

# Proyecto 1:

## Construye y programa tu casa domótica



### ***Descripción del proyecto:***

*Con esta guía aprenderás a construir y programar una puerta automática y una lámpara inteligente.*

***Nivel de dificultad: Fácil***

***Tiempo estimado: 4 horas***

### **Materiales:**

- Kit de robótica
- Bitbloq robotics
- Caja de cartón
- Cuerda
- Cinta adhesiva
- Tijeras
- Pegamento

### **Opcionales:**

- Impresora 3D
- BlocksCAD / FreeCAD / Tinkercad / Bitbloq 3D

## Construye tu casa domótica

En este proyecto los alumnos tienen que construir una casa que tenga una puerta automática y una lámpara inteligente.

Se puede construir tanto el interior como el exterior de la casa. Es importante que dejemos esta parte a elección de los alumnos, fomentando su creatividad e implicación en el proyecto.

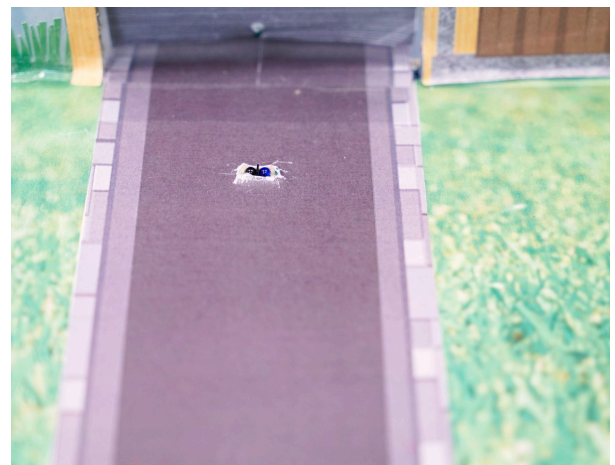
A continuación, se muestra un ejemplo del exterior de una casa, en la que se ha creado una puerta de garaje automática y una farola inteligente que ilumina el jardín.



Los elementos de la casa (jardín, carretera, casa...) se pueden diseñar con un editor de imágenes, buscar en Internet imágenes *Creative commons* (sin derechos de autor) para imprimir o pintar sobre el cartón o material utilizado.

Para decorar, se ha diseñado en 3D un coche y una farola, sobre ésta última se incorporarán los componentes necesarios para crear una luz inteligente.

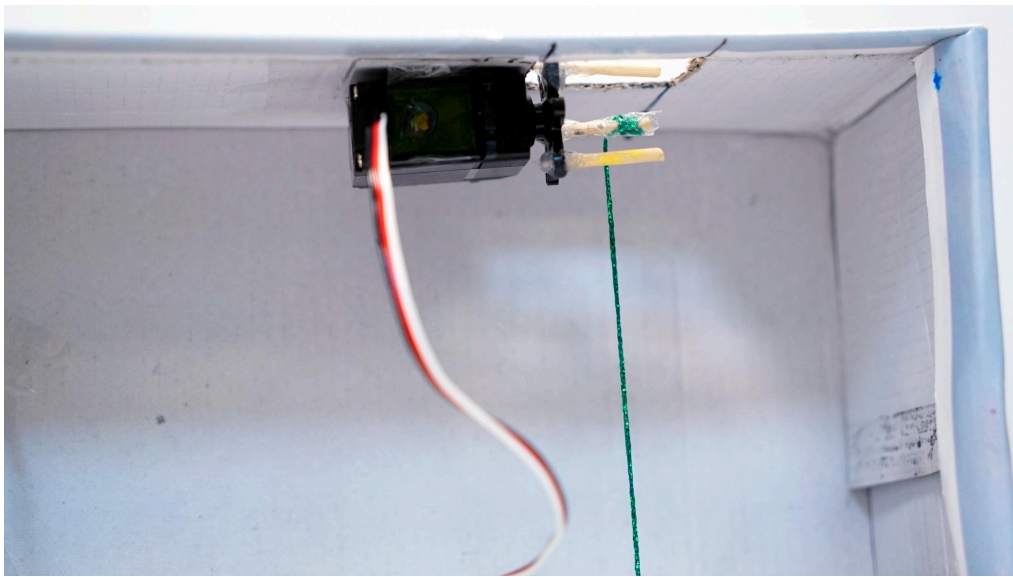
Para automatizar la apertura de la puerta del garaje, se ha colocado un sensor IR debajo de la carretera con el fin de que éste detecte la presencia de un coche.



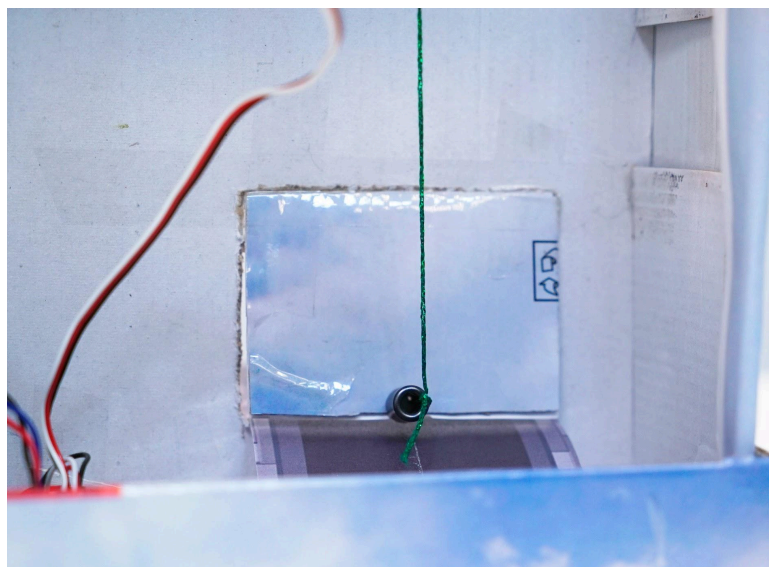
Para la apertura de la puerta, primero deberemos recortarla y luego utilizar alguno de los componentes del kit de robótica para que tire de ella. Podemos utilizar dos componentes:

**1) Un servo de rotación continua**, atando una cuerda a la puerta con el fin de que se vaya enrollando para abrirla.

Pegaremos el servo de rotación continua en la parte superior de la caja y para que se pueda enrollar la cuerda, le pondremos un cabezal al que pegaremos tres palos (sobre los que se irá enrollando la cuerda). Atamos la cuerda a uno de los palos, tal y como se muestra en la siguiente imagen:

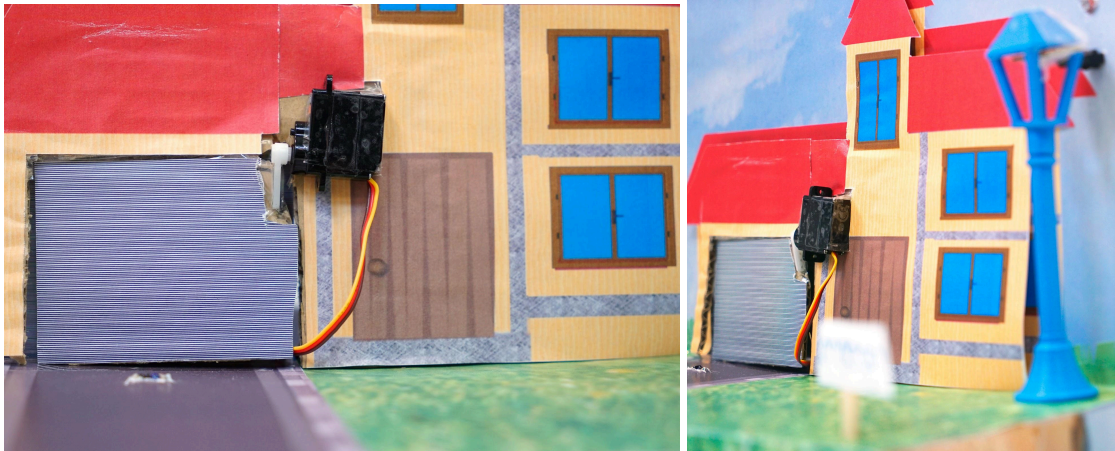


Por último, añadiremos un peso a la parte de detrás de la puerta para que ésta se cierre completamente.





2) Un **miniservo**, pegando el cabezal a la puerta tal y como se muestra en la siguiente imagen:



Para la lámpara inteligente, se ha buscado en *Thingiverse* el diseño de una farola (<http://www.thingiverse.com/thing:1361590>\*) y se ha impreso en 3D, pero podemos pedir a los alumnos que diseñen su propia lámpara o farola en 3D para imprimirla posteriormente.

A dicha farola se le ha incorporado un LED y un sensor de luz (LDR).



En la caja, encima de la farola, se ha hecho un pequeño agujero y se ha introducido la fotorresistencia del LDR, con el fin de que pueda comprobar la luz que hay.

También se ha hecho un agujero en la caja debajo de la farola para que pasen los cables.



## Experimentando con los componentes del kit

Antes de comenzar a programar la casa domótica, proponemos realizar una breve introducción y realizar unos sencillos ejercicios con el fin de que los alumnos aprendan a programar los componentes del kit de robótica que se van a utilizar.

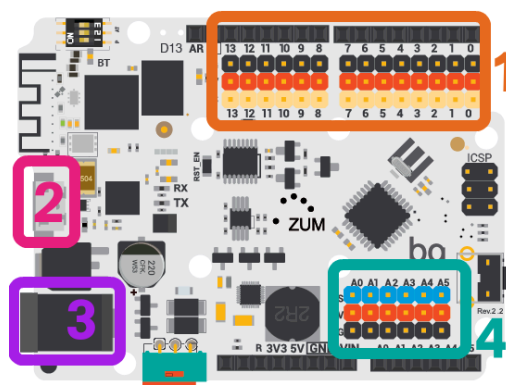
Comenzaremos explicando qué es la placa controladora. Ésta es el dispositivo que guarda toda la información que recibe a través de los sensores y la que controla cómo actuar. Podemos pedirles a los alumnos que busquen la placa en el kit y aprovechar para explicar sus partes:

**1. Pines Digitales:** En los que conectaremos los componentes digitales (aquellos que solo tienen dos valores: 0 y 1).

**2. Conexión USB:** Que utilizaremos para conectar con el ordenador.

**3. Entrada de alimentación:** En la que conectaremos las pilas o baterías.

**4. Pines Analógicos:** En los que conectaremos los componentes analógicos (aquellos que pueden tener valores de entre 0 y 1023).



A continuación, explicaremos que existen diversos tipos de componentes, que podemos dividir en sensores y actuadores:

- Un **sensor** es un componente que nos proporciona información sobre algo que ocurre alrededor, como por ejemplo el nivel de luz, la temperatura, un obstáculo cercano, etc. Podríamos decir que son equivalentes a nuestros órganos de los sentidos (vista, tacto, oído, etc.).
- Un **actuador** es un componente que puede realizar alguna acción, como por ejemplo emitir una luz, un sonido o moverse. Podríamos decir que son equivalentes al habla, las extremidades, etc.

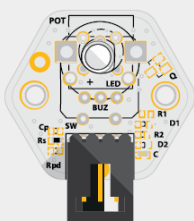
También podemos clasificar los componentes en cuanto a si utilizan señales digitales o analógicas:

- Un componente **digital** es aquel que solo puede diferenciar entre dos estados (0/1, verdadero/falso, encendido/apagado). Por ejemplo, un botón está pulsado o no lo está, un LED está encendido o apagado, etc.
- Un componente **analógico** es aquel que puede tener más de dos valores. Por ejemplo, la temperatura, la cantidad de luz, etc. Un sensor analógico, generalmente puede leer valores de entre 0 y 1023.

Una vez explicado todo lo anterior, procederemos a realizar ejercicios sencillos para experimentar con los componentes que necesitarán para su casa domótica.

Comenzaremos programando el **LED**:

Pediremos a los alumnos que busquen en el kit un LED. Les preguntaremos qué creen que es, un componente digital o analógico, y si es un sensor o actuador.

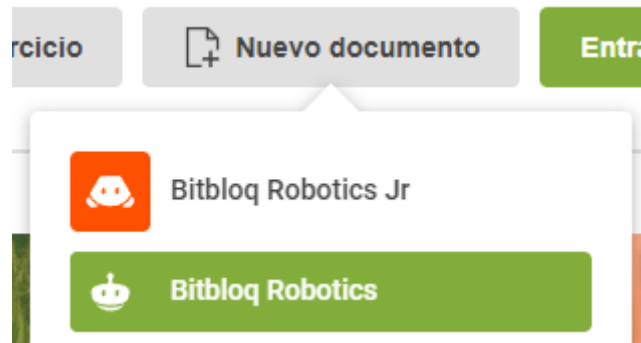


Un **LED (Light-Emitting Diode)** es un componente actuador capaz de emitir luz. Los LED son baratos, gastan muy poca energía y pueden llegar a ser muy luminosos. Solo tiene dos valores: encendido o apagado, por lo que es digital. Debemos conectarlo a uno de los pines digitales de nuestra placa.

Deberemos indicarles que se fijen en los cables de conexión del LED. Observarán tres cables: uno blanco (señal), uno rojo (voltaje) y uno negro (tierra). Para conectarlos tienen que coincidir los colores de los cables con los de los pines de la placa.



A continuación, les enseñaremos a programar que se encienda un LED. Para ello, les pediremos que conecten su placa al ordenador utilizando el cable USB, abran Bitblog ([bitblog.cc](http://bitblog.cc)) y hagan clic en *Nuevo - documento Bitbloq Robotics*.



La interfaz se divide en cuatro pestañas principales situadas a la izquierda:

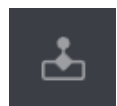


**Hardware:** donde elegiremos qué placa vamos a usar y qué componentes utilizaremos.



**Bloques:** donde programaremos nuestro proyecto por bloques. Este área está dividida en tres partes:

- Variables globales y funciones: Permite declarar variables globales y crear funciones.
- Instrucciones iniciales (*Setup*): Permite configurar acciones que se ejecutan sólo una vez al principio.
- Bucle principal (*Loop*): Todo lo que se encuentre dentro se ejecutará en bucle de forma indefinida.



**Diagramas:** donde programaremos nuestro proyecto por diagramas de flujo. Lo que programemos aquí se verá reflejado en *Bloques*, y viceversa.




**Información del proyecto:** donde añadiremos la información sobre el proyecto.

Les indicaremos que en la pestaña *Hardware* seleccionen la placa que vamos a utilizar (**Zum Core o Zum Core 2.0**) arrastrándola al centro de la pantalla. A continuación, deberán conectar el LED al mismo pin al que lo han conectado físicamente. Debemos advertir a los alumnos de que eviten conectar los componentes en el pin 0 y 1, debido a que esos pines son los que utiliza la placa para comunicarse con el ordenador y pueden dar problemas.

Una vez que tienen las conexiones realizadas, deberán ir a la parte de *Bloques* y programar en el apartado *Bucle principal (Loop)* de la siguiente manera:

#### — Bucle principal (Loop)



A continuación, deberán dar al botón *Cargar* , situado en la parte superior derecha, para que el programa se cargue en la placa y puedan observar que el LED se enciende.

Tras realizar este ejercicio, deberemos pedirles que intenten programar que el LED parpadee, para lo cual solo tendrán que programar que se encienda y se apague. La programación sería algo similar a la siguiente:

#### — Bucle principal (Loop)



Generalmente, los alumnos programan *Encender* y *Apagar el LED* sin utilizar el bloque *Esperar*, y suele sorprenderles que no funcione. Aprovecharemos para explicarles que el programa se ejecuta a muchísima velocidad y realmente sí está funcionando, solo que se enciende y se apaga tan deprisa que no podemos percibirlo. Para poder observar cómo se enciende y se apaga, hay que incorporar una pequeña espera, cuya duración (en milisegundos) podrán modificar cuanto quieran.

A continuación programaremos el **sensor de luz** o **LDR**:

Pediremos a los alumnos que busquen en el kit el sensor de luz y al igual que el LED, les preguntaremos qué creen que es, un componente digital o analógico, y si es un sensor o actuador.





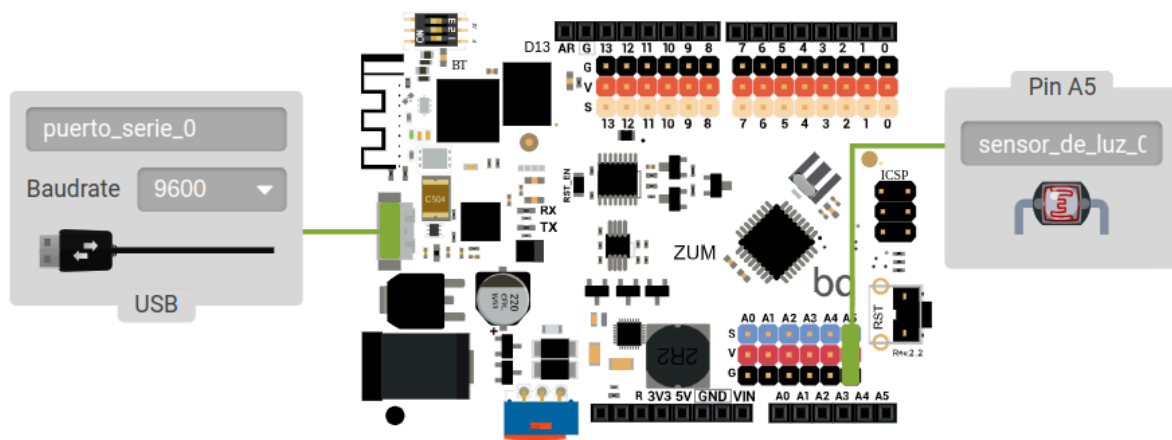
El **sensor de luz**, también conocido como LDR o fotorresistencia, es un sensor analógico que nos da una medida de la intensidad de luz. Como todos los sensores analógicos, puede dar valores comprendidos entre 0 y 1023, aunque el del kit devuelve valores de entre 0 (máxima oscuridad) y 800 (máxima iluminación).

Para comprobar cómo funciona este sensor, haremos con los alumnos un programa que permita ver por el puerto serie del ordenador los valores que recibe.



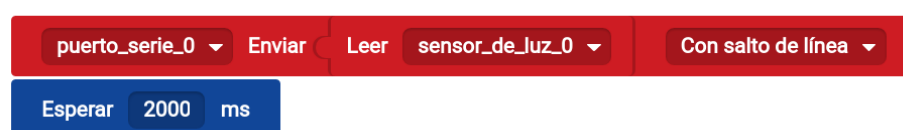
El **puerto serie** es el medio que tiene la placa controladora para comunicarse con otro aparato como el ordenador o tu móvil. Es muy útil para comprobar de un vistazo el valor de un sensor o una variable, ver el estado de un programa y, en definitiva, corregir y entender mejor los posibles errores que éste tenga.

Deberán conectar el sensor de luz (física y virtualmente) y el puerto serie, correspondiente con el USB, como un componente más en el apartado de *Hardware* de Bitbloq de la siguiente forma:



A continuación, comenzaremos a programar en el apartado *Bucle principal (Loop)* que muestre por puerto serie la lectura del sensor de luz y una espera de unos segundos:

— Bucle principal (Loop)



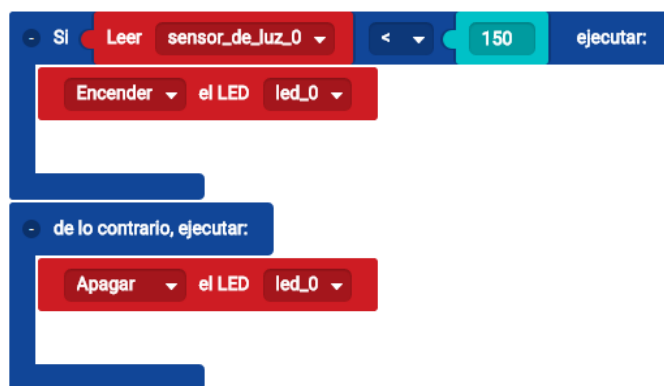
Tras esto, cargaremos el programa a la placa y pulsamos en *Ver*→ *Monitor puerto serie*, de la parte superior izquierda de la interfaz. De esta forma, aparecerán los valores de luz detectados por el sensor de luz en el aula en una ventana emergente.

Para comprobar cómo cambian los valores, podemos pedirle a los alumnos que tapen el sensor de luz y luego lo pongan cerca de un foco de luz para observar cómo los valores disminuyen o aumentan.



Por último, propondremos a los alumnos hacer la lámpara inteligente de su casa. Para ello, tendrán que programar que el LED sólo se encienda cuando el sensor de luz detecte que es de noche o haya oscuridad. Para programarlo, deberán utilizar una **sentencia condicional**, de manera que si el valor detectado por el sensor de luz es menor que X (por ejemplo 150), se encienda el LED, y si el valor es mayor, se apague. Debería quedar una programación similar a la siguiente:

— Bucle principal (Loop)



El valor de luz dependerá de la iluminación de la sala donde nos encontremos. Por ello, es importante antes de programar una luz inteligente, ver por puerto serie la cantidad de luz que hay en nuestra aula.

A continuación se introducirán los componentes que servirán para programar la puerta automática. Para ello, realizaremos con los alumnos unas actividades sencillas para que aprendan a programar el **sensor infrarrojo (IR)**, el **miniservo** y el **servo de rotación continua**.

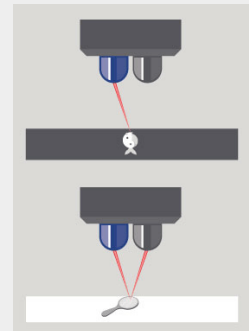
Comenzamos programando el **sensor infrarrojo** o **IR**:



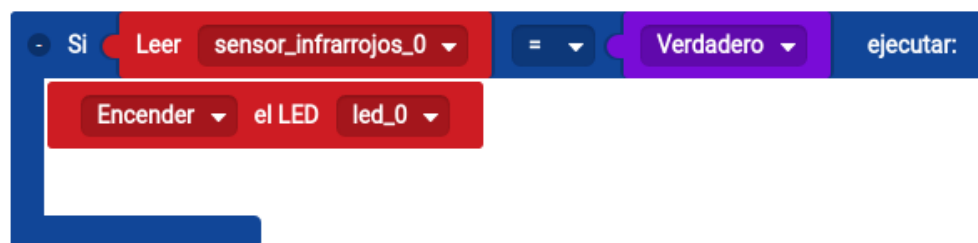
Un sensor infrarrojo o IR es un dispositivo que emite una luz infrarroja detectando la cantidad de luz reflejada. De esta forma es capaz de diferenciar entre blanco y negro.

Este componente es un sensor digital ya que devuelve 0/1 o verdadero/falso.

Funciona de forma que devuelve 1 o verdadero cuando detecta la luz reflejada en el blanco y 0 o falso cuando no la detecta, es decir, cuando el color es negro.



Para conocer cómo funciona este componente, pediremos a los alumnos que programen que cuando el sensor infrarrojo (IR) detecte que hay algo, es decir, cuando detecte 1 o verdadero, se encienda un LED.



Es posible que los alumnos, no programen qué tiene que hacer el LED cuando el sensor IR detecta 0 o falso. Para ello, tendremos que programar con el bloque *de lo contrario* que el LED se apague:

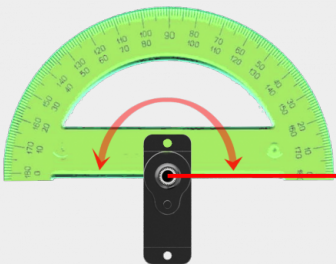


Una vez que tenemos la programación, para comprobar que funciona, tendremos que coger un papel negro y ponerlo en el IR. Cuando el papel esté encima del IR se encenderá el LED, si lo quitamos se apagará.

En la programación de la puerta automática de la casa, utilizaremos el IR para indicar que si detecta que hay un coche la puerta se abra, es decir, active el miniservo o el servo de rotación continua.

A continuación programaremos el **miniservo**:

Pediremos a los alumnos que busquen en el kit este componente y explicaremos qué es.



El **Miniservo** es un pequeño motor que es capaz de girar entre 0 y 180 grados, por lo que no puede dar vueltas completas. Es un actuador y un componente digital. Lo interesante de este tipo de motores es que nosotros decidimos a qué posición o ángulo deben moverse.

Propondremos un ejercicio para introducir el funcionamiento del miniservo, consistente en programar el movimiento del miniservo a los ángulos 0°, 45°, 90°, 135°, 180°.

Para conocer la posición del servo, le colocaremos uno de sus cabezales y lo conectaremos a un pin digital de la placa. Tras esto, para saber cuál es la posición correcta, programaremos en el *Setup* (para que sólo lo haga una vez al ejecutarse el programa) que el miniservo vaya a una posición, por ejemplo 90°. Lo que

normalmente sucede es que el cabezal del servo está mal puesto, por lo que lo quitaremos y lo pondremos mirando al centro, procurando que esté a un ángulo de 90°.

Tras esto, pediremos a los alumnos que programen que el miniservo vaya a las diferentes posiciones (0°, 45°, 90°, 135° y 180°). Es importante que debajo de cada bloque de movimiento pongan un bloque *Esperar*, ya que sino el miniservo se moverá tan deprisa que solo se verá el ángulo inicial y final.

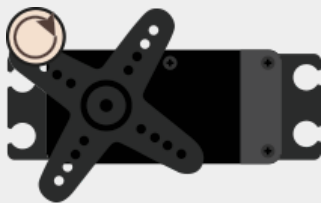
Además, añadir una espera debajo de cada bloque protege al miniservo de movimientos bruscos.



Para programar la puerta automática con el Miniservo, podemos hacer que vaya a diferentes ángulos, por ejemplo, desde 10° a 100°, para que se abra de forma progresiva.

**Nota:** para no forzar los mini servos, es recomendable no programarlos en sus posiciones extremo, es decir, ni en 0° ni en 180°, y hacerlo en su lugar en 10° o 170°.

Otra forma de hacer que la puerta se mueva es con el **Servo de rotación continua**.

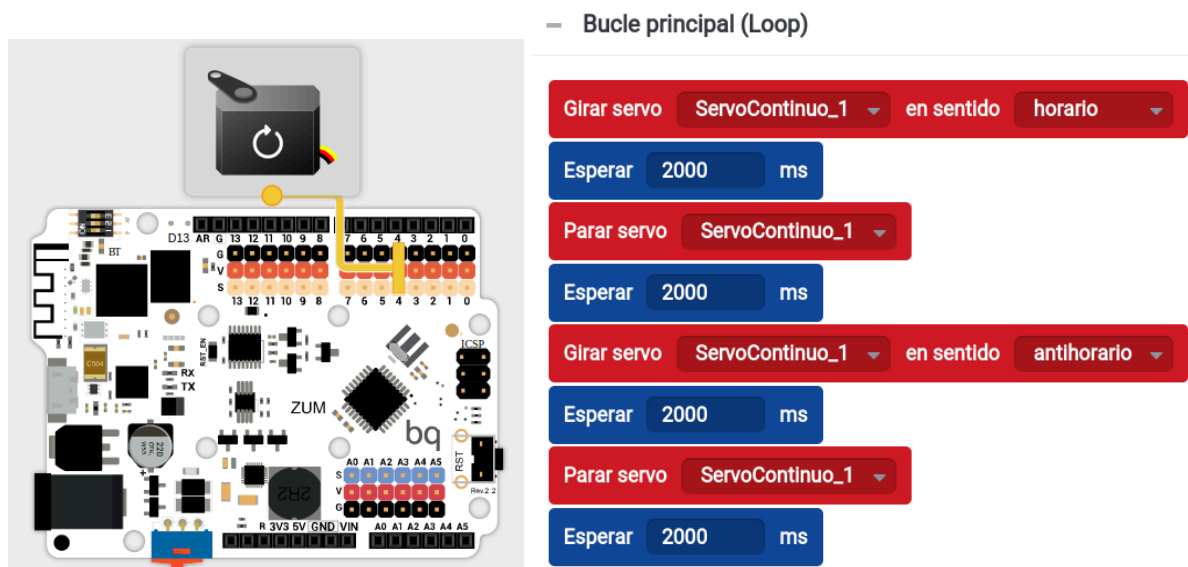


Un **servo de rotación continua** es un motor cuyo circuito electrónico nos permite controlar la dirección de giro. A diferencia del miniservo, no se detiene en una posición, sino que gira continuamente.

Para experimentar con este componente, podemos pedir a los alumnos que programen que el servo avance, se pare, retroceda y se vuelva a parar. Entre cada bloque de movimiento, deberán añadir una espera para indicar cuánto tiempo tiene que realizar esa acción o movimiento. Para ello, se realizará una programación



similar a la siguiente:



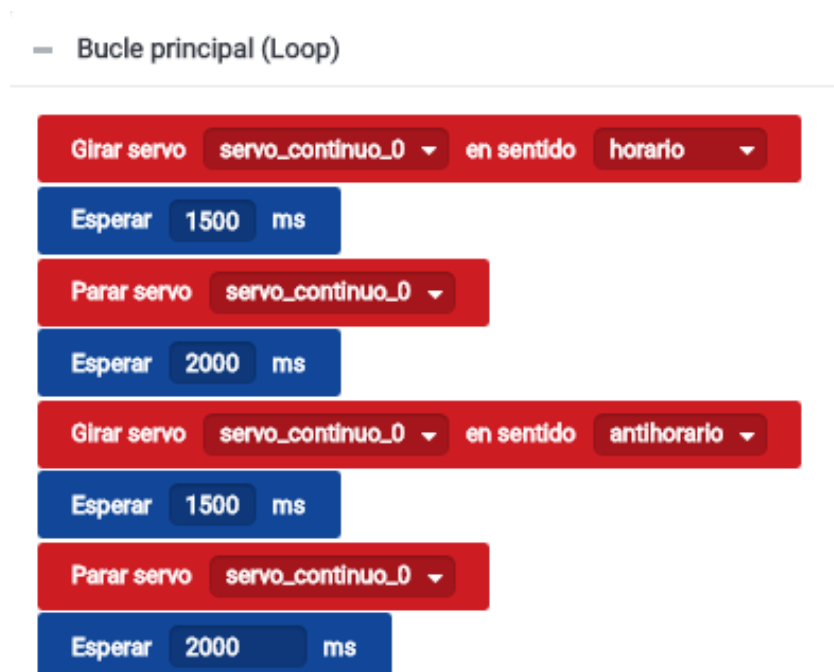
Tras esto, podemos pedirles que construyan un pequeño robot en el que peguen dos servos con cinta adhesiva con sus cabezales (a modo de rueda) y programen su movimiento. Un ejemplo es el siguiente:



## Programa tu casa domótica

Una vez que conocen el funcionamiento de estos componentes, tendrán que programar la puerta que han construido. Para ello, la programación variará en función de si se utiliza un miniservo o un servo de rotación continua.

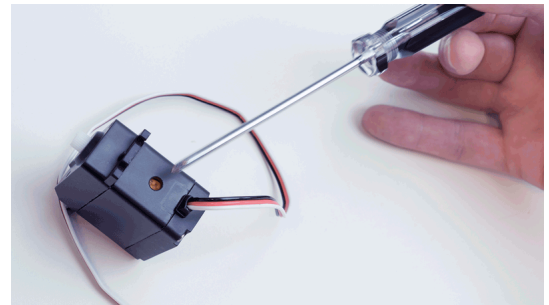
En el ejemplo, se ha utilizado un servo de rotación continua, de manera que éste tira de una cuerda que a su vez sube o baja la puerta. La programación que se ha utilizado es la siguiente:



El servo de rotación continua gira durante 1,5 segundos para abrir la puerta. Después se para y espera 2 segundos para que pase el coche. Por último, cierra la puerta (girando en sentido contrario durante 1,5 segundos) y se para otros 2 segundos. Al estar dentro del *Loop* la programación se repetirá continuamente.

La programación variará dependiendo de la posición del servo y de la longitud de la cuerda, por lo que los alumnos tendrán que ir modificando el tiempo de espera hasta conseguir que se abra y cierre la puerta correctamente.

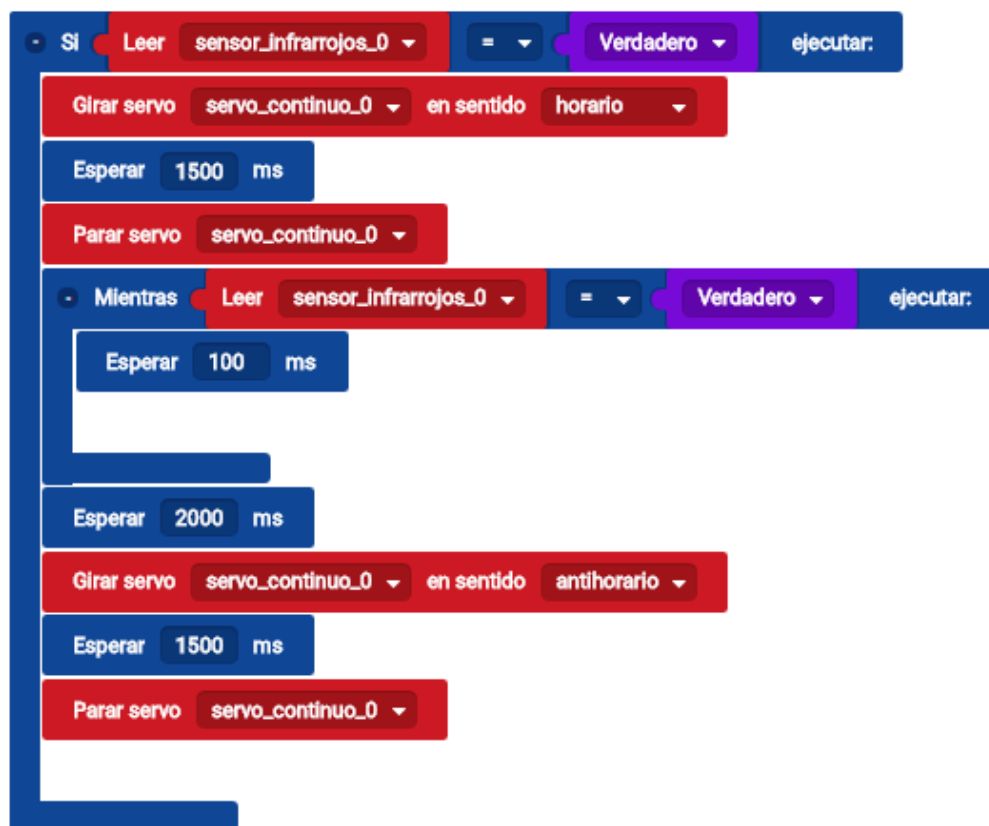
Es posible que el servo se mueva cuando tiene que estar parado y haya que calibrarlo. Para ello, deberemos **hacer un programa que pare el servo y cargarlo a la placa**. Comprobaremos si se mueve y en caso de que así sea, cogeremos un destornillador plano y giraremos lentamente en uno de los sentidos hasta que se pare.



Una vez que tienen la programación de la apertura de puerta, deberán integrarla con el sensor IR, de manera que la puerta sólo se abra cuando detecte que hay un coche.

La programación que tendrán que crear tendrá la siguiente función: cuando el IR detecta que hay un coche (verdadero), el servo de rotación continua gira para abrir la puerta y se para cuando termina. Una vez que la puerta está abierta, tenemos que indicar que mientras que el IR detecte que el coche está encima espere para que pase el coche.

#### — Bucle principal (Loop)



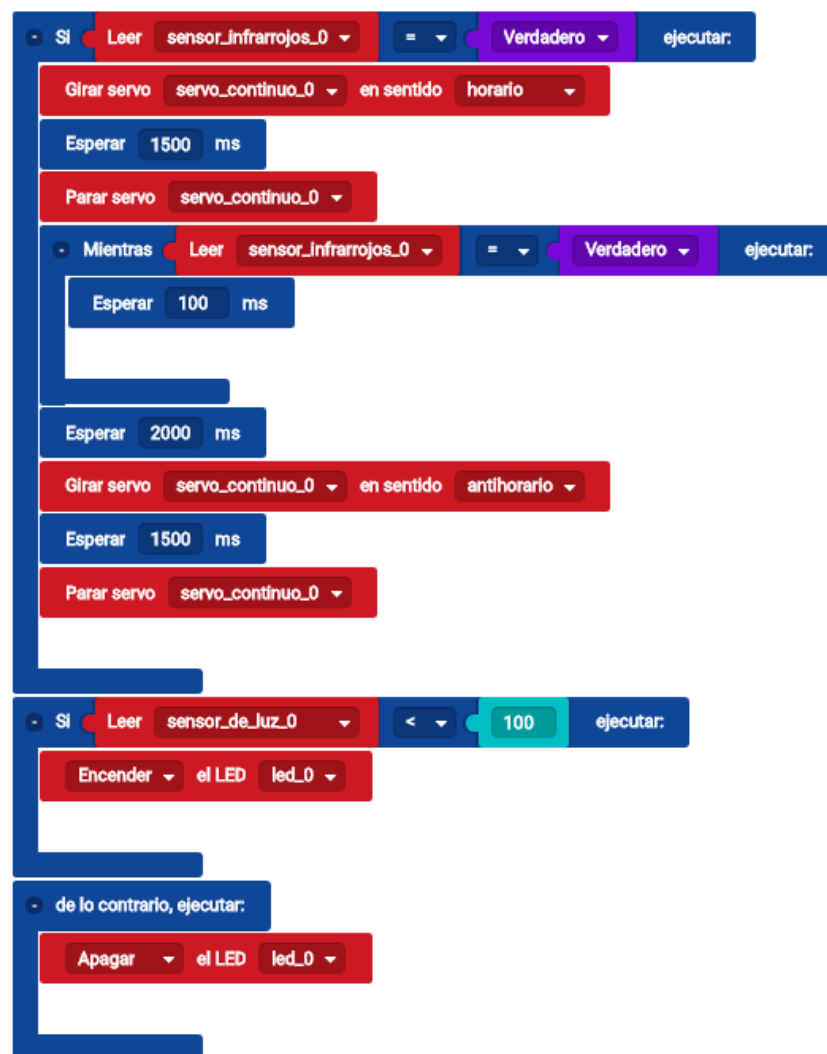
En el momento que no hay un coche, es decir, el IR detecta falso o 0, sale del bucle *Mientras* y ejecuta los bloques que tiene a continuación, es decir, espera un poco (2 segundos), para que le dé tiempo al coche a pasar, gira el servo para que baje la puerta y cuando termina de cerrarse, se para.

De esta manera, cuando vuelva a detectar que hay un coche, realizará el mismo proceso.

Una vez que tienen la programación de los dos elementos (luz inteligente y puerta automática), deberán juntarlas para que puedan funcionar a la vez. Para ello, sólo tendrán que añadir la programación de la luz debajo de la programación de la puerta.

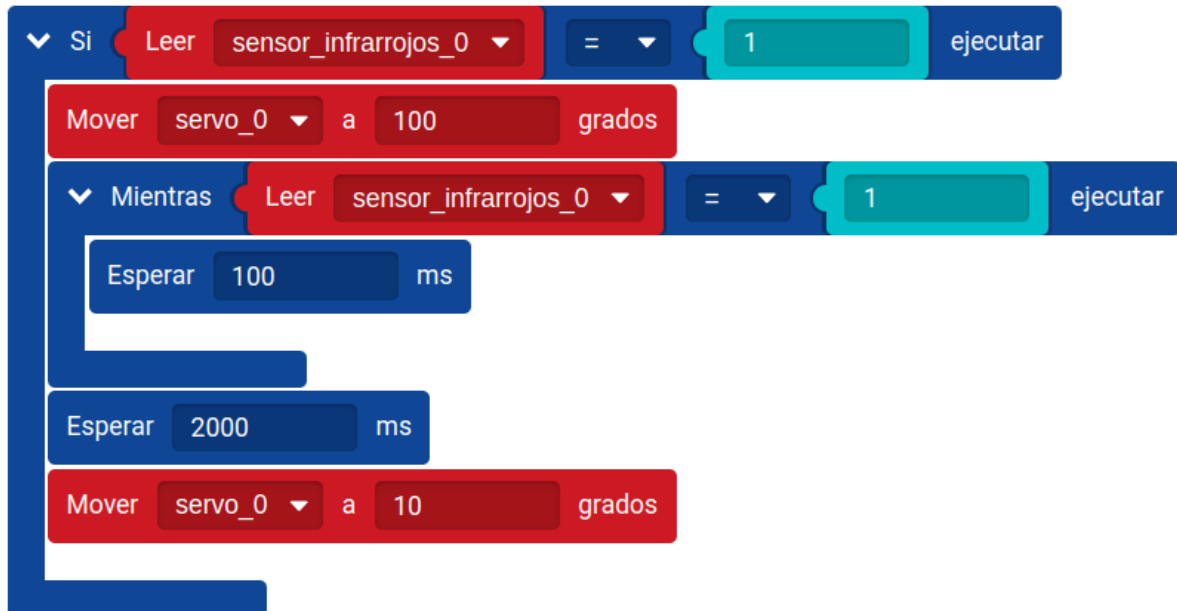
La programación completa del ejemplo es la siguiente:

— Bucle principal (Loop)



Si los alumnos han terminado de programar los dos elementos que se proponen, les pediremos que mejoren su casa programando otros elementos que se les ocurran.

Si hubiéramos usado el **Miniservo**, tendríamos que sustituir el interior del condicional del sensor IR, por estos bloques:



Si los alumnos han terminado de programar los dos elementos que se proponen, les pediremos que mejoren su casa programando otros elementos que se les ocurran.

Algunas mejoras que se pueden programar son las siguientes:

- Alarma.
- Cortinas o persianas automáticas.
- Ascensor.
- Comedero automático para mascotas.
- Timbre

Para programar estas mejoras, necesitarán conocer el resto de componentes del kit. Para ello, podemos pedir a los alumnos que vean las diferentes actividades y tutoriales disponibles en [bitblog.cc](http://bitblog.cc).




**Temporalización sugerida:**

**1ª Sesión:** Experimentación con el kit: encender y parpadear un LED y ver valores de luz del aula con el sensor de luz. Programar una luz inteligente.

**2ª Sesión:** Construcción de una casa. Diseño de lámpara o farola para incorporar luz inteligente.

**3ª Sesión:** Experimentación con sensor de distancia, Servo de rotación continua y Miniservo. Programación de una puerta automática.

**4ª Sesión:** Incorporación de mejoras y corrección de errores.

//Contacto:  Ayuda pedagógica:  
[retotech@fundacionendesa.org](mailto:retotech@fundacionendesa.org)