadero lenguaje de programación de la placa del robot, y lo hace legible y entendible. Además, siendo esto una de las grandes ventajas añadidas, la mayoría de los pseudocódigos están en casi todos los idiomas, permitiendo a los alumnos programar en su propio idioma.

Esto nos lleva al robot Mbot. Está basado en una placa Arduino Uno, y pensado especialmente para la enseñanza: los diferentes sensores y actuadores (los que vienen en el paquete básico, aunque hay muchos más posibles del mismo fabricante) están pensados para ofrecer prácticas entretenidas y, lo más importante, con diferentes niveles de dificultad, por lo que se puede utilizar con alumnos de diferentes edades y, durante un mismo curso escolar, crear prácticas con las que evolucionar en los conceptos.



Figura 3.2: Modelo Mbot utilizado

Este robot Mbot se programa utilizando un lenguaje llamado *Scratch* (3.3.1), un pseudo código muy gráfico y sencillo pero potente, que "recubre" la placa base. Sin embargo, al ser una placa Arduino y por lo tanto, reprogramable, siempre se puede programar en el propio lenguaje Arduino. Esto se comentará en profundidad más adelante, en el punto 4.

Como se puede observar en la imagen, el robot cuenta con dos motores conectados a la placa; contiene además cuatro puertos a los que poder conectar diferentes periféricos con los que trabajar (numerados del uno al cuatro), sin contar con los motores. El modelo mostrado es el básico, sin embargo se pueden cambiar los componentes, añadiendo y/o cambiando la estructura base, y crear "otro" robot (por ejemplo, en las figuras siguientes ?? ). Aunque en este Trabajo de Fin de Grado solo trabajaremos con la versión básica del Mbot, esta posibilidad de añadirle componentes o cambiarlos es muy interesante para los alumnos, ya que trabajan la mecánica y pueden utilizar más tipos de periféricos.



Figura 3.3: Posibles cambios en el Mbot

### 3.2.2.1. Sensores

En robótica un *sensor* es un periférico de entrada que, conectado a una placa base, recoge información del medio (cantidad de ruido, temperatura, distancia frontal, etc) y la envía a la placa, dejándola disponible para toma de decisiones. La cantidad de información que se pueda recibir sólo depende de la cantidad de sensores que se pueda conectar a la vez a la placa (cuatro, en este caso, si solo se trabajara con sensores y ningún actuador). En el robot básico (de la imagen ??) los sensores son:

**Sensor de ultrasonidos** Está colocado en el frente, y recoge información de **distancia** hasta un objeto, en *cm* (el valor máximo es 400)

**Sensor infrarrojo Sigue Líneas** Está ubicado de tal forma que lea, del "suelo", si el sensor está tapado o no, es decir: está sobre blanco o negro (de fondo, es un sensor binario). Se compone en realidad de dos sensores, izquierdo y derecho, por lo que habrá cuatro posibilidades, cada una codificada con un valor numérico (el valor que devuelve el sensor a la placa Arduino):

- Ningún sensor tapado: 0
- Sensor izquierdo tapado y derecho no: 1
- Sensor derecho tapado e izquierdo no: 2
- Los dos sensores tapados: 3

**Sensor de Luz** En este caso, está integrado en la placa, por lo que no es necesario utilizar un puerto para él. Nos da un valor de cantidad de luz en el ambiente, pudiéndolo utilizar para saber si hay más o menos luminosidad de la deseada. Por supuesto, para este valor "deseado" será necesario obtener un valor inicial de la habitación en la que nos encontremos, para poder establecer ese valor barrera (*threshold*)



(a) Ultrasonidos: sensor de distancia



(b) Infrarrojos: sensor siguelíneas



(c) sensor de luz integrado

Figura 3.4: Sensores

### 3.2.2.2. Actuadores

La definición de *actuador* es un periférico de salida al que la placa envía datos con los que éste realiza una acción de una forma u otra. Por ejemplo, teniendo una velocidad  $v_0$ , este valor es enviado a los motores, que se moverán a esa velocidad y no a otra.

Los actuadores, los que no están integrados en la placa directamente, se conectan a la placa a través de los mismos puertos que los sensores. Para poder usarlos, habrá que especificarle a la placa en qué puerto están conectados (igual que con los sensores). En nuestro robot básico, tenemos los siguientes actuadores:

**Leds** . Están integrados en la placa, en la parte superior, y compuesto por dos led individuales que poder combinar dependiendo de qué valor codificado se envíe a la placa:

- Los dos led: 0
- Sólo led derecho: 1
- Sólo led izquierdo: 2

Los led son RGB (*[red, green, blue]*), y el color de los dos led se codifica con un valor entre 0 y 255 para cada uno de los colores rojo, verde y azul. Así, por ejemplo, el rojo completo sería [0,255,0], el morado sería [255,0,255], el negro [0,0,0] o el blanco [255,255,255]. Estos led están codificados con valores decimales, en vez de hexadecimales como sería una codificación RGB tradicional, para facilitar la programación a los alumnos, que no conocerían el sistema hexadecimal.

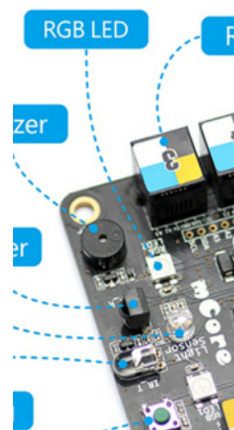
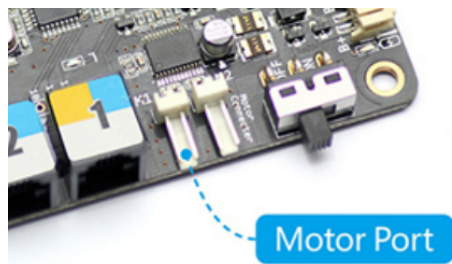


Figura 3.5: Led RGB integrados

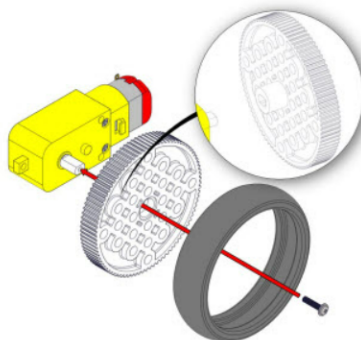
**Motores** En el paquete básico tenemos dos motores DC (de corriente continua) conectados a la placa en un conector específico para motores a los que conectar los dos cables, positivo y negativo. Los motores admiten como velocidad de entrada valores enteros entre  $[-255,255]$  (valores negativos para retroceder y positivos para avanzar), pudiendo dar valores diferentes a cada uno de ellos (para tener capacidad de hacer girar al robot).



(a) Puerto de conexión de los motores



(b) Motor DC



(c) Motor DC: montaje con rueda

Figura 3.6: Motores

**Zumbador** También integrado en la placa, emite notas musicales codificadas en nomenclatura

americana. La equivalencia, que los alumnos necesitan conocer, es la siguiente:

| Europea | Americana |
|---------|-----------|
| Do      | C         |
| RE      | D         |
| MI      | E         |
| FA      | F         |
| SOL     | G         |
| LA      | A         |
| SI      | B         |

Para utilizar diferentes notas más agudas o más graves, se utilizan números a continuación de las letras: **C0** es más grave que **C5**. Internamente, el zumbador entiende valores enteros, correspondientes en frecuencia con cada nota.



Figura 3.7: Zumbador integrado

### 3.3. Software

En esta sección describiremos el diferente software utilizado y el propósito de éste en el marco de este Trabajo Fin de Grado.

#### 3.3.1. Scratch y mBlock

Como se ha comentado anteriormente, el robot Mbot es programable con Scratch, un lenguaje de **programación por bloques**. Un *bloque de código* consiste en codificar en un "paquete" una sentencia completa de lenguaje (del lenguaje correspondiente, Arduino en este caso). Así, el estudiante que utilice Scratch, será capaz de utilizar una sentencia *if..else* o un bucle *for* de forma muy fácil y entendible, sin necesitar aprenderse todas las reglas de sintaxis del lenguaje real. A continuación, se muestran algunos ejemplos de bloques en Scratch correspondientes a los conceptos de programación más utilizados:



Figura 3.8: Algunos ejemplos de bloques en Scratch



Como puede observarse, las sentencias que se quieran repetir, o las variables de condición, tienen un lugar muy intuitivo donde colocarse. Además, pueden crearse variables, en las que guardar los valores de los sensores y poder utilizar como entrada de actuadores, o como valores de referencia.

Ciertamente, y aunque el lenguaje Scratch puede utilizarse como lenguaje de programación "tradicional", utilizando un escenario virtual con un personaje para observar el resultado del programa (sin robot físico), es mucho más completo al añadirle el módulo del robot deseado. Este módulo contiene bloques para recoger valores de los sensores o enviar valores a los actuadores, ya sean "on board" (integrados en la placa), o teniendo que especificar el puerto al que están conectados:

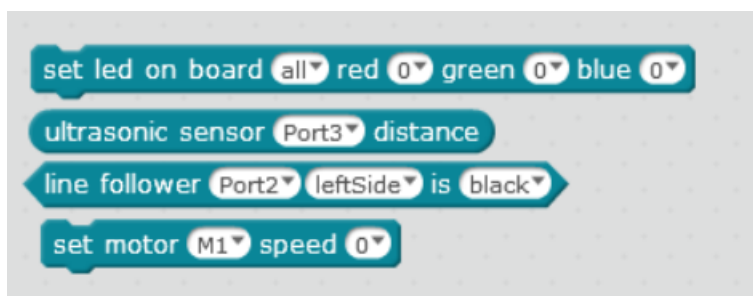


Figura 3.9: Ejemplos de bloques de Mbot en Scratch

Para utilizar Scratch y programar el robot, es necesario instalar un programa, que es el que contiene el compilador y el que permite conectar el robot al PC para enviarle el programa, llamado *mBlock*. El ejecutable se descarga de la página oficial del fabricante, [Makeblock](https://makeblock.com/), disponible para PC (también existe una versión simplificada para dispositivos móviles).

Una vez instalado el software, el proceso para poder empezar a usar el mBot es muy simple:

1. Conectar el robot al pc con el cable USB, y conectarlo con el programa (el robot debe estar encendido).

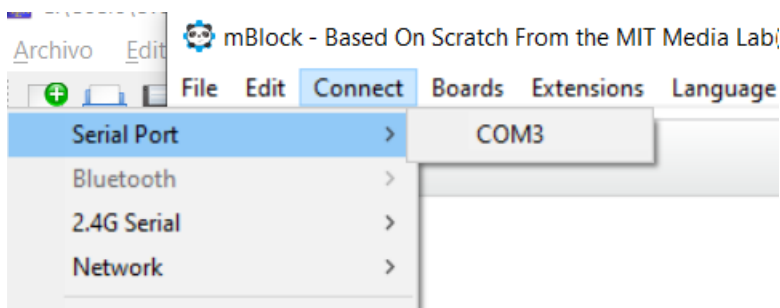


Figura 3.10: Conectar el mBot

2. Asegurarse de que la placa corresponde con el robot que tenemos

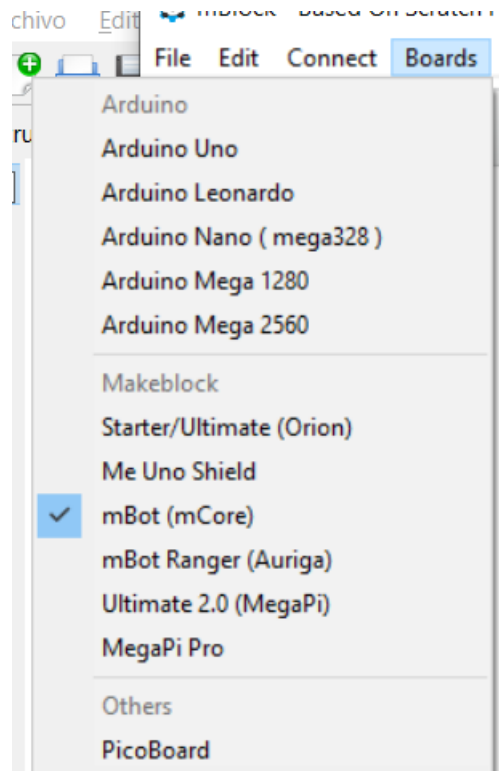


Figura 3.11: Placa mBot

3. Actualizar el programa Arduino subido a la placa base, para que funcione con Scratch (en caso de haber utilizado el robot con otro programa)

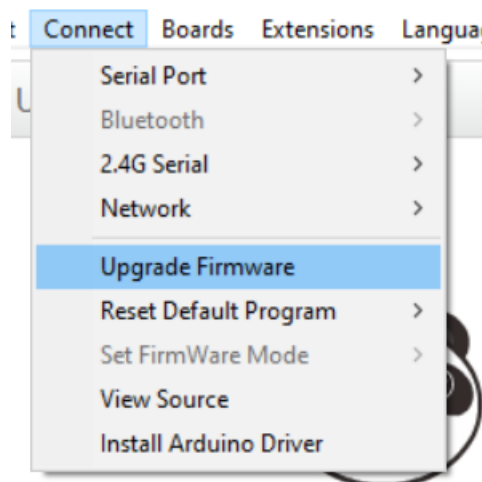


Figura 3.12: Actualizar firmware de la placa

4. Una vez programado algo de código, para subirlo al robot: click derecho sobre el código, 'upload to arduino', para compilar el programa, y otra vez a 'upload to arduino' cuando el compilador aparezca en la parte derecha de la pantalla, para subirlo a la placa

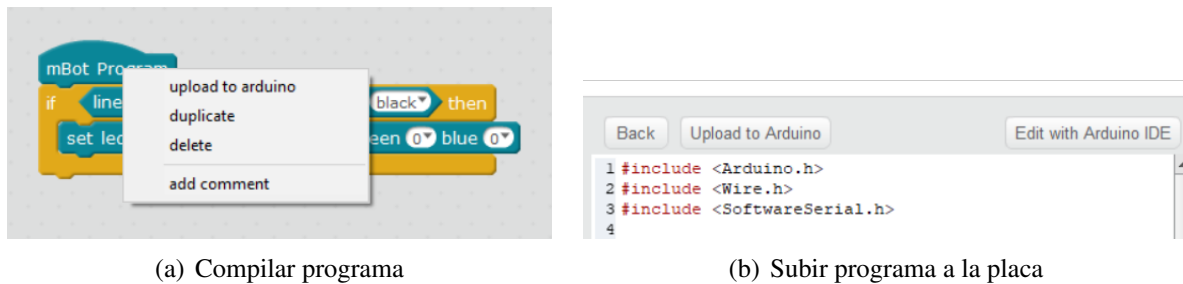


Figura 3.13: Subir programa a la placa del mBot

5. Una vez subido el programa, el robot comenzará a funcionar tal y como lo hayamos programado.

Este lenguaje, junto a la interfaz gráfica, es una perfecta primera aproximación a la programación y a la robótica, dando más importancia al aprendizaje conceptual que a la sintaxis y al lenguaje. Obviamente, los alumnos necesitarán que se les guíe, principalmente al comienzo del curso, en comprender las necesidades de incluir los diferentes bloques y las ventajas que produce en el código. El objetivo no será explicarles para que usar, por ejemplo, un bloque condicional, sino proponerles un ejercicio en el que, para llegar a la solución, necesitarán de forma intuitiva programar un condicional, y lleguen naturalmente a la necesidad de ello.

### 3.3.2. Arduino

Como se explicó cuando hablamos de la placa base en la sección 3.2.1, ésta se programa en **Arduino**. Este lenguaje de programación está basado en C++, y pensado para interactuar con objetos electrónicos.

Teniendo en cuenta el objetivo de este Trabajo Fin de Grado, ofrecer una posibilidad de programar el robot Mbot en Python, es necesario utilizar el lenguaje nativo de la placa base, con el fin de que el robot *entienda* las órdenes programadas en Python. Por lo tanto, para crear este "framework", es necesario instalar el entorno Arduino. A continuación se explican las instrucciones para instalar el entorno en Windows, y para configurarlo para el robot.

1. Descargar e instalar, Arduino IDE, el cual instala el entorno completo de Arduino. Para las versiones 8 y 10 de Windows, está disponible directamente en el Microsoft Store. Para otras versiones de Windows, está disponible en la [página web oficial](#).



Figura 3.14: Arduino IDE en el Microsoft Store

2. Descargar las librerías de [Makeblock](#) para añadirlas a Arduino y poder utilizarlas en nuestro entorno.

3. Incluir el archivo comprimido descargado desde el Arduino IDE:  
*Programa - Incluir librería - Añadir biblioteca .ZIP - Seleccionar el fichero - Abrir*
4. Seleccionar la placa básica que se va a conectar al IDE:  
*Herramientas - Placa - Seleccionar Arduino Uno*
5. Especificar el modelo exacto de placa al principio del programa de Arduino; en este caso la placa que estábamos utilizando es la mCore. Así se cargan todos los métodos correspondientes al modelo de placa; debe incluirse en cada programa de Arduino que se escriba.

```
#include "MeMCore.h"
```

6. Igual que con Scratch, primero se debe conectar el robot al software (con el robot enchufado al PC y encendido):  
*Herramientas - Puerto - Seleccionar el puerto en el que está el robot*  
Los puertos USB en Windows son *COM1*, *COM2*, *COM3*, etc.
7. Una vez tengamos un programa en Arduino listo para subir a la placa y probar con el robot:  
*Programa - Subir* o el botón rápido de "subir".  
Este proceso primero compilar el programa y, si está correcto, lo sube a la placa. Sin embargo, se puede compilar primero como comprobación (*Programa - Verificar*).

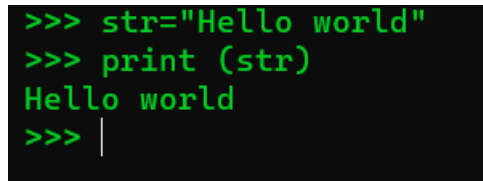
### 3.3.3. Python

Por último, utilizaremos **python** para programar la lógica de los ejercicios. El objetivo es que se programe en python sólo los ejercicios, habiendo paquetizado todo lo relativo al robot en Arduino, y sólo utilizando la conexión con éste para obtener datos de los sensores y enviar datos a los actuadores. La descripción de este proceso y de la lógica de ambos lados (PC y robot) se describirá en profundidad en el capítulo 4.

La razón de utilizar Python como lenguaje de programación es la misma que la del resto de componentes: la facilidad para los alumnos de acceder a ello. Es uno de los lenguajes con sintaxis más simple (por comparación, Arduino / C++ es muy cerrado y complejo), además de ser las palabras reservadas significativamente entendibles (aunque es cierto que en inglés: *while*, *print*, *read*, etc), así como la declaración de variables es más flexible. Además, y aparte de la cuestión educacional, el módulo *serial* para conexión con periféricos, está con la instalación base, y es mucho más sencillo que en otros lenguajes, haciéndolo perfecto para la electrónica.

La instalación de python en Windows también es muy sencilla:

1. Descargar el paquete de la [web oficial](#) y ejecutar el instalable descargado. También es posible, para Windows 10, obtenerlo desde el Microsoft Store, tal y como se hizo para Arduino.
2. Una vez descargado, comprobar que se puede ejecutar abriendo una consola -Powershell- y escribiendo *python*, se ejecutará la consola de python, y podremos probar que tenemos python instalado en nuestro entorno (*exit()* para salir). También es útil comprobar que hemos instalado la versión correcta (3.10): *python --version* en una consola de Powershell (no teniendo abierta la consola de python)

A screenshot of a Python shell with a black background and green text. The text shows three lines of code: the first line assigns the string "Hello world" to a variable named 'str', the second line prints the value of 'str', and the third line shows the output "Hello world". The prompt '>>>' is visible at the start of each line, and a vertical cursor is at the end of the third line.

```
>>> str="Hello world"
>>> print (str)
Hello world
>>> |
```

Figura 3.15: Comprobar python

3. Para ejecutar un programa escrito en python desde la consola de Windows:

```
py HelloWorld.py
```

Para escribir código en python, se puede utilizar cualquier editor de texto. El mismo instalador de python instala uno propio, *IDLE shell*, suficientemente simple y preparado particularmente para su sintaxis.

# Capítulo 4

## Plataforma PyBo-Kids

En este capítulo explicaremos el proceso seguido para desarrollar las bibliotecas Arduino y Python, explicando el código y las distintas necesidades que han surgido durante el desarrollo. Como se explicó en el Capítulo 3, se ha utilizado Arduino y su IDE nativo para esta programación, y Python 3 y un editor de texto estándar (en este caso, Visual Studio Code, de Microsoft) para la programación en Python.

### 4.1. Programa residente

El primer paso necesario en

### 4.2. Programa PC

### 4.3. Diseño

### 4.4. Resultado: programa PC y programa residente

# Capítulo 5

## Aplicación educativa

### 5.1. Contexto

### 5.2. Primera etapa: Scratch

#### 5.2.1. Objetivos docentes: metodología

#### 5.2.2. Prácticas

### 5.3. Segunda etapa: PyBo-Kids en Python

#### 5.3.1. Objetivos docentes: metodología

#### 5.3.2. Introducción a Python

#### 5.3.3. Prácticas

Esto es un ejemplo. Ver en [1] [2]

# Capítulo 6

## Conclusiones

Esto es un ejemplo. Ver en [1] [2]



# Bibliografía

- [1] DARWIN, C. *El origen de las especies por medio de la selección natural*. Editorial CSIC-CSIC Press, 2009.
- [2] DEL VALLE-INCLÁN, R. *Luces de bohemia*. Dirección General de Música y Teatro, 1984.