Best Practice for Open Training Materials

CC BY-SA 4.0, @tamberg, FHNW
Screenshots considered fair use

Source on G Docs
Keep it simple

Github repo as a single point of entry.
Slides per topic / lesson, links into repo.
Provide short, focused, working examples.
Simple to get started, possible to learn more.
Lesson 3: Sending Sensor Data to IoT Platforms

- Slides

Examples

- ESP8266_WiFiClient.ino
- ESP8266_WiFiClientSecure.ino
- ESP8266_WiFiClientSecureCaCert.ino
- ESP8266_WiFiClientSecureFingerprint.ino
- ESP8266_WiFiMacAddress.ino
- ESP8266_WifiNtpClient.ino
- ESP8266_WifiSetup.ino
- ESP8266_WiFiTimeClient.ino

Hands-on

- Hands-on Lesson 3
IoT reference model

Device → Local Gateway → Cloud Backend → Client

Sensor or Actuator

Physical Interaction

Virtual Interaction

User

3rd-party Service
ESP8266 Wi-Fi setup

#include <ESP8266WiFi.h>

void setup() {
    Serial.begin(115200); // for debug output
    WiFi.mode(WIFI_STA); // _AP|_AP_STA|_OFF
    WiFi.begin("MY_SSID", "MY_PASSWORD"); // TODO
    while (WiFi.status() != WL_CONNECTED) {
        delay(500);
    }
    Serial.println(WiFi.localIP());
}
```cpp
#include <ESP8266WiFi.h>

const char *ssid = "MY_SSID"; // TODO
const char *password = "MY_PASSWORD"; // TODO

void setup()
{
    Serial.begin(115200);
    Serial.print("\nConnecting to network ");
    Serial.println(ssid);
    WiFi.mode(WIFI_STA); // or WIFI_AP, WIFI_AP_STA, WIFI_OFF
    WiFi.begin(ssid, password);
    while (WiFi.status() != WL_CONNECTED) {
        delay(500);
    }
    Serial.print("Connected, IP = ");
    Serial.println(WiFi.localIP());
}
```
<table>
<thead>
<tr>
<th>Name</th>
<th>Last commit message</th>
<th>Last commit date</th>
</tr>
</thead>
<tbody>
<tr>
<td>.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>ESP8266_ThingSpeakWiFiClientSecure</td>
<td>Update.</td>
<td>2 years ago</td>
</tr>
<tr>
<td>ESP8266_ThingSpeakWiFiClientSecureWithTime</td>
<td>Update.</td>
<td>2 years ago</td>
</tr>
<tr>
<td>ESP8266_WiFiClient</td>
<td>Update.</td>
<td>2 years ago</td>
</tr>
<tr>
<td>ESP8266_WiFiClientSecure</td>
<td>Update.</td>
<td>2 years ago</td>
</tr>
<tr>
<td>ESP8266_WiFiClientSecureWithTime</td>
<td>Update.</td>
<td>2 years ago</td>
</tr>
<tr>
<td>ESP8266_WiFiClientSecureCaCert</td>
<td>Update ESP8266_WiFiClientSecureCaCert.ino</td>
<td>10 months ago</td>
</tr>
<tr>
<td>ESP8266_WiFiClientSecureFingerprint</td>
<td>Update.</td>
<td>10 months ago</td>
</tr>
<tr>
<td>ESP8266_WiFiMacAddress</td>
<td>Added example code.</td>
<td>4 years ago</td>
</tr>
<tr>
<td>ESP8266_WiFiNtpClient</td>
<td>Cleanup.</td>
<td>4 years ago</td>
</tr>
</tbody>
</table>
Provide support

Hardware kit with MCU, basic sensors, actuators.
Curated Github Wiki for details and troubleshooting.
Github issues to report bugs, typos, specific fixes, etc.
Github Classroom for individual, private copies of a repo.
IoT Hardware for CS Bachelor Students

CC BY-SA thomas.amberg@fhnw.ch, 24.01.2019

Abstract
Options and thoughts around Internet of Things hardware for computer science bachelor students.

Introduction
The following options were collected during the evaluation of IoT hardware for the course IoT Engineering [0] at FHNW, the University of Applied Sciences and Arts Northwestern Switzerland.

[0] https://www.fhnw.ch/de/studium/module/9980188

A simple reference model
A simple IoT reference model [1] is used to identify the main technical parts of connected products:

- Device (with Sensors & Actuators) <-> (Edge) Gateway <-> (Cloud) Backend <-> Client (Apps)


Learning objectives
The course IoT Engineering [0] introduces bachelor students to IoT, covering the following topics:

- Architecture of IoT systems and Internet-connected products
- Prototyping with beginner hardware, sensors and actuators
- Connectivity options for short and long-distance data transmission
IoT Engineering Wiki

The Wiki contains tools and hardware setup instructions which are referenced by lessons.

Found a typo or something missing? Submit an issue.

Development tools

Development environment

- Arduino
- VS Code

Command-line tools

- curl
- mqtt
- coap

Network analysis

- Wireshark

Hardware
Hardware

Hardware components:
- Arduino
- Command Line Tools
- Feather Huzzah ESP8266
- Feather M4 Express
- Feather nRF52840 Express
- FeatherWing RFM95W
- Grove Actuators
- Grove Adapters
- Grove Sensors
- IoT Books
- IoT Platforms
- Raspberry Pi Zero W
- Various
- VS Code

Linux computers:
- Raspberry Pi Zero W with Wi-Fi & BLE
Linux computers

- Raspberry Pi Zero W with Wi-Fi & BLE

Microcontrollers

- Feather Huzzah ESP8266 with Wi-Fi
- Feather nRF52840 Express with BLE

Extensions

- FeatherWing RFM95W LoRaWAN

Grove sensors

- Button
- Light Sensor v1.2
- Rotary Angle Sensor
- Temperature & Humidity Sensor
- Ultrasonic Ranger

Grove actuators

- 4-Digit Display
- Buzzer
- Chainable RGB LED
Feather Huzzah ESP8266
Thomas Amberg edited this page on Sep 28, 2019 - 74 revisions

Buy

- https://www.adafruit.com/product/3213 (w/ stacking headers, ~20$)

Code

Arduino

Setup (step-by-step)


Setup (for experts)

- On macOS, after install, go to System Preferences > Security & Privacy > General > Allow ...
- On macOS 10.12.6 Sierra or higher, if above does not work, try this USB driver legacy version: http://community.silabs.com/t5/Interface-Knowledge-Base/Legacy-OS-Software-and-Driver-Packages/ta-p/182585
- Preferences > Additional Boards Manager URL:
Pinout

Note: the Particle Grove adapter needs this fix and this fix to work with the ESP8266.

Temperature & Humidity Sensor (DHT11)

Buy


Pinout

Connect to Grove D<XY>

<table>
<thead>
<tr>
<th>Color</th>
<th>Signal</th>
</tr>
</thead>
<tbody>
<tr>
<td>Black</td>
<td>GND</td>
</tr>
<tr>
<td>Red</td>
<td>VCC (3.3V)</td>
</tr>
<tr>
<td>White</td>
<td>NC</td>
</tr>
<tr>
<td>Yellow</td>
<td>SIG (Digital)</td>
</tr>
</tbody>
</table>
Code

Arduino

ESP8266

- Library: Sketch > Include Library > Manage Libraries... > DHTesp > Install
- Example: File > Examples > DHT sensor library for ESPx > DHT_ESP8266
  - In the example code, make sure to use DHT11

```cpp
//dht.setup(..., DHTesp::DHT22);
dht.setup(..., DHTesp::DHT11);
```

nRF52840

- Library: Sketch > Include Library > Manage Libraries... > Grove DHT11 > Install
- Example: File > Examples > Grove Temperature and Humidity Sensor > DHTtester
  - In the example code, make sure to use DHT11

```cpp
#define DHTTYPE DHT22
#define DHTTYPE DHT11
```

Python on Raspberry Pi

- Library: https://github.com/Seeed-Studio/grove.py

Resources
Assignments

- **fhnw-iot-project-hs22**
  Group assignment for FHNW Iot (5ibb1) Teams HS22

- **fhnw-iot-work-00**
  Individual assignment

- **fhnw-iot-work-01**
  Individual assignment
Here's a better fix, using a 100 kΩ resistor:

And a pragmatic fix, which consumes a bit more energy, using a jumper cable:
Grove Adapter for Feather HUZZAH
ESP8266
Take pictures
Reduce clutter.
Use good lighting.
Show how it works.
Always take one more.
Enable reuse

CC0 1.0 for example code.

MIT license for student projects, libraries.

CC BY-SA 4.0 for slides, pictures, designs, hardware, etc.

Use open source tools, e.g. Fritzing for layouts, OpenScad for designs.
Attribution-ShareAlike 4.0 International (CC BY-SA 4.0)

This is a human-readable summary of (and not a substitute for) the license. Disclaimer.

You are free to:

- **Share** — copy and redistribute the material in any medium or format
- **Adapt** — remix, transform, and build upon the material for any purpose, even commercially.

The licensor cannot revoke these freedoms as long as you follow the license terms.
CC0 1.0 Universal (CC0 1.0)
Public Domain Dedication

No Copyright

The person who associated a work with this deed has dedicated the work to the public domain by waiving all of his or her rights to the work worldwide under copyright law, including all related and neighboring rights, to the extent allowed by law.

You can copy, modify, distribute and perform the work, even for commercial purposes, all without asking permission. See Other Information below.
Choose an open source license

An open source license protects contributors and users. Businesses and savvy developers won’t touch a project without this protection.

Which of the following best describes your situation?

I need to work in a community.
Use the license preferred by the community you’re contributing to or depending on. Your project will fit right in.
If you have a dependency that doesn’t have a license, ask its maintainers to add a license.

I want it simple and permissive.
The MIT License is short and to the point. It lets people do almost anything they want with your project, like making and distributing closed source versions.
Babel, .NET, and Rails use the MIT License.

I care about sharing improvements.
The GNU GPLv3 also lets people do almost anything they want with your project, except distributing closed source versions.
Ansible, Bash, and GIMP use the GNU GPLv3.
MIT License

A short and simple permissive license with conditions only requiring preservation of copyright and license notices. Licensed works, modifications, and larger works may be distributed under different terms and without source code.

Permissions
- Commercial use
- Distribution
- Modification
- Private use

Conditions
- License and copyright notice

Limitations
- Liability
- Warranty

MIT License

Copyright (c) [year] [fullname]

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

How to apply this license

Create a text file (typically named LICENSE or LICENSE.txt) in the root of your source code and copy the text of the license into the file. Replace [year] with the current year and [fullname] with the name (or names) of the copyright holders.

Optional steps
Add MIT to your project’s package description, if applicable (e.g., Node.js).
Things.Guide Resources

Things.Guide Resources is a collection of resources on Internet of Things (IoT) and Embedded Machine Learning.

About

Things.Guide is maintained by a small group of applied science teachers trying to share open course materials.

Wiki

The Wiki is a curated collection of links to things we wrote, used for teaching or find interesting.

- Wiki Home

IoT

IoT connects physical things, small devices with sensors and actuators, to the Internet.

- IoT Resources Wiki Page

Embedded ML

Embedded ML allows physical things to learn from sensor input, through inference.
Embedded ML

Thomas Amberg edited this page on May 16 · 114 revisions

Embedded Machine Learning Resources

Articles

Articles, blog posts, whitepapers, etc.

- https://peteward.com/2019/04/14/what-machine-learning-needs-from-hardware/
- ...

Books

- https://www.manning.com/books/deep-learning-with-python
Thanks.

https://github.com/tamberg/fhnw-iot
https://twitter.com/tamberg
https://things.guide