Willkommen

zum

LoRaWAN – Basteln mit dem Arduino

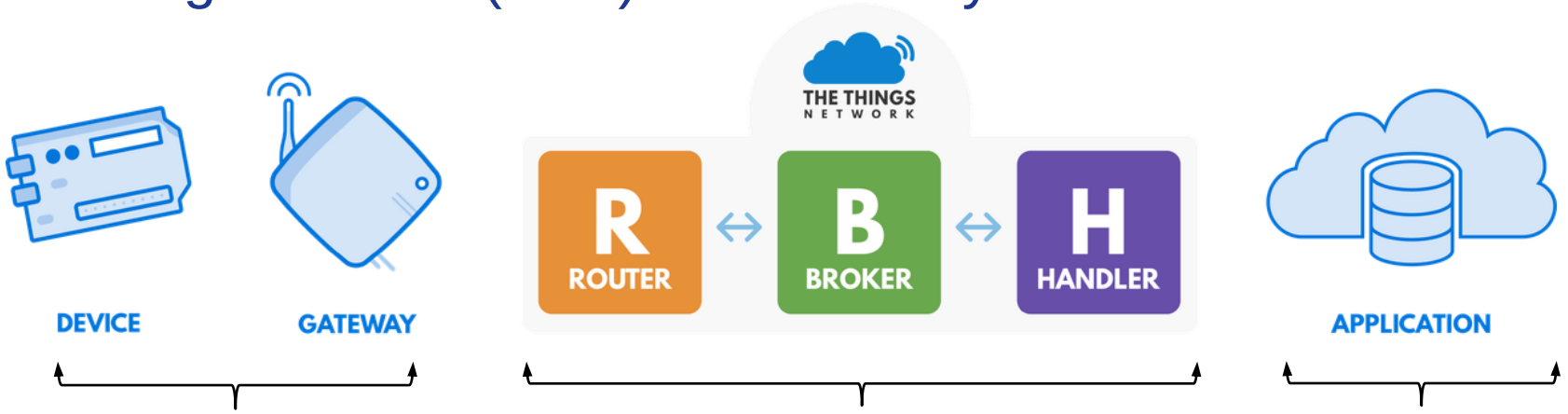
Technologiestiftung Berlin, 20.04.2018

Dr. Christian Hammel
Carolin Clausnitzer
Dr. Benjamin Seibel

THE THINGS NETWORK

TECHNOLOGIE STIFTUNG BERLIN

TheThingsNetwork (TTN) - Community-Netz



Baut, kauft, betreibt,…
der Nutzer

Zentrale Infrastruktur des TTN-Netzwerkes
Accounts, Keys, Routing, Datenhaltung, APIs

Schreibt u. betreibt
der Nutzer

Die zentrale Struktur kann jeder kostenlos nutzen. Sie bietet Geräte-Keys, Nutzerverwaltung, Paketrouting und API (HTTP, MQTT).

Kostenpflichtig: weitere Einnahmequellen der TTN Industries wie private Server, Integration in kommerzielle Nutzungen,…

LoRaWAN-Sprech: Device = Mote = Node = Endgerät; Gateway = Basisstation (= Router)

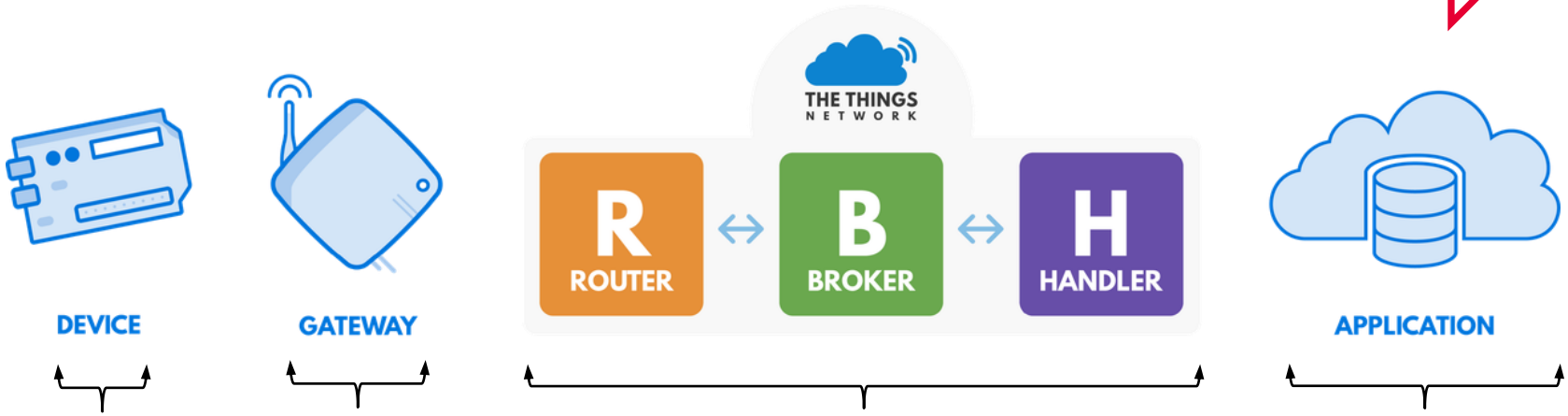
Stand:
- 500 Communities weltweit, 37.000 User, 3.550 Gateways, 20.000 Applications, Schwerpunkte NL, BE, CH, UK
- TTN-Berlin: 55 Contributors, 45 Gateways

Bild: Johan Stocking (TTN) / eigene Darstellung

TECHNOLOGIE STIFTUNG BERLIN

Worum geht es heute? Was machen wir?

Wir übertragen selbst gemessene Daten vom Arduino ins Netz (uplink)



Unser Arduino!

Vorhanden

Das benutzen wir einfach.

Hier spielt die Musik, aber heute nicht.

Wir schließen einen Sensor an.
Wir messen eine Temperatur.
Wir schalten eine LED ein.
Wir melden das Gerät bei TTN an.
Wir senden die Temperatur ins TTN.
Wir nutzen Beispielcode.

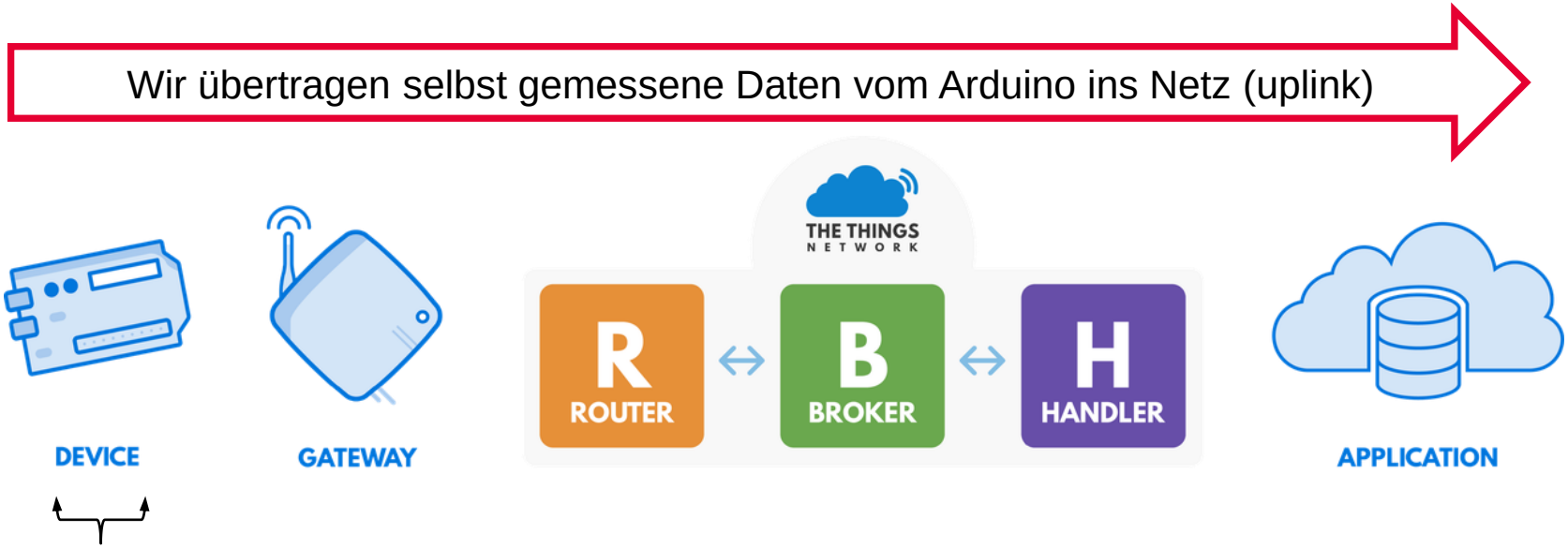
Teilnehmer sollten einen Account haben, damit sie ihr node anmelden und auf ihre Daten zugreifen können.

Wie melde ich den Arduino an?
Wo sehe ich meine Daten?

Wenn wir Zeit haben, lesen wir unsere Daten mit einem MQTT-Viewer.

TECHNOLOGIE STIFTUNG BERLIN

Was genau soll unser Device machen?



Unser Arduino

- Misst die aktuelle Temperatur mit einem Sensor
- Blinkt mit einer Kontroll-LED, wenn die Temperatur über 30° ist, damit wir direkt sehen können, ob unsers Schaltung und unser Code funktionieren
- Überträgt die Temperatur ins TTN (genauer: zum TTN-Broker)

TTN

- Zeigt uns in der Ansicht „Console“ unsere Temperatur an

MQTT-Viewer

- Zeigt die Daten im Browser oder auf dem Smartphone an, wenn noch Zeit ist

TECHNOLOGIE STIFTUNG BERLIN

Board zusammenbauen

1. **LoraRaWAN-Shield auf den Arduino montieren**
   (so aufstecken, dass er gewaltfrei passt und dass die Anschlüsse auf Arduino und Shield in Deckung sind)

2. **ANTENNE ANSCHRAUBEN !!!!!**
   (NIE ohne Antenne betreiben, das kann das Board zerstören)

Kommunikation Lappi / Arduino testen

1. **Arduino-IDE starten**

2. **Arduino über USB-Kabel anschließen**
   (NIE ohne Antenne!)

3. **Kommunikationstest**
   „Werkzeuge | Board | Mega“, „Werkzeuge | Prozessor | Mega2560“; „Werkz. | Port | $richtigerPort“;
   „Werkzeuge | Boardinformationen holen“
   Wenn etwas kommt: alles richtig angeschlossen,
   Wenn nicht oder Fehler: richtiger Port? Strom am Arduino? USB-Anschluss freigegeben (Linux)?

4. **LMIC (LoRaWAN-Bilbliothek einbinden)**
   „Sketch | Bibliothek einbinden | IBM LMIC framework“

5. **Testcode hochladen**
   Testcode der TSB (kommunikationstest_lappi_arduino.ino) in die IDE laden; „Sketch | hochladen“
   Keine Fehlermeldungen? Alles prima!

6. **Test**
   „Werkzeuge | serieller Monitor“: Wenn hier bei 115200 baud sinnvoller output kommt, ist alles prima.

Arduino-Code

1. **Arduino-Code:** Die IDE kompiliert den c-ähnlichen Code in Maschinencode, der dann auf den Arduino geladen wird. Deshalb kann man Code auch nicht von Arduino herunterkopieren.

2. **Programmaufbau:** Es gibt immer mindestens drei Teile,
   - einen Header, in dem includes, Variablen u.ä. definiert werden,
   - einen Teil void setup (){}, der genau einmal abgearbeitet wird und
   - einen Teil void loop (){}, der ständig wiederholt wird.

   Zusätzlich kann man an beliebiger Stelle eigene Funktionen ergänzen, die sich aus anderen Programmteilen heraus aufrufen lassen.

3. **Syntax**
   Funktionen: void FUNKTIONSNAME (Rückgaben) {Code der Funktion}
   Befehlszeilen: enden immer mit ;
   Kommentar: wird mit // eingeleitet und nicht kompiliert
   Konstanten und Variablen: Sind immer 2 Byte (16 Bit), wenn man nichts anders definiert.

4. **Prográmmchen (Sketches)**
   Konvention: enden auf .ino
   stehen in einem Ordner namens Arduino-Sketchbook in eurem Userverzeichnis.

5. **Beispiele für heute:** https://github.com/technologiestiftung

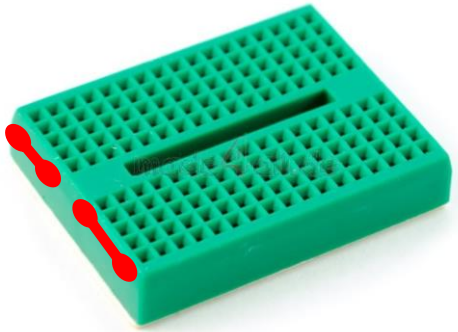
TECHNOLOGIE STIFTUNG BERLIN

Schaltungen aufbauen

Kontroll-LED: Arduino und Breadboard:

• Pin GND (auf der „Digitalseite“ vom Arduino –> Vorwiderstand –> kurzes LED-Bein (sw. Kabel, wenn da)

• Pin 5 (Arduino, Digitalseite) -> langes LED-Beinchen (rot für „+“, wenn verfügbar)



Diese 5 Anschlüsse sind verbunden.

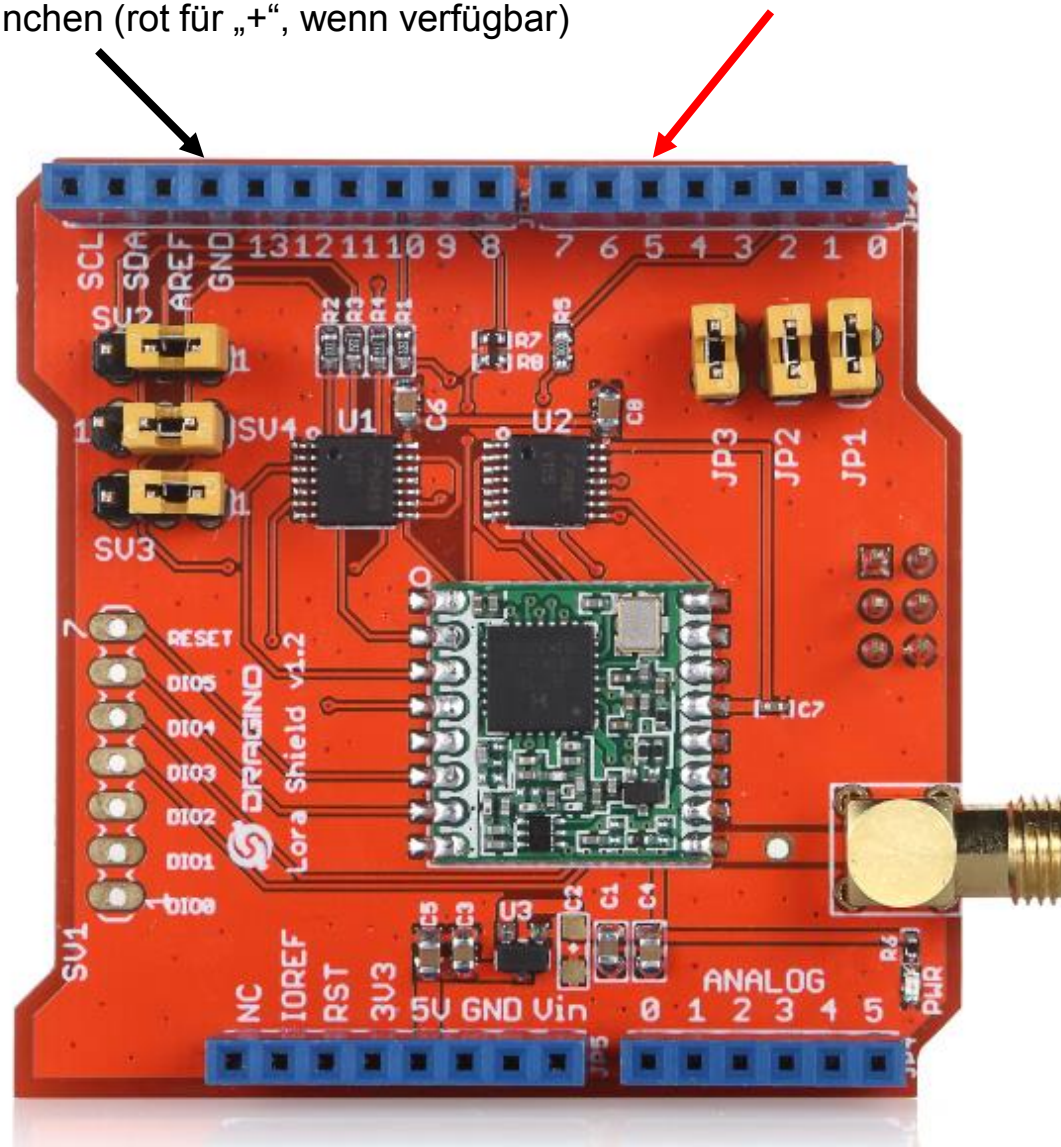
Sensor am Arduino anschließen:

• GND an GND (Analogseite)

• 5V an 5V (Analogseite)

• Messausgang auf Analogeingang 0

Ansicht:
flache Seite oben



Arduino bei TTN anmelden und Daten senden

1. **Bei TTN einloggen (wer keinen Account hat, muss sich einen anlegen)**
   TTN-Konsole:
   Anwendung anlegen,
   device anlegen,
   Settings, ABP, generiert Schlüssel und Device-ID

2. **Arduino IDE starten**
   TSB-Mustercode (temperatur_basteltreff.ino) in die IDE **aber noch nicht auf den Arduino** laden
   (die Schlüssel im Beispiel sind fake und produzieren Fehler).
   Die Schlüssel Network Session Key, Appskey und Device ID (=EUI) in den Mustercode einpflegen.
   Eine beliebige interne ID könnt ihr einbauen, wenn ihr mehrere Nodes auseinanderhalten wollt.

3. **Code hochladen**
   Wenn eure Schlüssel richtig eingetragen sind, dann seht ihr im seriellen Monitor Daten, die ihr erwartet und in eurer Konsole Daten, die noch nicht sinnvoll sind.
   Wenn der Sensor lange genug in der Hand war (>30°) geht die LED an.

4. **Payload-Decoder einrichten**
   Damit die an TTN gesendeten Daten wieder in eine sinnvolle Form kommen, legt ihr über die TTN-Konsole einen payload-decoder an. Code dafür (javascript) steht als Kommentar im Arduino-Testcode.

5. **Code verstehen**
   versuchen wir direkt im Code, dafür sind Kommentare im Code.

TECHNOLOGIE STIFTUNG BERLIN

Daten ansehen und prüfen

1. **Direkt bei TTN**
   TTN-Konsole: Bei eurer Application / Eurem Device könnt ihr die Daten live sehen und wenn der Decoder funktioniert, werden sie auch entschlüsselt.

2. **Von TTN auslesen**
   über http: geht, sprengt hier aber schnell den Rahmen
   über MQTT: für diesen Standard gibt es einige fertige Viewer.

3. **MQTT**
   liefert Datenfelder, die ein Viewer „abonniert“ und anzeigt. Ein Datenfeld hat Name und Wert und heißt in MQTT-Sprech „Topic“.
   Da wir eine End-zu-End-Verschlüsselung haben, braucht der Viewer einen Schlüssel zum Zugriff auf die Daten.
   Unsere Topics sehen so aus: AppID/devices/deviceID/up (/EUERDATENFELD)
   Wenn ihr viele devices habt, sollte das so aussehen AppID/devices/+/up (/EUERDATENFELD)
   Zugang: Host: eu.thethings.network ; Port: 1183 ; Username; App-ID ; Password: App-Access Key

4. **MQTT mit Chrome:** Plugin MQTT-Lens

5. **MQTT mit Android-Apps: MQTT Dash**, Linear MQTT Dashboard, MyMQTT
   (Tipp: App-Access-Key abtippen geht immer schief! Browser oder TTN Mobile können copy und paste)

TECHNOLOGIE STIFTUNG BERLIN

Coole Anwendungen? Tolle Bildungsmaterialien?

Daten in der Konsole sind toll aber mäßig sexy.

Wer mehr mit uns ausprobieren mag, ist herzlich willkommen.

Beispiele für schicke LoRaWAN-Anwendungen von Bürgern:

- Umweltsensoren: https://skopjepulse.mk/
- Badewasser im Dortmund-Ems-Kanal: http://wiekaltistderkanal.de/ui/#/0

Wer mit uns Bildungsmaterial und Unterlagen für die Hackingbox IoT entwickeln mag, ist ebenfalls herzlich willkommen.

Carolin Clausnitzer
clausnitzer@technologiestiftung-berlin.de
@caroclausnitzer

TECHNOLOGIE STIFTUNG BERLIN