



PHARO IoT

Present and Future

Alex Oliveira

alex.oliveira@msn.com
twitter.com/alex_oliveira

Marcus Denker

marcus.denker@inria.fr
twitter.com/marcusdenker

Nobert Hartl

norbert@2denker.de
twitter.com/NobertHartl

Summary

- 1 – Overview and improvements
- 2 – Collaborative work
- 3 – Projects using Pharo IoT
- 4 – Future

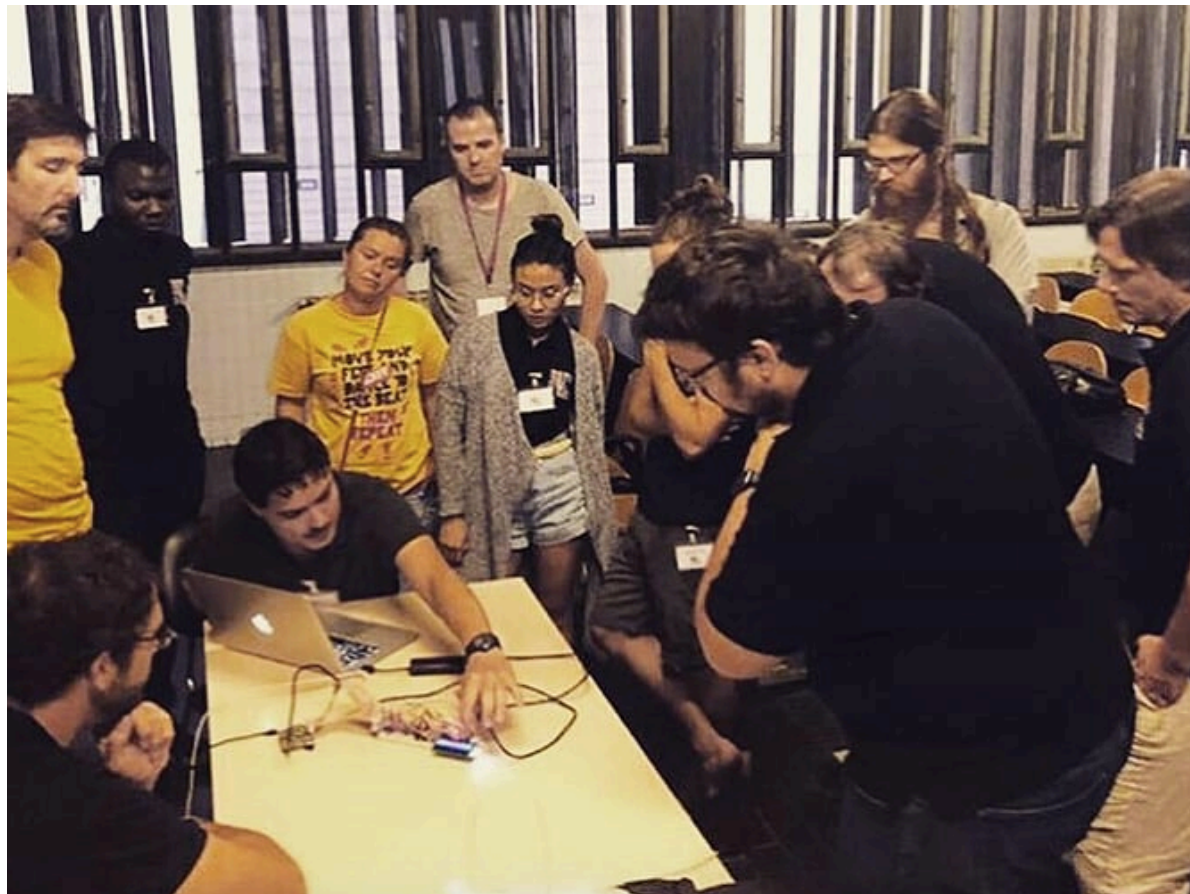
1 - Overview

- Created by **Rmod Team**, a research team from **INRIA** (France)
- Written by Denis Kudriashov in 2016/17
dionisiydk@gmail.com
- In 2018, Alex Oliveira joined the Rmod Team to continue the project

What is Pharo IoT?

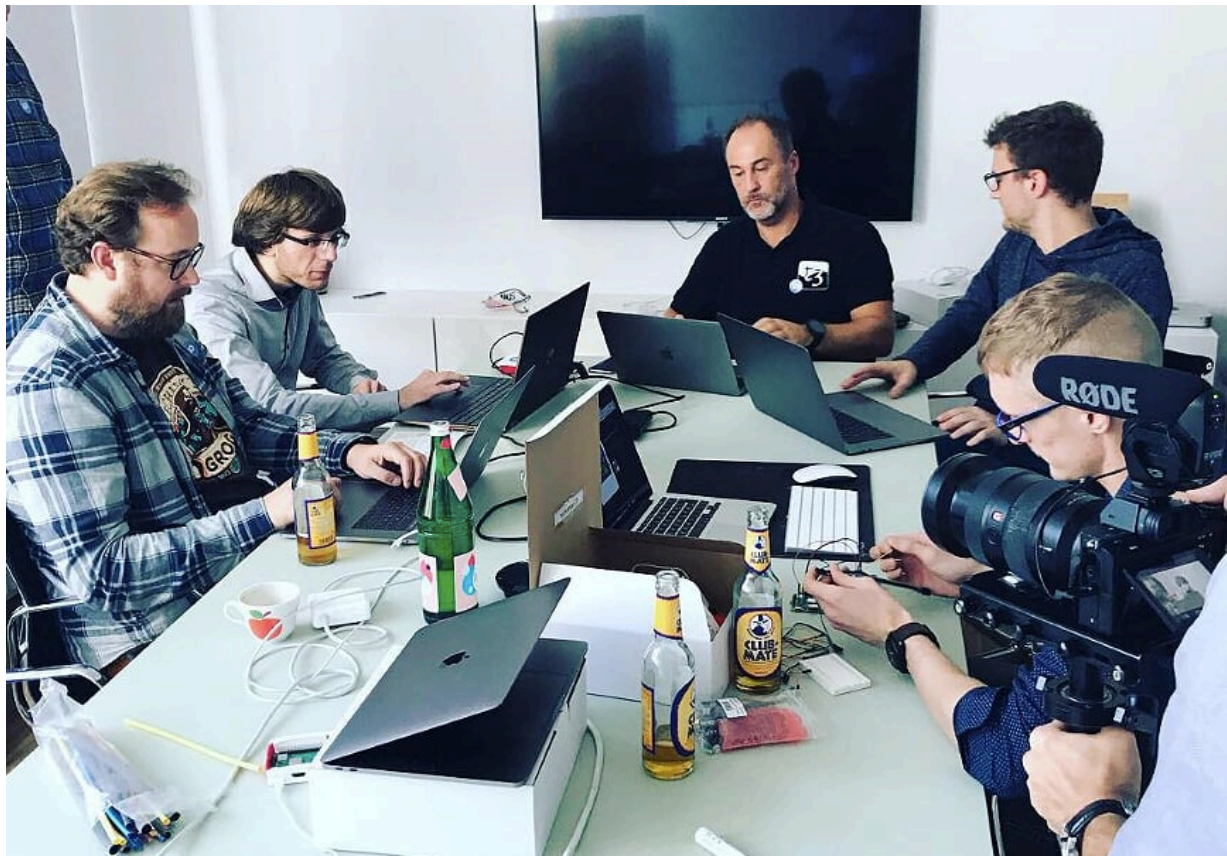
- A **Pharo image** running on IoT device (ARM VM)
 - A Pharo library to control GPIOs (PharoThings)
- A **Remote IDE**
 - Remote Playground, Browser, Inspectors
 - An advanced board inspector for **Raspberry PI**
- Other IoT Projects:
 - A Pharo library to control **Arduino** Devices (Firmata)

Pharo IoT in the world



IoT Workshop, ESUG
September 2018 - Cagliari, Italy

Pharo IoT in the world



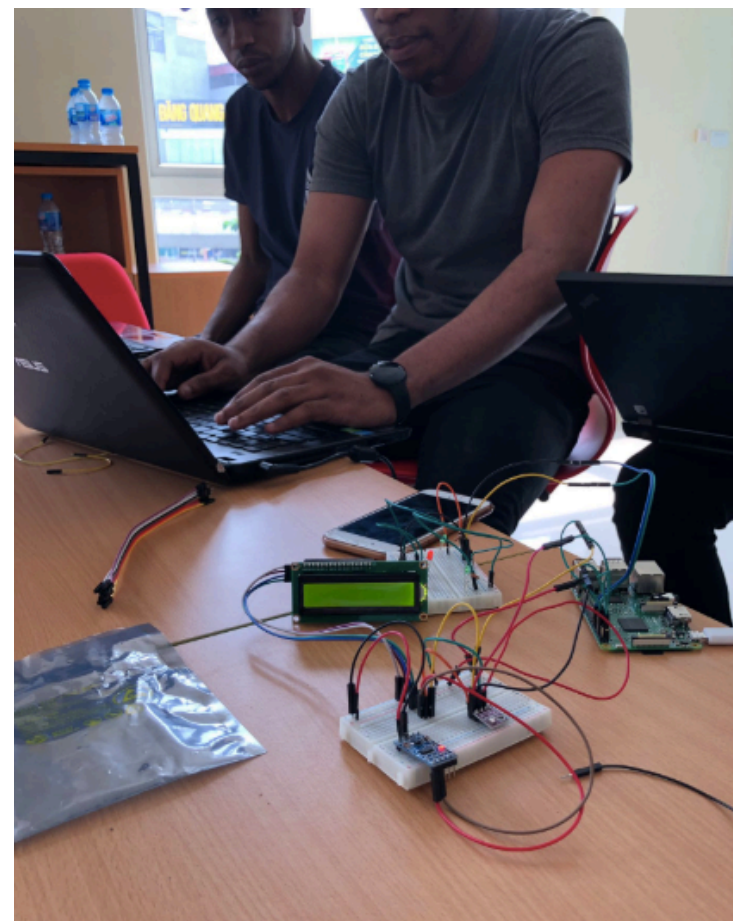
IoT Hackaton, Zweidenker GmbH
October 2018 - Cologne, Germany

Pharo IoT in the world



Live Programming IoT devices with PharoThings
January 2019 - Can Tho University, Vietnam

Pharo IoT in the world

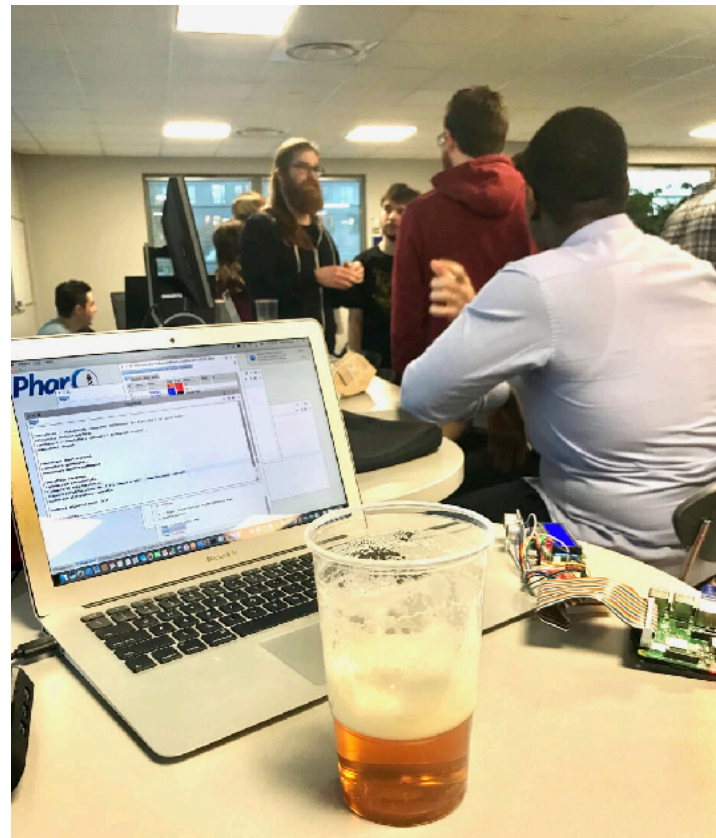


Live Programming IoT devices with PharoThings
May 2019 - International Francophone Institute, Vietnam

Pharo IoT in the world



ESUG Conference
European Smalltalk User Group
Sep 2018 Cagliari, Italy



INRIA
Pharo 10 Years
Nov 2018 Lille, France



USTH
University of Science and
Technology of Hanoi
Jan 2019 Hanoi, Vietnam

Improvements

1. Easy installation (**zero-conf** scripts) less than 1 min
2. Zero-conf pages hosted in Github
3. Everything packed (VMs, 32/64 images, 1 click-run files)
4. Installing **from scratch** with Raspbian in less than 10 min
5. Code improvements and support to **new sensors**
6. **Pharo IoT Booklet** with many lessons
7. Welcome window with code examples
8. Pharo IoT website **pharoiot.org**
9. Using Continuous Integration - **CI** Travis

How to easy install (zero-conf)

1. Run the command to download and extract the files:
 - **wget -O - get.pharoiot.org/server | bash**
2. Run TelePharo server:
 - 1 click on pharo-server file or...
 - type in terminal: **./pharo-server**

Less than 1 minute!

get.pharoiot.org

- We are using the Github Pages to host the zero-conf pages

The image shows two overlapping screenshots. The background screenshot is a web browser displaying the 'Pharo IoT Server Zeroconf Raspberry' page. The page content includes:

- Pharo IoT Server Zeroconf Raspberry**
- This script downloads `server.zip` file that contain:
- Pharo7 image 32 bit
- Pharo ARM VM
- Pharo IoT server installed
- Plattaform**
- Raspberry Pi running Raspbian
- Usage**
- `wget -O - - get.pharoiot.org/server | bash`
- Artifacts**
- pharo Script to run Pharo in the headless mode
- pharo-ui Script to run Pharo in UI mode
- pharo-server/ Start pharo in headless mode with TelePharo listening on po
- vm/ Directory containing the VM
- Pharo IoT Server Example**
- Start Pharo IoT server: `./pharo-server`
- Open Pharo user interface: `./pharo-ui`
- Start the server (Playground): `TlpRemoteUIManager registerOnpPort:40423`

The foreground screenshot is a GitHub repository view for 'pharo-iot / Ci'. It shows the repository structure and a list of files with their commit history:

File	Commit Message	Time
..		
client		10 days ago
multi		10 days ago
server		10 days ago
CNAME	Update CNAME	10 days ago
client.zip	Travis upload release files	6 days ago
index.html	Update index.html	9 days ago
multi.zip	Travis upload release files	6 days ago
pibakeryPharoloT.xml		10 days ago
server.zip	Travis upload release files	6 days ago

<https://github.com/pharo-iot/Ci/docs>

Everything packed

- + Pharo Image 32/64
- + PharoThings loaded
- + ARM VM
- + Windows, Linux, Mac VMs

Name	Size	Kind
pharo	2 KB	Unix executable
pharo-local	--	Folder
pharo-server	2 KB	Unix executable
pharo-ui	2 KB	Unix executable
pharo.bat	60 bytes	Batch File
Pharo7.0-32bit-890f474.sources	34,3 MB	Squeak...es File
PharoThings32.changes	1,5 MB	Squeak...es File
PharoThings32.image	57,4 MB	Pharo Image File
PharoThings64.changes	1,5 MB	Squeak...es File
PharoThings64.image	66,6 MB	Pharo Image File
vm	--	Folder
arm	--	Folder
linux32	--	Folder
linux64	--	Folder
osx64	--	Folder
win32	--	Folder

1-click run files

- + Pharo Image 32/64
- + PharoThings loaded
- + ARM VM
- + Windows, Linux, Mac VMs
- + **1 click run files**
 - pharo-ui
 - pharo-server
 - pharo
 - pharo.bat

```
pharo-server x
13 OS="win";
14 elif [[ "${TMP_OS}" = *mingw* ]]; then
15 OS="win";
16 elif [[ "${TMP_OS}" = *msys* ]]; then
17 OS="win";
18 else
19 echo "OS not identified. Try to run the correct VM on vm folder";
20 exit 1;
21 fi
22 # 1.2 Getting the architecture
23 if [[ "${TMP_ARCH}" = *arm* ]]; then
24 ARCH="arm";
25 elif [[ "${TMP_ARCH}" = *64* ]]; then
26 ARCH="64";
27 else
28 ARCH="32";
29 fi
30 # 1.3 Setting the correcty VM folder
31 if [[ "${ARCH}" = *arm* ]]; then
32 VM="arm";
33 elif [[ "${OS}" = *windows* ]]; then
34 VM="win32";
35 elif [[ "${OS}" = *mac* ]]; then
36 VM="osx32";
37 elif [[ "${OS}" = *linux* ]] && [[ "${ARCH}" = *64* ]]; then
38 VM="linux64";
39 elif [[ "${OS}" = *linux* ]] && [[ "${ARCH}" = *32* ]]; then
40 VM="linux32";
41 else
42 echo "VM not identified. Try to run the correct VM on vm folder";
43 exit 1;
44 fi
45
46 # Step 2 - Running the correcty VM and Pharo image
47 if [[ "${VM}" = *arm* ]]; then
48 vm/$VM/pharo --headless PharoThings32.image remotePharo --startServerOnPort=40423
49 elif [[ "${VM}" = *osx32* ]]; then
50 # some magic to find out the real location of this script dealing with symlink
51 DIR=`readlink "$0" || DIR="$0";`
52 DIR=`dirname "$DIR"`;
53 cd "$DIR"
54 DIR=`pwd`
55 cd - > /dev/null
56 # disable parameter expansion to forward all arguments unprocessed to the VM
57 set -f
58 # run the VM and pass along all arguments as is
59 "$DIR"/"vm/$VM/Pharo.app/Contents/MacOS/Pharo" --headless "$DIR"/PharoThings32
60 elif [[ "${VM}" = *linux64* ]]; then
61 vm/$VM/pharo --headless PharoThings64.image remotePharo --startServerOnPort=40
62 elif [[ "${VM}" = *linux32* ]]; then
63 vm/$VM/pharo --headless PharoThings32.image remotePharo --startServerOnPort=40
64 fi
```

Installing from scratch

- + Installing Raspbian
- + Download Pharo IoT
- + Set Hostname
- + Enable I2C and SPI
- + Connect on WiFi
- + Start server every boot

Keyboard, mouse or monitor not required

Less than 10 minutes!



Board modelling improvements

Inspector on a RpiBoard3B

a RpiBoard3B

P1 Devices Raw Meta

Value	Function	Name	Pin#	Pin#	Name	Function	Value
		3.3v	1	2	5v		
● out	SDA (I2C)	gpio2	3	4	5v		
	SCL (I2C)	gpio3	5	6	Ground		
	Clock	gpio4	7	8	gpio14	TXD (Serial) ● out	
		Ground	9	10	gpio15	RXD (Serial)	
		gpio17	11	12	gpio18	PWM	
		gpio27	13	14	Ground		
		gpio22	15	16	gpio23		
		3.3v	17	18	gpio24		
● in	MOSI (SPI)	gpio10	19	20	Ground		
	MISO (SPI)	gpio9	21	22	gpio25		
	SCLK (SPI)	gpio11	23	24	gpio8	CE (SPI)	
		Ground	25	26	gpio7	CE (SPI)	
	SDA (I2C)	gpio0	27	28	gpio1	SCL (I2C)	
		gpio5	29	30	Ground		
		gpio6	31	32	gpio12	PWM	
	PWM	gpio13	33	34	Ground		
	MISO (SPI)	gpio19	35	36			
		gpio26	37	38			
		Ground	39	40			

PotLCDHD44780Gpio>>configurePeripherals

BaselineOfPharoThin
PharoThings-Device
PharoThings-Device
PharoThings-Device

PotLCDHD44780
PotLCDHD44780Gpio
PotLCD1602Device !
PotLCDHD44780I2C

instance side
accessing
commands
controlling
initialization

configurePeripherals
connect
initializeRegisters
isConfigured
isConnected

All Packages | Scoped View | Flat | Hier. | Inst. side | Class side | Methods | Vars | Class ref.

? Comment | PotLCDHD44780G x | configurePeriphei x | + Inst. side method x

```

configurePeripherals
  "Pin mapping:
  -----
  VSS|VDD|V0 |RS |RW | E |D0 |D1 |D2 |D3 |D4 |D5 |D6 |D7 | A | K
    |  |  |27 |  |22 |  |  |  |  |  |25 |24 |23 |18 |  |12
                                [8 BIT]          4 BIT
  -----"

  modePin := 13 gpioHeader. "RS 1 character or 0 lcd commands"
  clockPin := 15 gpioHeader. "EN clock enable"
  dataPins := #(12 16 18 22) collect: [ :id | id gpioHeader ]. "D7 D6 D5 D4 [D3
bit 4bit or [8bit]"
  backlightPin := 32 gpioHeader "BL backlight PWM"
  
```

```

led1 := gpio2.
led1 beDigitalOutput.
led1 value:1.

led2 := gpio14.
led2 beDigitalOutput.
led2 value:1.

button1 := gpio10.
button1 beDigitalInput.
button1 value
  
```

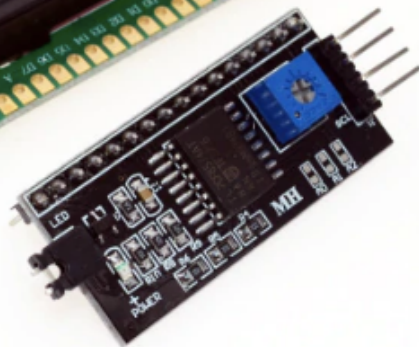
Board modelling improvements

1. Removed WiringPi numbers reference and start adopt BCM
2. Enable basic GPIO behaviour to all GPIOs
 - before were 14, now we can use 28 gpios
3. Add configure peripherals methods
4. Create the GPIO instance using *header number* or *GPIO number*

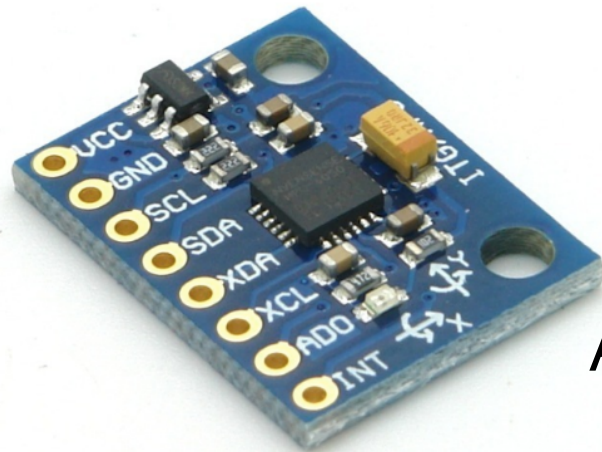
Support to new sensors



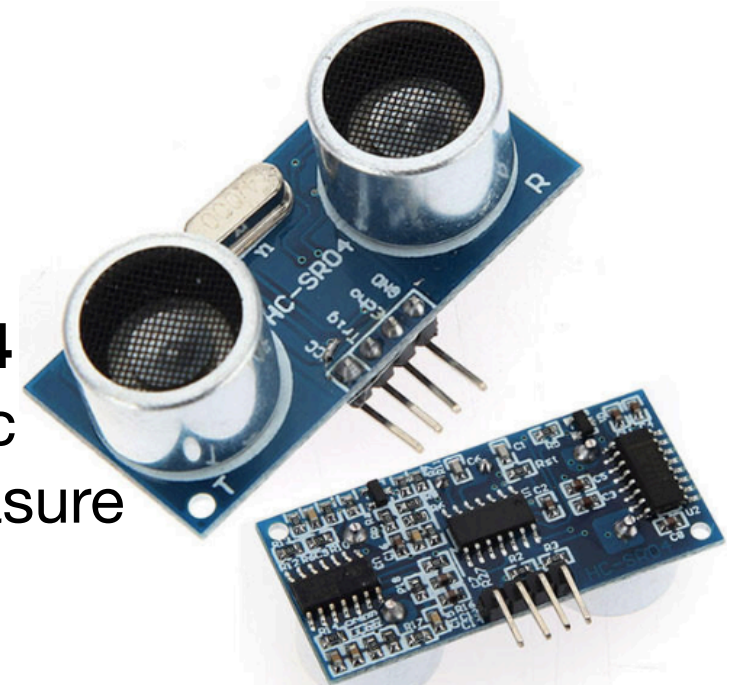
HD44780 I2C
LCD display



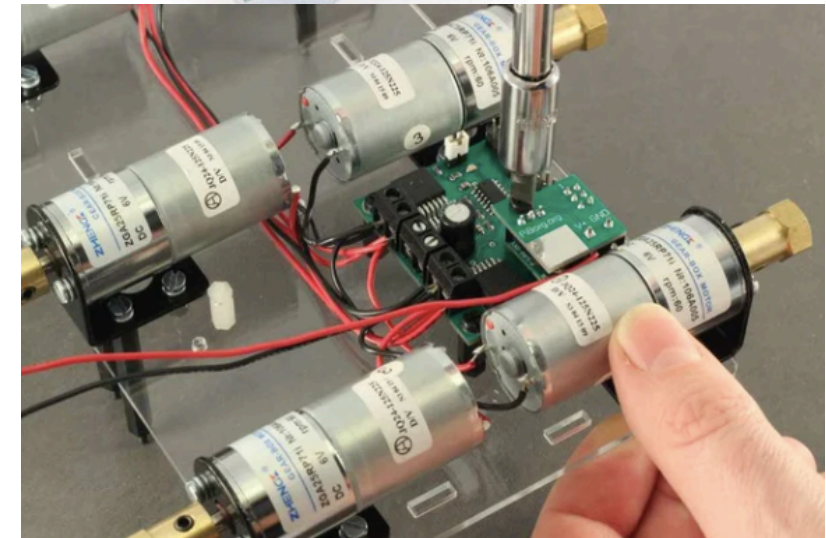
MPU6050
Gyroscope
Accelerometer
Temperature



HC-SR04
Ultrasonic
distance measure



PicoBorg
I2C motors



<https://github.com/oliveiraalex/PotHD44780Controller> (integrated in oficial repository)

<https://github.com/oliveiraalex/PotHCSR04> (integrated in oficial repository)

<https://github.com/oliveiraalex/PicoBorgReverseMotors>

<https://github.com/oliveiraalex/PotMPU6050Device>

PharoThings Booklet



<https://github.com/SquareBracketAssociates/Booklet-APharoThingsTutorial>

PharoThings Booklet

12.5 Creating the application

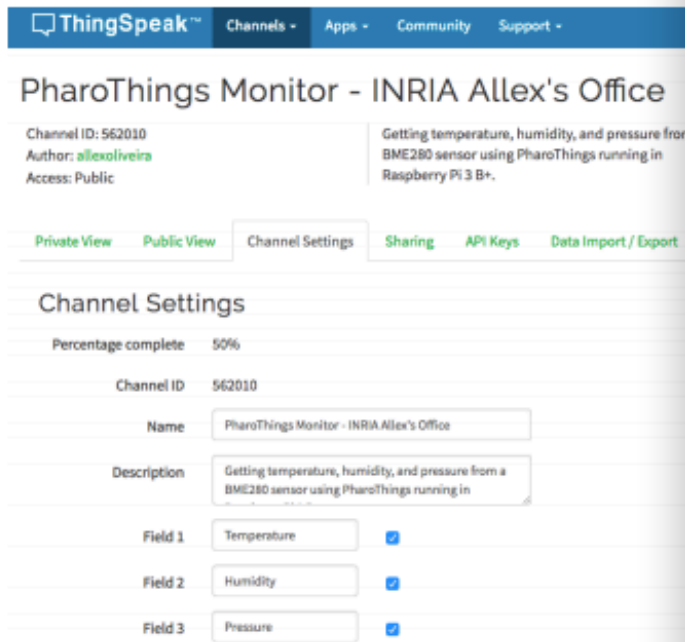


Figure 12-1 ThingSpeak Channel Configuration.

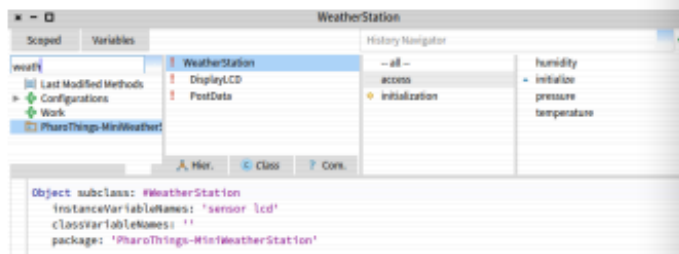


Figure 12-2 Mini Weather Station code.

play this information on the LCD and the second will send the data to cloud. Your final code will seem like the Picture 12-2.

4.9 Save your work

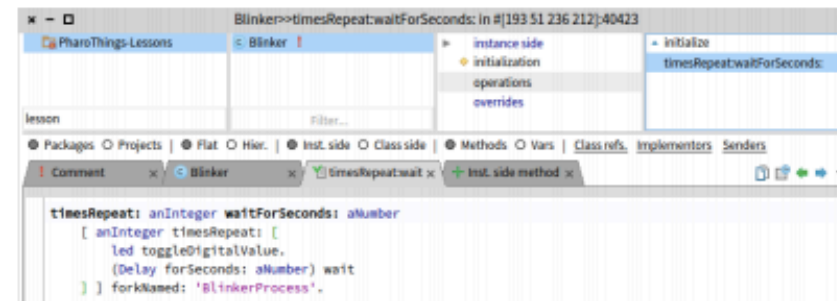


Figure 4-7 Creating an operation method.

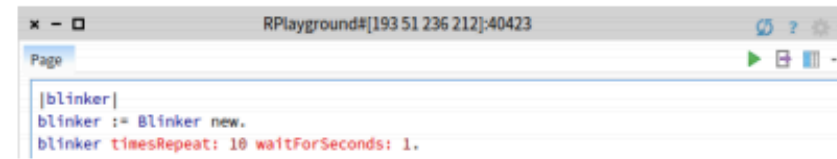


Figure 4-8 Remote playground.

```
|blinker|
blinker := Blinker new.
blinker timesRepeat: 10 waitForSeconds: 1.
```

Run this code, as shown in Figure 4-8 and... cool! Now your LED is blinking! And the better, you did this using object-oriented programming!

You do not need to change your code every time you wanna change these parameters. Just change the messages you send to the object and it will behave as you want.

4.9 Save your work

Don't forget to save your work remotely. To do this, run this command on your local playground:

```
[remotePharo saveImage.
```

Lesson 4 - LED Flowing Lights

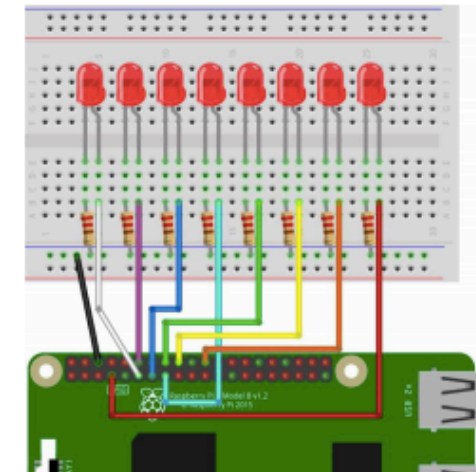


Figure 5-1 Schema connection 8 LEDs.

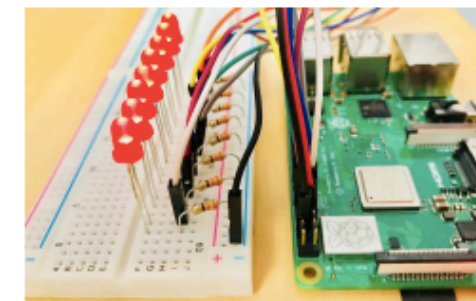


Figure 5-2 Physical connection 8 LEDs.

- Connect the Ground PIN from Raspberry in the breadboard blue rail (-).
- Then connect the 8 resistors from the blue rail (-) to a column on the breadboard, as shown below;
- Now push the LED legs into the breadboard, with the long leg (with the kink) on the right;
- And insert the jumper wires connecting the right column of each LED to GPIO from 0 to 7, as shown in the Picture 5-1.

Welcome Window PharoThings

The image shows two overlapping windows from the PharoThings environment. The background window is titled "Welcome" and displays a "PharoThings Quickstart guide". The foreground window is titled "Playground" and shows a list of commands being executed in a REPL.

Welcome Window Content:

- PharoThings Quickstart guide**
- Welcome to Pharo, an immersive live programming environment.
- This Pharo image already comes with PharoThings installed. PharoThings is a live programming platform for IoT projects based on Pharo.
- It includes:
 - Development tools to lively program, explore and debug remote boards (based on TelePharo)
 - Board modeling library which simplifies board configuration
- For more information, please visit here: <https://github.com/pharo-iot/PharoThings>
- Connecting in PharoThings server by IP**
- `remotePharo := TlpRemoteIDE connectTo: (TCPAddress ip: #[192 168 1 200]`
[Open In Playground](#)
- Connecting in PharoThings server by Hostname**
- `ip := NetNameResolver addressForName: 'pharothings-01'.`
`remotePharo := TlpRemoteIDE connectTo: (TCPAddress ip: ip po`
[Open In Playground](#)

Playground Window Content:

```
"Connecting in PharoThings server by IP"
remotePharo := TlpRemoteIDE connectTo: (TCPAddress ip: #[192 168 1 200]
port: 40423).

"Connecting in PharoThings server by Hostname"
ip := NetNameResolver addressForName: 'pharoiot-01'.
remotePharo := TlpRemoteIDE connectTo: (TCPAddress ip: ip port: 40423).

"Inspect remote board"
remoteBoard := remotePharo evaluate: [ RpiBoard3B current].
remoteBoard inspect.

"Open remote Playground, remote Browser and remote Process Browser"
remotePharo openPlayground.
remotePharo openBrowser.
remotePharo openProcessBrowser.
```

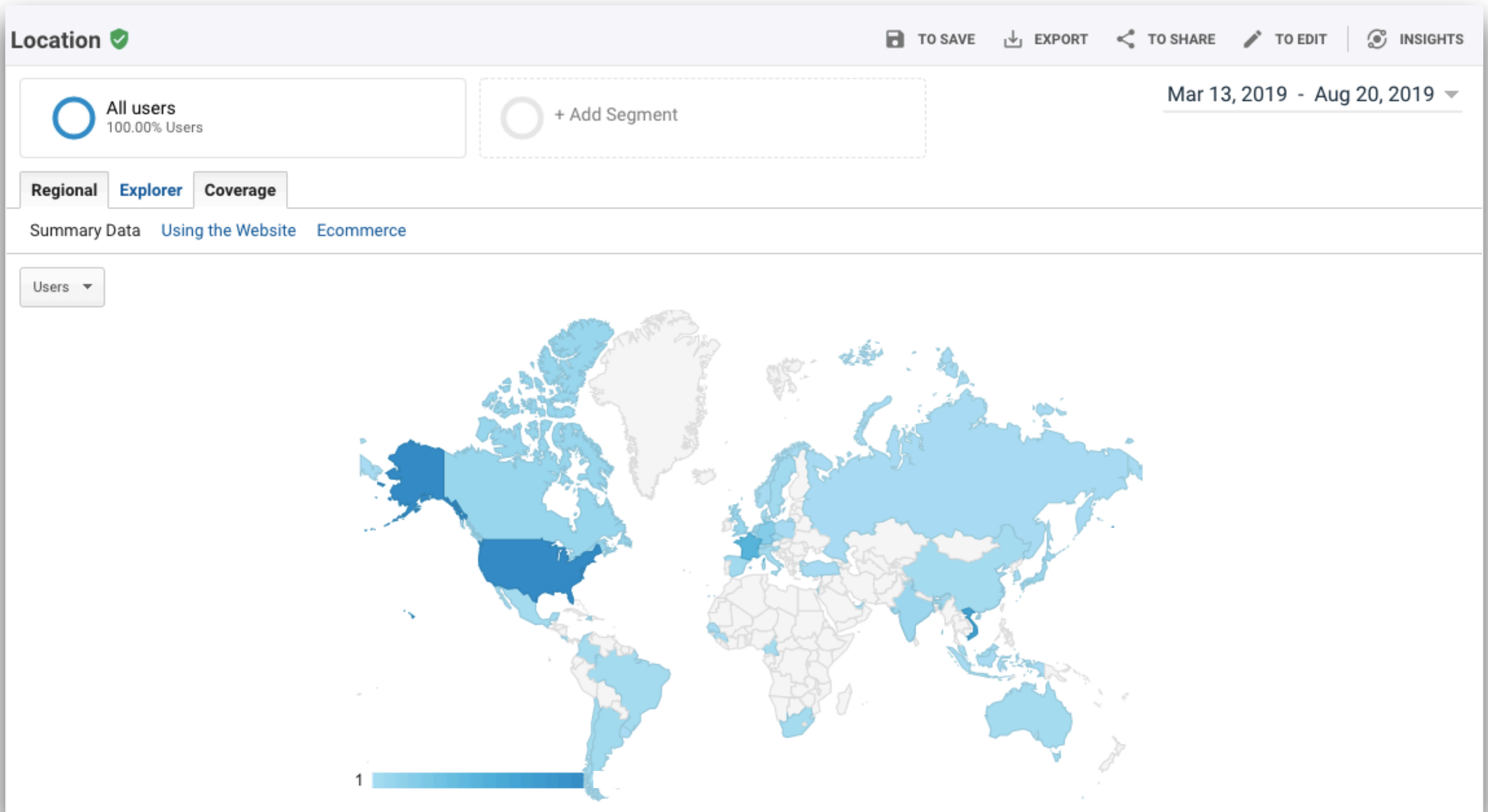
pharoiot.org

Everything in the same place, to facilitate the journey of the new user.

The screenshot shows the homepage of PHARO IoT. At the top left, the logo reads "PHARO IoT" with the tagline "Live programming platform" below it. To the right, a navigation menu includes "START HERE", "DOWNLOAD", "LESSONS", "PROJECT HUB", and "CONTACT". The main banner features a dark cityscape background with the headline "Unlock the power of Internet of Things" and the subtext "Start Pharo IoT's journey now and create IoT applications very quickly and easily". A prominent blue "START NOW" button is centered in the banner. Below the banner, the heading "Let's get right to the point" is followed by three white cards with blue icons and text:

- Install Raspberry Pi Runtime**: How to do a headless installation and set up your Raspberry.
- Install IDE on Mac Windows Linux**: 1 click to run Pharo IoT in Windows, Linux, Mac!
- Start PharoThings Lessons**: Learn how to create IoT applications quickly.

pharoiot.org



Continuous Integration

Travis CI on Pharo IoT

The image displays three overlapping screenshots from the GitHub repository `pharo-iot / Ci`.

- Top-left screenshot:** Shows the repository overview for `pharo-iot / Ci`. It indicates 32 commits, 1 branch, and 1 release. A commit by `oliveiraallex` titled "Travis upload release files" is highlighted.
- Top-right screenshot:** Shows a Travis CI build log for the commit `4ff58c2`. The build is successful, with 38 tests passed. The log includes steps like "Worker information", "Build system information", "Installing APT Packages", "Setting environment variables from repository settings", and "Skipping a deployment with the releases provider because this is a pre-release".
- Bottom-right screenshot:** Shows the "Releases" page for version `v0.1`. It lists three assets: `client.zip` (24.2 MB), `multi.zip` (78.5 MB), and `server.zip` (23.8 MB). The release was made by `oliveiraallex` 6 days ago.

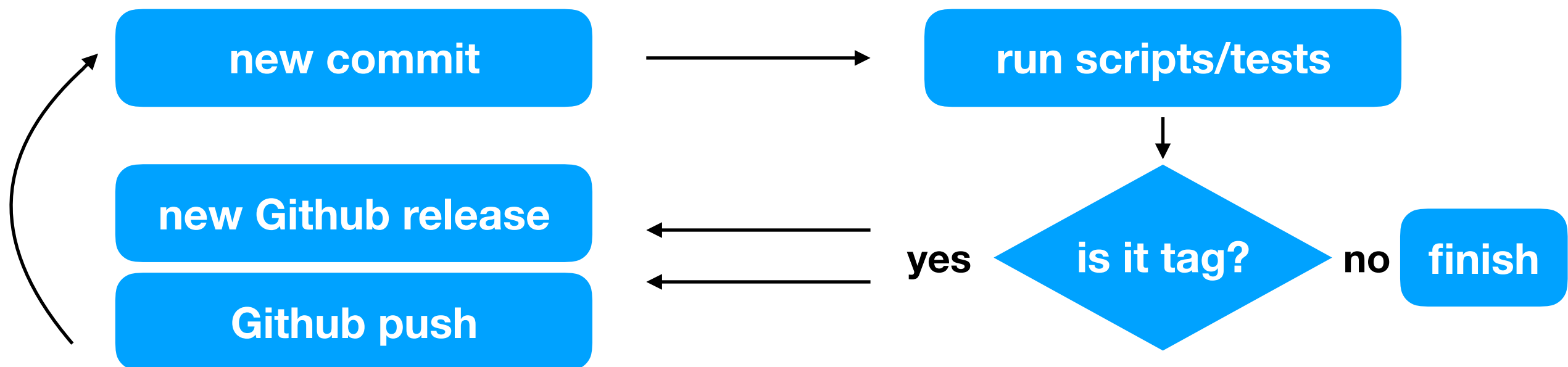
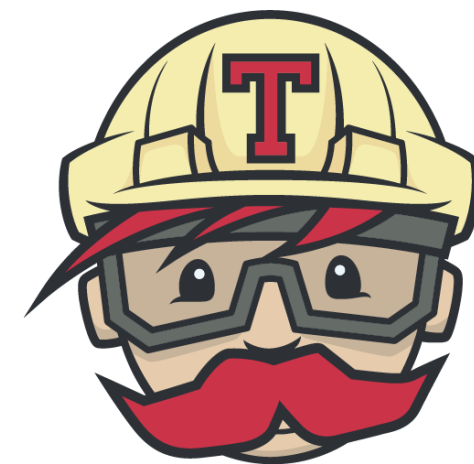
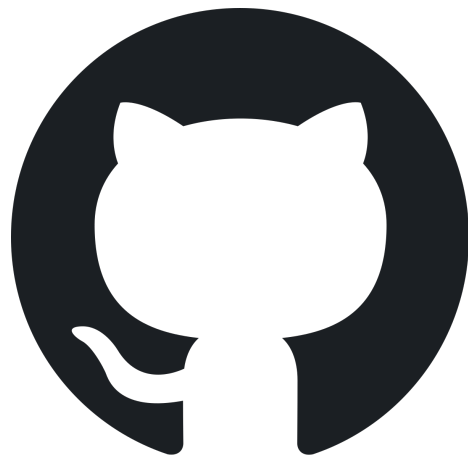
<https://github.com/pharo-iot/Ci>

Continuous Integration

Travis CI on Pharo IoT

github.com/pharo-iot/Ci/.travis.yml

travis-ci.org/pharo-iot/Ci



<https://github.com/pharo-iot/Ci>

Continuous Integration

<https://github.com/pharo-iot/Ci>

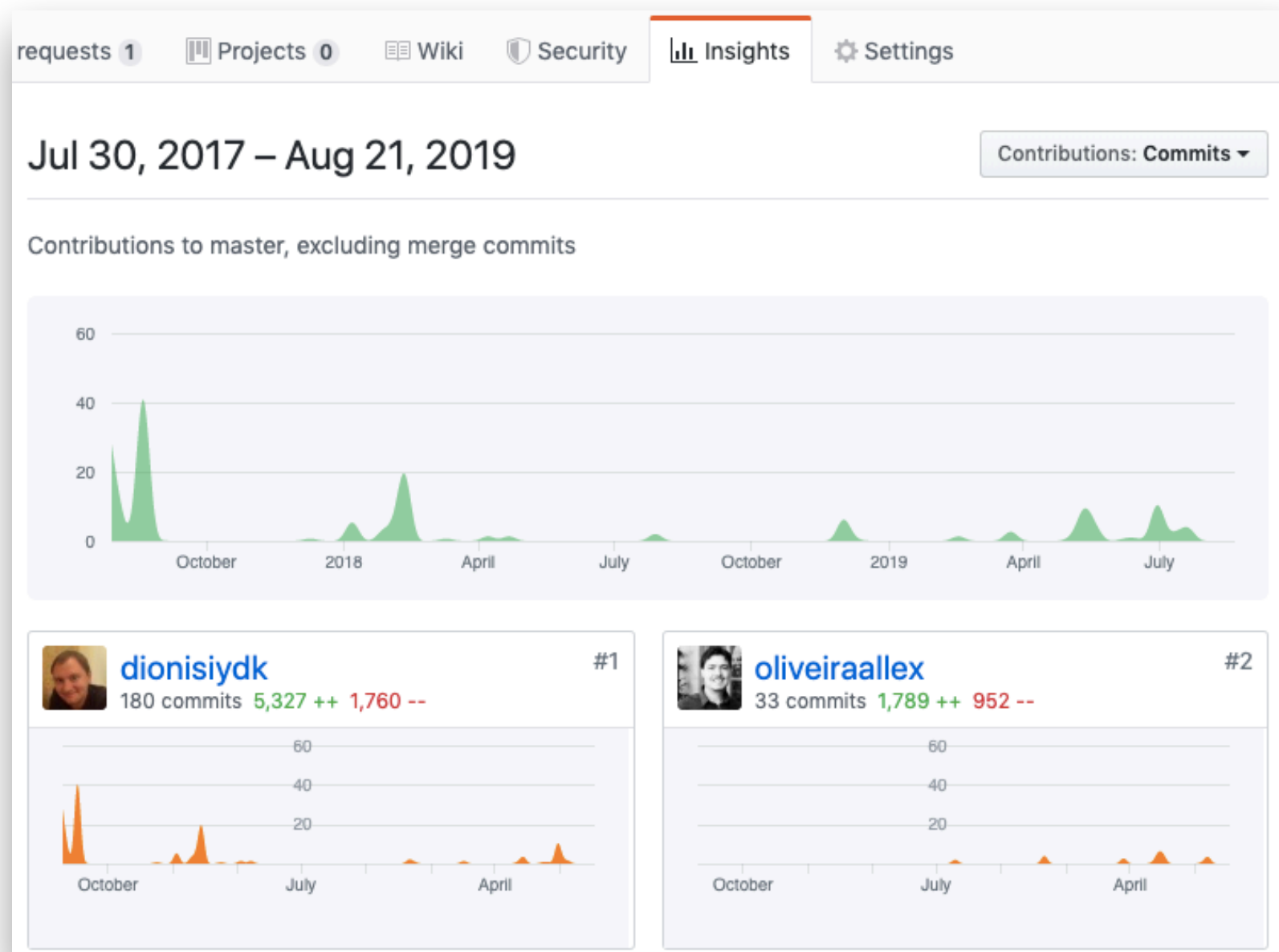
- Load PharoThings on image 32/64
- Download last VMs
- Create the 1 click-run files
- Zip everything
- Deploy
- Create PDF Booklet

2 - Collaborative work

- Denis Kudriashov (PharoThings improvements)
<https://github.com/dionisiydk>
- ZweiDenker (Minimal PharoThings image)
<https://github.com/noha/pharo-minimal>
- Bela IO, Jack Armitage (Pharo IoT on musical context)
<https://bela.io/>
- Serge Stinkwich (Pharo IoT lessons, Pharo IoT Booklet)
<https://github.com/SergeStinckwich>
- Do Hoang, Vietnam (Pharo IoT Booklet - Arduino Chapter)
<https://github.com/huyhoang8398>

PharoThings improvements

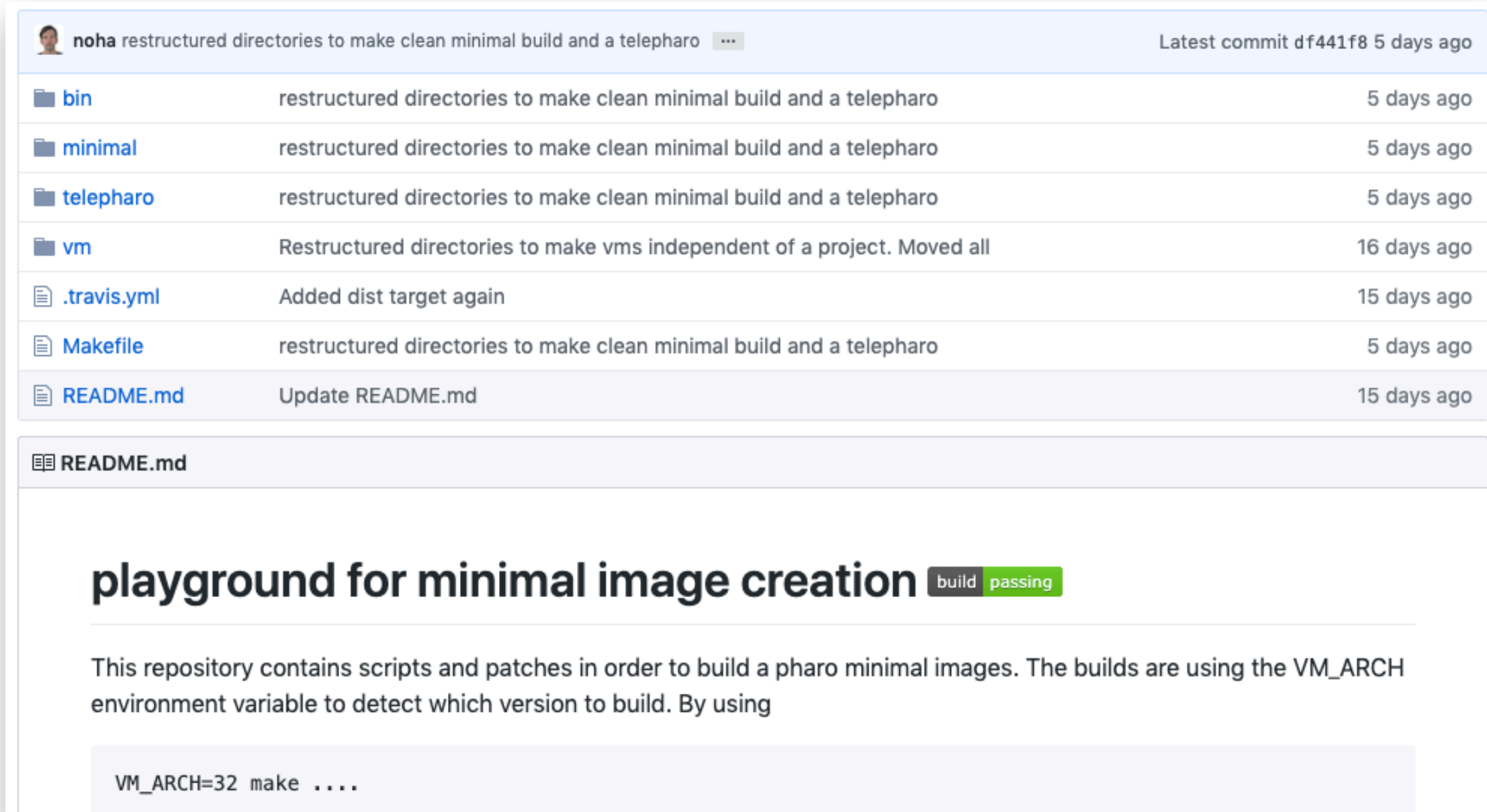
- Denis Kudriashov (Russia)



<https://github.com/dionisiydk>

Minimal PharoThings image

- Norbert Hartl (Germany)



The screenshot shows a GitHub repository page for 'noha' with the commit message 'restructured directories to make clean minimal build and a telepharo'. The repository contains several files and folders, including 'bin', 'minimal', 'telepharo', 'vm', '.travis.yml', 'Makefile', and 'README.md'. The 'README.md' file is expanded, showing the title 'playground for minimal image creation' with a 'build passing' status. The text in the README describes the repository's purpose: 'This repository contains scripts and patches in order to build a pharo minimal images. The builds are using the VM_ARCH environment variable to detect which version to build. By using' followed by a code block containing 'VM_ARCH=32 make'.

File/Folder	Description	Time
bin	restructured directories to make clean minimal build and a telepharo	5 days ago
minimal	restructured directories to make clean minimal build and a telepharo	5 days ago
telepharo	restructured directories to make clean minimal build and a telepharo	5 days ago
vm	Restructured directories to make vms independent of a project. Moved all	16 days ago
.travis.yml	Added dist target again	15 days ago
Makefile	restructured directories to make clean minimal build and a telepharo	5 days ago
README.md	Update README.md	15 days ago

playground for minimal image creation build passing

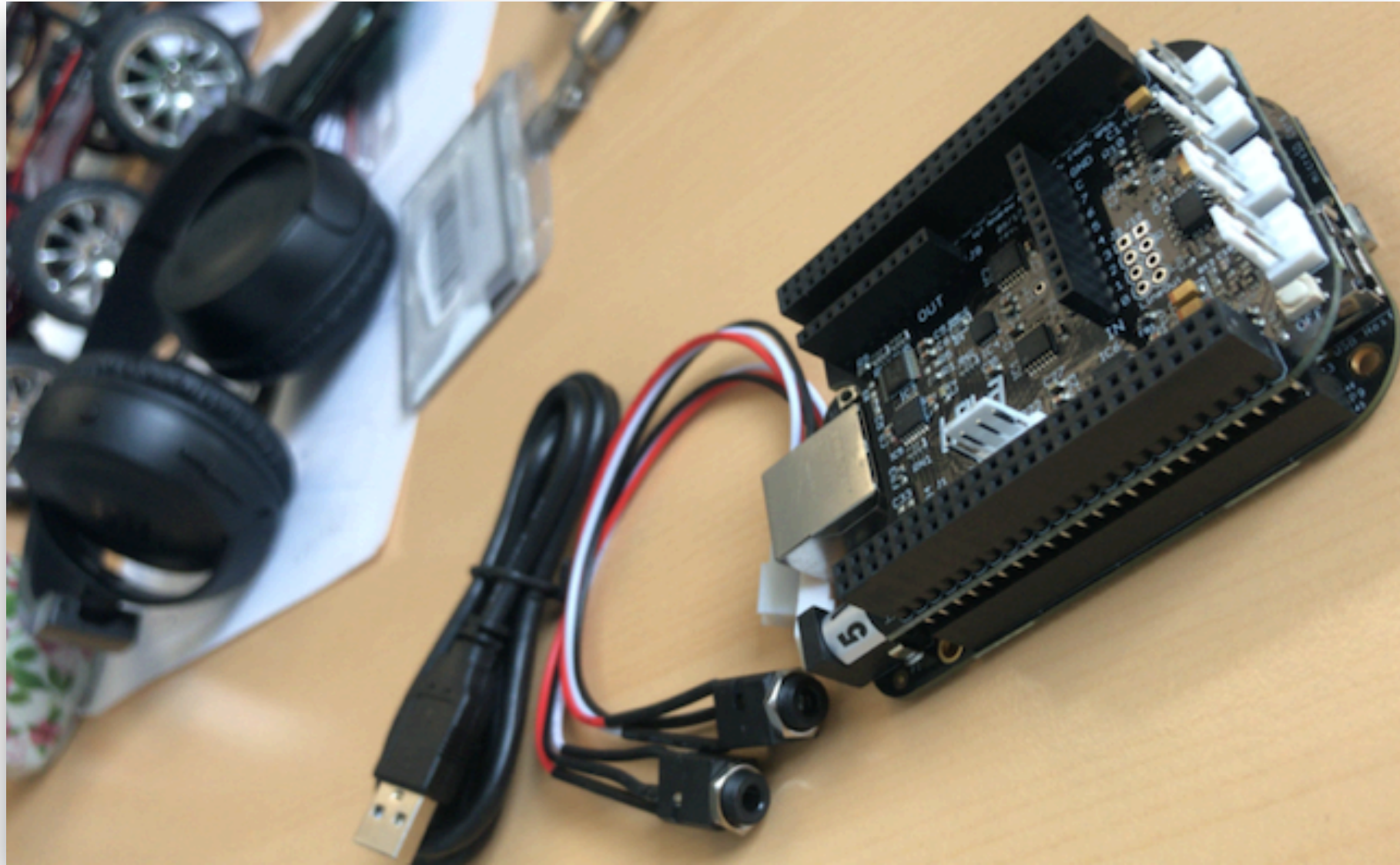
This repository contains scripts and patches in order to build a pharo minimal images. The builds are using the VM_ARCH environment variable to detect which version to build. By using

```
VM_ARCH=32 make ....
```

<https://github.com/noha/pharo-minimal>

Bela IO - Beaglebone

- Jack Armitage (UK)



<https://bela.io/>

Pharo IoT workshops and booklet collaborations

- Serge Stinckwich (France)



<https://twitter.com/sergestinckwich>

Pharo IoT Booklet

Arduino Chapter

- Do Hoang (Vietnam)

The screenshot shows a GitHub repository page for 'huyhoang8398 / Booklet-APharoThingsTutorial', which is a fork of 'SquareBracketAssociates/Booklet-APharoThingsTutorial'. The repository has 0 watches, 1 star, and 4 forks. The main navigation includes 'Code', 'Pull requests 0', 'Projects 0', 'Wiki', 'Security', and 'Insights'. The current branch is 'master'. Under the 'Commits on May 21, 2019' section, there are two commit entries, both titled 'init firmata chapter' and authored/committed by 'Do Duy Huy Hoang' on May 21, 2019. The first commit has a hash of 'da534cd' and the second has '272b584'. Each commit entry includes a copy icon, the hash, and a code icon.

<https://github.com/huyhoang8398/Booklet-APharoThingsTutorial>

3 - Projects using Pharo IoT

- Coffee Machine IoT
- Autonomous Robot
- Door opener

Coffee Machine IoT

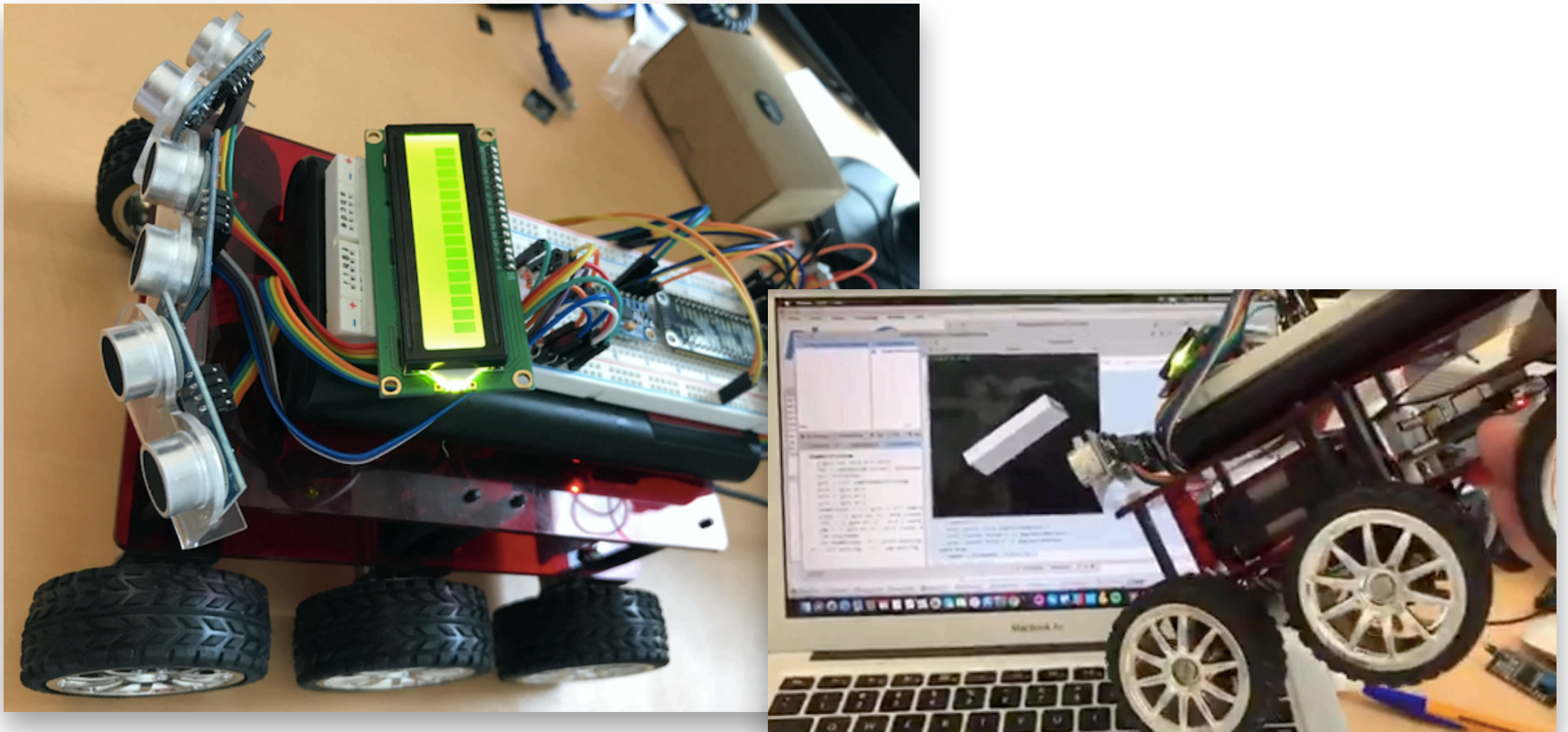
- Alex Oliveira



<https://github.com/oliveiraalex/CoffeeMachine>

Autonomous robot

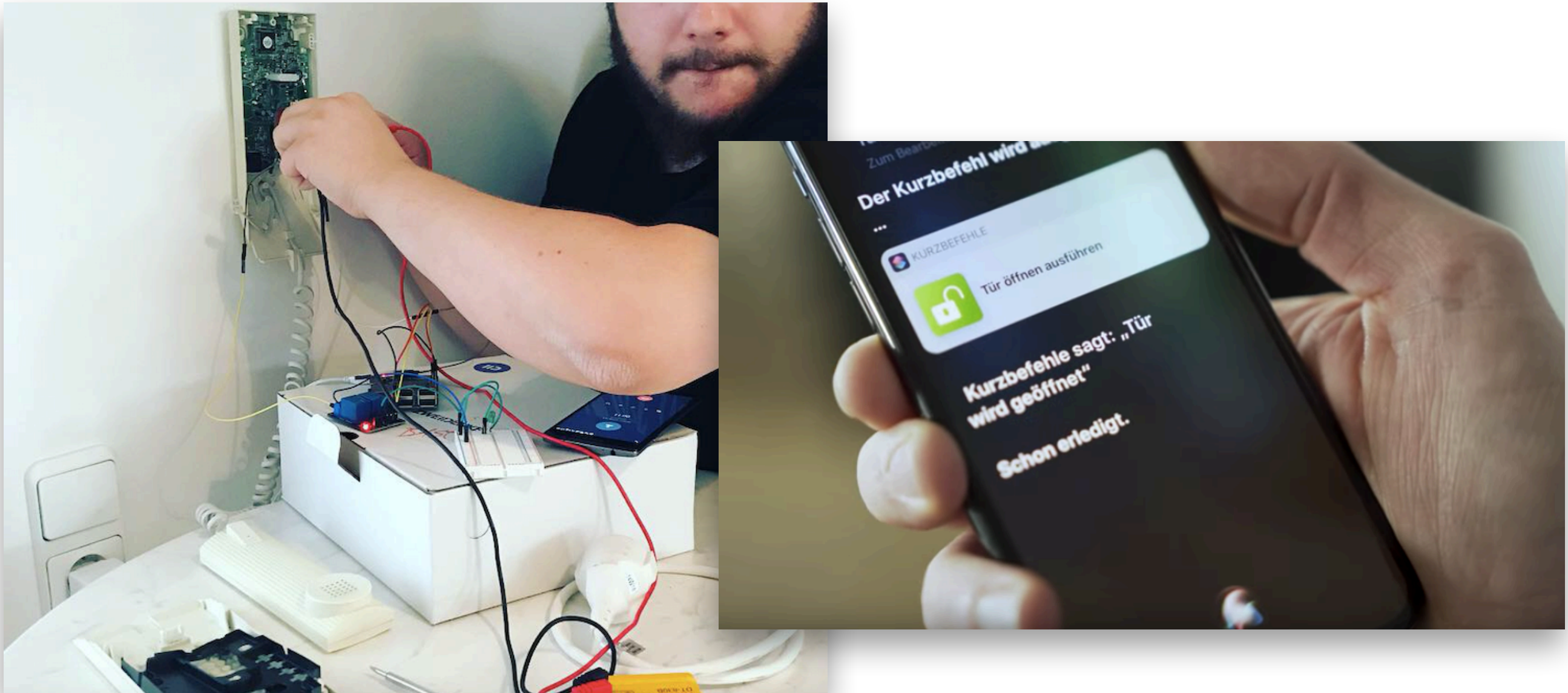
- Alex Oliveira and Steven Costiou



<https://www.youtube.com/watch?v=1j5WSCkIiKk>

Door opener

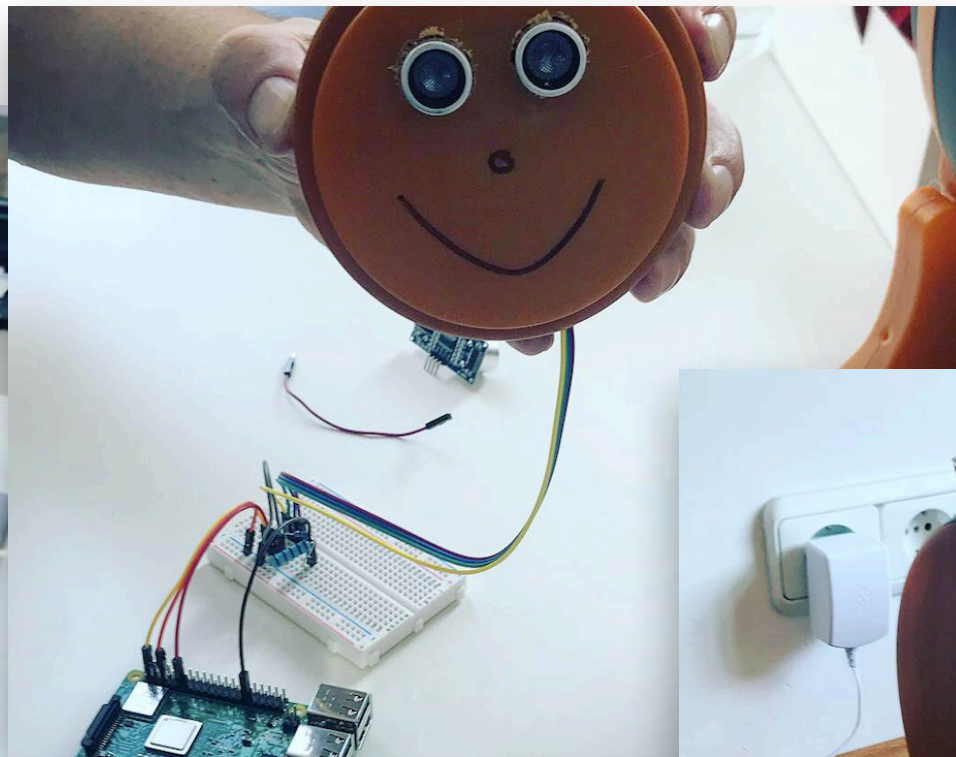
- ZweiDenker workshop



<https://www.youtube.com/watch?v=dII9FAatKyw>

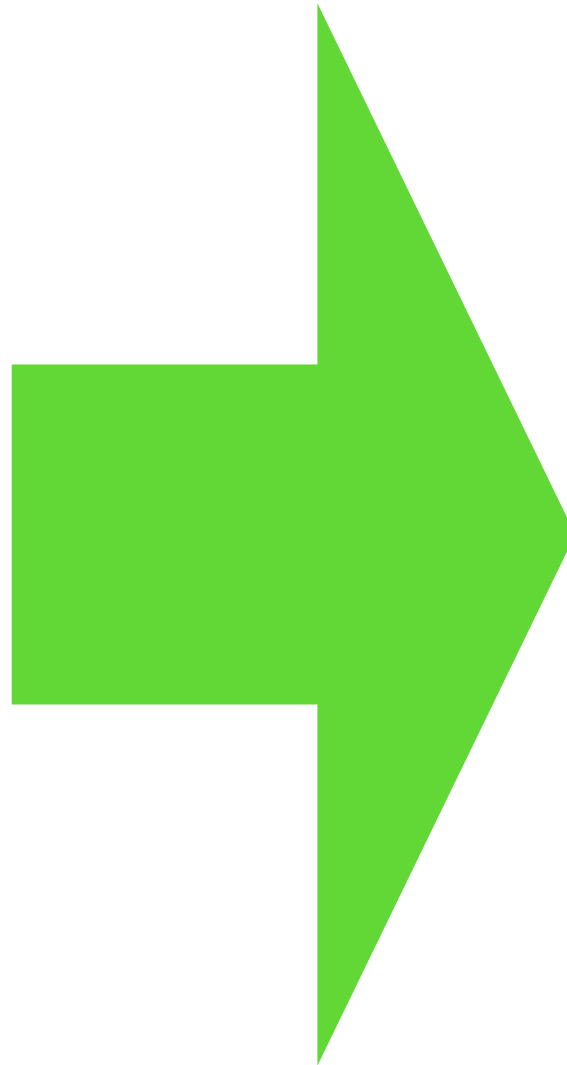
Wine measurer

- ZweiDenker workshop



<https://www.youtube.com/watch?v=dll9FAatKyw>

4 - Future



Booklet

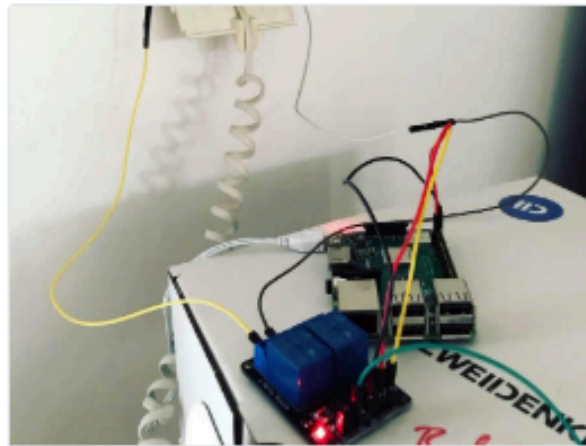
Creating drivers and application with Pharo IoT



pharoiot.org

- Share your IoT projects using Pharo with the community!

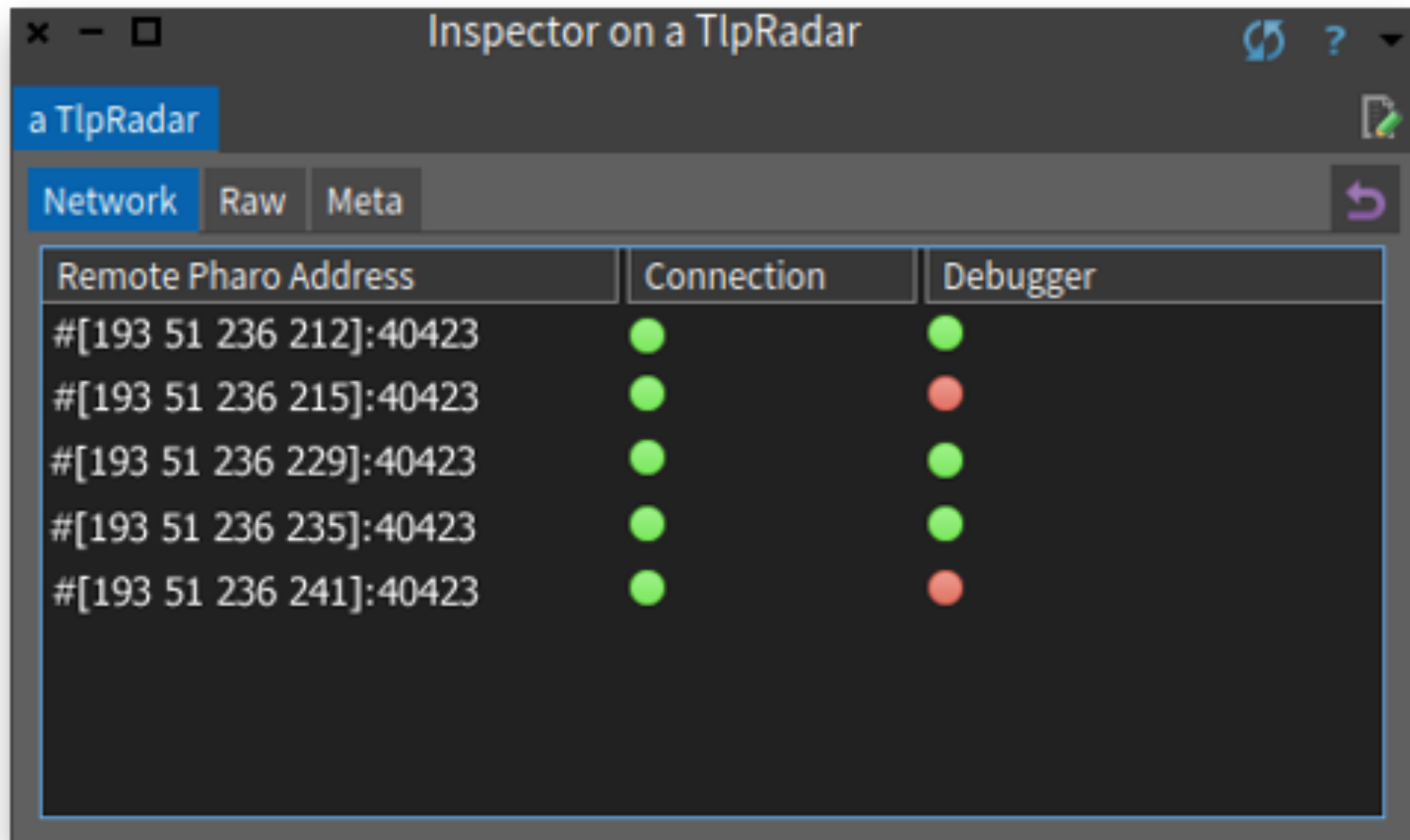
Community projects



[SUBMIT YOUR PROJECT](#)

Tele Radar

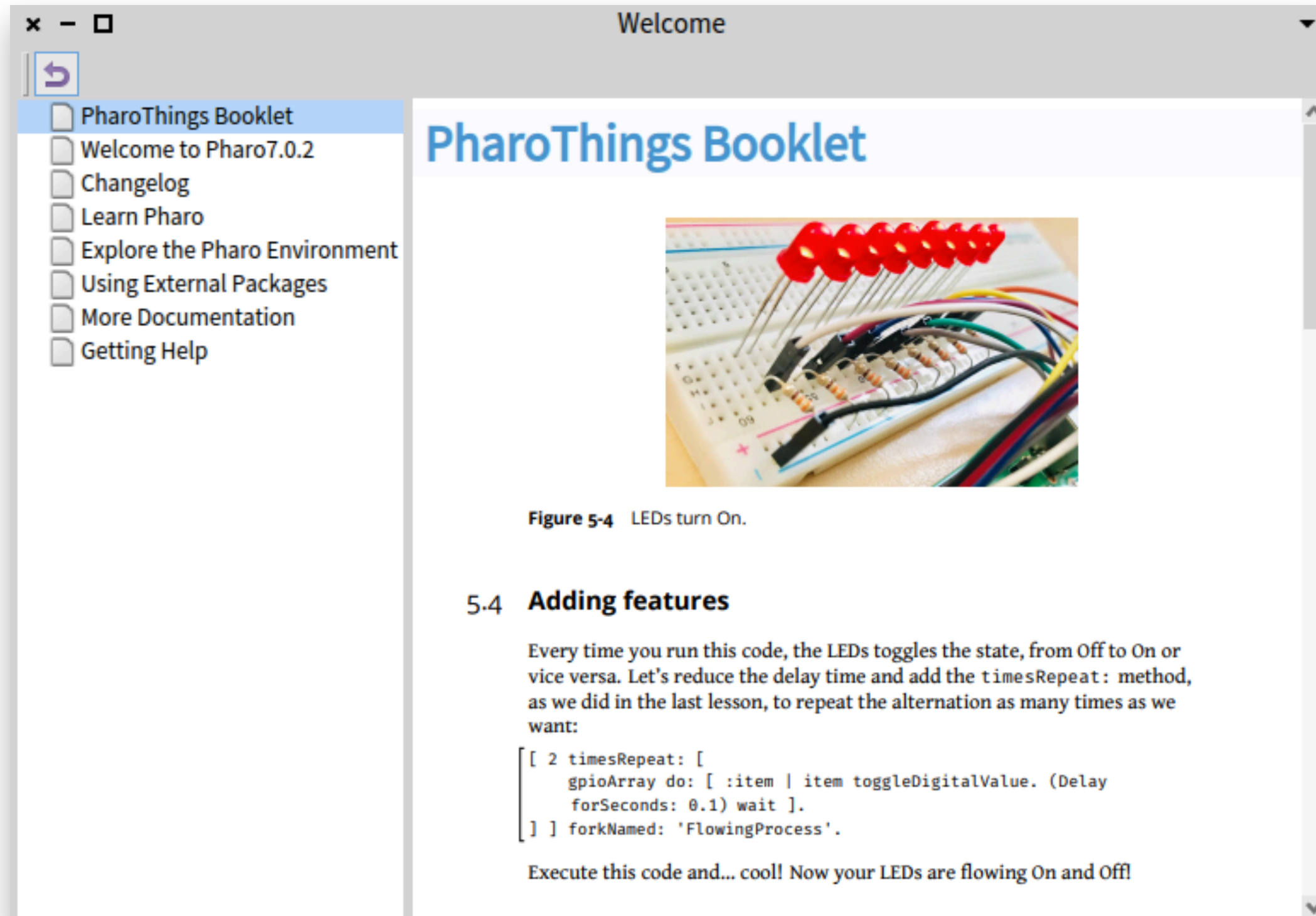
Automatic detection of running images in network
(TeleRadar using SSDP protocol)



The screenshot shows a window titled "Inspector on a TlpRadar" with a tab labeled "a TlpRadar". Below the tab are three sub-tabs: "Network" (selected), "Raw", and "Meta". A table displays the following data:

Remote Pharo Address	Connection	Debugger
#[193 51 236 212]:40423	●	●
#[193 51 236 215]:40423	●	●
#[193 51 236 229]:40423	●	●
#[193 51 236 235]:40423	●	●
#[193 51 236 241]:40423	●	●

PharoThings Booklet inside Pharo



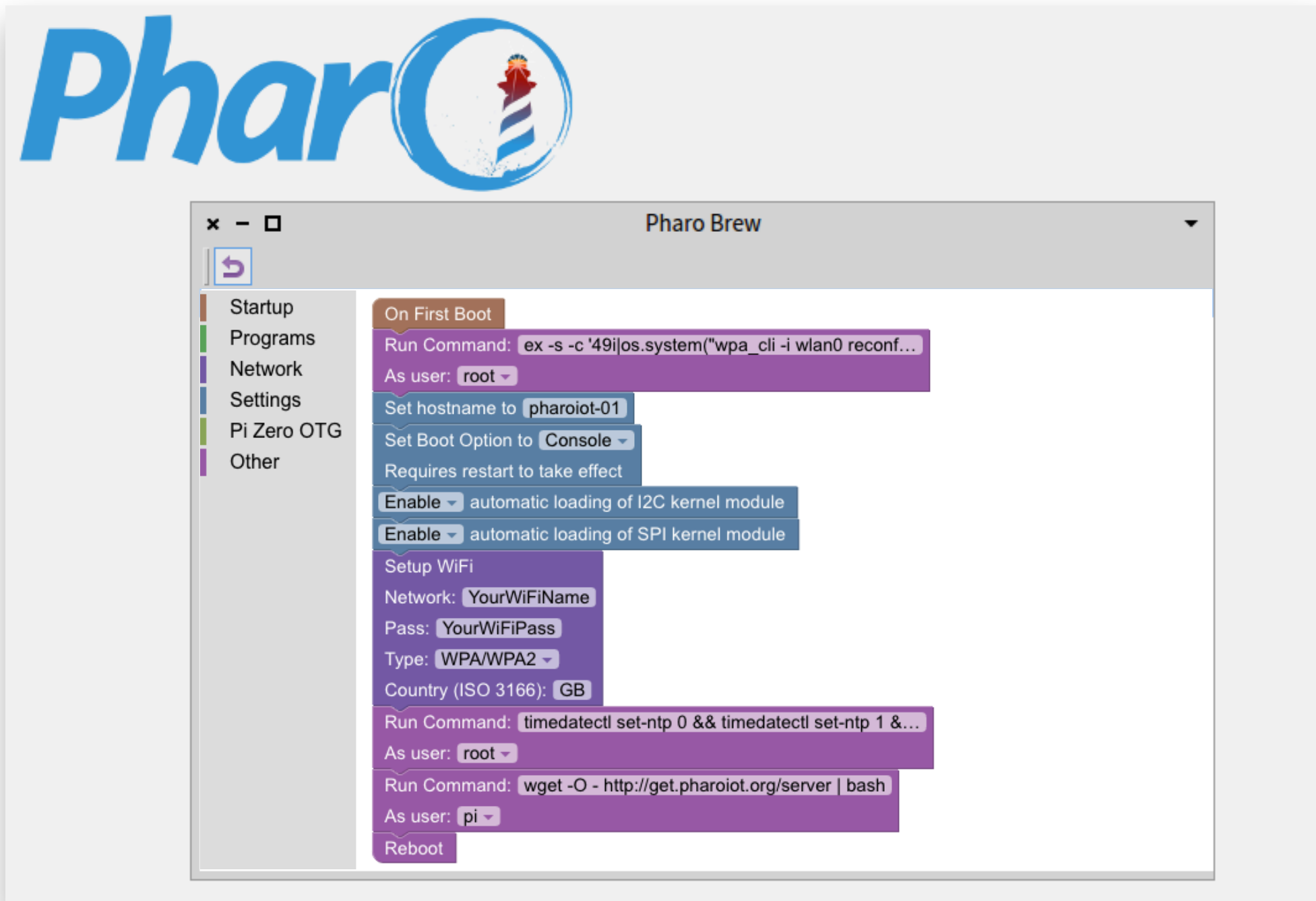
The screenshot shows a Pharo IDE window titled "Welcome". On the left is a sidebar with a navigation menu containing the following items: "PharoThings Booklet" (highlighted), "Welcome to Pharo7.0.2", "Changelog", "Learn Pharo", "Explore the Pharo Environment", "Using External Packages", "More Documentation", and "Getting Help". The main content area displays the "PharoThings Booklet" page, which features a blue header with the title "PharoThings Booklet". Below the header is a photograph of a breadboard with several red LEDs connected to a circuit. The caption below the image reads "Figure 5-4 LEDs turn On." Below the image is a section titled "5.4 Adding features". The text in this section explains that the code toggles the state of the LEDs and includes a code block for a process that repeats the toggle action. The code block is as follows:

```
[ 2 timesRepeat: [
  gpioArray do: [ :item | item toggleDigitalValue. (Delay
    forSeconds: 0.1) wait ].
] ] forkNamed: 'FlowingProcess'.
```

Execute this code and... cool! Now your LEDs are flowing On and Off!

Tool to “brew” SD Cards

- “brew” a new SD Card to inside Pharo (like PiBakery)



With Pharo IoT you can

- Dynamically update your running board
- Interact remotely with pins and boards
- Modify the system while it is running (create new board, change code)
- Make your changes persistent

get.pharoiot.org

**NOW IN LESS
THAN 1 MINUTE!**

THANKS!



Any questions?

allex.oliveira@msn.com

Presentation Information

This slides was presented at ESUG 2019, Cologne, Germany
<https://esug.github.io/2019-Conference/conf2019.html>

- Title: Pharo IoT - Present and Future
- Presenters:
 - Marcus Denker - marcus.denker@inria.fr
 - Norbert Hartl - norbert@2denker.de
 - Allex Oliveira - [linkedin.com/in/allex-oliveira](https://www.linkedin.com/in/allex-oliveira)

INRIA

<https://www.inria.fr/>

RMOD TEAM

<https://rmod.inria.fr/web>

PHARO PROJECT

<https://github.com/pharo-project/pharo>

PHAROTHOINGS PROJECT

<https://github.com/pharo-iot/PharoThings>

PHARO IoT

<http://get.pharoiot.org>