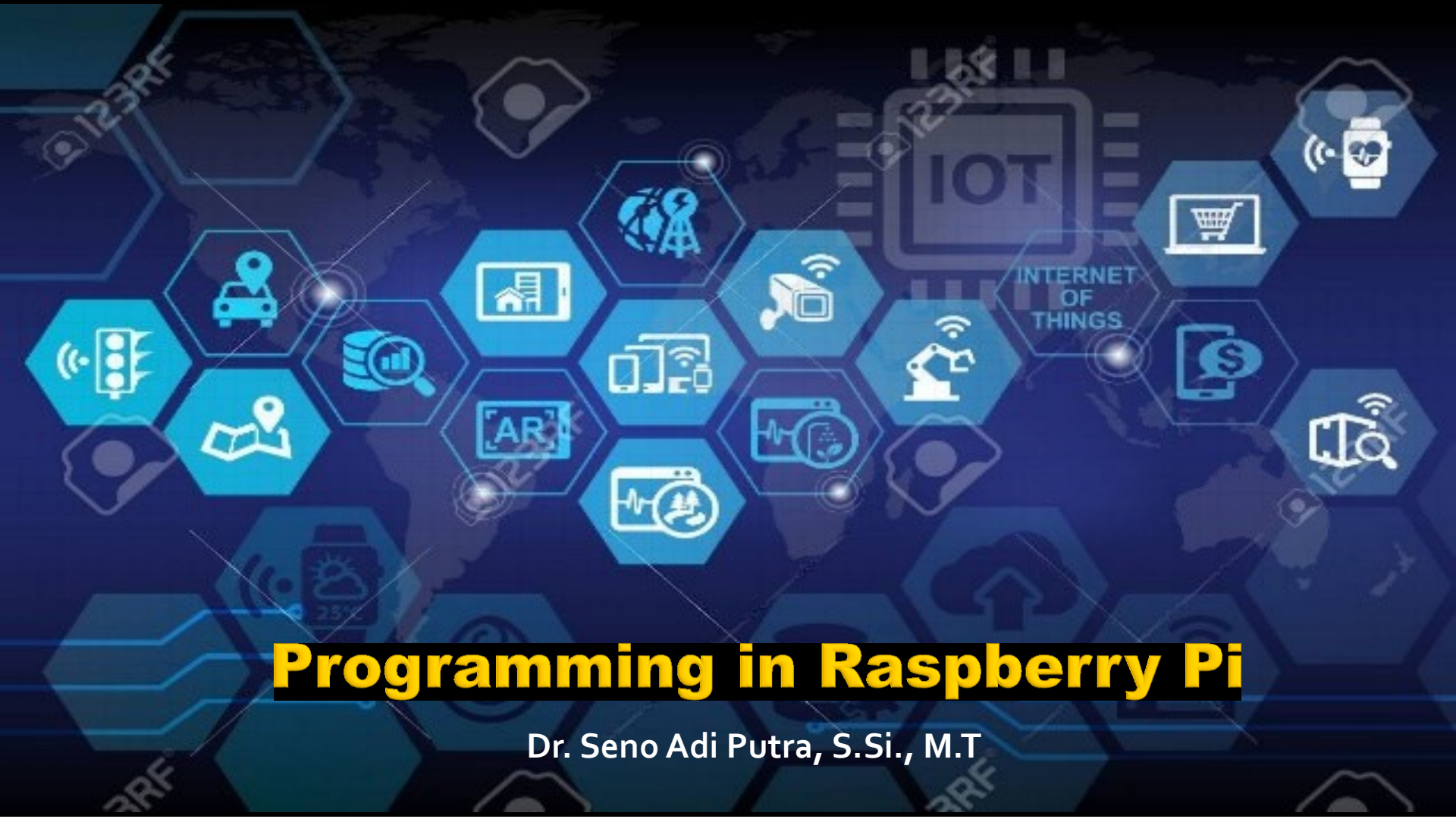


Programming in Raspberry Pi

Dr. Seno Adi Putra, S.Si., M.T

Laboratorium *Enterprise Intelligent System*
Kelompok Keilmuan *Cybernetics*
Fakultas Rekayasa Industri – Universitas Telkom

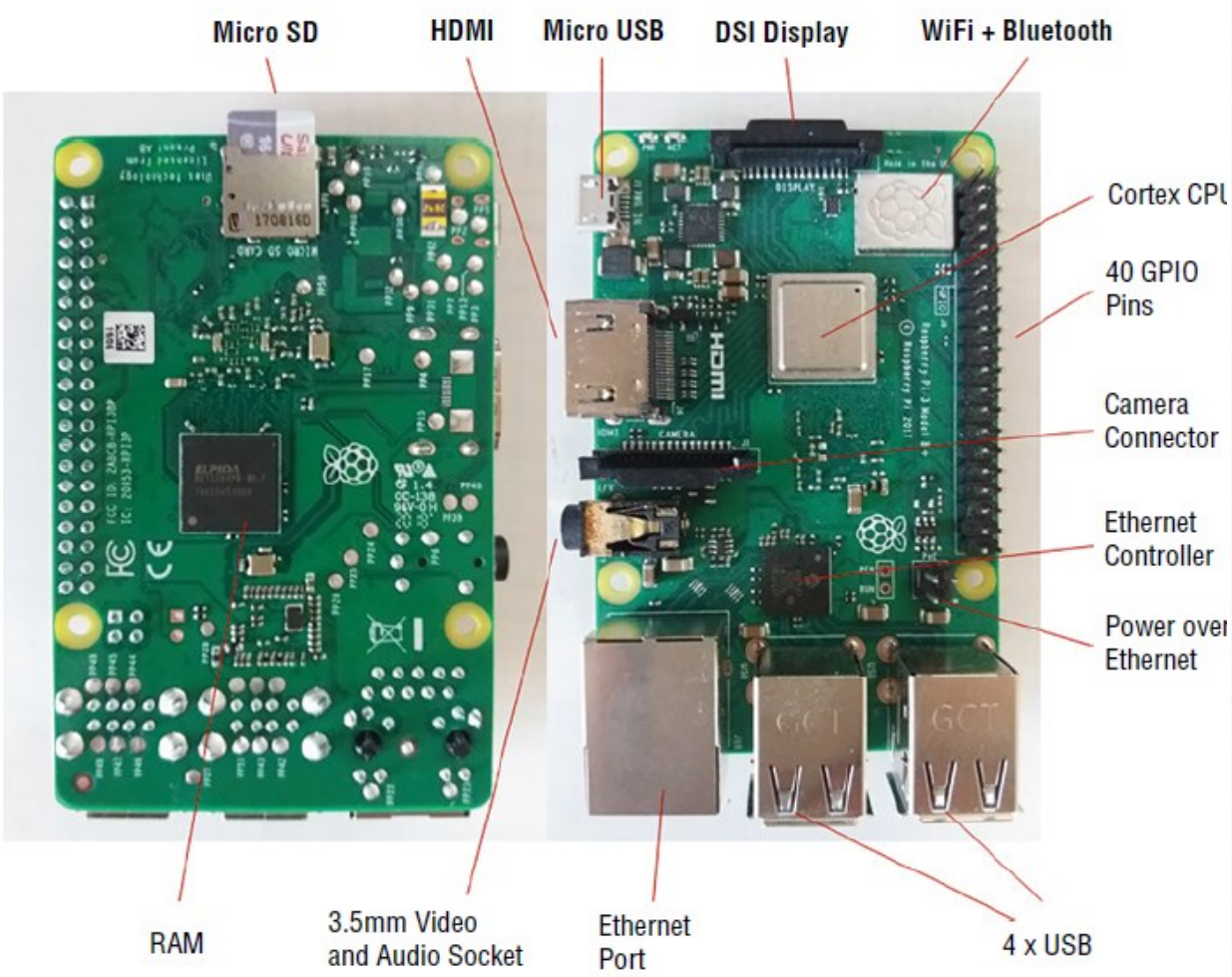
Agenda

- Raspberry Pi
- Raspberry Setup
- GPIO Example Code

Bagian I

Raspberry PI

Raspberry Pi



Raspberry Pi 3 Model B (J8 Header)

GPIO#	NAME				NAME	GPIO#
	3.3 VDC Power	1		2	5.0 VDC Power	
8	GPIO 8 SDA1 (I2C)	3		4	5.0 VDC Power	
9	GPIO 9 SCL1 (I2C)	5		6	Ground	
7	GPIO 7 GPCLK0	7		8	GPIO 15 TxD (UART)	15
	Ground	9		10	GPIO 16 RxD (UART)	16
0	GPIO 0	11		12	GPIO 1 PCM_CLK/PWM0	1
2	GPIO 2	13		14	Ground	
3	GPIO 3	15		16	GPIO 4	4
	3.3 VDC Power	17		18	GPIO 5	5
12	GPIO 12 MOSI (SPI)	19		20	Ground	
13	GPIO 13 MISO (SPI)	21		22	GPIO 6	6
14	GPIO 14 SCLK (SPI)	23		24	GPIO 10 CE0 (SPI)	10
	Ground	25		26	GPIO 11 CE1 (SPI)	11
30	SDAD (12C ID EEPROM)	27		28	SCLO (I2C ID EEPROM)	31
21	GPIO 21 GPCLK1	29		30	Ground	
22	GPIO 22 GPCLK2	31		32	GPIO 26 PWM0	26
23	GPIO 23 PWM1	33		34	Ground	
24	GPIO 24 PCM_FS/PWM1	35		36	GPIO 27	27
25	GPIO 25	37		38	GPIO 28 PCM_DIN	28
	Ground	39		40	GPIO 29 PCM_DOUT	29

Products

- **BeagleBone**
  - ARM Cortex CPU, Graphics Engine
  - RAM, 4 GB onboard flash,
  - two PRU 32-bit microcontrollers,
  - two USB ports, 10/100M Ethernet,
  - HDMI LCD interfaces,
  - serial port, ADC, I2C, SPI, PWM pins.
  - Price: $67.99.
  - https://beagleboard.org/

Raspberry Pi Quad-core ARM Cortex CPU, Broadcom GPU, RAM, four USB ports, 10/100 M Ethernet, 802.11n Wireless LAN, Bluetooth 4.0, HDMI port, and GPIO pins.
Price: $35.
https://www.raspberrypi.org/

- **Odroid**
  - Samsung Cortex CPUs, Mali GPU, RAM, eMMC flash storage,
  - Two USB 3.0, one USB 2.0, Gigabit Ethernet, HDMI port, and GPIO pins.
  - Price: $59.
  - https://www.hardkernel.com/

Bagian II

Raspberry Setup

Preparing Raspberry PI

- To run Java on the Raspberry Pi, you will first need to have the Raspberry Pi hardware. You can visit the Raspberry Pi web site to find the nearest distributor or simply get it from Amazon or eBay.

- Along with the Raspberry Pi board, you will also need a microSD card of 8GB or more to store the operating system called Raspbian, a GNU Debian Linux operating system.

- You can get a pre-installed Raspbian microSD card from either of the following:
  - **RS (search for *raspberry pi microSD*):** https://uk.rs-online.com/
  - **The Pi Hut:** https://thepihut.com/products/raspbian-preinstalled-sd-card

- Or you can download the operating image from https://www.raspberrypi.org/downloads/.

Preparing Raspberry PI

Interaction with Raspberry PI

- To interact with the Raspberry Pi board, you will need a USB keyboard, a USB mouse, and either a TV with an HDMI port or a standard computer monitor through an HDMI-to-VGA converter.

- Alternatively, you can connect the Raspberry Pi remotely using secure shell (SSH) or the Windows Remote Desktop Connection. To use SSH, you need to enable the SSH service on the Raspberry Pi first. You can enable the SSH service in one of three ways.
  - From Raspberry Pi Desktop, launch Raspberry Pi Configuration from the Preferences menu, navigate to the Interfaces tab, select Enabled next to SSH, and click OK.

  - From a Raspberry Pi terminal, type **sudo raspi-config**. From the configuration menu, select Interfacing Options, navigate to and select SSH, choose Yes, select OK, and choose Finish. Here sudo means execute the command as a superuser, that is, as an administrator.

  - From a Rasbperry Pi terminal, type the following commands to enable and start SSH service:

```
$ sudo systemctl enable ssh
$ sudo systemctl start ssh
```

Interaction with Raspberry PI

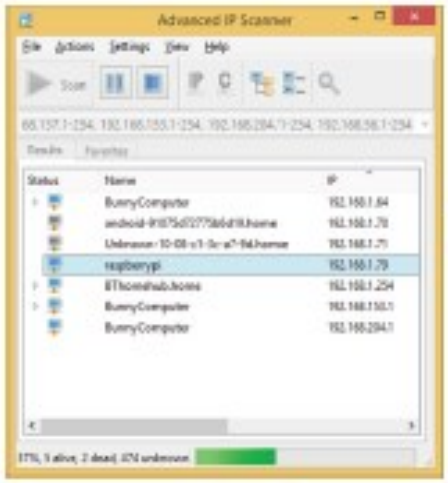
- Favorite approach is to connect Raspberry Pi using the Windows Remote Desktop Connection, as SSH only provides a text mode connection.

- You will need to install the **`xrdp`** and **`tightvncserver`** packages on your Raspberry Pi first, using the following commands:

```
$ sudo apt-get remove xrdp vnc4server tightvncserver
$ sudo apt-get install tightvncserver
$ sudo apt-get install xrdp
```

Interaction with Raspberry PI

- Connect your Raspberry Pi to the Internet using Ethernet cable or WiFi.
- To use the Windows Remote Desktop Connection to connect to Raspberry Pi, you will need to know the Raspberry Pi's IP address.
- If you don't have a TV with HDMI port or an HDMI-to-VGA converter, you can use a free program called the Advanced IP Scanner (https://www.advanced-ip-scanner.com/) to scan the Raspberry Pi's IP address
- After you find the IP address, you can then use Windows Remote Desktop Connection to remotely log in to the Raspberry Pi. After typing in the username and password (the default is *pi* and *raspberry*), you should see the Raspberry Pi desktop



Installing Java on Raspberry Pi

- Once the Raspberry Pi is up and running, you will need the Java JDK software, which should be included with the standard Raspbian operating system, including BlueJ Java IDE.

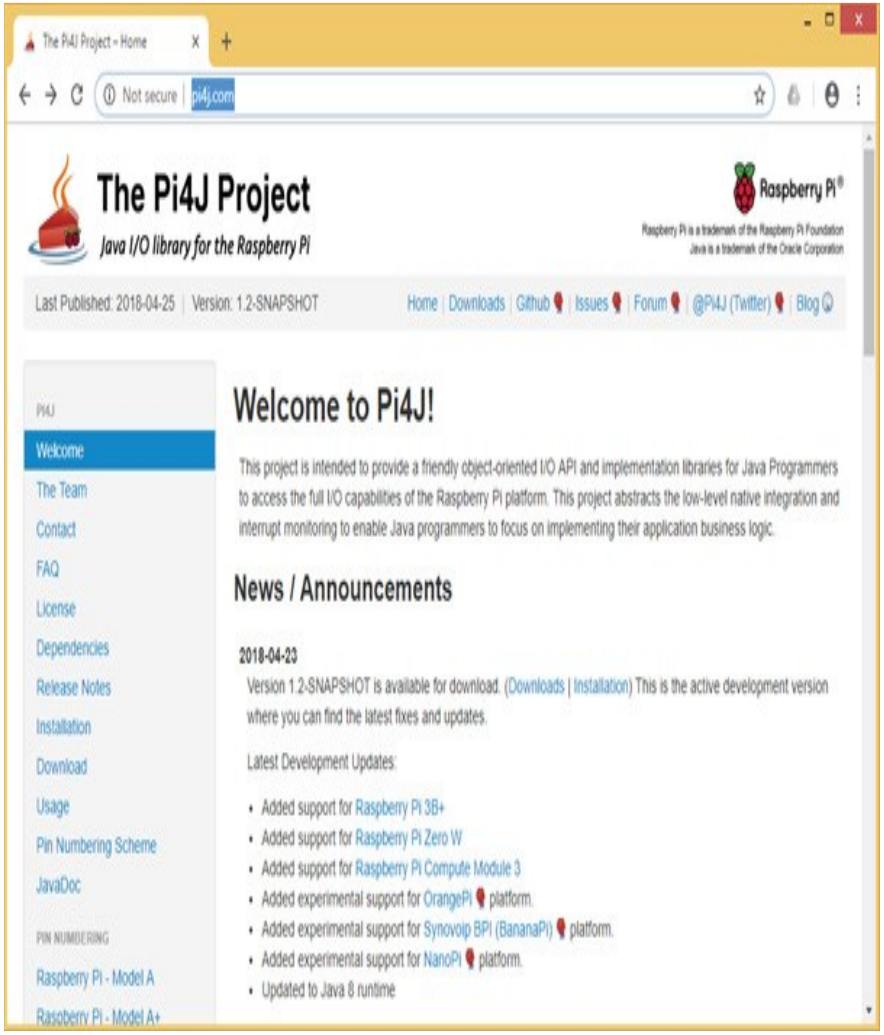
- If it is not or you want to install a different version of Java JDK software, you can simply run the following commands in a Raspbian terminal:

```
$ sudo apt-get update
$ sudo apt-get install oracle-java8-jdk
```

  These commands install Java 8 on the Raspberry Pi, because Java 9 and later versions are not easily available.

Installing Java on Raspberry Pi

- To read and write Raspberry Pi general-purpose input-output (GPIO) pins, you will also need to download and install the **`Pi4J`** library (http://pi4j.com/).
  - There are different ways of downloading and installing the Pi4J library. I prefer to download the entire source from GitHub (https://github.com/Pi4J/pi4j)

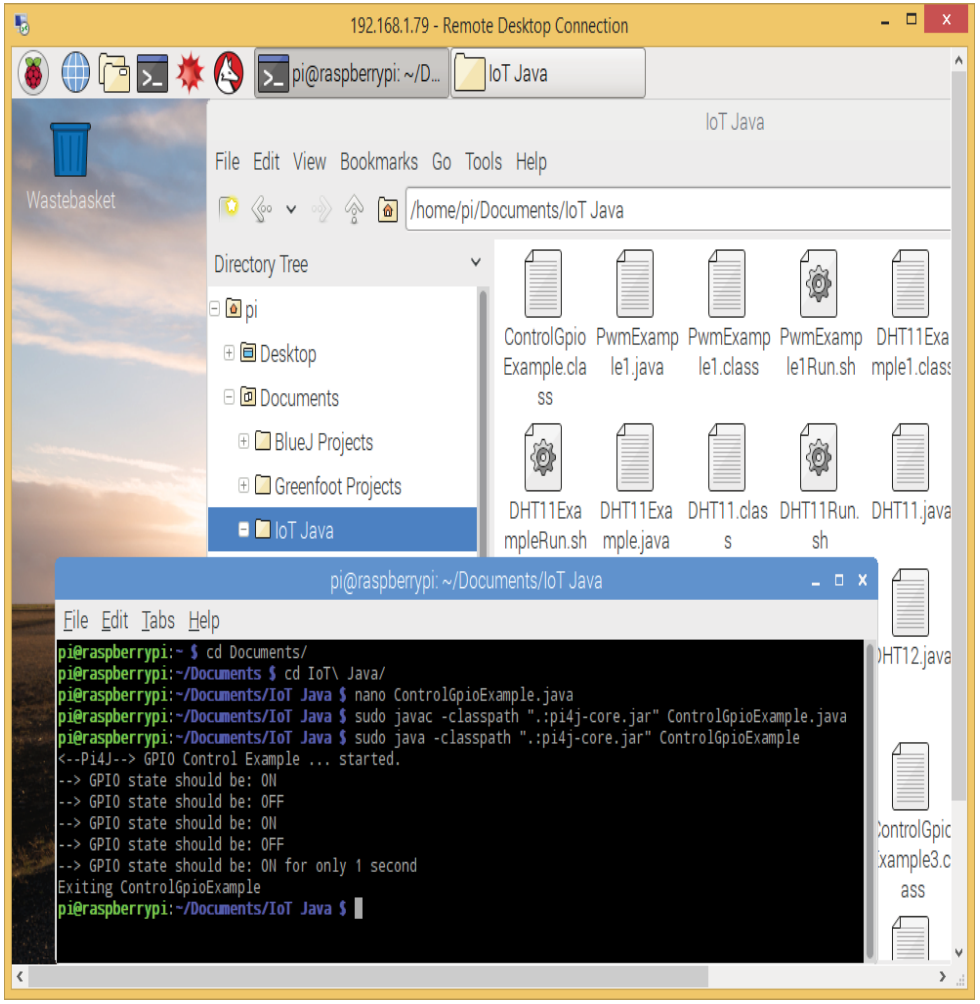


Bagian III

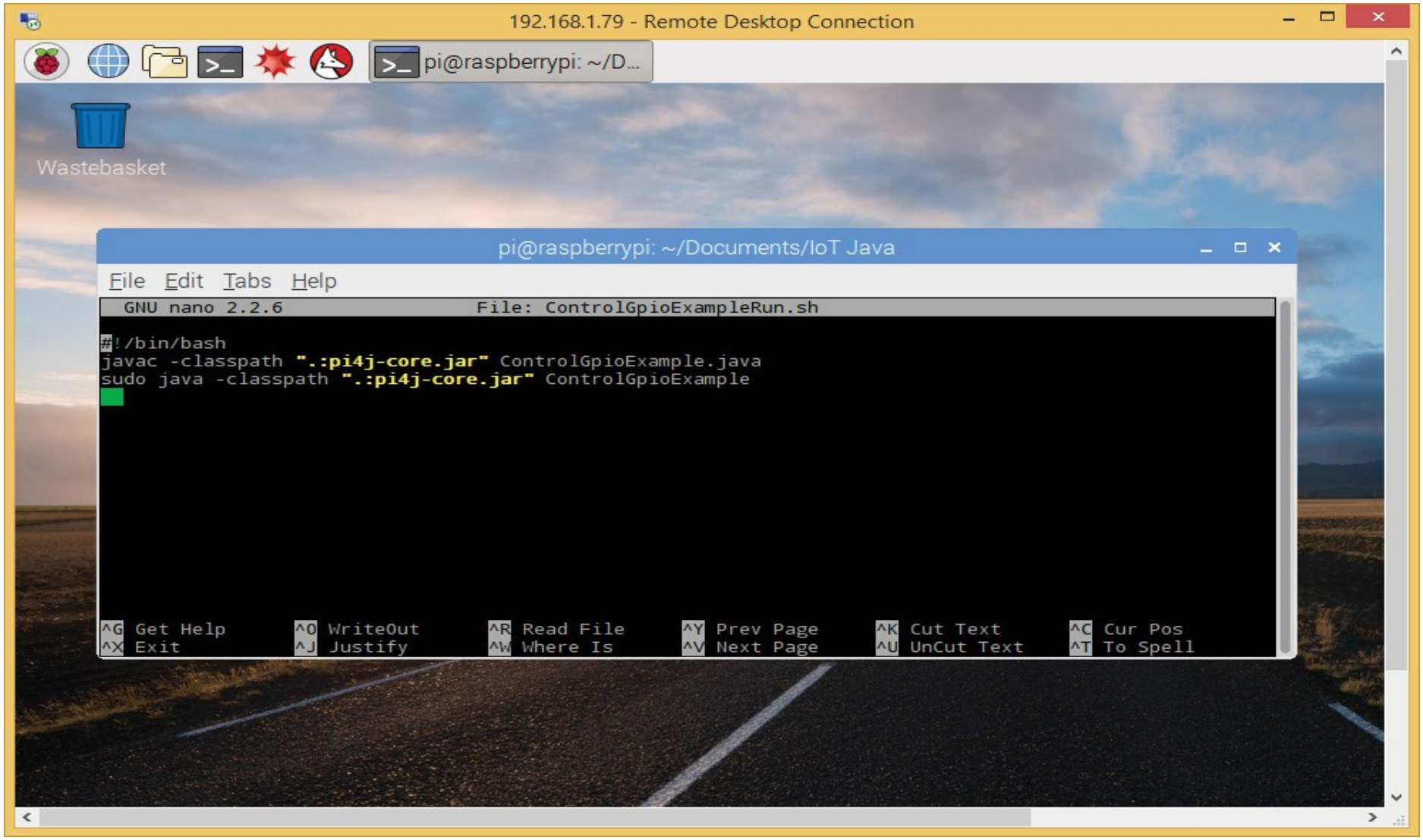
GPIO Example Code

Controlling a blinking LED

- To start, create a dedicated folder for your Java programs. In my case, I created a folder called `IoT Java` under the standard Documents folder
- From the Pi4J library download folder, find the file called `pi4j-core.jar` and copy it to the `IoT Java` folder.
- Find an example file called `Control-`
- `GpioExample.java` and copy it to the `IoT Java` folder.
- Then, from the Terminal window, use the cd command to navigate to the IoT Java folder. You can use the command nano ControlGpioExample.java to view and modify the code



Compiling The Code



Compiling The Code

- To compile and run the program, enter the following commands. It is important to include the `pi4j-core.jar` file in the classpath when compiling and running the program. It is also important to run the program as a superuser.

```
$ javac -classpath ".:pi4j-core.jar" ControlGpioExample.java
$ sudo java -classpath ".:pi4j-core.jar" ControlGpioExample
```

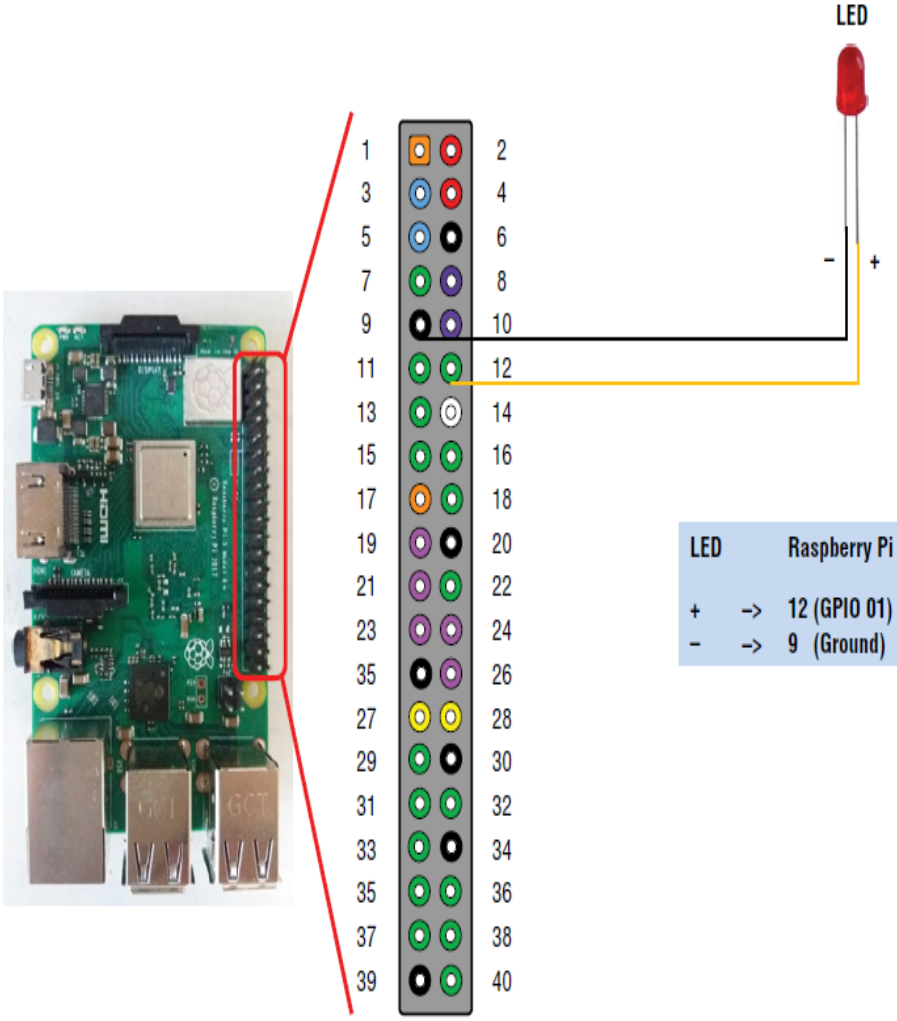
- If you don't want to retype all the lengthy classpath details each time you compile and run the program, you can also create a shell script that does this automatically; see Figure 7.11. In this case, I created a Bash shell script file called ControlGpioExample.sh, which has the following content:

```
#!/bin/Bash
javac -classpath ".:pi4j-core.jar" ControlGpioExample.java
sudo java -classpath ".:pi4j-core.jar" ControlGpioExample
```

- The hash exclamation mark (#!) is referred to as the *shebang*, followed by the path to the shell program. Here I'm using the Bash shell, but there are many other types of shell programs. The shebang must appear on the first line of the script file, and there must also be no spaces before the # or between the ! And the path to the shell program. To run the shell script, simply type the following:

```
$ bash ControlGpioExample.sh
```

Hardware Installation



LED		Raspberry Pi
+	->	12 (GPIO 01)
–	->	9 (Ground)

EXAMPLE 7.1 CONTROLGPIOEXAMPLE1.JAVA, **A SIMPLE GPIO BLINKING LED DEMO PROGRAM**

```
// Example 7.1 ControlGpioExample1.java
import com.pi4j.io.gpio.GpioController;
import com.pi4j.io.gpio.GpioFactory;
import com.pi4j.io.gpio.GpioPinDigitalOutput;
import com.pi4j.io.gpio.PinState;
import com.pi4j.io.gpio.RaspiPin;

public class ControlGpioExample1 {
    // create gpio controller
    final static GpioController gpio = GpioFactory.getInstance();
    // provision gpio pin #01 as an output pin and turn on
    final static GpioPinDigitalOutput pin = gpio.provisionDigitalOutputP
in(RaspiPin.GPIO_01, "MyLED", PinState.HIGH);

    public static void main(String[] args) throws InterruptedException {
        // set shutdown state for this pin
        pin.setShutdownOptions(true, PinState.LOW);

        pin.high();
        Thread.sleep(500);
        pin.low();
        Thread.sleep(500);
        pin.high();
        Thread.sleep(500);

        gpio.shutdown();
    }
}
```

Morse Code Demo

EXAMPLE 7.2 A SIMPLE MORSE CODE DEMO PROGRAM

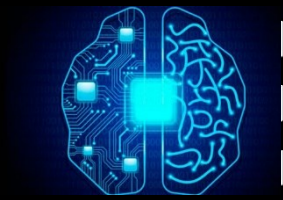
```
// Example 7.2 Morse Code demo program - MorseExample1.java
import com.pi4j.io.gpio.GpioController;
import com.pi4j.io.gpio.GpioFactory;
import com.pi4j.io.gpio.GpioPinDigitalOutput;
import com.pi4j.io.gpio.PinState;
import com.pi4j.io.gpio.RaspiPin;

public class MorseExample1 {
    // create gpio controller
    final static GpioController gpio = GpioFactory.getInstance();
    // provision gpio pin #01 as an output pin and turn on
    final static GpioPinDigitalOutput pin = gpio.provisionDigitalOutputP
in(RaspiPin.GPIO_01, "MyLED", PinState.HIGH);

    public static void main(String[] args) throws InterruptedException {
        // set shutdown state for this pin
        pin.setShutdownOptions(true, PinState.LOW);

        System.out.println("Morse Code 'A' - dot, dash, shortspace");
        dot();
        dash();
        shortspace();
        gpio.shutdown();
    }
    static void dot() throws InterruptedException {
        pin.high();
        Thread.sleep(300);
        pin.low();
        Thread.sleep(300);
    }
    static void dash() throws InterruptedException {
        pin.high();
        Thread.sleep(900);
        pin.low();
        Thread.sleep(300);
    }
    static void shortspace() throws InterruptedException {
        Thread.sleep(600);
    }
}
```

Terima Kasih



Laboratorium *Enterprise Intelligent System*
Kelompok Keilmuan *Cybernetics*
Fakultas Rekayasa Industri – Universitas Telkom