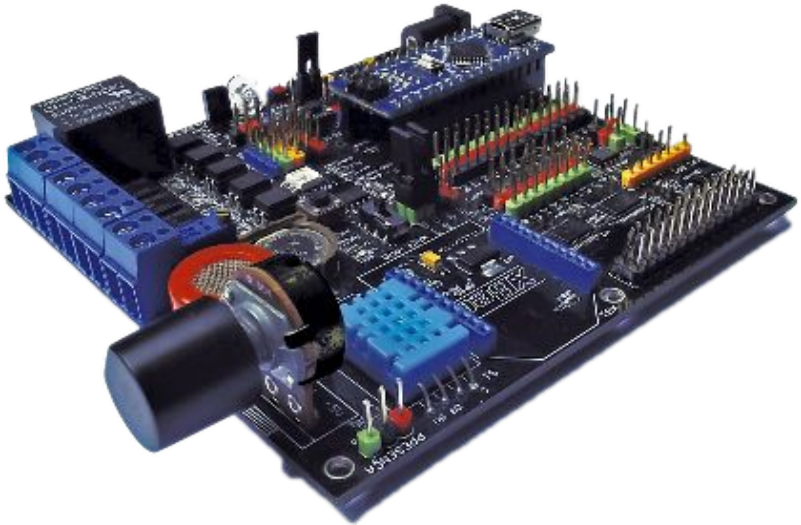

Globalcode

Java IoT Surfing API



www.globalcode.com.br/surfboard/en

Files for this class

https://portalaluno.toolscloud.net/redmine/projects/iotsurfboard/files

- [ ] Presentation: IoT_Surfing_Class_9_en.pdf

Java + IoT

- Java was born for Things:

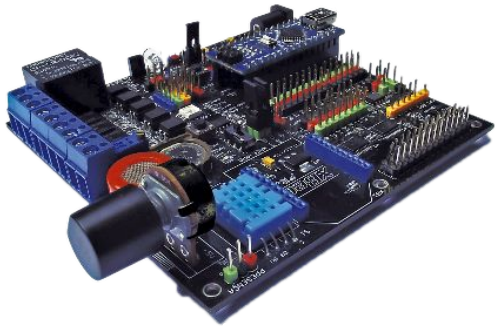
Java Platform

- Java ME: Very good shape in version 8 running on K64F Freescale
- Java SE: Nice performance on Raspberry Pi & Arm in general
- Java EE: now can run on different single-board computer!
- Java 9: modularity, profiles, etc. = Java On Devices!

Java IoT Surfing API

- Allows you to control your IoT Surfboard using Java:

USB CABLE
Serial Communication RXTX



Java IoT Surfing API: sample code

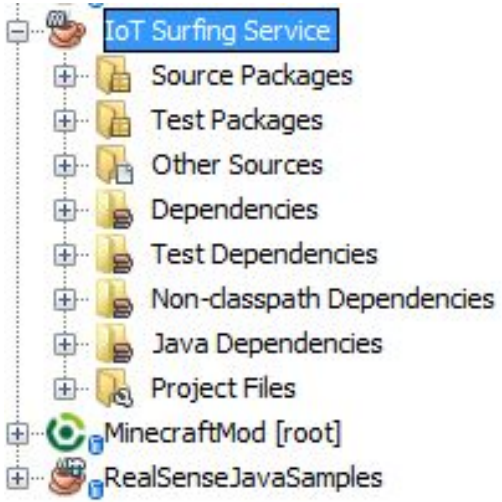
```
board = new IoTSurfboard("COM3", 9600);
System.out.println("Alcohol       :" +board.alcohol());
System.out.println("Temperature   :" +board.temperature());
System.out.println("Humidity      :" +board.humidity());
System.out.println("Light         :" +board.light());
System.out.println("Potentiometer:" +board.potentiometer());
System.out.println("Light         :" +board.light());
board.red(255);
Kernel.delay(1000);
```

RXTX Native Code

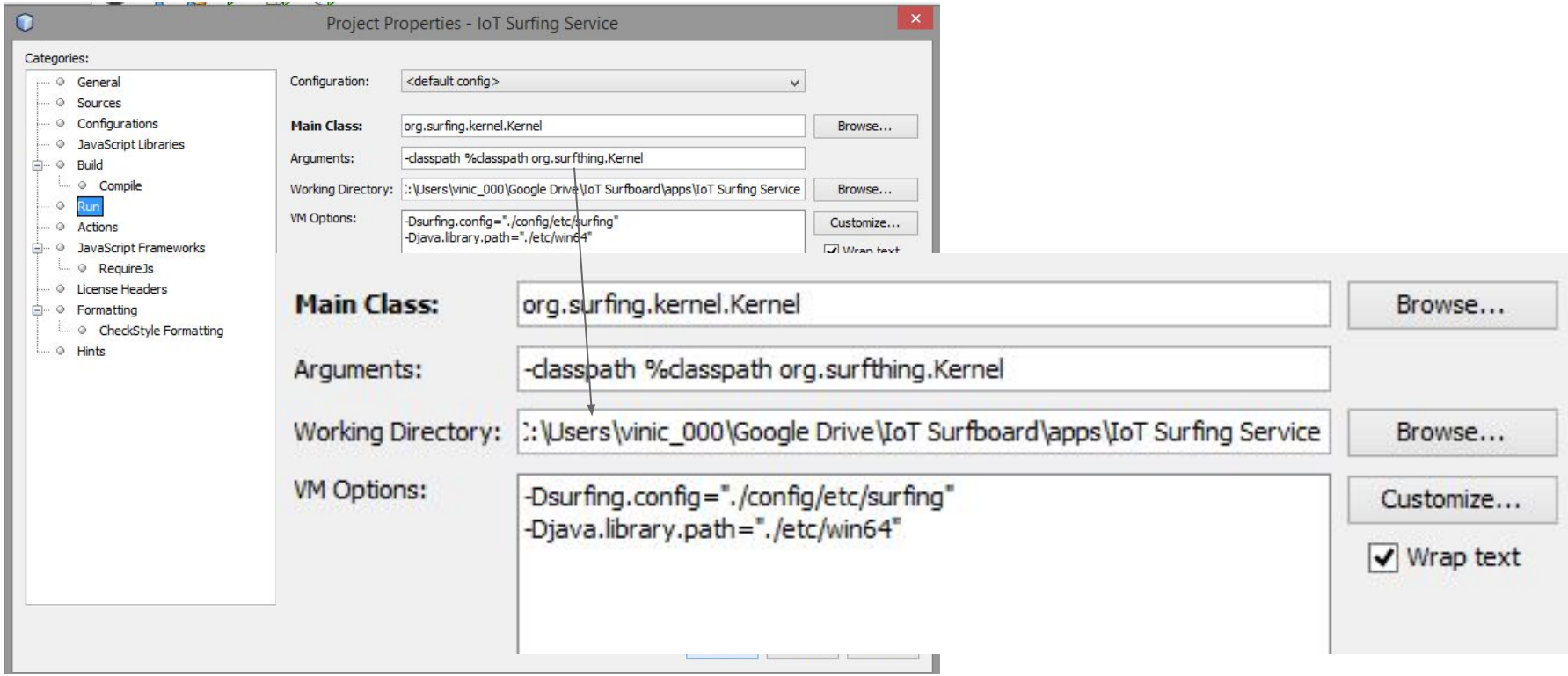
- We need to use RXTX library to establish a Serial communication
- Download & Install: http://rxtx.qbang.org/wiki/index.php/Download

Starting with Java IoT Surfing API + NetBeans

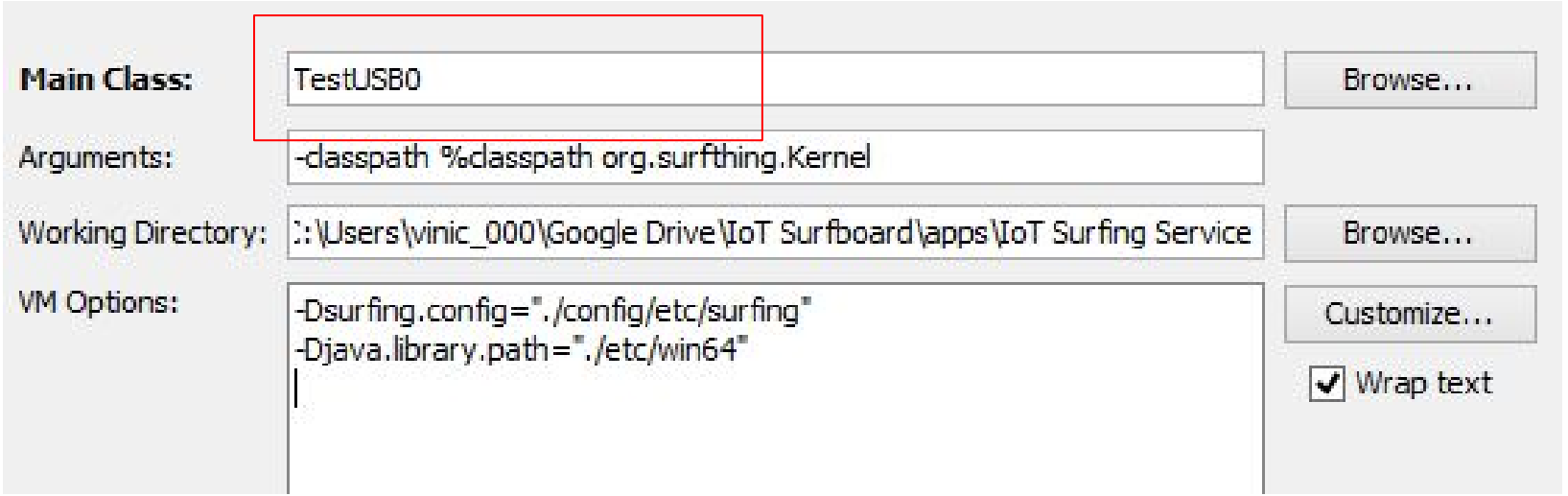
- Clone our repository: https://github.com/surfboard/service



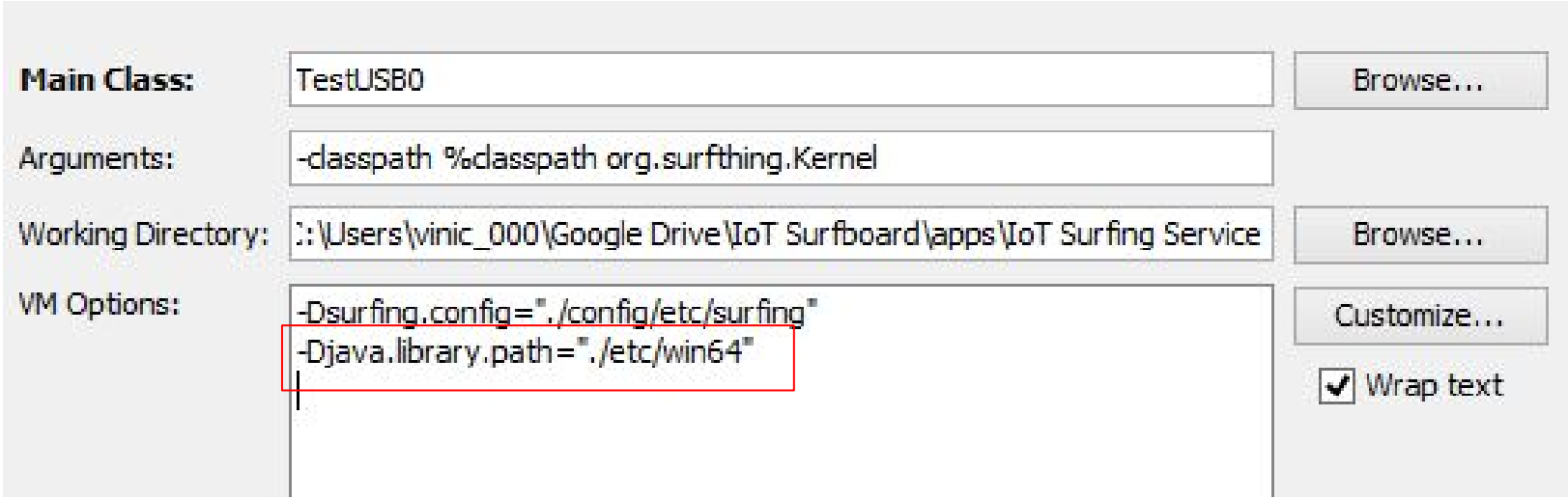
Fix the Working Directory:



Choose Main Class: TestUSB0



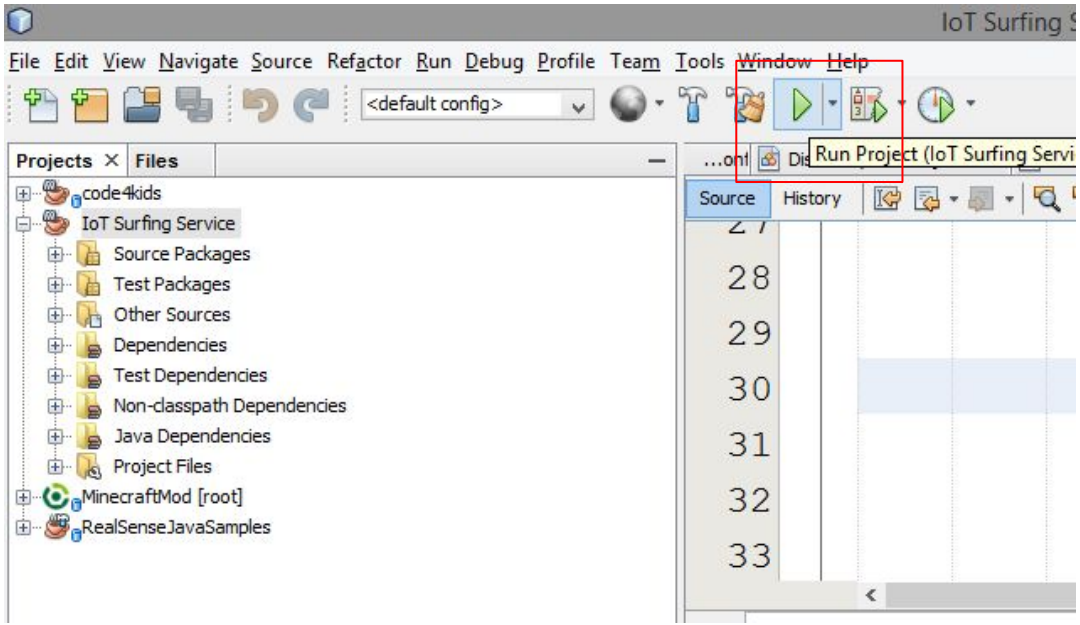
Finally setup RXTX native directory



Choose the Right Serial / Com Port

```
board = new IoTSurfboard("COM3", 9600);
System.out.println("Alcohol       :" +board.alcohol());
System.out.println("Temperature   :" +board.temperature());
System.out.println("Humidity      :" +board.humidity());
System.out.println("Light         :" +board.light());
System.out.println("Potentiometer:" +board.potentiometer());
System.out.println("Light         :" +board.light());
board.red(255);
Kernel.delay(1000);
```

Run the Project



It May fail the first time!

Run the Project

```
INFO: Connection Stabilished with //./COM3
Alcohol       :0
Temperature   :26.0
Humidity      :21.0
Light         :22
Potentiometer:872
Light         :22
Feb 08, 2016 9:54:42 AM org.surfing.device.SerialDevice close
INFO: Closing device on //./COM3
Closing FINAL //./COM3 port
```

LIVE DEMO

IoT Surfing Service

- Java Microkernel for IoT
- Provides Access to your IoT Surfboard via REST and MQTT
- MANY Services: speech, camera, FTP, Serial, Persistence, Device Discovery
- Each Service has a Service or MQTT sub-class + Config File
- Perform GREAT on Raspberry Pi!

Service Example

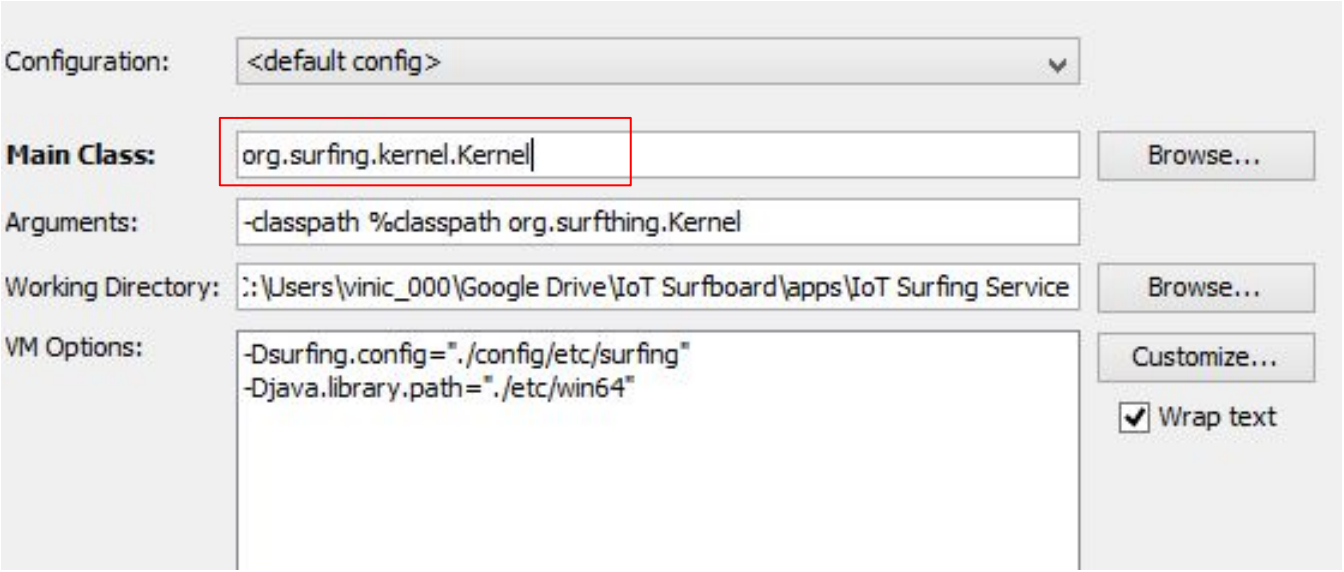
```
@Path("/myservice")
public class MyService extends MQTTController {
    @GET
    @Produces("text/html")
    @Path("/s1/{name}")
    public String execute(@PathParam("name") String name) {
        System.out.println("Name " + name);
        return "";
    }
    public void processMessage(String msg) {
        for (Device device : Kernel.getInstance().getDevices()) {
            Thing t = device.getThings().get(msg);
            if(t.getName().equals(msg)) {
                AudioTTS.speak(msg + " value is " + t.getLastValue(), true);
            }
```

myservice

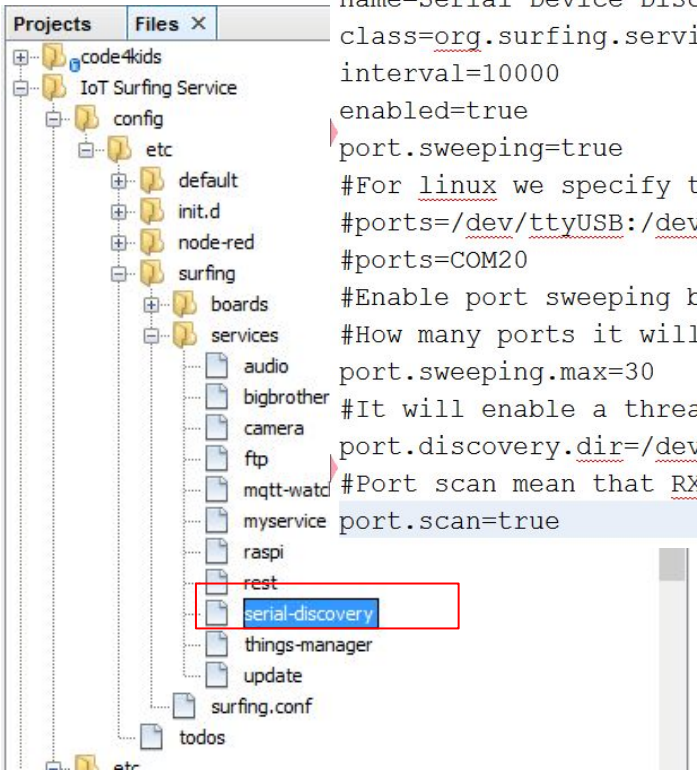
Código-Fonte | Histórico

```
1 name=My Sensor Audio Service
2 class=org.surfing.service.sample.MyService
3 interval=0
4 enabled=true
5 mqtt.queue=globalcode/things/audio
6
7
```

1. Choose class Kernel to Run

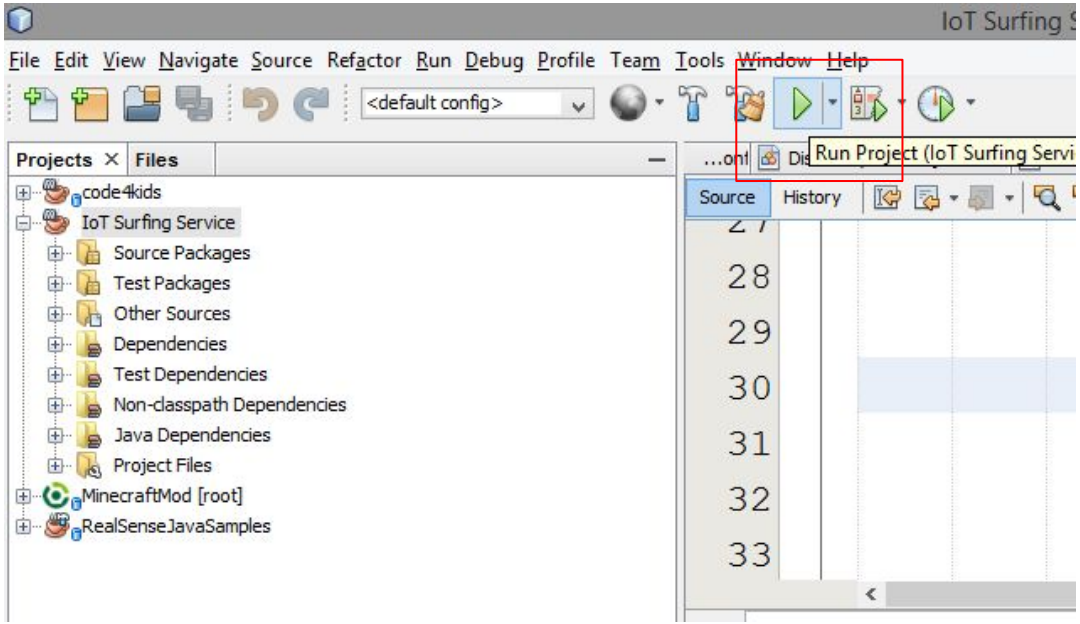


2. Check serial-discovery service config




```
name=Serial Device Discovery
class=org.surfing.service.serial.DiscoveryService
interval=10000
enabled=true
port.sweeping=true
#For linux we specify types os FT232, BT and Zigbee explorer ports...
#ports=/dev/ttyUSB:/dev/ttyACM:/dev/rfcomm
#ports=COM20
#Enable port sweeping based on the list above
#How many ports it will scan, /dev/ttyUSB0, /dev/ttyUSB1 until port.sweeping.max number
port.sweeping.max=30
#It will enable a thread to watch the specified directory and try to check if the name m
port.discovery.dir=/dev
#Port scan mean that RXTX will use CommPortIdentifier.getPortIdentifiers() and trust in
port.scan=true
```

3. Run the project



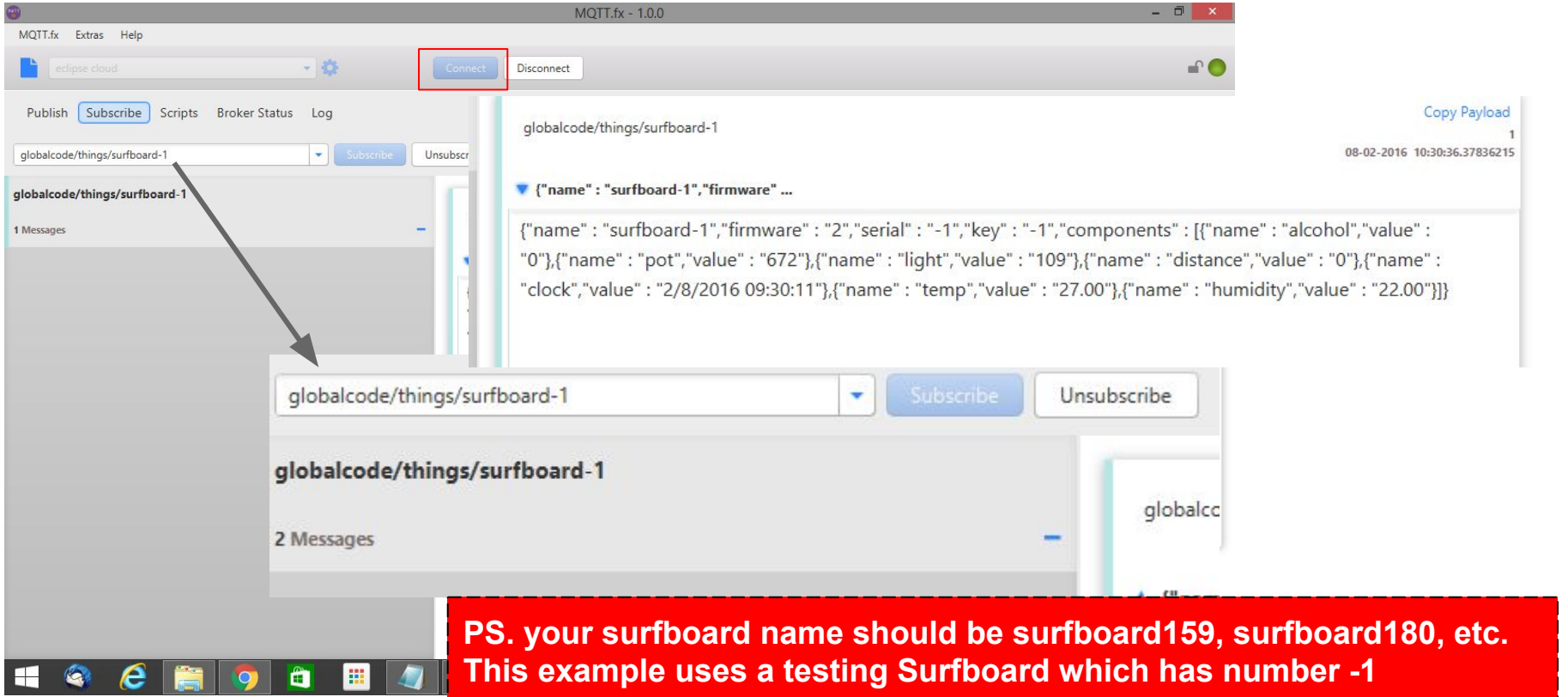
4. Plug your IoT Surfboard and Check the output

```
Feb 08, 2016 10:25:17 AM org.surfing.service.serial.DiscoveryService scanP
INFO: Scaning port: COM3
Feb 08, 2016 10:25:17 AM org.surfing.service.serial.DiscoveryService scanP
INFO: Serial Device Port found COM3. Trying to discovery this device.
Feb 08, 2016 10:25:17 AM org.surfing.device.SerialDevice open
INFO: Connection Stabilished with //./COM3
Cleaning initial data #1 null
Cleaning initial data #1 null
Cleaning initial data #1 null
Feb 08, 2016 10:25:19 AM org.surfing.service.mqtt.MQTTBaseService fixConne
INFO: Connection Stablished!
Feb 08, 2016 10:25:19 AM org.surfing.service.mqtt.MQTTController fixConnec
INFO: MQTT Receiver Subscribing globalcode/things
Feb 08, 2016 10:25:20 AM org.surfing.kernel.Kernel initServices
```

5. Manage your Surfboard via MQTT

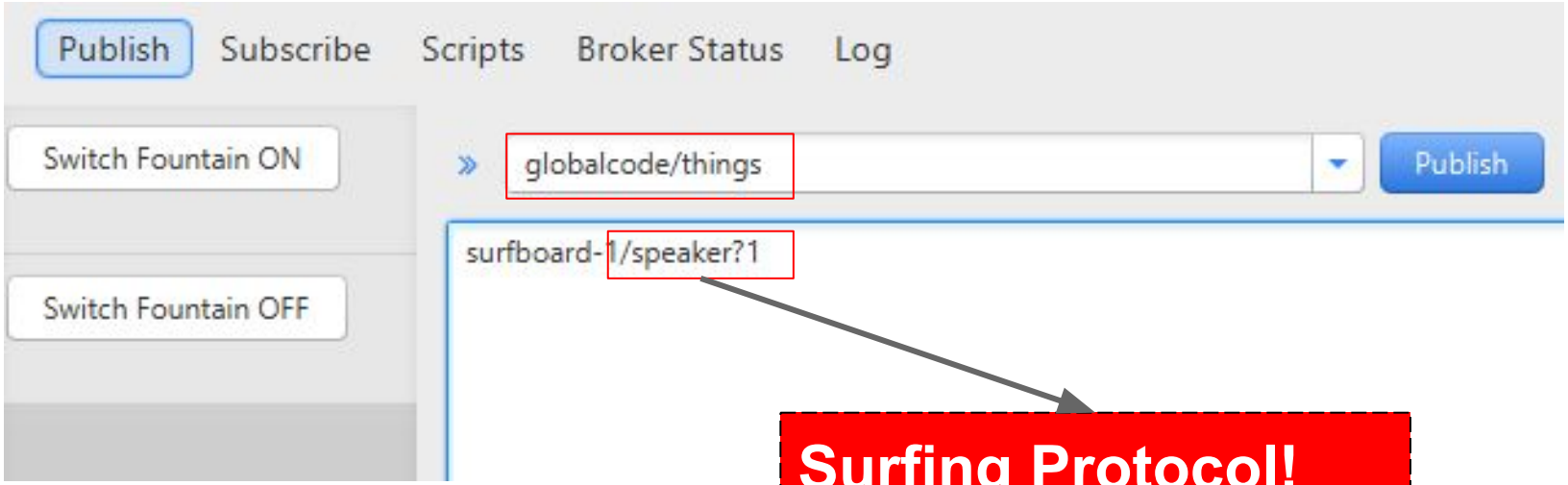
- By default MQTT service will use the Eclipse sandbox: iot.eclipse.org
- Two different Queues:
  - Sensors: globalcode/things/surfboardX (X = your surfboard number)
  - Control: globalcode/things
- You can subscribe globalcode/things/surfboardX to start receiving your sensors value!

6. Subscribe Sensors Queue using MQTT.fx



PS. your surfboard name should be surfboard159, surfboard180, etc. This example uses a testing Surfboard which has number -1

7. Control Actuactors using MQTT.fx



Try this:
***/speaker?1**

8. Control Your Surfboard via REST

- http://localhost:8888/things/data/surfboardX/[sensor]

  http://localhost:8888/things/data/surfboard-1/temp

- http://localhost:8888/things/surfboardX/[actuactor]/[value]

  http://localhost:8888/things/surfboard-1/speaker/1

  http://localhost:8888/things/surfboard-1/blue/100

NOW IS ABOUT HAVING FUN!!!!

LIVE DEMO

Summary

- IoT Surfing API provides access to your board via Serial Communication;
- IoT Surfing Service is a Serial - TCP/IP Gateway
- MQTT + REST = NICE COMBO!
- Now you can control your Surfboard using ANY platform!

IoT Surfboard + Arduino + Java = LOVE!