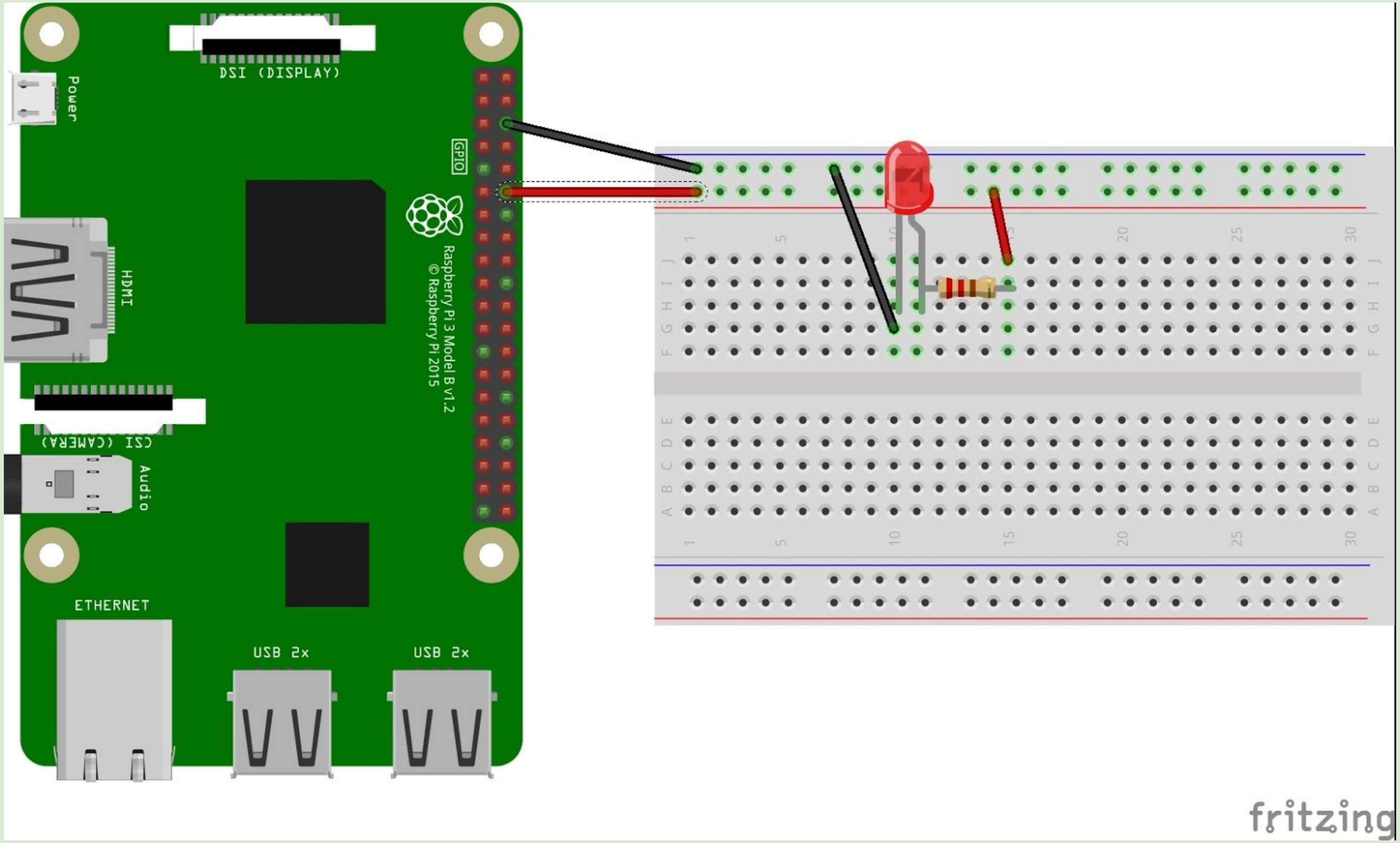
RaspberryPi3 :: Java :: Lec_JavaRasPi3_0140 :: Ch02010_LedOnOff

1. Fritzing Layout :



2. Usage :

2.1. 넷빈 자바 프로젝트에 Pi4J 라이브러리 추가합니다 :

Netbeans and pi4j importing[1] :

Prerequisite:

1.) Download and extract the Pi4J JAR files somewhere on your local system (http://pi4j.com/download.html)

2.) Download and install latest NetBeans IDE for Java Developers.

In NetBeans:

1.) File > New Project > Java > Java Application

2.) Enter a project name such as "Ch02010_LedOnOff"

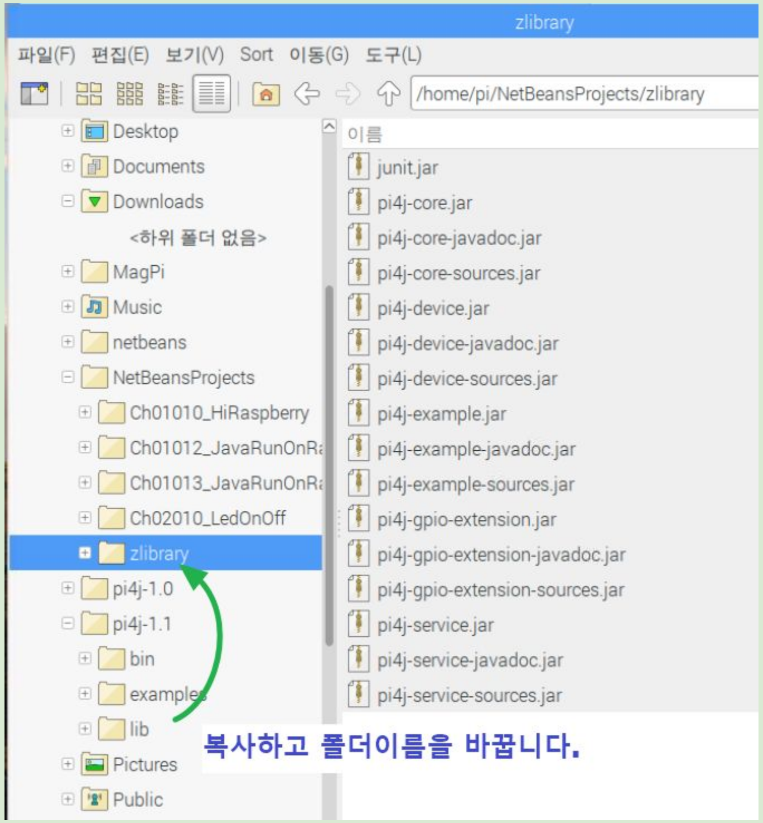
3.) Make sure the "Create Main Class" option is checked and use

[1] https://lb.raspberrypi.org/forums/viewtopic.php?t=120072 Netbeans and pi4j importing

"Ch02010_LedOnOff" for the class name.

4.) Click "Finish" to create project template.

5.) In the project's tree, right-click the "Libraries" folder and select "Add Library"



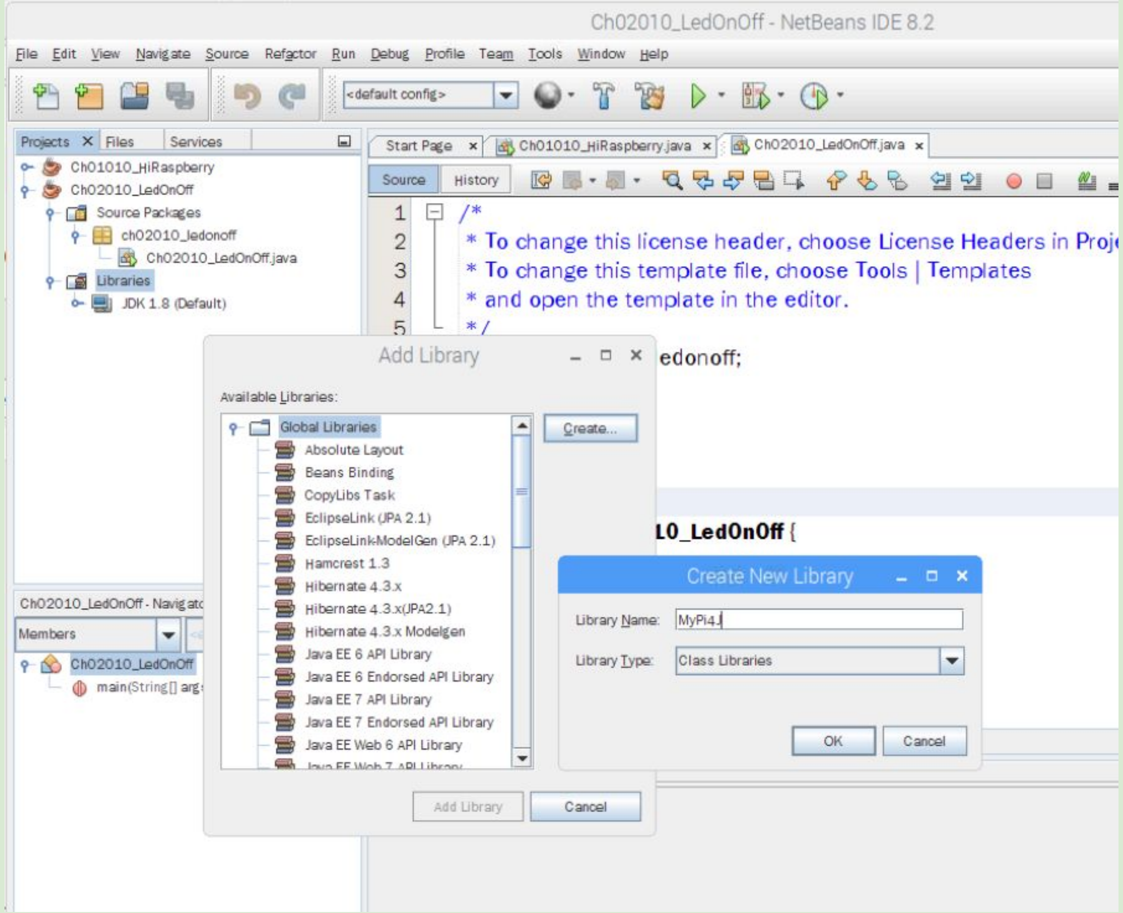
/home/pi/NetBeansProjects/zlibrary

```
junit.jar
pi4j-core.jar
pi4j-core-javadoc.jar
pi4j-core-sources.jar
pi4j-device.jar
pi4j-device-javadoc.jar
pi4j-device-sources.jar
pi4j-example.jar
pi4j-example-javadoc.jar
pi4j-example-sources.jar
pi4j-gpio-extension.jar
pi4j-gpio-extension-javadoc.jar
pi4j-gpio-extension-sources.jar
pi4j-service.jar
pi4j-service-javadoc.jar
pi4j-service-sources.jar
```

복사하고 폴더이름을 바꿉니다.

RaspberryPi3 :: Java :: Lec_JavaRasPi3_0140 :: Ch02010_LedOnOff

6.) "Create" a new library using Library Name: "MyPi4J"



```
/*
 * To change this license header, choose License Headers in Proj
 * To change this template file, choose Tools | Templates
 * and open the template in the editor.
 */
```

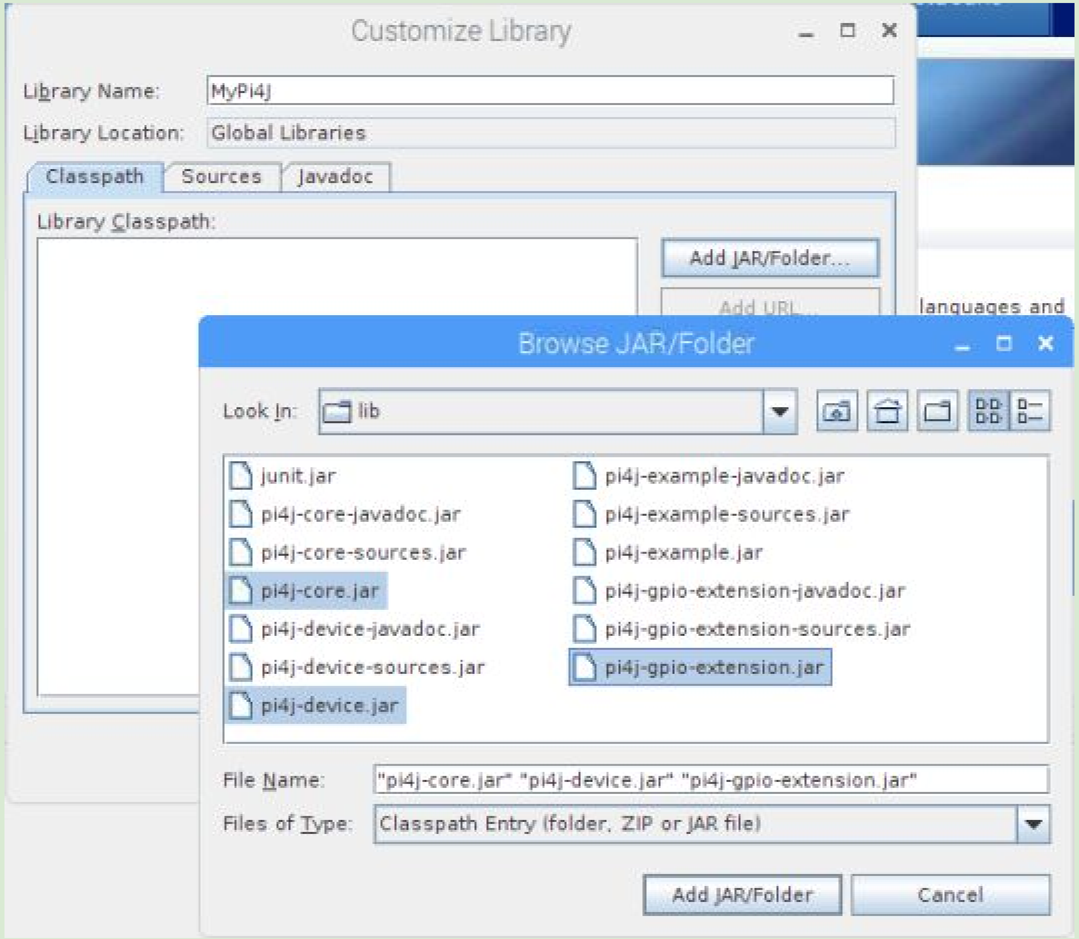
Create New Library

Library Name: MyPi4J

Library Type: Class Libraries

RaspberryPi3 :: Java :: Lec_JavaRasPi3_0140 :: Ch02010_LedOnOff

7.) Make sure the "Classpath" option is selected and then "Add JAR/Folder" - select the "pi4j-core.jar" file from your local file system (assuming you already have downloaded Pi4J on your computer)



Customize Library

Library Name: MyPi4J

Library Location: Global Libraries

Classpath | Sources | Javadoc

Library Classpath:

Browse JAR/Folder

Look In: lib

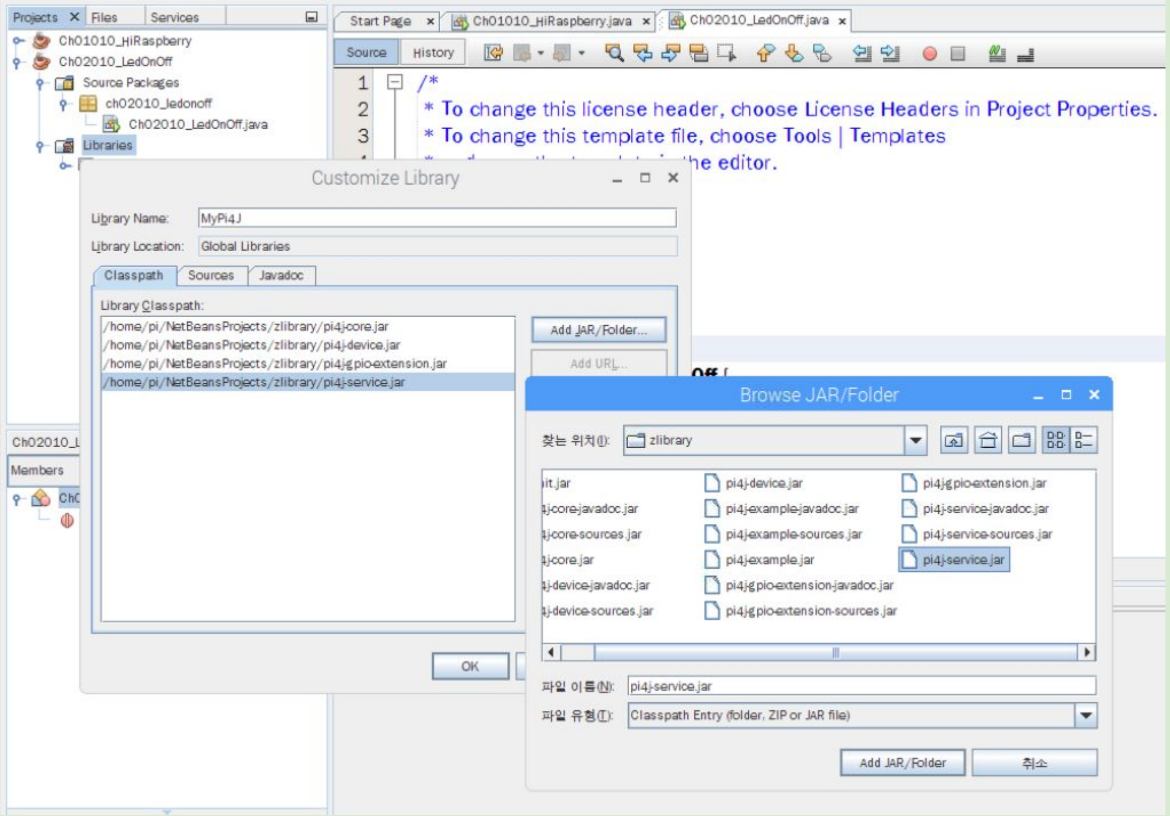
junit.jar
pi4j-core-javadoc.jar
pi4j-core-sources.jar
pi4j-core.jar
pi4j-device-javadoc.jar
pi4j-device-sources.jar
pi4j-device.jar
pi4j-example-javadoc.jar
pi4j-example-sources.jar
pi4j-example.jar
pi4j-gpio-extension-javadoc.jar
pi4j-gpio-extension-sources.jar
pi4j-gpio-extension.jar

File Name: "pi4j-core.jar" "pi4j-device.jar" "pi4j-gpio-extension.jar"

Files of Type: Classpath Entry (folder, ZIP or JAR file)

(pi4j-1.2 버전의 경우 선택할 JAR 파일이 3개입니다).

RaspberryPi3 :: Java :: Lec_JavaRasPi3_0140 :: Ch02010_LedOnOff



```
/*
 * To change this license header, choose License Headers in Project Properties.
 * To change this template file, choose Tools | Templates
 * and open the template in the editor.
```

Library Name: MyPi4J

Library Location: Global Libraries

Library Classpath:

/home/pi/NetBeansProjects/zlibrary/pi4j-core.jar
/home/pi/NetBeansProjects/zlibrary/pi4j-device.jar
/home/pi/NetBeansProjects/zlibrary/pi4j-gpio-extension.jar
/home/pi/NetBeansProjects/zlibrary/pi4j-service.jar

찾는 위치(I): zlibrary

it.jar
4j-core-javadoc.jar
4j-core-sources.jar
4j-core.jar
4j-device-javadoc.jar
4j-device-sources.jar
pi4j-device.jar
pi4j-example-javadoc.jar
pi4j-example-sources.jar
pi4j-example.jar
pi4j-gpio-extension-javadoc.jar
pi4j-gpio-extension-sources.jar
pi4j-gpio-extension.jar
pi4j-service-javadoc.jar
pi4j-service-sources.jar
pi4j-service.jar

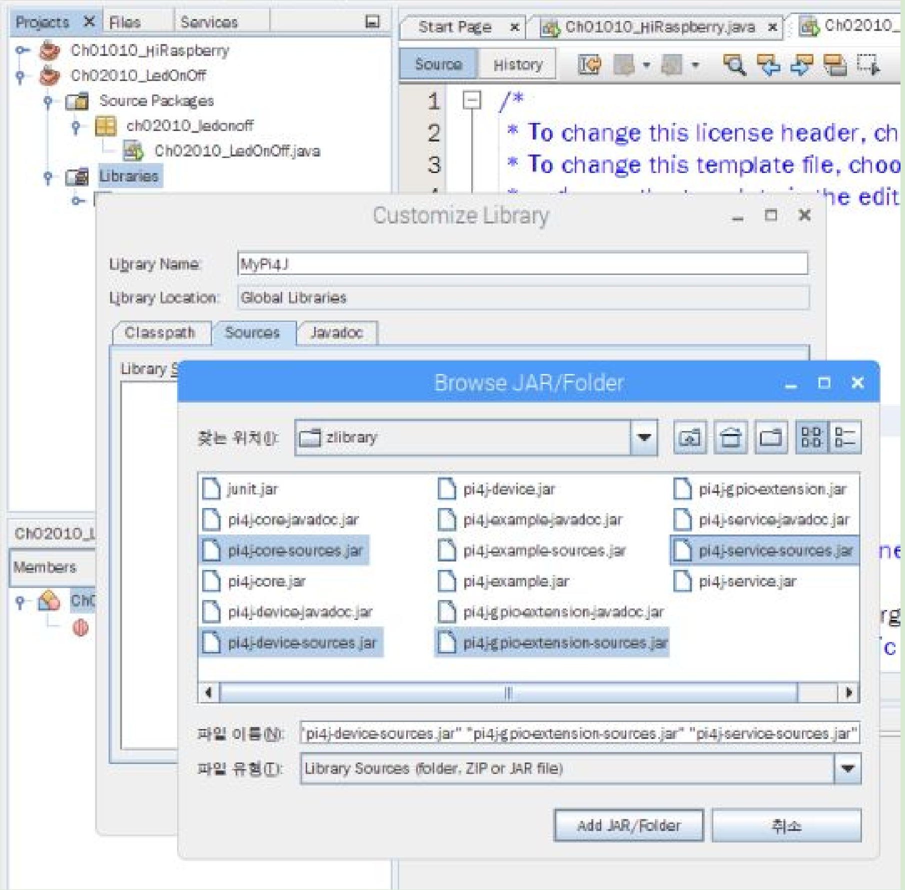
파일 이름(N): pi4j-service.jar

파일 유형(T): Classpath Entry (folder, ZIP or JAR file)

(pi4j-1.0 버전의 경우 선택할 JAR 파일이 4개입니다).

RaspberryPi3 :: Java :: Lec_JavaRasPi3_0140 :: Ch02010_LedOnOff

8.) Select the "Sources" option and add the "pi4j-core-sources.jar".



```
/*
 * To change this license header, ch
 * To change this template file, choo
```

Library Name: MyPi4J

Library Location: Global Libraries

찾는 위치(I): zlibrary

junit.jar
pi4j-core-javadoc.jar
pi4j-core-sources.jar
pi4j-core.jar
pi4j-device-javadoc.jar
pi4j-device-sources.jar
pi4j-device.jar
pi4j-example-javadoc.jar
pi4j-example-sources.jar
pi4j-example.jar
pi4j-gpio-extension-javadoc.jar
pi4j-gpio-extension-sources.jar
pi4j-gpio-extension.jar
pi4j-service-javadoc.jar
pi4j-service-sources.jar
pi4j-service.jar

파일 이름(N): "pi4j-device-sources.jar" "pi4j-gpio-extension-sources.jar" "pi4j-service-sources.jar"

파일 유형(T): Library Sources (folder, ZIP or JAR file)

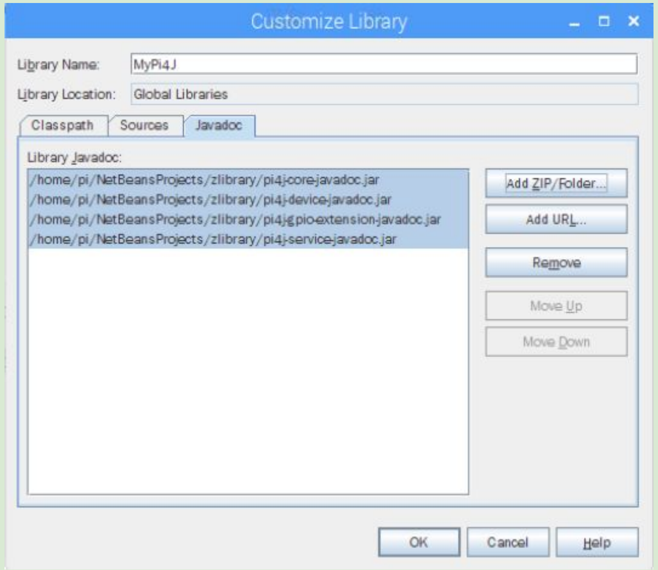
(pi4j-1.0 버전의 경우 선택할 JAR 파일이 4개입니다).

RaspberryPi3 :: Java :: Lec_JavaRasPi3_0140 :: Ch02010_LedOnOff

9.) Select the "JavaDoc" option and add the "pi4j-core-javadoc.jar".

10.) Select the "OK" button to finish adding the Pi4J Library to NetBeans.

11.) You should now be back on the "Add Library" dialog. Make sure that "MyPi4J" is selected and click "Add Library" to add it to your project.



Customize Library

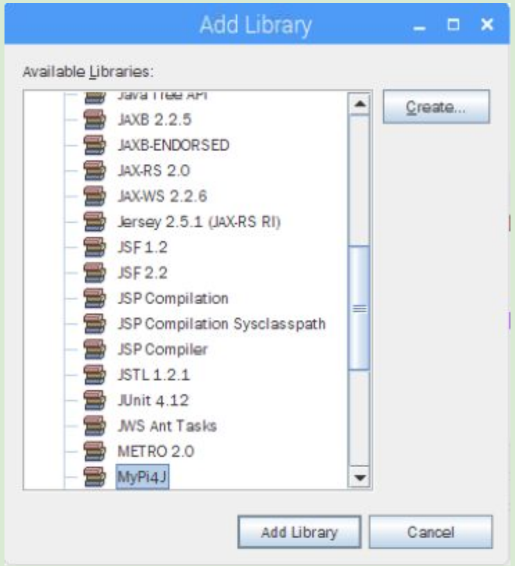
Library Name: MyPi4J

Library Location: Global Libraries

Library Javadoc:

```
/home/pi/NetBeansProjects/zlibrary/pi4j-core-javadoc.jar
/home/pi/NetBeansProjects/zlibrary/pi4j-device-javadoc.jar
/home/pi/NetBeansProjects/zlibrary/pi4j-gpio-extension-javadoc.jar
/home/pi/NetBeansProjects/zlibrary/pi4j-service-javadoc.jar
```

→

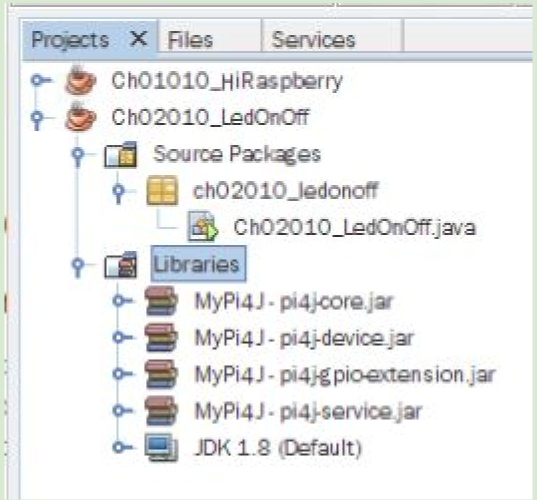


Add Library

Available Libraries:

```
Java Tree API
JAXB 2.2.5
JAXB-ENDORSED
JAX-RS 2.0
JAX-WS 2.2.6
Jersey 2.5.1 (JAX-RS RI)
JSF 1.2
JSF 2.2
JSP Compilation
JSP Compilation Sysclasspath
JSP Compiler
JSTL 1.2.1
JUnit 4.12
JWS Ant Tasks
METRO 2.0
MyPi4J
```

→



```
Ch01010_HiRaspberry
Ch02010_LedOnOff
    Source Packages
        ch02010_ledonoff
            Ch02010_LedOnOff.java
    Libraries
        MyPi4J - pi4j-core.jar
        MyPi4J - pi4j-device.jar
        MyPi4J - pi4j-gpio-extension.jar
        MyPi4J - pi4j-service.jar
        JDK 1.8 (Default)
```

12.) Next, copy all the ControlGpioExample source code from https://raw.githubusercontent.com/Pi4J/[2] ... ample.java and paste it into your local ControlGpioExample.java source file replacing the existing source.

13.) Select the "Run" menu item and then "Build Project" sub menu item. This should compile the program and make sure there are no error. Look at the bottom of the IDE for the "Output" window and you should see a "BUILD SUCCESSFUL" message in the output text.

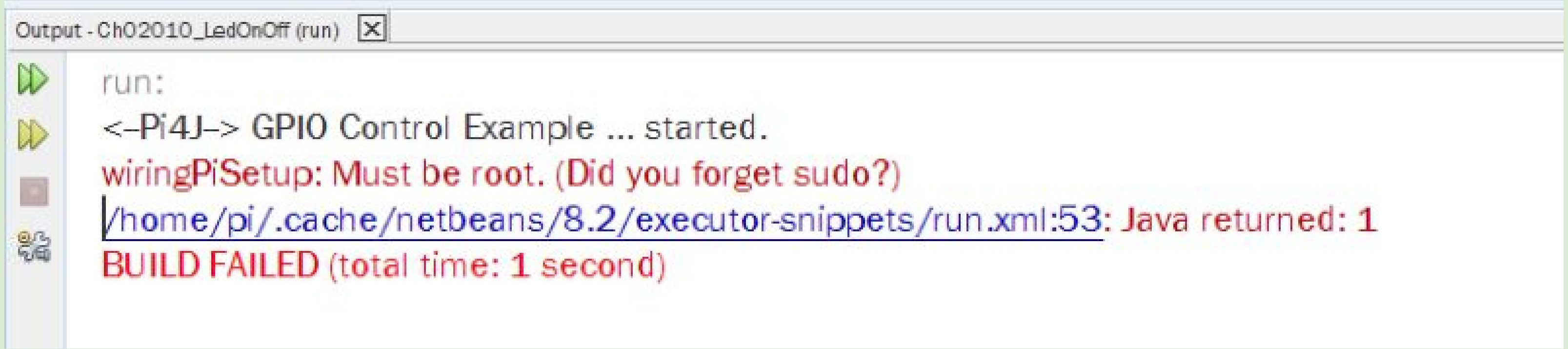
2.2. 라즈베리파이에서 넷빈을 이용하여 Simple GPIO Control using Pi4J 예제[3]를 실행하면 빌드는 되지만 오류가 발생하여, 터미널에서 빌드될 때 생성된 /home/pi/NetBeansProjects/Ch02010_LedOnOff/dist 폴더의 README.TXT를 참조하여 sudo 권한으로 Java 파일을 실행합니다

[2] https://raw.githubusercontent.com/Pi4J/pi4j/master/pi4j-example/src/main/java/ControlGpioExample.java

[3] http://pi4j.com/example/control.html Simple GPIO Control using Pi4J
https://github.com/Pi4J/pi4j/blob/master/pi4j-example/src/main/java/ControlGpioExample.java

RaspberryPi3 :: Java :: Lec_JavaRasPi3_0140 :: Ch02010_LedOnOff

2.2.1. 오류내용 :



```
run:
<--Pi4J--> GPIO Control Example ... started.
wiringPiSetup: Must be root. (Did you forget sudo?)
/home/pi/.cache/netbeans/8.2/executor-snippets/run.xml:53: Java returned: 1
BUILD FAILED (total time: 1 second)
```

2.2.2. 라즈베리파이에서 직접 Ch02010_LedOnOff Java 파일 실행합니다.

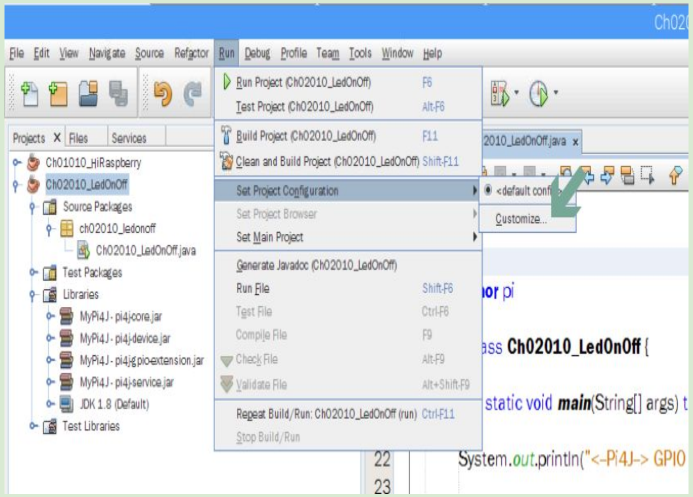
```
pi@raspberrypi:~/NetBeansProjects/Ch02010_LedOnOff/dist $ sudo java -jar
"Ch02010_LedOnOff.jar"
<--Pi4J--> GPIO Control Example ... started.
--> GPIO state should be: ON
--> GPIO state should be: OFF
--> GPIO state should be: ON
--> GPIO state should be: OFF
```

RaspberryPi3 :: Java :: Lec_JavaRasPi3_0140 :: Ch02010_LedOnOff

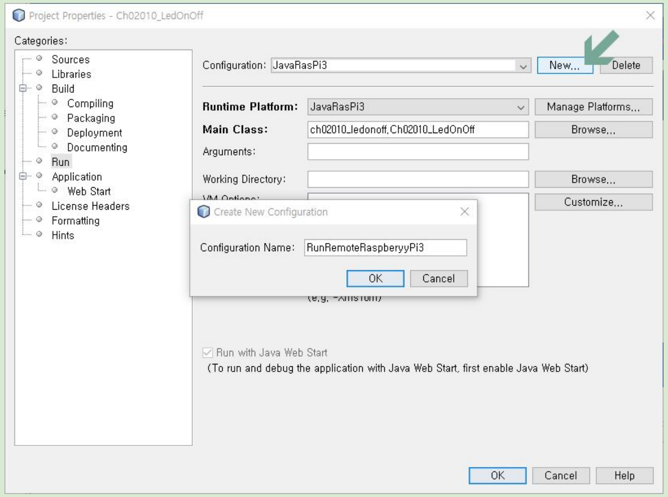
```
--> GPIO state should be: ON for only 1 second
Exiting ControlGpioExample
pi@raspberrypi:~/NetBeansProjects/Ch02010_LedOnOff/dist $
```

RaspberryPi3 :: Java :: Lec_JavaRasPi3_0140 :: Ch02010_LedOnOff

2.3. 넷빈에서 원격으로 Java 프로젝트를 실행하는 설정방법 알아보기 :


File Edit View Navigate Source Refactor Run Debug Profile Team Tools Window Help
Run Project (Ch02010_LedOnOff) F6
Test Project (Ch02010_LedOnOff) Alt-F6
Build Project (Ch02010_LedOnOff) F11
Clean and Build Project (Ch02010_LedOnOff) Shift-F11
Set Project Configuration
Set Project Browser
Set Main Project
Generate Javadoc (Ch02010_LedOnOff)
Run File Shift-F6
Test File Ctrl-F6
Compile File F9
Check File Alt-F9
Validate File Alt+Shift-F9
Repeat Build/Run: Ch02010_LedOnOff (run) Ctrl-F11
Stop Build/Run
● <default conf
Customize...
Projects Files Services
Ch01010_HiRaspberry
Ch02010_LedOnOff
Source Packages
ch02010_ledonoff
Ch02010_LedOnOff.java
Test Packages
Libraries
MyPi4J - pi4j-core.jar
MyPi4J - pi4j-device.jar
MyPi4J - pi4j-gpio-extension.jar
MyPi4J - pi4j-service.jar
JDK 1.8 (Default)
Test Libraries

1.) In NetBeans, select the "Run" menu item and then "Set Project Configuration" sub menu item and click "Customize".



Configuration: JavaRasPi3

Runtime Platform: JavaRasPi3

Main Class: ch02010_ledonoff.Ch02010_LedOnOff

Arguments:

Working Directory:

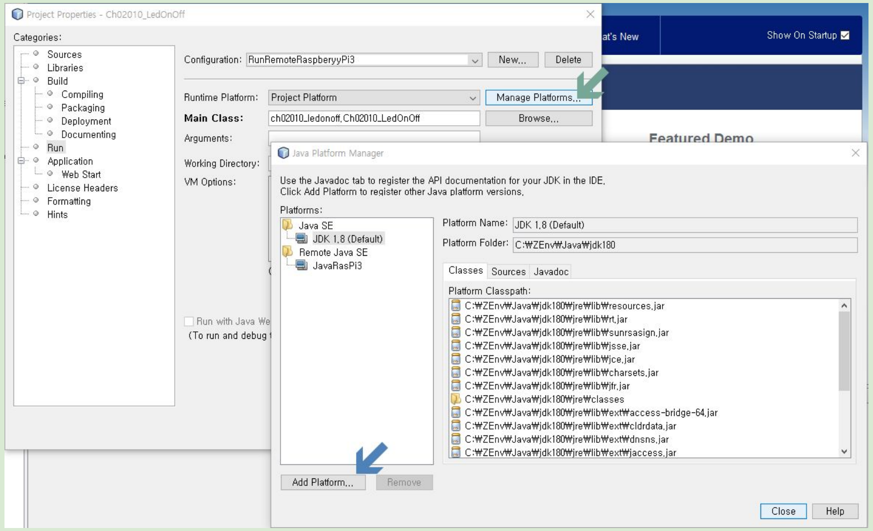
Configuration Name: RunRemoteRaspberyyPi3

☑ Run with Java Web Start

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

2.) Click "New" button to create a new configuration. Provide a name such as "Raspberry Pi" for the configuration.

RaspberryPi3 :: Java :: Lec_JavaRasPi3_0140 :: Ch02010_LedOnOff



Project Properties - Ch02010_LedOnOff

Configuration: RunRemoteRaspberyyPi3

Runtime Platform: Project Platform

Main Class: ch02010_ledonoff.Ch02010_LedOnOff

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.

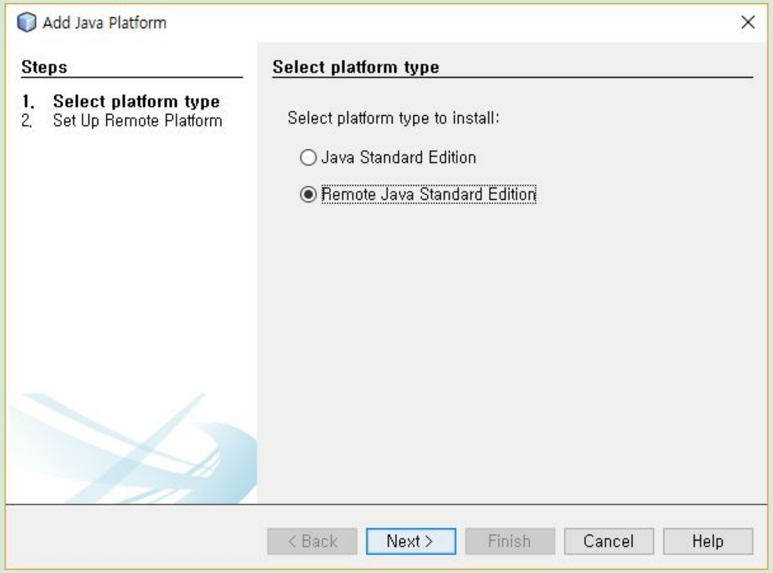
Platform Name: JDK 1.8 (Default)

Platform Folder: C:₩ZEnv₩Java₩jdk180

Platform Classpath:

```
C:₩ZEnv₩Java₩jdk180₩jre₩lib₩resources.jar
C:₩ZEnv₩Java₩jdk180₩jre₩lib₩rt.jar
C:₩ZEnv₩Java₩jdk180₩jre₩lib₩sunrsasign.jar
C:₩ZEnv₩Java₩jdk180₩jre₩lib₩jsse.jar
C:₩ZEnv₩Java₩jdk180₩jre₩lib₩jce.jar
C:₩ZEnv₩Java₩jdk180₩jre₩lib₩charsets.jar
C:₩ZEnv₩Java₩jdk180₩jre₩lib₩jfr.jar
C:₩ZEnv₩Java₩jdk180₩jre₩classes
C:₩ZEnv₩Java₩jdk180₩jre₩lib₩ext₩access-bridge-64.jar
C:₩ZEnv₩Java₩jdk180₩jre₩lib₩ext₩cldrdata.jar
C:₩ZEnv₩Java₩jdk180₩jre₩lib₩ext₩dnsns.jar
C:₩ZEnv₩Java₩jdk180₩jre₩lib₩ext₩jaccess.jar
```

3.) Next, click "Manage Platforms", then "Add Platform".



Add Java Platform

Select platform type

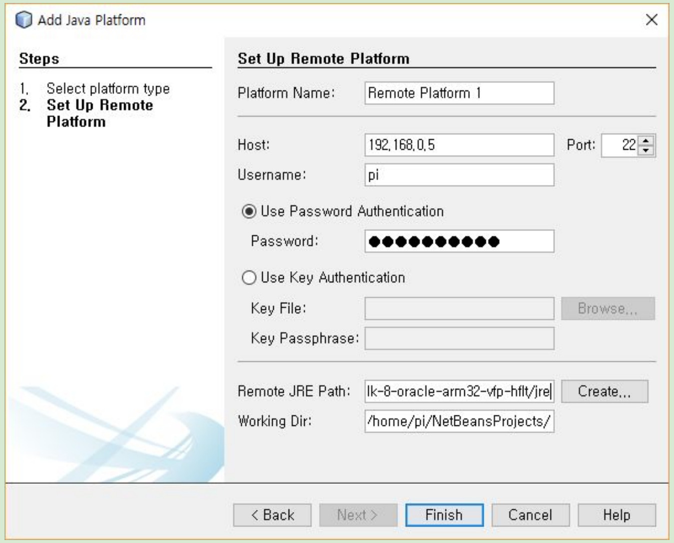
Select platform type to install:

○ Java Standard Edition

● Remote Java Standard Edition

4.) Select "Remote Java Standard Edition" and click "Next" button.

RaspberryPi3 :: Java :: Lec_JavaRasPi3_0140 :: Ch02010_LedOnOff



Set Up Remote Platform

Platform Name: Remote Platform 1

Host: 192.168.0.5 Port: 22

Username: pi

● Use Password Authentication

Password: ●●●●●●●●●●●

○ Use Key Authentication

Key File:

Key Passphrase:

Remote JRE Path: lk-8-oracle-arm32-vfp-hflt/jre

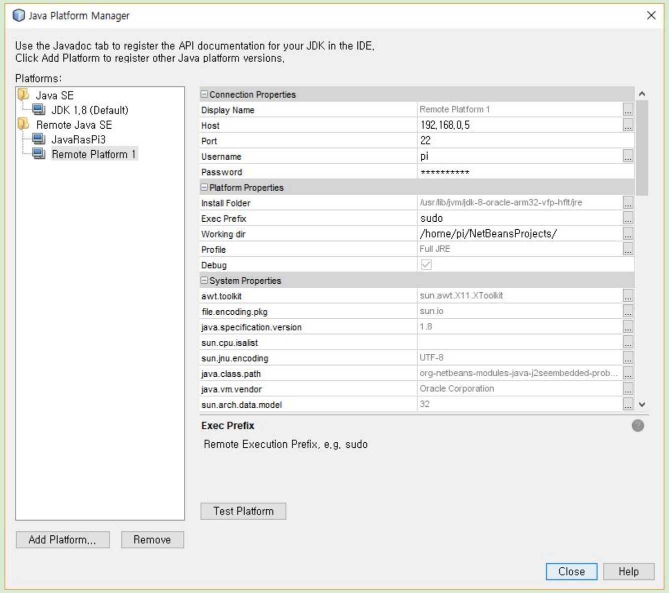
Working Dir: /home/pi/NetBeansProjects/

5.) Provide a platform name such as "RPI-Platform"

6.) Provide the host name/address, username "pi" and password "raspberry".

7.) For the "Remote JRE Path"field, use: /usr/lib/jvm/jdk-8-oracle-arm32-vfp-hflt/jre

8.) Click "Finish" button.



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:

⊟ Connection Properties	
Display Name	Remote Platform 1
Host	192.168.0.5
Port	22
Username	pi
Password	**********
⊟ Platform Properties	
Install Folder	/usr/lib/jvm/jdk-8-oracle-arm32-vfp-hflt/jre
Exec Prefix	sudo
Working dir	/home/pi/NetBeansProjects/
Profile	Full JRE
Debug	☑
⊟ System Properties	
awt.toolkit	sun.awt.X11.XToolkit
file.encoding.pkg	sun.io
java.specification.version	1.8
sun.cpu.isalist	
sun.jnu.encoding	UTF-8
java.class.path	org-netbeans-modules-java-j2seembedded-prob...
java.vm.vendor	Oracle Corporation
sun.arch.data.model	32

Exec Prefix

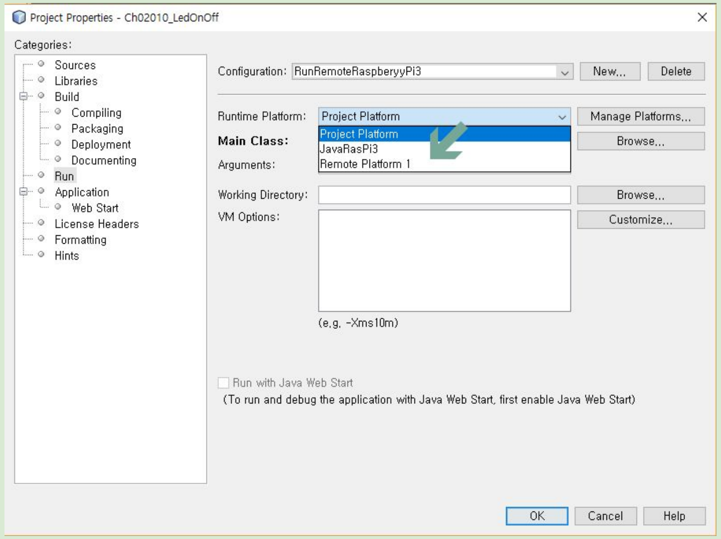
Remote Execution Prefix, e.g. sudo

9.) Now you should be at the "Java Platform Manager" dialog.

10.) Look for the "Exec Prefix" field under "Platform Properties". Edit this field and enter "sudo" (We need sudo to launch GPIO programs.)

11.) Click "Close" to close the "Java Platform Manager" dialog.

RaspberryPi3 :: Java :: Lec_JavaRasPi3_0140 :: Ch02010_LedOnOff



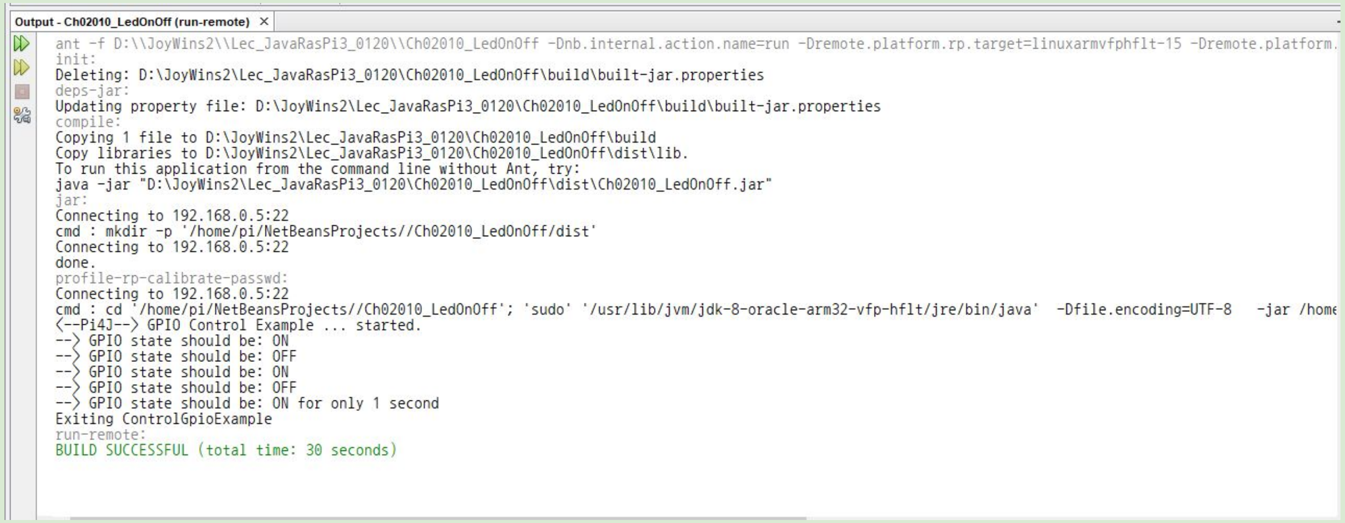
Project Properties - Ch02010_LedOnOff

Configuration: RunRemoteRaspberyyPi3

Runtime Platform: Project Platform

12.) Now back at the "Project Properties" dialog, select from the dropdown menu, Runtime Platform = "RPI-Platform". Click "OK" to close.

13.) From the NetBeans menu, select the "Run" menu item and then "Run Project" sub menu item.



Output - Ch02010_LedOnOff (run-remote)

```
ant -f D:\\JoyWins2\\Lec_JavaRasPi3_0120\\Ch02010_LedOnOff -Dnb.internal.action.name=run -Dremote.platform.rp.target=linuxarmvfphflt-15 -Dremote.platform.
init:
Deleting: D:\JoyWins2\Lec_JavaRasPi3_0120\Ch02010_LedOnOff\build\built-jar.properties
deps-jar:
Updating property file: D:\JoyWins2\Lec_JavaRasPi3_0120\Ch02010_LedOnOff\build\built-jar.properties
compile:
Copying 1 file to D:\JoyWins2\Lec_JavaRasPi3_0120\Ch02010_LedOnOff\build
Copy libraries to D:\JoyWins2\Lec_JavaRasPi3_0120\Ch02010_LedOnOff\dist\lib.
To run this application from the command line without Ant, try:
java -jar "D:\JoyWins2\Lec_JavaRasPi3_0120\Ch02010_LedOnOff\dist\Ch02010_LedOnOff.jar"
jar:
Connecting to 192.168.0.5:22
cmd : mkdir -p '/home/pi/NetBeansProjects//Ch02010_LedOnOff/dist'
Connecting to 192.168.0.5:22
done.
profile-rp-calibrate-passwd:
Connecting to 192.168.0.5:22
cmd : cd '/home/pi/NetBeansProjects//Ch02010_LedOnOff'; 'sudo' '/usr/lib/jvm/jdk-8-oracle-arm32-vfp-hflt/jre/bin/java'  -Dfile.encoding=UTF-8   -jar /home
<--Pi4J--> GPIO Control Example ... started.
--> GPIO state should be: ON
--> GPIO state should be: OFF
--> GPIO state should be: ON
--> GPIO state should be: OFF
--> GPIO state should be: ON for only 1 second
Exiting ControlGpioExample
run-remote:
BUILD SUCCESSFUL (total time: 30 seconds)
```

RaspberryPi3 :: Java :: Lec_JavaRasPi3_0140 :: Ch02010_LedOnOff

3. Java File :

   Lec_JavaRasPi3_0140/Ch02010_LedOnOff/src/ch02010_ledonoff/Ch02010_LedOnOff.java

```
/*
 * To change this license header, choose License Headers in Project Properties.
 * To change this template file, choose Tools | Templates
 * and open the template in the editor.
 */
package ch02010_ledonoff;


import com.pi4j.io.gpio.GpioController;
import com.pi4j.io.gpio.GpioFactory;
import com.pi4j.io.gpio.GpioPinDigitalOutput;
import com.pi4j.io.gpio.PinState;
```

RaspberryPi3 :: Java :: Lec_JavaRasPi3_0140 :: Ch02010_LedOnOff

```
import com.pi4j.io.gpio.RaspiPin;


/*
...NetBeans :: wiringPiSetup: Must be root (Did you forget sudo ?)[4]
...NetBeans :: WiringPi에서 "Unable to determine hardware version. I see : Hardware :
BCM2835 '오류[5]
 */
public class Ch02010_LedOnOff  {


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


        System.out.println("<--Pi4J--> GPIO Control Example ... started.");


```

[4] https://raspberrypi.stackexchange.com/questions/54208/wiringpisetup-must-be-root-did-you-forget-sudo wiringPiSetup: Must be root (Did you forget sudo ?)

[5] https://qiita.com/jollyjoester/items/ba59e5d43e28b701f120 WiringPiで「Unable to determine hardware version. I see: Hardware　: BCM2835」エラー

RaspberryPi3 :: Java :: Lec_JavaRasPi3_0140 :: Ch02010_LedOnOff

```
        // create gpio controller
        final GpioController gpio = GpioFactory.getInstance();

        // provision gpio pin #01 as an output pin and turn on
        final GpioPinDigitalOutput pin = gpio.provisionDigitalOutputPin(RaspiPin.GPIO_01,
"MyLED", PinState.HIGH);

        // set shutdown state for this pin
        //...Java Code Examples for
com.pi4j.io.gpio.GpioPinDigitalOutput.setShutdownOptions()[6]
        //   configure the pin shutdown behavior; these settings will be
        //   automatically applied to the pin when the application is terminated.
        pin.setShutdownOptions(true, PinState.LOW);
```

[6] https://www.programcreek.com/java-api-examples/?class=com.pi4j.io.gpio.GpioPinDigitalOutput&method=setShutdownOptions Java Code Examples for com.pi4j.io.gpio.GpioPinDigitalOutput.setShutdownOptions()

RaspberryPi3 :: Java :: Lec_JavaRasPi3_0140 :: Ch02010_LedOnOff

```
System.out.println("--> GPIO state should be: ON");

Thread.sleep(5000);

// turn off gpio pin #01
pin.low();
System.out.println("--> GPIO state should be: OFF");

Thread.sleep(5000);

// toggle the current state of gpio pin #01 (should turn on)
pin.toggle();
System.out.println("--> GPIO state should be: ON");
```

RaspberryPi3 :: Java :: Lec_JavaRasPi3_0140 :: Ch02010_LedOnOff

```
Thread.sleep(5000);

// toggle the current state of gpio pin #01  (should turn off)
pin.toggle();
System.out.println("--> GPIO state should be: OFF");

Thread.sleep(5000);

// turn on gpio pin #01 for 1 second and then off
System.out.println("--> GPIO state should be: ON for only 1 second");
pin.pulse(1000, true); // set second argument to 'true' use a blocking call

// stop all GPIO activity/threads by shutting down the GPIO controller
```

RaspberryPi3 :: Java :: Lec_JavaRasPi3_0140 :: Ch02010_LedOnOff

```
        // (this method will forcefully shutdown all GPIO monitoring threads and scheduled
tasks)
        gpio.shutdown();

        System.out.println("Exiting ControlGpioExample");
    }
}
```

4. Result Image :