

Smart Devices

Agenda

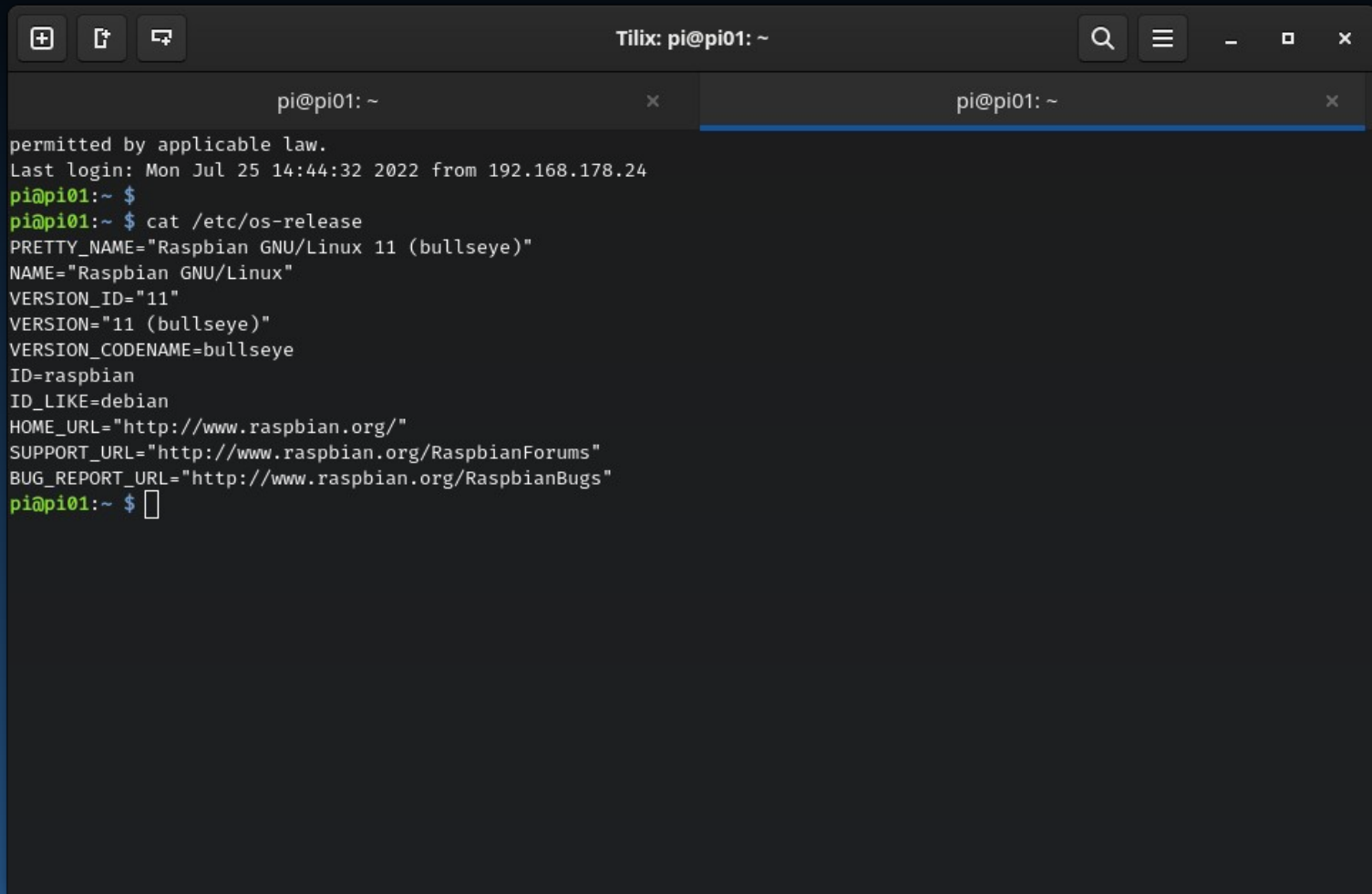
- Requirements
- Preparation
- Concept
- Programming
 - Web Server
 - Auto Start Web Server
 - Auto Start Browser

Requirements

Requirements

- Software
 - Raspberry Imager
 - Raspberry Pi OS
 - Fritzing
 - Putty or other SSH client

Requirements



A terminal window titled "Tilix: pi@pi01: ~" with standard window controls. It contains two tabs, both labeled "pi@pi01: ~". The active tab shows the output of the command `cat /etc/os-release`, which displays the following system information:

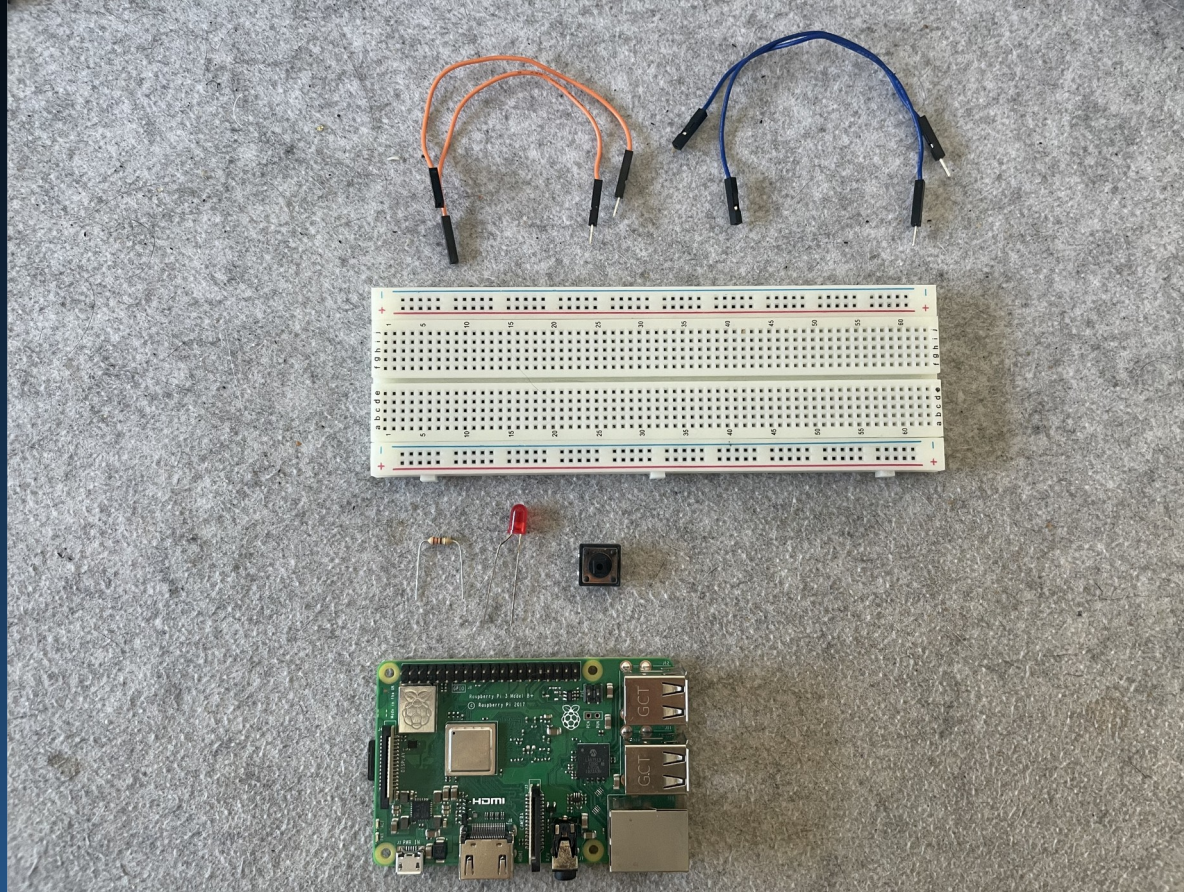
```
permitted by applicable law.  
Last login: Mon Jul 25 14:44:32 2022 from 192.168.178.24  
pi@pi01:~ $  
pi@pi01:~ $ cat /etc/os-release  
PRETTY_NAME="Raspbian GNU/Linux 11 (bullseye)"  
NAME="Raspbian GNU/Linux"  
VERSION_ID="11"  
VERSION="11 (bullseye)"  
VERSION_CODENAME=bullseye  
ID=raspbian  
ID_LIKE=debian  
HOME_URL="http://www.raspbian.org/"  
SUPPORT_URL="http://www.raspbian.org/RaspbianForums"  
BUG_REPORT_URL="http://www.raspbian.org/RaspbianBugs"  
pi@pi01:~ $
```

Requirements

Hardware

- Raspberry Pi
- Breadboard
- LED
- Resistor
- Cables
- Some Display

Programming - Preparation



Preparation

Preparation

- Manual Install

- Download lite image

- ```
$ wget https://downloads.raspberrypi.org/raspios_lite_armhf_latest
```

- Download regular image

- ```
$ wget https://downloads.raspberrypi.org/raspios_armhf_latest
```

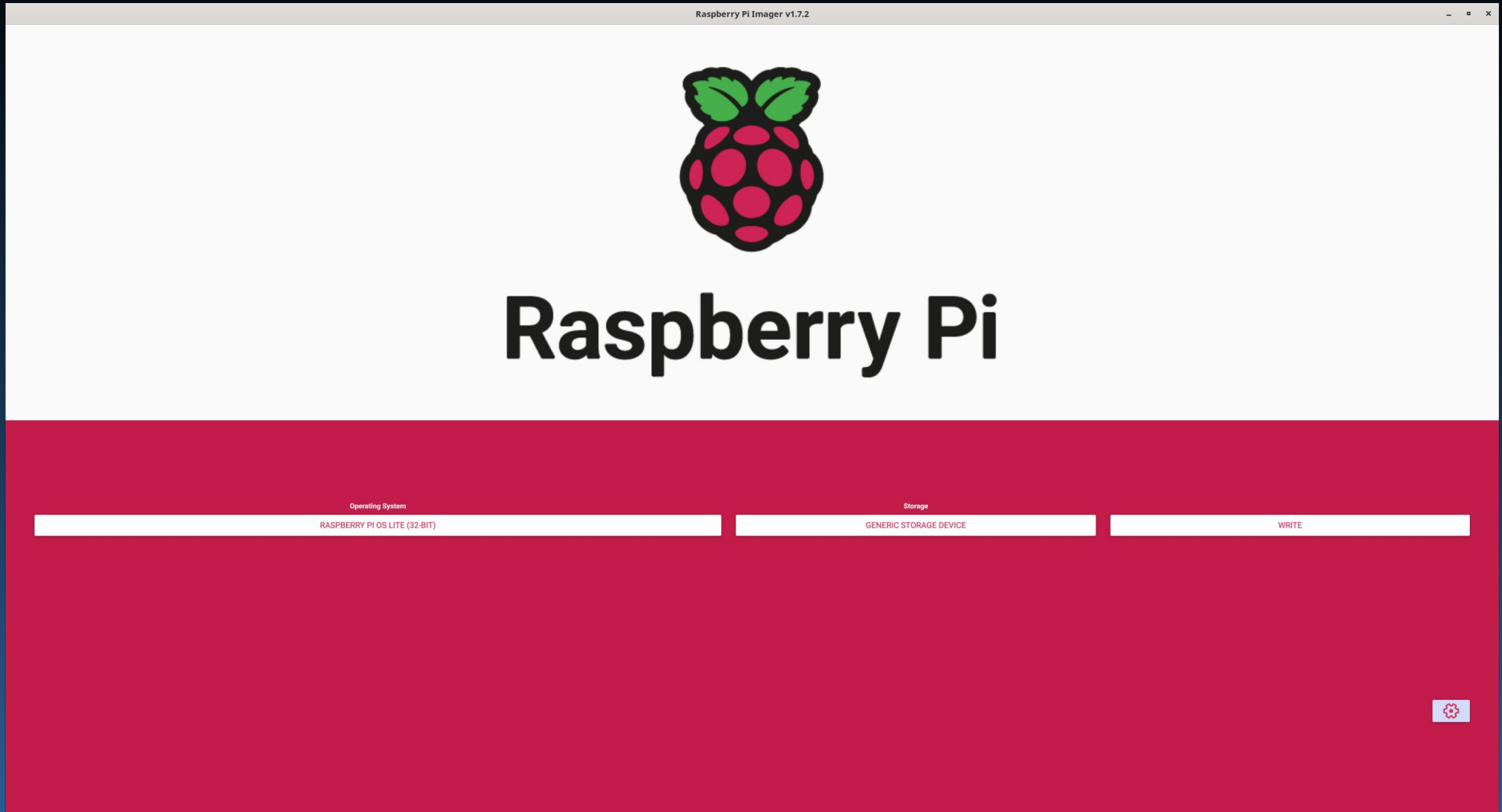
- Write SD card

- ```
$ unzip -p <IMAGE> | sudo dd of=/dev/mmcblk0 oflag=sync status=progress bs=4M
```

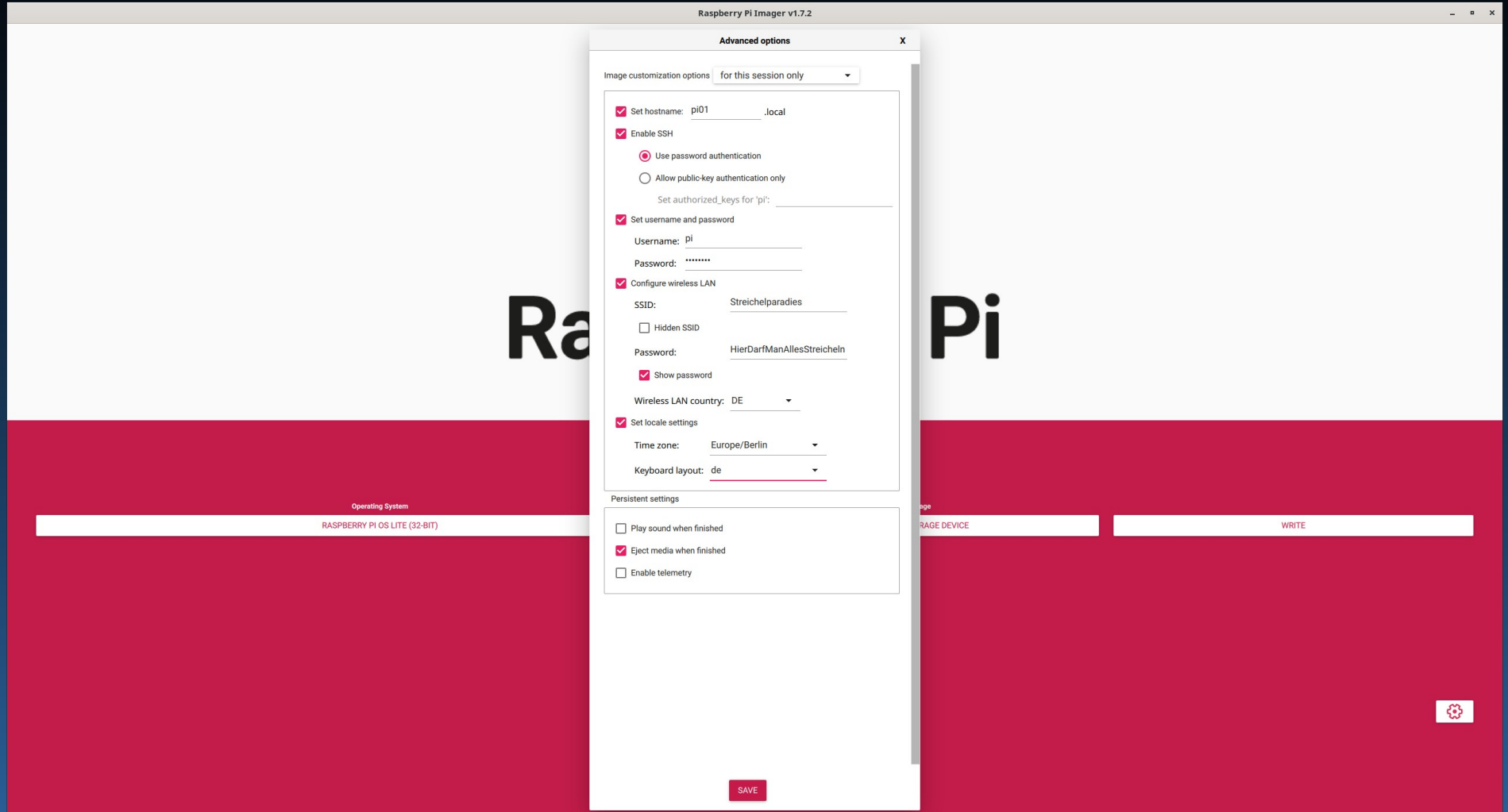
- Guided Install

- Download Raspberry Imager

# Preparation

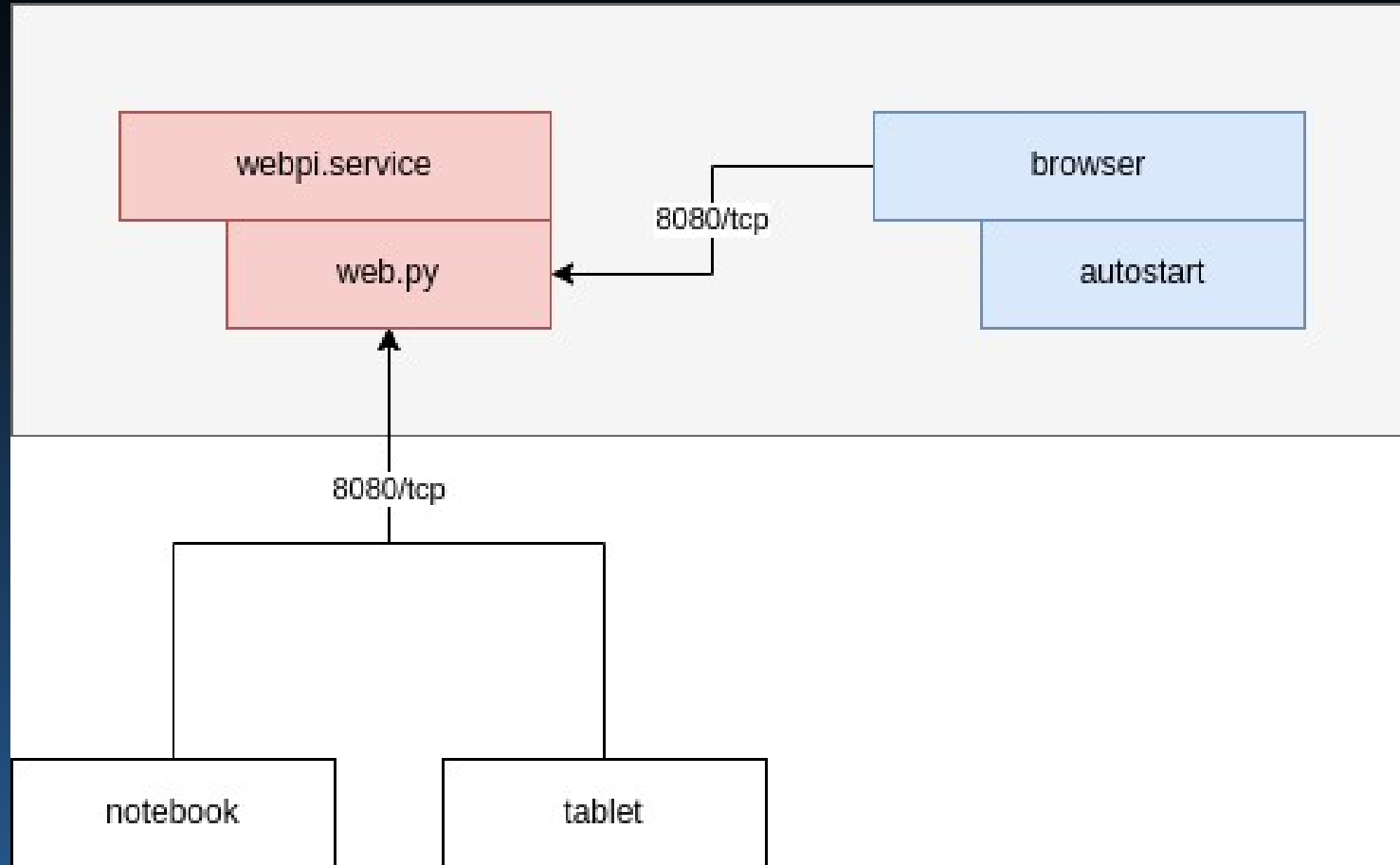


# Preparation



Concept

# Concept

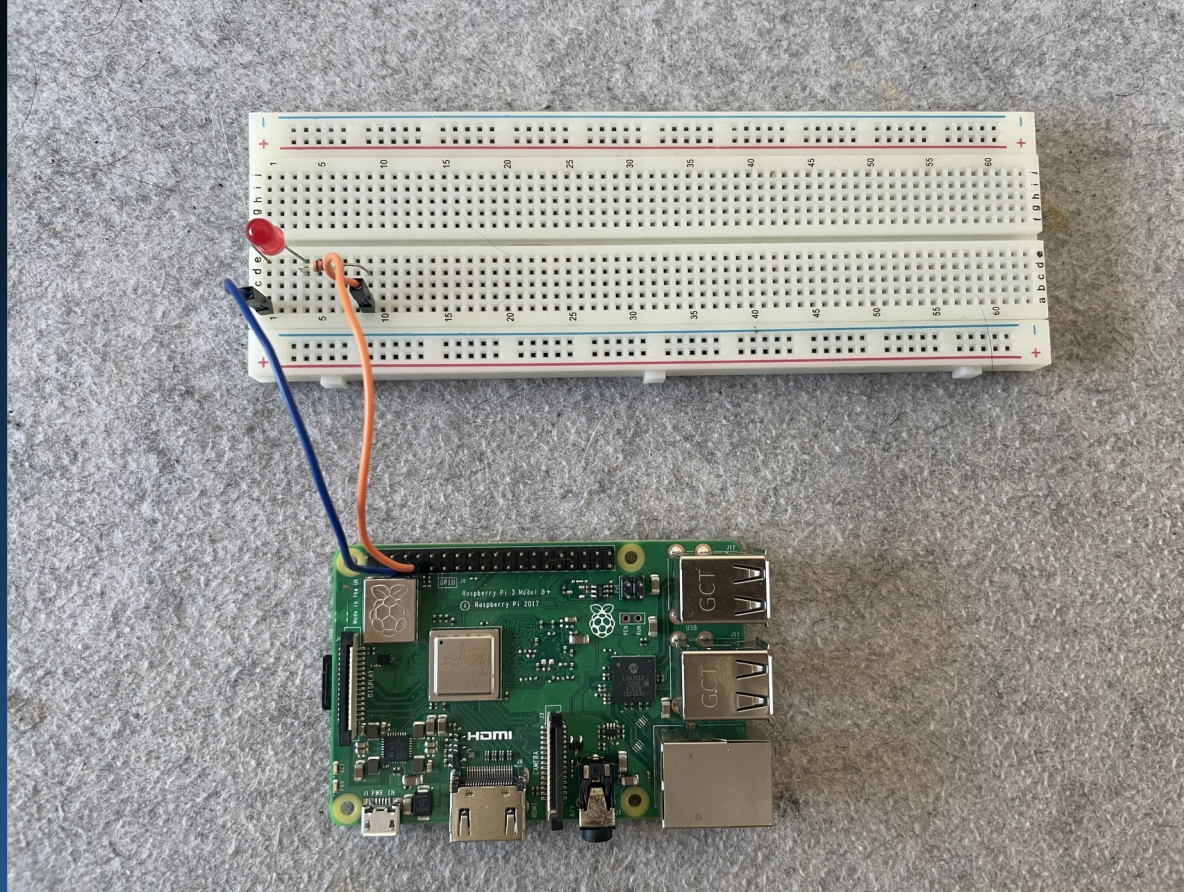


# Programming

# Programming - Preparation

- Create a working directory  
`$ mkdir project`
- Navigate there  
`$ cd project/`
- Start a new file  
`$ nano led_1.py`
- Continue with the exercises

# Programming – LED and Web Server





# Programming – LED 1

```
from gpiozero import LED
from time import sleep

led = LED(4)

led.on()
sleep(1)
led.off()
```

# Programming – Web Server 1

```
from http.server import BaseHTTPRequestHandler, HTTPServer
import time

hostName = ""
serverPort = 8080

class MyServer(BaseHTTPRequestHandler):

 def do_GET(self):
 self.send_response(200)
 self.send_header("Content-type", "text/html")
 self.end_headers()
 html = '''
 <html>
 <body>
 style="width:960px; margin: 20px auto;">
 <h1>Welcome to my Raspberry Pi</h1>
 <form action="/" method="POST">
 Turn LED_1 :
 <input type="submit" name="LED_1" value="On">
 <input type="submit" name="LED_1" value="Off">
 </form>
 </body>
 </html>
 '''
 self.wfile.write(html.encode("utf-8"))

Main

if __name__ == "__main__":
 webServer = HTTPServer((hostName, serverPort), MyServer)
 print("Server started http://%s:%s" % (hostName, serverPort))

 try:
 webServer.serve_forever()
 except KeyboardInterrupt:
 pass

 webServer.server_close()
 print("Server stopped.")
```

# Programming – Web Server 2

```
...SNIP...
def do_POST(self):

 content_length = int(self.headers['Content-Length'])
 post_data = self.rfile.read(content_length).decode("utf-8")
 print(post_data)

 if post_data == 'LED_1=On':
 led_1.on()
 elif post_data == 'LED_1=Off':
 led_1.off()

 self._redirect('/')
...SNIP...
```

# Programming – Web Server 3

```
...SNIP...
```

```
 def _redirect(self, path):
 self.send_response(303)
 self.send_header('Content-type', 'text/html')
 self.send_header('Location', path)
 self.end_headers()
```

```
...SNIP...
```

# Configuring – Autostart Web Server

## My Smart Pi

Turn LED\_1 :

Turn LED\_2 :

# Configuring – Autostart Web Server

```
Copy your server program
$ sudo cp web_3.py /usr/local/bin/
```

```
Create a new systemd service unit
$ sudo nano /etc/systemd/system/webpi.service
```

```
/etc/systemd/system/webpi.service
```

```
[Unit]
Description=A bit about your service

[Service]
Type=simple
ExecStart=python /usr/local/bin/web_3.py
Restart=on-failure

[Install]
WantedBy=multi-user.target
```

# Configuring – Autostart Web Server

```
Enable the server
$ sudo systemctl daemon-reload
$ sudo systemctl enable webpi.service
$ sudo systemctl start webpi.service
```

```
Test
$ netstat -tlpn
$ curl localhost:8080
```

# Configuring – Autostart Browser





# Configuring – Autostart Browser

```
Install (this will take a while)
$ sudo apt-get update
$ sudo apt-get install ttf-mscorefonts-installer \
 x11-xserver-utils midori lxsession lightdm
```

```
Configure Autostart Desktop (with automatic login)
$ sudo raspi-config
$ reboot
```

# Configuring – Autostart Browser

```
Change Autostart
$ sudo nano .config/lxsession/LXDE/autostart
```

```
.config/lxsession/LXDE/autostart
@midori -e fullscreen -a http://localhost:8080
@xset s noblank
@xset s off
@xset -dpms
```

## Docs & Links

- <https://www.raspberrypi.org/>
- <https://www.raspberrypi.com/documentation/computers/os.html#gpio-and-the-40-pin-header>
- <https://gpiozero.readthedocs.io/en/stable/>
- <https://github.com/dschier-wtd/presentations/>