



# Building Internet of Things applications with .

## COMPOSE and JavaScript

Charalampos Doukas  
@buildingiot

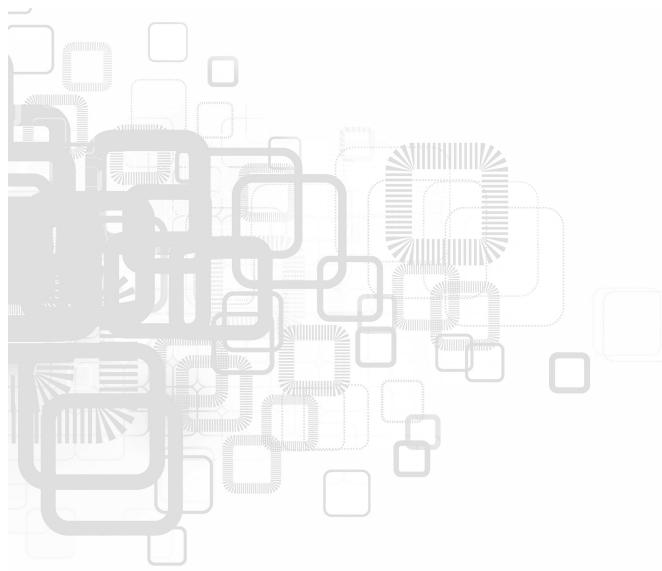


**SENZATIONS**  
9th Summer School on IoT Applications  
31.08.2014. - 06.09.2014.  
Biograd na Moru, Croatia



# Building Internet of Things applications with COMPOSE and JavaScript

## PART A – Some Basics

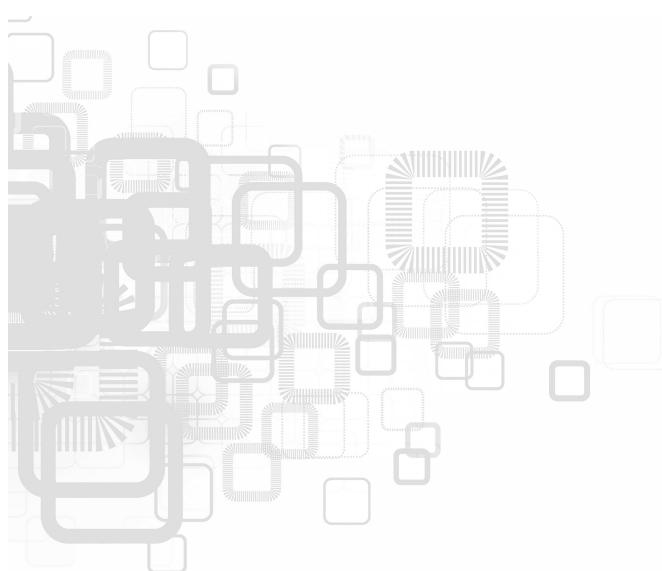


# IoT: The main components

Devices – ‘Smart Objects’ – ‘Things’

Networks

Online Services

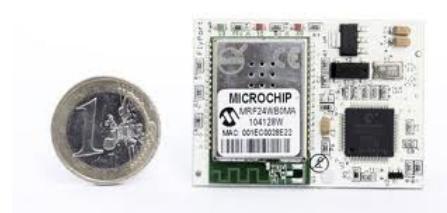
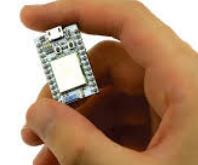


# Devices

- Prototyping platforms (some IoT oriented)
- Arduino (....)
  - <http://postscapes.com/arduino-wifi>
- openPicus Flyport (WiFi, Ethernet, GPRS, Enocean)
- Libelium WaspMote
- mBed by ARM
- Pinocc.io (mesh network support, RF+WiFi)
- Spark core (WiFi)
- Electric Imp
- Tessel (WiFi, Node.js)



powered by  
spark



# Devices

- SODAQ (WiFi, Xbee, ..)
- XinoRF
- PanStamps
- Intel boards
- TI CC3200 Evaluation board
- Domain specific
  - BITalino (Wearables)
  - Thingsquare
  - Flutter

<http://postscapes.com/internet-of-things-diy>

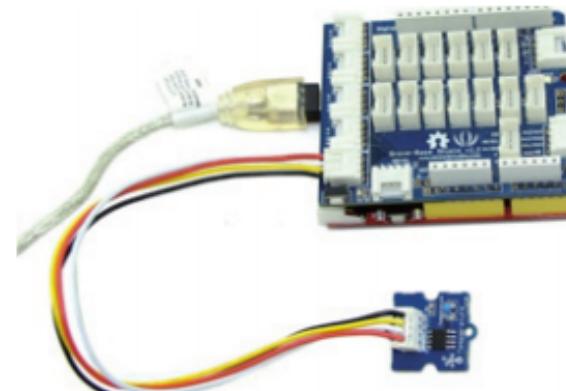
<http://postscapes.com/internet-of-things-hardware>

# Gateways

- Why do I need a gateway?
- RaspberryPi
  - <http://postscapes.com/raspberry-pi-wireless-options>
- BeagleBone
- UDOO
- Intel Galileo
- Commercial:
  - SmartThings

# Sensors & Actuators

- Gas & Air quality
- Barometric pressure
- Temperature & Humidity
- Light & Sound
- Motion
- Flex & Force
- Position
- Magnetic fields
- Electricity
- Biometrics



# Sensors & Actuators

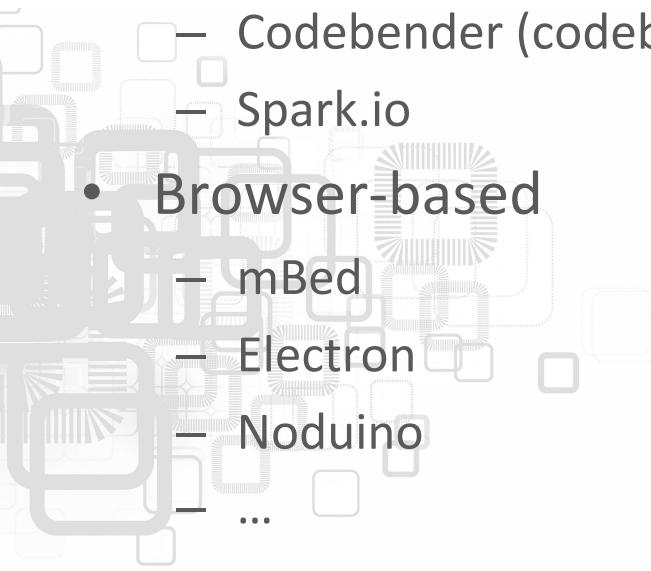
- Proximity & Presence
- Weight
- Liquids & Liquid flow
- Radiation
- ...



# Sensors & Actuators

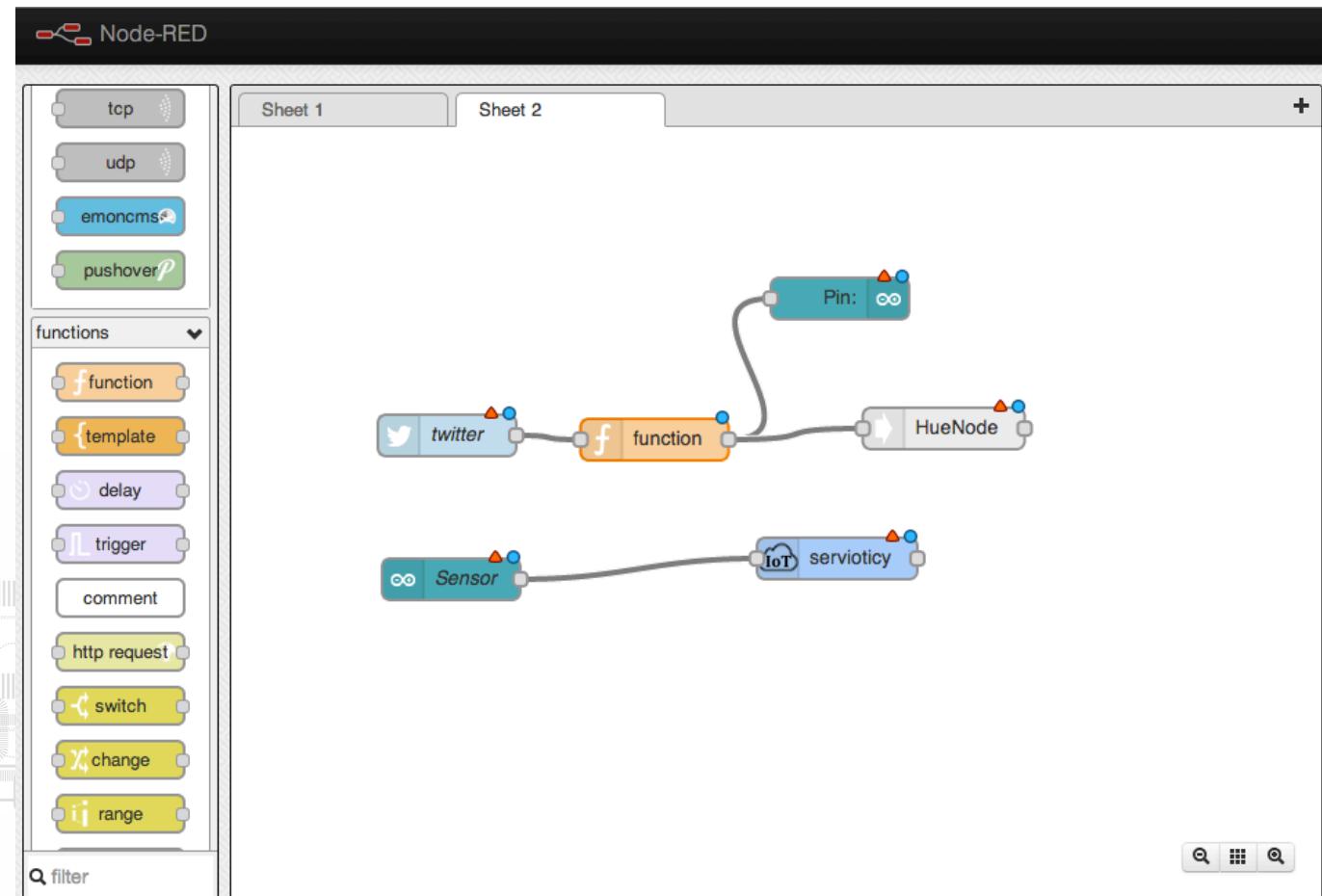
- Move things (motors)
- Activate (switches)
- Interfaces
  - Sound
  - Light
  - Displays
- Remote interfaces
  - Social
  - Email, text, ...
  - Phone (twilio)

# Software

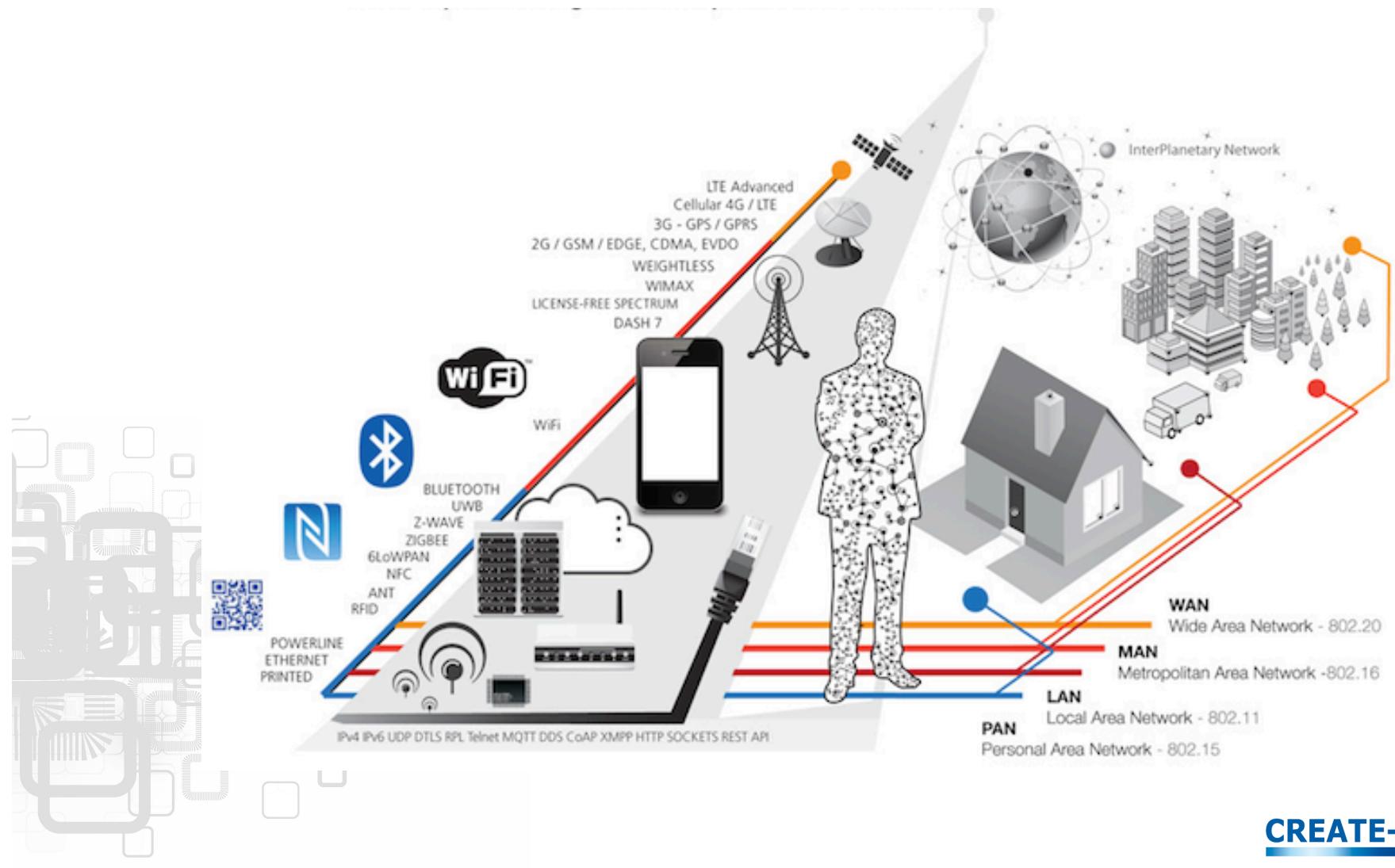
- Standard IDEs
  - Arduino IDE
  - Eclipse
- Code – less:
  - Scratch
- Cloud-based
  - Codebender ([codebender.cc](http://codebender.cc))
  - Spark.io
- Browser-based
  - mBed
  - Electron
  - Noduino
- ...

# Workflow - based

- Node-RED



# Networks



# Protocols

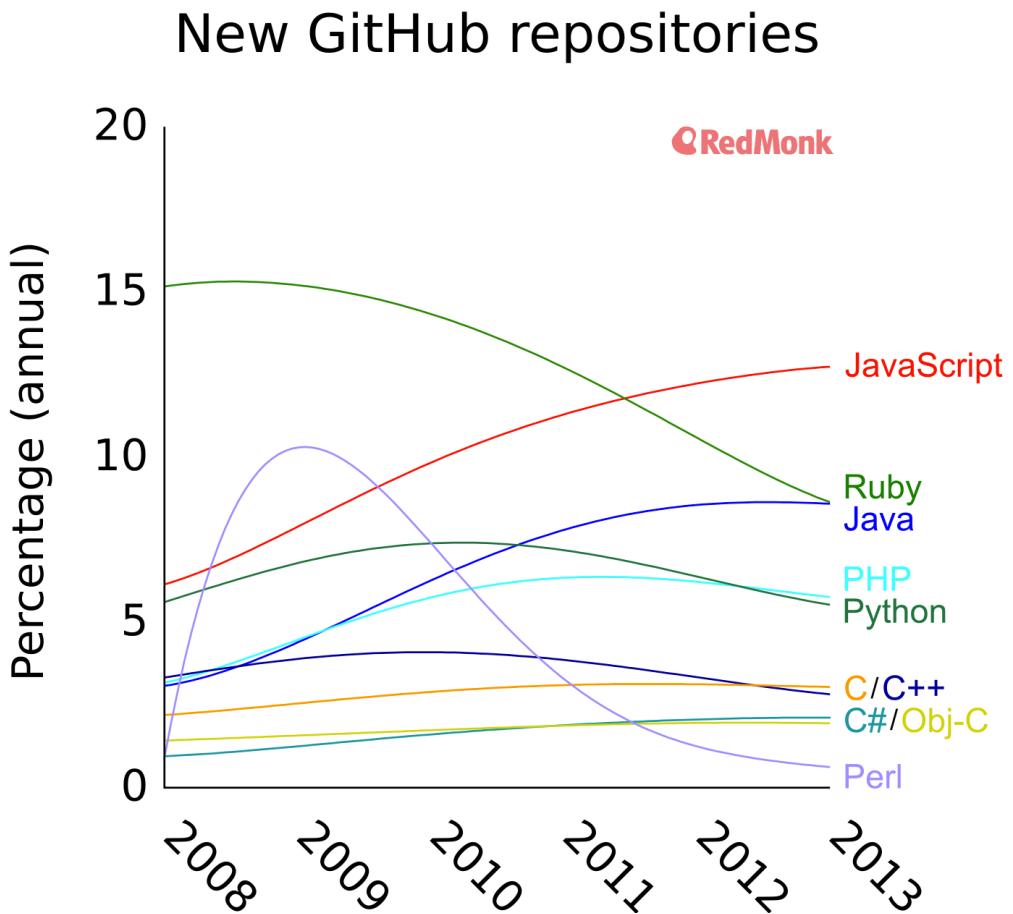
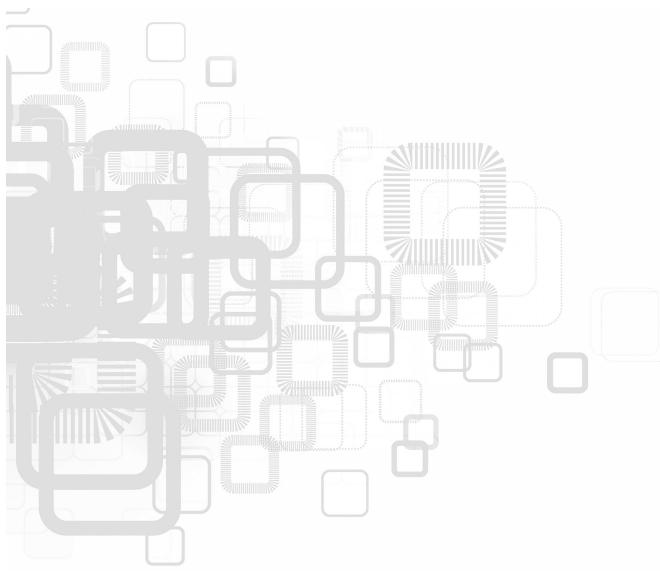
- Web based:
  - HTTP
    - REST
    - WebSockets
- M2M (lower resources + actuation)
  - MQTT
  - CoAP - LWM2M
  - XMPP
  - STOMP
  - ...

# Online Services

- Sensor Data
  - Pachube/Cosm/Xively
  - Paraimpu
  - TheThingsystem.io
- Device Management
- Full PaaS

# IoT & JavaScript

Why JavaScript?



# IoT & JavaScript

## Why JavaScript?

coap

Search

We've found 116 repository results

Sort: Best match ▾

 <a href="#">coapp/coapp.org</a>	JavaScript	★ 25	15
Website			
Last updated on Oct 2, 2013			
 <a href="#">openwsn-berkeley/coap</a>	Python	★ 10	4
A CoAP Python library			
Last updated 14 days ago			
 <a href="#">mcollina/node-coap</a>	JavaScript	★ 34	7
CoAP - Node.js style			
Last updated yesterday at 9:18 AM			
 <a href="#">dgiannakop/Arduino-CoAP</a>	C++	★ 21	8
Basic CoAP library for Arduino			
Last updated on Feb 8			

# IoT & JavaScript

## Why JavaScript?

We've found 766 repository results

Sort: Most stars ▾



[adamvr/MQTT.js](#)

An MQTT library for node.js

Last updated 2 days ago

JavaScript ★ 398 ⚡ 112



[tokudu/AndroidPushNotificationsDemo](#)

A example of an android app that receives push notifications using MQTT.

Last updated on Feb 17, 2011

Java ★ 353 ⚡ 228



[mcollina/mosca](#)

The multi-transport MQTT broker for node.js. It supports AMQP, Redis, MongoDB, ZeroMQ or just MQTT.

Last updated 9 days ago

JavaScript ★ 263 ⚡ 45



[knolleary/pubsubclient](#)

A client library for the Arduino Ethernet Shield that provides support for MQTT.

Last updated on Feb 8

C++ ★ 190 ⚡ 76

# IoT & JavaScript

## Why JavaScript?

We've found 5,104 repository results

Sort: Most stars ▾



[joewalnes/websocketd](#)

Go ★ 3,235 ⚡ 141

Turn any program that uses STDIN/STDOUT into a **WebSocket** server. Like inetd, but for **WebSockets**.

Last updated 20 days ago



[square/SocketRocket](#)

Python ★ 2,598 ⚡ 484

A conforming Objective-C **WebSocket** client library.

Last updated 22 days ago



[sockjs/sockjs-client](#)

JavaScript ★ 2,423 ⚡ 191

**WebSocket** emulation - Javascript client

Last updated 8 days ago



[Atmosphere/atmosphere](#)

Java ★ 1,765 ⚡ 411

Realtime Client Server Framework for the JVM, supporting **WebSockets** and Cross-Browser Fallbacks Support

Last updated 10 hours ago

# IoT & JavaScript

## Software

- Noduino
- A simple and flexible JavaScript and Node.js Framework for accessing basic Arduino controls from Web Applications using HTML5, Socket.IO and Node.js

### Listening for Events on Arduino

Catch a button push event and fade on your LED

Buttons and LEDs provide a simple interface to listen for events. Use `Button.on()` or `LED.on()` for triggering your code if a button is pushed or your LED is switching modes. Multiple events for a single event are called the order they have been assigned.

Use `Board.receive()` for catching all incoming data streams. You probably need this if you plan to create a **Real-Time Web Application** with noduino. Buzzword...

```
1. var Noduino = new NoduinoObj({debug: true, host: 'http://localhost:8090'}, Connector);
2. Noduino.connect(function(err, board) {
3.   if (err) { return console.log(err); }
4.   board.withButton({pin: 13}, function(err, Button) {
5.     if (err) { return console.log(err); }

6.
7.     Button.on('push', function() {
8.       console.log('Button pushed');
9.     });
10.
11.     Button.push();
12.   });
13. });
```

[Connect to Arduino](#)

# IoT & JavaScript

## Software

- Device.js
- DeviceJS is a JavaScript based development platform for reacting to sensors and controlling devices. It's built on top of Google's V8 JavaScript engine, Node.js and a real-time JSON database



DeviceJS lets you sense and control the physical world around you with JavaScript.

### Overview

DeviceJS is a JavaScript based development platform for reacting to sensors and controlling the devices. It's built on Google's V8 JavaScript engine, Node.js and a real-time JSON database. A DeviceJS application can run on a single local device or in different locations. You can use it to build sensor networks which need to events, gather physical data from other sensors or automate systems. DeviceJS provides agents from a developer stand point. It will be protocols such as I2C/I2S, SPI, RS485, RS422, WiFi, Zigbee, BLE, communication and all types of hardware protocols.

### Why JavaScript?

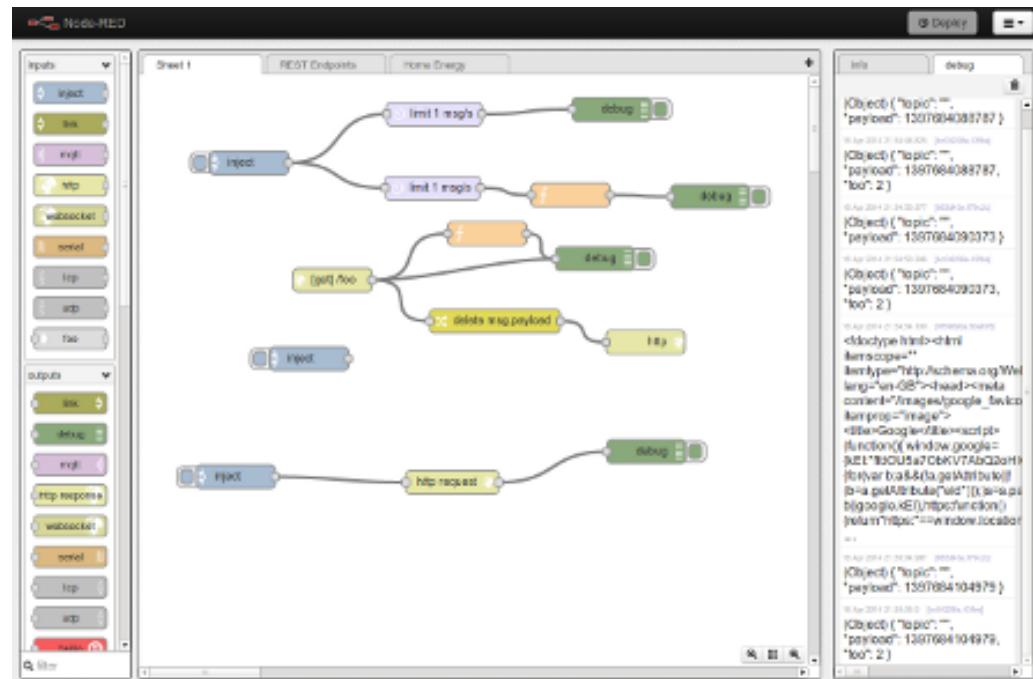
JavaScript is by far the most widely used language in the open source community. It's easy to write simple or powerful enough to do complex work. Most importantly it's very well known by many programmers.

**One script, execute it everywhere**

# IoT & JavaScript

## Software

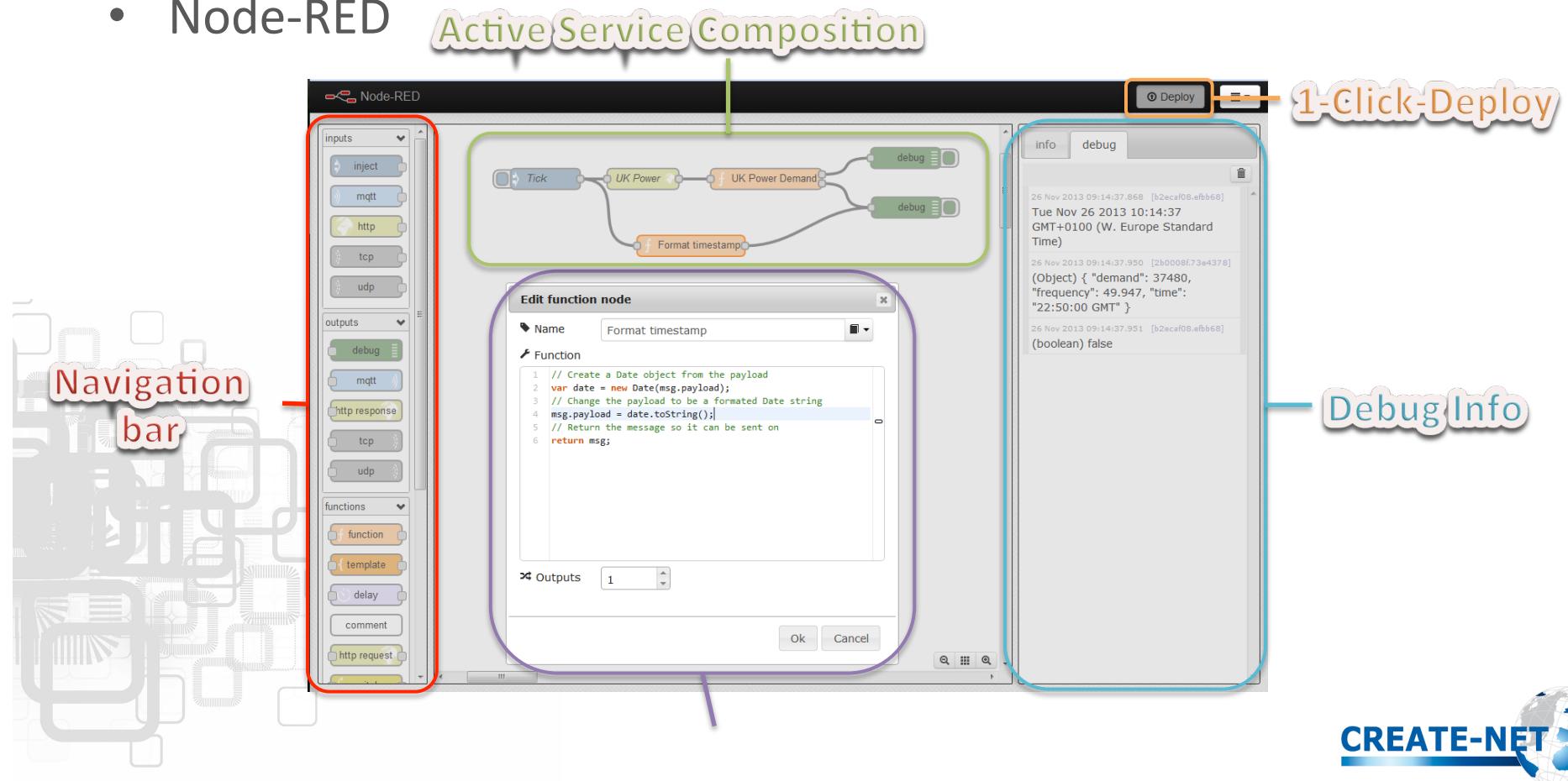
- Node-RED
  - Node-RED is a tool for wiring together hardware devices, APIs and online services
  - Web-based
  - O/S



# IoT & JavaScript

## Software

- Node-RED



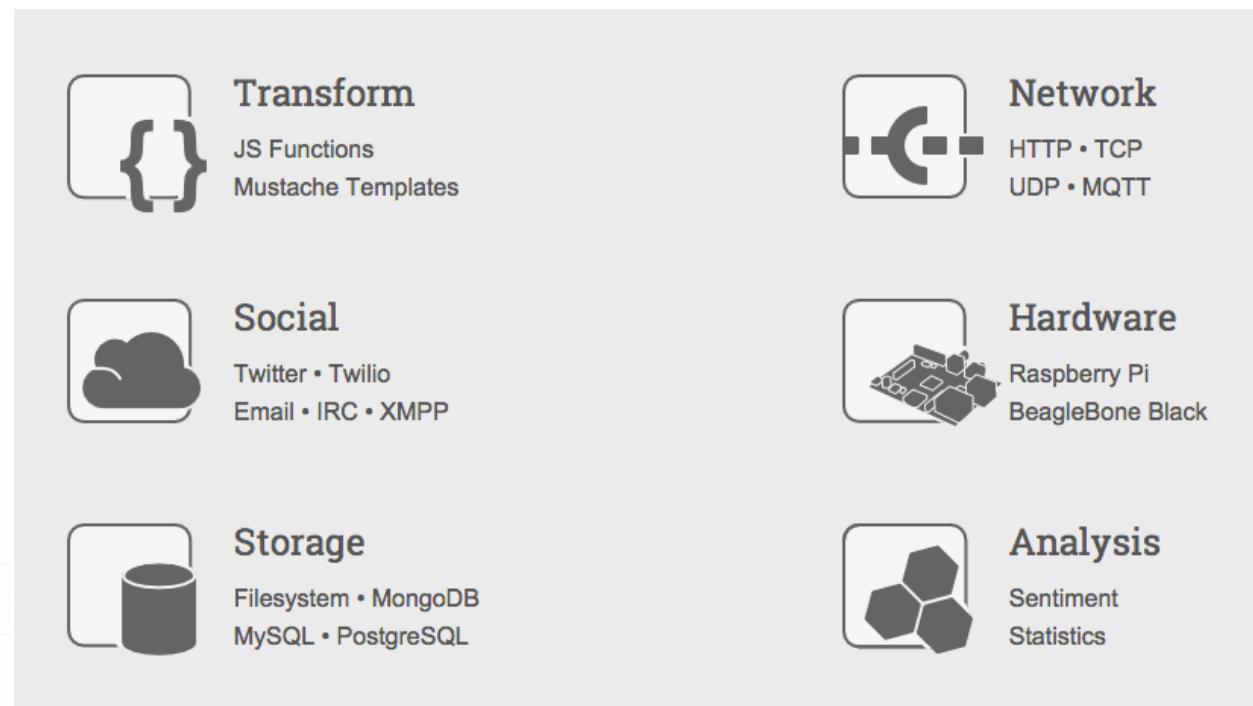
# IoT & JavaScript

## Software

- Node-RED
- Nodes available:

Over 40

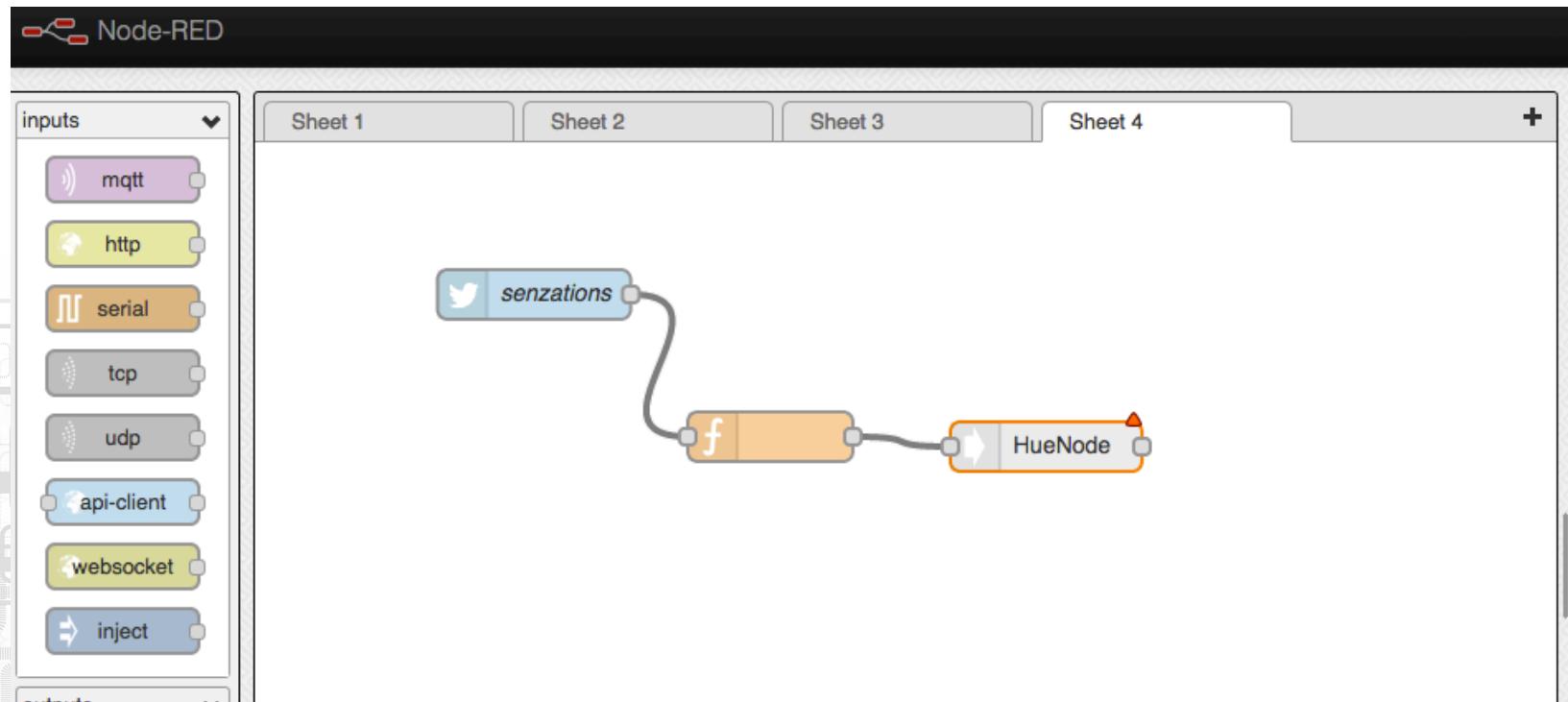
user contributed  
nodes



# IoT & JavaScript

## Software

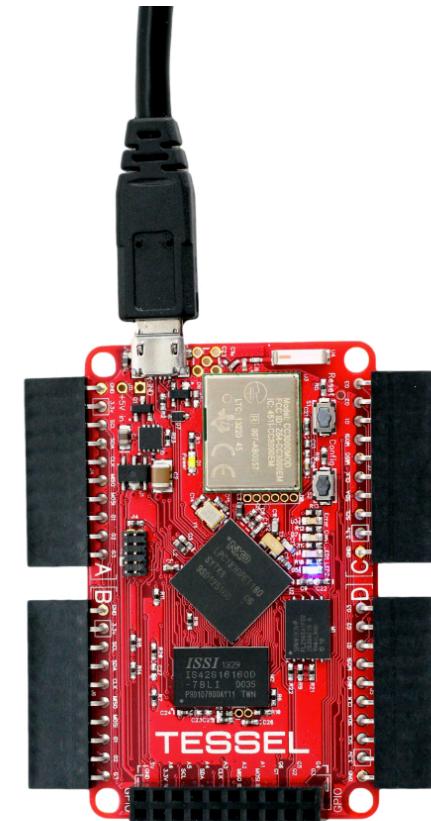
- Node-RED



# IoT & JavaScript

## Hardware

- Tessel
  - 180mhz ARM Cortex-M3 LPC1830
  - 32mb SDRAM
  - 32mb Flash
  - TI CC3000 Wifi Radio
  - 16-pin GPIO bank for prototyping
  - Open source code, open source hardware
  - NPM module support

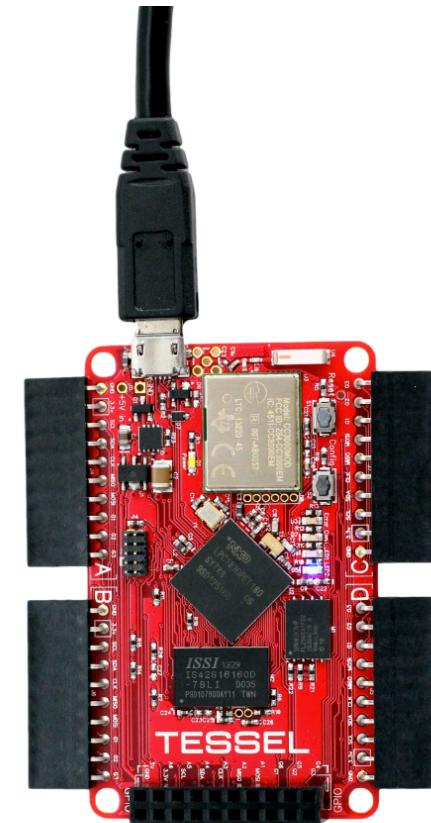


# IoT & JavaScript

## Hardware

- Tessel

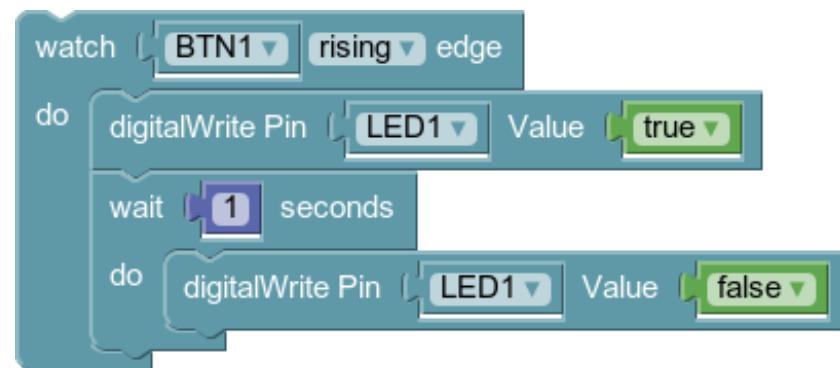
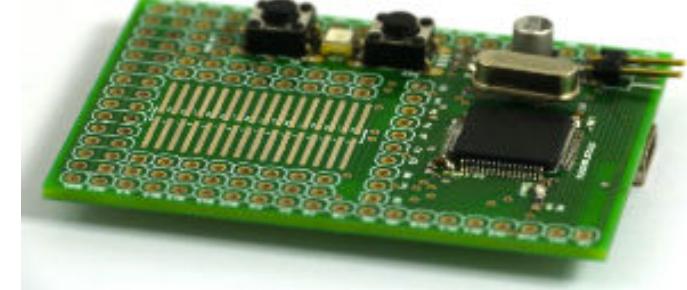
```
*****
This Bluetooth Low Energy module demo scans
for nearby BLE peripherals. Much more fun if
you have some BLE peripherals around.
*****  
  
var tessel = require('tessel');
var blelib = require('ble-ble113a');  
  
var ble = blelib.use(tessel.port['A']);  
  
ble.on('ready', function(err) {
  console.log('Scanning...');
  ble.startScanning();
});  
  
ble.on('discover', function(peripheral) {
  console.log("Discovered peripheral!", peripheral.toString());
});
```



# IoT & JavaScript

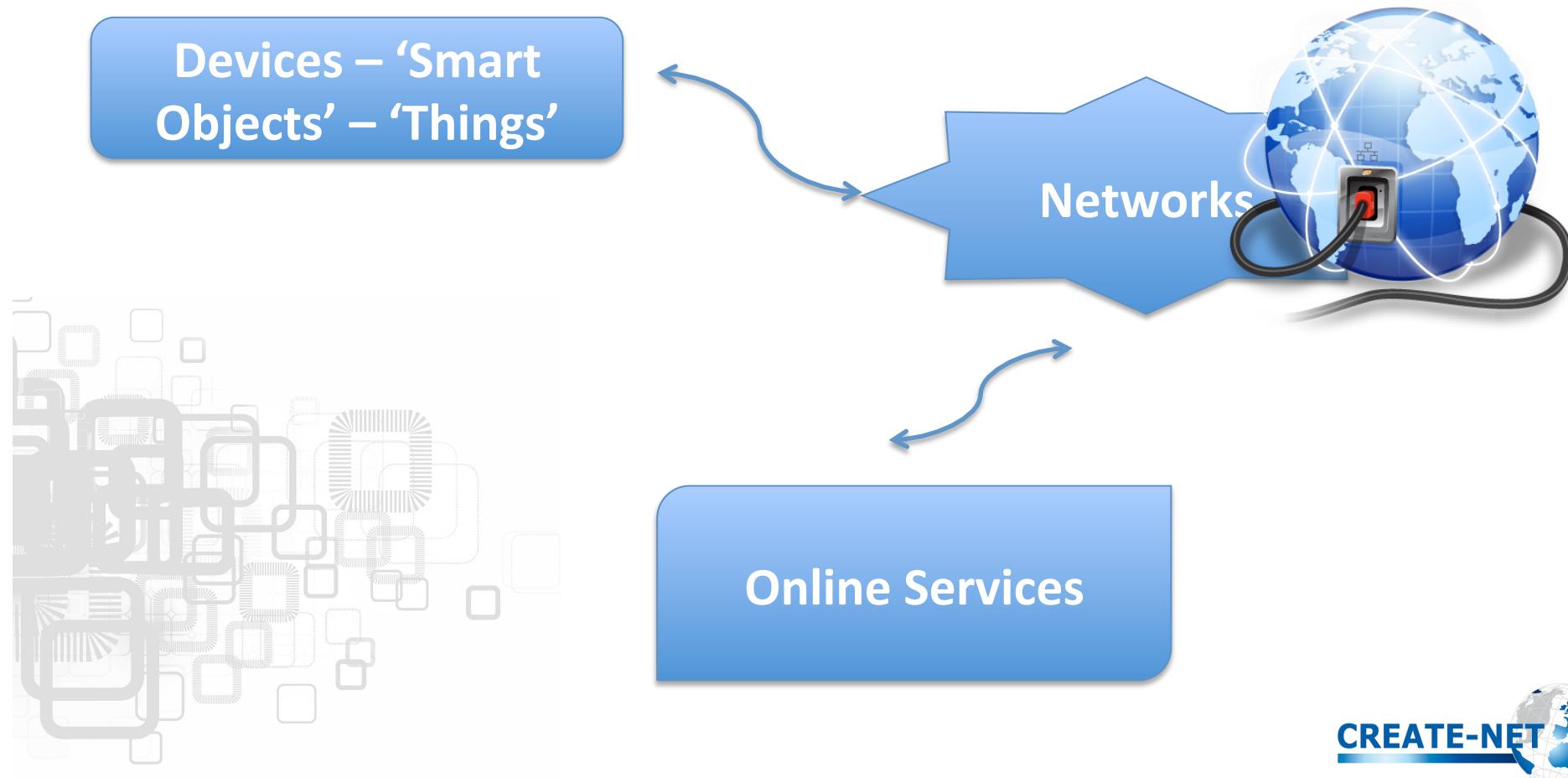
## Hardware

- Espruino
  - TM32 32-bit 72MHz ARM Cortex M3 CPU
  - 256KB of Flash memory, 48KB of RAM
  - 44 GPIO Pins
  - WiFi Support with the TI CC3000
  - Open source code, open source hardware
  - Web-based IDE
  - Code-less IDE



# Building IoT Applications

What else does it take?



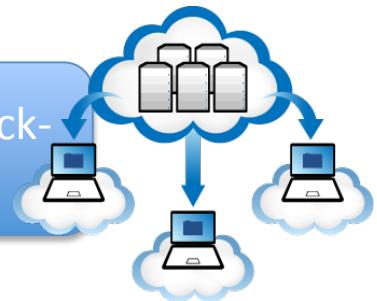
# Building IoT Applications

What else does it take?

Many APIs to connect and integrate



Deploy & Manage Back-End



Management & Data Maintenance



# Building IoT Applications

What else does it take?



Software

Hardware

Data?

Libraries

APIs

Users?

# Building IoT Applications

What else does it take?



Resources

scalable  
Hosting

Management

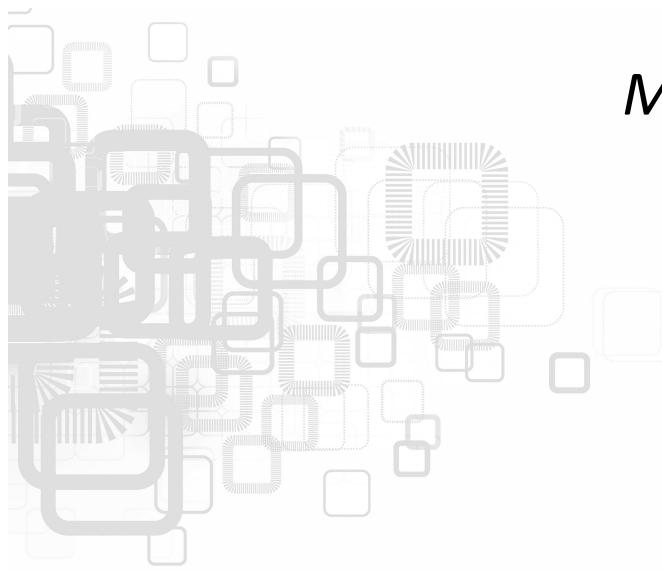
Outsourcing



JavaScript

comprise

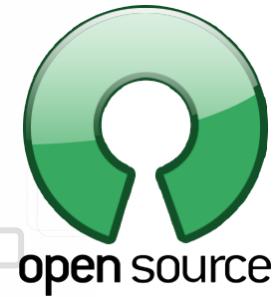
*More than sensor data storing...*



# What is COMPOSE?



**Collaborative Open Market to Place  
Objects at your Service**



Scalable Paas

IoT Marketplace

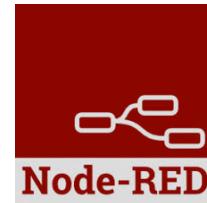
Development Tools

Sensor  
Communication  
Technologies





## IoT Platform As A Service



Node-RED

MQTT  
WebSockets  
XMPP  
...



Apache  
STORM

compose Service Discovery

Security



CLOUD  
FOUNDRY™

  
Couchbase

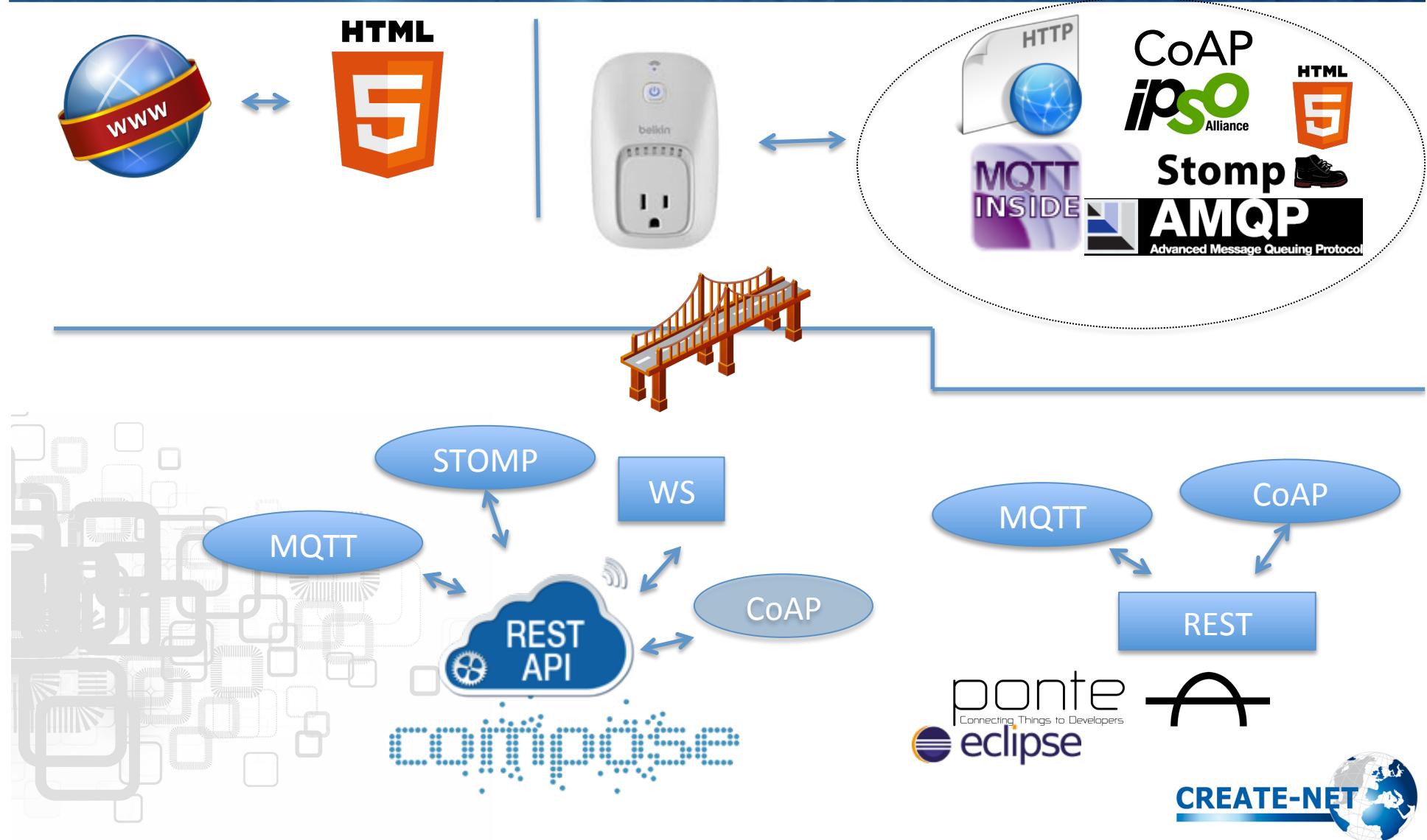


# COMPOSE at heart

## servIoTicy

- IoT Streaming made easy!
- RESTful API for:
  - Storing device (sensor) data
  - Querying & Retrieving the data
  - Pushing data to the device
  - Subscribing to notifications (e.g., new sensor data arrived)
- Free and Open Source
- Downloadable bundle (Ubuntu/Debian Image)
- Soon -> CloudFoundry installation

# WWW -> IoT



# COMPOSE at heart

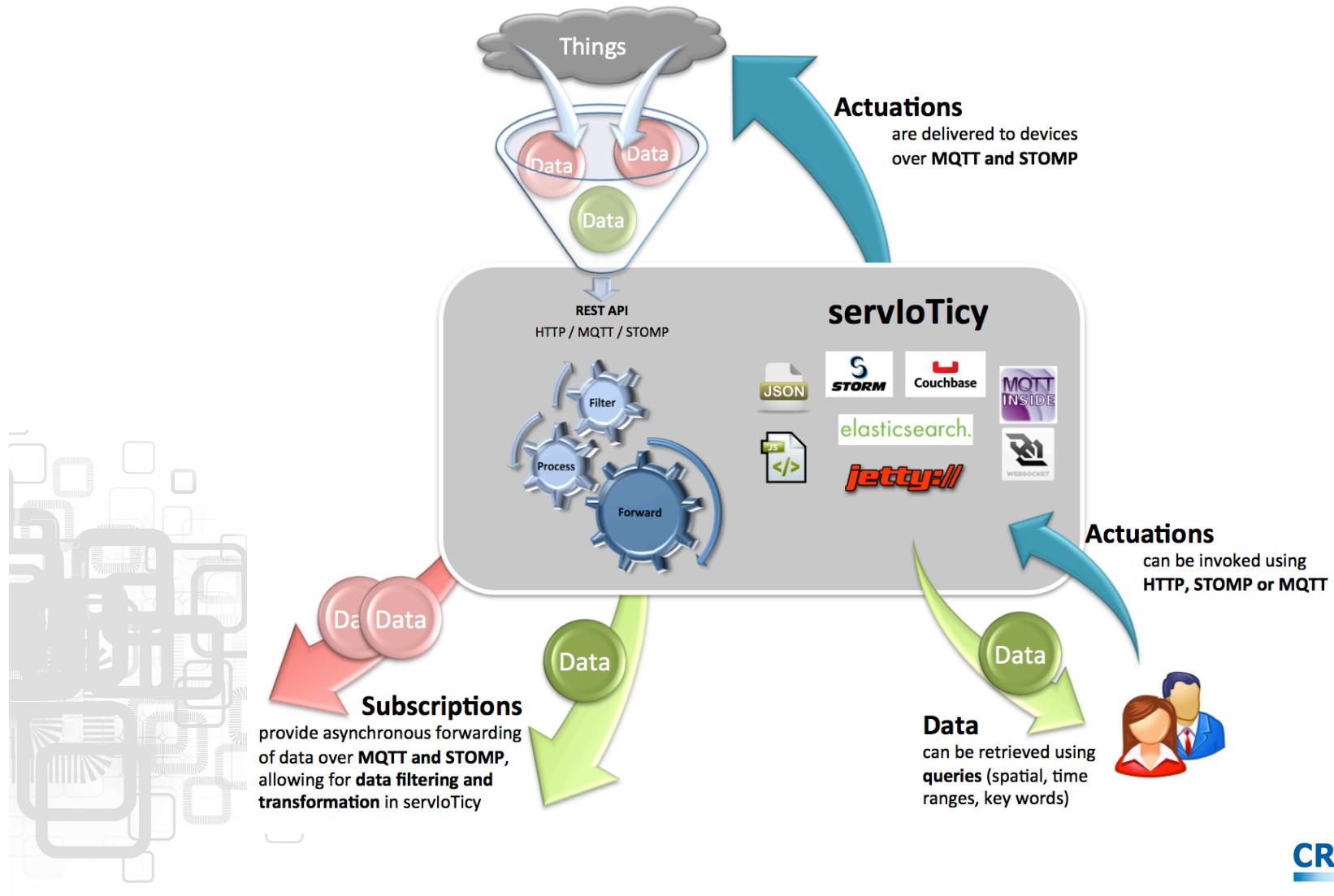
## servloTicy

- Integration of popular frameworks:
- Apache STORM
- ElasticSearch
- CouchBase DB
- Apache Apollo – ActiveMQ
  - WS/STOMP/MQTT/AMQP/...

• Jetty

• ...

# COMPOSE



# COMPOSE



servIoTicy  
*IoT streaming made easy*



COMPOSE SDKs



<https://github.com/compose-eu>

<http://www.servioticy.com>

<http://www.gluethings.com>



# COMPOSE

## Developer Friendly!

Conceptual  
idea from  
Node-RED

Active Service Composition

The screenshot shows the Node-RED interface with the following components:

- Navigation bar:** A red box highlights the left sidebar containing inputs (inject, mqtt, http, tcp, udp), outputs (debug, mqtt, http response, tcp, udp), and functions (function, template, delay, comment, http request).
- Flow diagram:** A green box highlights a flow starting from a "Tick" node, followed by "UK Power" and "UK Power Demand" nodes, both connected to a "Format timestamp" function node, which then connects to two "debug" output nodes.
- Edit function node dialog:** A purple box highlights the "Edit function node" dialog for the "Format timestamp" node. It shows the name "Format timestamp" and the function code:

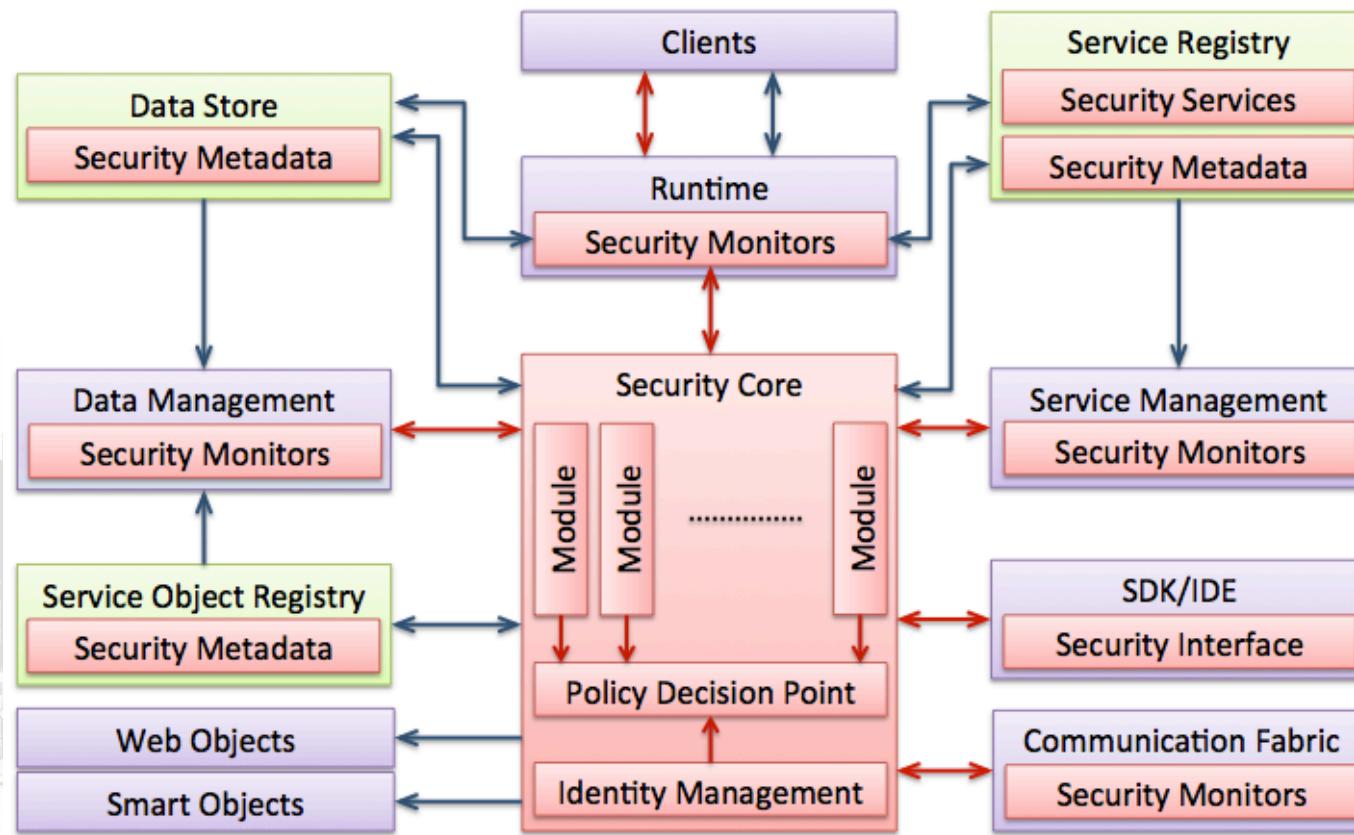
```
1 // Create a Date object from the payload
2 var date = new Date(msg.payload);
3 // Change the payload to be a formated Date string
4 msg.payload = date.toString();
5 // Return the message so it can be sent on
6 return msg;
```
- Debug tab:** A blue box highlights the "info" tab of the debug panel, displaying log entries and a message history.

Annotations on the right side of the interface:

- 1-Click-Deploy:** Points to the "Deploy" button at the top right of the interface.
- Customize source code:** Points to the "Edit function node" dialog.
- Debug Info:** Points to the "info" tab of the debug panel.

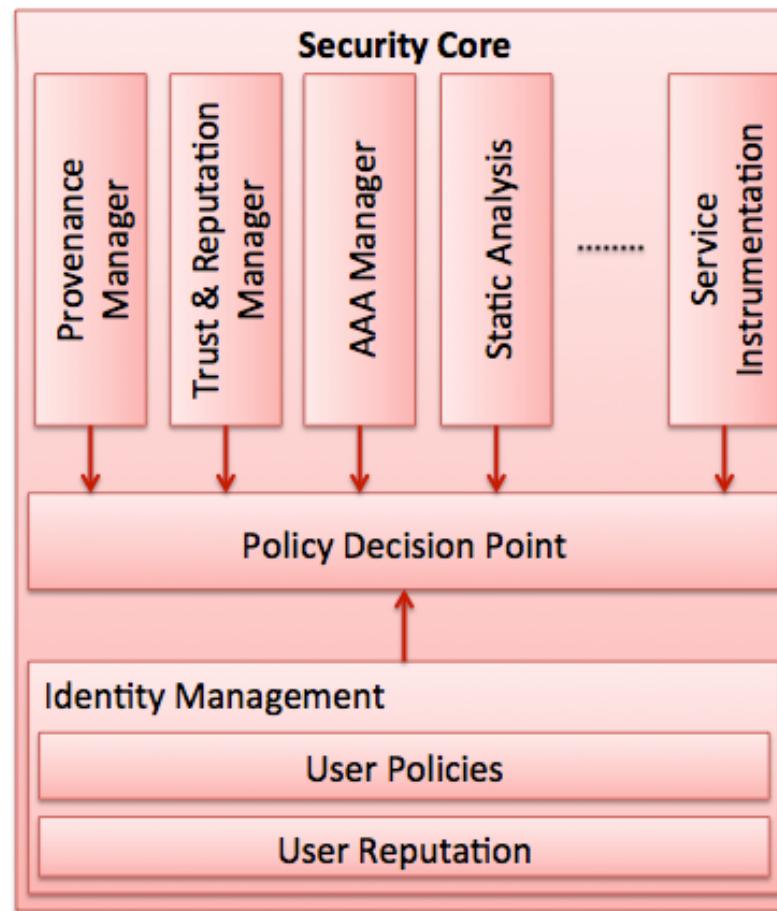
# Meet COMPOSE

**Security taken seriously!**



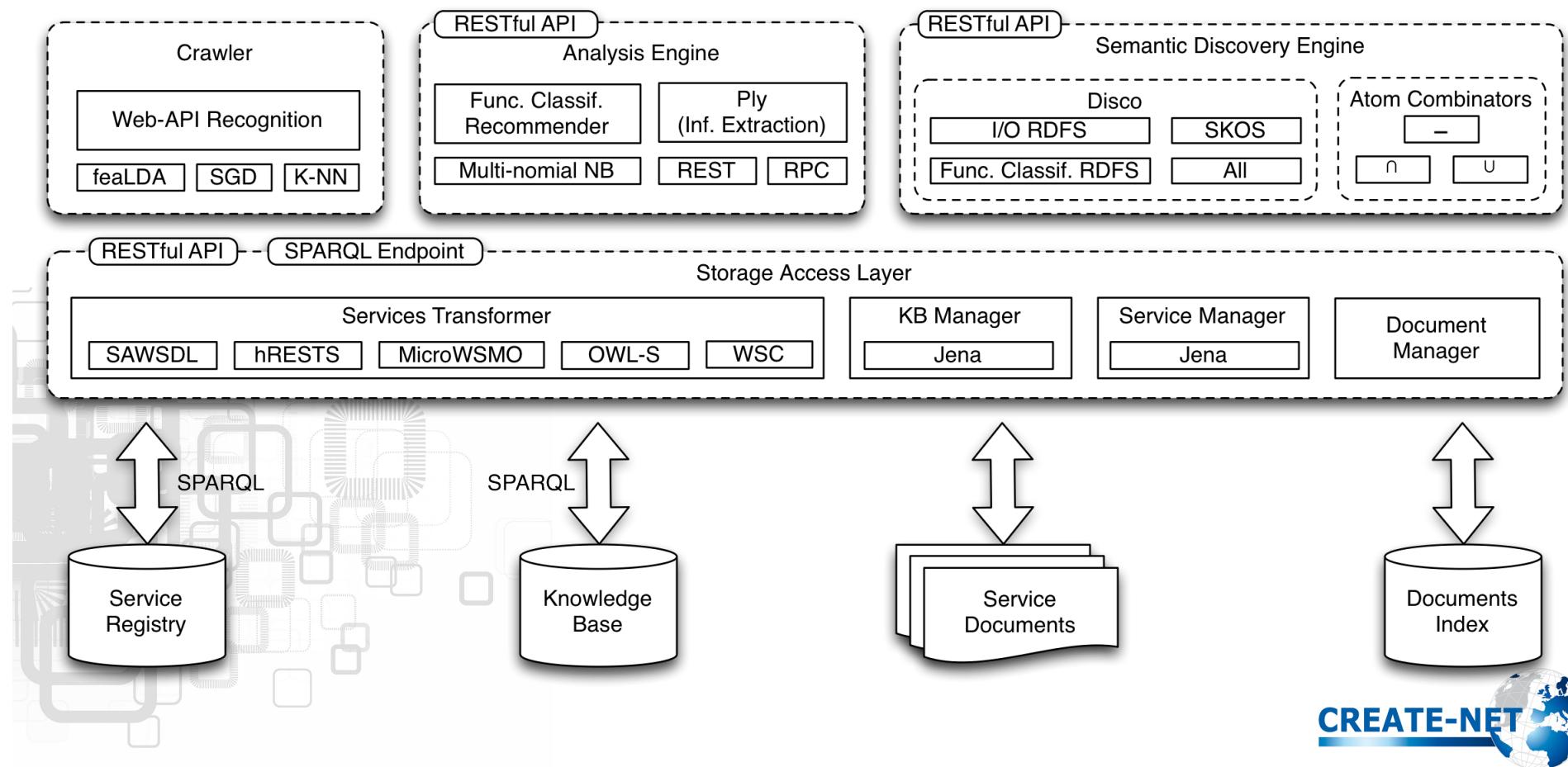
# Meet COMPOSE

**Security taken seriously!**



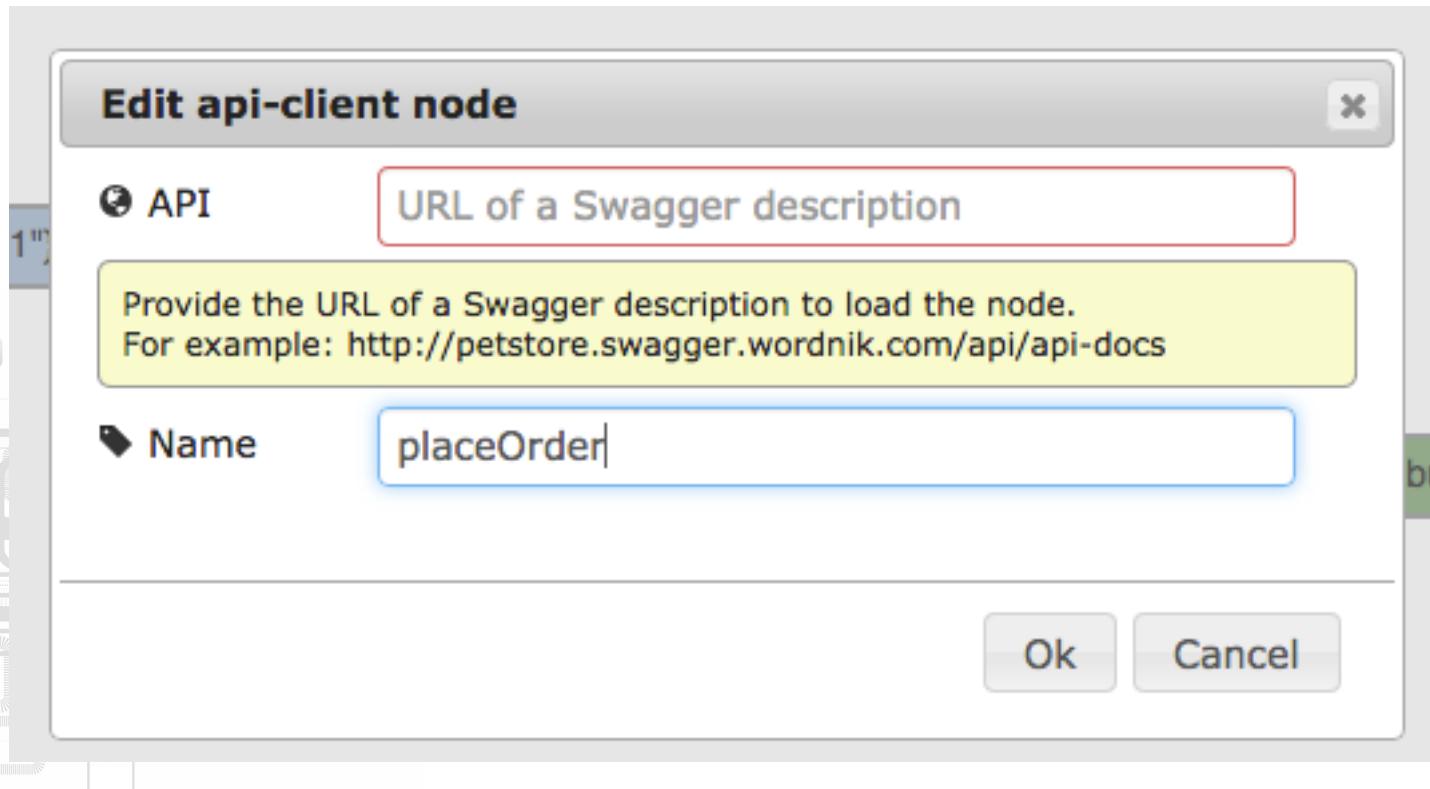
# Meet COMPOSE

## With Service Discovery!



# Meet COMPOSE

## With Service Discovery!



# Meet COMPOSE

## With Service Discovery!

The screenshot shows a modal dialog titled "Edit api-client node". Inside the dialog, there is an "API" section with a text input field containing the URL "http://petstore.swagger.wordnik.com/api/api-". Below this, a yellow box contains the text "Swagger Sample App:" followed by a detailed description of the Petstore server and how to use it.

**Swagger Sample App:**  
This is a sample server Petstore server. You can find out more about Swagger at <http://swagger.wordnik.com> or on irc.freenode.net, #swagger. For this sample, you can use the api key "special-key" to test the authorization filters

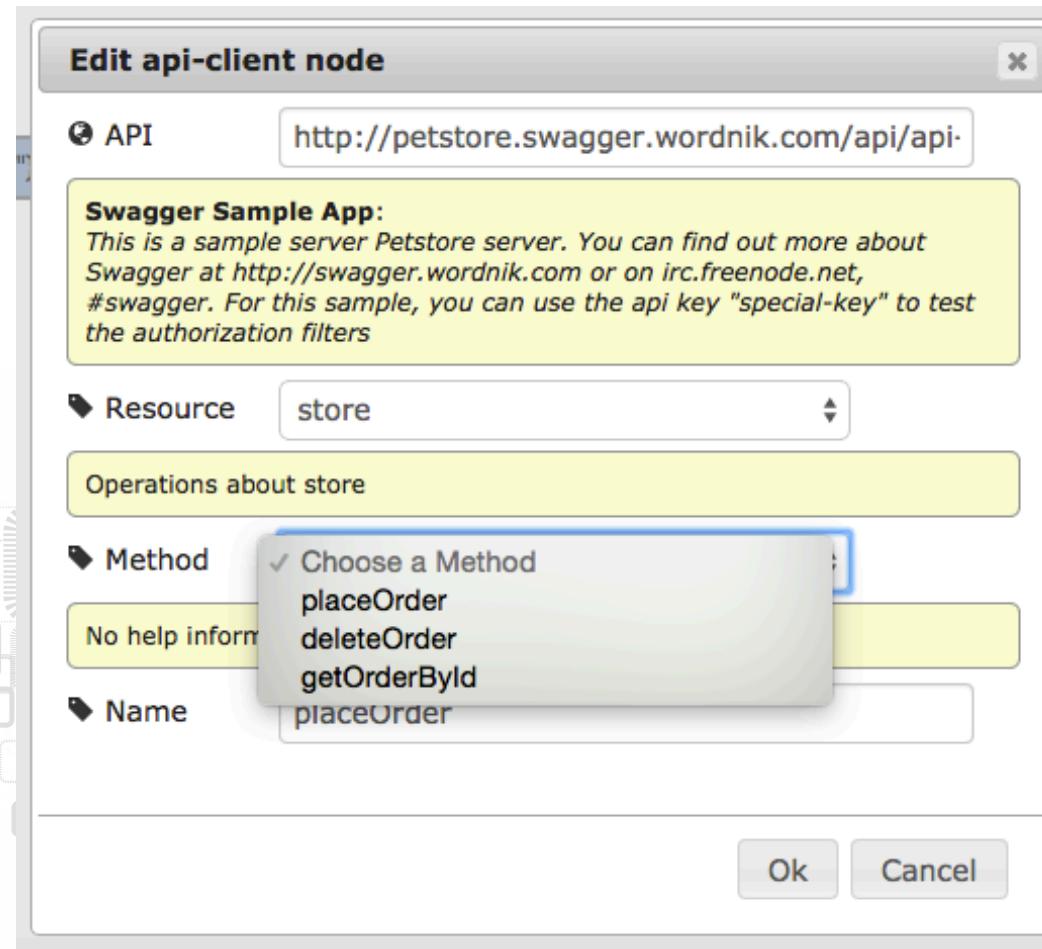
Below the API section, there is a "Resource" dropdown menu with the following options:

- Choose a Resource (selected)
- pet
- user
- store

There is also a "Name" field containing "placeOrder". At the bottom of the dialog are "Ok" and "Cancel" buttons.

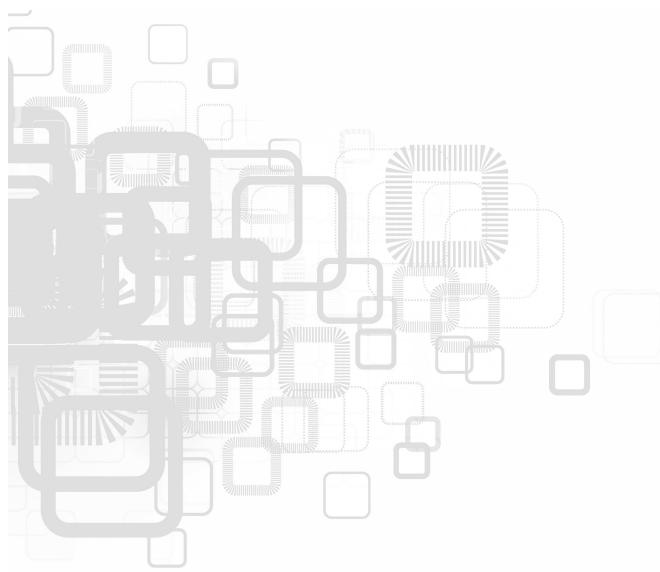
# Meet COMPOSE

## With Service Discovery!



# So how to use COMPOSE?

IDEA!



# So how to use COMPOSE?

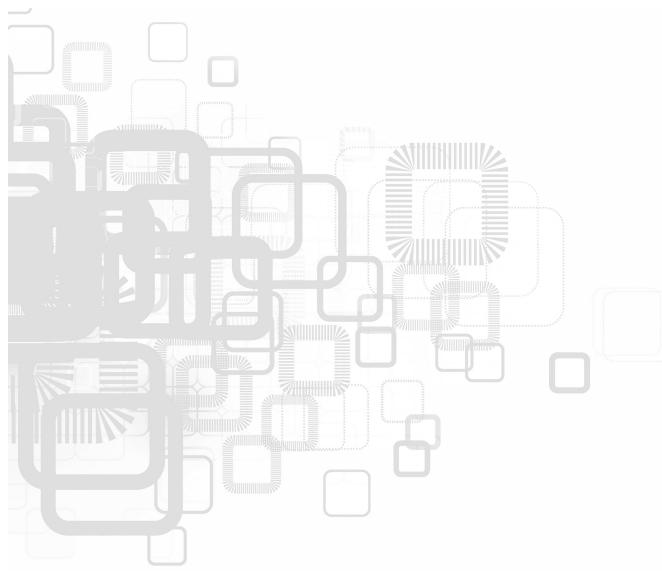
IDEA!



Smart  
Device



Smart  
Device



# So how to use COMPOSE?

IDEA!

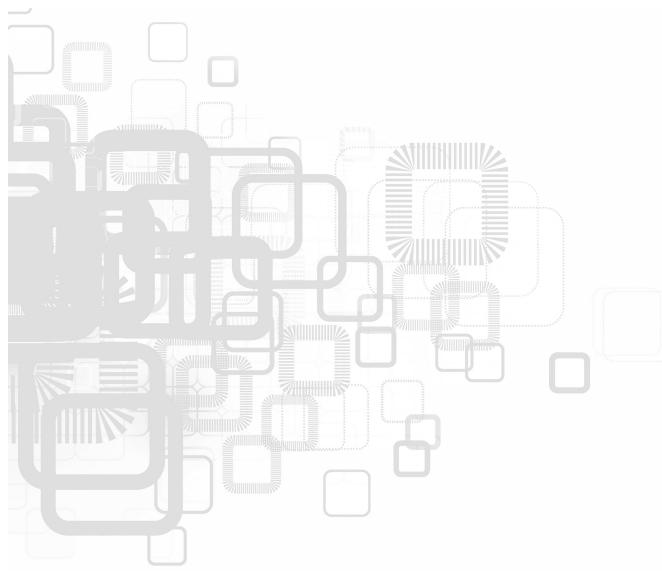
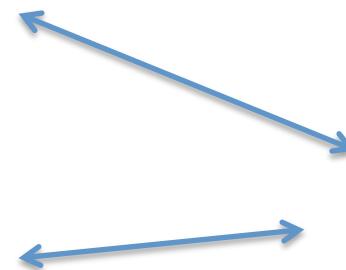


Smart  
Device

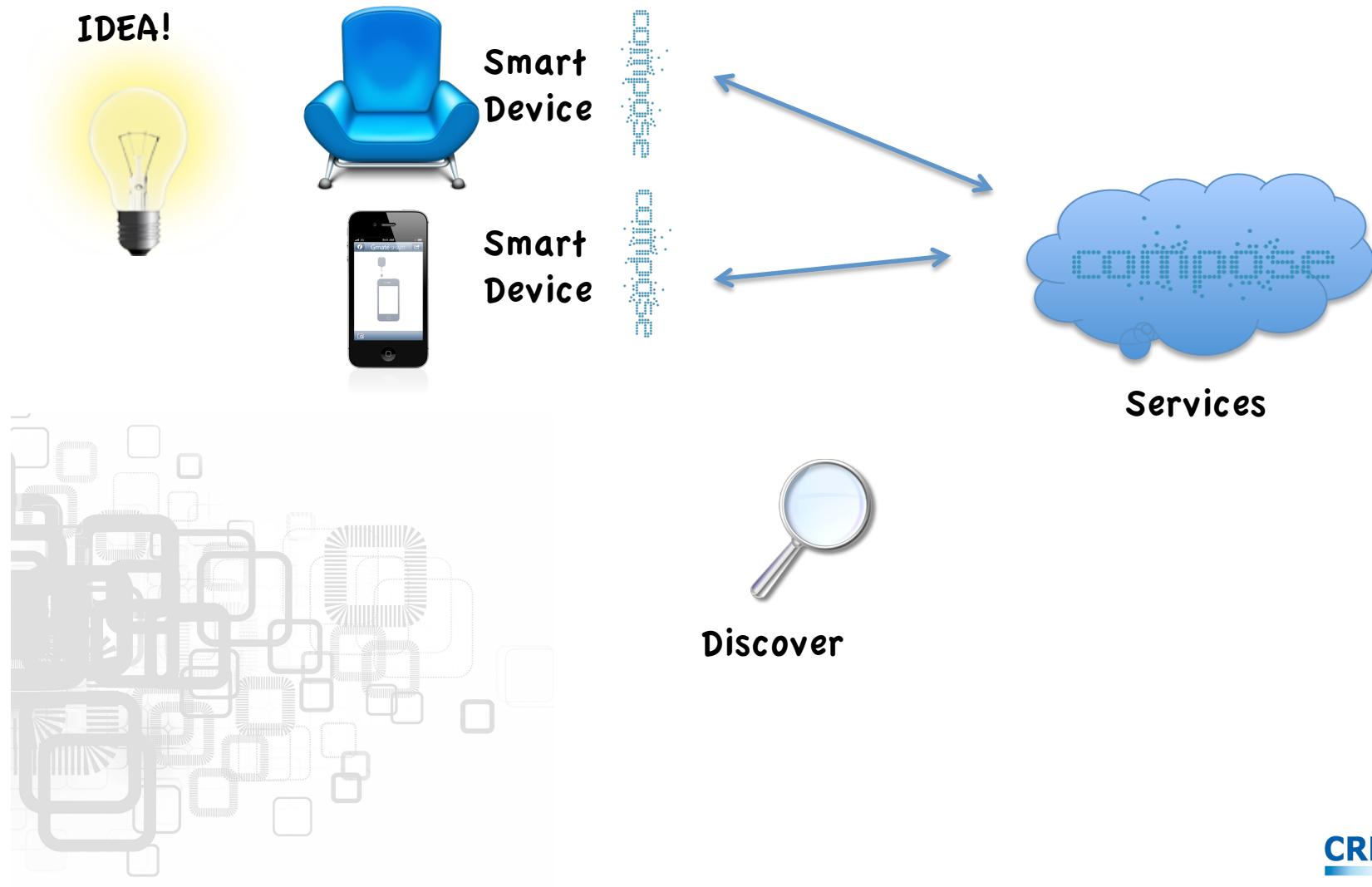


Smart  
Device

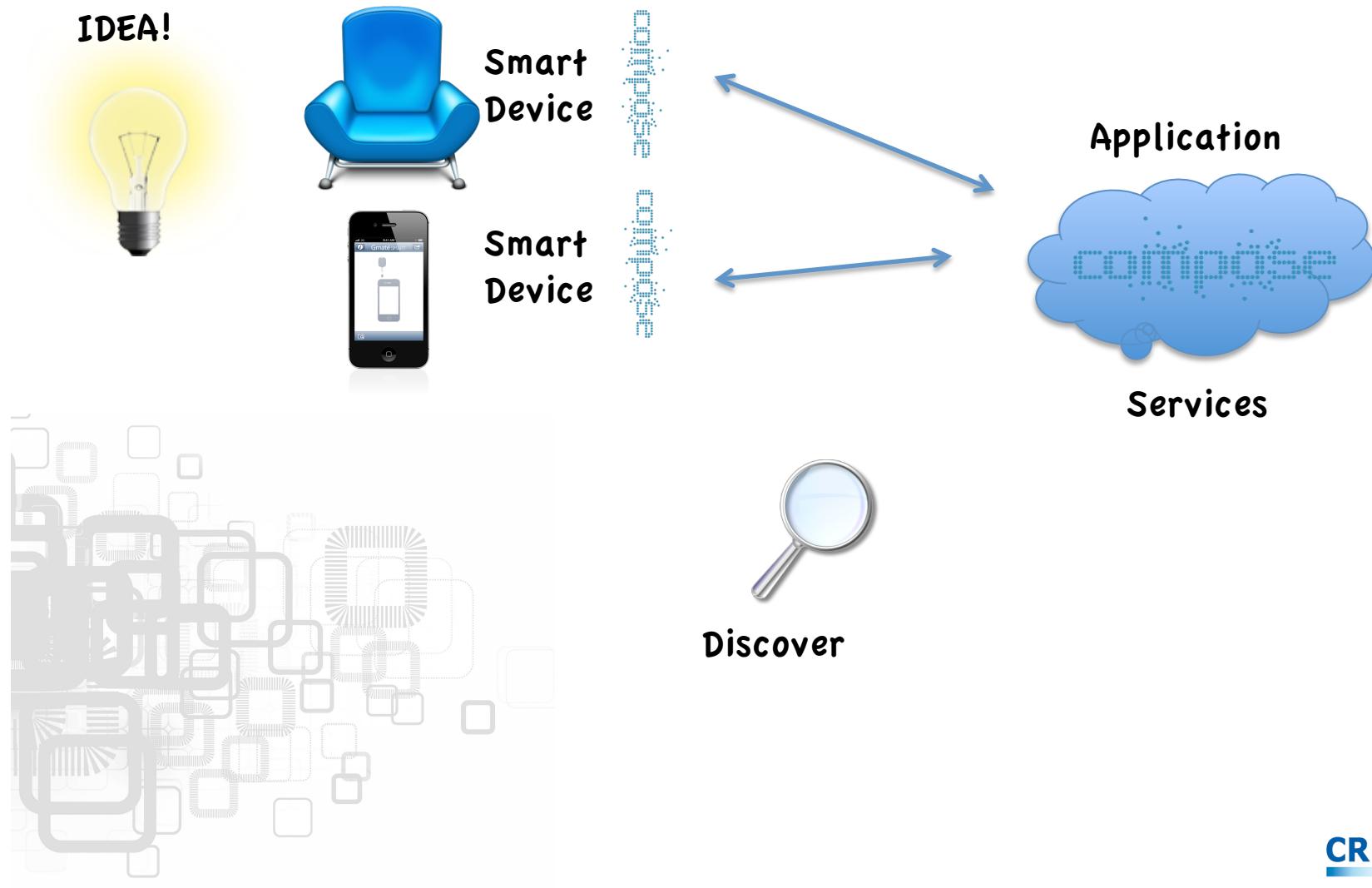
compose  
compose



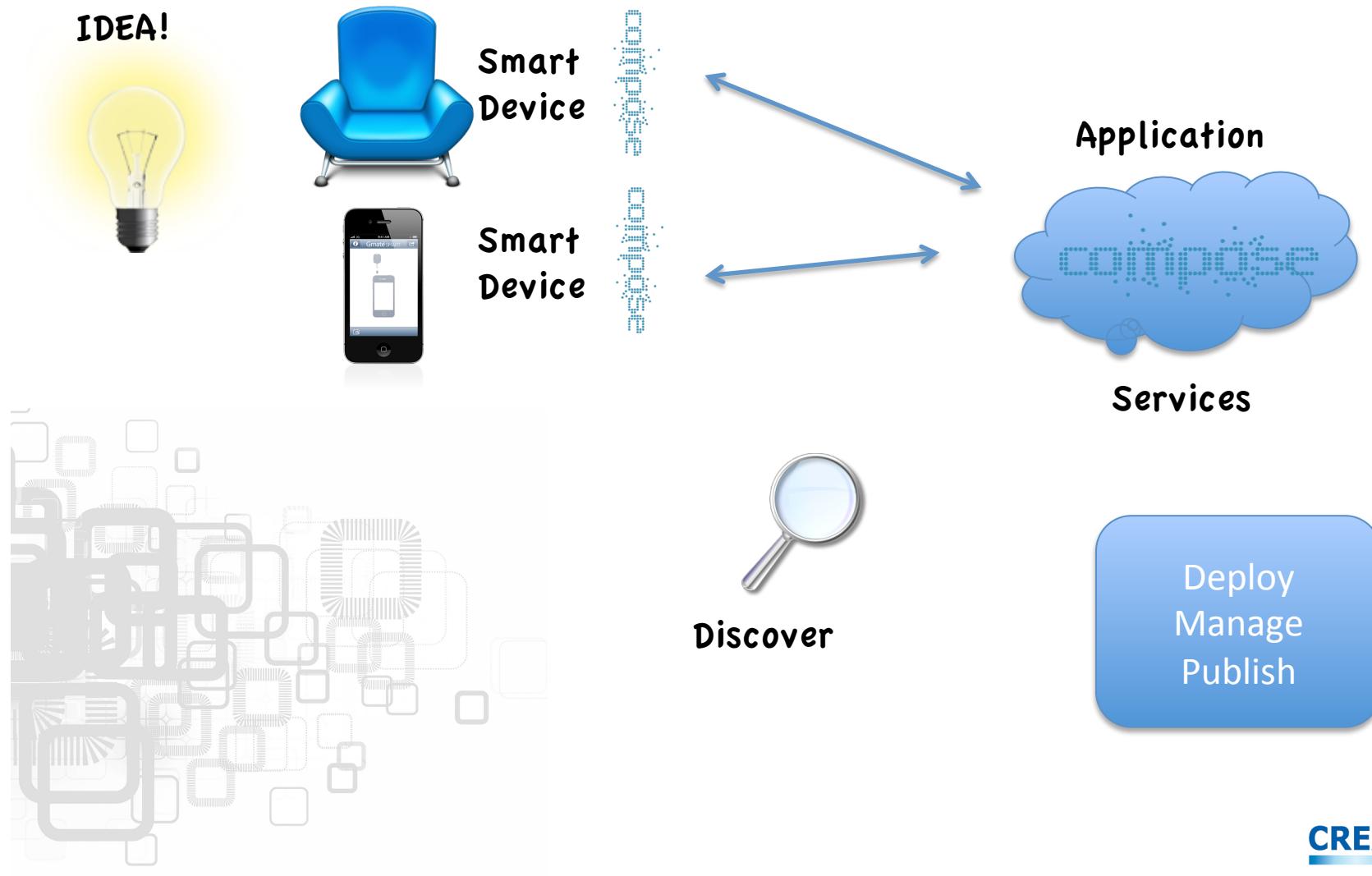
# So how to use COMPOSE?



# So how to use COMPOSE?



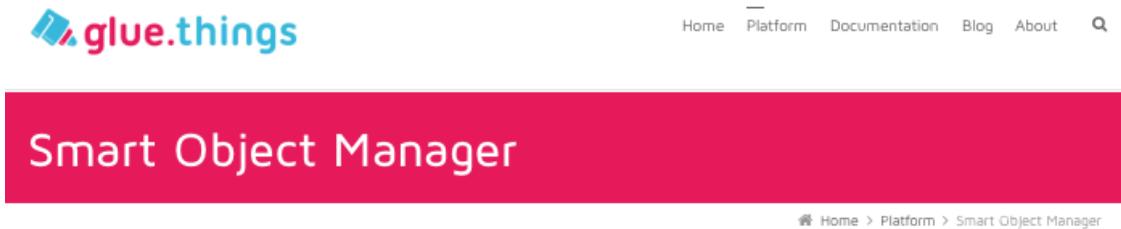
# So how to use COMPOSE?



# Step by Step...

Step 1

<http://www.gluethings.com/platform/smart-object-manager/>



Login Please enter your credentials.

A screenshot of a login form titled "Login". The form has two input fields: "Email" and "Password". Below the fields is a link "Not yet registered for the glue.things platform? Click here". At the bottom are "Login" and "Cancel" buttons. The background of the entire slide features a subtle, abstract graphic of interconnected circuit boards and data lines in shades of grey and white.

# Step by Step...

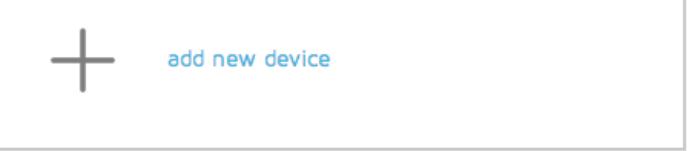
Step 2

<http://www.gluethings.com/platform/smart-object-manager/>

Logout

## My Devices



 SparkCore	 lightsensor
 temp_sensor	 DHT22
 barometer	 + add new device

**CREATE-NET** 

# Step by Step...

Step 2

<http://www.gluethings.com/platform/smart-object-manager/>

Add Smart Object Please add a name and description.

## General Information

Name

Smart Object Name

Description

Smart Object Description

Sensor Type

--- choose sensor type ---

Create

Cancel

# Step by Step...

Step 3

<http://www.gluethings.com/platform/smart-object-manager/>



Delete



temp\_sensor

Smart Object ID: 1404399517118b7f02434bf6e4bceb82046a65fb63bc

Smart Object Mode: public

Smart Object Status: //

Smart Object Last Update: 7/3/14 4:58:37 PM

API Token:

NDAwNmZjNWYtNTYxMS00YjVkJTIIYzMtZTlxNTNhMzAwYjAONjg5NjjNDct  
MWUzNC00ZDY4LTgyZDltODkzMjVjNDIOYTii

API Endpoint: <http://api.servioticy.com/>

temperature in

Data Stream Name: temperature

Data Stream Description:

Last Updated: 7/3/14 4:58:37 PM

[Raw data](#)

Data Stream Endpoint

<http://api.servioticy.com/1404399517118b7f02434bf6e4bceb82046a65fb63bcstreams/temperature>



# Step by Step...

## Step 4

[https://github.com/compose-eu/  
COMPOSE\\_client\\_libraries](https://github.com/compose-eu/COMPOSE_client_libraries)

```
//Initialise the client object for connectivity:  
EthernetClient client;  
...  
void setup() {  
    ....  
    //Set the Authorisation key from COMPOSE:  
    setAuthkey("YOUR_KEY_GOES_HERE");  
  
    //as an example, read an analog pin input:  
    String value = String(analogRead(A0));  
  
    //Populate the channels with data  
    setChannelData("temperature", value);  
    setChannelData("user", "arduino");  
    setChannelData("location", "universe");  
  
    //Store the data into the SO, defining the stream name and the Service Object ID:  
    submitSO("ArduinoSensor", "1395851631319a3fe402338134ca498cfef5d00fd8ace");  
}
```

# Step by Step...

Step 5

<http://www.gluethings.com/platform/smart-object-manager/>



## temperature in

Data Stream Name: sensor

Data Stream Description:

Last Updated: 7/3/14 5:41:38 PM

[Raw data](#)

Value: 25.000000 , Date: 7/3/14 5:49:07 PM

Value: , Date: 7/3/14 5:49:14 PM

Value: 25.000000 , Date: 7/3/14 5:49:25 PM

Value: , Date: 7/3/14 5:49:32 PM

Value: 25.100000 , Date: 7/3/14 5:49:43 PM

Value: , Date: 7/3/14 5:49:50 PM

Value: 25.000000 , Date: 7/3/14 5:50:00 PM

Value: , Date: 7/3/14 5:50:08 PM

# Step by Step...

## Step 6

<http://your.nodered.installatino:1880>

The screenshot shows the Node-RED interface running in a browser window at [localhost:1880/#](http://localhost:1880/). The left sidebar contains various node categories: IoT (temperature, Pin2, every 2 minutes, getTemp), analysis (sentiment), advanced (arduino, HueNode, Discover, ping, rawserial, rawserial, arduino, exec), and a filter node. The main canvas displays a flow across four sheets:

- Sheet 1:** A "temperature" node is connected to a "test" node. A "Pin2" node is also connected to the "test" node. The output of the "test" node goes to a "debug" node.
- Sheet 2:** An "every 2 minutes" node is connected to a "getTemp" node. The output of the "getTemp" node goes to a "debug" node.
- Sheet 3:** The output of the "debug" node from Sheet 1 is connected to a "Tweet" node. The output of the "Tweet" node goes to a "f" node.
- Sheet 4:** The output of the "f" node goes to a "HueNode" node.

The right panel shows the configuration for the "HueNode" node, which is of type "HueNode" with ID "eb2ab2a8.14d55". It has properties for name, username, discovery\_mode, lamp\_id, color (set to EBF5FF), brightness (set to 100), and lamp\_status. A note below the configuration states: "This node implements some basic functionality for managing a Philips Hue wireless Lamp system. To use it you need to have obtained a valid auth token (or username) from your Philips Hue Bridge. Read [here](#) on how to do this. You can enter the ID (1, 2, ...) of a Lamp and turn it ON or OFF, set the color and the brightness (0->100). By setting the status to AUTO, you can set the lamp parameters using the message on the input node as follows:

- msg.lamp sets the lamp ID

"



# Step by Step...

Step 6

COMPOSE - Cloudfoundry

Node.JS , Java

COMPOSE APIs, SDKs



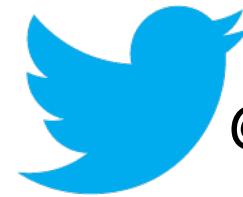
It scales...



# Find out more!



<http://www.compose-project.eu>



@COMPOSE\_Project



<https://www.facebook.com/groups/Compose.Eu.Project/>



Stay in touch!

# HACKATHON



WORKSHOP!



Competitions



# Charalampos Doukas

# Researcher

- Information & Communication Systems Engineer
  - PhD in Health Informatics
  - Senior Researcher in CREATE-NET
  - COMPOSE Project

# Maker



- IoT Blogger
    - Many DIY IoT Projects
  - Workshops (Makerland, Makerfaire...)
  - Author
    - Building Internet of Things with the Arduino
  - Consultancy

