RaspberryPi GPIO Tutorial: LEDs and buttons

Link to this page: catrob.at/RaspberryPi

First steps with Raspberry Pi in Pocket Code: create blinking LEDs, use buttons as input

What you will learn:

- set up your Raspberry Pi
- control LEDs/outputs from Pocket Code
- use buttons/inputs from a Raspberry Pi in Pocket Code

If you encounter any bugs, please let us know and file a report. Thank you!

Table of Contents:

Install RaspberinoServer on your Raspberry Pi

For the following steps, you either have to enter the following commands in the terminal of your Paspberry Pi using a keyboard and a display, or you can connect to your Raspberry Pi via SSH from your computer or phone.

1. Download the installer

   ```
   wget http://catrob.at/installraspberino -O install-raspberino.sh
   ```

2. Run the installer as root

   ```
    sudo sh install-raspberino.sh
   ```

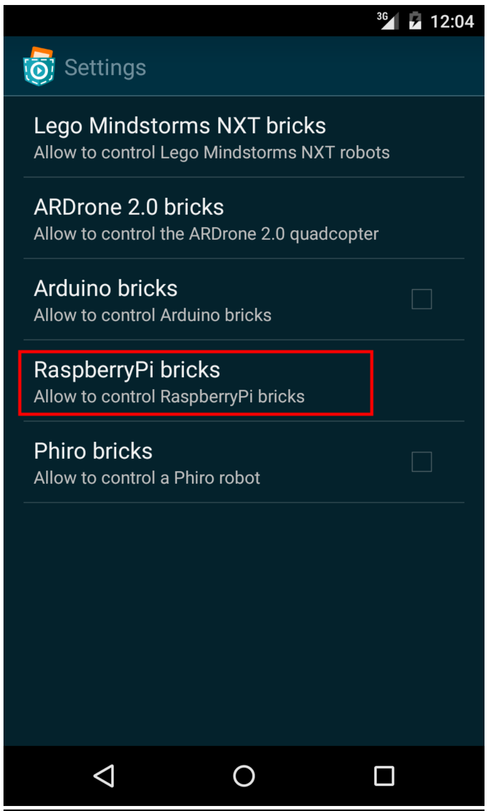
   Enter "y" to start the installation
   The installer downloads the server and installs everything. From now on the RaspberinoServer should be ready to use and start automatically when the Raspberry Pi is powered on.

   To update the server, just run the installer again (Step 2).

Enable and Configure Raspberry Pi in Pocket Code

If you haven't already done so, you'll have to enable and configure your Raspberry Pi under Settings in "Raspberry Pi bricks". Otherwise you can skip this section and move straight to the tutorials.

1. **tick** the check-box to enable Raspberry Pi bricks
2. change the settings if necessary:
   a. enter IP-/Host-Address
      i. very often **"raspberrypi"** works as default host address
      ii. if your RPi has a keyboard + monitor: find out the IP-address from a terminal via ifconfig
      iii. your router might list all connected computers within the network
      iv. if all of that fails: find out the IP-address via nmap
   b. Port
      i. the default is **10000**
      (don't change it unless you changed the script on the Raspberry Pi)
   c. GPIO version
      i. select the Raspberry Pi model you are using.


12:04

Settings

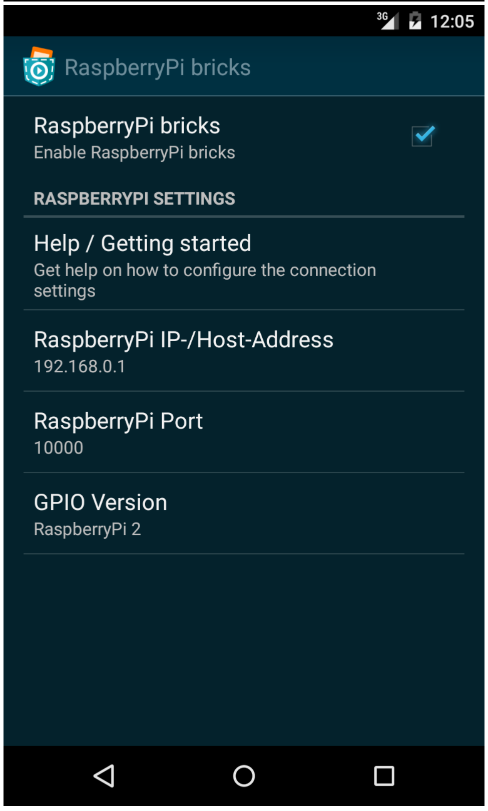
Lego Mindstorms NXT bricks
Allow to control Lego Mindstorms NXT robots

ARDrone 2.0 bricks
Allow to control the ARDrone 2.0 quadcopter

Arduino bricks
Allow to control Arduino bricks

RaspberryPi bricks
Allow to control RaspberryPi bricks

Phiro bricks
Allow to control a Phiro robot


12:05

RaspberryPi bricks

RaspberryPi bricks ✓
Enable RaspberryPi bricks

RASPBERRYPI SETTINGS

Help / Getting started
Get help on how to configure the connection settings

RaspberryPi IP-/Host-Address
192.168.0.1

RaspberryPi Port
10000

GPIO Version
RaspberryPi 2

Now you can add Raspberry Pi bricks in your Projects. 🙂

The Raspberry Pi Bricks and Sensor in a Nutshell

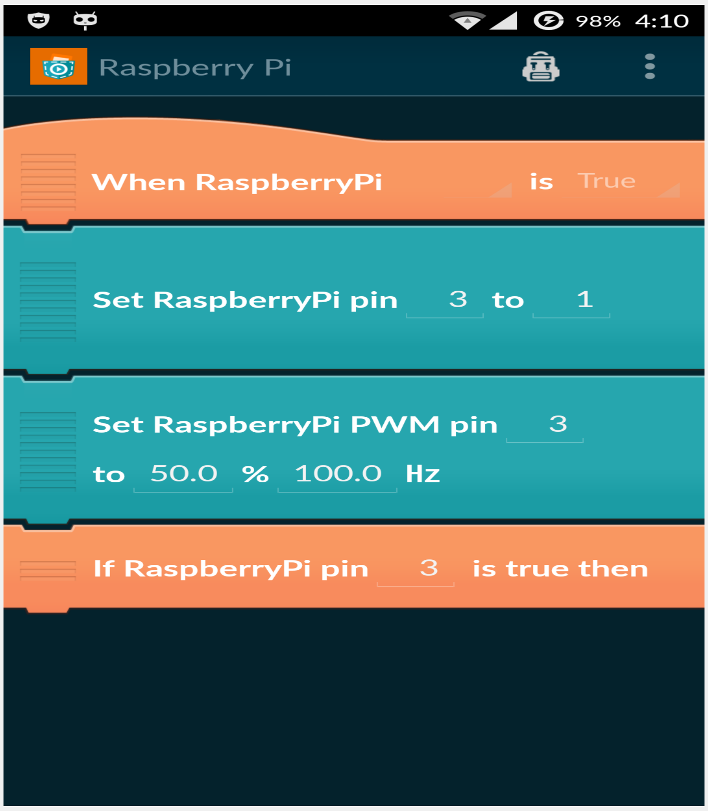
This section shortly explains the Raspberry Pi bricks.

Pin-Numbers:

Your Raspberry Pi offers a huge number of GPIO (general purpose input/output) pins. The number of pins available depends on your Raspberry Pi version. Pocket Code uses the pin numbering of the board layout (the top left pin in pin 1). Note that not all pin numbers correspond to a GPIO pin.

To find out more about the available pins and their corresponding numbers, have a look here: Raspberry Pi Pinout

Bricks:



When RaspberryPi is True

Set RaspberryPi pin 3 to 1

Set RaspberryPi PWM pin 3 to 50.0 % 100.0 Hz

If RaspberryPi pin 3 is true then

When Raspberry Pi pin <*pin*> is *True/False*:

- this is a script-brick, meaning that is the first one (no predecessor)
- the bricks placed under such a script-brick get executed, whenever the value of a given input-pin changes to True/False

Set Raspberry Pi pin <*pin*> to <*value*>:

- sets the output of a given <pin> to a given <value>
- allowed values: 0 or 1 (otherwise the brick has no effect)

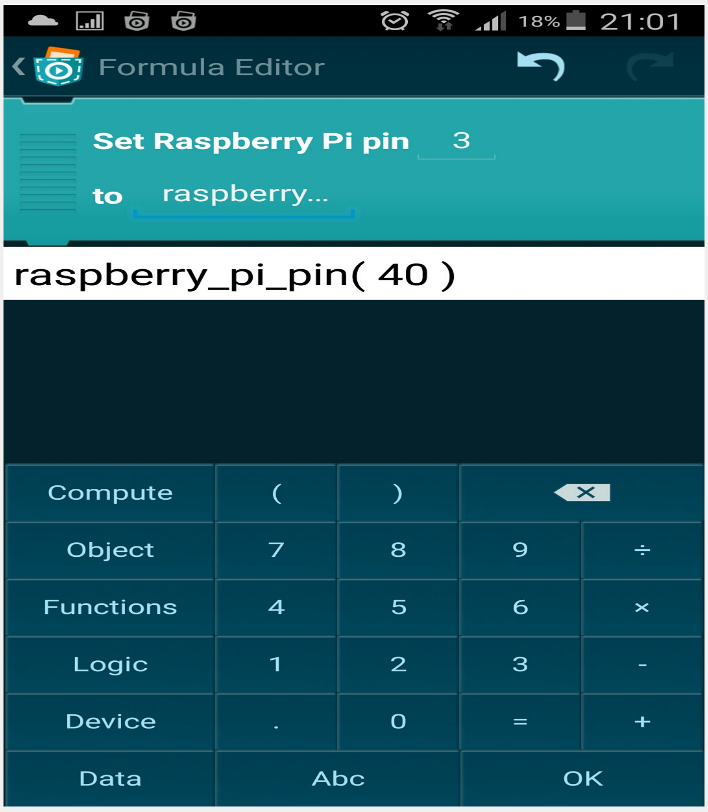
Set Raspberry Pi PWM pin <*pin*> to <*duty cycle*> %, <percentage> Hz:

- PWM (pulse width modulation) can be used to obtain an analogue behaviour with a digital output
- this can be used to dim LEDs, control servos, etc.
- here you can find an explanation about how PWM works.
- see also our Raspberry Pi LED stripe demo/tutorial

If Raspberry Pi pin <pin> is true, then:

- this brick works similar to the standard If - Then - else brick
- if the <*pin*> input is 1 (aka True or High), the then-part is executed, otherwise the else-part is executed

Sensor:



Set Raspberry Pi pin 3 to raspberry...

raspberry_pi_pin(40)

The raspberry_pi_pin(<pin>) sensor:

is another way besides the If Raspberry Pi pin-Brick to use digital inputs.

- can be used to retrieve the input value of a given <pin>
- available in the *Formula Editor* via the *Device* button
- the resulting input value is either 0 or 1
  - (for invalid pin numbers the returned value is 0)

<<<< The example sets the GPIO-output pin 3 to the value of the GPIO-input pin 40.

Tutorial 1: Blinking LED (Hello World)

Everyone starts with a simple Hello World program when learning a new programming language. When using a Raspberry Pi, "Hello World" simply means a blinking LED.

Step1: connect the circuit

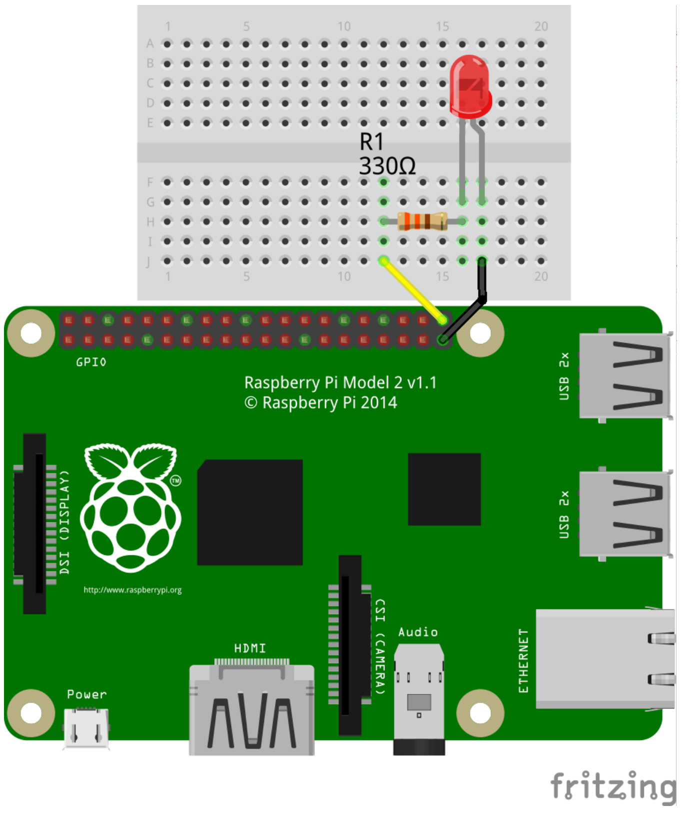
What you need:

- LED
- Resistor (around 220 - 330 Ohm)
- Bread board + wires

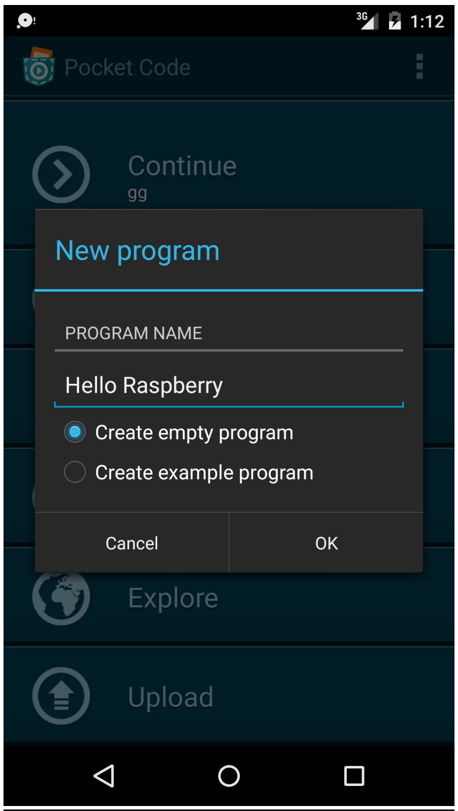
The LED typically has a longer and a shorter leg. The shorter leg has to be connected to GND on a Raspberry Pi (black wire).

The other end of the leg is connected via a resistor (here 330 Ohm) to a GPIO pin (here: pin 40 using the yellow wire).

Note that the holes in one line (marked green) are linked together.



Step2: Create your first Program using Raspberry Pi bricks in Pocket Code


1:12

Pocket Code

Continue
gg

Explore

Upload

New program

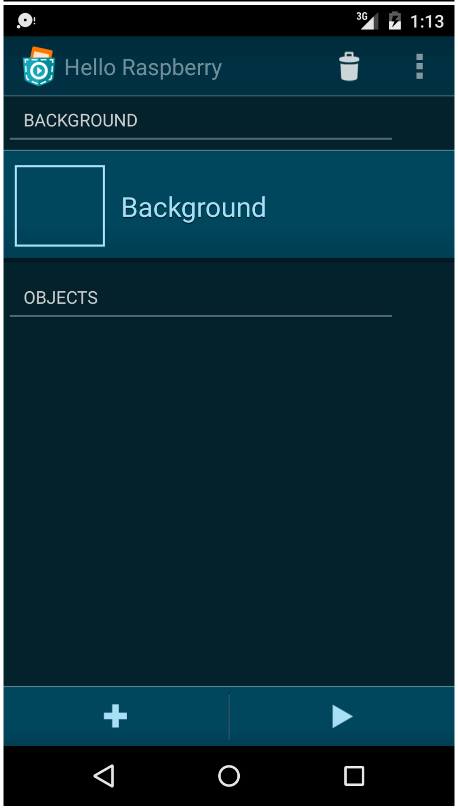
PROGRAM NAME

Hello Raspberry

[✓ Create empty program]

Create example program

Cancel | OK

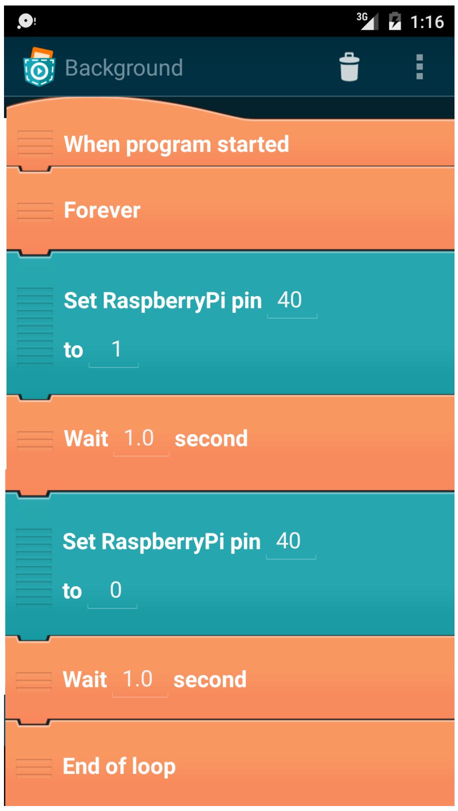

1:13

Hello Raspberry

BACKGROUND

Background

OBJECTS



Background

When program started

Forever

Set RaspberryPi pin 40
to 1

Wait 1.0 second

Set RaspberryPi pin 40
to 0

Wait 1.0 second

End of loop

1. create new **empty program**
2. go to Background -> Scripts
3. place the code bricks as in the image

Finally, press the **Play** button. If your code is correct and everything is wired correctly, your LED should blink every second. Well done! 🙂

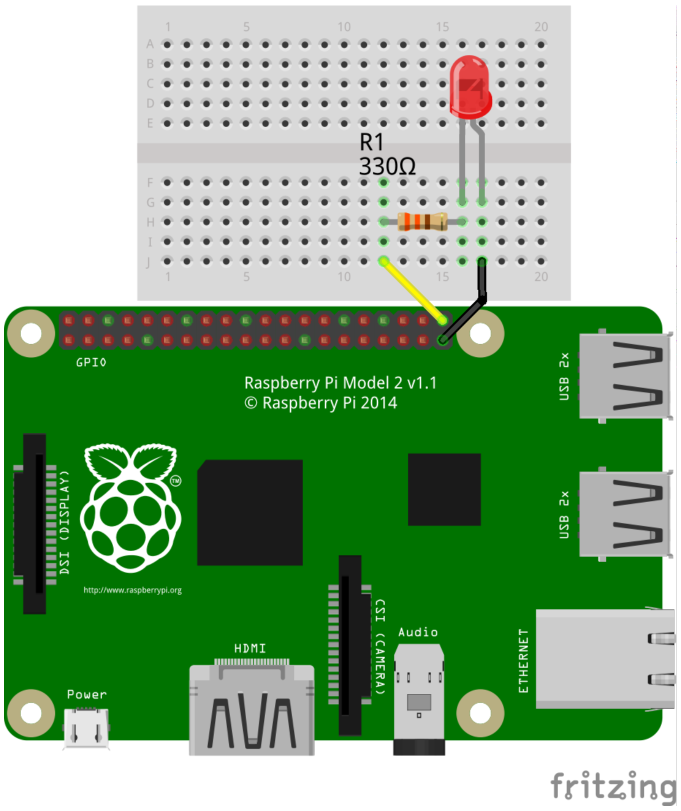
Tutorial 2: Dimming a LED (PWM)

The previous tutorial turned a LED on and off periodically. If you have done the previous tutorial, you can reuse the circuit for this tutorial.

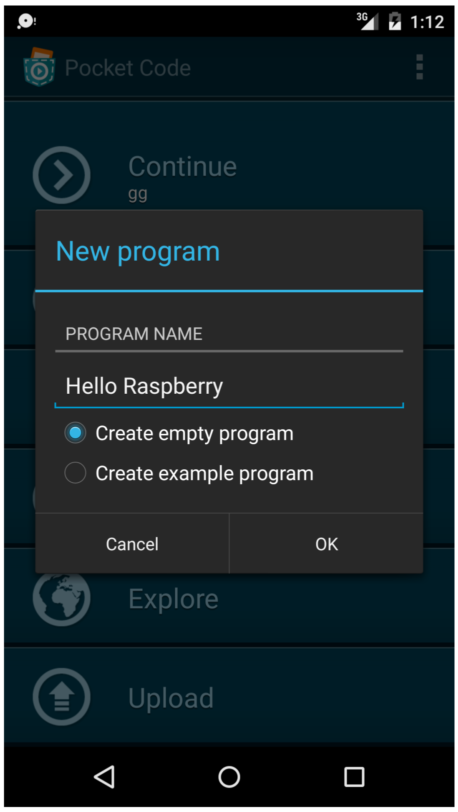
PWM (pulse width modulation) can be used to obtain an analogue behaviour with a digital output. This can be used to dim LEDs. If you want to know more about PWM, you can find an explanation about how PWM works here.

Step1: connect the circuit

the same as in Tutorial 1 (blinking light).



Step2: Create your first Program using Raspberry Pi bricks in Pocket Code



New program

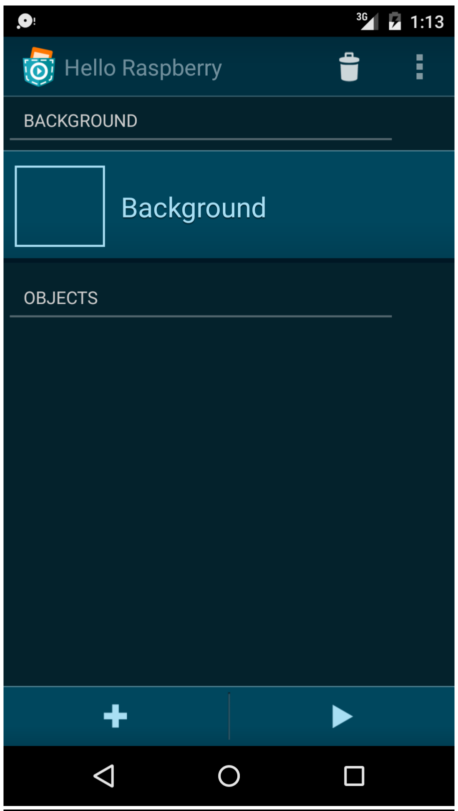
PROGRAM NAME

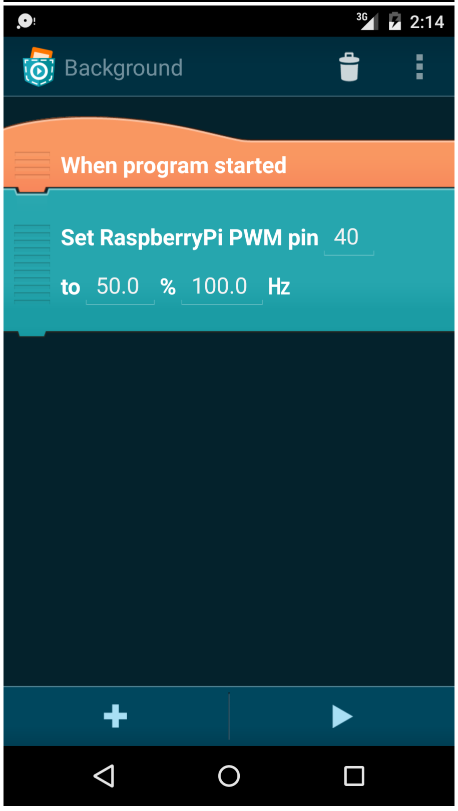
Hello Raspberry

[✓ Create empty program]

Create example program

Cancel OK





When program started

Set RaspberryPi PWM pin 40 to 50.0 % 100.0 Hz

1. create new **empty program**
2. go to Background -> Scripts
3. place the code bricks as in the image

Now start the program by pressing **Play**. You can try out different percentages (instead of 50%).

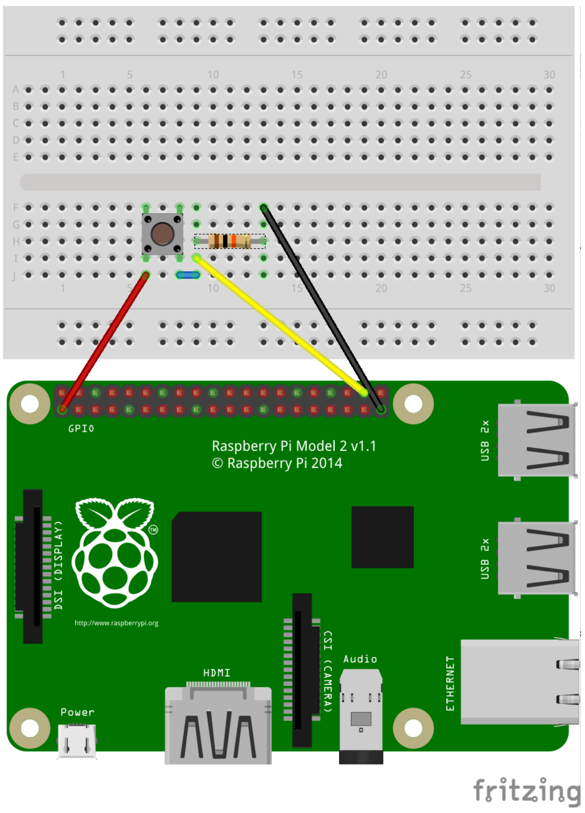
The Raspberry Pi dims the led by switching on and off the LED very fast. A higher percentage means that the time "on" is longer, whereas a lower percentage means that the LED is "off" most of the time. If this happens very fast (here the frequency is 100Hz, meaning that the LED is turned on and off 100 tmes per second), then the LED is dimmed.

If you set the PWM pin to 50% and 0.5 Hz, you'll get the same result as with the code in Tutorial 1 🙂

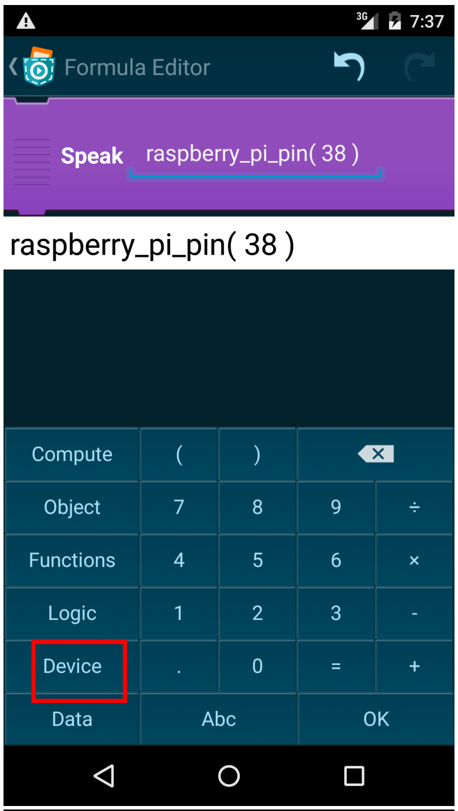
Tutorial 4: read input values

This tutorial shows how to read the pin value of a Raspberry Pi. When starting the following example, your phone speaks either the value "1" or "0", depending on the button state

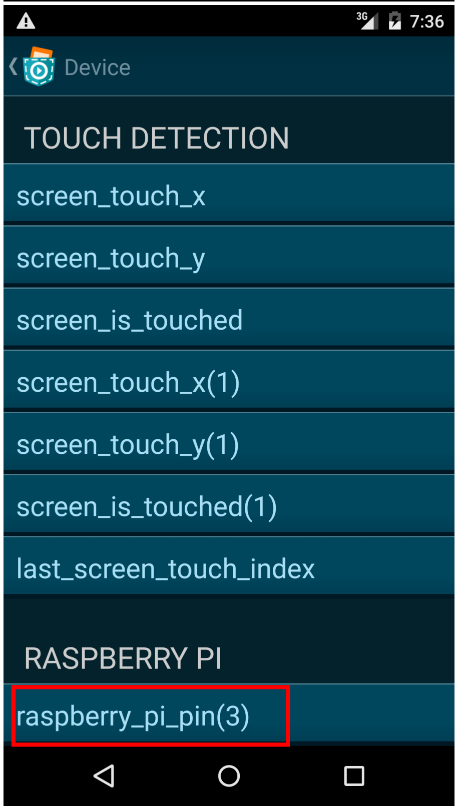
Step1: connect the circuit

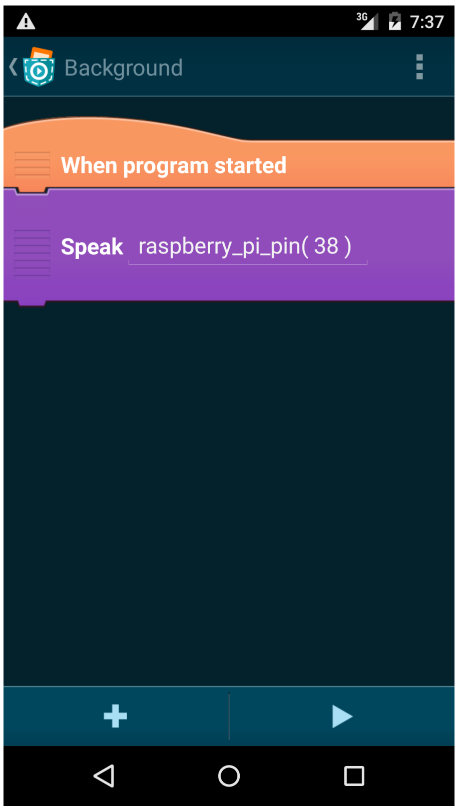


Step2: create the Program in Pocket Code



raspberry_pi_pin(38)





When program started

Speak raspberry_pi_pin(38)

1. add a new "Speak" brick
2. click into the speak text to open the formula editor
3. delete the text and click "Device"
4. scroll down to the RASPBERRY PI section and choose the raspberry_pi_pin() sensor
5. select the pin number where you wired the button. (Here pin 38)

Another method to read an input value would be to use the Raspberry Pi "If pin .. is set" brick

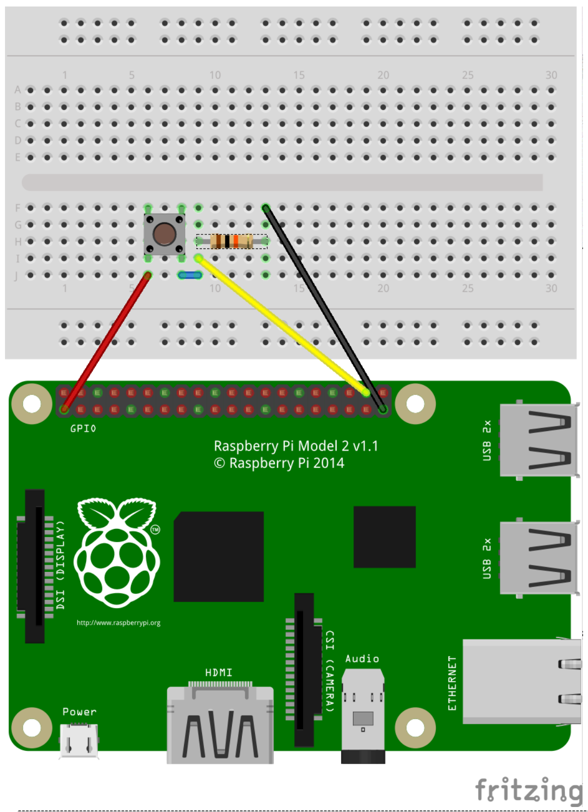
Tutorial 4: recognize input events (The invisible bird)

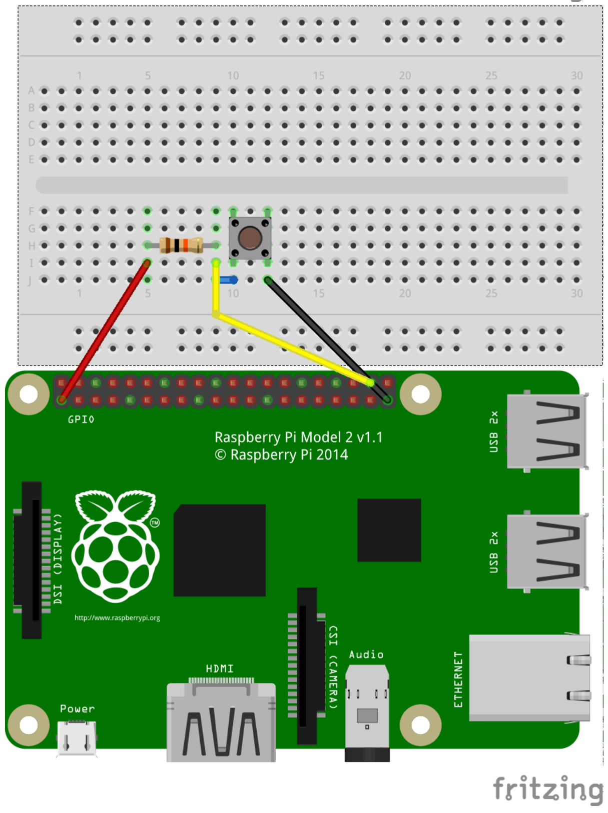
In this tutorial, we are going to remix the default Pocket Code program. Whenever a button on the raspberry Pi is pressed, the bird should be invisible, and when the button is released, the bird should be visible again.

What you need:

- push-button
- Resistor (10kOhm and more)
- BreadBoard

Step1: connect the circuit



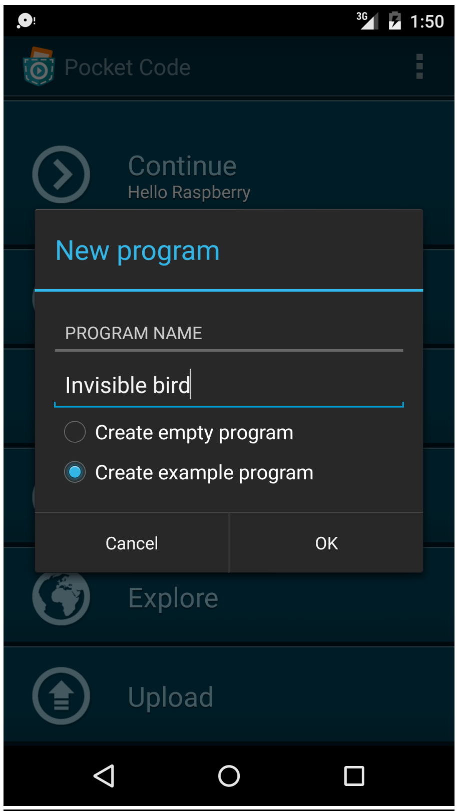


There are two possible ways how to use a button as input on a Raspberry Pi:

1. using a pull-down resistor (first image): when the button is pressed, the input value is high (otherwise low).
2. using a pull-up resistor (second image): when the button is pressed, the input value is low (otherwise high).

You can try out both variants if you want and see what the difference is. 😉

Step2: create the program in Pocket Code


Pocket Code

Continue
Hello Raspberry

New program

PROGRAM NAME

Invisible bird

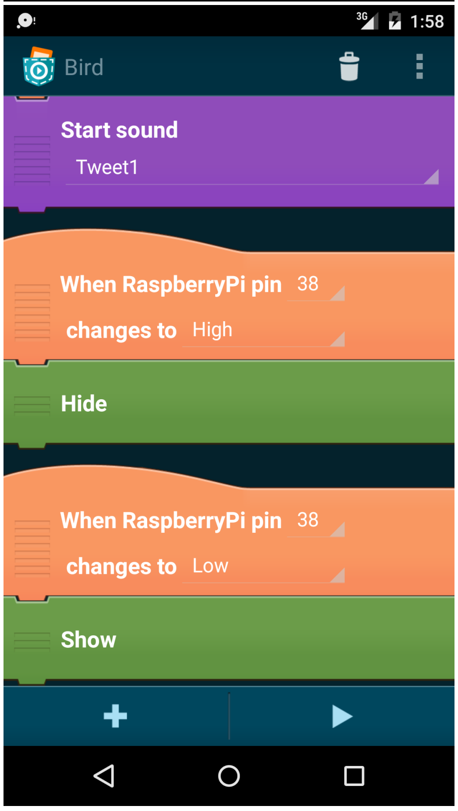
( ) Create empty program

(●) Create example program

Cancel OK

Explore

Upload


Bird

Start sound
Tweet1

When RaspberryPi pin 38
changes to High

Hide

When RaspberryPi pin 38
changes to Low

Show

1. create new **example program**
2. go to Bird -> scripts
3. add the two "When Raspberry Pi pin changed" bricks as shown in the tutorial

Start the program by pushing the **Play** button. Try out pushing and releasing the button.In one situation, the bird will be invisible, in the other it will be visible again, depending on if you have built the circuit with a pull-up or pull-down resistor. 🙂

Further project ideas

Now you should already know how to use the pins of your Pi with Pocket Code. If you are looking for a new project, check out our tutorial for a LED strip light where you set the color with your phone: Raspberry Pi LED stripe demo/tutorial

If you have created your own exciting projects, please let us know! 🙂