Scala 物联网：在 Raspberry Pi 上使用 Scala 开发物联网应用

Phodal Huang

October 24, 2017

目录

原文链接：https://www.wandianshenme.com/play/scala-iot-raspberry-pi-mosquitto-build-iot-proje

在这里，您可以学习：如何使用 Scala 将 Raspberry Pi 的温度发送给代理，作为你的第一个 Scala 基础的 IoT 应用程序。

让我们开始我们的第一个 IoT 应用程序的旅程，以使世界变得更美好（我永远不会错过机会模拟 Hooli!）。

在这个博客中，Scala 和 IoT 这两项技术终于碰撞到一起，我们将会做许多其他的事情，如：

- 在 Raspberry Pi 上设置 Scala sbt 环境
- 使用 Scala 开发您的第一个 IoT 应用程序
- 在 RaspberryPi 上部署开发的应用程序。

最后，我们要做到这一点。

对于那些还没有关注这个系列的人来说，这是迄今为止发生的事情（也是一些基础内容）：

1. Scala-IoT: Introduction to Internet Of Things.
2. Scala-IoT: What is MQTT? How is it lightweight ?
3. Scala-IoT: Getting started with Raspberry Pi without Monitor or Screen.

到目前为止，我们对 IoT 和 MQTT 有了基本了解，我们设置了我们的 Raspberry Pi，以便我们可以在笔记本屏幕上访问 Raspberry Pi 的桌面！

步骤 1：设置 Scala 环境：安装 Scala

我们将安装 Scala 和 sbt 环境：

1.`ssh` 到 Raspberry Pi

```
ssh pi@192.168.0.1
```

2. 在 Raspberry Pi 上下载 Scala

```
wget https://downloads.lightbend.com/scala/2.11.8/scala-2.11.8.deb
```

3. 使用 `dpkg` 命令安装 Scala

```
sudo dpkg -i scala-2.11.8.deb
```

4. 使用 scala-repl 检查 Scala 是否正确安装

```
1 scala
```

5. 您已成功安装 Scala。

现在我们有一个 Scala 的基本设置，但是我们想要制作项目，所以我们现在需要 sbt！

步骤 2：安装 sbt 并设置 sbt 环境

要安装 sbt，我们必须在 shell 上运行以下命令：

```
1 echo "deb https://dl.bintray.com/sbt/debian /" | sudo tee -a
    /etc/apt/sources.list.d/sbt.list
2 sudo apt-key adv --keyserver hkp://keyserver.ubuntu.com:80 --recv
    2EE0EA64E40A89B84B2DF73499E82A75642AC823
3 sudo apt-get update
4 sudo apt-get install sbt
```

现在我们已经建立了环境，下一步，就可以开发应用程序。

步骤 3：使用 Scala 开发您的第一个基本 IoT 应用程序

概念（General Idea）

那么，这个应用程序将会做这些事：将 Raspberry Pi 的 CPU 的温度发送到具有温度主题的 MQTT 服务器，并且客户端将部署在您的笔记本电脑上。

所以，这里我们使用的是 Mosquitto 作为 MQTT 服务器，因为我一直在使用的 akka-mqtt 服务器仍然是 WIP。

发送温度事件

所以，这里是发送 Raspberry Pi 温度的代码。

```
1 package com.knoldus
2
3 import com.typesafe.config.ConfigFactory
4 import org.eclipse.paho.client.mqttv3.{MqttClient, MqttException,
    MqttMessage}
```

```
5 import org.eclipse.paho.client.mqttv3.persist.MemoryPersistence
6 import scala.sys.process._
7 /** * * MQTT publisher * * @author Shivansh * @mail shiv4nsh@gmail.com * */
8 object MQTTPublisher extends App {
9   val config = ConfigFactory.load()
10  val url = "192.168.0.8"
11  val port = config.getInt("mosquitto-server.port")
12  def publishToserver() = {
13    println("Hey I am publishing")
14    val brokerUrl = s"tcp://$url:$port"
15    val topic = "TemperatureEvent"
16    val tempCommand = "/opt/vc/bin/vcgencmd measure_temp"
17
18    def getMessage = s"Temperature of CPU at ${System.currentTimeMillis()}
          is ${tempCommand.!!.split("=")(1)} "
19
20    var client: MqttClient = null
21    val persistence = new MemoryPersistence
22    try {
23      client = new MqttClient(brokerUrl, MqttClient.generateClientId,
            persistence) client
24      .connect()
25      val msgTopic = client.getTopic(topic)
26      val message = new MqttMessage(getMessage.getBytes("utf-8"))
27      while (true) {
28        val msgPublished = msgTopic.publish(message) msgPublished
              println(s"Publishing the data topic ${msgTopic.getName}
              message: ${message.toString}") Thread
29        .sleep(1000)
30      }
31    } catch {
32      case exception: MqttException =>
            println(s"ExceptionOccured:$exception ")
33    } finally {
34      client.disconnect()
35    }
```

```
36   } publishToserver
37 }
```

所以在这里，我们正在做的是：运行一个获取温度的命令，并且我们正在制作一个主题 “Temperature Event”，并且我们将使用 Raspberry Pi 的当前温度来发布消息。

这个应用程序将在 Raspberry Pi 上运行。

设置 **Mosquitto** 服务器

现在，这个应用程序正在将数据发送到，在笔记本电脑或 Raspberry Pi 上运行的 Mosquitto 服务器（在任何你想要的地方）。在我的情况下，我正在笔记本电脑上运行 Mosquitto 服务器。

您可以从这里下载 Mosquitto 服务器。

这是一个简单的 tar 文件，只是我们使用以下命令来解压缩它，然后运行 Mosquitto 服务器。

```
1 tar -xvzf mosquitto-1.4.10.tar.gz
```

然后运行 `mosquitto` 启动 MQTT 服务器。

接受温度事件

该代码将部署在您的笔记本电脑上，我们将从 mosquitto 服务器获取主题 “Temperature Events” 的消息

所以，这里是接收温度事件的代码：

```
1 package com.knoldus
2
3 import com.typesafe.config.ConfigFactory
4 import org.eclipse.paho.client.mqttv3._
5 import org.eclipse.paho.client.mqttv3.persist.MemoryPersistence
6
7 /** * * MQTT subcriber * * @author Shivansh * @mail shiv4nsh@gmail.com * */
8 object MQTTSubscriber extends App {
9   val config = ConfigFactory.load()
10  val url = config.getString("mosquitto-server.url")
11  val port = config.getInt("mosquitto-server.port")
```

```
12   def subscribeToCommands() {
13 val brokerUrl = s"tcp://$url:$port"    val topic = "TemperatureEvent"
14     val persistence = new MemoryPersistence
15     val client = new MqttClient(brokerUrl, MqttClient.generateClientId,
           persistence) client
16     .connect client
17     .subscribe(topic)
18     val callback = new MqttCallBackImpl client
19     .setCallback(callback)
20   } subscribeToCommands
21 }
22
23 class MqttCallBackImpl extends MqttCallback {
24   override def messageArrived(topic: String, message: MqttMessage): Unit = {
25     println(s"Receiving Data | Topic : $topic | Message : $message")
26   }
27
28   override
29
30   def connectionLost(cause: Throwable): Unit = {
31     println(cause.toString)
32   }
33
34   override
35
36   def deliveryComplete(token: IMqttDeliveryToken): Unit = {
37     println(s"Delivered Message :${token.getMessage}")
38   }
39 }
```

这里我们只是收到数据

对于订阅者和发布者，我们在这里使用 Eclipse Paho 库。

您可以在我的 GitHub 代码库上轻松找到整个代码：scala-mqtt-client-rasberrypi-starter-kit

步骤 4：部署应用程序

部署应用程序的步骤如下：

1. 我们正在考虑这里的两个设备解决方案：`Dev_Laptop`（Subscriber）和 `Dev_RasberryPi`（Publisher）

2. 因此在你的 Dev_Laptop 上启动 Mosquitto

3. 在 mosquitto 的配置文件 application.conf 上配置服务的端口和 url。

4. 使用以下命令创建此项目的程序集 jar。

```
sbt assembly
```

5. 使用 scp 命令将 jar 复制到 Raspberry Pi。

```
scp raspi-mqtt-client.jar pi@:/home/pi/Projects/scala
```

6. 在 Dev_RaspberryPi 上使用下面的命令运行 Publisher

```
java -cp raspi-mqtt-client.jar com.knoldus.MQTTPublisher
```

这将启动发布者（publisher）将温度事件发送给代理。

7. 在 Dev_Laptop 上使用下面的命令运行 Subscriber

```
java -cp raspi-mqtt-client.jar com.knoldus.MQTTSubscriber
```

Hurray!

现在，您已经使用 Scala 制作了第一个基础的 IoT 应用程序，可以将 Raspberry Pi 的温度发送给代理，用户可以订阅这些事件。

注意：请记住，订户也可以是 Raspberry Pi; 完全取决于您的架构。

你也可以从这里找到使用 Lightbend 激活器的代码。

所以在未来的博客中，我们将在 Raspberry Pi 上部署相同的应用程序，并尝试使用 Apache Spark 获取 Stream。

原文链接：https://www.wandianshenme.com/play/scala-iot-raspberry-pi-mosquitto-build-iot-proje