WebThings

Second W3C Workshop on the Web of Things

Ben Francis & Kathy Giori

2019-06-04

Mozilla

“Our mission is to ensure the Internet is a global public resource, open and accessible to all.

An Internet that truly puts people first, where individuals can shape their own experience and are empowered, safe and independent.”

Mozilla IoT Team

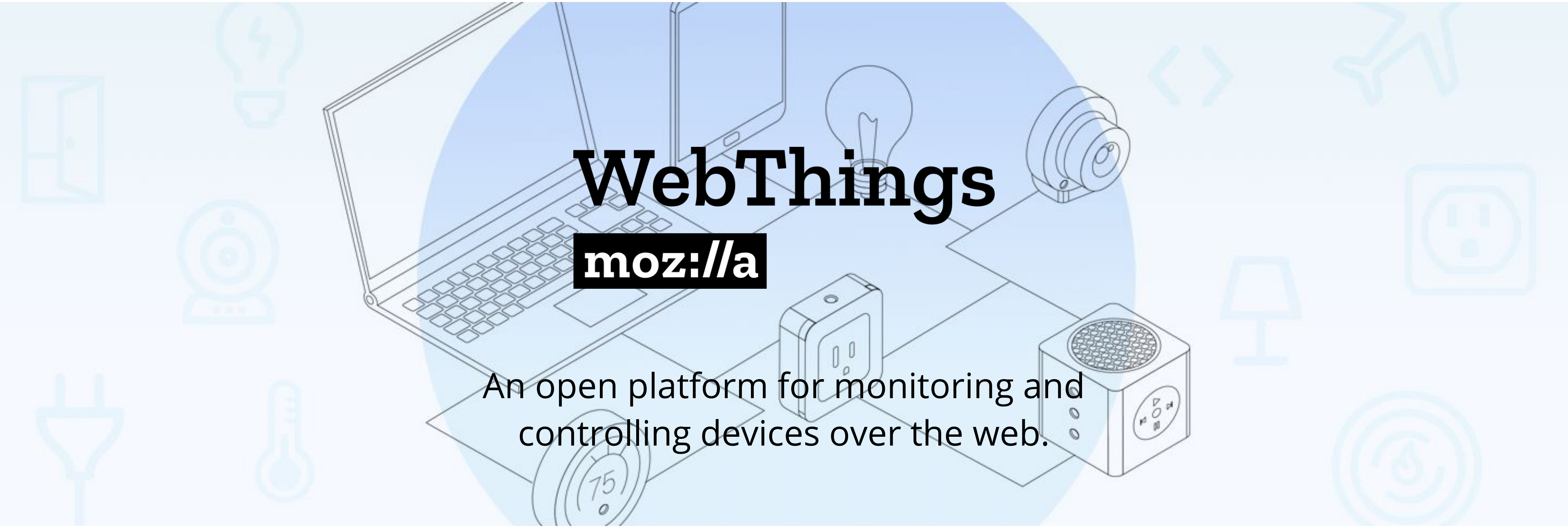
"Our mission is to create a Web of Things implementation which embodies Mozilla's values and helps drive IoT standards for security, privacy and interoperability."

Ben Francis
Product Owner

Kathy Giori
Staff Evangelist

iot.mozilla.org

Mozilla WebThings


WebThings
moz://a
An open platform for monitoring and controlling devices over the web.

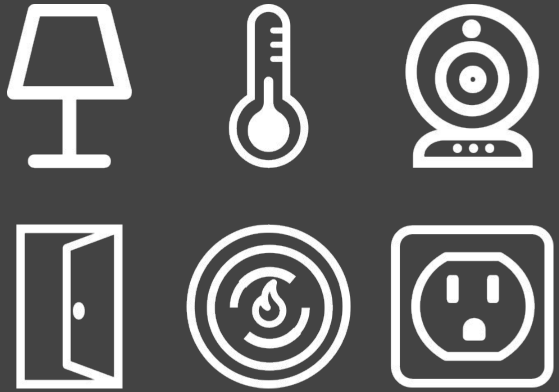
Mozilla's open source implementation of the Web of Things.

Mozilla WebThings Components

WebThings Gateway
moz://a

A software distribution for smart home gateways focused on privacy, security and interoperability

WebThings Framework
moz://a



A collection of reusable software components to help developers build their own web things

moz://a

WebThings Gateway Use Cases

Turn appliances on and off remotely and monitor power consumption.

Turn a light on, sound an alarm or be alerted if motion is detected.

Be alerted on your smartphone if smoke is detected.

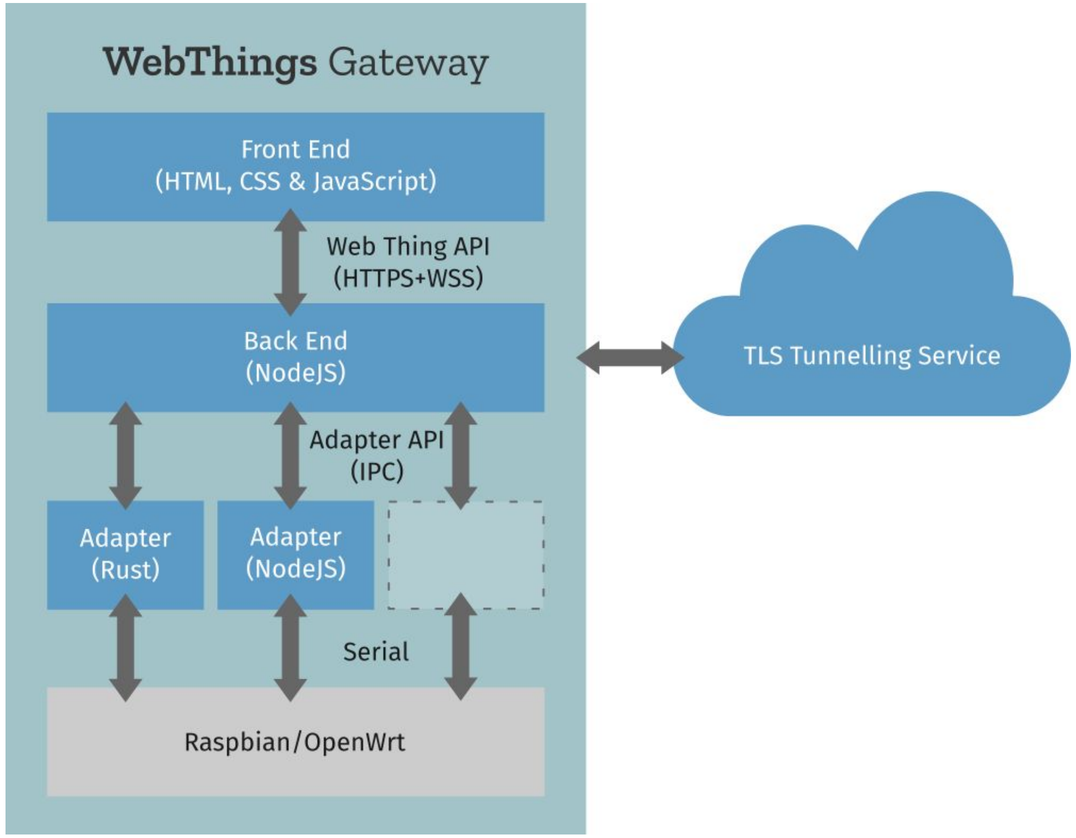
Check what time the kids got home.

Expand your smart home with existing smart home devices, without the need for additional apps.

Authorize third party apps & services to access your home monitoring data.

moz://a

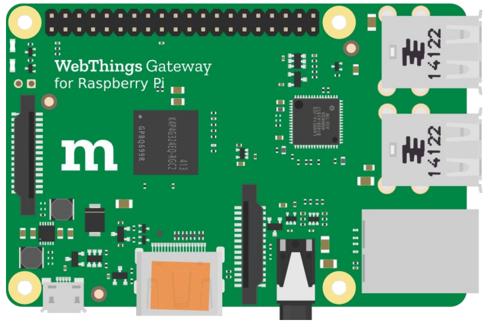
WebThings Gateway Architecture



moz://a

Build Your Own WebThings Gateway

1. Get your hands on a **Raspberry Pi® 3** single board computer. It has WiFi, Bluetooth, GPIO and more built in.

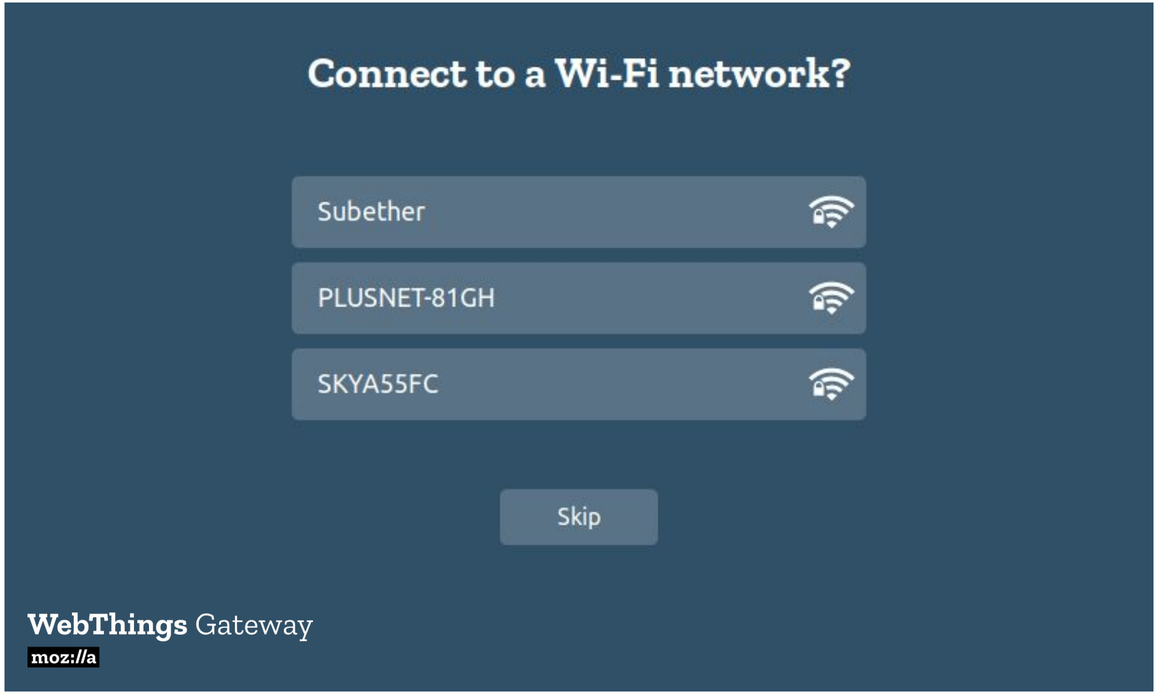


2. To use your gateway with other wireless protocols like ZigBee and Z-Wave you will need USB dongles.

3. Download the pre-built Raspberry Pi OS image from **iot.mozilla.org/gateway** and flash it onto an SD card.

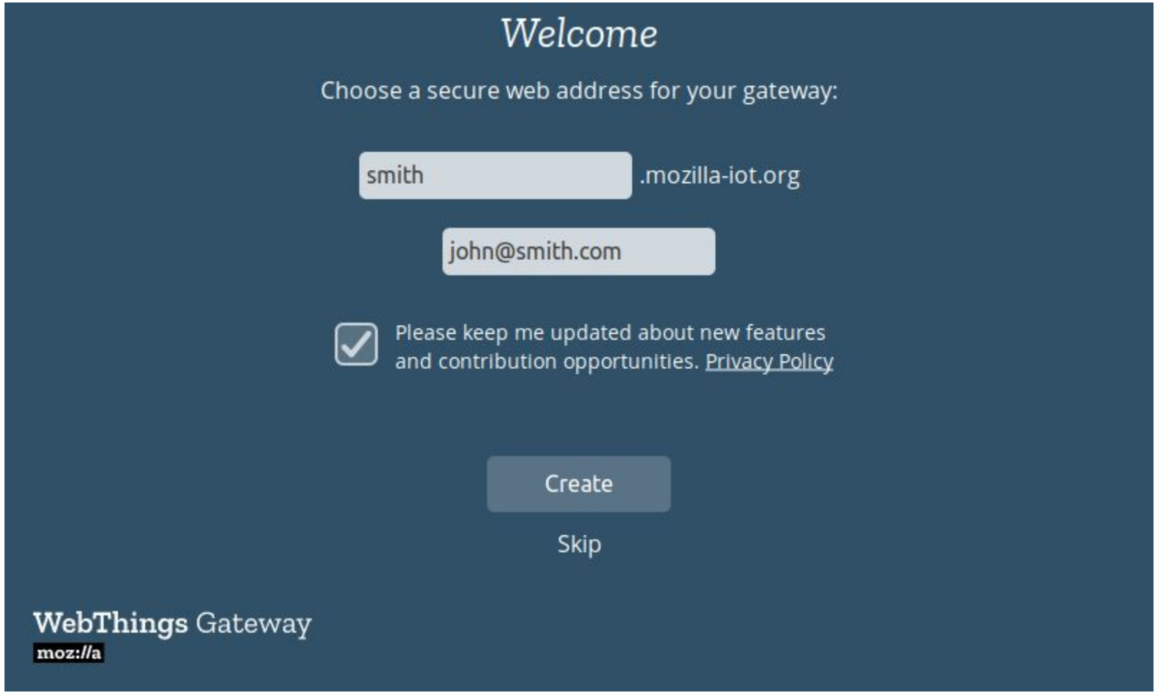
moz://a

WebThings Gateway UI



Connect to Wi-Fi Network

WebThings Gateway UI


Welcome

Choose a secure web address for your gateway:

smith .mozilla-iot.org

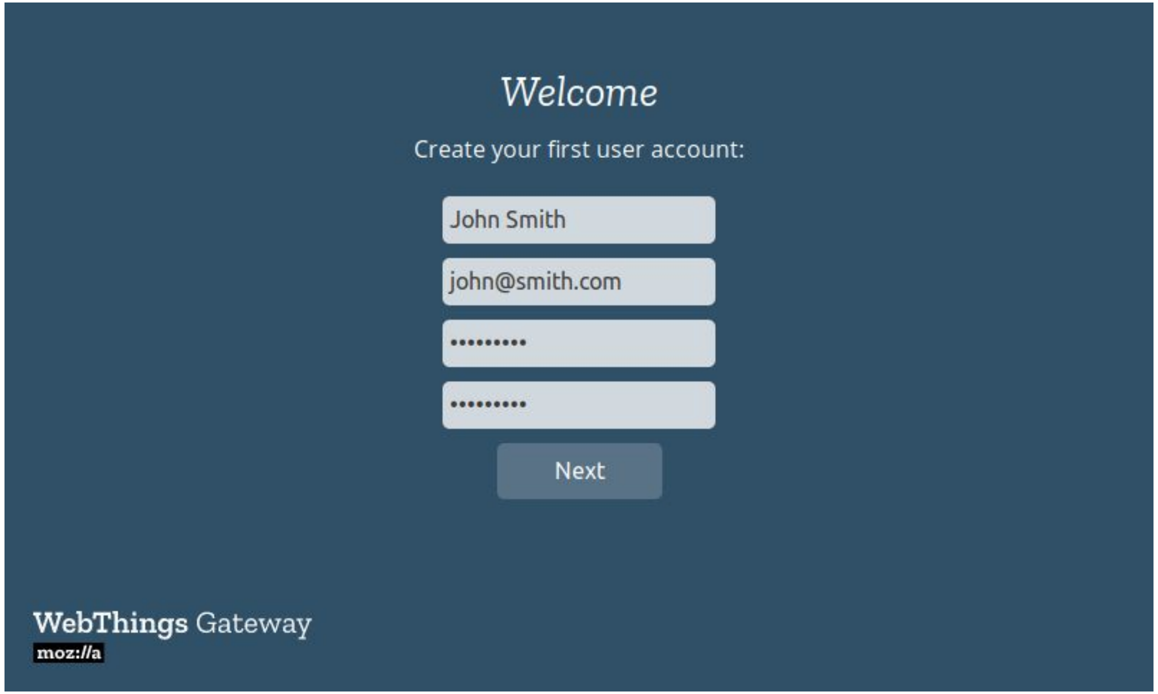
john@smith.com

✓ Please keep me updated about new features and contribution opportunities. Privacy Policy

Choose a Subdomain

moz://a

WebThings Gateway UI


Welcome

Create your first user account:

John Smith

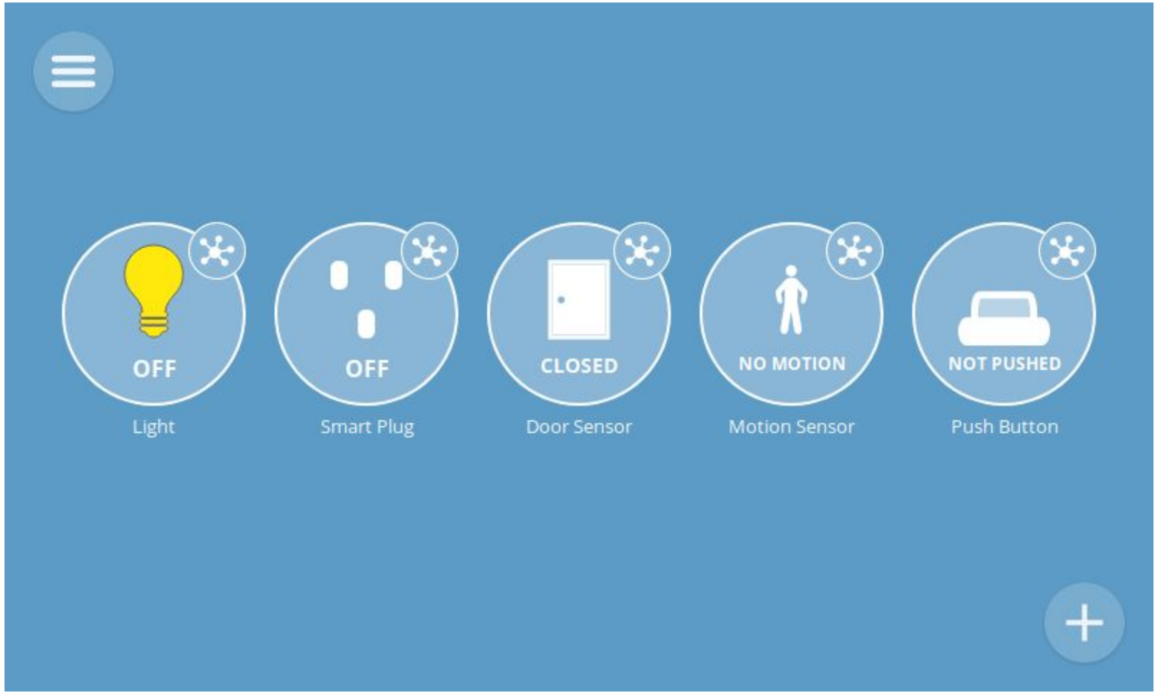
john@smith.com

••••••••••

••••••••••

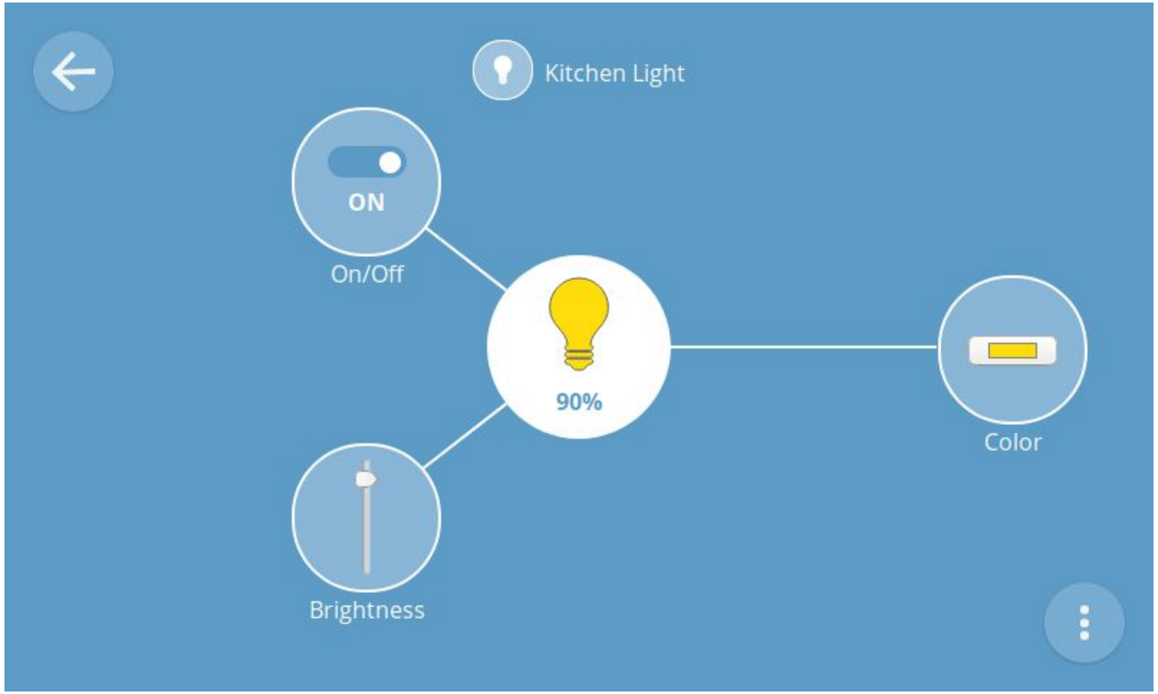
Create User Account

WebThings Gateway UI



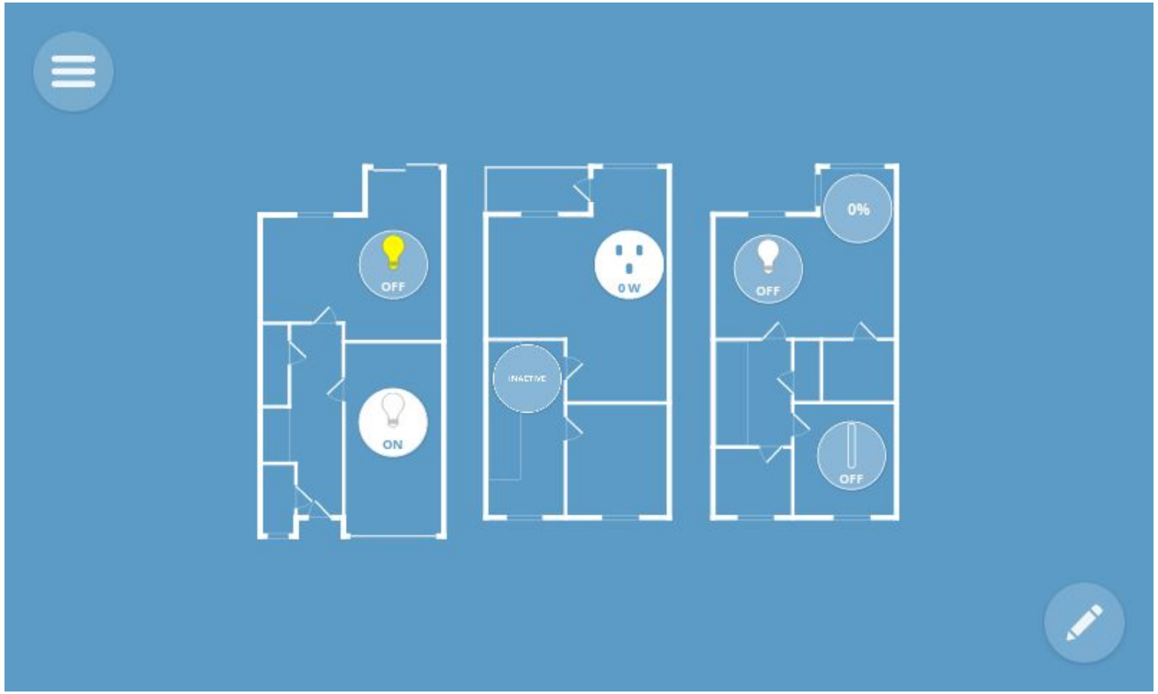
Things UI

WebThings Gateway UI



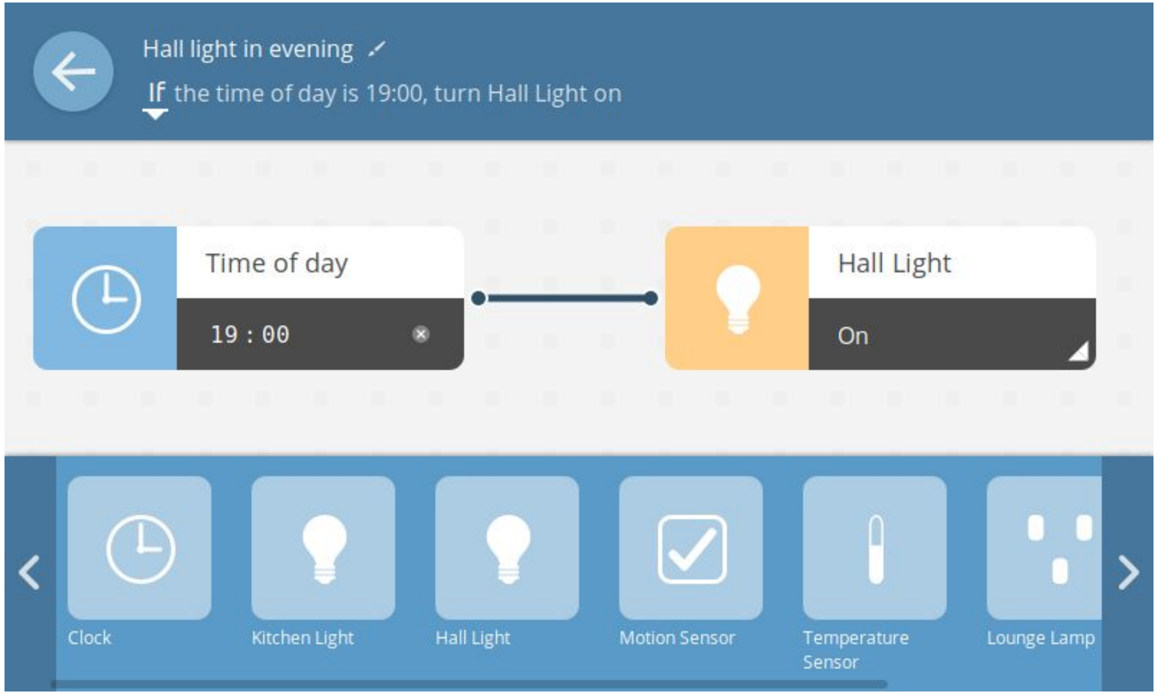
Thing Detail

WebThings Gateway UI



Floorplan

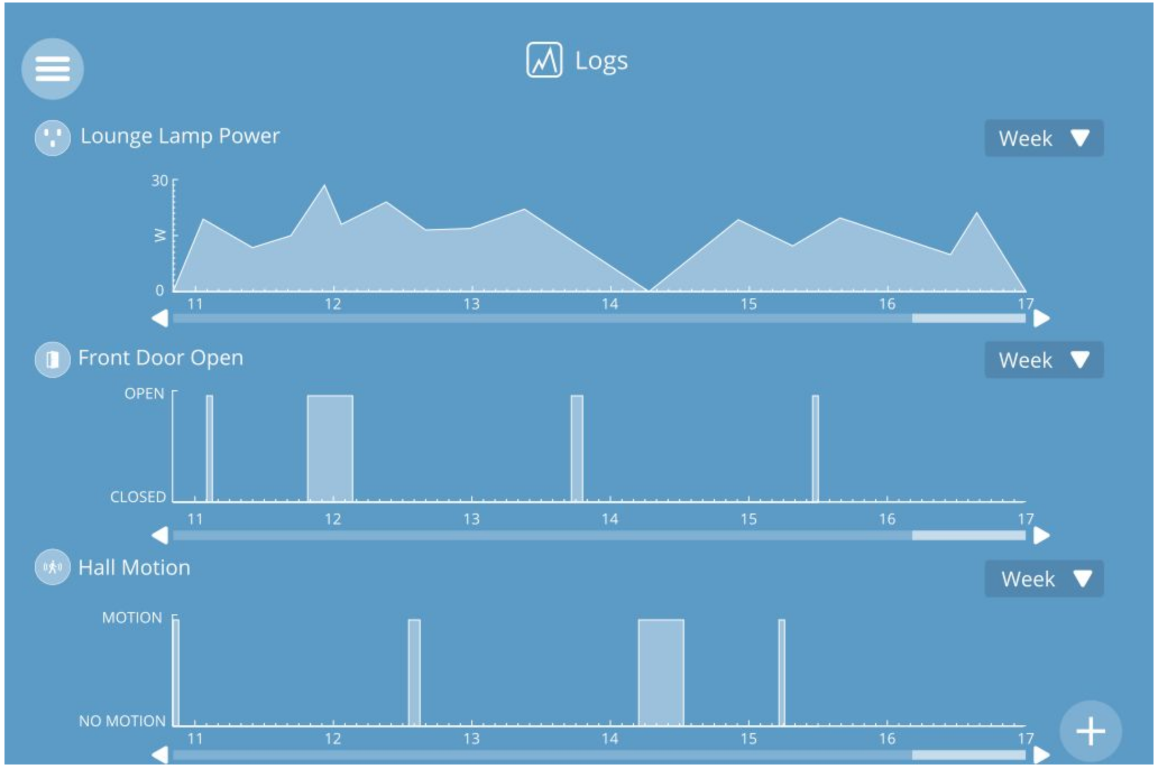
WebThings Gateway UI


Hall light in evening

If the time of day is 19:00, turn Hall Light on

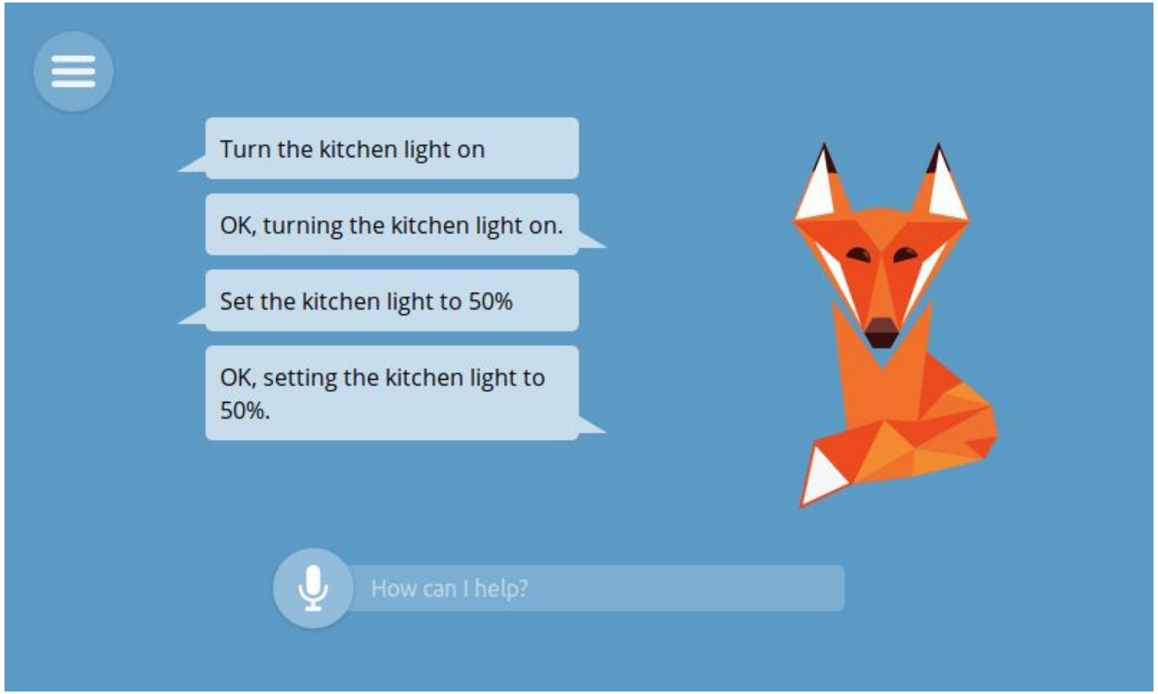
Rules Engine

WebThings Gateway UI



Logging

WebThings Gateway UI


Turn the kitchen light on

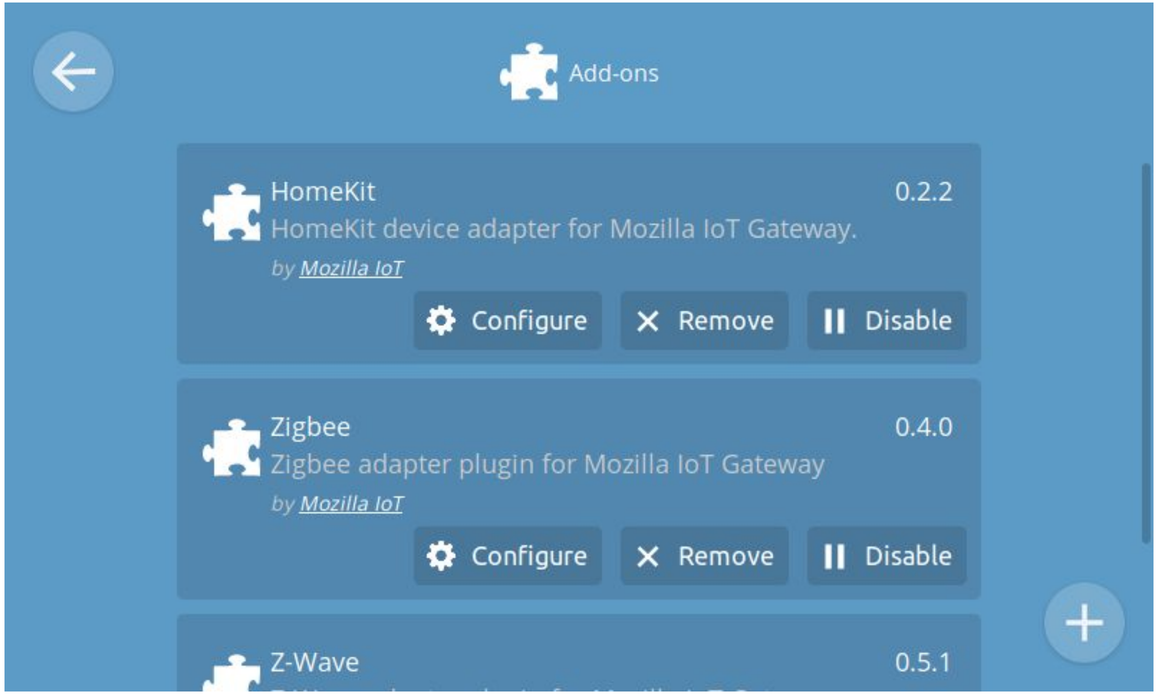
OK, turning the kitchen light on.

Set the kitchen light to 50%

OK, setting the kitchen light to 50%.

Smart Assistant

WebThings Gateway UI



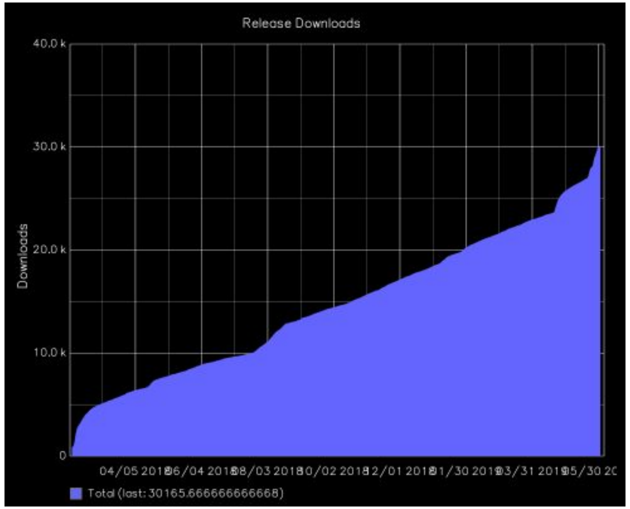
HomeKit 0.2.2
HomeKit device adapter for Mozilla IoT Gateway.
by Mozilla IoT

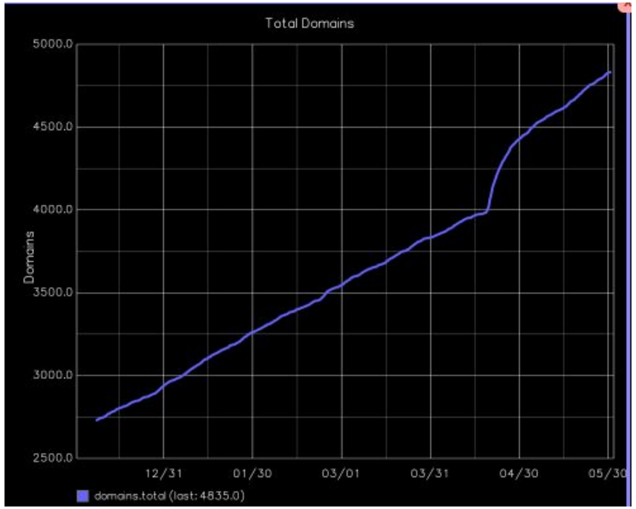
Zigbee 0.4.0
Zigbee adapter plugin for Mozilla IoT Gateway
by Mozilla IoT

Z-Wave 0.5.1

Add-ons

WebThings Gateway Metrics





30k

downloads

5k

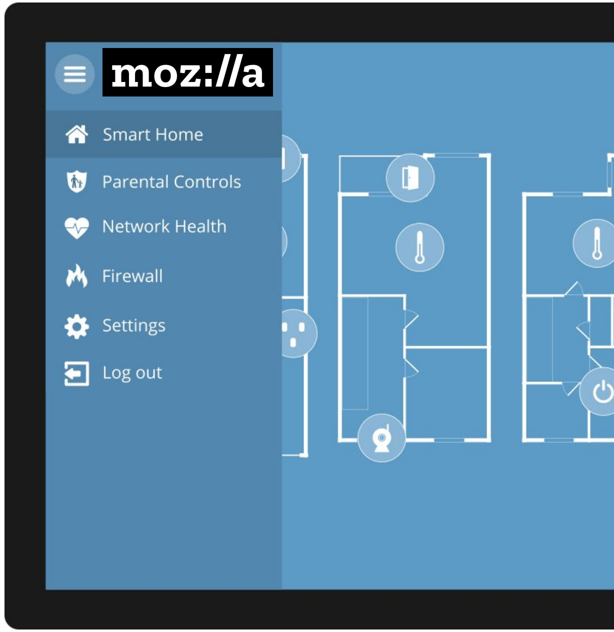
subdomain registrations

Future: Consumer Wireless Routers

Powered by
WebThings
moz://a

Build WebThings Gateway directly into consumer wireless routers. Linux distribution based on OpenWrt.

- Smart Home
- Family Internet Safety
- Home Network Health



moz://a

WebThings Framework

WebThings Framework

moz://a

Web thing libraries for a range of popular programming languages.

Mozilla maintained libraries:

- Node.js
- Python
- Java
- Rust
- Arduino
- MicroPython

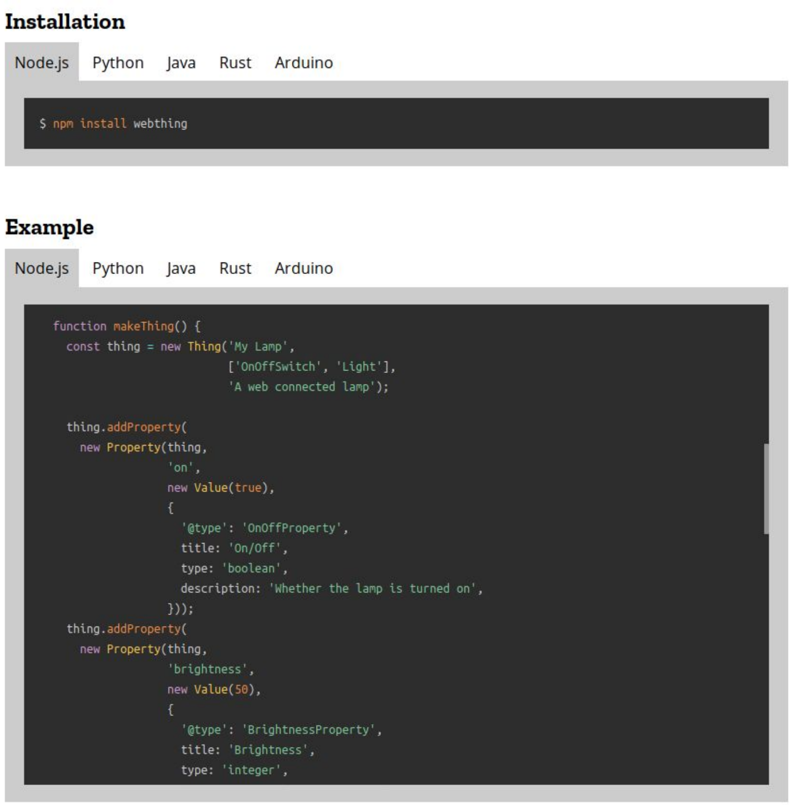
Third Party libraries:

- IoT.js
- Moddable
- C#

iot.mozilla.org/framework

moz://a

WebThings Framework Examples



Installation

Node.js | Python | Java | Rust | Arduino

```
$ npm install webthing
```

Example

Node.js | Python | Java | Rust | Arduino

```
function makeThing() {
  const thing = new Thing('My Lamp',
                         ['OnOffSwitch', 'Light'],
                         'A web connected lamp');

  thing.addProperty(
    new Property(thing,
                 'on',
                 new Value(true),
                 {
                   '@type': 'OnOffProperty',
                   title: 'On/Off',
                   type: 'boolean',
                   description: 'Whether the lamp is turned on',
                 }));
  thing.addProperty(
    new Property(thing,
                 'brightness',
                 new Value(50),
                 {
                   '@type': 'BrightnessProperty',
                   title: 'Brightness',
                   type: 'integer',
```

iot.mozilla.org/framework

moz://a

Challenges

Challenge 1 - HTTPS on Local Networks

Problem:

- HTTPS requires an Internet connection to verify authenticity of certificates
- Local networks can use self-signed certificates but users get scary browser warnings
- Have to choose between secure or local

Solution:

1. ~~Service Workers? (can't be tricked)~~
2. ~~Alt-Svc HTTP header? (only implemented in browsers for HTTP/2)~~
3. Separate .local domain with plain HTTP for local access (mDNS)
4. Home router as local certificate authority?

W3C HTTPS on Local Networks Community Group

moz://a

Challenge 2 - HTTP on Resource Constrained Devices

Problem:

- HTTP is too heavyweight for some constrained devices
- Not well suited to some IoT use cases (e.g. server push)

Solution:

1. Gateway bridges low power wireless protocols (e.g. Zigbee, Z-Wave, Bluetooth) to the web
2. WebSockets can provide server push (alternatives being long-polling,or Server-Sent Events), but still heavyweight
3. CoAP offers a lightweight alternative to HTTP with push, but isn't natively supported in web browsers
4. HTTP/3? HTTP/4?

moz://a

Challenge 3 - Authenticating Web Things

Problem:

- Currently web things built with the WebThings Framework are un-authenticated on the local network
- Rely on the gateway to safely proxy them to the Internet using HTTPS + JSON Web Tokens + OAuth

Solution:

The “security” member of the Thing Description?

Challenge 4 - Scripting API

Problem:

- Current Scripting API aimed mainly at JavaScript
- Programming language agnostic API not really practical
- On the client side existing JavaScript APIs like fetch and WebSocket can be used
- No browser vendor has expressed an interest in implementing the Scripting API directly

Solution:

- WebThings Framework implements different APIs in the flavour of each language via libraries, doesn't follow the Scripting API specification.
- WebThings Gateway uses existing fetch and WebSocket JavaScript APIs on the client side to communicate with the Web Thing API

Challenge 5 - Declarative Protocol Bindings

Problem:

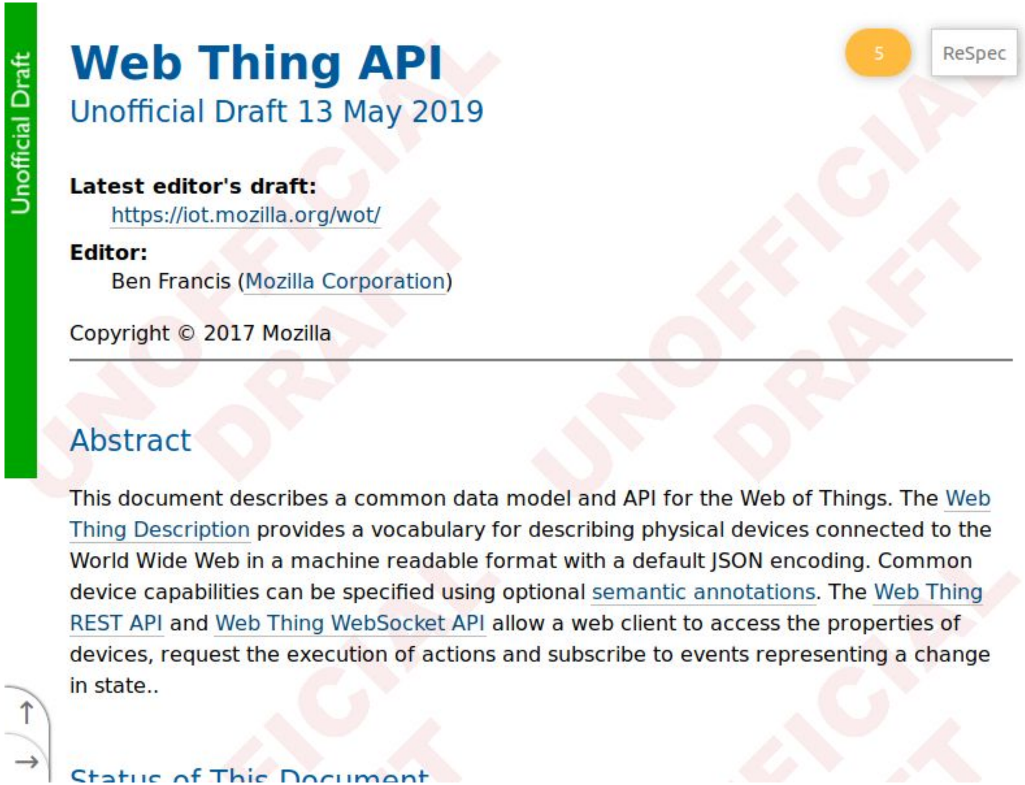
- Protocol binding templates using forms/hypermedia controls (declarative protocol bindings) aim to be flexible enough to use any existing protocol and describe any API/sub-protocol, but in practice we've found they are not expressive enough to do so (e.g. multi-resource interactions, error conditions, WebSocket sub-protocols, non-web)
- This flexibility adds significant complexity to client/gateway implementations
- Enabling divergence in protocols & APIs/sub-protocols for WoT makes ad-hoc interoperability impractical (not every WoT client can talk to every WoT device)

Solution:

- Defined a standard API (concrete protocol binding) for communicating with devices over HTTP & WebSockets
- Use gateway adapters to bridge other existing protocols to this API

moz://a

Web Thing API



Unofficial Draft

5 ReSpec

Web Thing API

Unofficial Draft 13 May 2019

Latest editor's draft:
https://iot.mozilla.org/wot/

Editor:
Ben Francis (Mozilla Corporation)

Abstract

This document describes a common data model and API for the Web of Things. The Web Thing Description provides a vocabulary for describing physical devices connected to the World Wide Web in a machine readable format with a default JSON encoding. Common device capabilities can be specified using optional semantic annotations. The Web Thing REST API and Web Thing WebSocket API allow a web client to access the properties of devices, request the execution of actions and subscribe to events representing a change in state..

Status of This Document

- **Web Thing Description** (with simple web "links" rather than complex "forms")
- **Web Thing REST API** (An HTTP sub-protocol)
- **Web Thing WebSocket API** (a WebSocket sub-protocol)

moz://a

Proposal

- Create a new “**Web Thing Protocol**” specification to complement the WoT Thing Description specification:
    - Web Thing Description - describes a device’s capabilities and provides simple links to URLs on the web to communicate with it
    - Web Thing Protocol - defines standard concrete sub-protocol(s) for communicating with a device over the web (HTTP, WebSockets, and possibly CoAP)
- Encourage the use of **gateways to bridge non-web protocols** (e.g. MQTT, AMQP, Zigbee etc.) and existing web-based cloud APIs to the Web Thing Protocol
- Consider whether a **future version of HTTP** could remove the need for CoAP to further reduce the number of protocols and improve ad-hoc interoperability

moz://a

Charter Proposals

1. Re-charter the WoT Working Group to continue working on the WoT Thing Description specification, and consider how it could be used without the need for protocol binding templates
2. Do not advance the WoT Scripting API and WoT Binding Templates specifications as standards track documents (WoT Architecture document should just be informative)
3.
    a. Incubate concrete WoT protocol bindings (webthing subprotocol for HTTP, WebSockets, CoAP) in the Interest Group with a view to standardisation via the W3C or IETF (and ensure there is a mechanism to hook into the Thing Description)

    and/or

    b. Add a "Web Thing Protocol" specification as a normative deliverable for the new WoT Working Group Charter

Demo & Questions

WebThings

moz://a

iot.mozilla.org @MozillaIoT

moz://a