ORACLE®

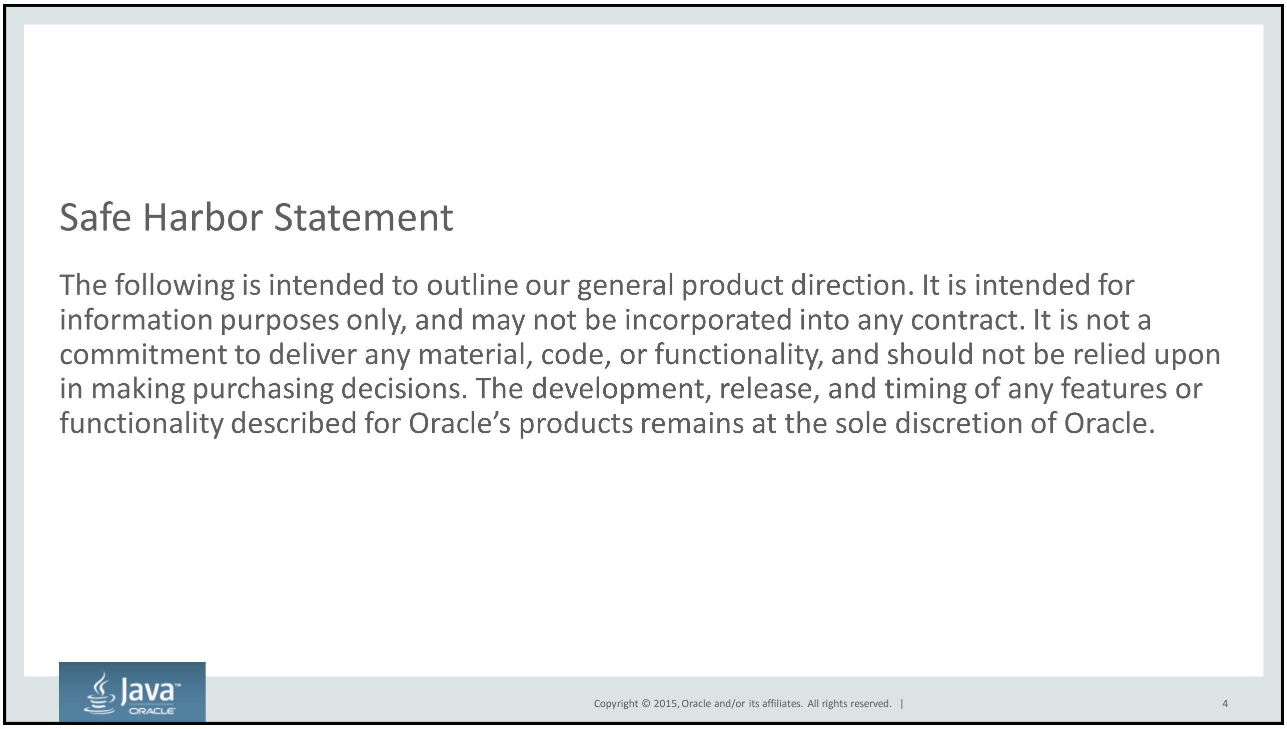
Java™

ORACLE®

Working with the GrovePi + RaspberryPi + Java8

Java and the Internet of Things

Eduardo Moranchel
Java Curriculum Developer

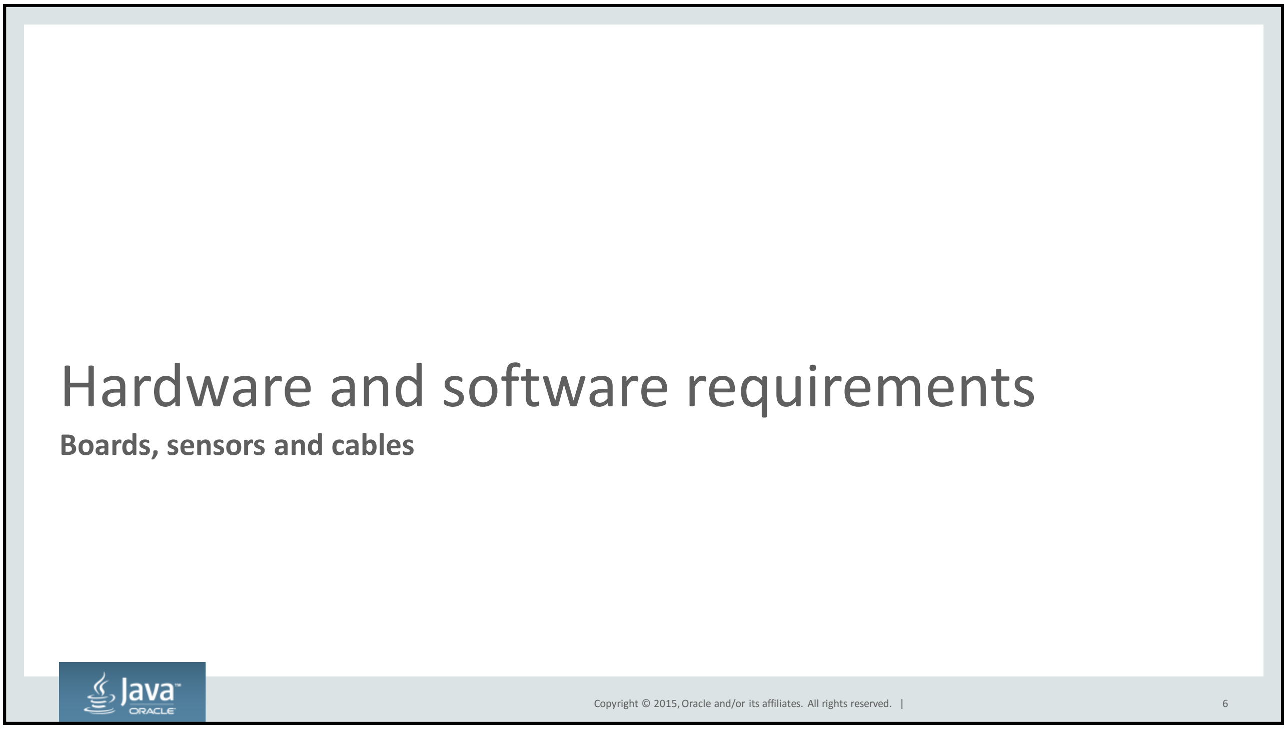


Safe Harbor Statement

The following is intended to outline our general product direction. It is intended for information purposes only, and may not be incorporated into any contract. It is not a commitment to deliver any material, code, or functionality, and should not be relied upon in making purchasing decisions. The development, release, and timing of any features or functionality described for Oracle's products remains at the sole discretion of Oracle.

Program Agenda

1. Hardware and software requirements
2. Setup
3. The GrovePi Java libary
4. Build your own project



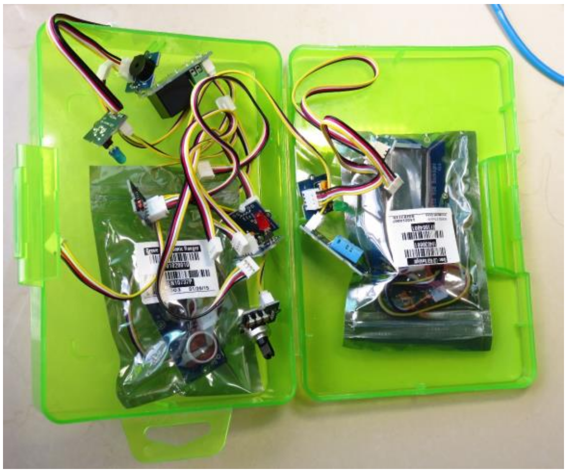
Hardware and software requirements

Boards, sensors and cables

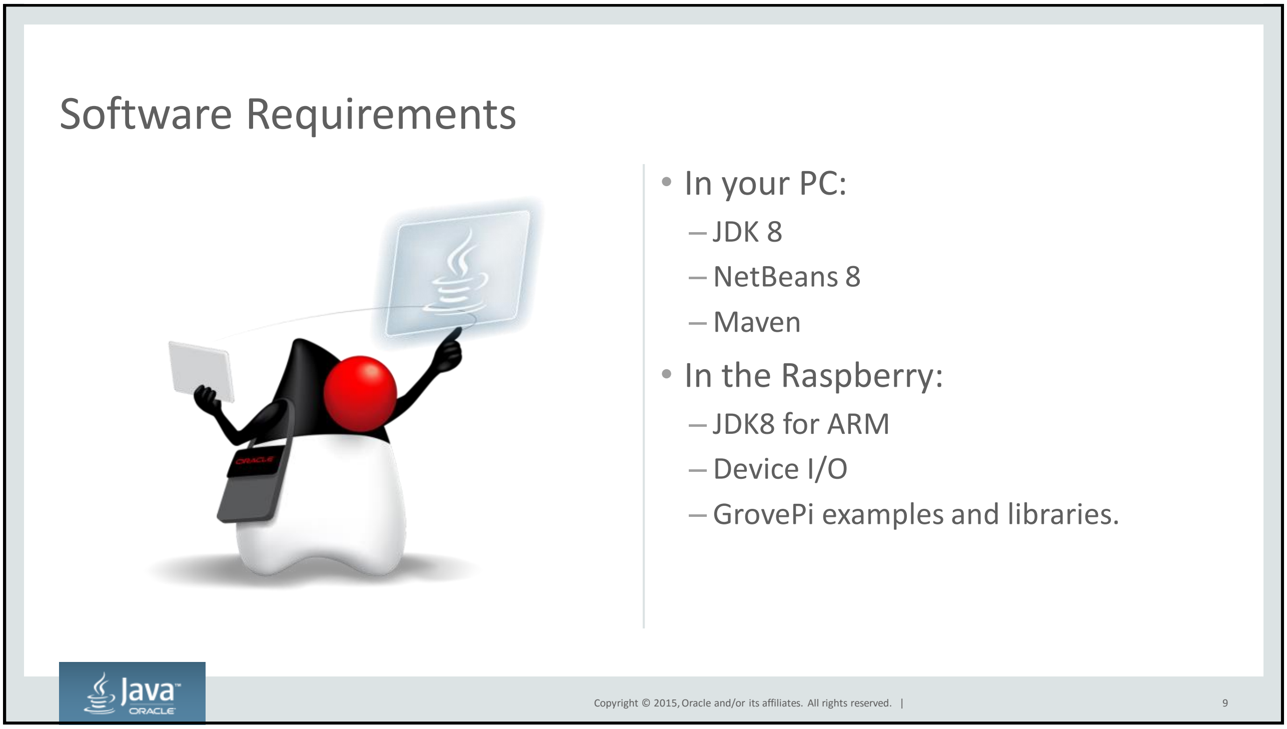
Hardware Requirements

- GrovePi Starter Kit
- Raspberry Pi B (or later)
- SDCard (or MicroSD) 2gb or more.
- Ethernet cable
  - USB Ethernet adapter for some PCs
- MicroUSB power cable

Inside the GrovePi Starter Kit

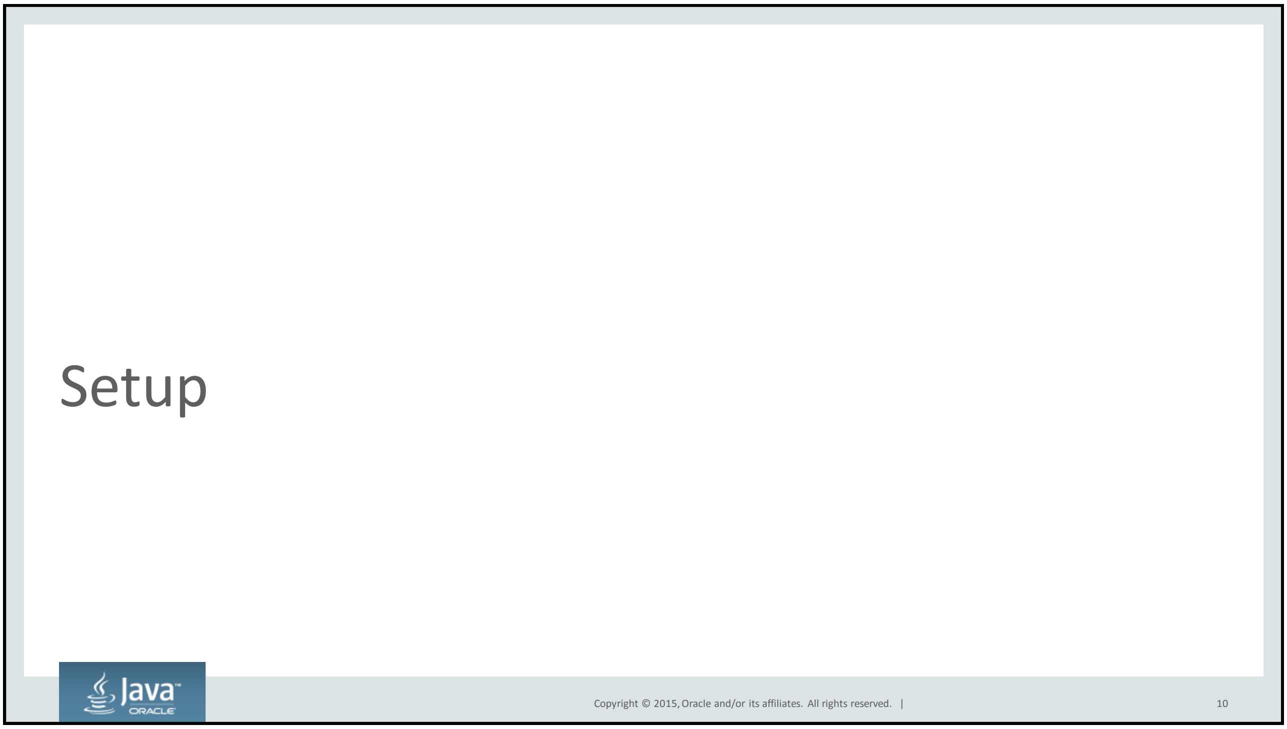


- GrovePi + board
- Temperature & humidity
- Relay
- Ultrasonic Ranger
- LCD RGB Backlight
- Buzzer
- Sound Sensor
- Light sensor
- Button
- Potenciometer
- Leds



Software Requirements

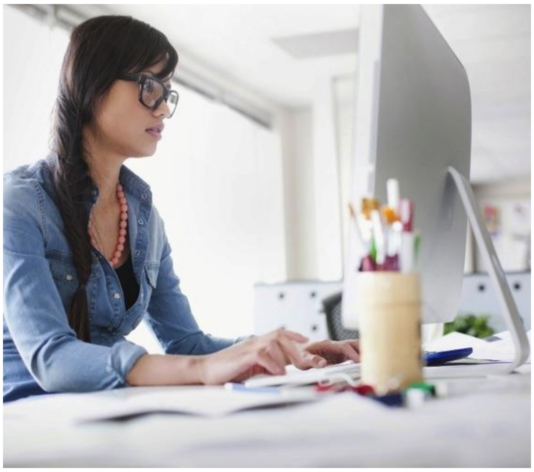
- In your PC:
  - JDK 8
  - NetBeans 8
  - Maven
- In the Raspberry:
  - JDK8 for ARM
  - Device I/O
  - GrovePi examples and libraries.



Setup

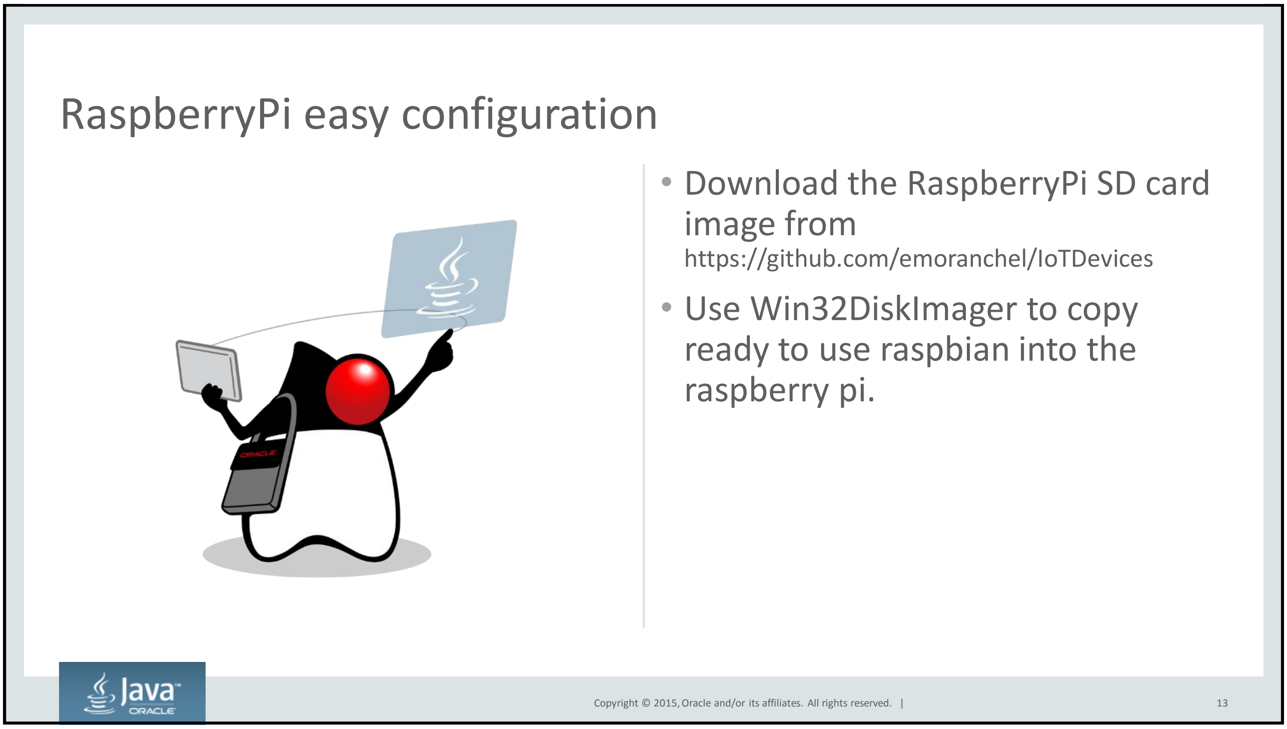
PC Setup

- Install JDK 8
- Install Maven
- Install NetBeans 8



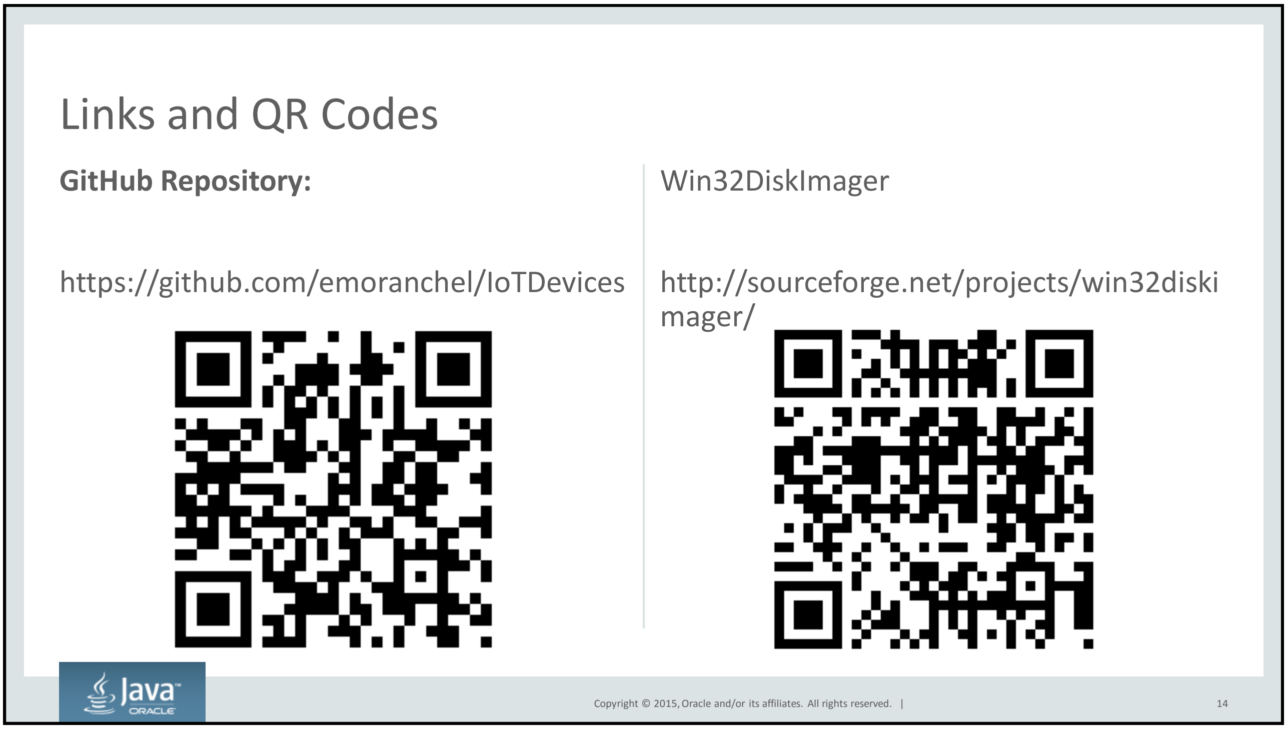
RaspberryPi Configuration

- Install Raspbian in the SD
- Connect pi to screen, keyboard, mouse and setup static IP
  - Install DHCP (optional)
- Install JDK8 for ARM
- Enable i2c, install i2c utils.
- Install the DeviceIO library
- Install GrovePi examples



RaspberryPi easy configuration

- Download the RaspberryPi SD card image from https://github.com/emoranchel/IoTDevices
- Use Win32DiskImager to copy ready to use raspbian into the raspberry pi.



Links and QR Codes

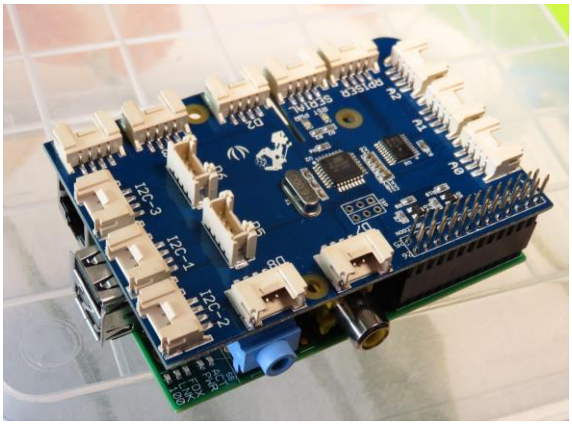
GitHub Repository:

https://github.com/emoranchel/IoTDevices

Win32DiskImager

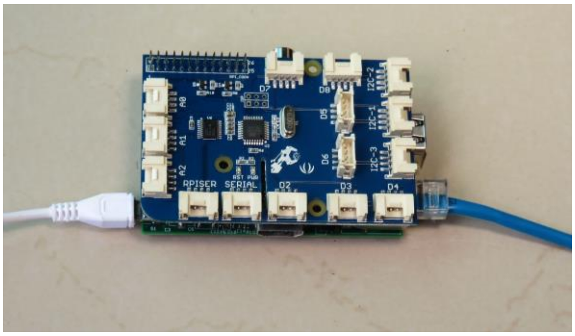
http://sourceforge.net/projects/win32diskimager/

Connect GrovePi



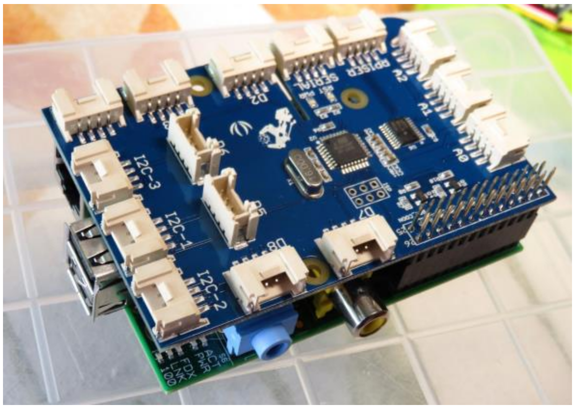
Connect cables

- Ethernet
- USB Cable



Putty (ssh) to 192.168.127.1

Username: pi
Password: raspberry



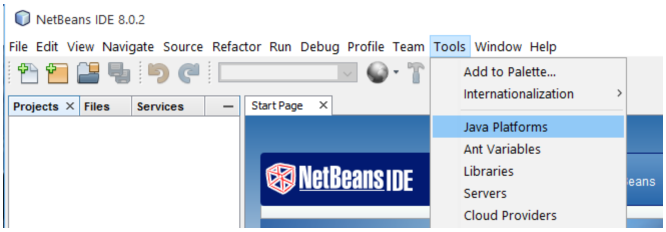
The Pi has already installed:

- Java 8 with Device IO
- I2C tools
- DHCP server and static IP
- Automatic IP in Wireless
- Security policy for DIO
- dio.jar

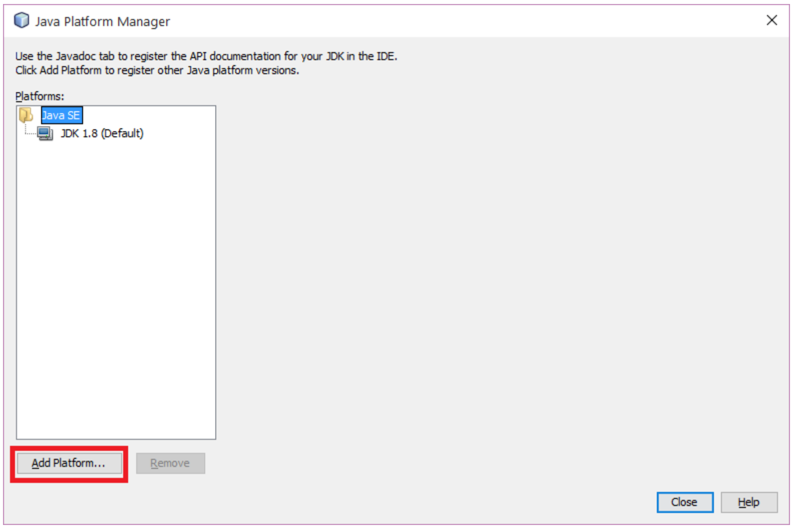
* The raspbian distribution does not have GUI components (X server)

NetBeans Configuration

- Setup Java Platforms



NetBeans Configuration



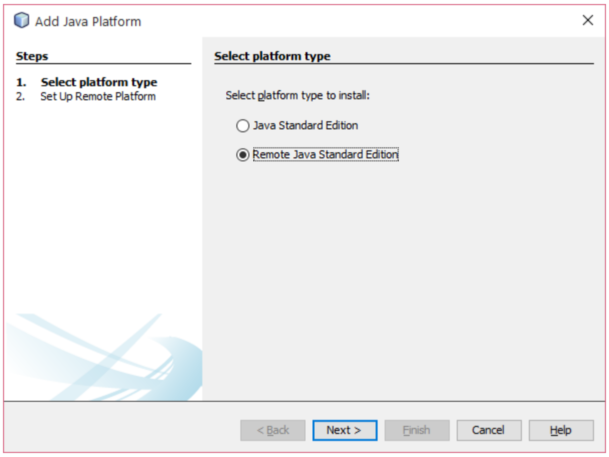
Java Platform Manager

Use the Javadoc tab to register the API documentation for your JDK in the IDE.
Click Add Platform to register other Java platform versions.

Platforms:

- Java SE
  - JDK 1.8 (Default)

NetBeans Configuration



Add Java Platform

Steps

1. **Select platform type**
2. Set Up Remote Platform

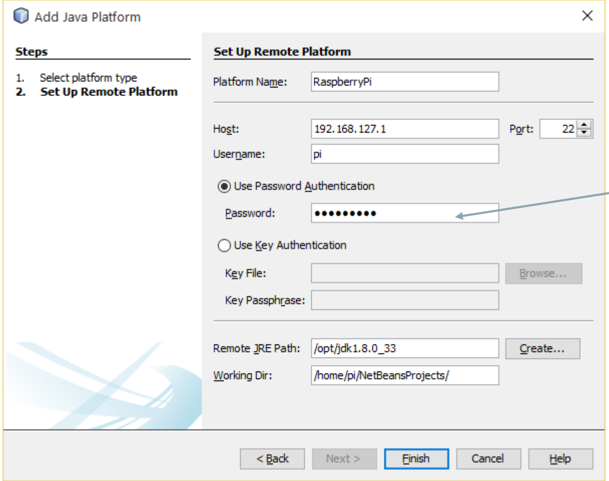
Select platform type

Select platform type to install:

○ Java Standard Edition

● Remote Java Standard Edition

NetBeans Configuration



Add Java Platform

Steps

1. Select platform type
2. **Set Up Remote Platform**

Set Up Remote Platform

Platform Name: RaspberryPi

Host: 192.168.127.1 Port: 22

Username: pi

● Use Password Authentication

Password: •••••••••

○ Use Key Authentication

Key File:

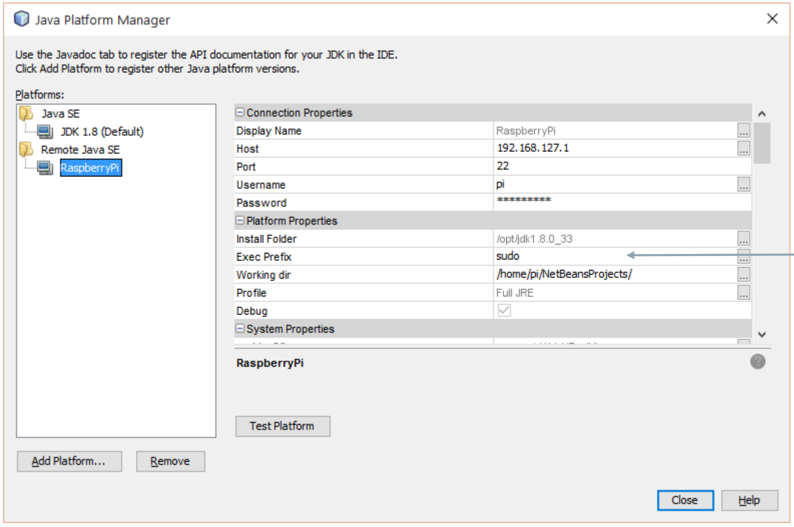
Key Passphrase:

Remote JRE Path: /opt/jdk1.8.0_33

Working Dir: /home/pi/NetBeansProjects/

raspberry

NetBeans Configuration



Java Platform Manager

Use the Javadoc tab to register the API documentation for your JDK in the IDE.
Click Add Platform to register other Java platform versions.

Platforms:

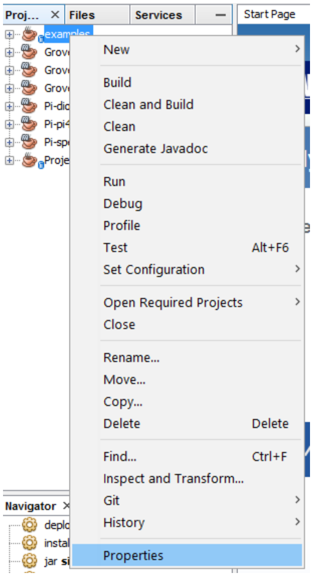
- Java SE
  - JDK 1.8 (Default)
- Remote Java SE
  - RaspberryPi

⊟ Connection Properties	
Display Name	RaspberryPi
Host	192.168.127.1
Port	22
Username	pi
Password	*********
⊟ Platform Properties	
Install Folder	/opt/jdk1.8.0_33
Exec Prefix	sudo
Working dir	/home/pi/NetBeansProjects/
Profile	Full JRE
Debug	☑
⊟ System Properties	

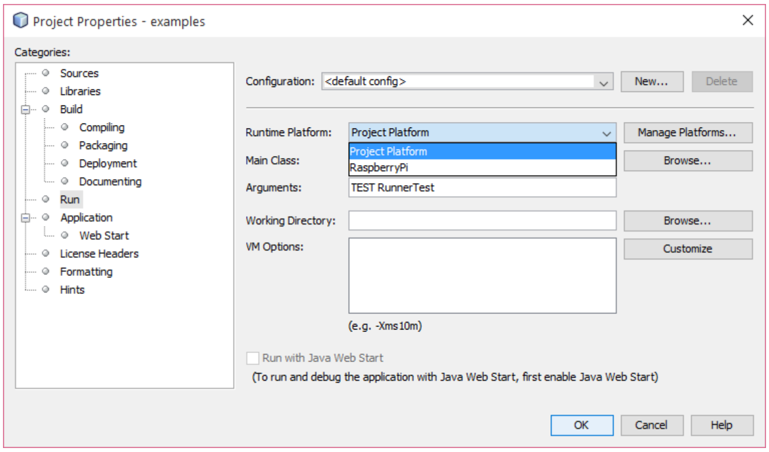
RaspberryPi

sudo

Deploy Project to the raspberry



Deploy Project to the raspberry



Configuration: <default config>

Runtime Platform: Project Platform

Project Platform
RaspberryPi

Main Class:

Arguments: TEST RunnerTest

Working Directory:

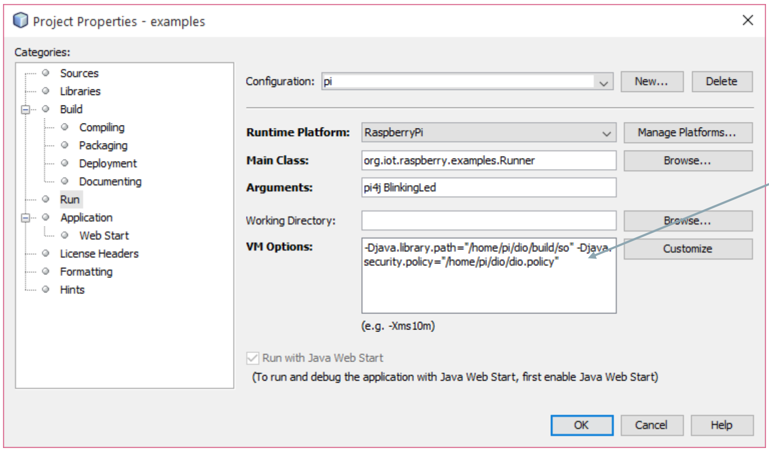
VM Options:

(e.g. -Xms10m)

□ Run with Java Web Start

(To run and debug the application with Java Web Start, first enable Java Web Start)

Deploy Project to the raspberry



Project Properties - examples

Categories:

- Sources
- Libraries
- Build
  - Compiling
  - Packaging
  - Deployment
  - Documenting
- Run
- Application
  - Web Start
- License Headers
- Formatting
- Hints

Configuration: pi

Runtime Platform: RaspberryPi

Main Class: org.iot.raspberry.examples.Runner

Arguments: pi4j BlinkingLed

Working Directory:

VM Options: -Djava.library.path="/home/pi/dio/build/so" -Djava.security.policy="/home/pi/dio/dio.policy"

(e.g. -Xms10m)

☑ Run with Java Web Start

(To run and debug the application with Java Web Start, first enable Java Web Start)

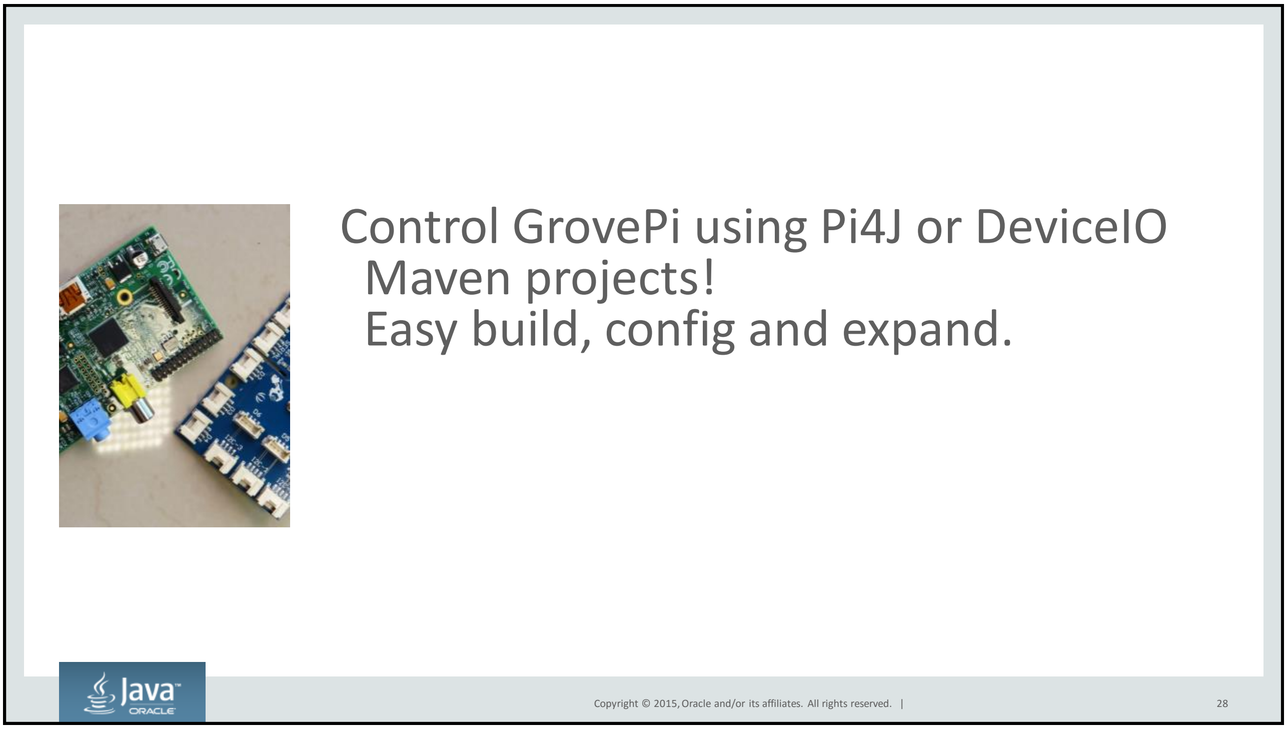
-Djava.library.path=
"/home/pi/dio/build/so"
-Djava.security.policy=
"/home/pi/dio/dio.policy"

The GrovePi Java library

How to use the GrovePi board and sensors using Java

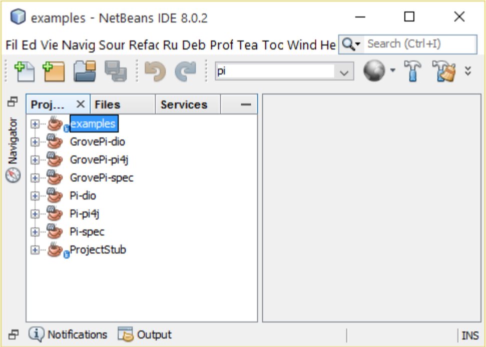
Download

https://github.com/emoranchel/IoTDevices



Control GrovePi using Pi4J or DeviceIO
Maven projects!
Easy build, config and expand.

Projects



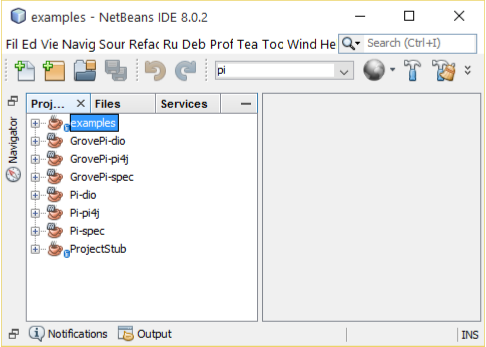
examples - NetBeans IDE 8.0.2

examples
GrovePi-dio
GrovePi-pi4j
GrovePi-spec
Pi-dio
Pi-pi4j
Pi-spec
ProjectStub

Pi-*: Projects to control the raspberry pi GPIO pins

GrovePi-*: Projects to control the grovepi board and connected devices.

Projects



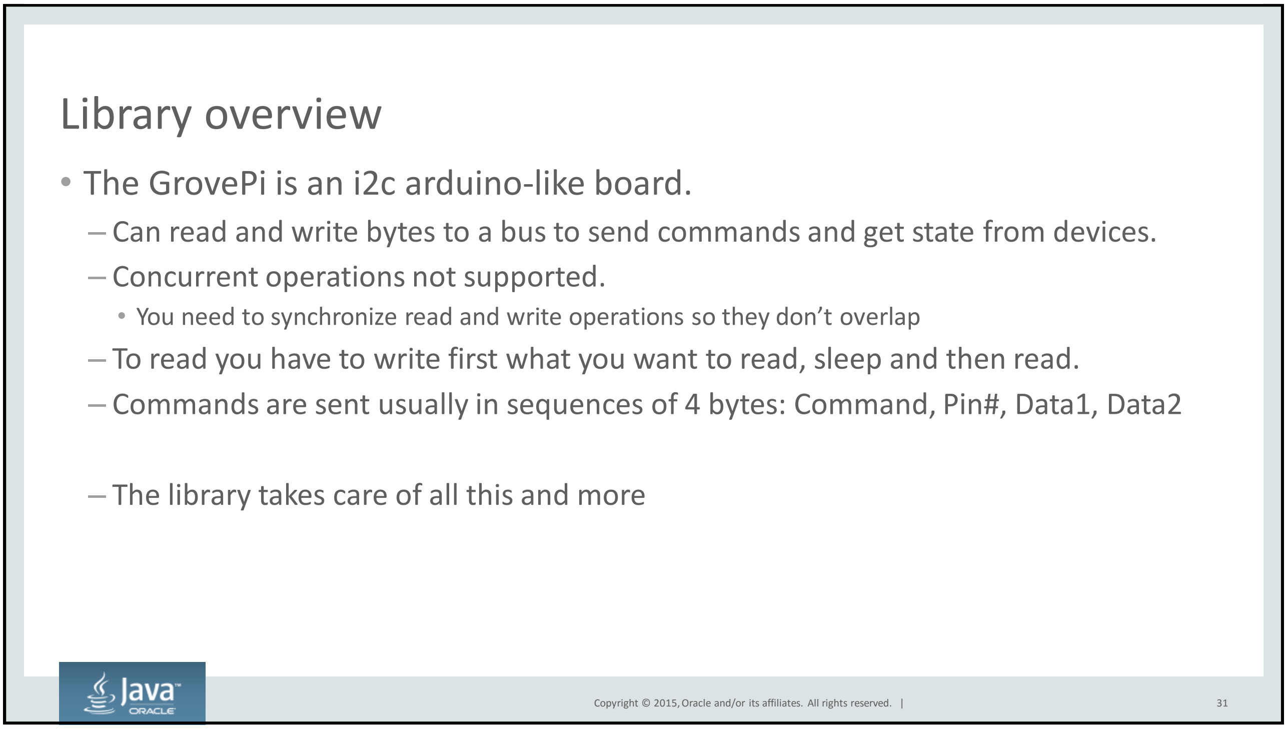
examples - NetBeans IDE 8.0.2

examples
GrovePi-dio
GrovePi-pi4j
GrovePi-spec
Pi-dio
Pi-pi4j
Pi-spec
ProjectStub

*-spec: the specification libraries, defines operations on devices.

*-pi4j: Implementations using the pi4j library

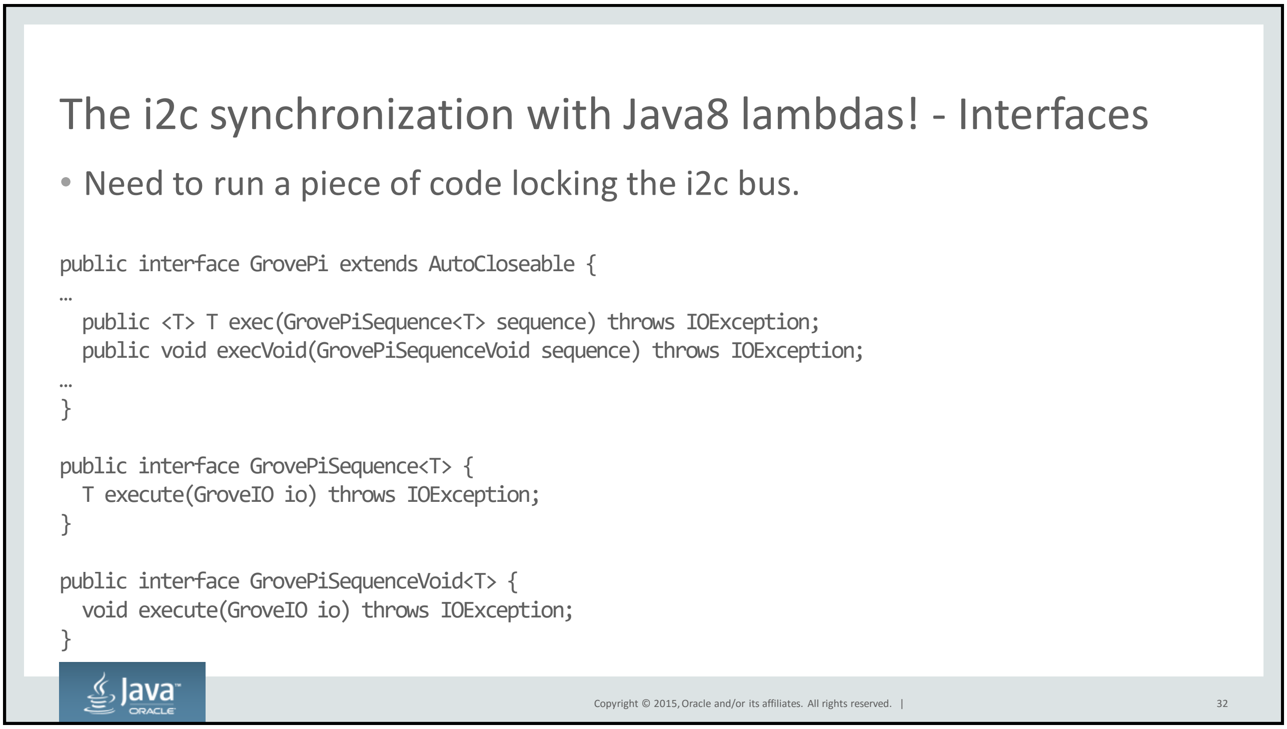
*-dio: Implementation using the DeviceIO library



Library overview

- The GrovePi is an i2c arduino-like board.
  - Can read and write bytes to a bus to send commands and get state from devices.
  - Concurrent operations not supported.
    - You need to synchronize read and write operations so they don't overlap
  - To read you have to write first what you want to read, sleep and then read.
  - Commands are sent usually in sequences of 4 bytes: Command, Pin#, Data1, Data2

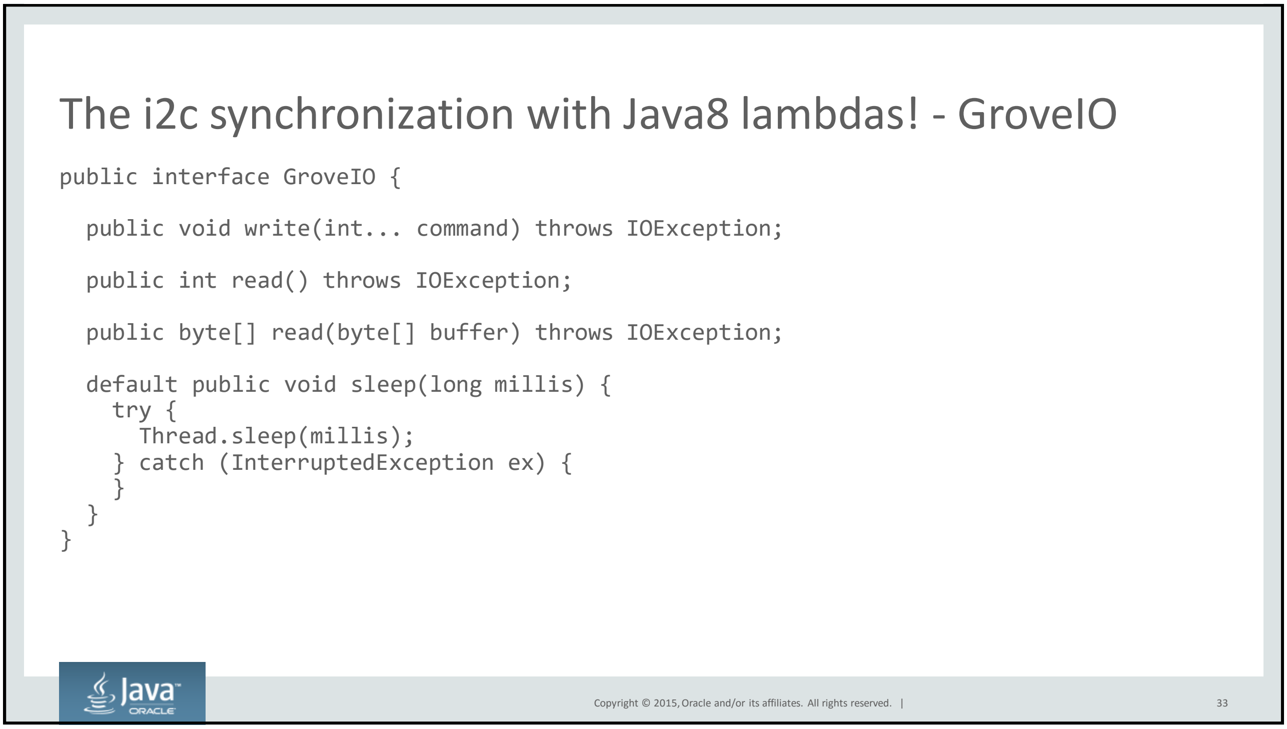
  - The library takes care of all this and more



The i2c synchronization with Java8 lambdas! - Interfaces

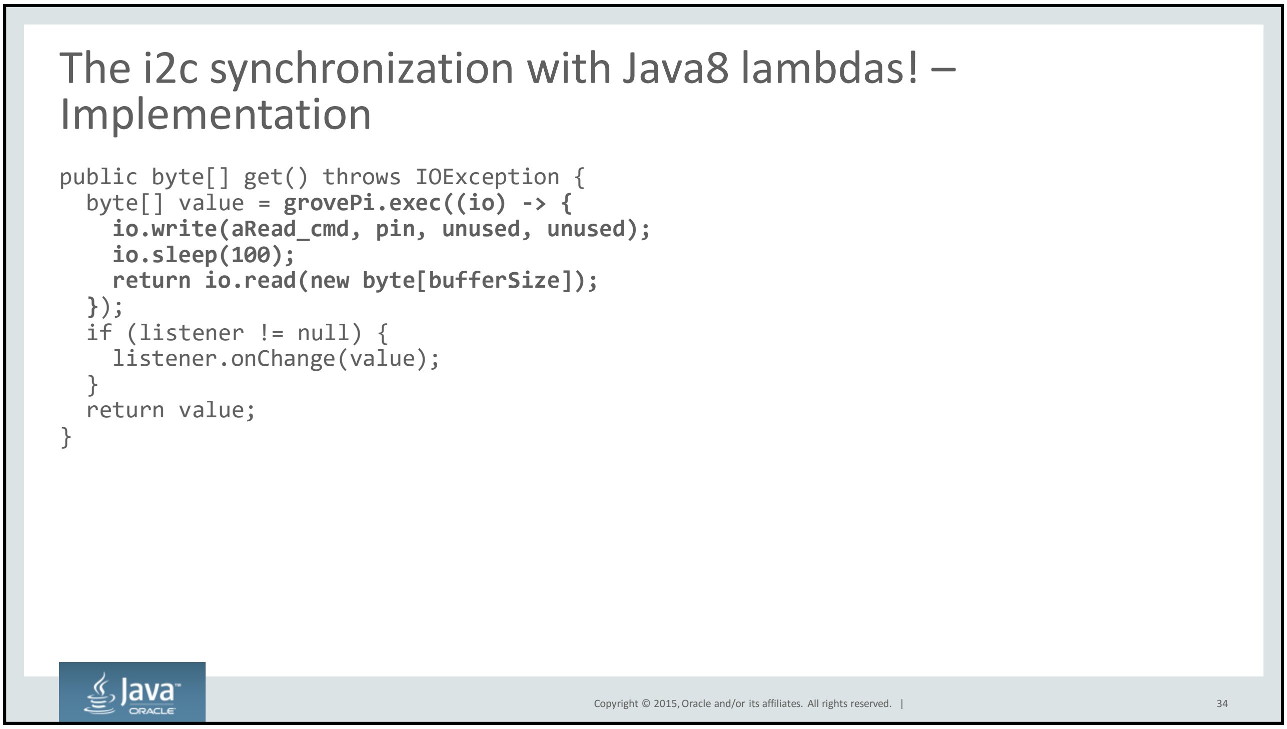
- Need to run a piece of code locking the i2c bus.

```
public interface GrovePi extends AutoCloseable {
…
  public <T> T exec(GrovePiSequence<T> sequence) throws IOException;
  public void execVoid(GrovePiSequenceVoid sequence) throws IOException;
…
}

public interface GrovePiSequence<T> {
  T execute(GroveIO io) throws IOException;
}

public interface GrovePiSequenceVoid<T> {
  void execute(GroveIO io) throws IOException;
}
```



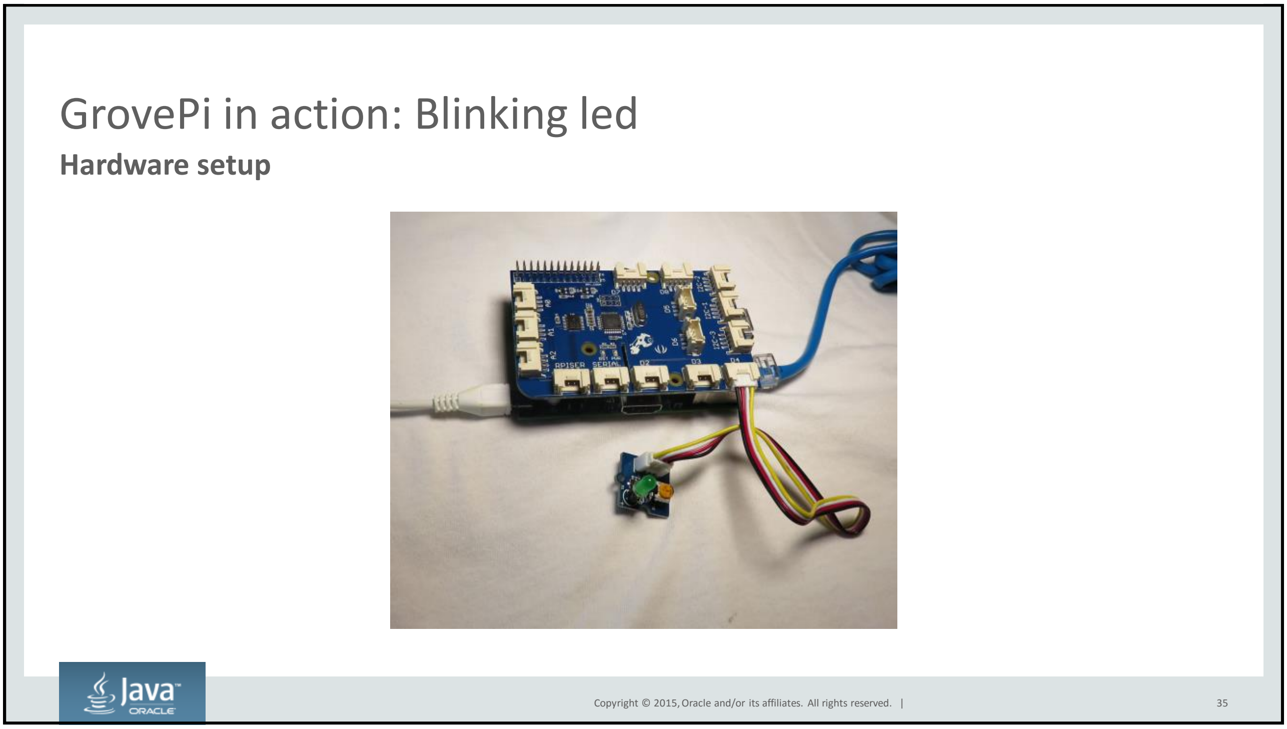
The i2c synchronization with Java8 lambdas! - GroveIO

```
public interface GroveIO {

  public void write(int... command) throws IOException;

  public int read() throws IOException;

  public byte[] read(byte[] buffer) throws IOException;

  default public void sleep(long millis) {
    try {
      Thread.sleep(millis);
    } catch (InterruptedException ex) {
    }
  }
}
```



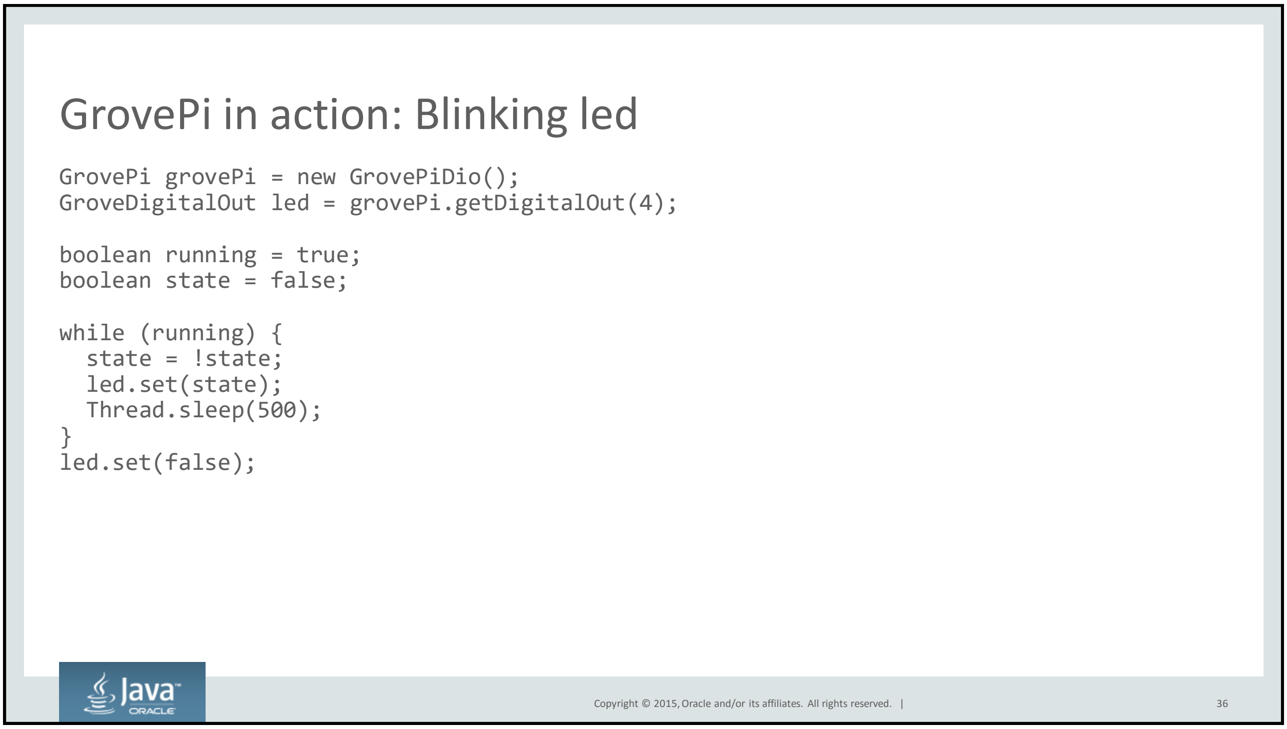
The i2c synchronization with Java8 lambdas! – Implementation

```
public byte[] get() throws IOException {
  byte[] value = grovePi.exec((io) -> {
    io.write(aRead_cmd, pin, unused, unused);
    io.sleep(100);
    return io.read(new byte[bufferSize]);
  });
  if (listener != null) {
    listener.onChange(value);
  }
  return value;
}
```



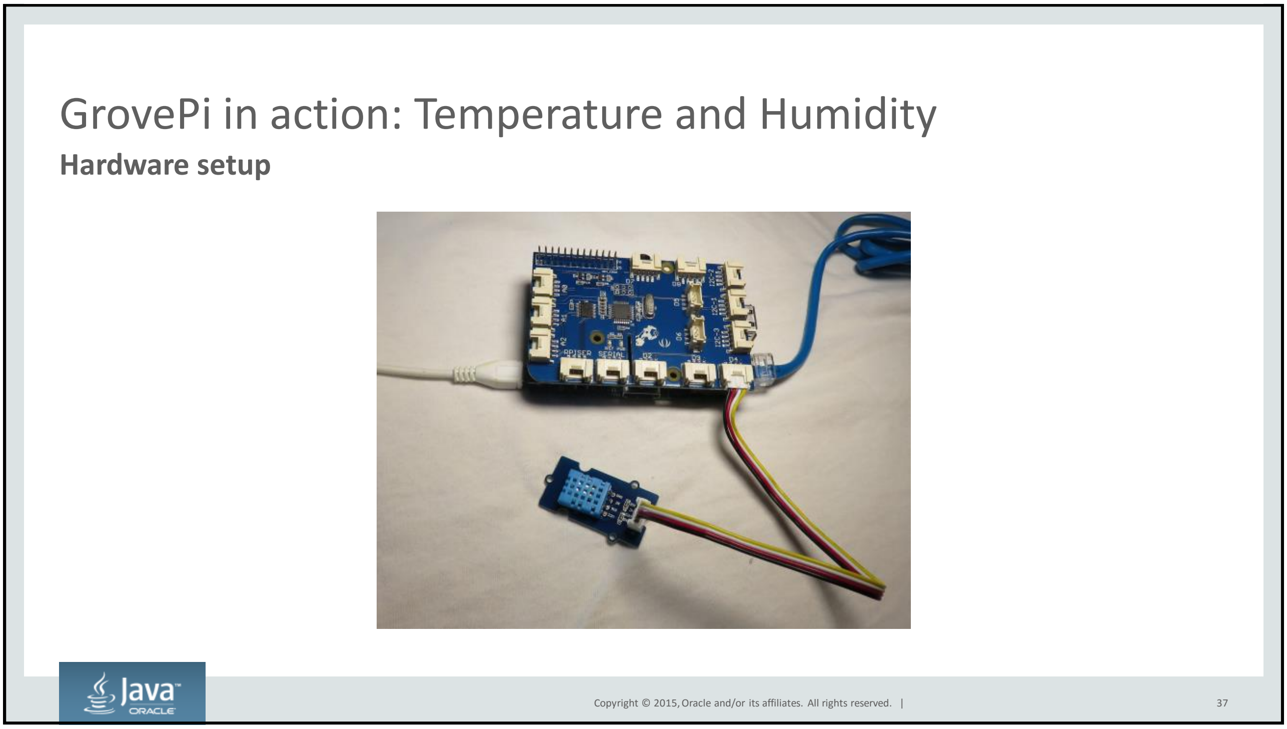
GrovePi in action: Blinking led

Hardware setup



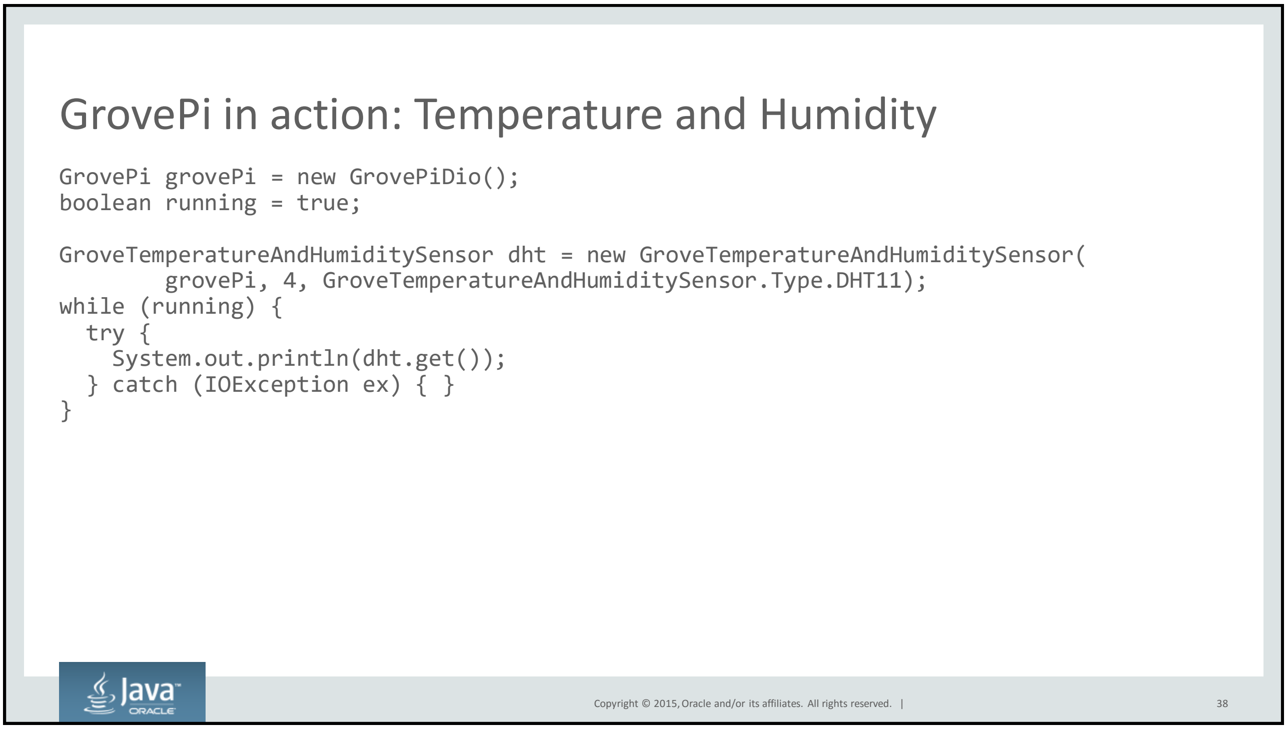
GrovePi in action: Blinking led

```
GrovePi grovePi = new GrovePiDio();
GroveDigitalOut led = grovePi.getDigitalOut(4);

boolean running = true;
boolean state = false;

while (running) {
  state = !state;
  led.set(state);
  Thread.sleep(500);
}
led.set(false);
```



GrovePi in action: Temperature and Humidity

Hardware setup

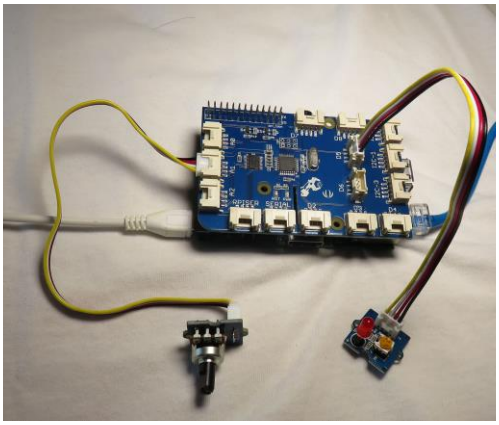


GrovePi in action: Temperature and Humidity

```
GrovePi grovePi = new GrovePiDio();
boolean running = true;

GroveTemperatureAndHumiditySensor dht = new GroveTemperatureAndHumiditySensor(
        grovePi, 4, GroveTemperatureAndHumiditySensor.Type.DHT11);
while (running) {
  try {
    System.out.println(dht.get());
  } catch (IOException ex) { }
}
```

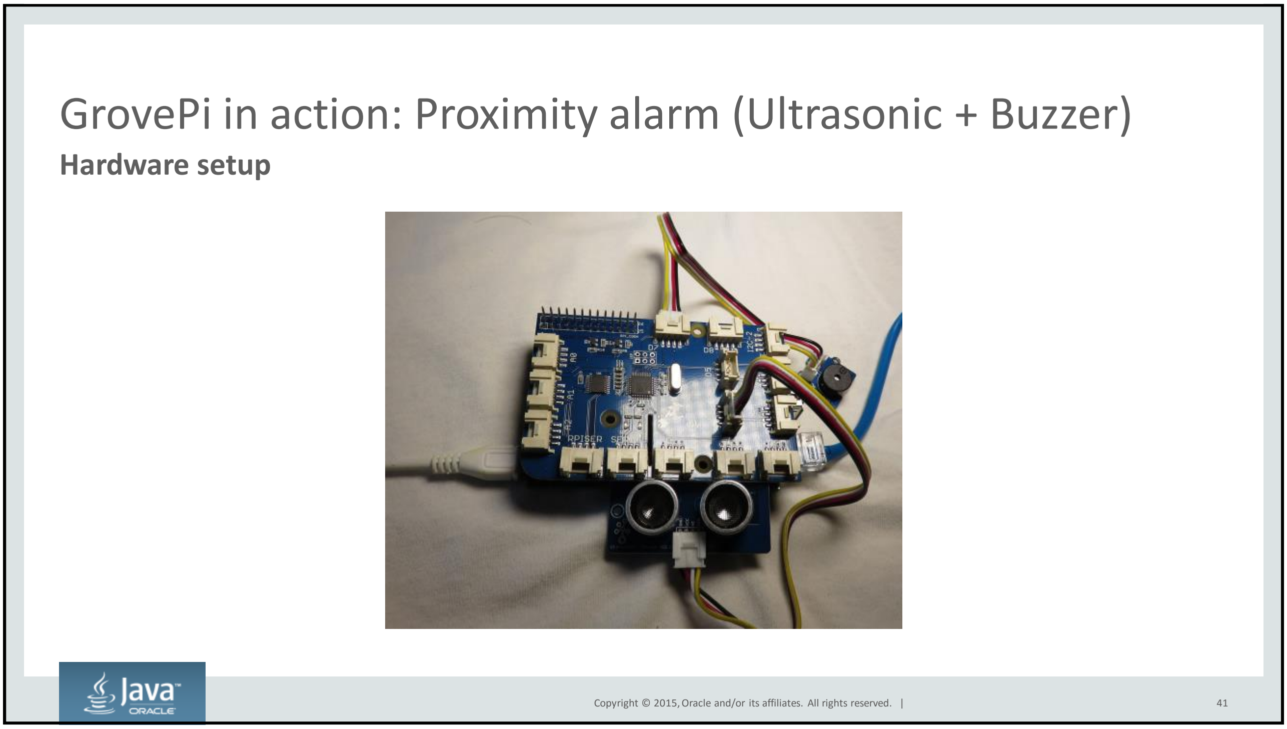
GrovePi in action: Dim a led (Rotary + Led)

Hardware setup



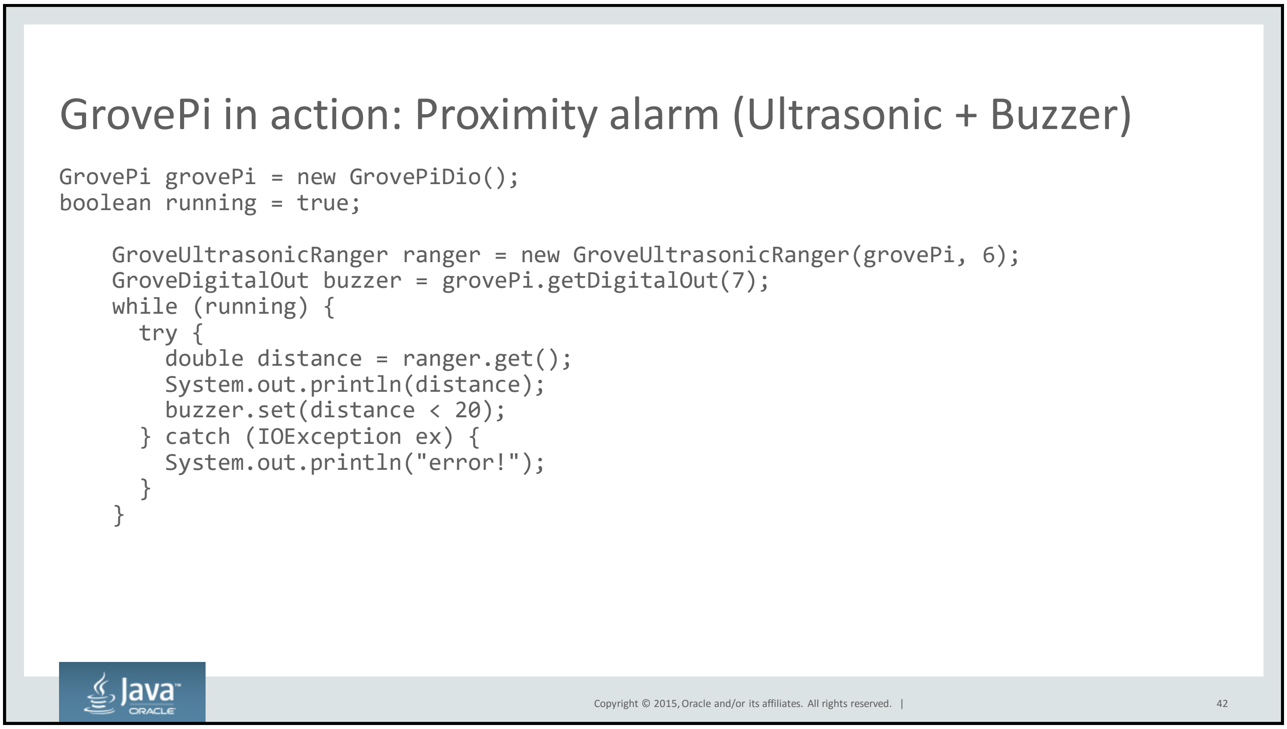
GrovePi in action: Dim a led (Rotary + Led)

```
GrovePi grovePi = new GrovePiDio();
boolean running = true;

 GroveRotarySensor rotarySensor = new GroveRotarySensor(grovePi, 1);
    GroveLed blueLed = new GroveLed(grovePi, 5);
    while (running) {
      try {
        GroveRotaryValue value = rotarySensor.get();
        int brightness = (int) (value.getFactor() * GroveLed.MAX_BRIGTHNESS);
        blueLed.set(brightness);
      } catch (IOException ex) {
        System.out.println("Error");
      }
    }
    blueLed.set(false);
```



GrovePi in action: Proximity alarm (Ultrasonic + Buzzer)

Hardware setup



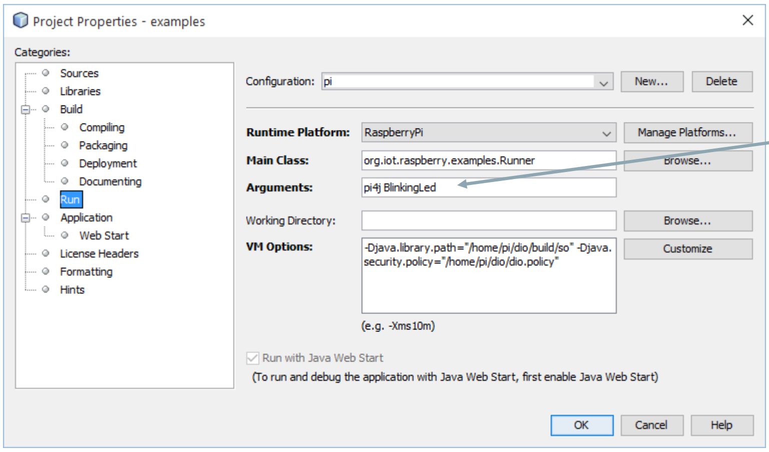
GrovePi in action: Proximity alarm (Ultrasonic + Buzzer)

```
GrovePi grovePi = new GrovePiDio();
boolean running = true;

    GroveUltrasonicRanger ranger = new GroveUltrasonicRanger(grovePi, 6);
    GroveDigitalOut buzzer = grovePi.getDigitalOut(7);
    while (running) {
      try {
        double distance = ranger.get();
        System.out.println(distance);
        buzzer.set(distance < 20);
      } catch (IOException ex) {
        System.out.println("error!");
      }
    }
```

Running the examples

- Download the IoTDevices projects
- Run the build.bat batch to build all the projects and deviceIO jar (maven)
- Open all the IoTDevices projects in NetBeans
- Open the examples Project properties
- In the run tab the arguments define
  - The implementation to use: [pi4j | dio]
  - The name of the class to run
- Run by right-clicking the Project and choosing run from the menu
- Stop by running the example again

Running the examples



Configuration: pi

Runtime Platform: RaspberryPi

Main Class: org.iot.raspberry.examples.Runner

Arguments: pi4j BlinkingLed

Working Directory:

VM Options: -Djava.library.path="/home/pi/dio/build/so" -Djava.security.policy="/home/pi/dio/dio.policy"

(e.g. -Xms10m)

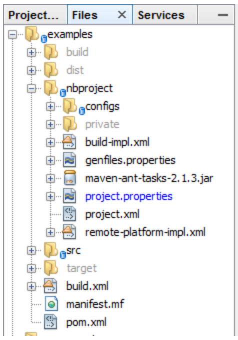
☑ Run with Java Web Start

(To run and debug the application with Java Web Start, first enable Java Web Start)

Runs the BlinkingLed class using the pi4j implementation

How does it work?

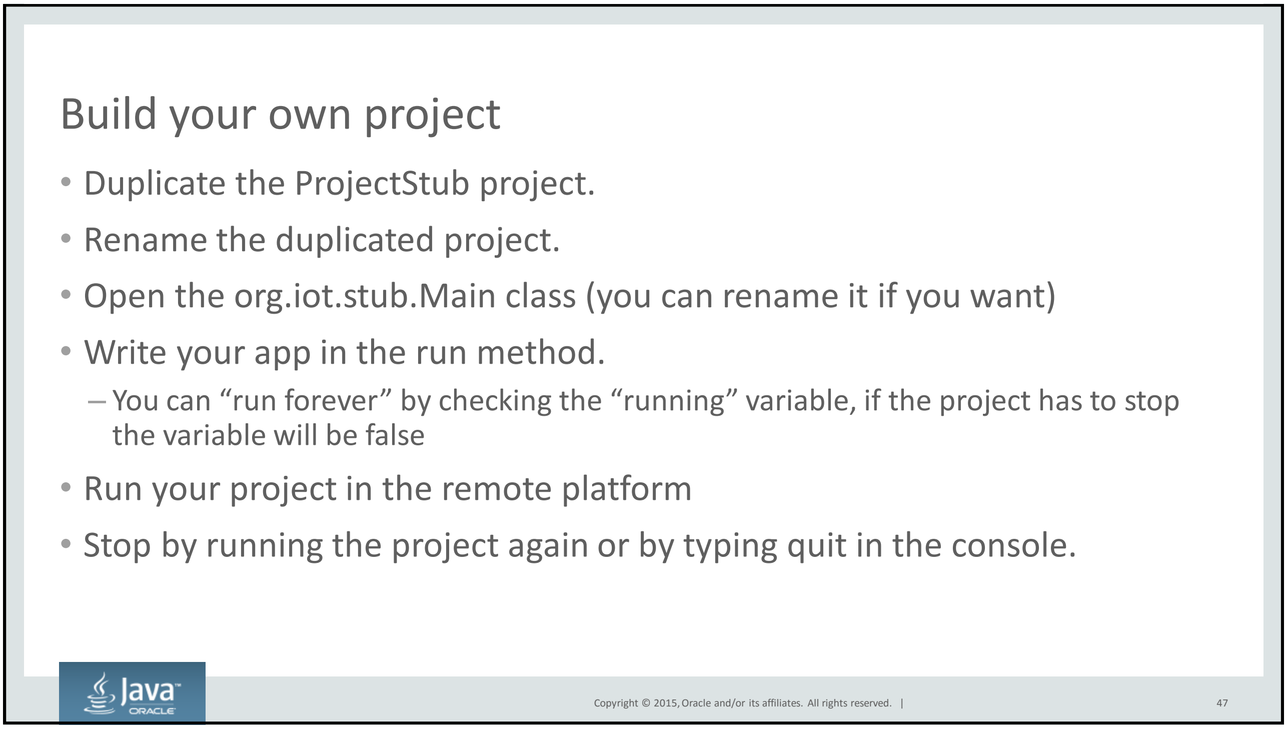
The examples project is a NetBeans project that uses a plugin to download and use Maven dependencies. Normally it is impossible to run a maven project in a remote platform.



```
examples
    build
    dist
    nbproject
        configs
        private
        build-impl.xml
        genfiles.properties
        maven-ant-tasks-2.1.3.jar
        project.properties
        project.xml
        remote-platform-impl.xml
    src
    target
    build.xml
    manifest.mf
    pom.xml
```

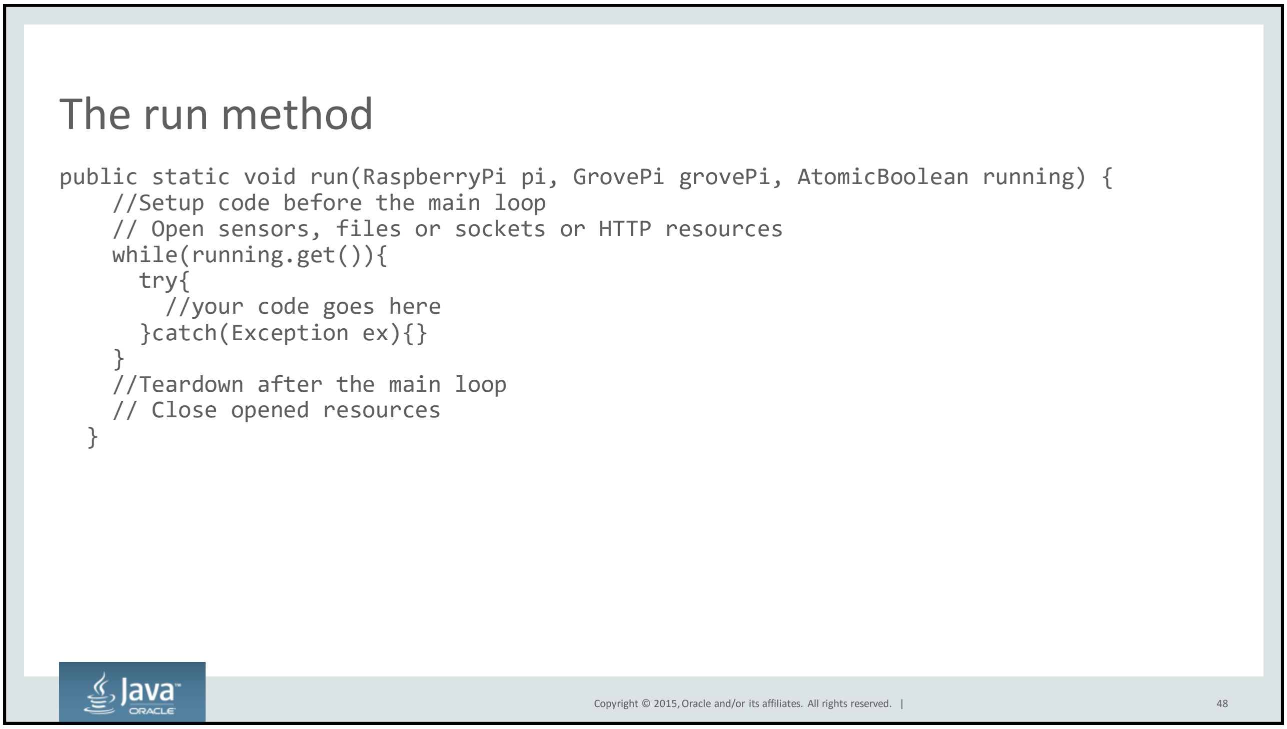
Build your own projects

Easy and fast IoT prototyping using the GrovePi and Java



Build your own project

- Duplicate the ProjectStub project.
- Rename the duplicated project.
- Open the org.iot.stub.Main class (you can rename it if you want)
- Write your app in the run method.
  - You can “run forever” by checking the “running” variable, if the project has to stop the variable will be false
- Run your project in the remote platform
- Stop by running the project again or by typing quit in the console.



The run method

```
public static void run(RaspberryPi pi, GrovePi grovePi, AtomicBoolean running) {
    //Setup code before the main loop
    // Open sensors, files or sockets or HTTP resources
    while(running.get()){
      try{
        //your code goes here
      }catch(Exception ex){}
    }
    //Teardown after the main loop
    // Close opened resources
  }
```

CREATE THE FUTURE

Integrated Cloud

Applications & Platform Services

Java™
ORACLE®

ORACLE®