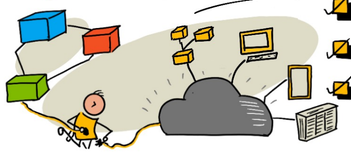
CONNECT

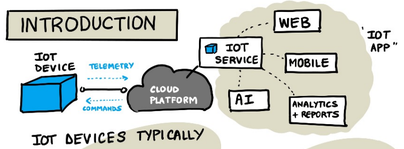
YOUR DEVICES TO THE INTERNET

LEARN TO

- SEND & RECEIVE MESSAGES
- CONNECT LIGHT TO MQTT BROKER
- CONNECT DEVICE TO INTERNET



INTRODUCTION

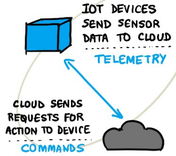


IOT DEVICES TYPICALLY CONNECT TO A SINGLE CLOUD IOT SERVICE – USING A STANDARD COMMUNICATION PROTOCOL

THAT SERVICE THEN CONNECTS TO THE REST OF YOUR IOT APP!

IN THIS LESSON

LET'S EXPLORE HOW DEVICES SEND/RECEIVE MESSAGES...

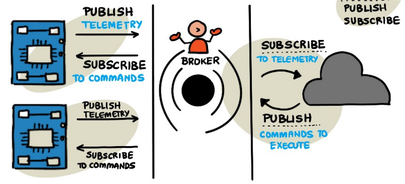


- COMMUNICATION PROTOCOLS
- MESSAGE QUEUING TELEMETRY TRANSPORT (MQTT)
- TELEMETRY
- COMMANDS

COMMUNICATION PROTOCOL

THE MOST POPULAR PARADIGM IS PUB/SUB

PUBLISH SUBSCRIBE



COMMUNICATION PROTOCOL

PUBLISH/SUBSCRIBE MESSAGING PROTOCOLS INVOLVE A BROKER

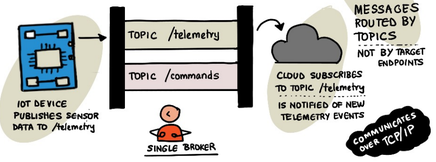
EX: MQTT

OTHER COMMUNICATION PROTOCOLS INCLUDE:

- ADVANCED MESSAGE QUEUING PROTOCOL — AMQP
- HYPERTEXT TRANSFER PROTOCOL (SECURE) — HTTP / HTTPS

MESSAGE QUEUEING TELEMETRY TRANSPORT | MQTT

- SINGLE BROKER
- MULTIPLE CLIENTS
- NAMED TOPICS

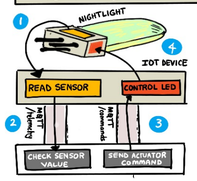


LET'S ASSUME YOU HAVE A LOT OF IOT DEVICES

HOW DO YOU MAKE SURE YOUR MQTT BROKER CAN HANDLE ALL MESSAGES?

CONNECT YOUR DEVICE TO AN MQTT BROKER

CONNECT YOUR IOT NIGHTLIGHT TO THE INTERNET, ENABLE IT TO BE REMOTE CONTROLLED



ECLIPSE MOSQUITTO = OPEN SOURCE PUBLIC MQTT BROKER YOU CAN USE FOR QUICK TESTS

CAUTION: NOT SECURE

DO RESEARCH — WHAT SCENARIO REQUIRES SENSOR DATA EVALUATION FROM MULTIPLE SENSORS BEFORE ACTUATOR COMMAND IS SENT?

A DEEPER DIVE INTO MQTT

- MQTT TOPICS CAN HAVE A HIERARCHY
- MESSAGE "QOS" DETERMINES DELIVERY GUARANTEES

① /telemetry /#

② /telemetry humidity

③ /telemetry temperature

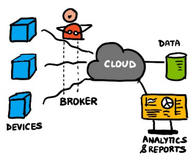
② and ③ are subtopics Publish/subscribe to get only messages from that subset

① is a wildcard Subscribe to get messages from ALL subtopics!

A DEEPER DIVE INTO MQTT

MQTT DOES NOT ACTUALLY "QUEUE" UP MESSAGES FOR SUBSCRIBED CLIENTS

TELEMETRY

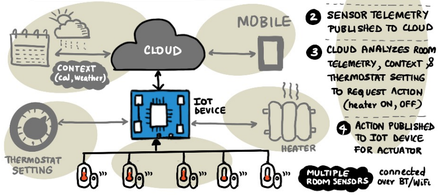


ORIGINS

"TELEMETRY" = GREEK WORD MEANING "TO MEASURE REMOTELY"

IT IS THE ACT OF GATHERING DATA FROM SENSORS AND SENDING IT TO THE CLOUD FOR PROCESSING

TELEMETRY EXAMPLE



1. INTERNET CONNECTED THERMOSTAT DEVICE
2. SENSOR TELEMETRY PUBLISHED TO CLOUD
3. CLOUD ANALYZES ROOM TELEMETRY, CONTEXT & THERMOSTAT SETTING TO REQUEST ACTION (heater ON, OFF)
4. ACTION PUBLISHED TO IOT DEVICE FOR ACTUATOR

connected over BT/Wifi

TELEMETRY EXAMPLE

THE INTERNET CONNECTED THERMOSTAT IOT DEVICE HAS

- TEMPERATURE SENSORS IN THERMOSTAT, IN REMOTE ROOM A, IN REMOTE ROOM B
- ACTUATOR TO HEATER TURN IT ON, TURN IT OFF, INCREASE/DECREASE SETTING

TELEMETRY DATA EXAMPLE

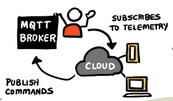
- thermostat_temperature — value = 18°C — measured by thermostat itself
- living room_temperature — value = 19°C — measured by room A temp sensor
- bedroom_temperature — value = 21°C — measured by room B temp sensor

SEND TELEMETRY FROM IOT DEVICE

NEXT TASK: SEND LIGHT-LEVEL TELEMETRY TO MQTT

- SEND LIGHT LEVEL TELEMETRY TO /telemetry TOPIC ON MQTT
- DATA SENT ENCODED AS JSON (KEY/VALUE PAIRS)
- FOLLOW THE APPROPRIATE GUIDE FOR NEXT STEPS (ARDUINO OR SINGLE BOARD COMPUTER OPTIONS)

RECEIVE TELEMETRY FROM MQTT BROKER



"SERVER" CODE THAT SUBSCRIBES TO TELEMETRY, ANALYZES CONTEXT, SENDS DECISION (COMMANDS) BACK TO IOT DEVICE

TELEMETRY HAS LIMITED VALUE WITHOUT SOME WAY TO PROCESS IT FOR DECISIONS

SERVER TYPICALLY RUNS IN CLOUD – FOR NOW, RUN IT IN LOCAL SERVER

INSTALL & CONFIGURE PYTHON ENV

Here is our recommended development setup!

1. INSTALL PYTHON & PIP — CONFIGURE USING PIP INSTALL PAHO-MQTT PKG
2. INSTALL VS CODE — POPULAR EDITOR. INSTALL PYLANCE EXTENSION

REFER TO GUIDE FOR DETAILS...

WRITE THE SERVER CODE

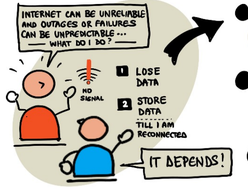
1. CREATE APP.PY
2. LAUNCH VS CODE
3. CHECK: PYTHON ENV RUNS
4. KILL PRE-EXISTING VS CODE TERMINAL
5. LAUNCH VS CODE TERMINAL (active env)
6. EDIT APP.PY CODE
7. RUN APP.PY
8. VALIDATE SETUP (messages in terminal)

DONE!

HOW OFTEN SHOULD TELEMETRY BE SENT?

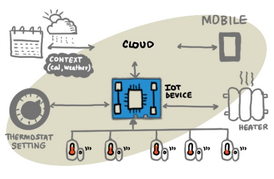
- HIGHER FREQUENCY HAS BETTER RESPONSES TO CHANGE BUT USES MORE RESOURCES!
- THERMOSTAT EXAMPLE
  - * EVERY SECOND = TOO MUCH
  - * ONCE A DAY = TOO LITTLE
  - * EVERY FEW MINUTES = ENOUGH!
- BETTER SAFE THAN SORRY More frequent if mission critical

LOSS OF CONNECTIVITY



- THERMOSTAT: TEMPERATURE HAS LESS VALUE WITH TIME – DON'T STORE IT!
- MACHINERY: OPERATIONAL DATA CAN ALWAYS BE USED FOR PROFILING AND PREDICTIVE MAINTENANCE – STORE IT!
- MQTT WON'T DO THIS FOR YOU. WRITE CLIENT/SERVER CODE...

SEND COMMANDS TO MQTT



COMMANDS ARE: MESSAGES FROM CLOUD TO DEVICE WITH A REQUEST FOR ACTION (e.g. ACTUATOR REQUEST)

EDIT SERVER CODE TO SEND COMMAND TO SPECIFIC TOPIC...

EXPLORE HOW COMMANDS CAN BE DIRECTED AT ONE VS MANY USING MULTIPLE TOPICS

HANDLE COMMANDS ON THE IOT DEVICE

COMMANDS ARE PUBLISHED BY SERVER VIA MQTT TOPIC

IOT DEVICES MUST

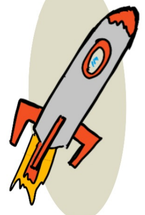
- SUBSCRIBE TO RELEVANT TOPIC
- LISTEN FOR MESSAGES ON TOPIC
- HANDLE MESSAGES – TAKING THE APPROPRIATE ACTION TO CONTROL LED

LOSS OF CONNECTIVITY

CONTEXT IS KEY

- DOES THE LATEST COMMAND OVERRIDE PREVIOUS ONES? IGNORE OLD COMMANDS
- DO COMMANDS NEED TO BE PROCESSED IN ORDER? (DO X, THEN DO Y)

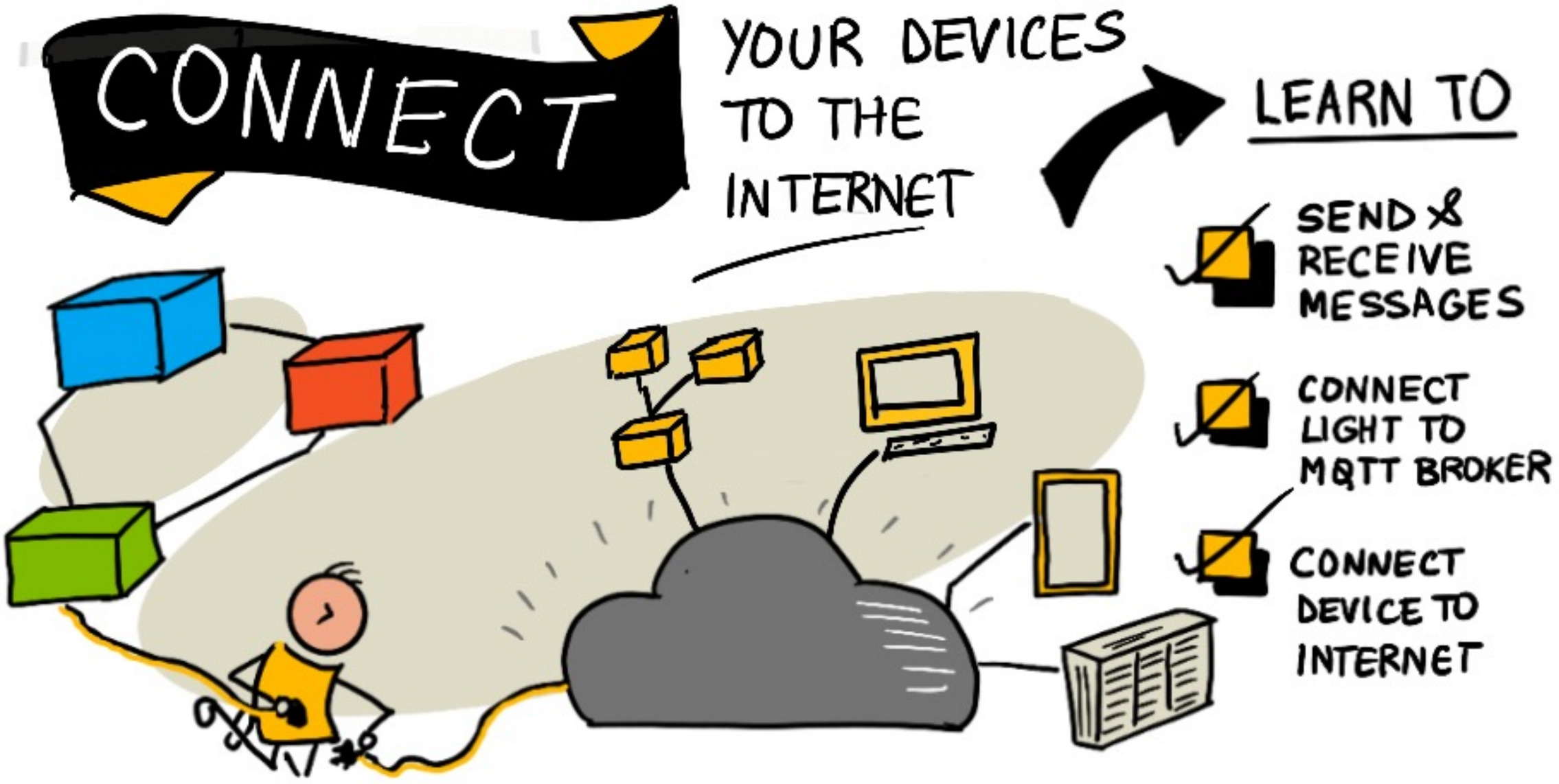
SEND/RESEND COMMANDS IN THE DESIRED ORDER



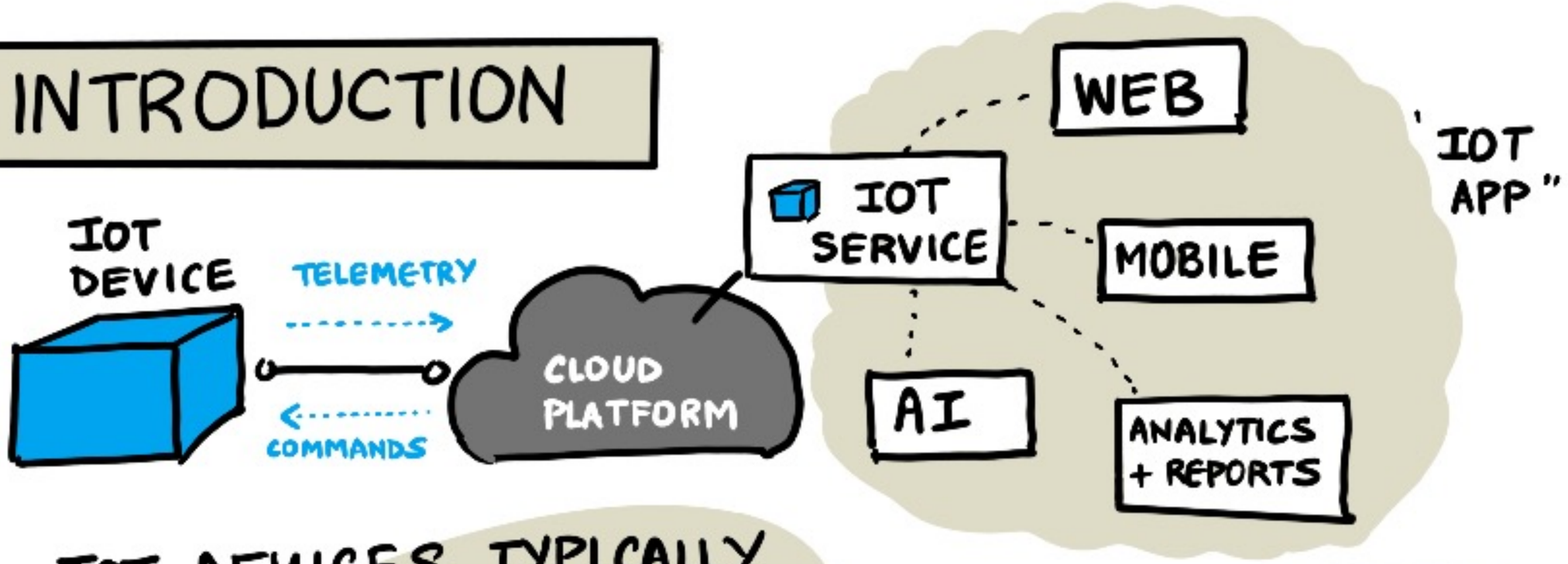
CHALLENGE

- LOOK AT YOUR PREVIOUS LISTS OF IOT DEVICES
- FOR EACH DEVICE – WHAT MESSAGES MIGHT THEY BE SENDING? RECEIVING?
- ARE THEY SECURE? WHAT TELEMETRY DO THEY CAPTURE? SEND?

NEXT: TO THE FARM TO PREDICT PLANT GROWTH


CONNECT
YOUR DEVICES TO THE INTERNET
LEARN TO
SEND & RECEIVE MESSAGES
CONNECT LIGHT TO MQTT BROKER
CONNECT DEVICE TO INTERNET

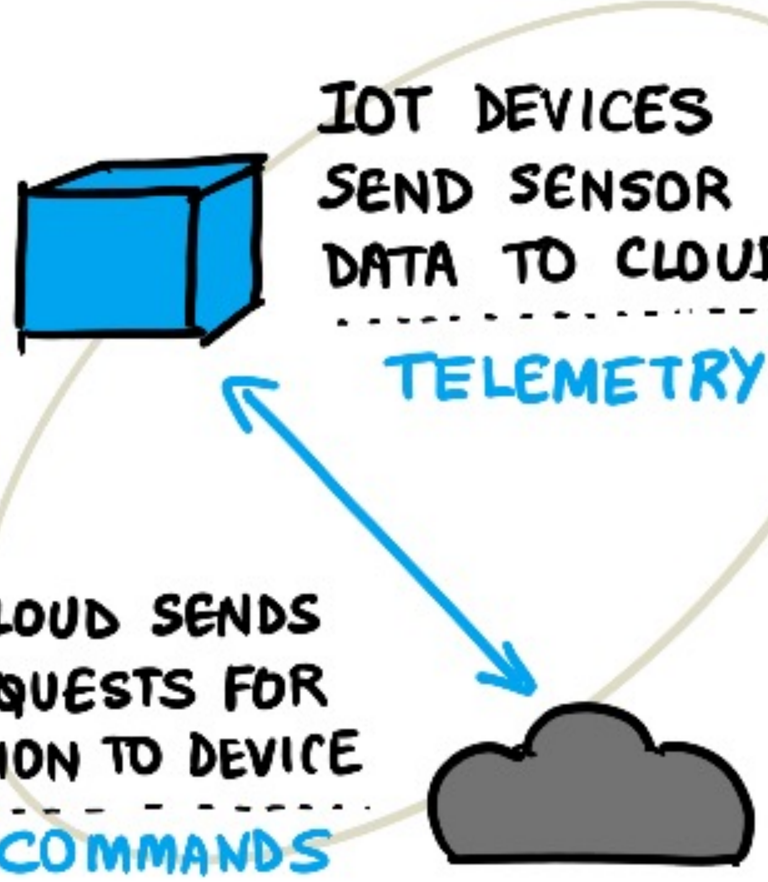
INTRODUCTION



IOT DEVICES TYPICALLY CONNECT TO A SINGLE CLOUD IOT SERVICE – USING A STANDARD COMMUNICATION PROTOCOL

THAT SERVICE THEN CONNECTS TO THE REST OF YOUR IOT APP!

IN THIS LESSON



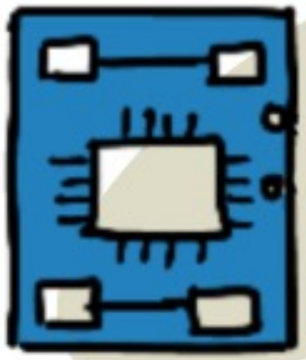
LET'S EXPLORE HOW DEVICES SEND/RECEIVE MESSAGES....

- COMMUNICATION PROTOCOLS
- MESSAGE QUEUING TELEMETRY TRANSPORT (MQTT)
- TELEMETRY
- COMMANDS

COMMUNICATION PROTOCOL

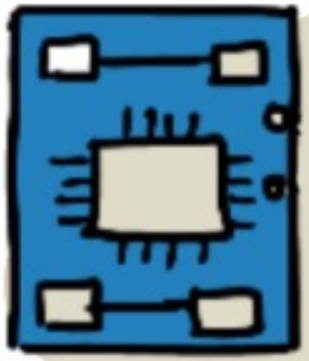
THE MOST POPULAR PARADIGM IS PUB/SUB

PUBLISH
SUBSCRIBE



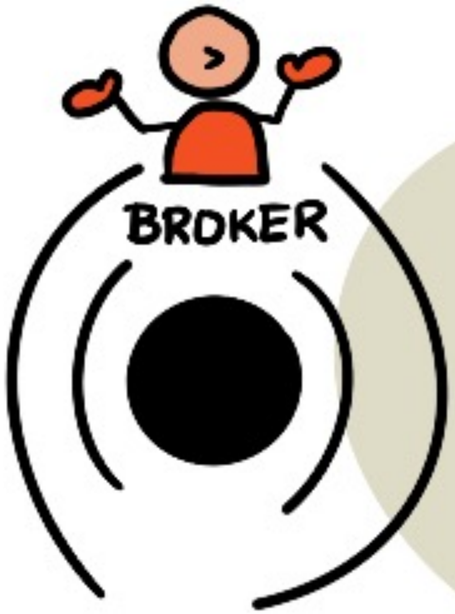
PUBLISH
TELEMETRY

SUBSCRIBE
TO COMMANDS



PUBLISH
TELEMETRY

SUBSCRIBE
TO COMMANDS



SUBSCRIBE
TO TELEMETRY

PUBLISH
COMMANDS TO
EXECUTE

COMMUNICATION PROTOCOL

PUBLISH/SUBSCRIBE MESSAGING PROTOCOLS INVOLVE A BROKER

EX: MQTT

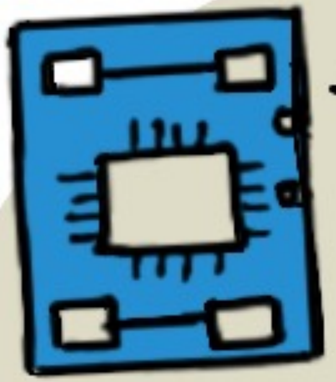
OTHER COMMUNICATION PROTOCOLS INCLUDE:

ADVANCED MESSAGE QUEUING PROTOCOL — AMQP

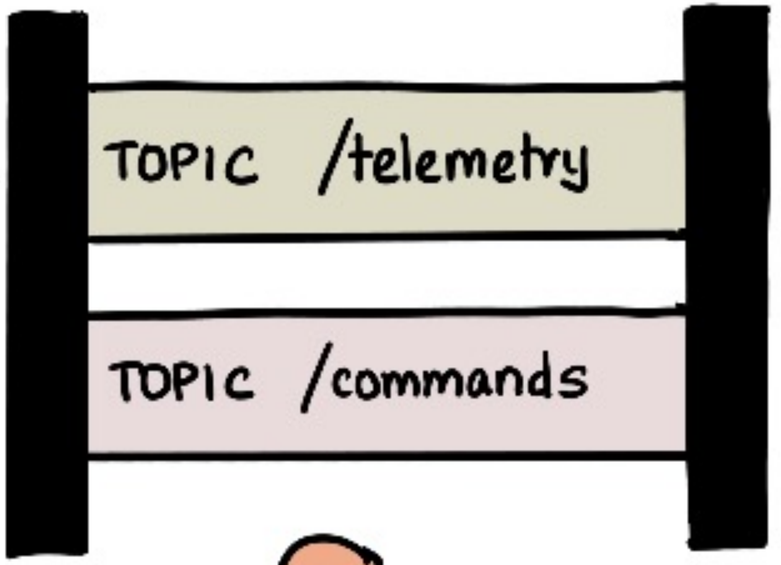
HYPERTEXT TRANSFER PROTOCOL (SECURE) — HTTP, HTTPS

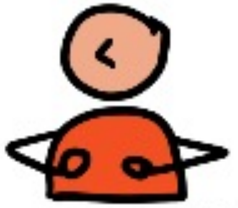
MESSAGE QUEUEING TELEMETRY TRANSPORT | MQTT

SINGLE BROKER
MULTIPLE CLIENTS
NAMED TOPICS



IOT DEVICE
PUBLISHES SENSOR
DATA TO /telemetry





SINGLE BROKER

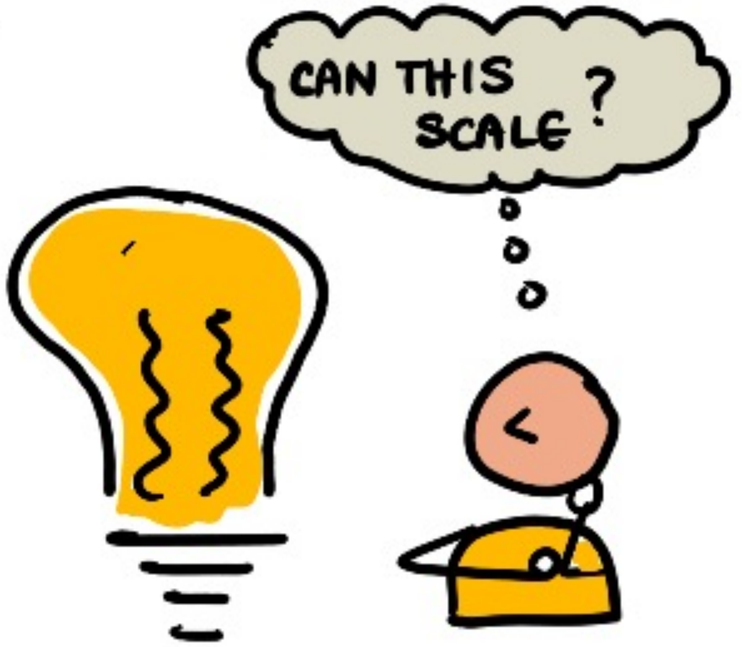
CLOUD SUBSCRIBES
TO TOPIC /telemetry

IS NOTIFIED OF NEW
TELEMETRY EVENTS

MESSAGES
ROUTED BY
TOPICS

NOT BY TARGET
ENDPOINTS

COMMUNICATES
OVER TCP/IP

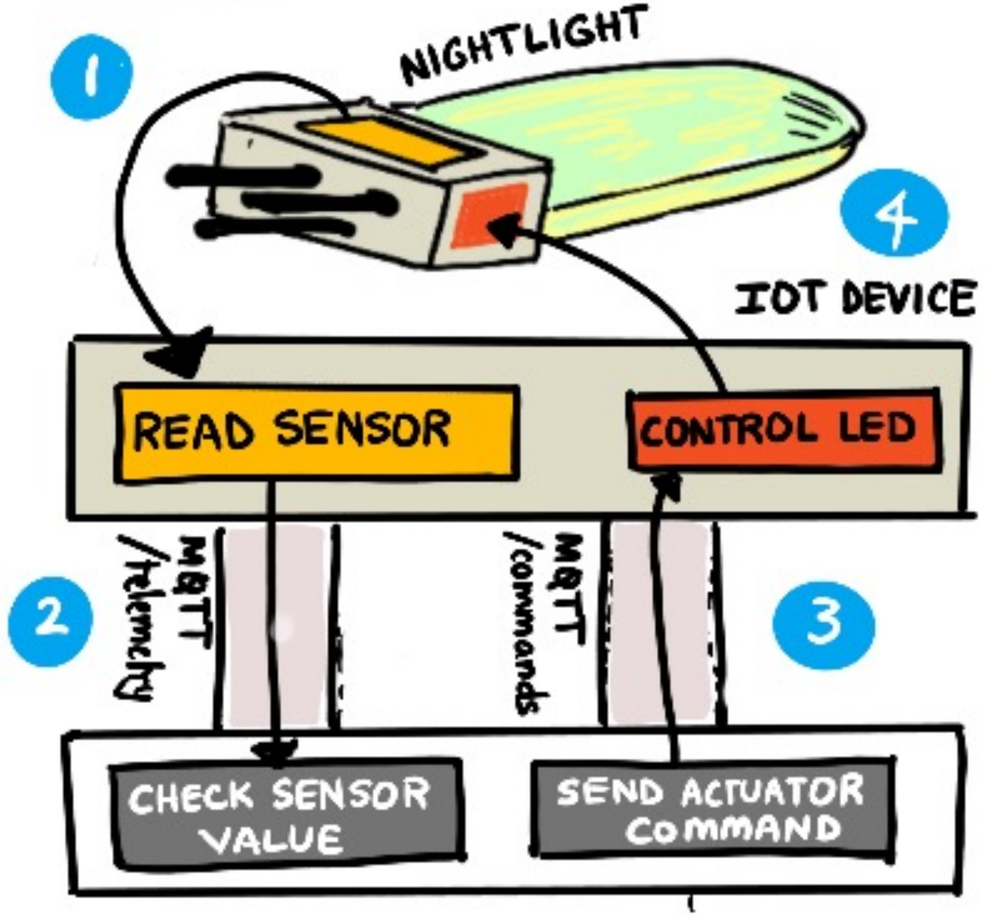


DO YOUR RESEARCH

LET'S ASSUME YOU HAVE A IOT LOT OF DEVICES

HOW DO YOU MAKE SURE YOUR MQTT BROKER CAN HANDLE ALL MESSAGES ?

CONNECT YOUR DEVICE TO AN MQTT BROKER



→ CONNECT YOUR IOT NIGHTLIGHT TO THE INTERNET, ENABLE IT TO BE REMOTE CONTROLLED

ECLIPSE MOSQUITTO = OPEN SOURCE PUBLIC MQTT BROKER YOU CAN USE FOR QUICK TESTS

CAUTION : NOT SECURE

DO RESEARCH

WHAT SCENARIO REQUIRES SENSOR DATA EVALUATION FROM MULTIPLE SENSORS BEFORE ACTUATOR COMMAND IS SENT?

A DEEPER DIVE INTO MQTT

- MQTT TOPICS CAN HAVE A HIERARCHY
- MESSAGE "QOS" DETERMINES DELIVERY GUARANTEES

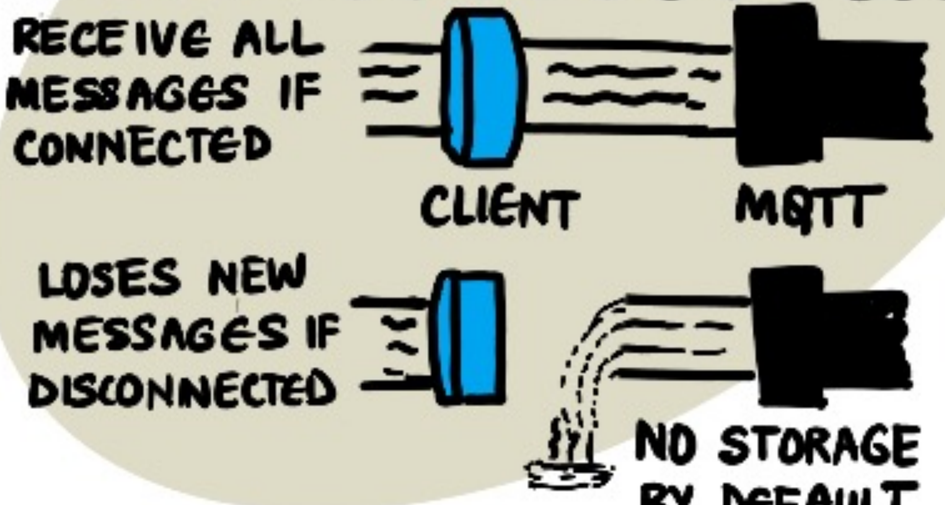
1. /telemetry /*
2. /telemetry humidity
3. /telemetry temperature

(2) and (3) are subtopics
Publish/subscribe to get only messages from that subset.

(1) is a wildcard
Subscribe to get messages from ALL subtopics!

A DEEPER DIVE INTO MQTT

MQTT DOES NOT ACTUALLY "QUEUE" UP MESSAGES FOR SUBSCRIBED CLIENTS



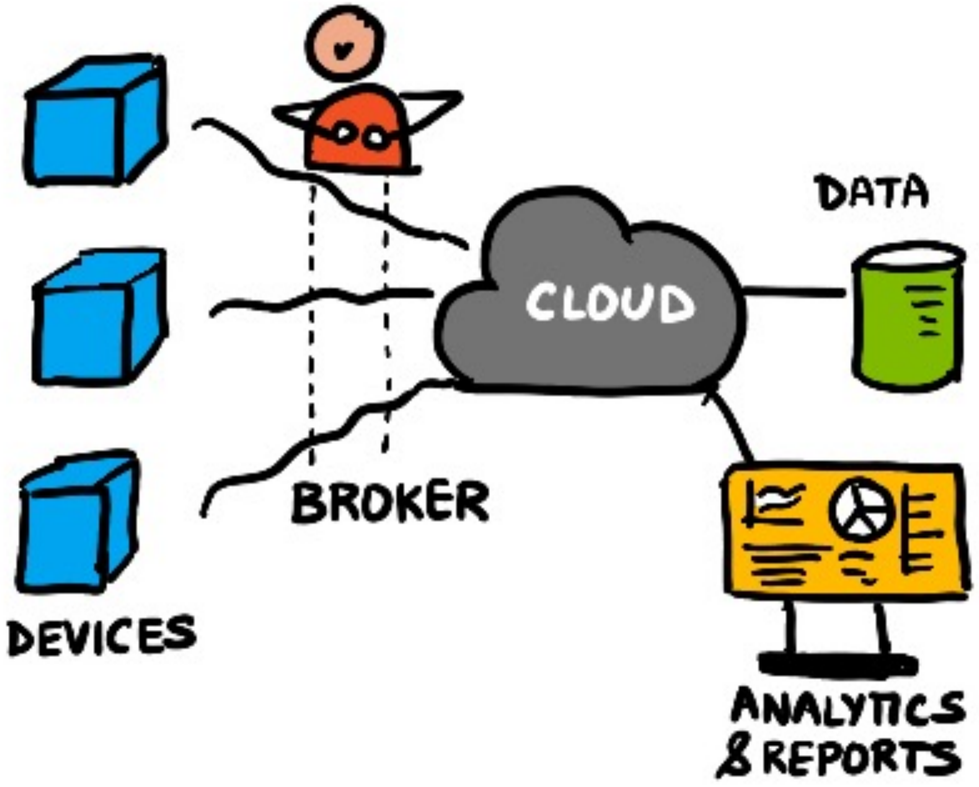
KEEP ALIVE FUNCTION

CHECKS IF THE CONNECTION IS ALIVE DURING LONG GAPS IN MESSAGES

RETAINED FLAG

SET IT TO ENSURE LAST MESSAGE ON TOPIC IS SAVED FOR LATE SUBSCRIBERS

TELEMETRY

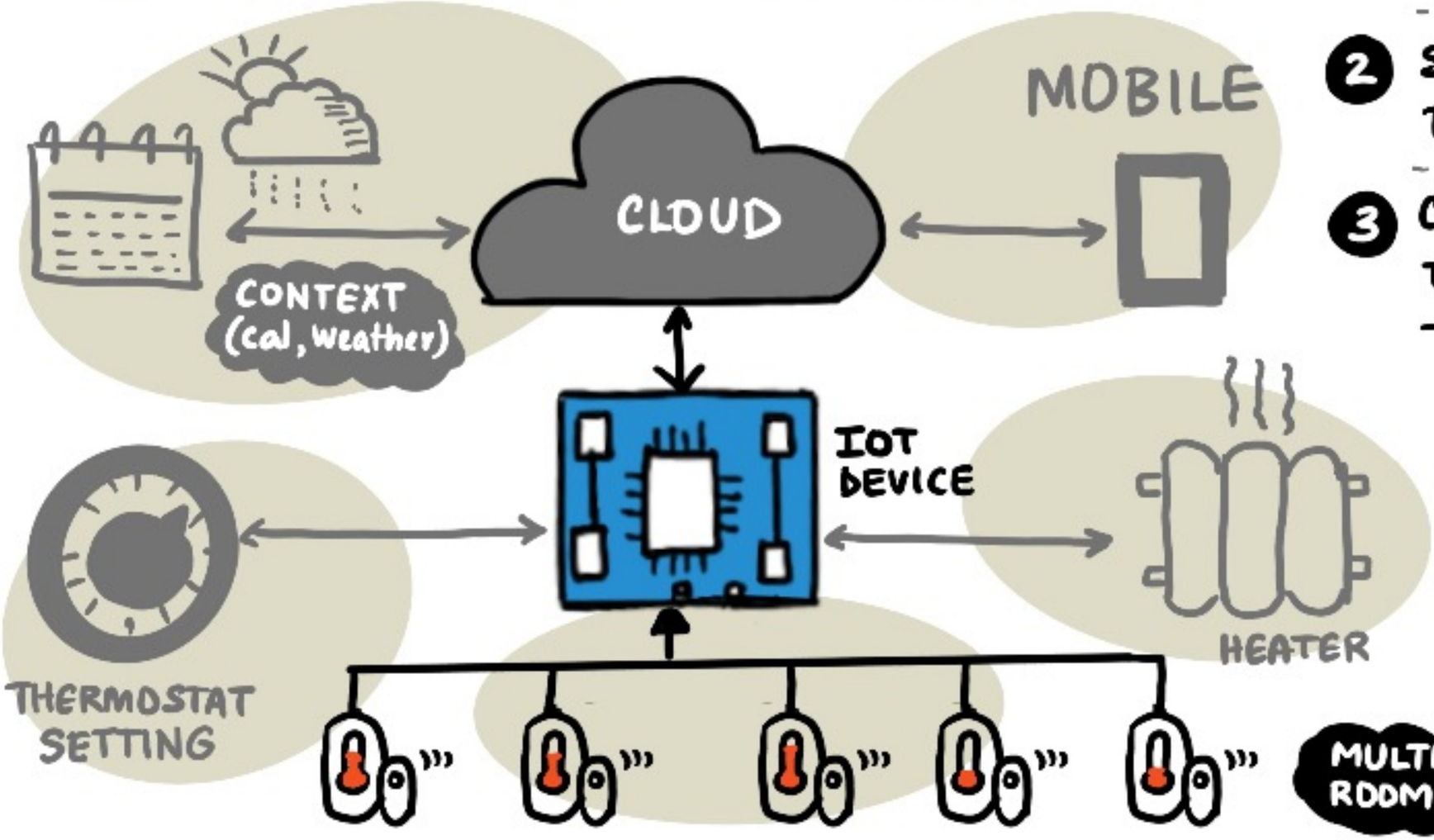


ORIGINS

"TELEMETRY" = GREEK WORD MEANING "TO MEASURE REMOTELY"

IT IS THE ACT OF GATHERING DATA FROM SENSORS AND SENDING IT TO THE CLOUD FOR PROCESSING

TELEMETRY EXAMPLE



1. INTERNET CONNECTED THERMOSTAT DEVICE
2. SENSOR TELEMETRY PUBLISHED TO CLOUD
3. CLOUD ANALYZES ROOM TELEMETRY, CONTEXT & THERMOSTAT SETTING TO REQUEST ACTION (heater ON, OFF)
4. ACTION PUBLISHED TO IOT DEVICE FOR ACTUATOR

MULTIPLE ROOM SENSORS connected over BT/WiFi

TELEMETRY EXAMPLE

THE INTERNET CONNECTED THERMOSTAT IOT DEVICE HAS

- TEMPERATURE SENSORS
  IN THERMOSTAT,
  IN REMOTE ROOM A,
  IN REMOTE ROOM B
- ACTUATOR TO HEATER
  TURN IT ON,
  TURN IT OFF,
  INCREASE/DECREASE SETTING

TELEMETRY DATA EXAMPLE

- **thermostat_temperature**
  value = 18°C
  measured by thermostat itself
- **living room_temperature**
  value = 19°C
  measured by room A temp sensor
- **bedroom_temperature**
  value = 21°C
  measured by room B temp sensor

SEND TELEMETRY FROM IOT DEVICE

IOT
NIGHTLIGHT

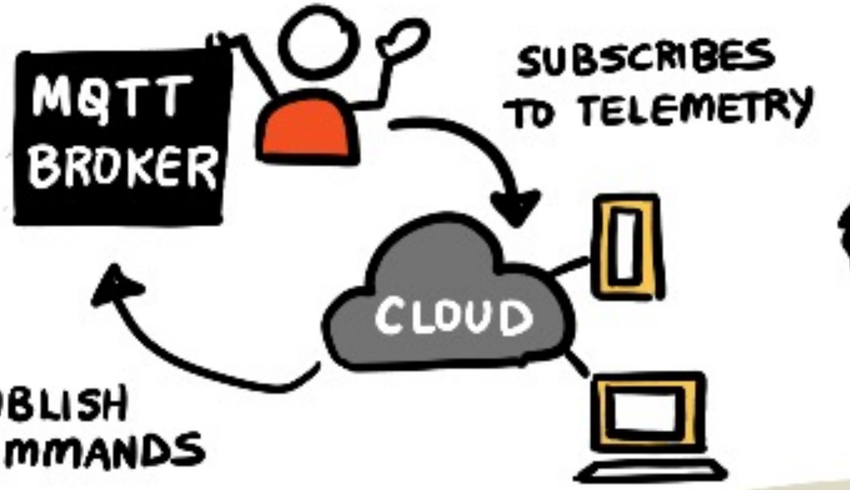
NIGHTLIGHT
PROJECT

NEXT TASK:

SEND LIGHT-LEVEL
TELEMETRY TO MQTT

- SEND LIGHT LEVEL TELEMETRY TO /telemetry TOPIC ON MQTT
- DATA SENT ENCODED AS JSON (KEY/VALUE PAIRS)
- FOLLOW THE APPROPRIATE GUIDE FOR NEXT STEPS (ARDUINO OR SINGLE BOARD COMPUTER OPTIONS)

RECEIVE TELEMETRY FROM MQTT BROKER



"SERVER" CODE THAT SUBSCRIBES TO TELEMETRY, ANALYZES CONTEXT, SENDS DECISION (COMMANDS) BACK TO IOT DEVICE

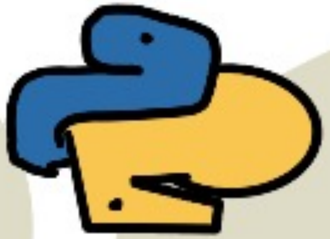
TELEMETRY HAS LIMITED VALUE WITHOUT SOME WAY TO PROCESS IT FOR DECISIONS

SERVER TYPICALLS RUNS IN CLOUD - FOR NOW, RUN IT IN LOCAL SERVER

INSTALL & CONFIGURE PYTHON ENV

READY TO CODE?

Here is our recommended development setup!



1 INSTALL PYTHON & PIP

CONFIGURE USING PIP
INSTALL PAHO-MQTT PKG

2 INSTALL VS CODE

POPULAR EDITOR. INSTALL PYLANCE EXTENSION

REFER TO GUIDE FOR DETAILS...

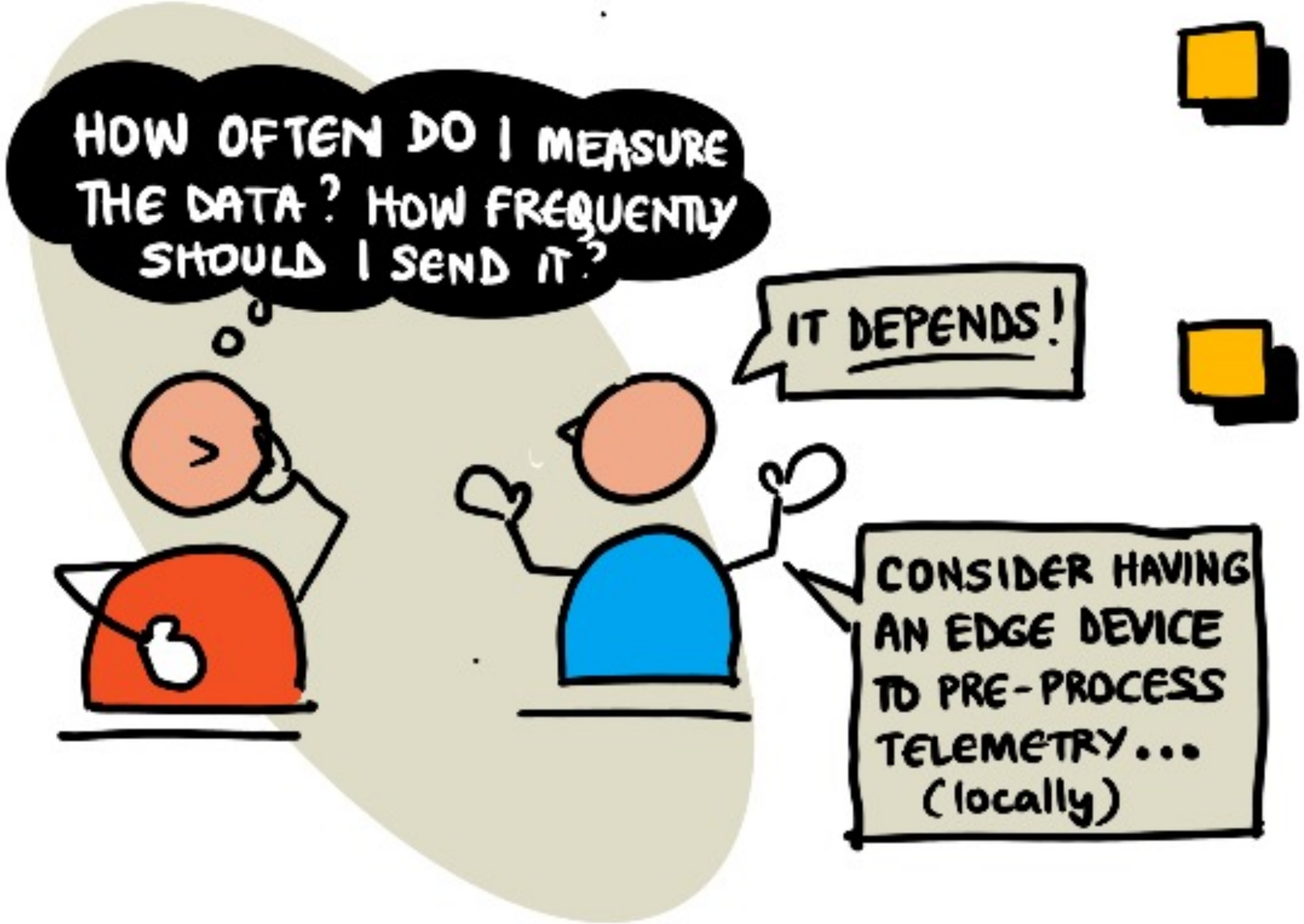
WRITE THE SERVER CODE

8 STEPS TO SETUP YOUR SERVER!

1. CREATE APP.PY
2. LAUNCH VS CODE
3. CHECK: PYTHON ENV RUNS
4. KILL PRE-EXISTING VS CODE TERMINAL
5. LAUNCH VS CODE TERMINAL (active env)
6. EDIT APP.PY CODE
7. RUN APP.PY
8. VALIDATE SETUP (messages in terminal)

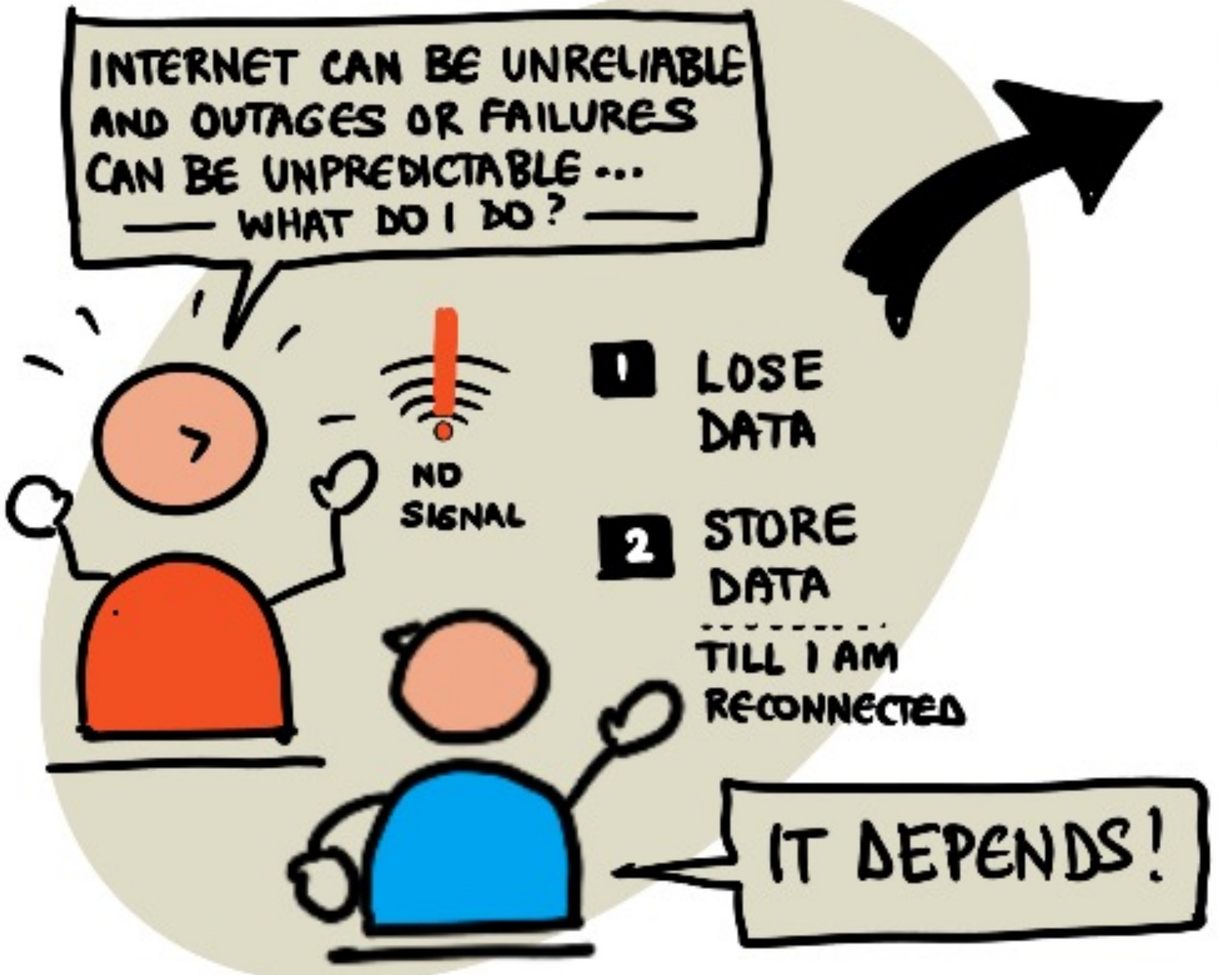
DONE!

HOW OFTEN SHOULD TELEMETRY BE SENT?



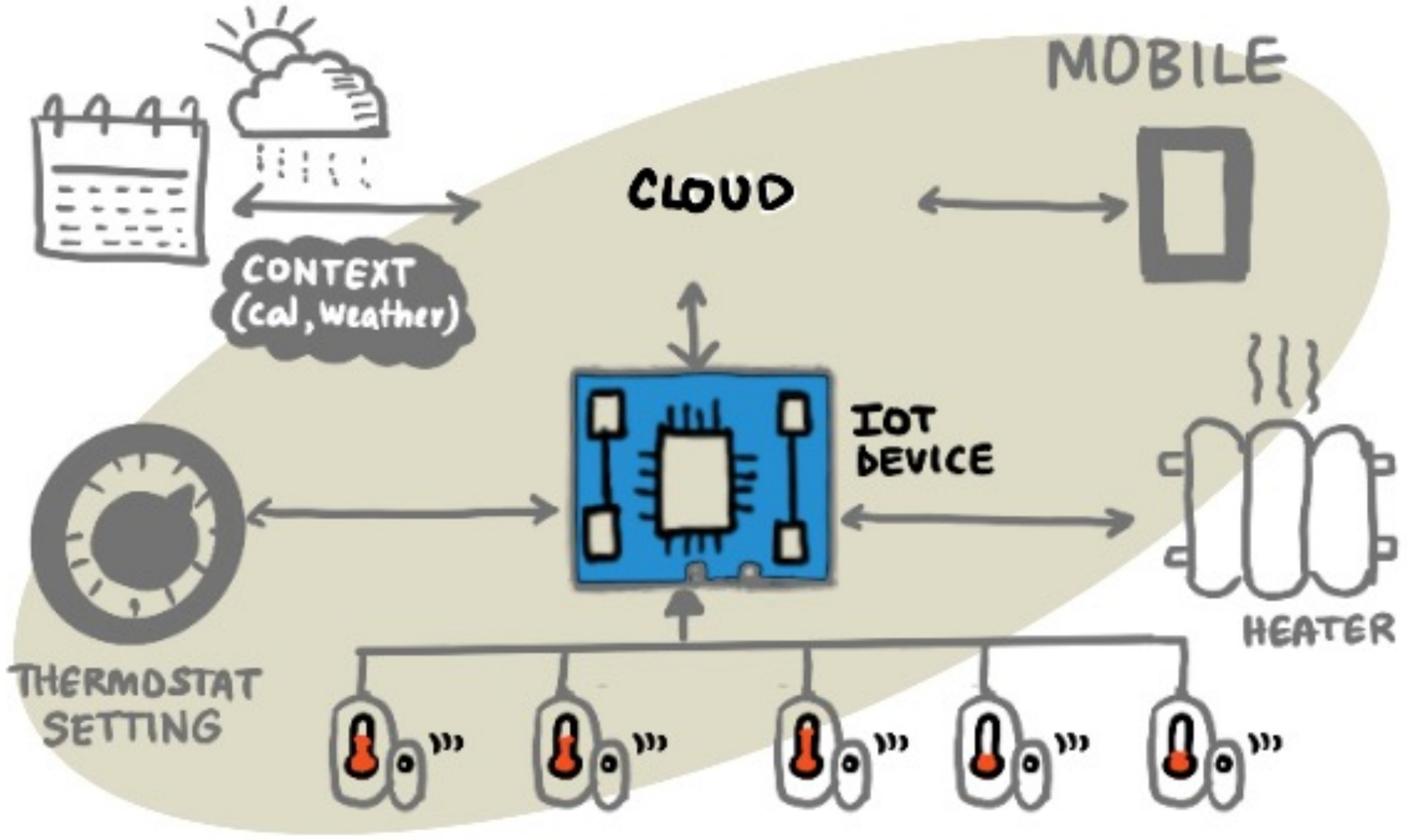
- HIGHER FREQUENCY HAS BETTER RESPONSES TO CHANGE BUT USES MORE RESOURCES!
- THERMOSTAT EXAMPLE
  - * EVERY SECOND = TOO MUCH
  - * ONCE A DAY = TOO LITTLE
  - * EVERY FEW MINUTES = ENOUGH!
- BETTER SAFE THAN SORRY
  - More frequent if mission critical

LOSS OF CONNECTIVITY



- **THERMOSTAT:**
  TEMPERATURE HAS LESS VALUE WITH TIME — DON'T STORE IT!
- **MACHINERY:**
  OPERATIONAL DATA CAN ALWAYS BE USED FOR PROFILING AND PREDICTIVE MAINTENANCE — STORE IT!
- MQTT WON'T DO THIS FOR YOU. WRITE CLIENT/SERVER CODE...

SEND COMMANDS TO MQTT



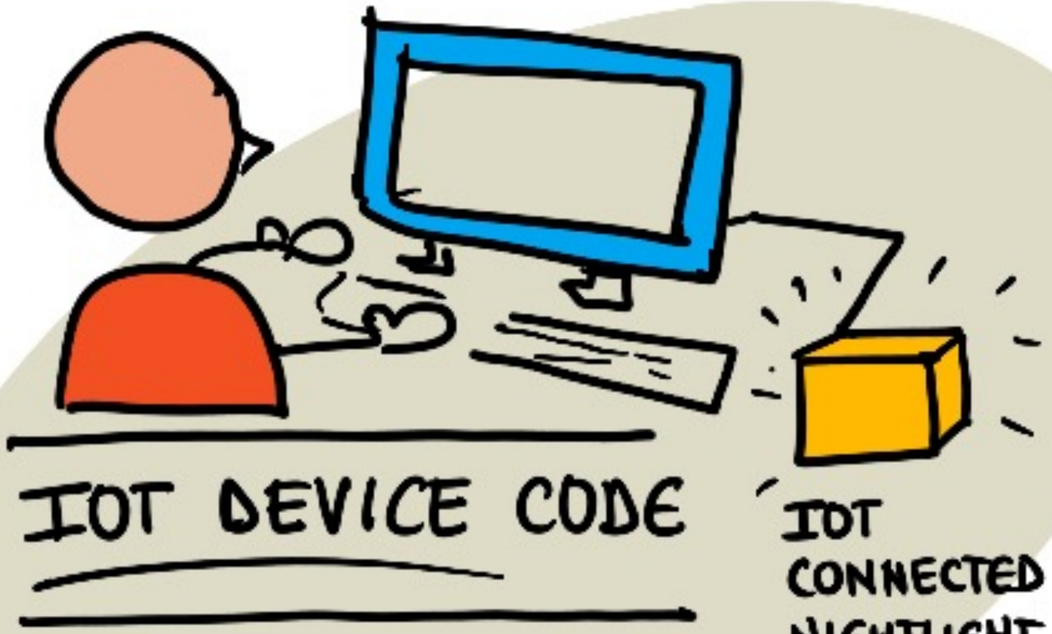
COMMANDS ARE:

MESSAGES FROM CLOUD TO DEVICE WITH A REQUEST FOR ACTION (e.g. ACTUATOR REQUEST)

EDIT SERVER CODE TO SEND COMMAND TO SPECIFIC TOPIC ...

EXPLORE HOW COMMANDS CAN BE DIRECTED AT ONE VS. MANY USING MULTIPLE TOPICS

HANDLE COMMANDS ON THE IOT DEVICE

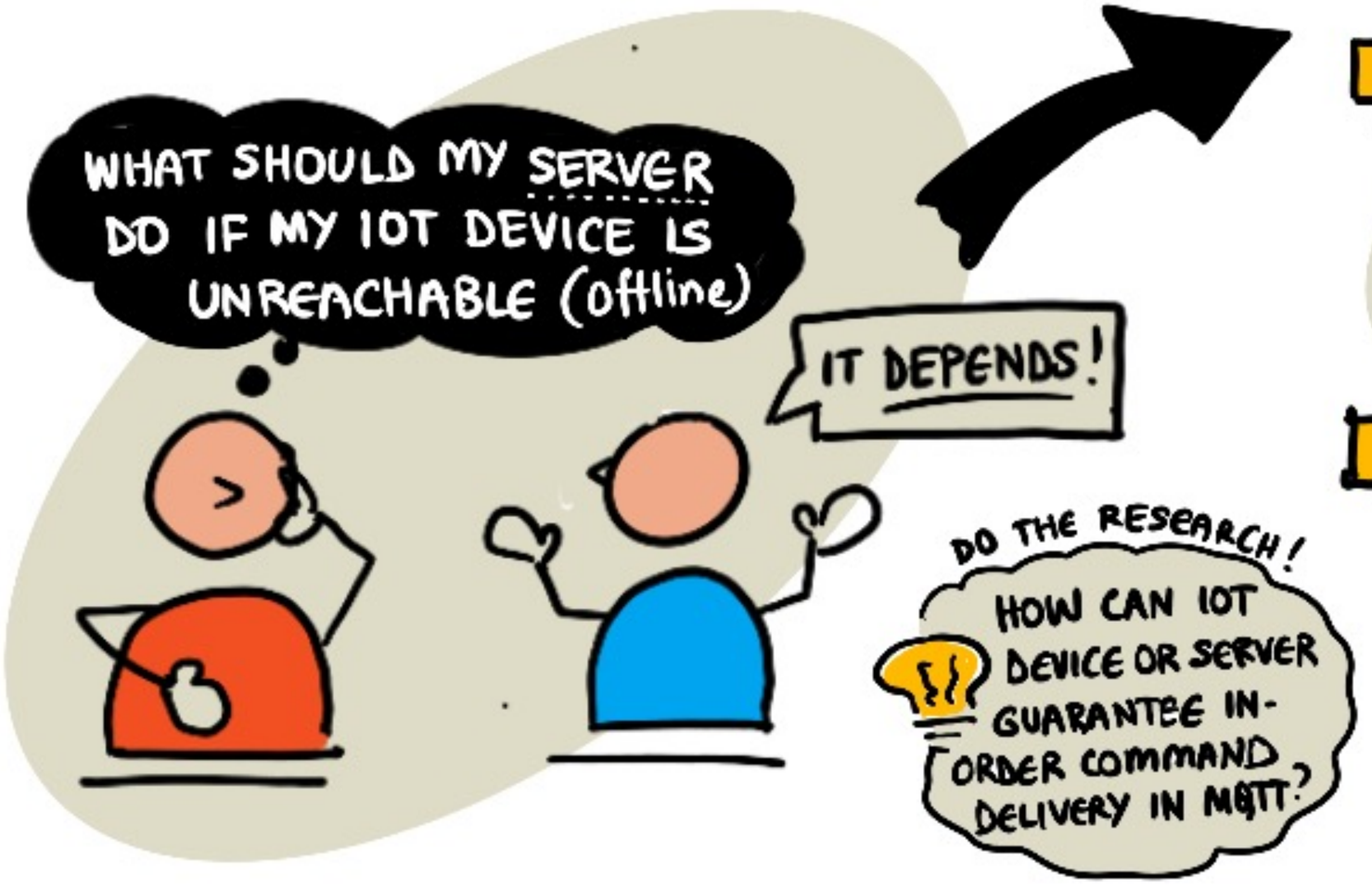


COMMANDS ARE PUBLISHED BY SERVER VIA MQTT TOPIC

IOT DEVICES MUST

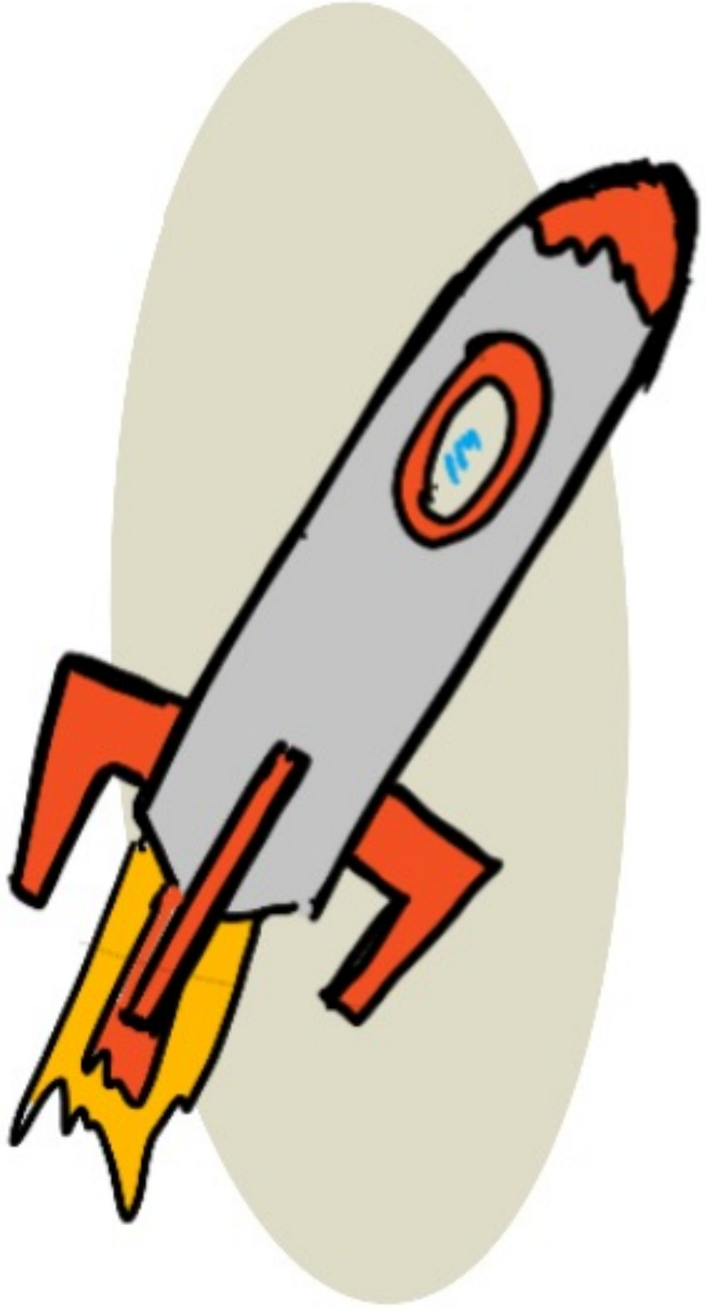
- SUBSCRIBE TO RELEVANT TOPIC
- LISTEN FOR MESSAGES ON TOPIC
- HANDLE MESSAGES – TAKING THE APPRORIATE ACTION TO CONTROL LED

LOSS OF CONNECTIVITY



CONTEXT IS KEY

- DOES THE LATEST COMMAND OVERRIDE PREVIOUS ONES?
  - IGNORE OLD COMMANDS
- DO COMMANDS NEED TO BE PROCESSED IN ORDER? (DO X, THEN DO Y)
  - SEND/RESEND COMMANDS IN THE DESIRED ORDER



CHALLENGE

- LOOK AT YOUR PREVIOUS LISTS OF IOT DEVICES
- FOR EACH DEVICE - WHAT MESSAGES MIGHT THEY BE SENDING? RECEIVING?
- ARE THEY SECURE? WHAT TELEMETRY DO THEY CAPTURE? SEND?

NEXT: TO THE FARM TO PREDICT PLANT GROWTH


CONGRATULATIONS
CREATED BY
@ SKETCHTHEDOCS