Developing RePhone applications in Eclipse on Linux

Tested on Ubuntu 15.10 & 16.04 64-bit

LoBo 04/2016

You can use my GitHub repository (https://github.com/loboris/RePhone_on_Linux) which contains RePhone development environment prepared for Eclipse on Ubuntu Linux and some documentations and tools.

Clone GitHub repository
`git clone https://github.com/loboris/RePhone_on_Linux.git`

or download zip file and unpack to some directory
https://github.com/loboris/RePhone_on_Linux/archive/master.zip

Prepare your Rephone for development

This is the only step which has to be done under Windows

Flash the RePhone firmware using FirmwareUpdater.exe included in FirmwareUpdate directory.
If RePhone firmware is not already selected, click on Other and select `SEEED02A_DEMO_BB.cfg` **in** `FirmwareUpdate/firmware/LinkIt_Device/RePhone/W15.19.p2-uart` **directory**

Flashing can be tricky, but eventually you will succeed and you RePhone will be ready.

The purpose of that flashing is to disable the MT2502 debug output and prepare the Rephone USB ports to be used for accessing the application and AT Commands interfaces.

If your Rephone is already flashed with
`SEEED02A_DEMO_BOOTLOADER_V005_MT2502_MAUI_11CW1418SP5_W15_29.bin` **firmware, you can skip this step.**

Prepare your Linux (Ubuntu) distribution for Rephone

To enable RePhone mass storage mode you have to modify `/lib/udev/rules.d/40-usb_modeswitch.rules`
Edit the file (as root), find and comment the line:

```
ATTR{idVendor}=="0e8d", ATTR{idProduct}=="0002", RUN+="usb_modeswitch '%b/%k'"
```

so that it looks like that:

```
#ATTR{idVendor}=="0e8d", ATTR{idProduct}=="0002", RUN+="usb_modeswitch '%b/%k'"
```

To prevent Ubuntu accessing RePhone modem interface create the file (as root):
`/etc/udev/rules.d/98-rephone.rules`

and place in it the line:

```
ATTRS{idVendor}=="0e8d", ATTRS{idProduct}=="0023", ENV{ID_MM_DEVICE_IGNORE}="1"
```

REBOOT

When Rephone is connected in mass storage mode, it's internal drive will appear as /dev/sdX (5MB drive), where X will be different letter on your system.
The may appear in your File manager as MEDIATEK FLASH DISK.
You can mount it and transfer the files.

When RePhone is connected in app mode, two serial devices will appear:

`/dev/ttyACM0`	**the application usb serial port interface**
`/dev/ttyACM1`	**MT2502 modem interface (AT Commands)**

The Mobile broadband connection may appear in Network Manager, if it does, DISABLE IT!, otherwise Ubuntu will try to communicate to MT2502 modem interface.
Once disabled it will stay disabled, no need to do it every time.

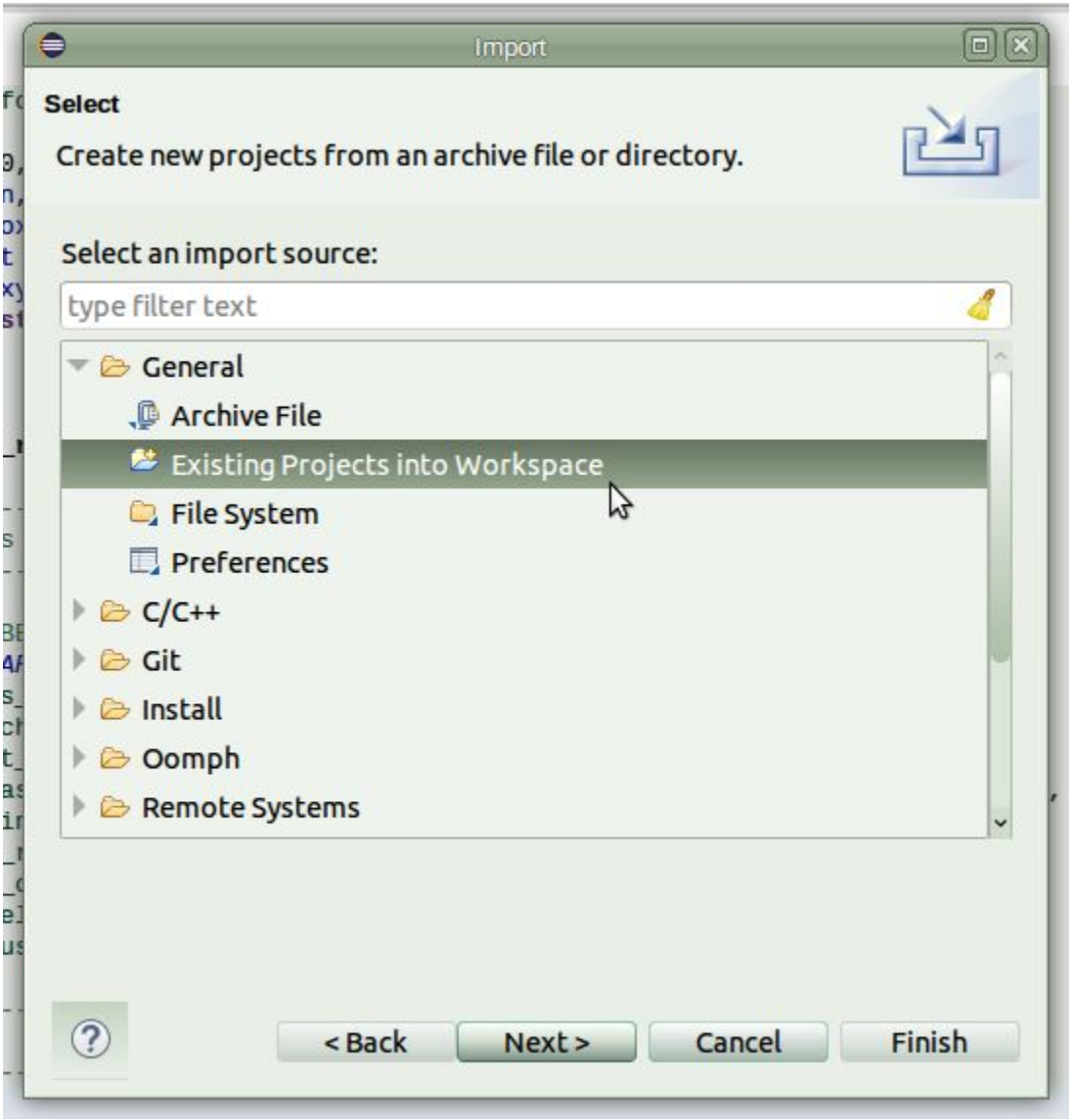
Prepare development environment for Rephone

Install necessary packages:
`build-essentials`, `gcc-arm-none-eabi`
You can also install `minicom` or other terminal software to be able to communicate with your RePhone application.

Download the latest Eclipse installer from https://www.eclipse.org/downloads
Unpack and run the Eclipse installer (`eclipse-inst`), select **Eclipse IDE for C/C++ Developers**

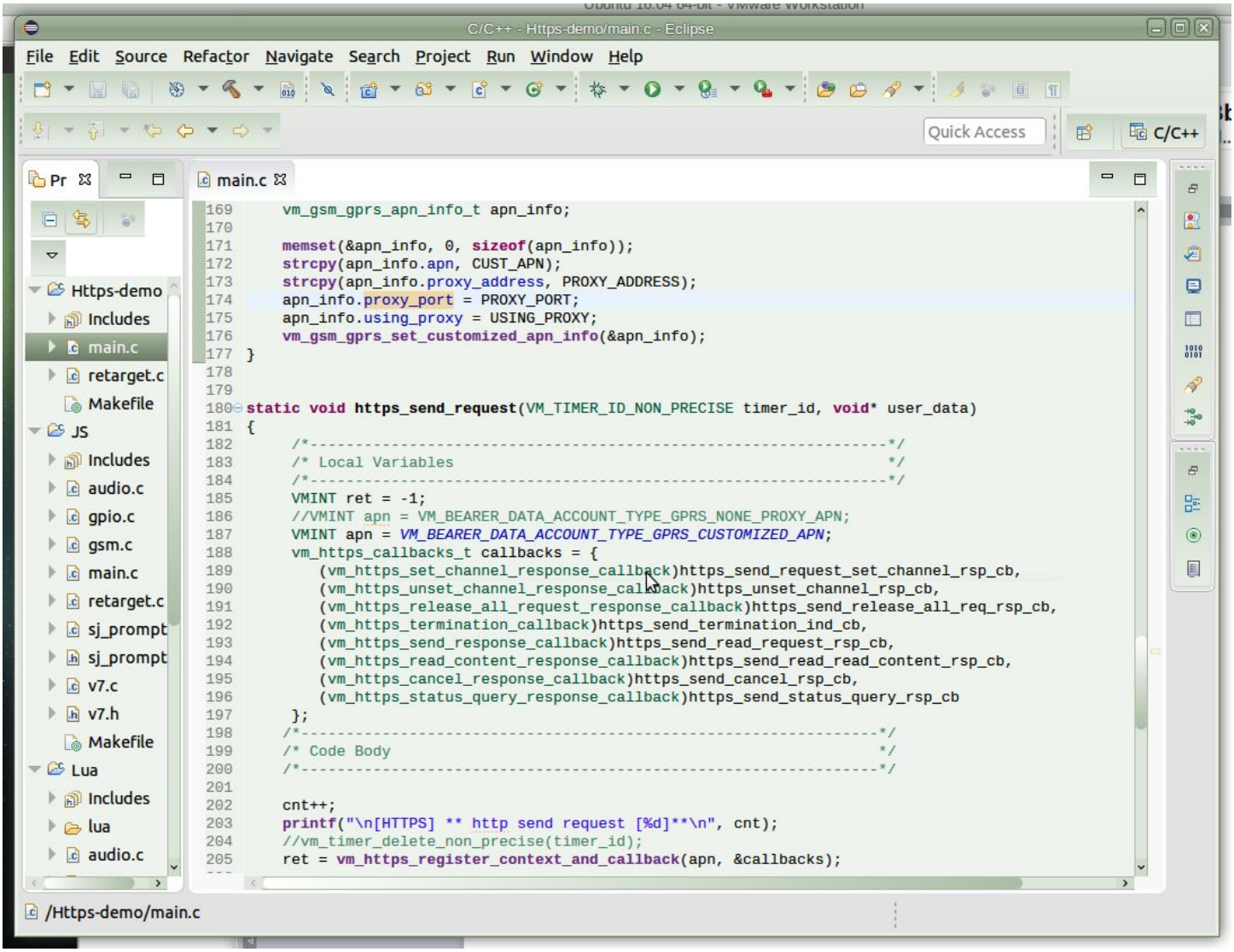
After installation is finished, you can lunch eclipse and select your workspace.

To import the RePhone projects downloaded from my GitHub repository, select
File->Import->General->Existing Projects into Workspace



Click Next, then select root directory of the downloaded repository and click Finish.

You will have 3 projects ready for build.




```
169     vm_gsm_gprs_apn_info_t apn_info;
170
171     memset(&apn_info, 0, sizeof(apn_info));
172     strcpy(apn_info.apn, CUST_APN);
173     strcpy(apn_info.proxy_address, PROXY_ADDRESS);
174     apn_info.proxy_port = PROXY_PORT;
175     apn_info.using_proxy = USING_PROXY;
176     vm_gsm_gprs_set_customized_apn_info(&apn_info);
177 }
178
179
180 static void https_send_request(VM_TIMER_ID_NON_PRECISE timer_id, void* user_data)
181 {
182      /*----------------------------------------------------------------*/
183      /* Local Variables                                                */
184      /*----------------------------------------------------------------*/
185      VMINT ret = -1;
186      //VMINT apn = VM_BEARER_DATA_ACCOUNT_TYPE_GPRS_NONE_PROXY_APN;
187      VMINT apn = VM_BEARER_DATA_ACCOUNT_TYPE_GPRS_CUSTOMIZED_APN;
188      vm_https_callbacks_t callbacks = {
189         (vm_https_set_channel_response_callback)https_send_request_set_channel_rsp_cb,
190         (vm_https_unset_channel_response_callback)https_unset_channel_rsp_cb,
191         (vm_https_release_all_request_response_callback)https_send_release_all_req_rsp_cb,
192         (vm_https_termination_callback)https_send_termination_ind_cb,
193         (vm_https_send_response_callback)https_send_read_request_rsp_cb,
194         (vm_https_read_content_response_callback)https_send_read_read_content_rsp_cb,
195         (vm_https_cancel_response_callback)https_send_cancel_rsp_cb,
196         (vm_https_status_query_response_callback)https_send_status_query_rsp_cb
197      };
198     /*----------------------------------------------------------------*/
199     /* Code Body                                                      */
200     /*----------------------------------------------------------------*/
201
202     cnt++;
203     printf("\n[HTTPS] ** http send request [%d]**\n", cnt);
204     //vm_timer_delete_non_precise(timer_id);
205     ret = vm_https_register_context_and_callback(apn, &callbacks);
```

If you are creating your own project, or if you are on different Linux distro, check if the right path to gcc-arm-none-eabi compiler is set in the `Makefile`.
You may also need to place the symbolic link to you `make` into tools directory.

After building the project, just copy the .vxp file to RePhone in mass storage mode.
Don't forget to update "`autostart.txt`" with the new .vxp file name.