



PHARO IoT

Installation Improvements and
Continuous Integration

Alex Oliveira

IoT Engineer @ RMoD Team, INRIA

alex.oliveira@msn.com

Summary

- 1 – Getting experience
- 2 – Making the process easy
- 3 – Continuous Integration
- 4 – The 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)

What is Pharo IoT?



TelePharo
Remote
Communication



Inspector on a PotRemoteBoard (a RpiBoard3I)

a PotRemoteBoard (a RpiBoard3B in 192.168.1.14:40423)

P1	Devices	Raw	Meta
		3.3v	1
2		SDA (I2C)	3
3		SCL (I2C)	5
4		GPIO7	7
		Ground (0v)	9
17		GPIO0	11
27		GPIO2	13
22		GPIO3	15
		3.3v	17
10		MOSI (SPI)	19
9		MISO (SPI)	21
11		SCLK (SPI)	23
		Ground (0v)	25
0		SDA (I2C)	27
5		GPIO21	29
6		GPIO22	31

"a PotBoardConnector(P1): gpio0..gpio27 vars are bound to pins"
self

Firmata



Motivation

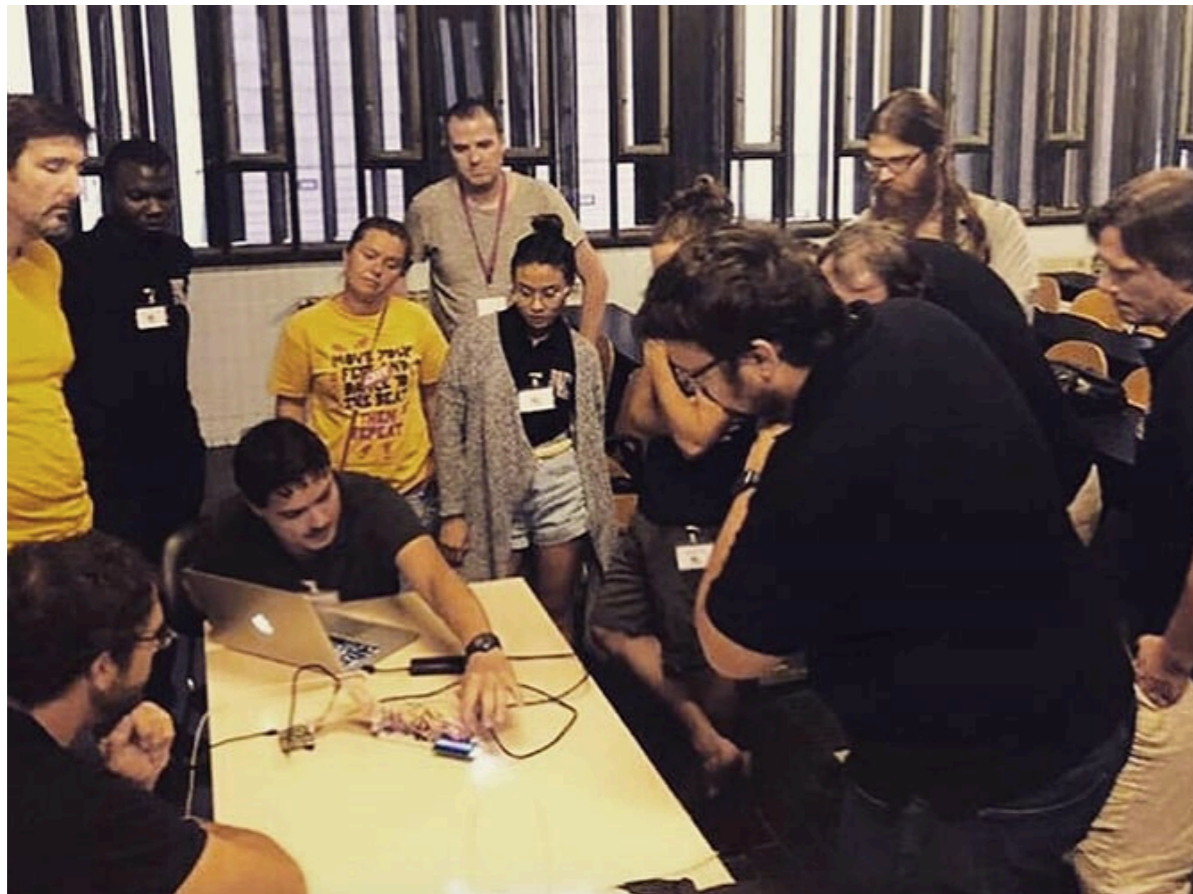
- It's not very easy to get started with PharoThings for anyone who is not a Pharo/Smalltalk user.
- How can we automate the process to make the developer's life easier?

Before we automate something, we should ask:

How can we improve it?

How to know what to improve?

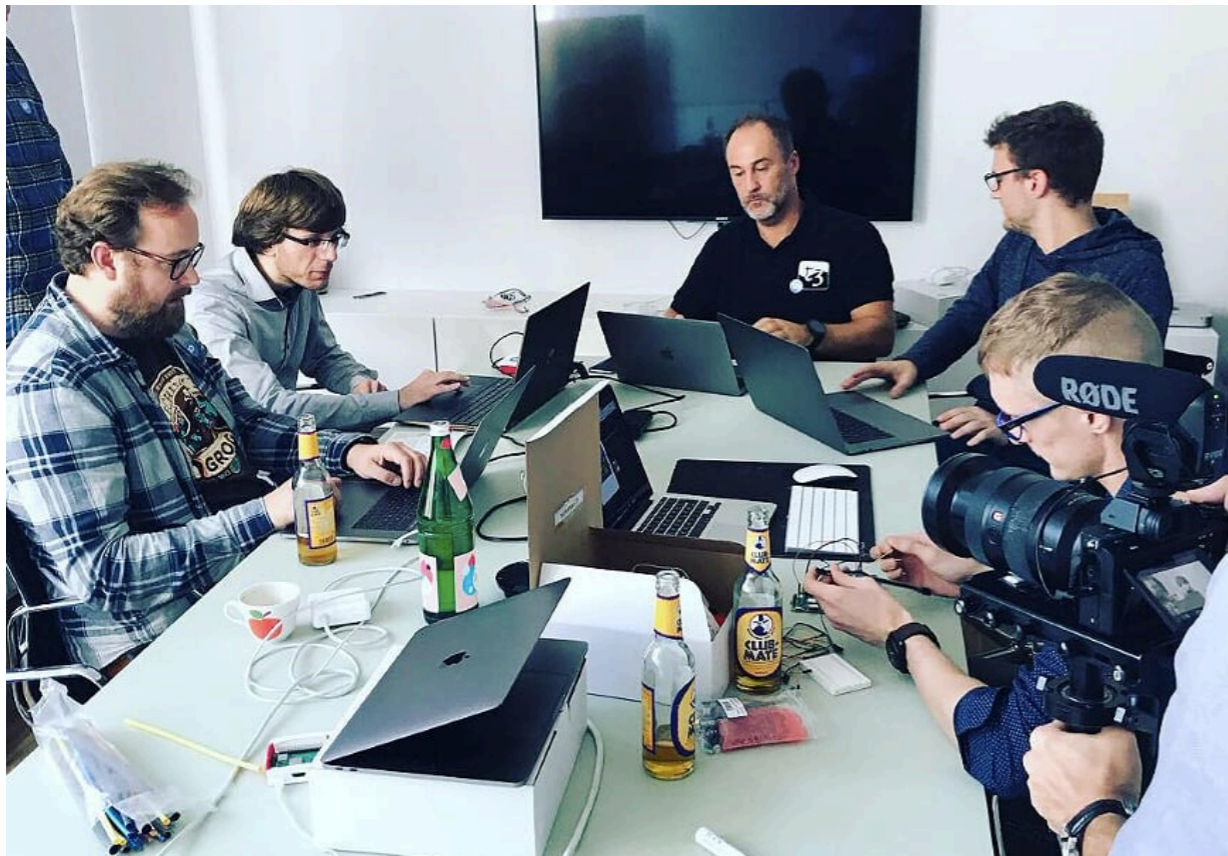
Real world experience



IoT Workshop, ESUG
September 2018 - Cagliari, Italy

How to know what to improve?

Real world experience



IoT Hackaton, Zweidenker GmbH
October 2018 - Cologne, Germany

How to know what to improve?

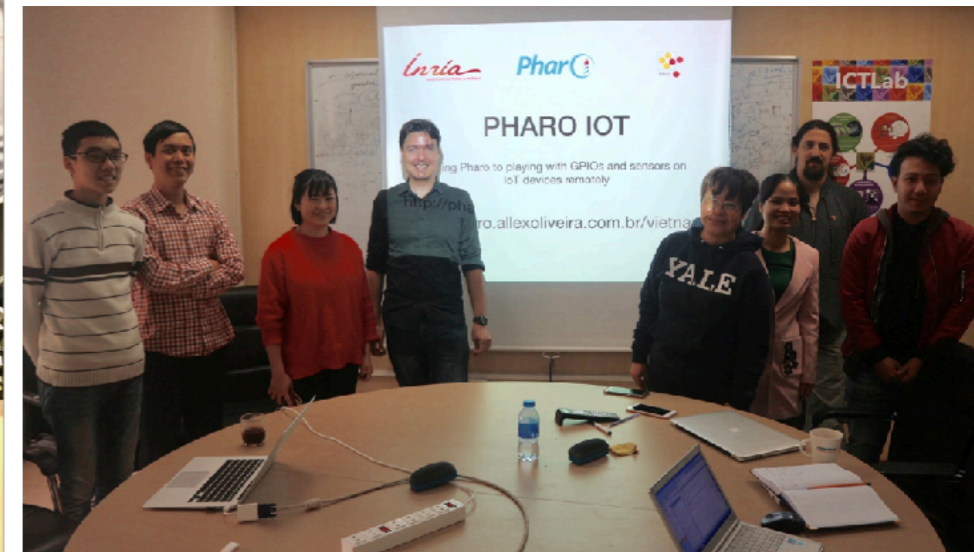
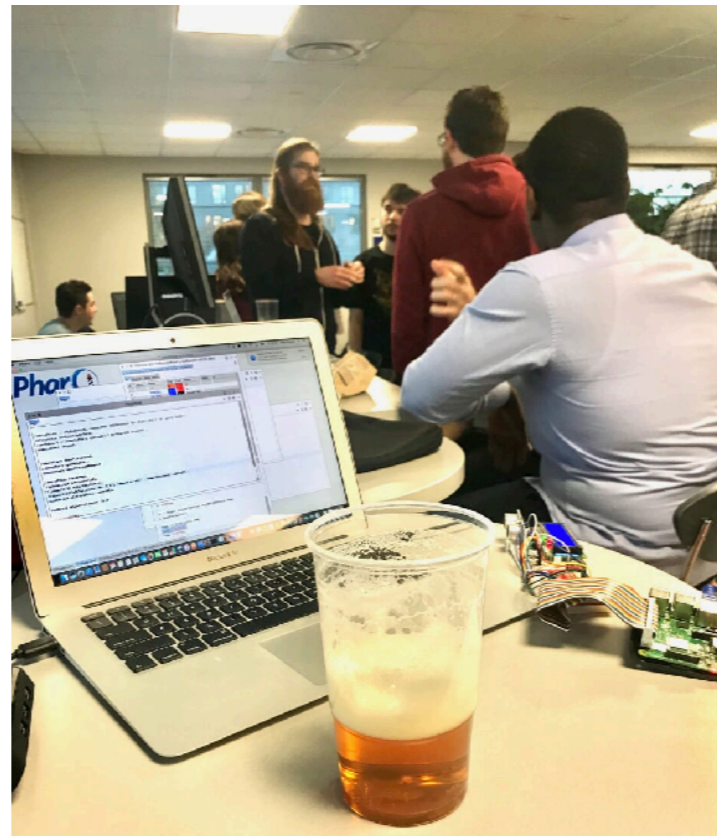
Real world experience



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

How to know what to improve?

Real world experience



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

2 - Difficulties

1. Is not so easy to start (many steps needed to run it);
2. Takes a long time to start (get everything);
3. Download VMs to different OS (ARM, Linux, Win, OSX);
4. Need the same version on server/client;
5. Needs a keyboard/screen on Raspberry to install from scratch;
6. No many application examples;
7. Search example code: copy and paste (pdf bad formatting);

2 - Difficulties

1. Is not so easy to start (many steps needed to run it);
2. Takes a long time to start (get everything);

How to install (manually)

1

2

3

4

5

6

1. Install PharoLauncher and create a new image
2. Load PharoThings project from Github
3. Copy ARM VM and Pharo Image to Raspberry Pi
4. Set the run permissions "chmod +x"
5. Run TelePharo server in a big command line
6. Extra configurations: set hostname, WiFi, autorun

How to install (manually)

1

2

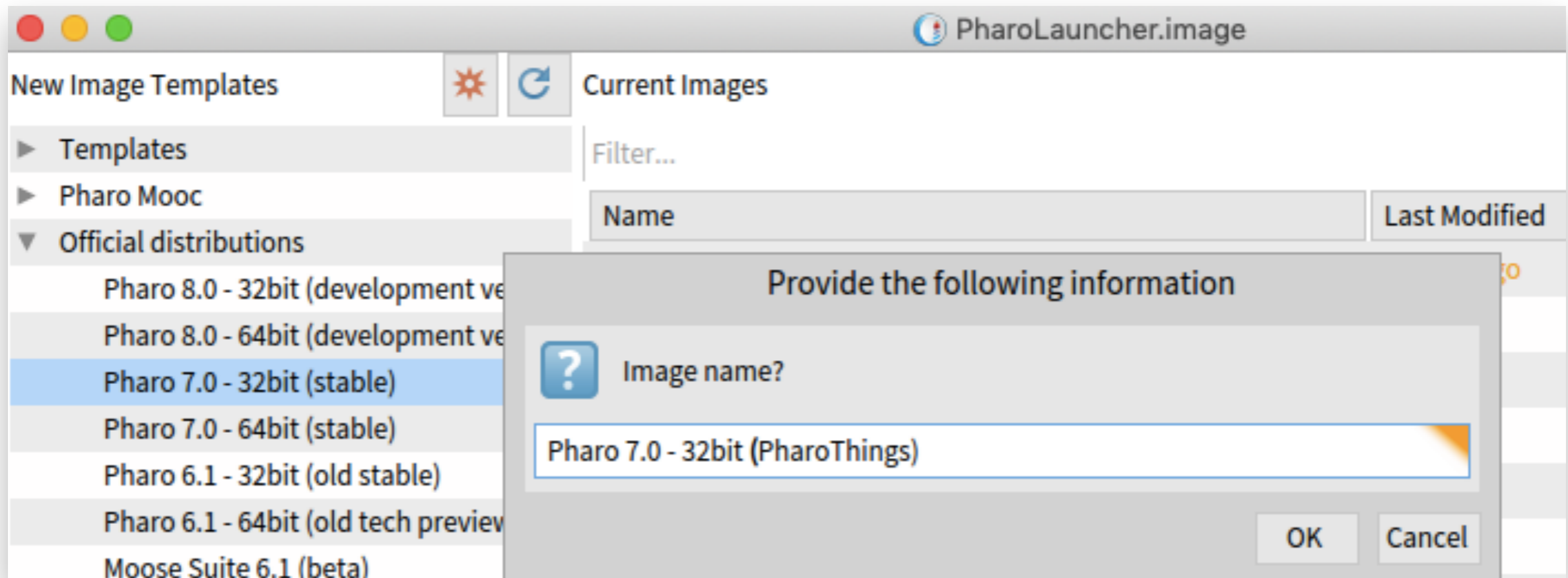
3

4

5

6

1. Install PharoLauncher and create a new image



How to install (manually)

1

2

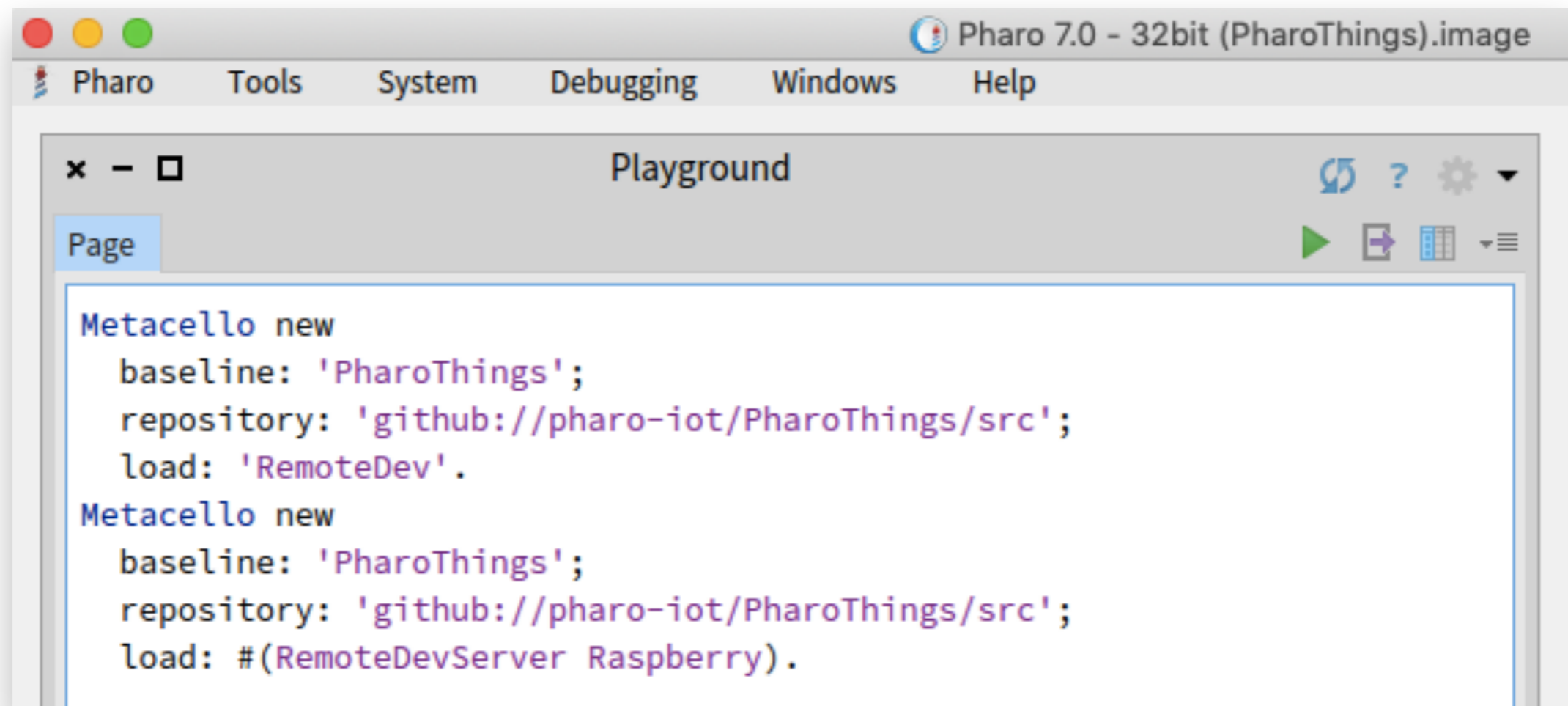
3

4

5

6

2. Load PharoThings project from Github



The screenshot shows a Pharo 7.0 - 32bit (PharoThings).image window. The menu bar includes Pharo, Tools, System, Debugging, Windows, and Help. The Playground window is open, showing a code editor with the following Metacello code:

```
Metacello new
  baseline: 'PharoThings';
  repository: 'github://pharo-iot/PharoThings/src';
  load: 'RemoteDev'.
Metacello new
  baseline: 'PharoThings';
  repository: 'github://pharo-iot/PharoThings/src';
  load: #(RemoteDevServer Raspberry).
```

How to install (manually)

1

2

3

4

5

6

3. Copy ARM VM and Pharo Image to Raspberry Pi

The image shows a Mac file manager window for a folder named 'pharothings' and a terminal window on a Raspberry Pi. The Mac window displays a directory listing with columns for Name, Size, and Kind. The Raspberry Pi terminal window shows the output of the 'ls -l' command, listing files and directories with their permissions, sizes, and dates.

Name	Size	Kind
bin	--	Folder
pharo	2 KB	Unix executable
lib	--	Folder
pharo		
5.0-201804182009		
meta-inf.ston		
pharo		
pharo-local		
pharo.version		
Pharo32.changes		
Pharo32.image		
PharoDebug.log		

```
pi@pharoiot-01: ~/pharothings — -ssh pi@pharoiot-01...
pi@pharoiot-01:~/pharothings $ ls -l
total 51668
-rw-r--r-- 1 pi pi 8789682 May 14 2018 Pharo32.changes
-rw-r--r-- 1 pi pi 44062856 May 14 2018 Pharo32.image
-rw-r--r-- 1 pi pi 28333 May 14 2018 PharoDebug.log
[drwxr-xr-x 2 pi pi 4096 May 30 2018 bin ]
[drwxr-xr-x 3 pi pi 4096 May 30 2018 lib ]
-rw-r--r-- 1 pi pi 163 May 14 2018 meta-inf.ston ]
-rw-rw-rw- 1 pi pi 2062 Apr 18 2018 pharo ]
[drwxr-xr-x 6 pi pi 4096 May 30 2018 pharo-local ]
-rw-r--r-- 1 pi pi 2 May 14 2018 pharo.version ]
```


How to install (manually)

1

2

3

4

5

6

4. Set the run permissions "chmod +x"

```
pi@pharoiot-01: ~/pharothings — -ssh pi@pharoiot-01 — 80x24
[pi@pharoiot-01:~ $ cd pharothings/
[pi@pharoiot-01:~/pharothings $ chmod +x pharo
[pi@pharoiot-01:~/pharothings $ chmod +x lib/pharo/5.0-201804182009/pharo
[pi@pharoiot-01:~/pharothings $ ls -l
total 51668
-rw-r--r-- 1 pi pi 8789682 May 14 2018 Pharo32.changes
-rw-r--r-- 1 pi pi 44062856 May 14 2018 Pharo32.image
-rw-r--r-- 1 pi pi 28333 May 14 2018 PharoDebug.log
drwxr-xr-x 2 pi pi 4096 May 30 2018 bin
drwxr-xr-x 3 pi pi 4096 May 30 2018 lib
-rw-r--r-- 1 pi pi 163 May 14 2018 meta-inf.ston
-rwxr-xr-x 1 pi pi 2062 Apr 18 2018 pharo
drwxr-xr-x 6 pi pi 4096 May 30 2018 pharo-local
-rw-r--r-- 1 pi pi 2 May 14 2018 pharo.version
```

How to install (manually)

1

2

3

4

5

6

5. Run TelePharo server in a big command line

```
pi@pharoiot-01: ~/pharothings — -ssh pi@pharoiot-01 — 76x24
pi@pharoiot-01:~/pharothings $ ./pharo --headless Pharo32.image remotePharo
--startServerOnPort=40423

'a TlpRemoteUIManager is registered on port 40423'
```

How to install (manually)

1

2

3

4

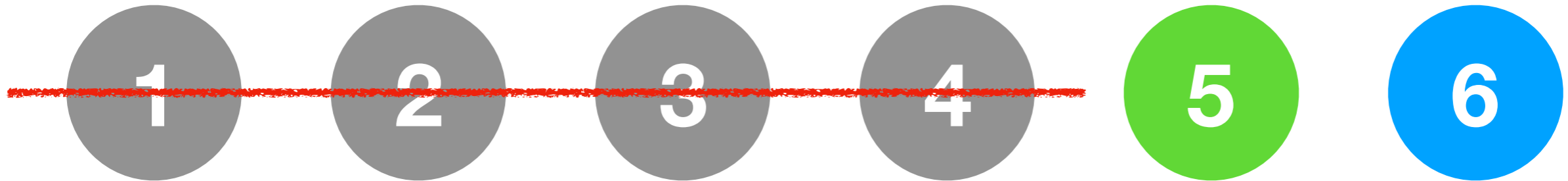
5

6

6. Extra configurations

- Auto connect on WiFi network
- Set hostname
- Autostart TelePharo server every boot
- Set boot to console (disable the Linux UI)
- Enable I2C and SPI kernel modules

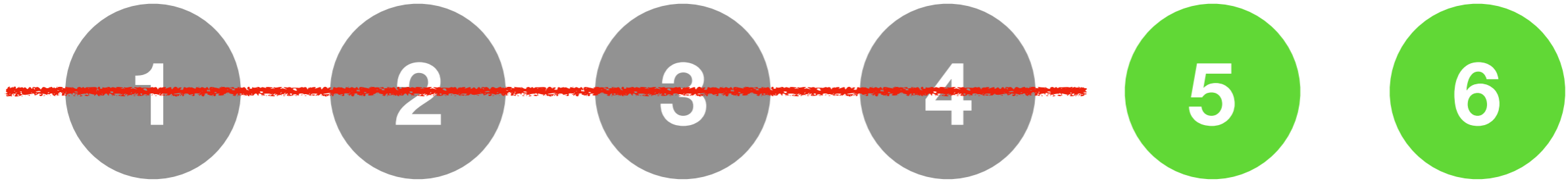
How to install (zero-conf)



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!

How to install (zero-conf)



6. Extra configurations

The screenshot shows the PiBakery web interface. The title is 'PiBakery'. There are three buttons: 'Import', 'Export', and 'Update'. The left sidebar has a menu with 'Startup', 'Programs', 'Network', 'Settings', 'Pi Zero OTG', and 'Other'. The main content area is divided into two sections: 'On Next Boot' and 'On Every Boot'. The 'On Next Boot' section has several configuration options: 'Set hostname to pharoiot-01', 'Set Boot Option to Console', 'Requires restart to take effect', 'Enable automatic loading of SPI kernel module', and 'Setup WiFi' (with fields for Network, Pass, Type, and Country). The 'On Every Boot' section has a 'Run Command' field with the command `cd /home/pi/pharoiot-server/ && ./pharo-server &` and 'As user: pi'. There is also a 'Reboot' button at the bottom.

2 - Difficulties

1. ~~Is not so easy to start (many steps needed to run it);~~
2. ~~Takes a long time to start (get everything);~~
3. Download VMs to different OS (ARM, Linux, Win, OSX);
4. Need the same version on server/client;

Packing everything

- + 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
```


2 - Difficulties

1. ~~Is not so easy to start (many steps needed to run it);~~
2. ~~Takes a long time to start (get everything);~~
3. ~~Download VMs to different OS (ARM, Linux, Win, OSX);~~
4. ~~Need the same version on server/client;~~
5. Needs a keyboard/screen on Raspberry to install from scratch;

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!



2 - Difficulties

1. ~~Is not so easy to start (many steps needed to run it);~~
2. ~~Takes a long time to start (get everything);~~
3. ~~Download VMs to different OS (ARM, Linux, Win, OSX);~~
4. ~~Need the same version on server/client;~~
5. ~~Needs a keyboard/screen on Raspberry to install from~~
~~scratch;~~
6. No many application examples;

PharoThings Booklet



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

PharoThings Booklet

12.5 Creating the application

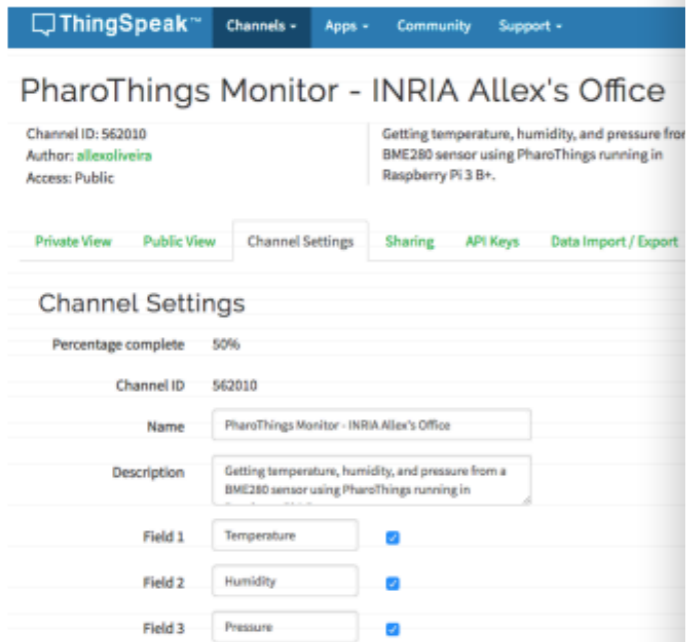


Figure 12-1 ThingSpeak Channel Configuration.

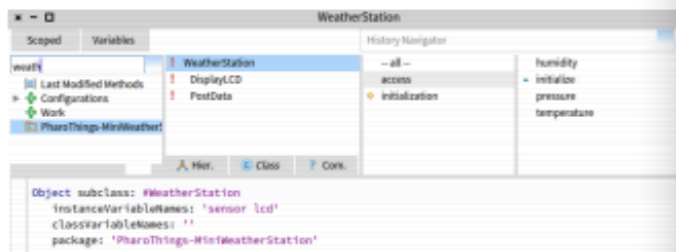


Figure 12-2 Mini Weather Station code.

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

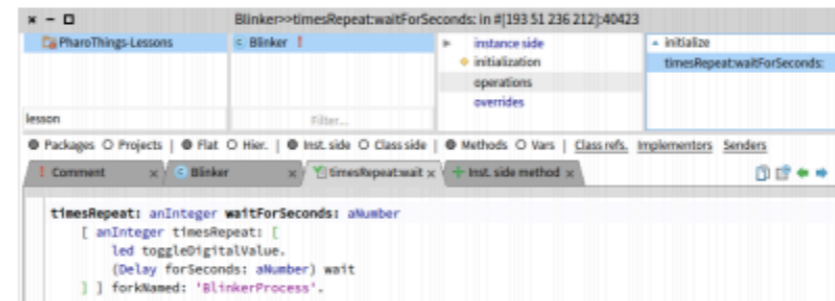


Figure 4-7 Creating an operation method.

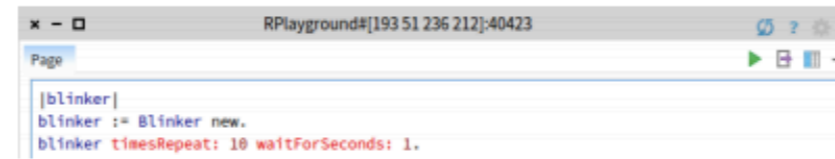


Figure 4-8 Remote playground.

```
|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

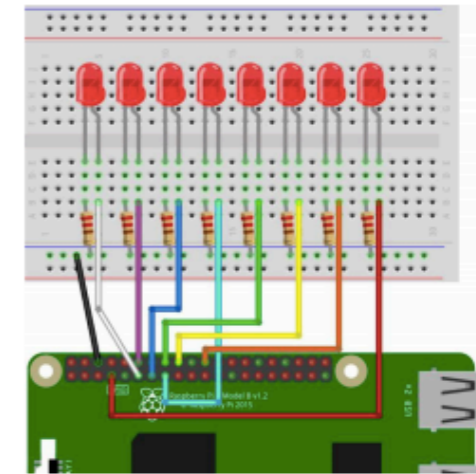


Figure 5-1 Schema connection 8 LEDs.

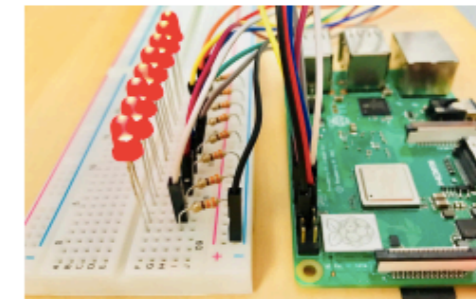


Figure 5-2 Physical connection 8 LEDs.

- 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.

2 - Difficulties

1. ~~Is not so easy to start (many steps needed to run it);~~
2. ~~Takes a long time to start (get everything);~~
3. ~~Download VMs to different OS (ARM, Linux, Win, OSX);~~
4. ~~Need the same version on server/client;~~
5. ~~Needs a keyboard/screen on Raspberry to install from~~
~~scratch;~~
6. ~~No many application examples;~~
7. Search example code: copy and paste (pdf bad formatting);

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] port: 40423).
```

[Open In Playground](#)

Connecting in PharoThings server by Hostname

```
ip := NetNameResolver addressForName: 'pharothings-01'.  
remotePharo := TlpRemoteIDE connectTo: (TCPAddress ip: ip port: 40423).
```

[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.
```

2 - Difficulties

1. ~~Is not so easy to start (many steps needed to run it);~~
2. ~~Takes a long time to start (get everything);~~
3. ~~Download VMs to (VM, Linux, Win, OSX);~~
4. ~~Need the same environment;~~
5. ~~Needs a keyboard/mouse to install from scratch;~~
6. ~~No many application examples;~~
7. ~~Search example code: copy and paste (pdf bad formatting);~~



3 - Continuous Integration

After these improvements,
now we can automate the process:

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

How to automate?

3 - Continuous Integration

Travis CI on PharoThings Booklet

- With each commit in Github, a new instance of a Travis virtual machine is created and the scripts executed
- The booklet is created using Pillar, a Pharo tool-suite to generate documentation, books, websites and slides
- Deploy in Github releases is done in each changes on the booklet
- The process is transparent, anyone can see and contribute to improvements

3 - Continuous Integration

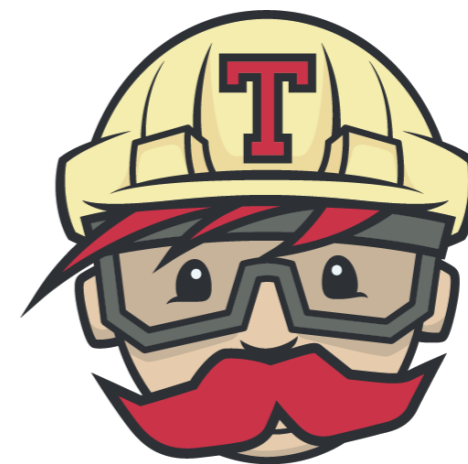
Travis CI on PharoThings Booklet

github.com/SquareBracketAssociates/Booklet-APharoThingsTutorial/.travis.yml

travis-ci.org/SquareBracketAssociates/Booklet-APharoThingsTutorial

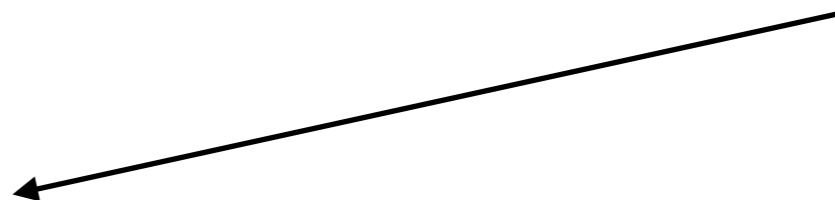


new commit



run scripts/tests

new Github release



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

3 - Continuous Integration

Travis CI on PharoThings Booklet

The image displays three overlapping screenshots from a GitHub repository named 'SquareBracketAssociates / Booklet-APharoThingsTutorial'.

- Left Screenshot:** Shows the repository overview. It includes a 'Code' tab, navigation links for Issues, Pull requests, Projects, Wiki, Insights, and Settings. Below the repository name, it states 'PharoThings supports live development of IoT related application.' and shows a commit history with 63 commits and 1 branch. A file list includes 'Chapters', '_support', 'images_', '.gitignore', '.travis.yml', 'README.md', 'index.pillar', and 'pillar.conf'. A 'README.md' preview is visible at the bottom, featuring a 'build passing' badge.
- Middle Screenshot:** Shows the content of the '.travis.yml' file. The file is 35 lines long (27 sloc) and 1.08 KB. It is a configuration for Travis CI, specifying the language as 'smalltalk', the OS as 'linux', and the environment as 'Pharo-7.0'. It includes instructions for installing pillar, running unit and integration tests, and generating a PDF from the pillar markup.
- Right Screenshot:** Shows a Travis CI build log for a 'continuous' build. The build is marked as a 'Pre-release' and was completed 8 days ago. The log includes a 'Continuous build' section with a link to the build log: <https://travis-ci.org/SquareBracketAssociates/Booklet-APharoThingsTutorial/builds/511461352/>. Below the log, there are three assets: 'APharoThingTutorial-wip.pdf' (2.63 MB), 'Source code (zip)', and 'Source code (tar.gz)'.

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

3 - Continuous Integration

Travis CI on Pharo IoT

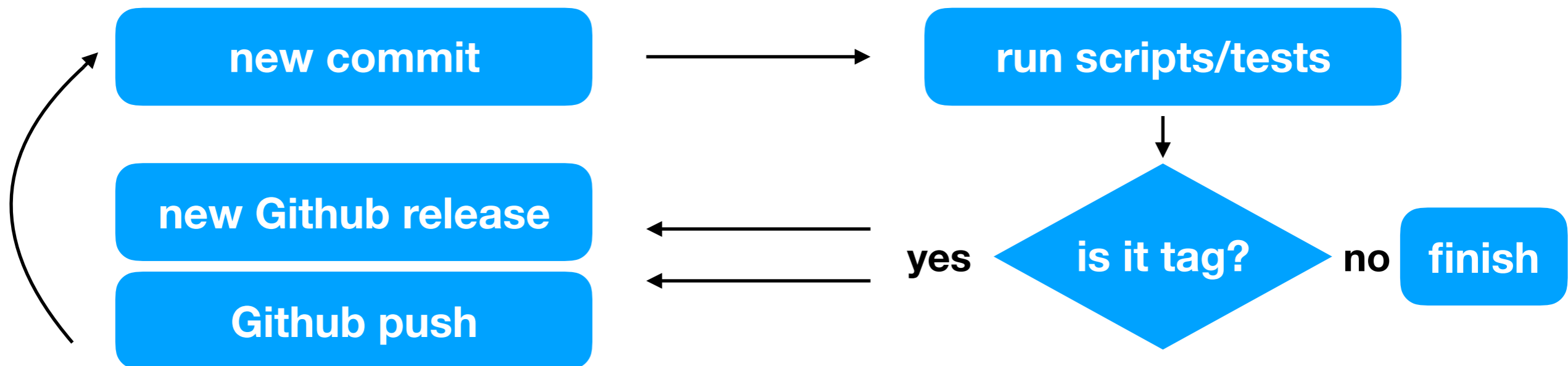
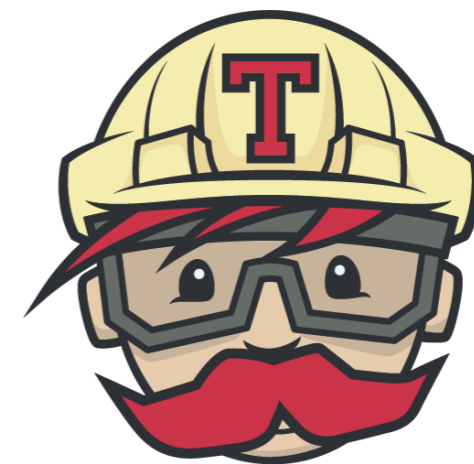
- With each commit in Github, a new instance of a Travis virtual machine is created and the scripts executed
- Deploy is done only when a new Github Tag is created
- PharoThings is loaded in 32/64 images
- Zip files with everything (VMs, Images, 1 click-run) are up in Github releases. We put it inside the repo also.
- The process is transparent, anyone can see and contribute to improvements

3 - 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>

3 - 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 'Travis upload release files' commit by oliveiraallex is highlighted.
- Top-right screenshot:** Shows a Travis CI build log for commit 4ff58c2. The build is successful, with 38 tests passed. The log includes steps like 'Worker information', 'Build system information', 'Installing APT Packages', and 'Setting environment variables from repository settings'.
- Bottom-right screenshot:** Shows the 'Releases' page for version v0.1. It lists five assets: client.zip (24.2 MB), multi.zip (78.5 MB), server.zip (23.8 MB), Source code (zip), and Source code (tar.gz).

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

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 port 40423
- 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 commit history:

- Repository: pharo-iot / Ci
- Branch: master
- Path: Ci / docs /
- Commit history table:

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>

github.com/pharo-iot

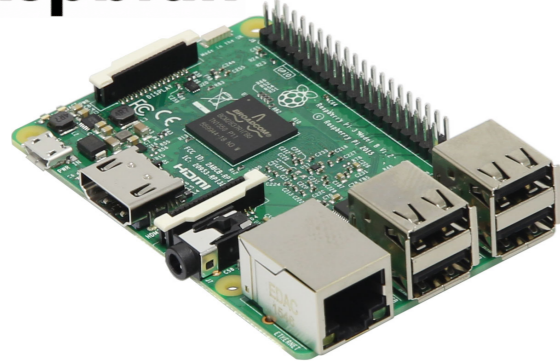
- We are starting using Github Projects to work in a more transparent and dynamic way.

The screenshot displays the GitHub Projects interface for the 'pharo-iot' organization. The top navigation bar includes 'Search or jump to...', 'Pull requests', 'Issues', 'Marketplace', and 'Explore'. Below the organization name, there are tabs for 'Repositories 5', 'People 7', 'Teams 0', 'Projects 1', and 'Settings'. The main content area shows a Kanban board with three columns: 'To do' (6 items), 'In progress' (3 items), and 'Done' (13 items). A large green text overlay reads 'You can take some tasks!'. The tasks listed include adding sensor code, creating tutorials, moving content, and setting up CI.

Column	Count	Task	Added by
To do	6	Add the HCRS04 sensor code in PharoThings repository	oliveiraalex
		Add the LCD1602 code in PharoThings repository	oliveiraalex
		Test and change some code in PharoThings booklet	oliveiraalex
In progress	3	check that all repos are part of pharoiot GitHub (seamless and so on)	MarcusDenker
		Creating some videos tutorials	oliveiraalex
Done	13	Moved all conf pages and tutorials to get.pharoiot.org	oliveiraalex

<https://github.com/orgs/pharo-iot/projects/1>

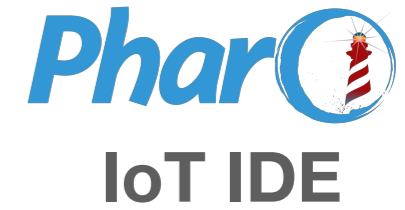
Now you can



< 10 minutes



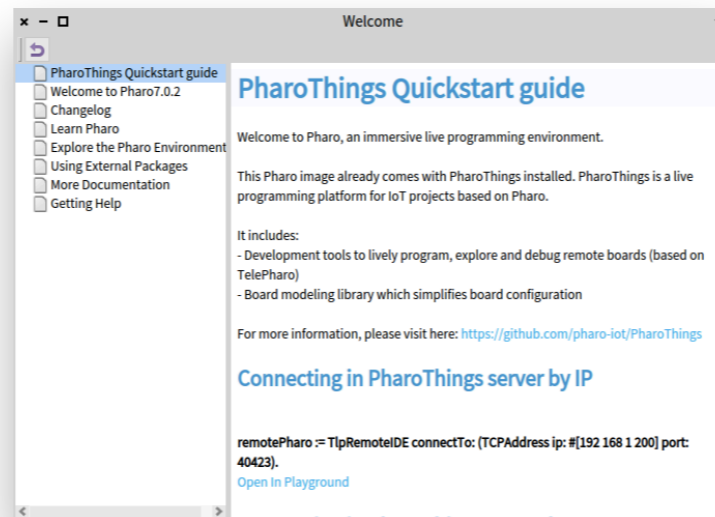
< 1 minute



< 1 minute



Booklet

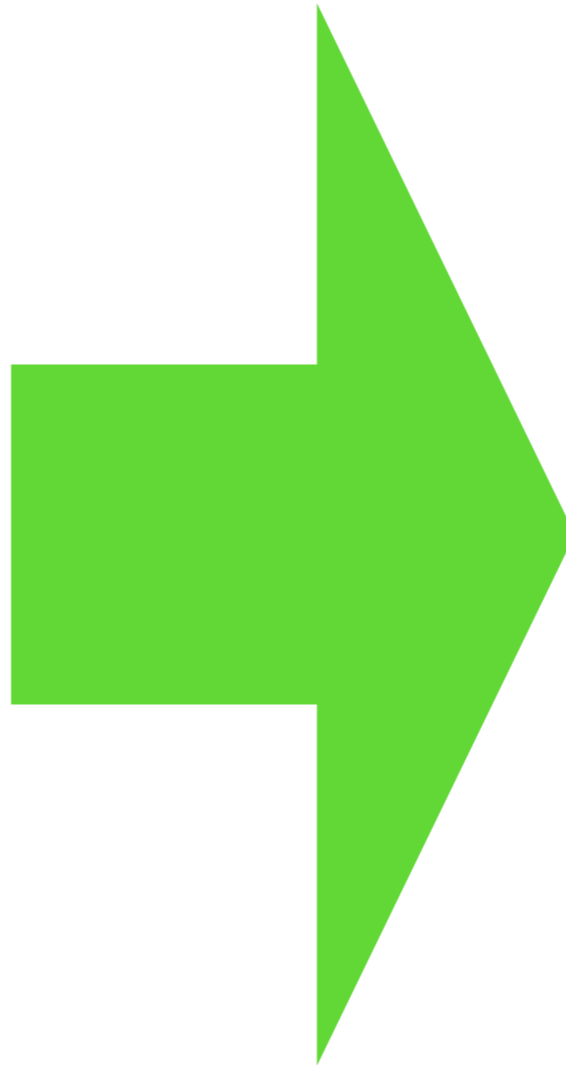


Quickstart Guide



Be part of the project!

4 - Future



pharoiot.org

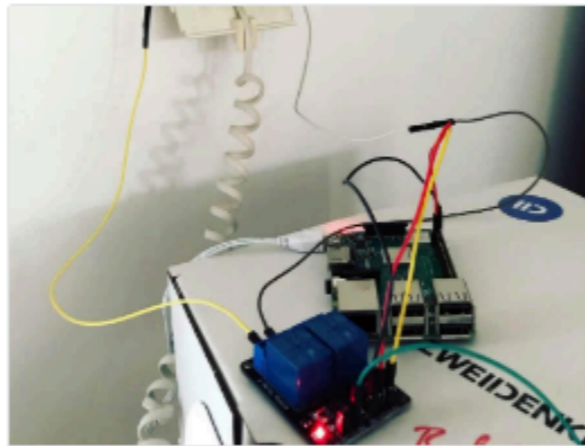
Everything in the same place, to facilitate the journey of the new user.

The screenshot shows the homepage of PHARO IoT, a live programming platform. The header includes the logo 'PHARO IoT' and the tagline 'Live programming platform', along with navigation links for 'START HERE', 'DOWNLOAD', 'LESSONS', 'PROJECT HUB', and 'CONTACT'. The main hero section features a dark background with a city skyline at night, the headline 'Unlock the power of Internet of Things', and a sub-headline 'Start Pharo IoT's journey now and create IoT applications very quickly and easily'. A prominent blue 'START NOW' button is centered. Below this, a section titled 'Let's get right to the point' contains three white cards with blue icons and text: 1. 'Install Raspberry Pi Runtime' with a Wi-Fi icon and subtext 'How to do a headless installation and set up your Raspberry.' 2. 'Install IDE on Mac Windows Linux' with a folder icon and subtext '1 click to run Pharo IoT in Windows, Linux, Mac!'. 3. 'Start PharoThings Lessons' with a book icon and subtext 'Learn how to create IoT applications quickly.'

pharoiot.org

- Share your IoT projects using Pharo with the community!

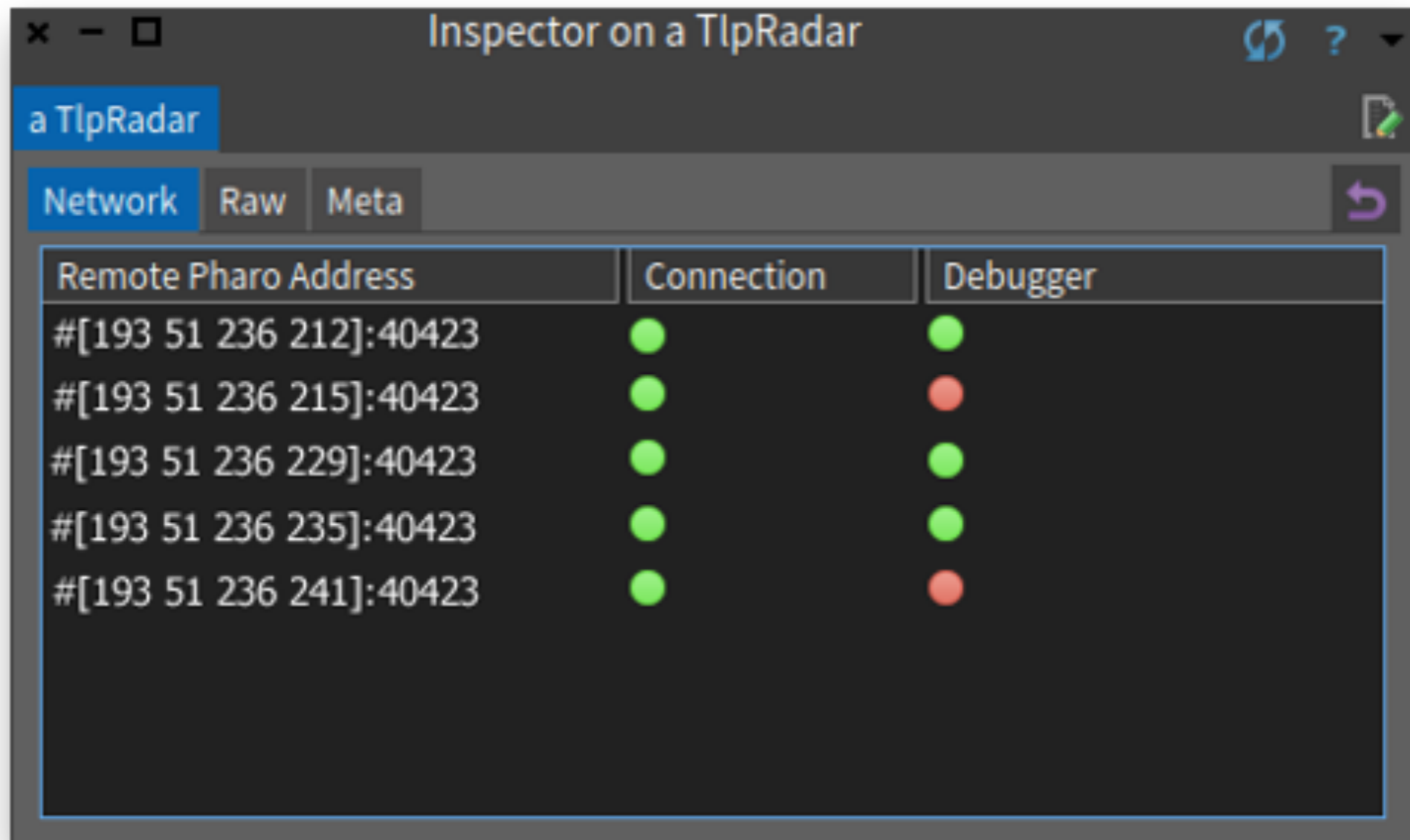
Community projects



[SUBMIT YOUR PROJECT](#)

Tele Radar

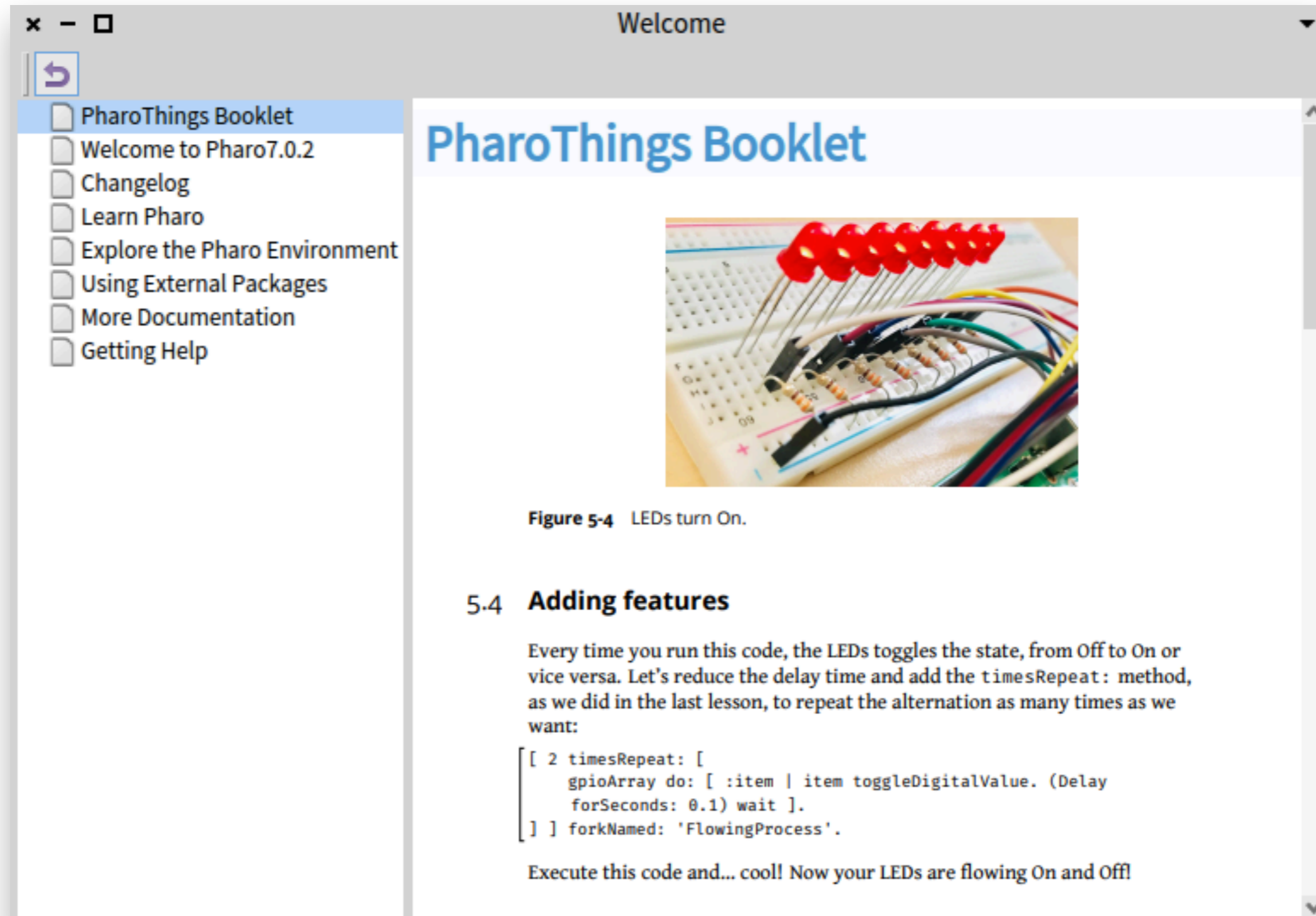
Automatic detection of running images in network
(TeleRadar using SSDP protocol)



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 (selected)
- Welcome to Pharo7.0.2
- Changelog
- Learn Pharo
- Explore the Pharo Environment
- Using External Packages
- More Documentation
- Getting Help

The main content area displays the "PharoThings Booklet" page. At the top, the title "PharoThings Booklet" is shown in blue. Below the title is a photograph of a breadboard with several red LEDs connected to a circuit. The LEDs are all illuminated, appearing as bright red dots.

Figure 5-4 LEDs turn On.

5.4 Adding features

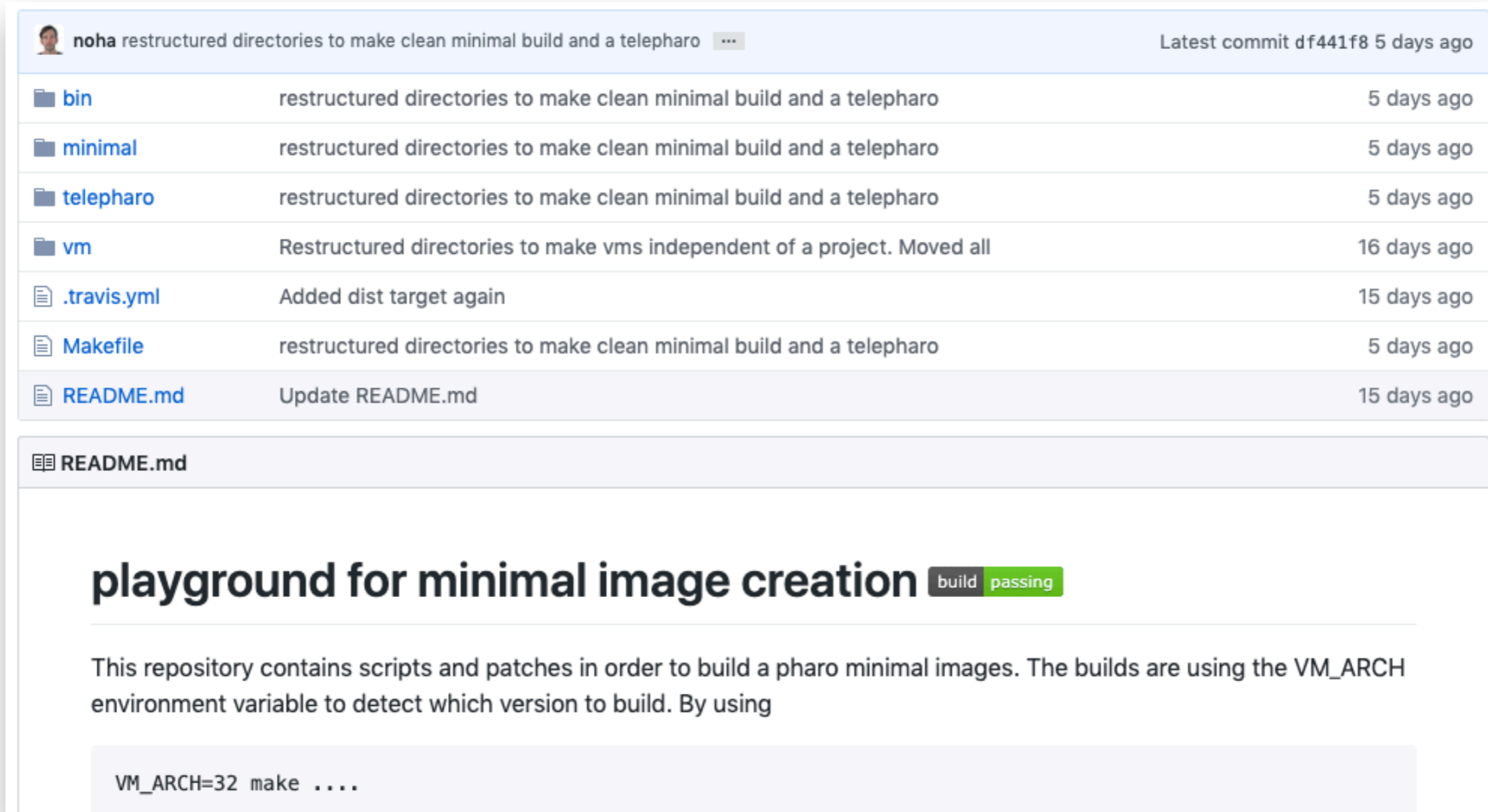
Every time you run this code, the LEDs toggles the state, from Off to On or vice versa. Let's reduce the delay time and add the `timesRepeat:` method, as we did in the last lesson, to repeat the alternation as many times as we want:

```
[ 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!

Minimal PharoThings image

- Norbert Hartl



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 and includes a code snippet: 'VM_ARCH=32 make'.

File/Folder	Commit Message	Time Ago
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

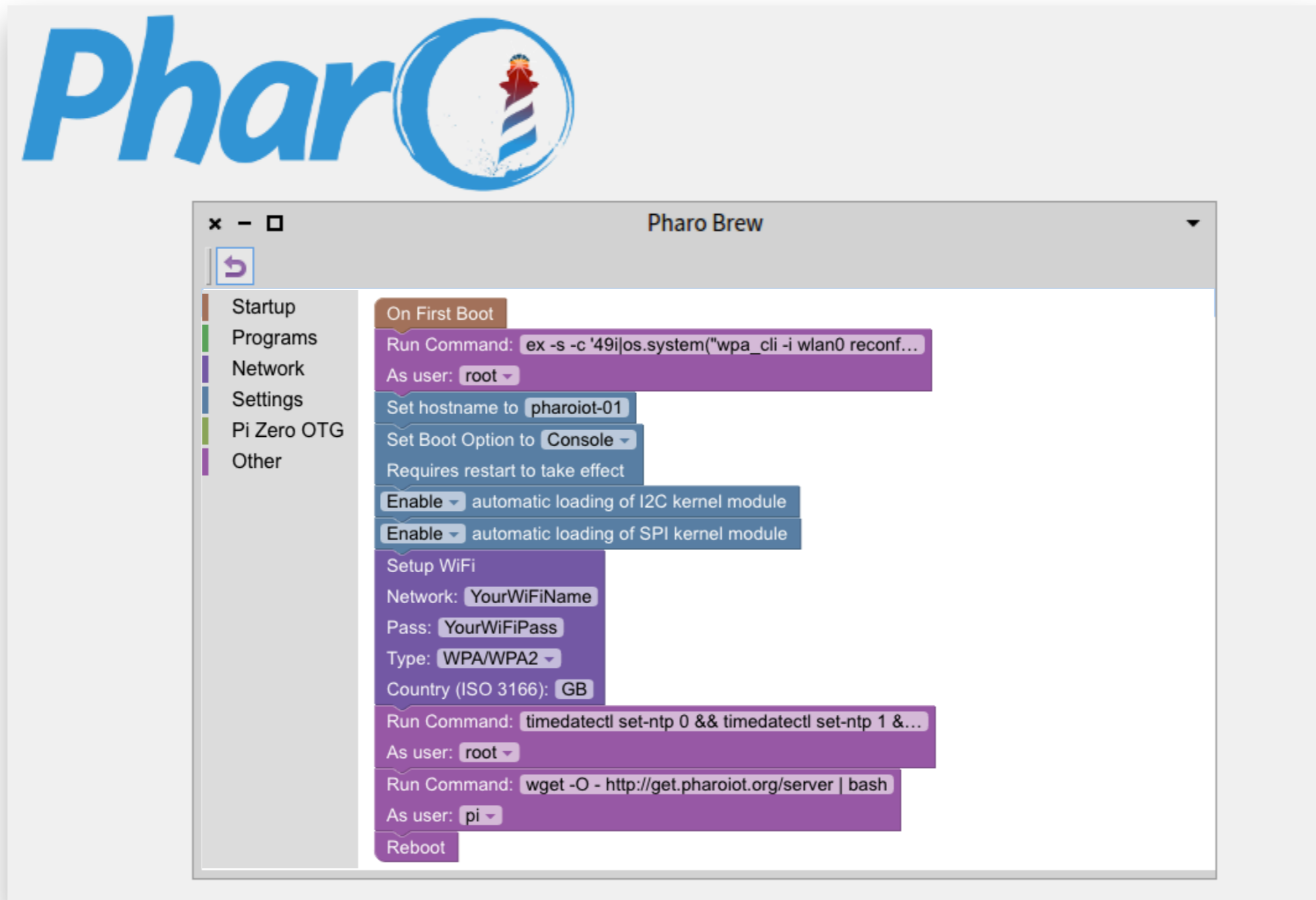
This repository contains scripts and patches in order to build a pharo minimal images. The builds are using the VM_ARCH environment variable to detect which version to build. By using

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

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

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 10 MINUTES!**

THANKS!



Any questions?

allex.oliveira@msn.com

Presentation Information

This slides was presented at Pharo Days 2019, Lille, France

<https://pharo.org/2019PharoDays>

- Title: Pharo IoT - Installation Improvements and Continuous Integration
- Presenters:
Alex Oliveira - [linkedin.com/in/alex-oliveira](https://www.linkedin.com/in/alex-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>