**Application Note for the RTU Program Development**

**Lu Zhen**

**20/05/2015**

The RTU program is Java Based program and is featured in OS-Cross design. To develop the RTU program, the first requirement is the java software development platform. At present, NetBeans, Eclipse and IntelliJ IDEA are popular in use. All of them have the version free in charge. For the RTU program development, both of NetBeans and Eclipse are employed as they are especially suitable for different OS applications.

The installation of development platforms, i.e. NetBeans and Eclipse, is quite easy as long as following the instructions in web-sites below:

NetBeans: <http://www.oracle.com/technetwork/articles/javase/jdk-netbeans-jsp-142931.html>

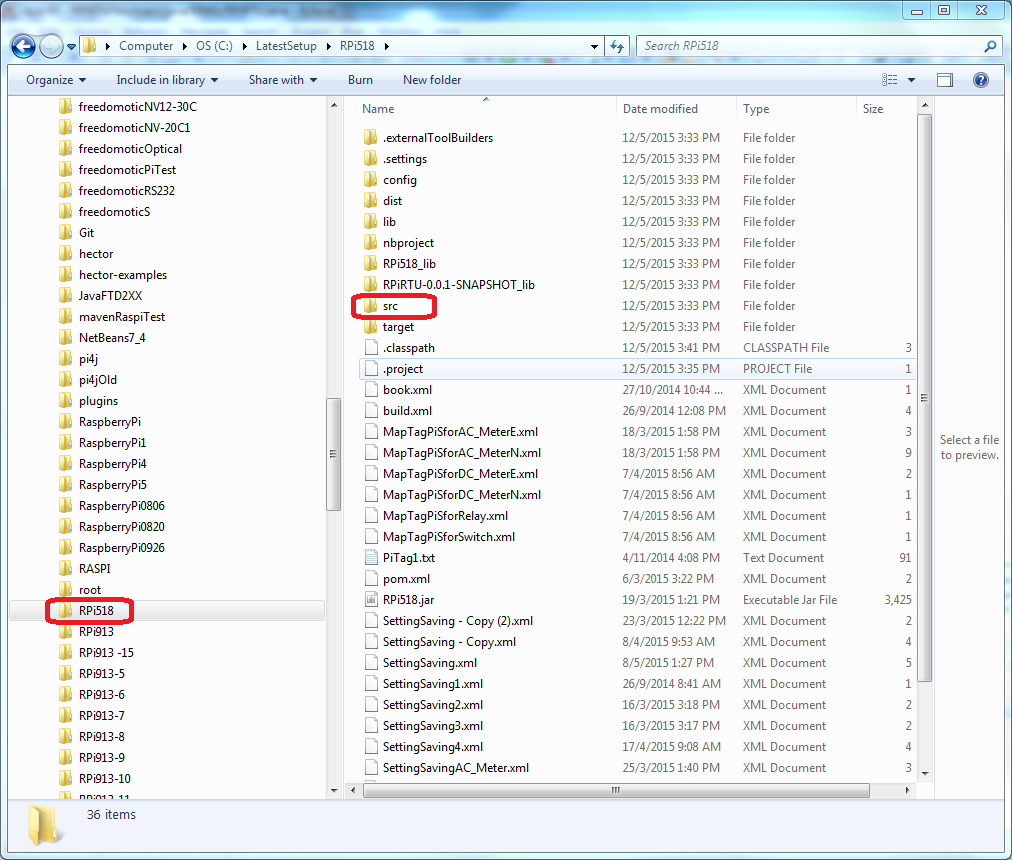
Eclipse: <https://eclipse.org/downloads/>

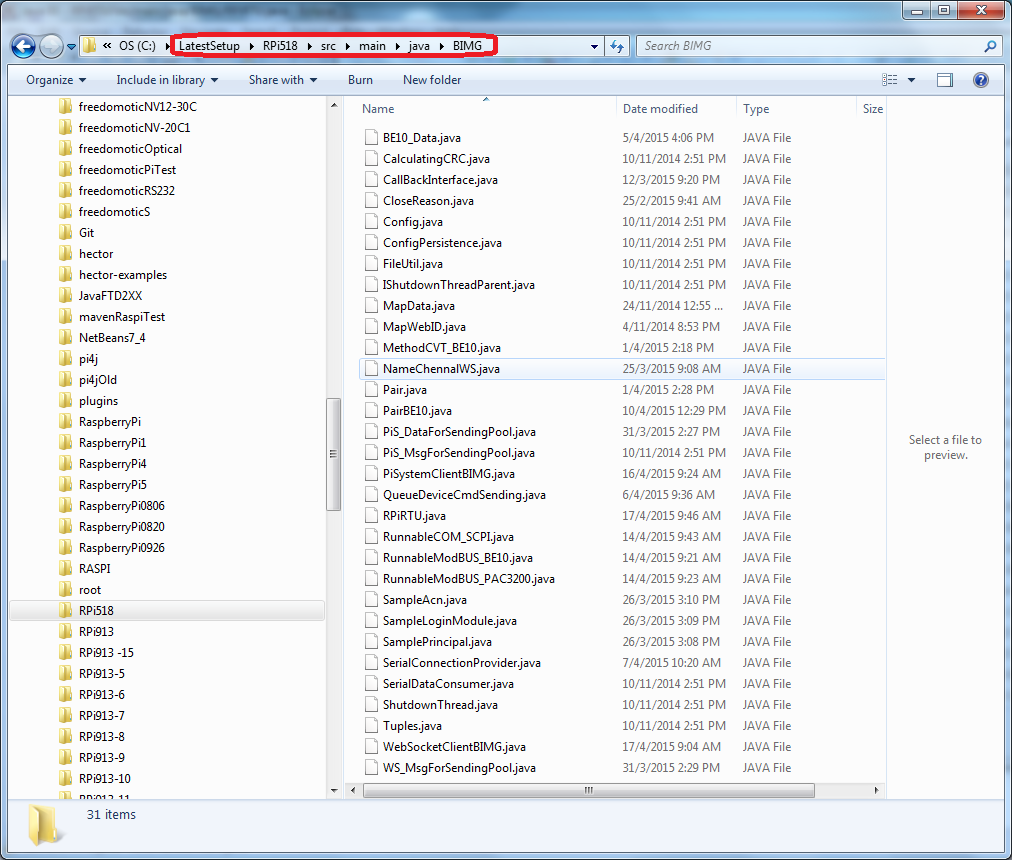
However, how to establish the development environment on the platforms and how to use those in software developing may need a long learning period to master for newer in Java.

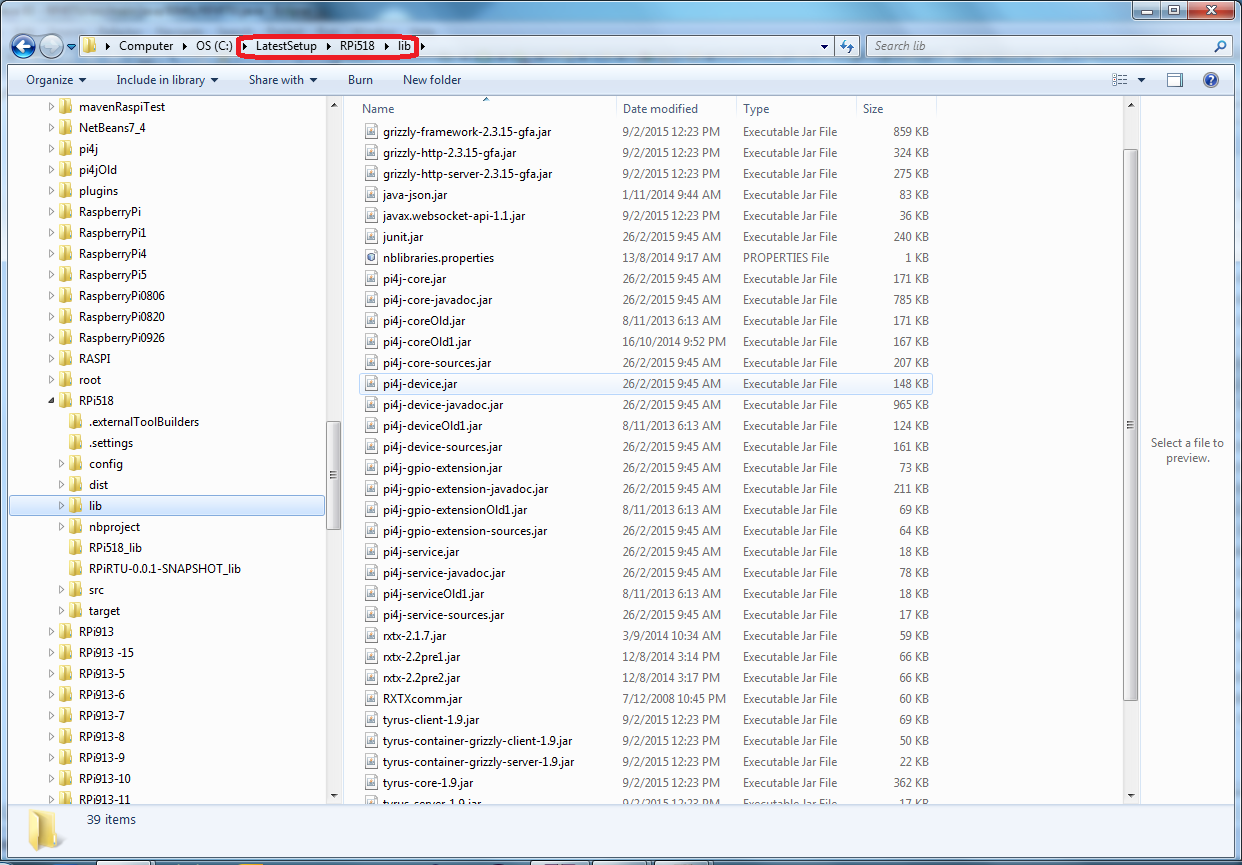
To avoid mentioning these development platform issues, one of the easiest ways is to use the existing development environments that have been established before. On the RTU program development PC (SBB-SGH242RHSF), both development environments of NetBeans and Eclipse have been well established and actually, the latest version of the RTU program is produced on these development environments. Therefore, to facilitate the software development’s handover, all of source code of the RTU program is installed, compiled and error-free proved as well as executable-file produced by reference the existing development environments on the development PC (SBB-SGH242RHSF).

### File Map for the Developing Environment of RTU Program

Although the RTU program can be produced from the same source code, the file-map for the developing environment is set up in different root-fodder location and different file map structure in the root-fodders. In Eclipse platform, the root-fodder location and file map structure are shown in below maps.

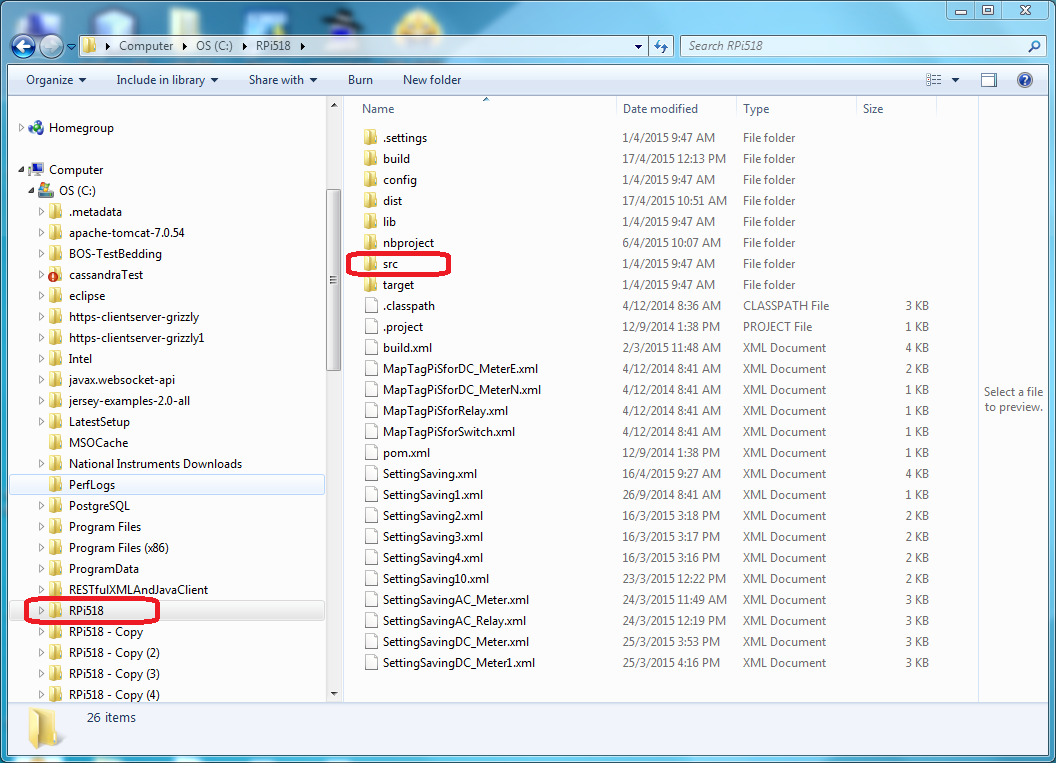
The root-fodder >>>> 

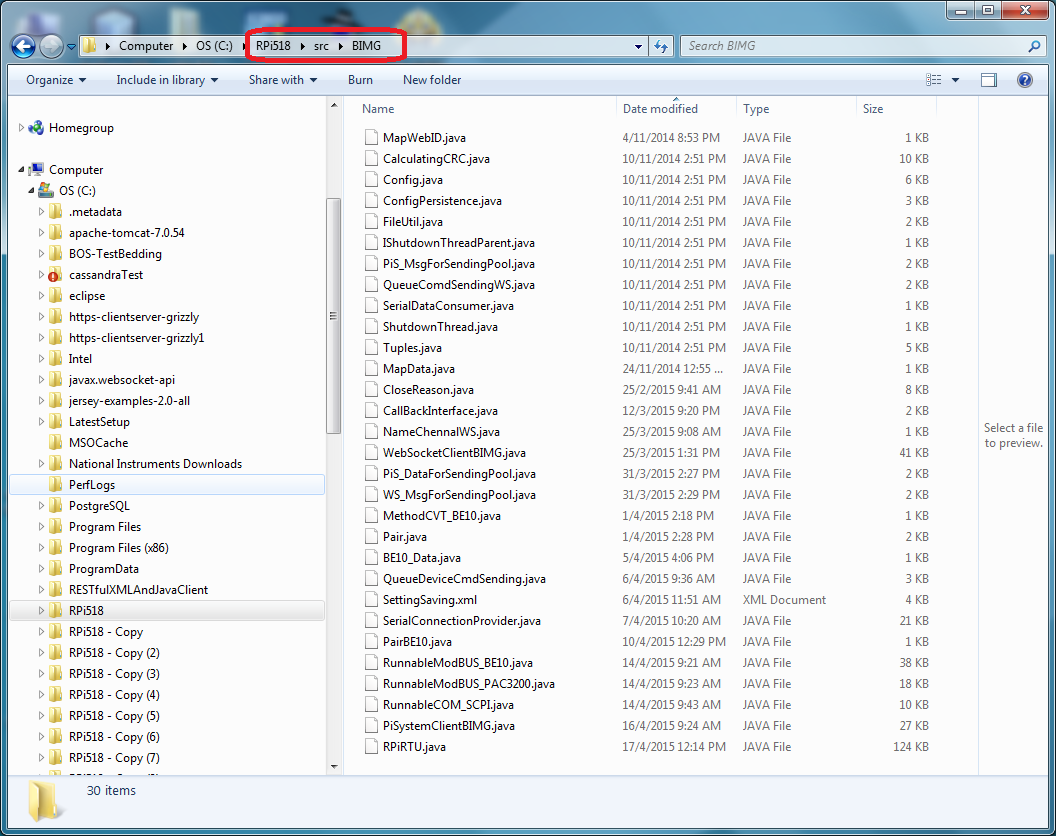
The source-code-fodder >>>>

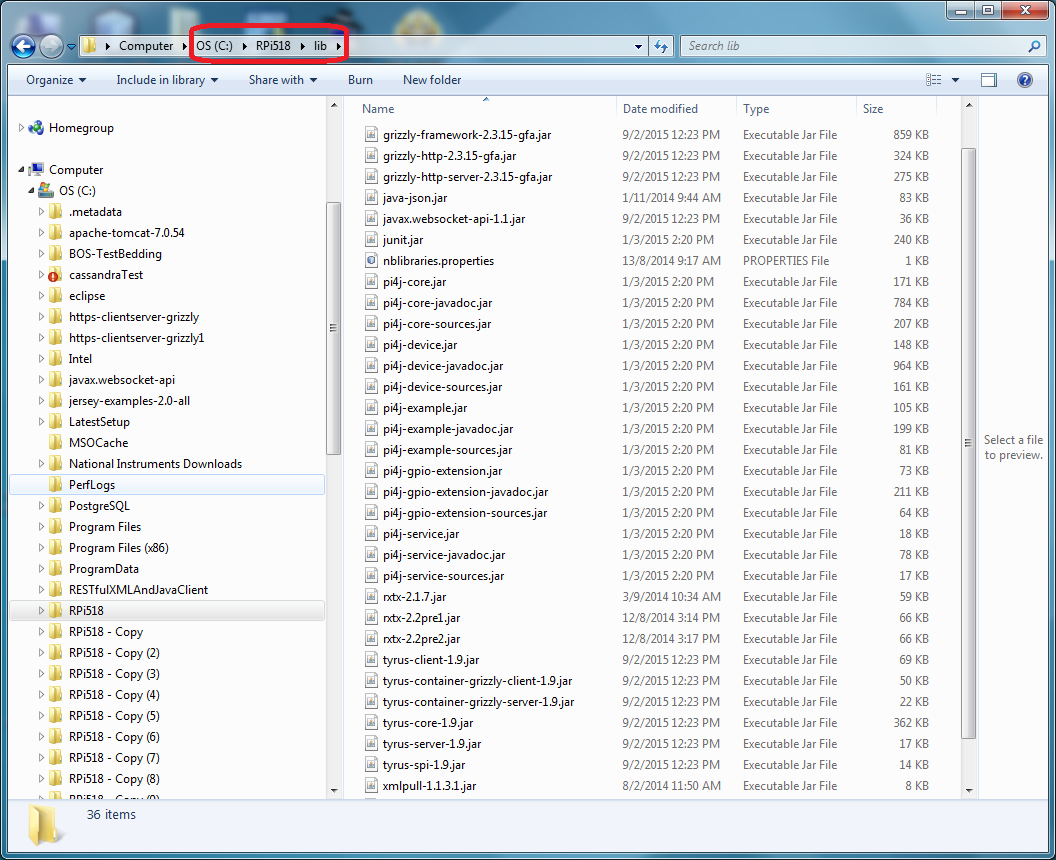
And the library fodder >>>> 

Initially, Eclipse development platform is employed on the RTU program developing on account of its friendly IDE interface and technical support. However, there is no convenient way to develop the the RTU program on Raspberry Pi which is low-end computer in Linux OS. When the remote development function released (in NetBeans 8.0) by Oracle, NetBeans development platform is also employed on the RTU program developing to take the advantage of remote developing Java program on Raspberry Pi. As the result, the Eclipse development platform is for the RTU program developing in Windows OS, while the NetBeans development platform is for the RTU program developing in Linux (Raspberry Pi).

So for the convenience of NetBeans development platform, the root-fodder is duplicated to a new place, and its location and file map structure are shown in below maps.

The root-fodder >>>> 

The source-code-fodder >>>>

And the library fodder >>>> 

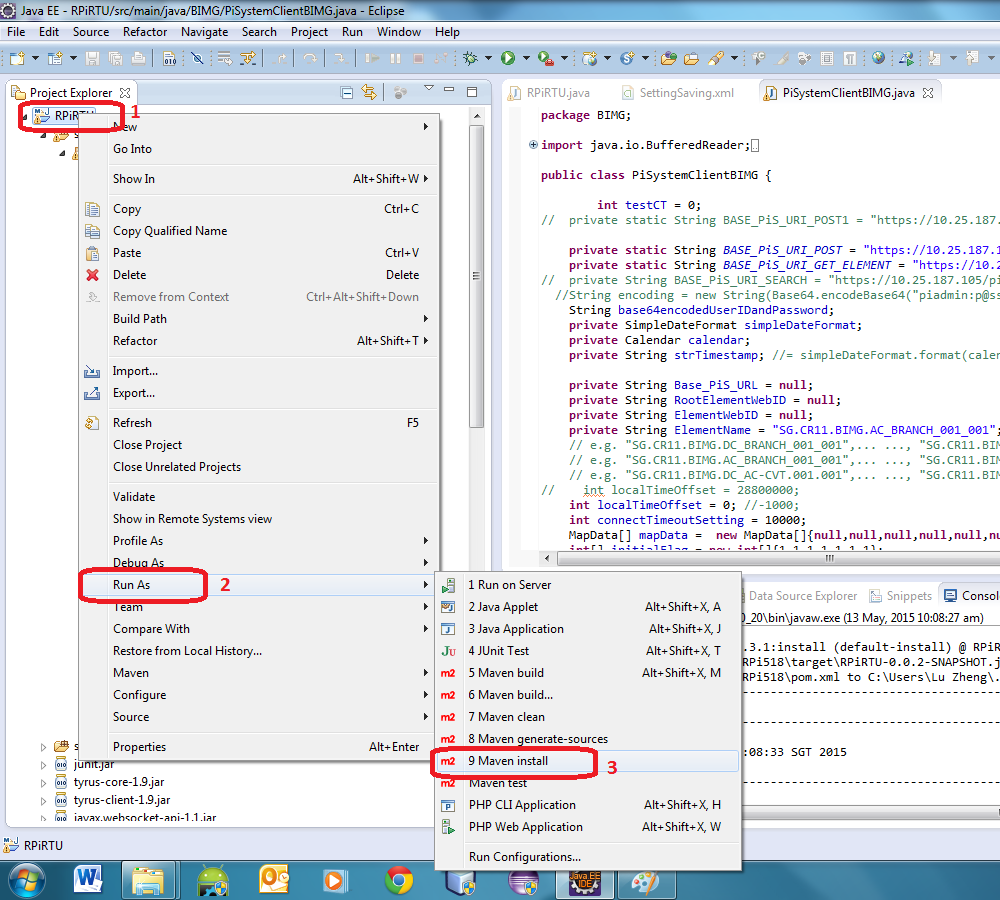
### The Procedures of Making executable jar File of RTU Program

To make the executable jar file from source code is a simple process as well as the developing environment has properly setup and the source code is free of error.

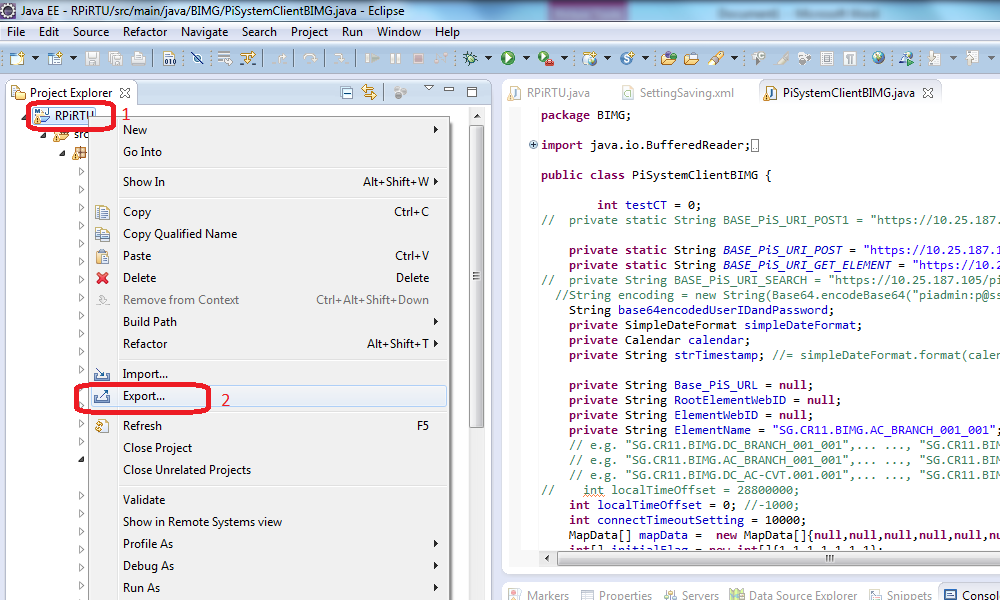
1. **The Procedure on Eclipse development platform**

In Eclipse, the executable jar file usually needs to export to a target fodder after the RTU program source code being successfully compiled. To do so, please follow steps below:

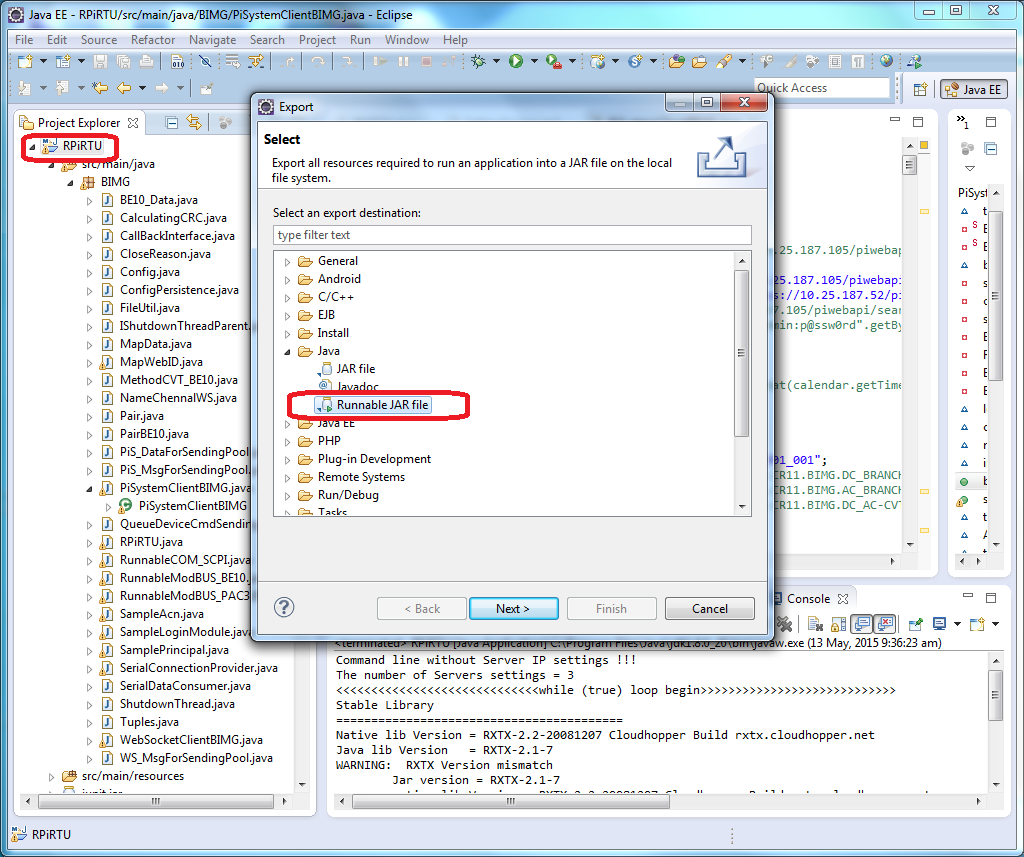
***Step 1:*** Right-kick the project “RPiRTU” in Project Explorer. Then a popup-menu appears. Following the 1,2&3, left-kick the menu items circled in red. This step will invoke a complete compiling of RTU program source code. (Only when the compiling result is error-free, go to next step)



***Step 2:*** Right-kick the project “RPiRTU” in Project Explorer. And then left-kick the menu item “Export”.

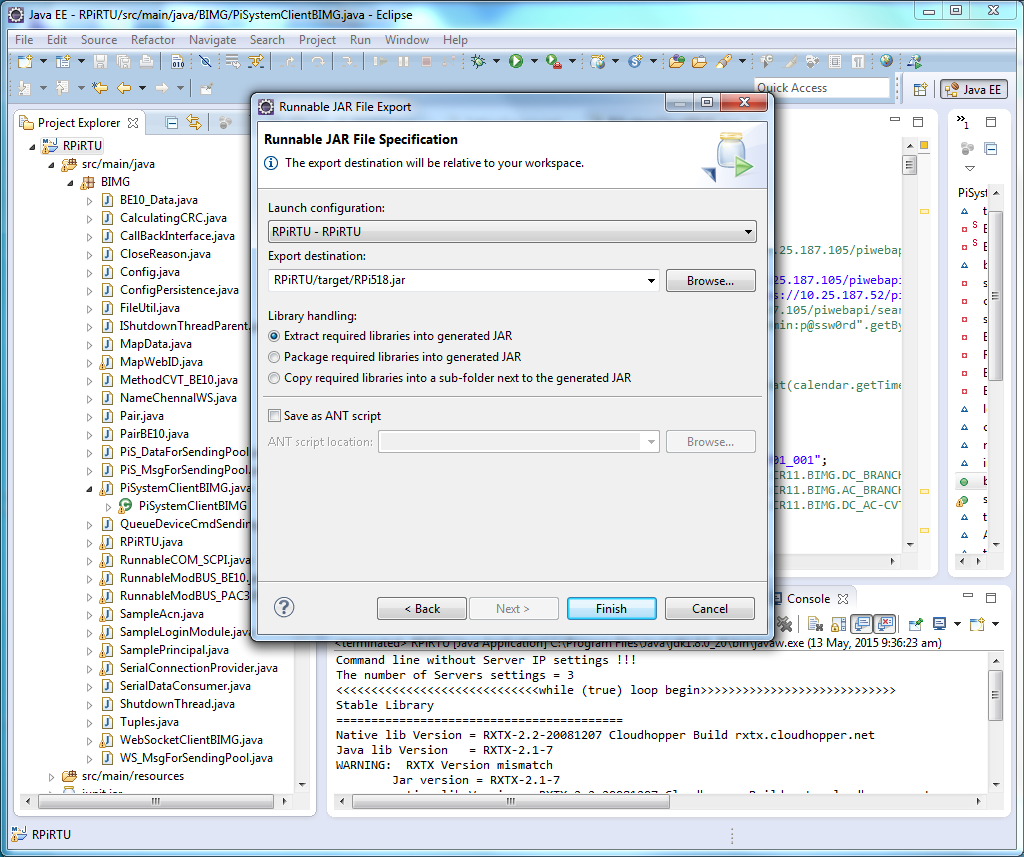


Then in the following dialog, select the item “Runnable JAR file” and kick Next.



And setup the dialog as shown below and then kick Finish. This step will make an executable jar file “RPi518.jar” of RTU program to the distinction fodder “c:/LatastSetup/Rpi518/target”.

**Note: the RTU program development PC (SBB-SGH242RHSF) may not be the distinction machine of the visual RTU hosted. Please copy this executable jar file “RPi518.jar” to the distinction machine thereafter.**

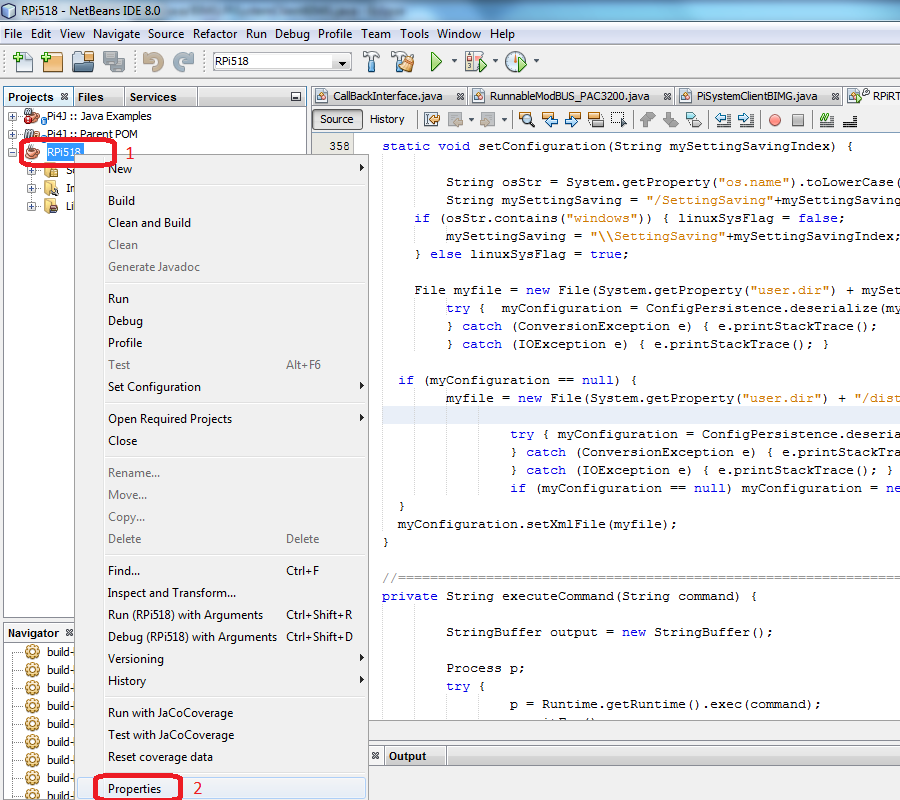


1. **The Procedure on NetBeans development platform**

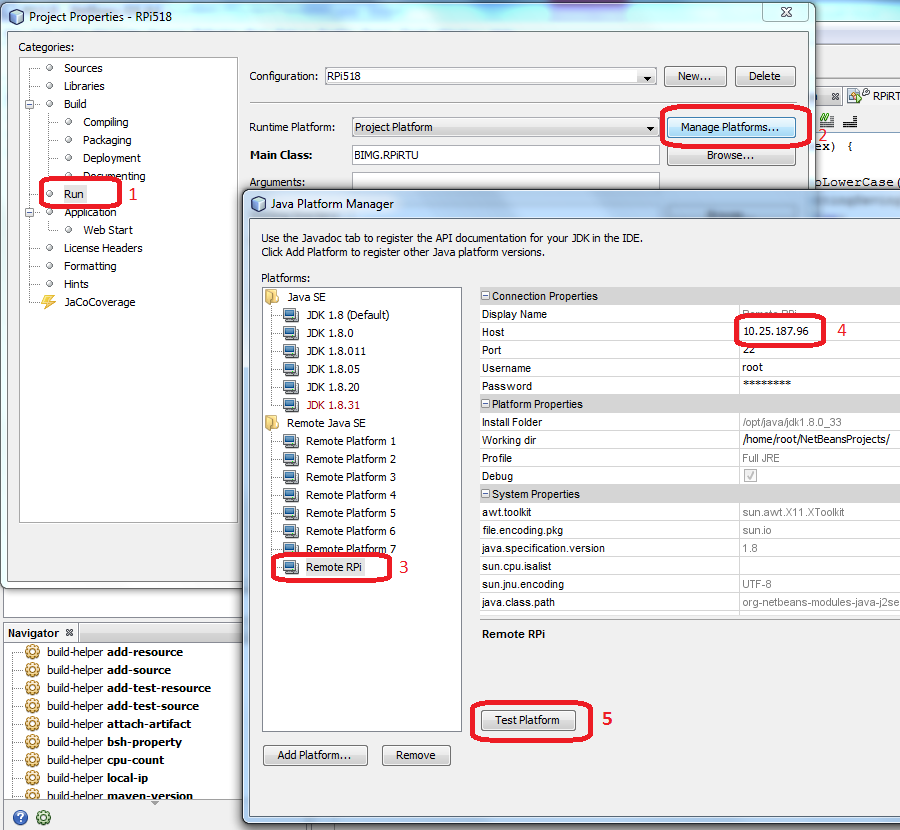
In NetBeans, the executable jar file of RTU program is targetted on Raspberry Pi since NetBeans 8 supports the remote development mode. Therefore, a Raspberry Pi B+ with proper development environment settings— having installed the proper Disk-Images file on MicroSDHC, having the GUI (monitor, keyboard and mouse) and its Ethernet port (RJ45) should link to the Local Area Network (LAN) which should belong to the same domain with the development machine. Please follow steps below:

***Step 1:*** Power-on the Raspberry Pi. Then change the system file “profile” (please reference the document: The RTU Program Environment Setup and Installation Procedure) to setup-environment mode. And then reboot the Raspberry Pi.

***Step 2:*** check the Raspberry Pi host IP address using the Linux command “ifconfig”. For example, the IP address = “*10.25.187.86*”. Then right-kick the project “RPiRTU” on Projects of the NetBeans 8.0, and then left-kick the menu item “properties”. A dialog for project properties setting will appear (please see the following two pictures).

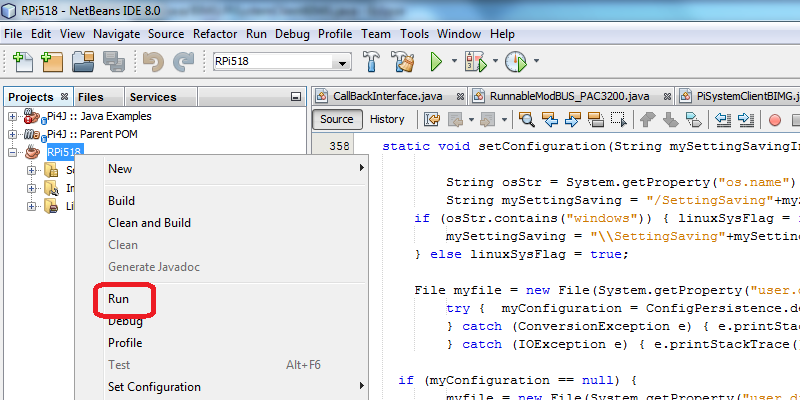


Following the 1, 2, 3&4, left-kick the items circled in red. And then change Host IP to “*10.25.187.86*”.



This step can accord NetBeans Remote Host IP Setting with the developing Raspberry Pi’s IP. If the settings need to verify, just kick “Test Platform” button (red circle 5) to start a checking process.

***Step 3:*** Right-kick the project “RPiRTU” on Projects of the NetBeans 8.0, and then left-kick the menu item “Run”. This step will invoke a remote installation of RTU program executable jar file onto the developing Raspberry Pi, which copies all of the files in sub-fodder “dist” to the distinction (i.e. Raspberry Pi). After that, NetBeans begin to run RTU program in a remote mode. I don’t recommend using this remote mode to debug the RTU program as the RTU program’s real running performance may be different from the performance in this remote mode.



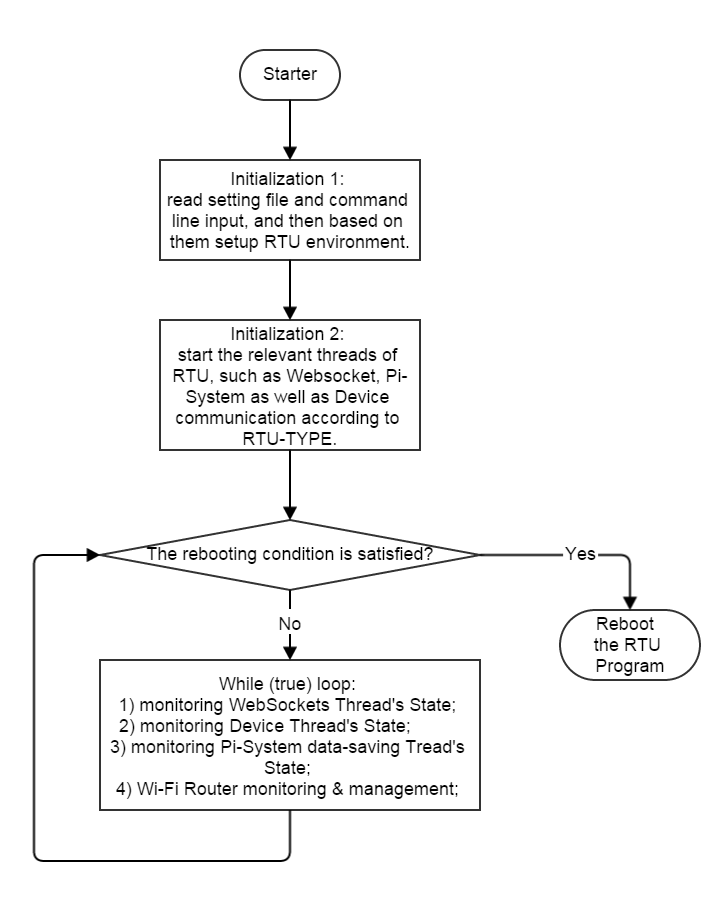
***Step 4:*** Change back the system file “profile” (please reference the document: The RTU Program Environment Setup and Installation Procedure) to running-environment mode and double check all of the settings files on the developing Raspberry Pi. Then reboot the Raspberry Pi.

***Step 5:*** After verifying the new RTU Program’s operation in various conditions, the MicroSDHC is ready to move to the real running location (the specific point of BIMG).

### Introduction of the structure of the RTU Program

There are totally 31 files in the source code fodder “BIMG”. Some of them are still under-developing. (please see the list table below)

|  |  |
| --- | --- |
| File Name | Description |
| BE10\_Data.java\* | Data class for BE10. (under-developing) |
| CalculatingCRC.java | Modbus CRC calculating class |
| CallBackInterface.java | Call back interface for multi-threads’ data/commands exchange |
| CloseReason.java | WebSocket’s sub-class |
| Config.java | Class for setting file reading |
| ConfigPersistence.java | Class for setting file reading |
| FileUtil.java | Brief setting procedure describing |
| IShutdownThreadParent.java |  |
| MapData.java | Class for Pi-System data-saving |
| MapWebID.java | Class for Pi-System data-saving |
| MethodCVT\_BE10.java | Enum using to BE10 |
| NameChennalWS.java | Enum using to Websocket communication |
| Pair.java | Specific Pair class using in RTU program |
| PairBE10.java\* | Specific Pair class using for BE10 () |
| PiS\_DataForSendingPool.java | Data sending pool class for Pi-System |
| PiS\_MsgForSendingPool.java |  |
| PiSystemClientBIMG.java | Data sending thread class for Pi-System |
| QueueDeviceCmdSending.java | Data sending pool class for device |
| RPiRTU.java | Main class |
| RunnableCOM\_SCPI.java | Data/commands exchange thread class for SCPI equipment. |
| RunnableModBUS\_BE10.java\* | Data/commands exchange thread class for BE10 (under-developing). |
| RunnableModBUS\_PAC3200.java | Data/commands exchange thread class for PAC3200 |
| SampleAcn.java\* | Class for Kerberos (under-developing). |
| SampleLoginModule.java\* | Class for Kerberos (under-developing). |
| SamplePrincipal.java\* | Class for Kerberos (under-developing). |
| SerialConnectionProvider.java | Class for serial port communication. |
| SerialDataConsumer.java | Class for serial port communication. |
| ShutdownThread.java | Shutdown class |
| Tuples.java |  |
| WebSocketClientBIMG.java | Data/commands exchange thread class for Websocket |
| WS\_MsgForSendingPool.java | Data sending pool class for WebSocket |

**The Flow-Charts of the RTU Program**

Using high-level language, such as Java, in BIMG design is a good-try. However, the traditional one-thread programming in Java can’t fulfil the high-responding requirement of real-time control. To overcome this problem, this RTU Program employs the multi-threads programming, i.e. there are at least three threads running simultaneously in the RTU Program time of time.

The multi-threads programming is good at real-time control, although it introduces the complication in data-exchange between the various threads, which requires high-responding-speed. As a result, it is hard to debug in the RTU Program developing process. Actually, I just realized how difficult and time-consuming to draw such kind of the RTU Program’s Flowcharts when I need to draw some of the flowcharts.

The chart on the right is the brief flowchart for the RTU Program main thread.

Inside the main thread’s While (true) loop, several monitoring tasks are implemented as in the following flowcharts:

I have deployed the new version of RTU Program in SinBerBEST BIMG Test-Bedding this week. The new version program has realized the functions that are described on my previous work summary report sent to you.

As the RTU designer, I am proud of my work having done and therefore I am willing to take the responsibility of any problem occurred on the new version of RTU Program (limited in the existing functions of RTU).

In the previous RTU design road-map, it will upgrade the Pi-System data-saving function to buck-data-saving mode after OSIsoft’s PI Web API 2015 R2 has been released in 30th June 15’, which would enhance the Pi-System data-saving performance with better speed. Nevertheless, I have to leave this work behind as I am leaving SinBerBEST in May15’. I hope to finish the upgrade work myself and give the RTU design a good-end; however, I am not the person to make this decision. I just want to say that the upgrade’s work-load is about 2~3 weeks for me when PI Web API 2015 R2 is available.

Thank you.