WATER CONSERVATION

SAVING THE WORLD ONE SMART SPRINKLER AT A TIME

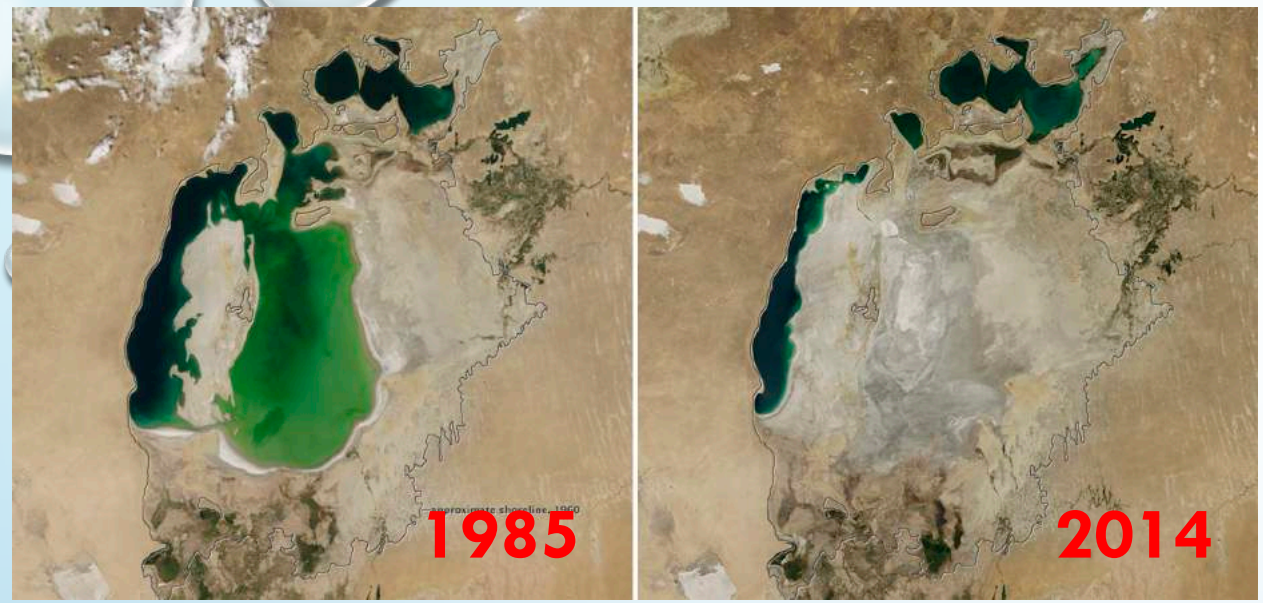
TECH CHALLENGE 2016 – THE WEATHER COMPANY

SAMANTHA CHAN MARY KOMOR CONG LI KENDRICK WONG

HOW OFTEN HAVE WE SEEN THIS?

Government Water Restriction

People Do Not Always Honour Water Restrictions

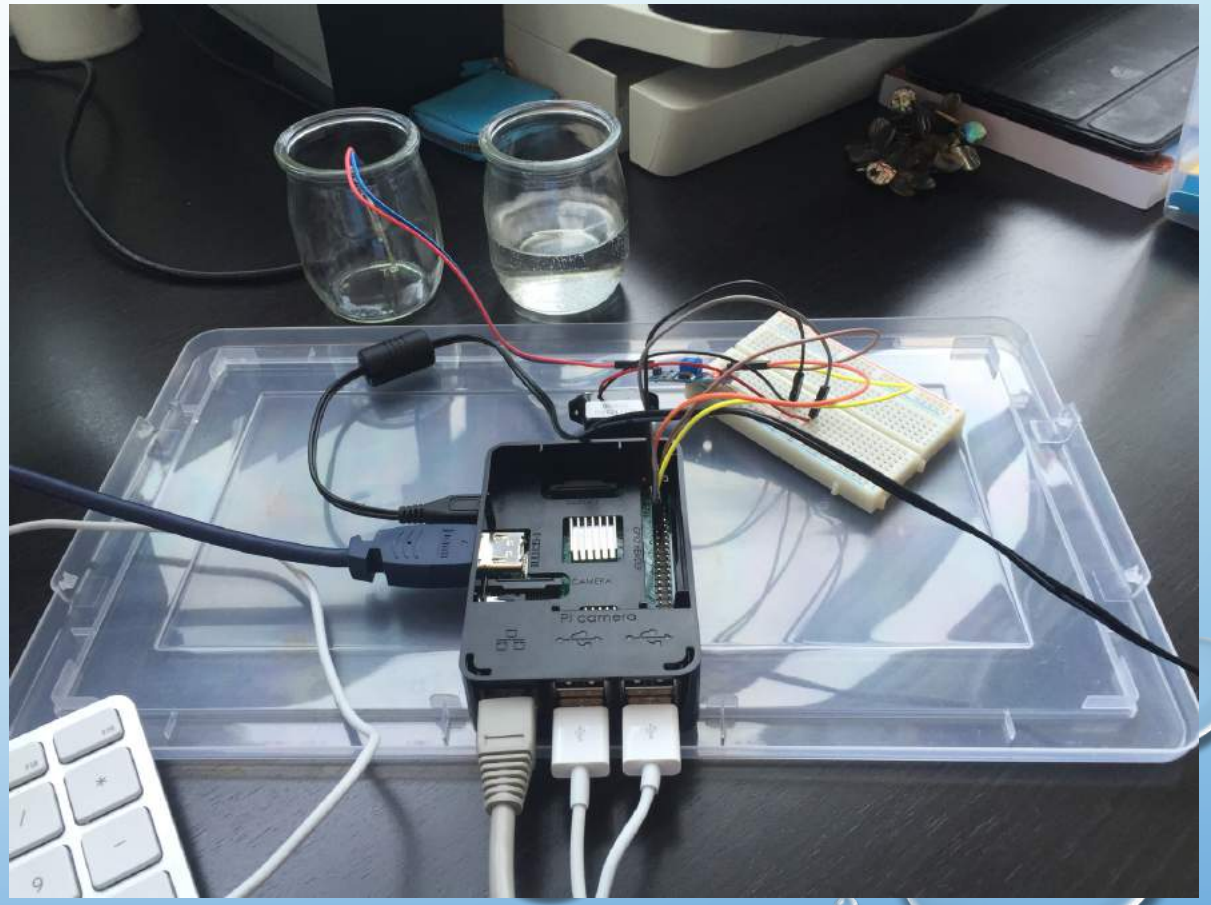


The Aral Sea's Eastern Basin Is Dry for First Time in 600 Years

The Colorado River Is Dying

WATER CONSERVATION

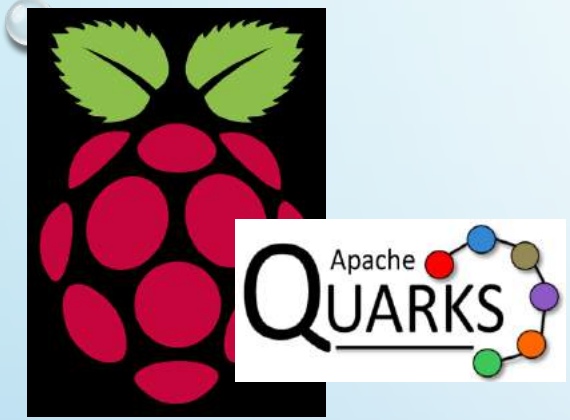
- USING SMARTER SPRINKLERS
  - LOCAL, CONTINUOUS, REAL TIME ANALYSIS TO DETERMINE THE NEED TO WATER
  - CONNECTS TO BACK END SYSTEMS FOR IN-DEPTH ANALYSIS
  - CHECKS SOIL MOISTURE
  - TAKES WEATHER INTO ACCOUNT
  - HONORS GOVERNMENT WATER RESTRICTIONS
  - EFFECTIVE IRRIGATION BASED ON HISTORICAL AND PREDICTIVE ANALYSIS



DEMO

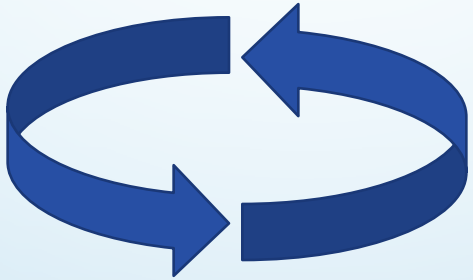
- SIT BACK AND ENJOY THE SHOW

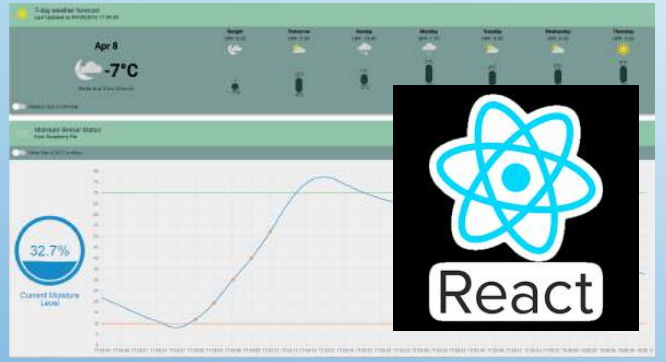
Smart sprinkler device
The hub that connects the actual sprinkler to back end systems



Smart sprinkler – Raspberry Pi
Local analytics running on Apache Quarks

Analytics to determine whether the sprinkler can be turned on
-- Weather forecast
-- Governance (water ban)





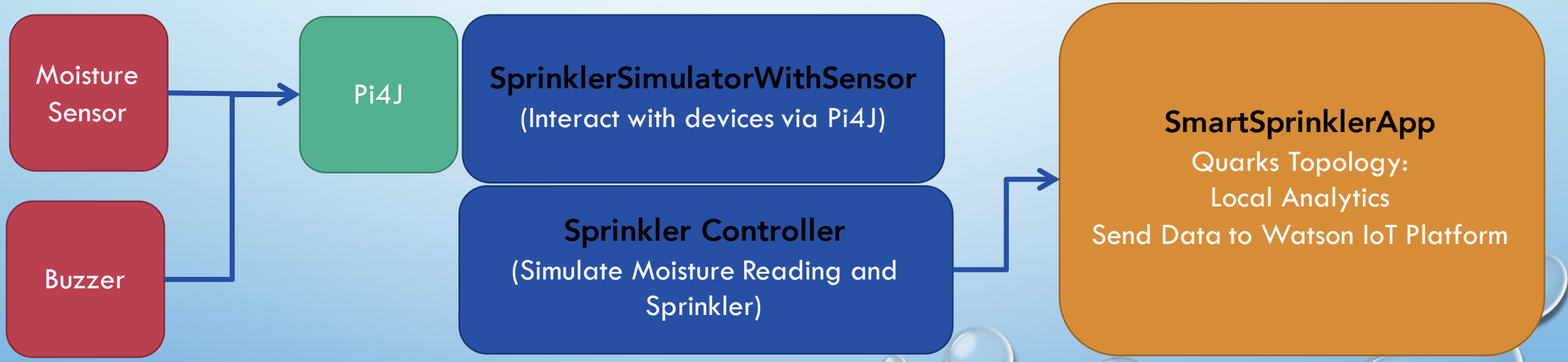
Rapid web UI development

Visualization server
Connects the browser UI to other components

The Weather Company API
Provides weather data

QUARKS APPLICATION DESIGN

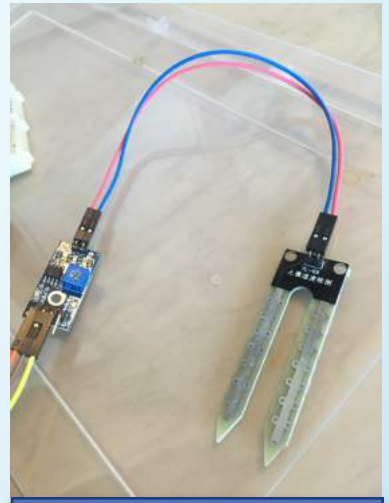
- Continuously monitors moisture reading from soil
- Calculate rolling average from a window of data
- If the soil is too dry, send request to turn on sprinkler
- Only turn on sprinkler if the backend service sends a command to turn on sprinkler

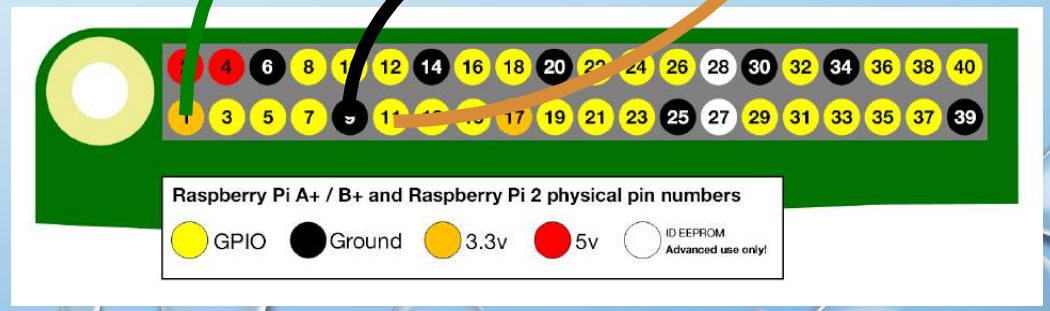


The Setup – Moisture Sensor

- Raspberry Pi 2

- VCC – 3v3 (Pin 1)
- GND – GND (Pin 9)
- DO (Digital Out) – GPIO 17 (Pin 11)
  - Provide reading whether there is moisture detected
  - Returns LOW if there is moisture, HIGH if it's dry.

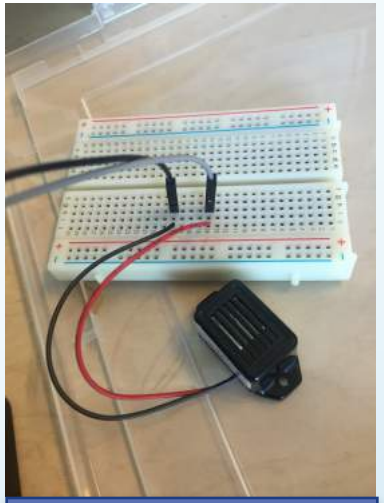




The setup – Sprinkler (aka buzzer)

- Raspberry Pi 2

- GND – GND (Pin 6)
- GPIO 27 (Pin 13)



SPRINKLER CONTROLLER

```java
/**
 * Sprinkler controller interface for application
 *
 *
 */
public interface ISprinklerController extends Supplier<Reading>{

    /**
     * Turn the sprinkler on or off
     * @param on true if turning sprinkler on, false otheriwse
     */
    public void setSprinkler(boolean on);

    /**
     *
     * @return if the sprinkler is on or off
     */
    public boolean isSprinklerOn();

    /**
     * Schedule for rain to come
     * @param delay - how long to delay rain
     * @param duration - how long to rain for
     */
    public void scheduleRain(final long delay, final long duration);

    /**
     * Check if rain is scheduled
     * @return true if rain is scheduled, false otherwise
     */
    public boolean isRainComing();

    /**
     *
     * @return current moisture level
     */
    public double getMoisture();

}
```

- Get Moisture Reading, Control Sprinkler, Control Rain Simulation

- Provide moisture reading to Quarks Toopology
  - Implements Supplier<Reading> interface to provide data to Quarks Topology

- Reading – implements Serializable

```java
//sensor:  id, lat, long, sprinklerOnOff, moisture

public class Reading implements Serializable {

    // coordinates from GPS
    private double latitude = 43.7001100;
    private double longitude = -79.4163000;

    // sprinkler is on or off
    private boolean sprinkler_on = false;

    // current moisture level
    private double moisture = 60;
```

SPRINKLER SIMULATOR

```
10 // This class implements Supplier<Reading> to provide Reading object for Topology
11 public class SprinklerSimulatorWithSensor extends SprinklerSimulator {
12
13     private static final long serialVersionUID = 1L;
14
15     GpioPinDigitalOutput buzzer;
16     GpioPinDigitalInput moistureSensor;
17
18     final GpioController gpio = GpioFactory.getInstance();
```

PROVISION GPIO PINS

```
public SprinklerSimulatorWithSensor() {

    moistureSensor = gpio.provisionDigitalInputPin(RaspiPin.GPIO_00);

    buzzer = gpio.provisionDigitalOutputPin(RaspiPin.GPIO_02, PinState.LOW);
    buzzer.setShutdownOptions(true, PinState.LOW);
}
```

- Pi4J follows abstract pin number pin from WiringPi Project
- http://pi4j.com/pins/model-2b-rev1.html#

- Moisture Sensor: Pin 11 -> GPIO_00 in Pi4J
- Buzzer: Pin 13 -> GPIO_02 in Pi4J

PROVIDING MOISTURE READING

```
@Override
public Reading get() {
    double moisture = getMoisture();
    return new Reading(moisture, isSprinklerOn());
}
```

- SprinklerSimulatorWithSensor
- Implements Supplier<Reading>

```
public double getMoisture() {

    boolean isMoistureDetected = moistureSensor.isLow();

    System.out.println("Lots of moisture from sensor: " + isMoistureDetected);

    // if sensor detects that moisture is high
    if (isMoistureDetected)
    {
        moisture = 950;
    }

    return super.getMoisture();
}
```

CONTROLLING SPRINKLER / BUZZER

```
@Override
public void setSprinkler(boolean on) {
    super.setSprinkler(on);

    if (on)
    {
        buzzer.setState(PinState.HIGH);
    }
    else {
        buzzer.setState(PinState.LOW);
    }
}
```

POLLING FROM SPRINKLER CONTROLLER

```
public class SmartSprinklerApp {

    public static final double THRESHOLD_LOW = 100;
    public static final double THRESHOLD_HIGH = 700;

    public static void main(String[] args) throws Exception {

        ISprinklerController smartSprinkler = new SprinklerSimulatorWithSensor();

        DirectProvider dp = new DirectProvider();
        Topology topology = dp.newTopology();

        // poll from sensor once every second
        TStream<Reading> moistureReading = topology.poll(smartSprinkler, 1000, TimeUnit.MILLISECONDS);
```

CALCULATING ROLLING AVERAGE

```
// set up window of 9 readings
TWindow<Reading, ?> lastNReadings = moistureReading.last(9, t -> {
    return 0;
});

// calculate the rolling average of data from the window
TStream<Reading> avg = lastNReadings.aggregate((window, partition) -> {
    double sum = 0;
    for (Reading r : window) {
        sum += r.getMoisture();
    }
    double avgReading = sum / window.size();

    // turn sprinkler off if the soil has enough moisture
    if (avgReading >= THRESHOLD_HIGH) {
        smartSprinkler.setSprinkler(false);
    }

    System.out.println("Avg: " + avgReading);
    return new Reading(avgReading, smartSprinkler.isSprinklerOn());
});
```

MAKING WATER REQUESTS

```
// Send request to water if, the soil is too dry, and the sprinkler is
// not on, and no rain is scheduled
TStream<Reading> dry = avg
        .filter(reading -> (reading.getMoisture() < THRESHOLD_LOW && !smartSprinkler.isSprinklerOn() && !smartSprinkler.isRainComing()));

// send to IoTF to request to turn on sprinkler
IotDevice device = new IotfDevice(topology, new File("./device.cfg"));

// convert to json object
TStream<JsonObject> json = dry.map(v -> {
    System.out.println("Send Request: " + v.toJson());
    return v.toJsonObject();
});

// send request via device
device.events(json, "waterRequest", QoS.FIRE_AND_FORGET);
```

COMMANDS FROM IoT PLATFORM

```
// listen for commands from IotF
TStream<JsonObject> responses = device.commands(new String[0]);

// process the command
responses.sink(res -> {
    boolean sprinkler = ((JsonObject) res.get("payload")).get("approval").getAsBoolean();
    String reason = ((JsonObject) res.get("payload")).get("reason").getAsString();

    // set sprinkler on / off based on water request approval
    smartSprinkler.setSprinkler(sprinkler);

    // it's going to rain, so make rain!!!
    if (!sprinkler && reason.indexOf("rain") > -1) {
        smartSprinkler.scheduleRain(10000, 10000);
    }
});

// submit this topology for it to run
dp.submit(topology);
```

BACK-END ANALYSIS USING STREAMING ANALYTICS

Device events

Device commands

com.ibm.streamsx.iot (toolkit)

com.ibm.streamsx.iot.watson.apps::IotPlatform
(pre-built application)

Subscribe

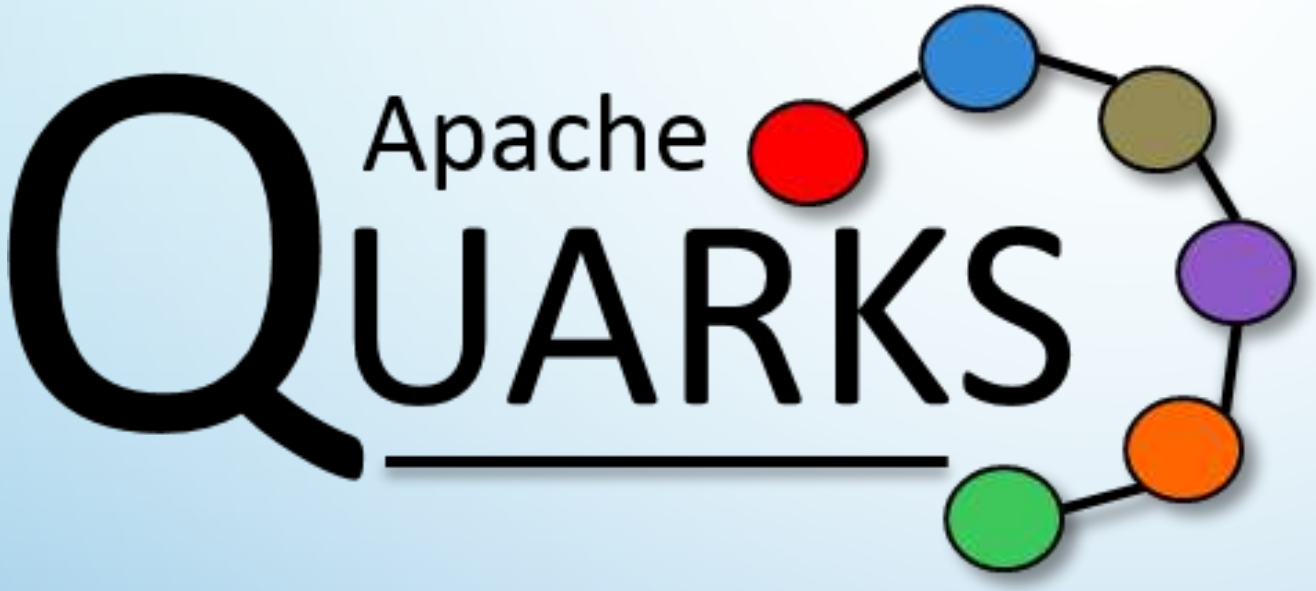
Publish

Analytics
(application)

RESOURCES

- Apache Quarks
  - http://quarks.incubator.apache.org
- Recipes @ DeveloperWorks:
  - https://developer.ibm.com/recipes/tutorials/apache-quarks-on-pi-to-watson-iot-foundation/
  - https://developer.ibm.com/recipes/tutorials/connect-apache-quarks-on-pi-to-the-streaming-analytics-service/
- IBM Streams:
  - http://www.ibm.com/analytics/us/en/technology/stream-computing/
- Streaming Analytics Service
  - https://console.ng.bluemix.net/catalog/services/streaming-analytics
- IoT Platform Bluemix Service
  - http://www.ibm.com/cloud-computing/bluemix/internet-of-things/

THANK YOU



http://quarks.incubator.apache.org

Apache Quarks is currently undergoing Incubation at the Apache Software Foundation.☐