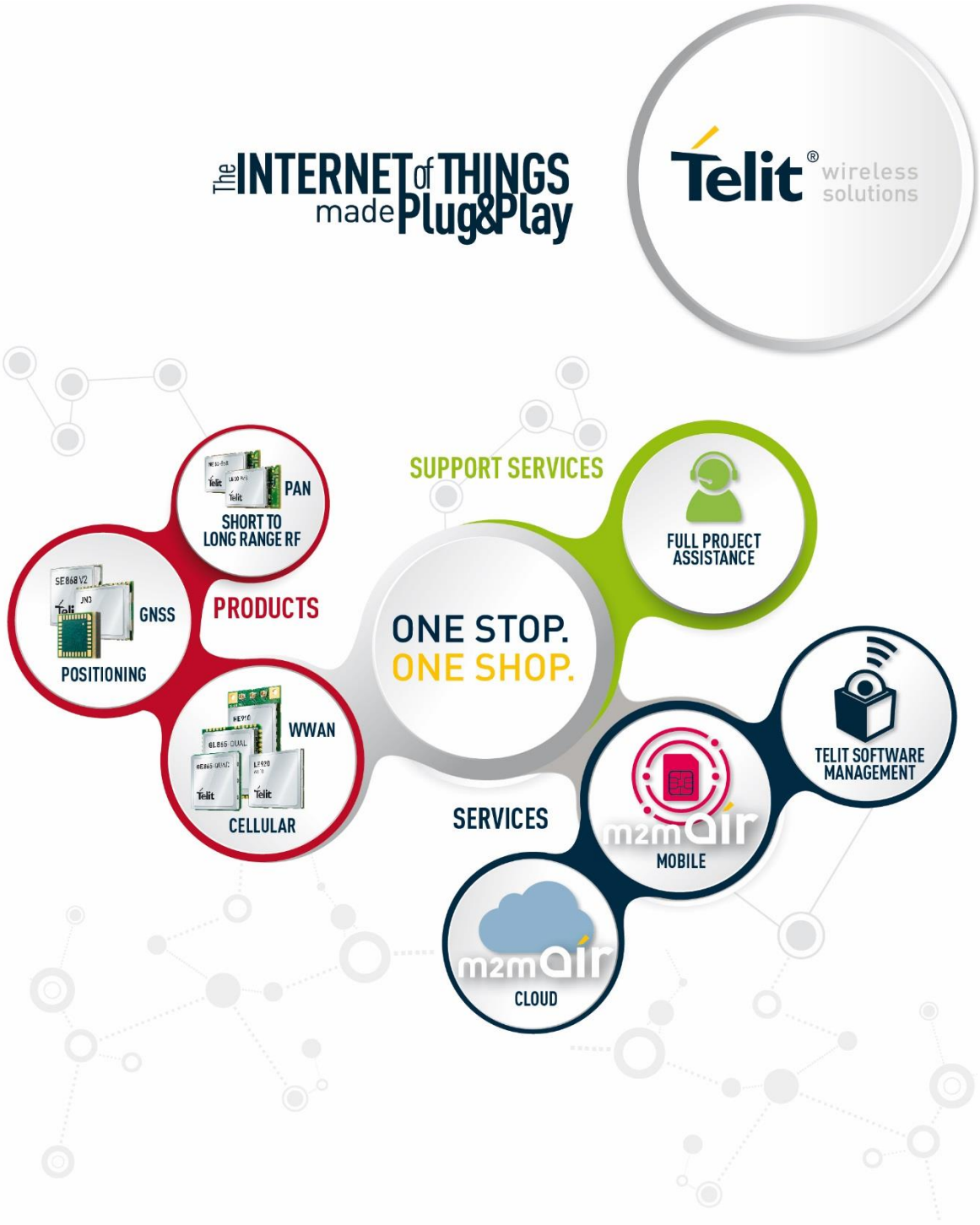


APPZONE LINUX USER GUIDE

APPLICABILITY TABLE

PRODUCTS

- LE910 SERIES
- LE920 AUTO SERIES

FAMILY	SW VERSIONS
LE910	17.01.XXX
LE920	17.01.XXX

SPECIFICATIONS SUBJECT TO CHANGE WITHOUT NOTICE

LEGAL NOTICE

These Guidelines are general guidelines pertaining to the installation and use of Telit's Integrated Application Development Environment ("IDE"). Telit and its agents, licensors and affiliated companies make no representation as to the suitability of these Guidelines for your particular needs and Telit disclaims any and all warranties, expressed or implied, relating to the contents of this document. Furthermore, this document shall not be construed as providing any representation or warranty with respect to any Telit Product whether referenced herein or not.

It is possible that this document may contain references to, or information about Telit products, services and programs, that are not available in your region. Such references or information shall not be construed to mean that Telit intends to make available such products, services and programs in your area.

HIGH RISK USES

The IDE is not intended for the design of software for use in hazardous environments or environments requiring fail-safe controls, including the operation of nuclear facilities, aircraft navigation or aircraft communication systems, air traffic control, life support, or weapons systems ("High Risk Activities"). Without derogation, Telit, its licensors and its supplier(s) specifically disclaim any expressed or implied warranties with respect to the use of the IDE for development of software for such High Risk Activities and specifically disclaim all liability with respect to such use.

TRADEMARKS

The names Telit, deviceWISE, deviceWISE by Telit, secureWISE, secureWISE by Telit, Telit IoT MODULES, Telit IoT PORTAL, Telit IoT PLATFORM, Telit IoT PLATFORMS, Telit IoT CONNECTIVITY, Telit IoT SERVICES and their associated logos are trademarks of Telit Communications PLC, its subsidiaries or affiliates in the United States and/or other countries. Other company or product names are the trademarks of their respective owners. All trademark rights are reserved.

THIRD PARTY RIGHTS

The IDE may contain or may be provided in conjunction with third party software (the "third party software"), including some or all of those detailed in the NOTICES file provided to you during installation of the IDE. You acknowledge that not all third party software detailed in the NOTICES file are necessarily used or provided in the particular version of the IDE you install and use. Refer to the IDE's EULA and the NOTICES file for licensing information pertaining to the third party software.

CONTENTS

1. INTRODUCTION

1.1. Scope

This document describes how to install and use AppZone Linux. The API required to develop applications, also referred to as the Mobile Connection Manager (MCM) API. The MCM API allows a subset of services provided by Qualcomm's MDM chipsets to be accessible to Linux applications.

The development environment is based on Eclipse IDE. For complete documentation of the Eclipse IDE, refer to Eclipse documentation at http://www.eclipse.org/.

1.2. Audience

This document is intended for software developers who will be using the MCM API.

This document provides the public interfaces necessary to use the features provided by the MCM API.

A functional overview and information on leveraging the interface functionality are also provided. This document assumes that the user is familiar with Linux programming.

1.3. Contact information and support

For general contact, technical support services, technical questions and report documentation errors contact Telit Technical Support at:

TS-EMEA@telit.com

TS-AMERICAS@telit.com

TS-APAC@telit.com

Alternatively, use:

http://www.telit.com/support

For detailed information about where you can buy the Telit modules or for recommendations on accessories and components visit:

http://www.telit.com

Our aim is to make this guide as helpful as possible. Keep us informed of your comments and suggestions for improvements.

Telit appreciates feedback from the users of our information.

1.4. Text conventions

Caution or Warning – Alerts the user to important points about integrating the module, if these points are not followed, the module and end user equipment may fail or malfunction.

Tip or Information – Provides advice and suggestions that may be useful when integrating the module.

All dates are in ISO 8601 format, i.e. YYYY-MM-DD.

1.5. Related documents

[1] ioe_cm_api_interface_spec, 80-NJ803-1 C

[2] AppZone Linux API Reference Guide 80496ST10725A

2. FUNCTIONAL OVERVIEW

AppZone Linux is an integrated development tool based on Eclipse IDE that can run on both Windows and Linux. This environment provides a set of template files that form the skeleton for creating applications that can run on Telit modules.

AppZone Linux is a high-level connectivity framework that makes available a rich set of functions that can be called by a client application running in the Linux user space. The API offers multi-client architecture in the Linux user space, allowing multiple processes to concurrently leverage resources exported via different interfaces. A client can be any Linux user space process, such as an application or daemon.

The APIs are message based. To perform a desired command (for example, to establish a data connection), a client sets applicable parameters within a message and invokes the function API to send the message. The response to the command is sent back to the client after the command is processed. The client can also register to receive indications via a callback mechanism. When invoked, the callback receives an indication along with a payload. The payload contains the description of the event received.

For more information on the APIs, see the *AppZone Linux API Reference Guide*.

Figure 2-2 illustrates how to send an MCM_VOICE_DIAL_REQ_V01 message, both synchronously and asynchronously

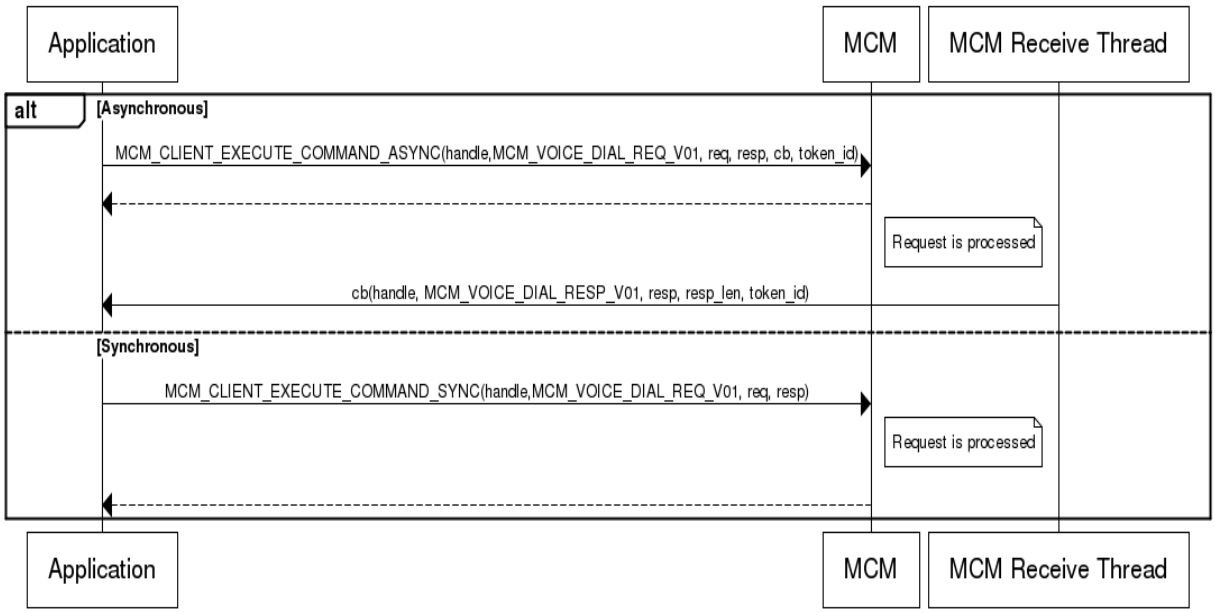


Figure 2-2 Asynchronous and synchronous voice dialing call flow

3. INSTALL APPZONE LINUX

3.1. Hardware and software requirements

To install AppZone Linux you must:

- Have administrator permissions
- Know which module you will be using

The following table lists the requirements for installing AppZone Linux:

OS	Windows 7 (32bit or 64bit), Windows 10 Linux – Ubuntu LTS 12.04, 14.04, 16.04 versions (as listed in https://wiki.ubuntu.com/Releases).
RAM	1 Gb
Free disk space	750 MB free disk space

3.2. Install on Windows

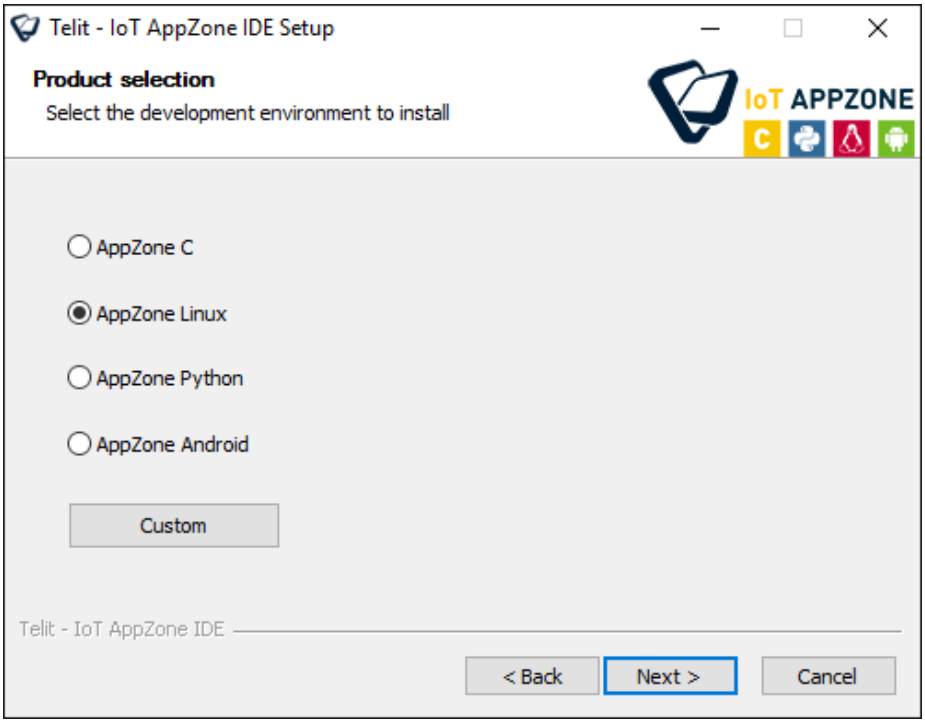
To install AppZone Linux on Windows:

1. Access the Telit portal: http://www.telit.com/support/programming-tools/iotappzone/developers/.
2. Click the AppZone installation file.

   Note: You must have an internet connection to install AppZone Linux because the installation files are downloaded after you select which environments to install.

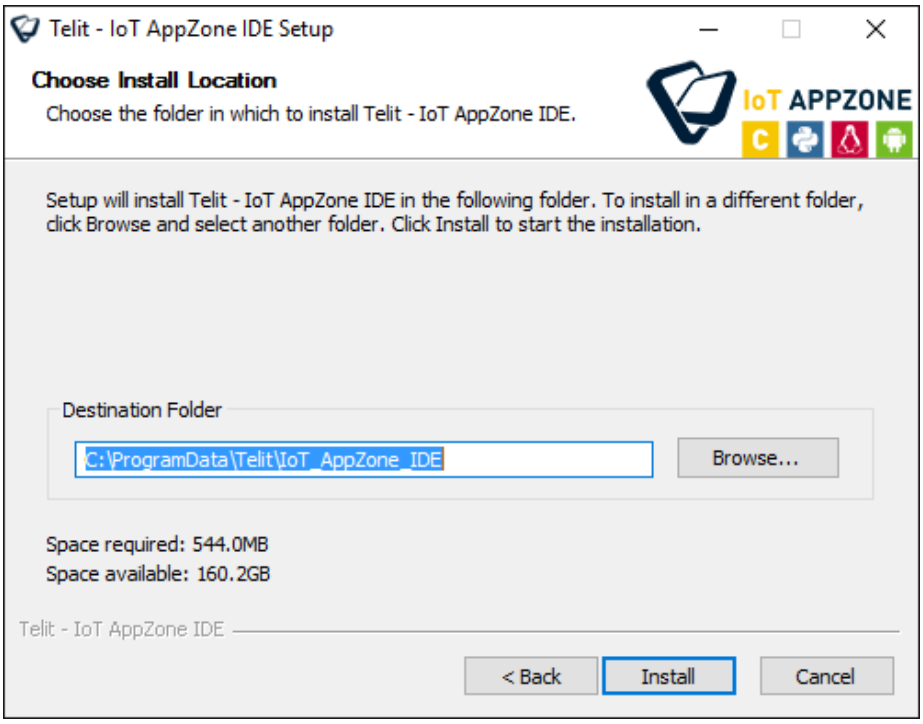
3. If Java 32-bit is not yet installed, a message opens notifying you that you must install Java.
   a. Click **OK**. The browser opens with the Java Installation page.
   b. Under **JRE** click **Download**. The JRE download page opens.
   c. Above the table, select **I Agree**.
   d. Click the x86 JRE versions to install.
   e. When the installation ends, click **OK**. The AppZone C installation windows opens.
4. Click **Next** to continue.
5. Read the license agreement. If you agree, click **I Agree** to continue.

6. Select AppZone Linux, and then click **Next**.



To install more than one development environment, click **Custom** and select the development environments that you want to install.

7. Choose the destination folder in which to install AppZone Linux. You can leave the default folder or click **Browse** to select a new destination folder.



8. Click **Install** to begin the installation.

   The installation wizard downloads the files that are required for the development environment that you selected.

9. Click **Finish** to close the AppZone Linux installation wizard.

3.3. Install on Linux

To install AppZone Linux on a Linux host:

1. Download the deb file from the Telit portal:

   http://www.telit.com/support/programming-tools/iotappzone/developers/.

   Note: You must have an internet connection to install AppZone Linux because the installation files are downloaded after you select which environments to install.

2. If you are installing AppZone Linux on Ubuntu 12.04, you must first Update Keys on Ubuntu 12.04.
3. Install libgtk2-perl by entering the following command:

   ```
   sudo apt-get install libgtk2-perl -y
   ```

4. If you are installing on Ubuntu 16.04, you must use Java version 1.7 or higher.

   We recommend that you use Java 1.8. To install Java 1.8, enter the following command:

   ```
   sudo apt-get install openjdk-8-jre -y
   ```

5. Update repository and tools:
   a. Add APT repository with adb tools by entering the following command:

      ```
      sudo add-apt-repository ppa:nilarimogard/webupd8
      sudo add-apt-repository ppa:terry.guo/gcc-arm-embedded
      ```

   b. Update available package list by entering the following command:

      ```
      sudo apt-get update
      ```

6. Install AppZone Linux. Enter the following command:

   ```
   sudo dpkg -i <downloaded deb file>.deb
   sudo apt-get install –f -y
   ```

7. Read the license agreement. If you accept, press Tab and then click **OK** to proceed.
8. For custom installation, you can chose which components to install in addition to AppZone Linux.
9. Press Tab and then click **OK** to install.

   By default, AppZone Linux is installed in the following path: /opt/DevelopEnv/eclipse.

10. To start AppZone Linux, enter the following command:

    ```
    /opt/DevelopEnv/eclipse/DevelopEnv
    ```

    AppZone Linux opens in Eclipse.

    You can also launch AppZone Linux by clicking the Ubuntu logo in the main panel and searching for AppZone.

3.3.1. Update Keys on Ubuntu 12.04

If you are installing on Ubuntu 12.04, enter the following commands:

```
sudo add-apt-repository ppa:phablet-team/tools -y
sudo apt-get update
sudo add-apt-repository ppa:terry.guo/gcc-arm-embedded -y
```

In some cases, you might see the following error message:

gpg: "tag:launchpad.net:2008:redacted" not a key ID: skipping

recv failed

To resolve this error, add the public key manually by entering following command:

```
sudo apt-get update
```

For example, if the error is as follows:

```
GPG error:http://ppa.launchpad.net precise Release: The following signatures
couldn't be verified because the public key is not available: NO_PUBKEY
6D1D8367A3421AFB
```

Enter the following command to add the key:

sudo apt-key adv --keyserver keyserver --recv-keys key

For example:

sudo apt-key adv --keyserver keyserver.ubuntu.com --recv-keys ***6D1D8367A3421AFB***

3.4. Connect the module

Before you start writing an application, you can verify that your module is connected and identified by your computer.

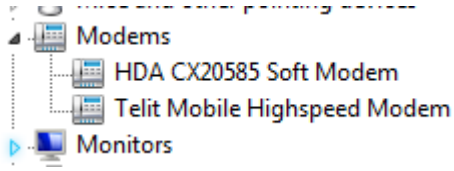
- Connect the module on Windows
- Connect the module on Linux

3.4.1. Connect the module on Windows

To confirm that your module is recognized by your computer:

1. Click **Start**, and then click **Control Panel**.
2. Click **Hardware and Sound**.
3. Click **Device Manager**.
4. Expand **Modems** and verify that a Telit module is listed.

For example:



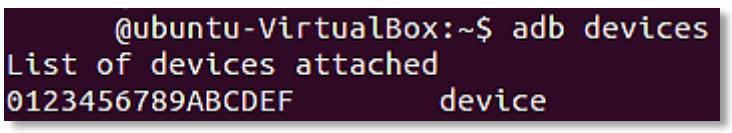
3.4.2. Connect the module on Linux

To confirm that your module is recognized by Linux, connect your module to the computer and then enter the following command:

```
adb devices
```

Make sure that your devices is listed under the list of devices.

For example:



If adb does not detect your device, you may not have enough permissions. To enable adb to recognize a device, restart adb by entering the following commands:

```
echo -e "0x18d1\n0x1bc7" >> ~/.android/adb_usb.ini
sudo cp ~/.android ~root
adb kill-server && sudo adb devices
```

To allow users to access the devices, configure the udev rules:

1. Add the following lines to the /etc/udev/rules.d/51-android.rules file:

```
SUBSYSTEM=="usb", ATTR{idVendor}=="1bc7", MODE="0666", GROUP="plugdev"
SUBSYSTEM=="usb", ATTR{idVendor}=="18d1", MODE="0666", GROUP="plugdev"
```

2. Make this file accessible to everyone and then trigger updating udev rules by entering the following commands:

```
sudo chmod a+r /etc/udev/rules.d/51-android.rules
```

```
sudo udevadm trigger
```

3. Verify that your username is part of the **plugdev** and **dialout** groups by entering the following command:

```
groups | grep 'plugdev\dialout'
```

4. If your username does not appear on the list, add your `user` by entering the following commands:

```
adduser `whoami` plugdev
```

```
adduser `whoami` dialout
```

5. Log out to apply the changes, and then log in again.

3.5. Uninstall AppZone Linux

To uninstall AppZone Linux from Linux:

1. Back up existing projects.

Warning: If you do not back up your projects before uninstalling AppZone Linux, your projects will be deleted.

2. Close running instances of IoT AppZone Linux IDE.
3. Enter the following command:

```
sudo apt-get purge <package name>
```

By default, enter the following command:

sudo apt-get purge telit-appzone-installer

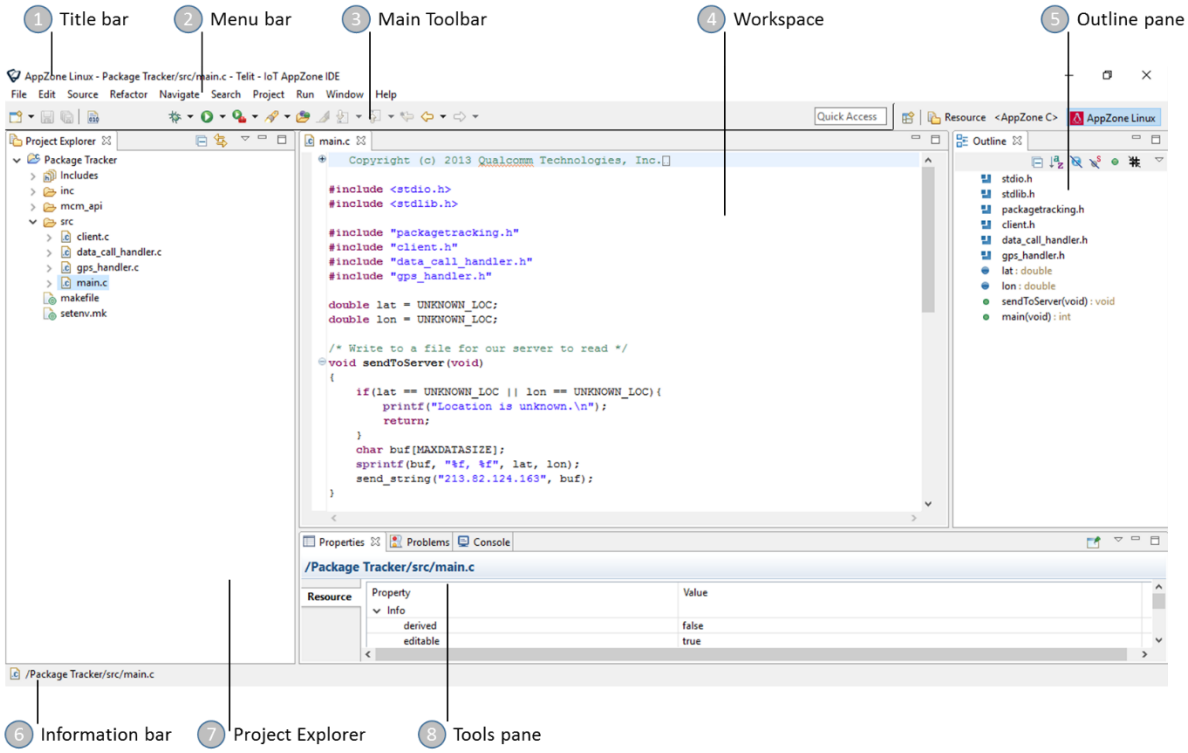
4. GET STARTED

To open the development environment on Windows:

You must have administrator privileges or the development environment will not load.

1. Click **Start > Telit > AppZoneIDE > AppZoneIDE**. The Workspace Launcher window opens.
2. Select the workspace in which you want to work, and then click **OK**.

The following figure describes the main sections of the AppZone C window:



You can optimize the location of the panes and the tabs to suit your workflow. You can rearrange the layout of the panes and the tabs.

The main window contains the following sections:

1. Titlebar – Displays information on the environment, the current project, and the file that is currently displayed in the workspace in the following format: AppZone C <*Project name*>/<*File name*>
2. Menu Bar – Provides options that enable you to perform the most common operations on items. Most of these options also appear in the pop-up menus when you right-click items. The menu bar also contains options related to the entire application.
3. Main Toolbar – Provides buttons for navigation and most commonly used operations.

If you have installed more than one development environment, the toolbar contains a Resource button. Use the resource button to change the perspective and switch to another development environment.

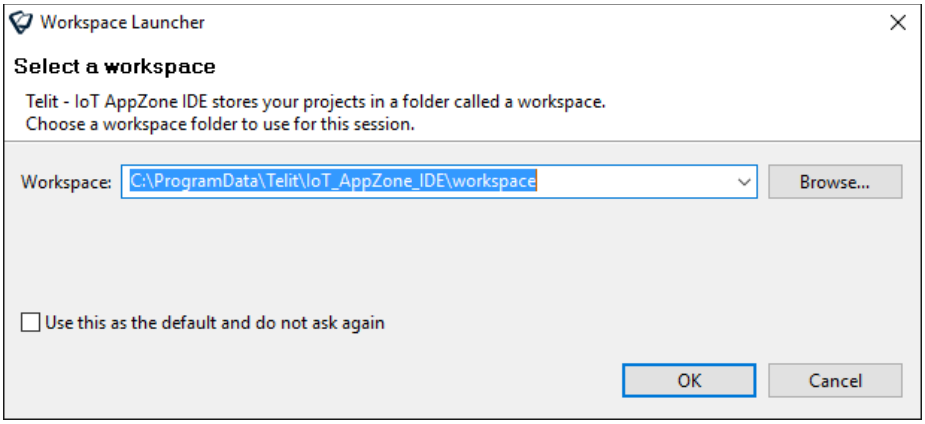
4. Workspace – Displays the details of the file that you select in the Project Explorer pane. In the workspace you define write and edit the code.
5. Outline Pane – Provides an outline of your current project.
6. Information Bar – Displays information on the file that is currently displayed in the workspace in the following format: /<*Project name*>/<*File name*>
7. Project Explorer – Displays all files in the current project.
8. Tools Pane – Displays the following views:
    - Problems – Displays system-generated errors, warnings, or information associated with a resource.
    - Tasks – Displays all the tasks in the project. The view displays tasks associated with specific files, specific lines in specific files, as well as generic tasks that are not associated with any specific file.
    - Console – Displays a build console that enables viewing the status of the current build.
    - Properties – Displays the properties of the file that is currently selected in the Project Explorer pane.

5. CREATE AN APPLICATION

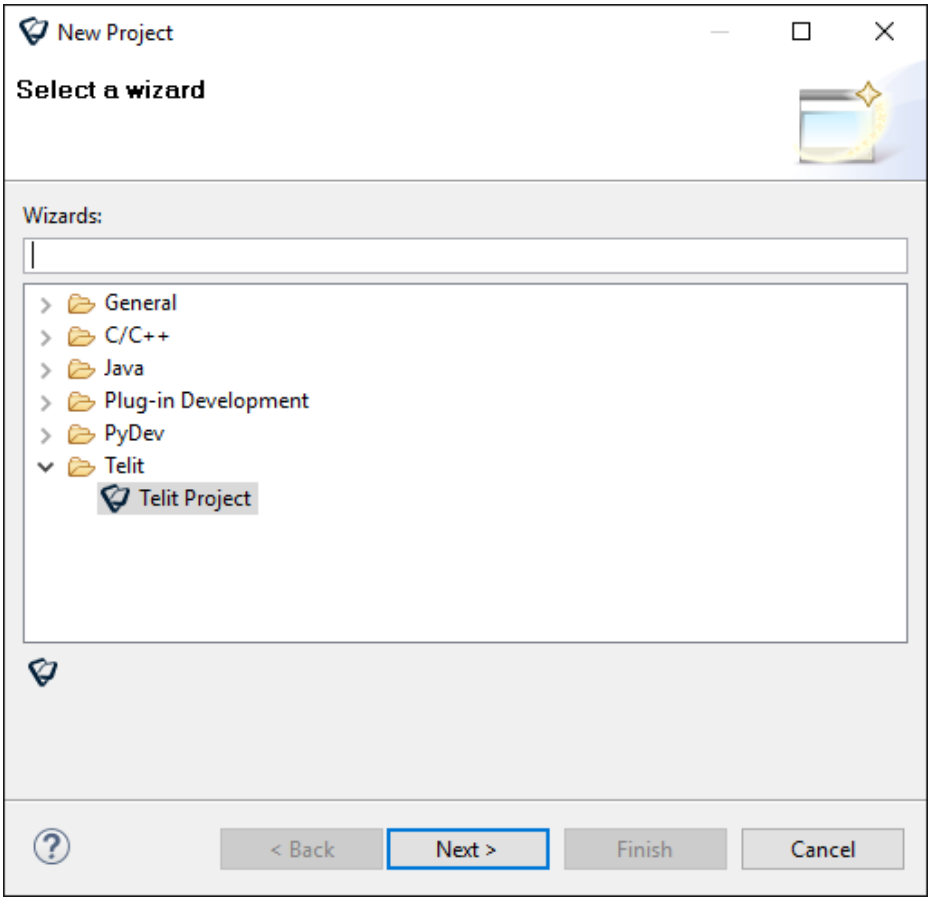
You can create new applications from scratch or use an example application.

To create a new application:

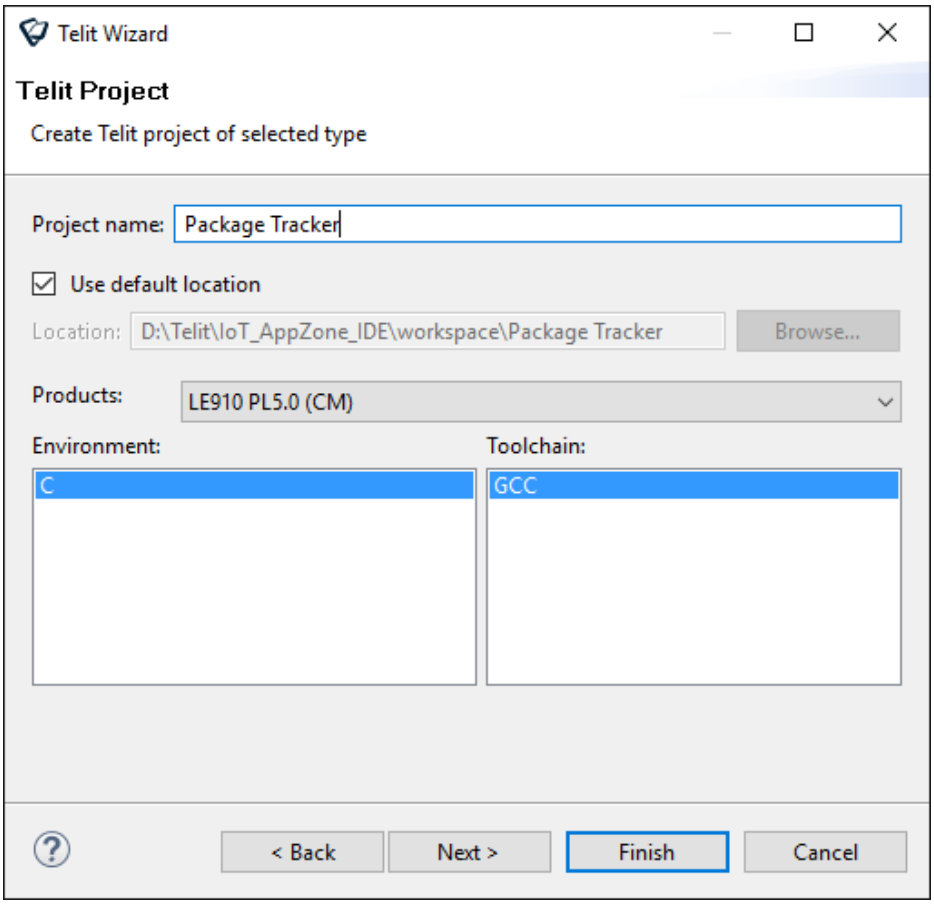
1. Click **Start > All applications > Telit > AppZoneIDE > AppZoneIDE**. The Workspace Launcher window opens.
2. In the Workspace field, type the location of the workspace or click Browse.



3. Click **OK**. The AppZone Linux IDE opens.
4. From the menu select **File > New >** Project. . The New Project window opens.



5. To create a new project, expand **Telit** and then select **Telit Project**.
6. Click **Next**. The Telit Project window opens.

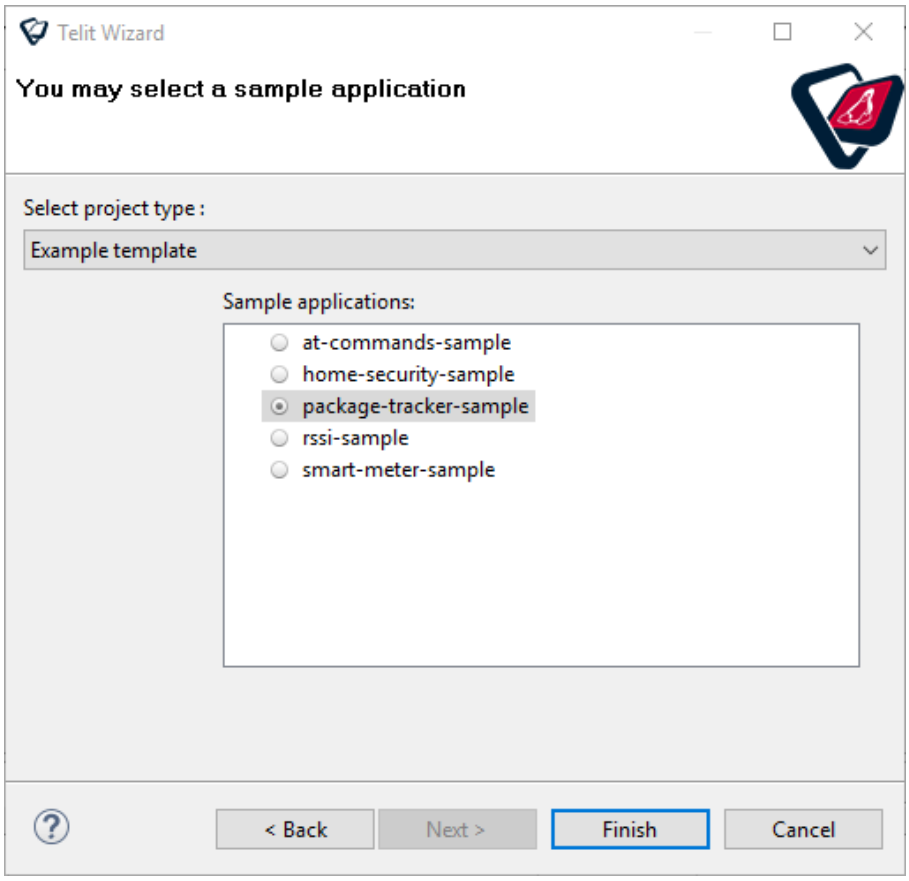


a. In the Project name field, type a name for the project.
b. To change the default location in which the project is saved:
c. Clear the Use default location checkbox.
d. In the Location field type the new location or click Browse.
e. In the Products field, select the module for which you want to develop the application.
f. In the Environment field, select C.
g. In the Toolchain field, select GCC.

7. Click **Next**. A window enabling to select the project type opens.

8. Select the type of project that you want to create:

   - Empty project – Create an empty project. From the Select project type list, select **Empty Project.**
   - Example project – Create a project based on an example application. From the Select project type list, select **Example Template**.

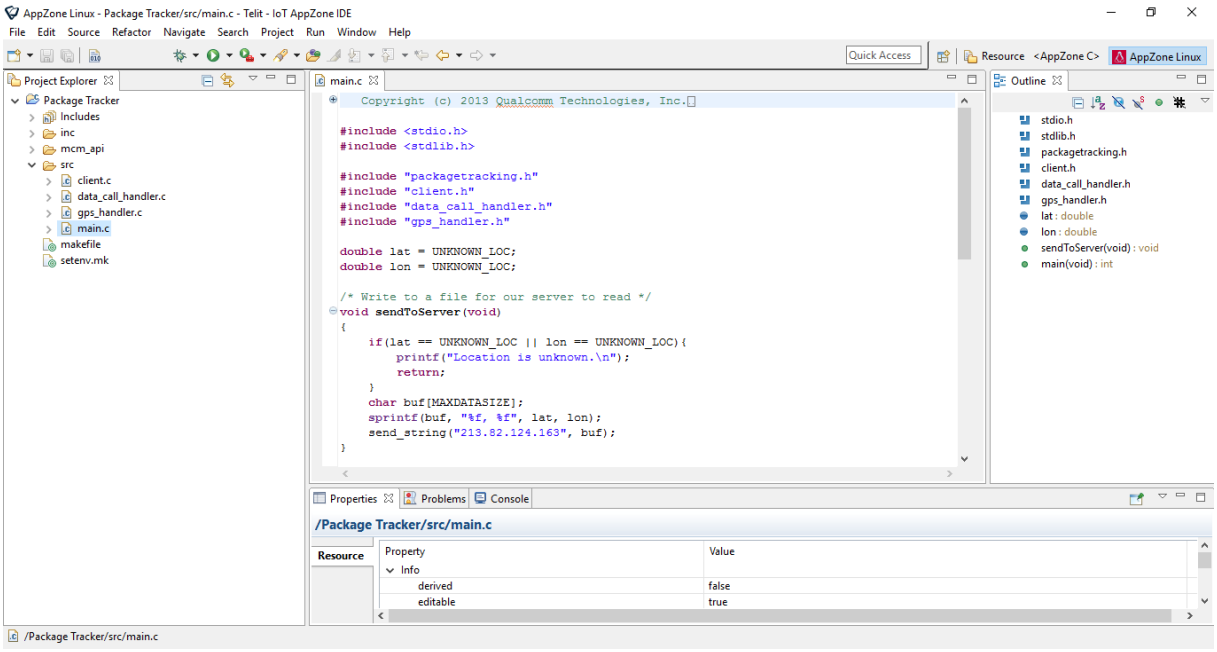


Select the checkbox next to the example application that you want to use as a basis for your project. You can select one of the following example applications:

- **at-commands-sample** – Simulates sending AT-Commands and receiving the responses. The application contains definitions with the sample AT-Command (AT+COPS? By default), and then displays the results in the terminal.
- **home-security-sample** – Simulates a home security system. The home_security_config.txt file contains the details of the application server, the phone number for emergency, and the timeout for disabling the alarm. The application receives certain security events, such as: detected movement, opened door, and turn on/off alarm. According to the received event, the application sends the information to the server, performs a voice call, or sets the alarm accordingly.
- **package-tracker-sample** – Simulates a package tracking system. The package location is monitored by the application and its coordinates are sent periodically to a server.
- **rssi-sample** – Simulates sending AT+CSQ AT-Command and displaying the RSSI in the terminal every 5 seconds.
- **smart-meter-sample** – Simulates a gas metering system. The smart_meter_config.txt file contains the phone number of the application server or the operator, the phone number for emergency, and the interval between the application operations. The smart_meter_values_text.txt file contains the detected meter values. The application monitors the gas meter level and sends SMS messages with the meter level to the server or the operator. I a gas leak is detected, the application makes an emergency call to the gas company.

9. Click **Finish**. AppZone Linux opens.

If you created a project based on example application, AppZone Linux contains the source files for that example.



6. COMPILE AN APPLICATION

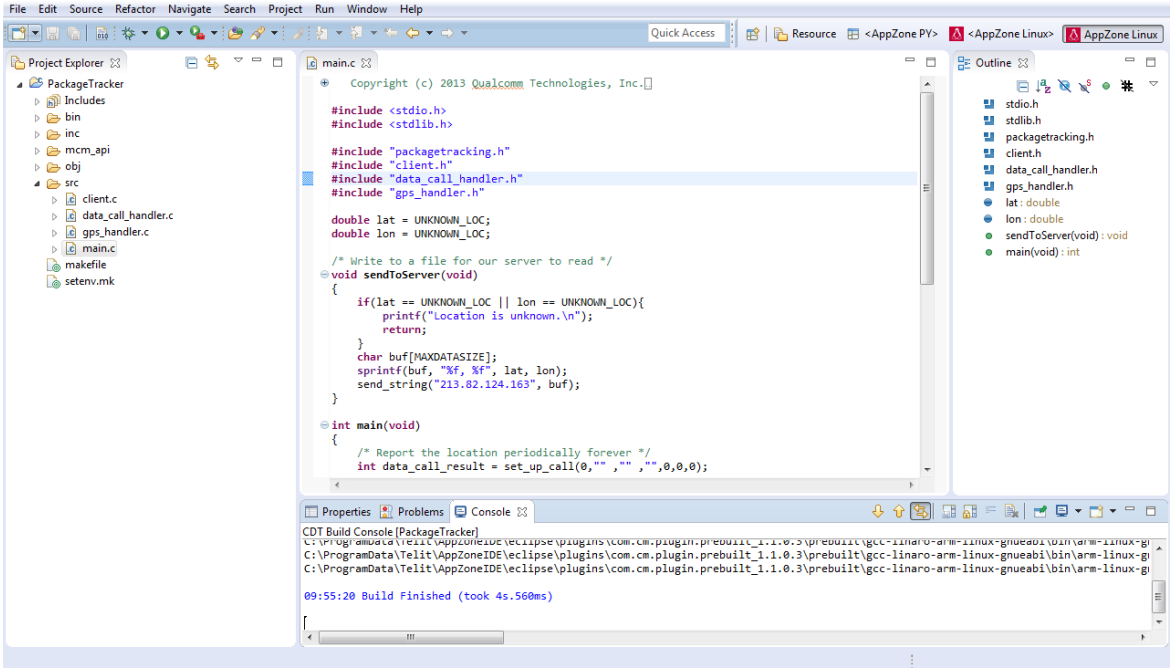
AppZone Linux provides a plugin that allows you to use skeleton files that are automatically generated when creating a project from the Eclipse IDE.

To compile the application:

1. In the Project Explorer, select the project that you want to compile.
2. From the menu, select **Project > Build Project**.

   You can see the results of the build process in the **Console** view. Click on its tab in the Tools pane to bring the view forward if it is not currently visible.

   If it is not visible, you can open it by selecting **Window > Show View > Console**.



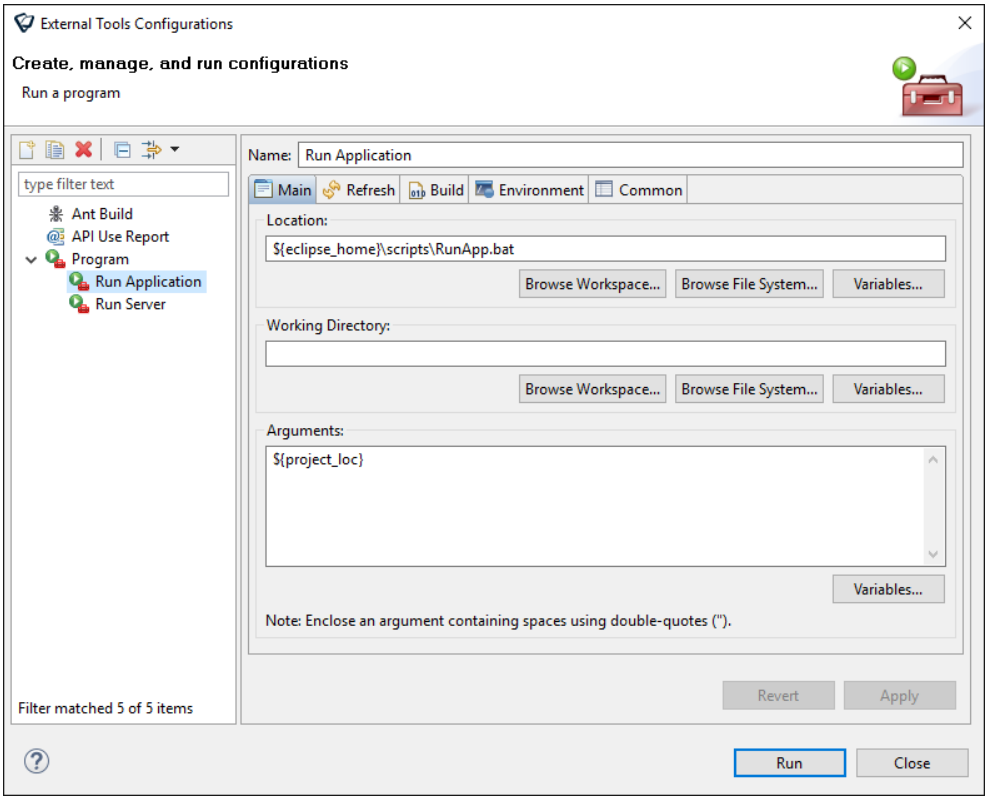
The application executable file is created under the bin folder.

7. DOWNLOAD AND RUN THE APPLICATION ON THE MODULE FROM ECLIPSE

AppZone Linux contains a default workspace that enables you to download and run the application directly from Eclipse.

To download the application into the module and run it from Eclipse:

1. In the Project Explorer view, select the project that you want to download to the module.
2. In the Toolbar click **Run Application** (), and then select **External Tools Configuration**. The External Tools Configuration window opens.



3. Expand the Program section and the select **Run Application**.

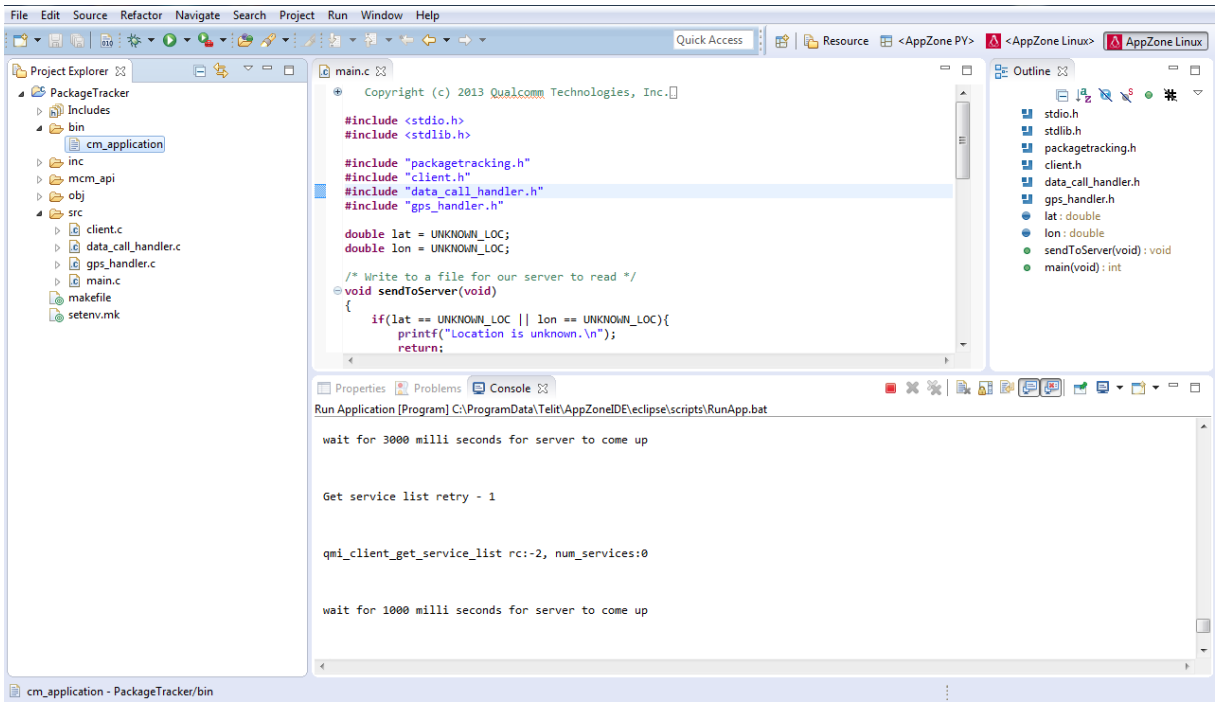
   The properties of the script file that runs the application are displayed.

> If you do not click Run Application even though it is selected, you may get an error after you click Run.

4. Click **Run**.

   The Run command runs the RunApp script that is located in the eclipse/RunApp folder.

5. View the ADB messages that are displayed when the application runs in the Console view.



After running the script, the application is loaded and run on the module.

The RunApp script also configures the module to run the application each time it starts after power up.

To manually configure the module to start the application automatically each time it starts, see the Configure the application auto-start option section.

To disable the auto-start option, see the Disable the application auto-start option.

To use a different workspace (and not the preconfigured default workspace) in your development process, configure the RunApp script to load your application into the module from Eclipse.

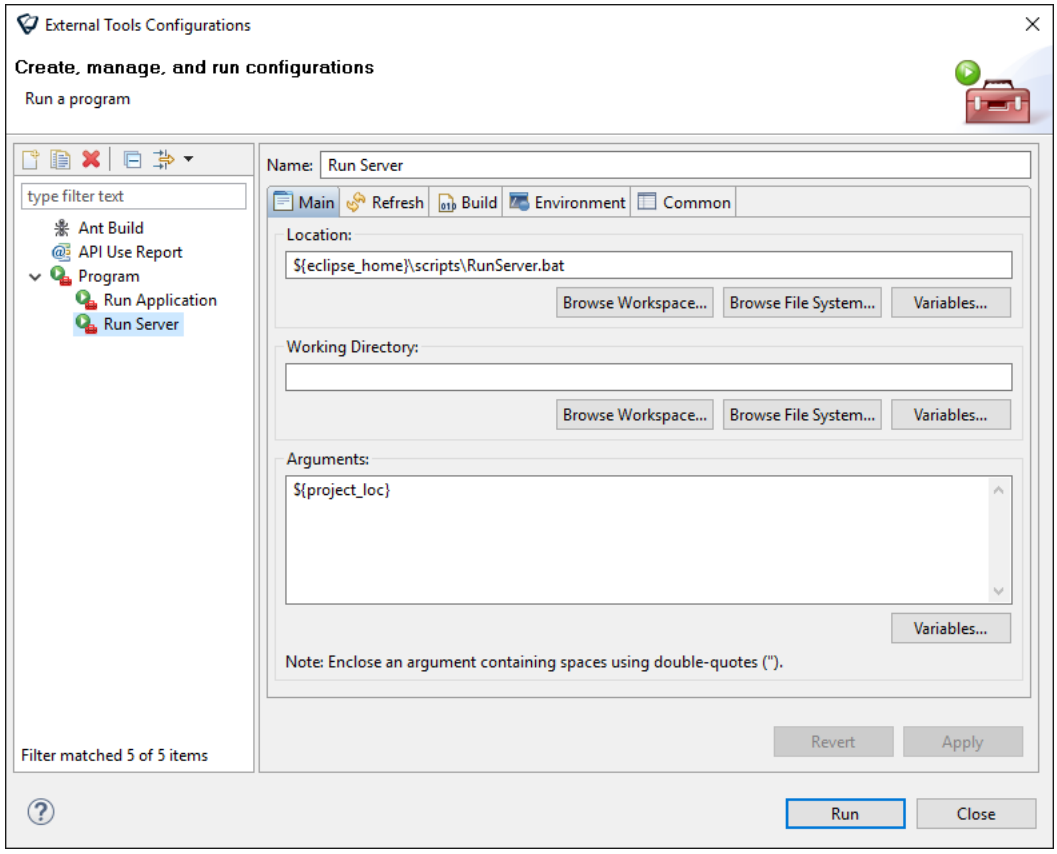
For information on how to configure the script manually, see Configure the RunApp script.

For detailed information on the AppZone Linux APIs, see the *AppZone Linux API Reference Guide*.

8. DEBUG THE APPLICATION

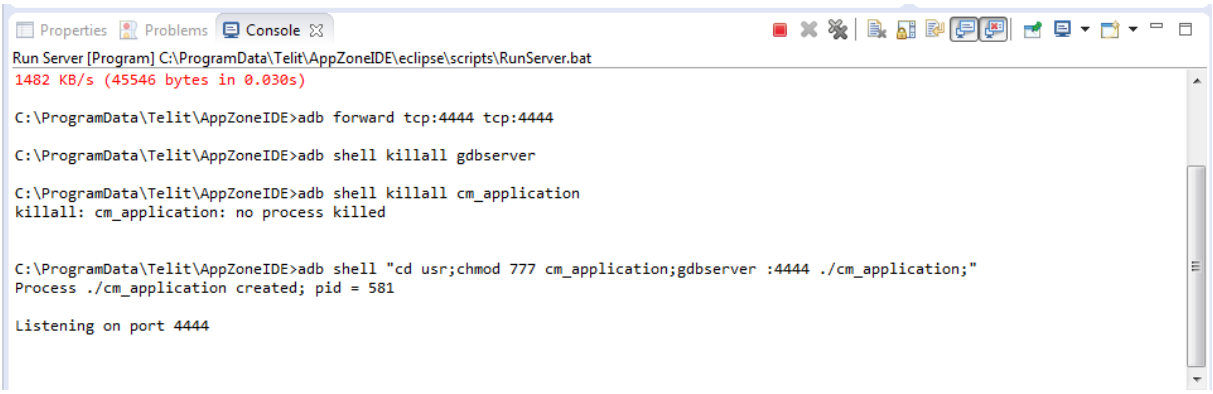
To debug an application in Eclipse:

1. In the Project Explorer view, select the project that you want to debug.
2. In the toolbar, click **Run Application** ( ) and then select **External Tools Configuration**. The External Tools Configuration window opens.

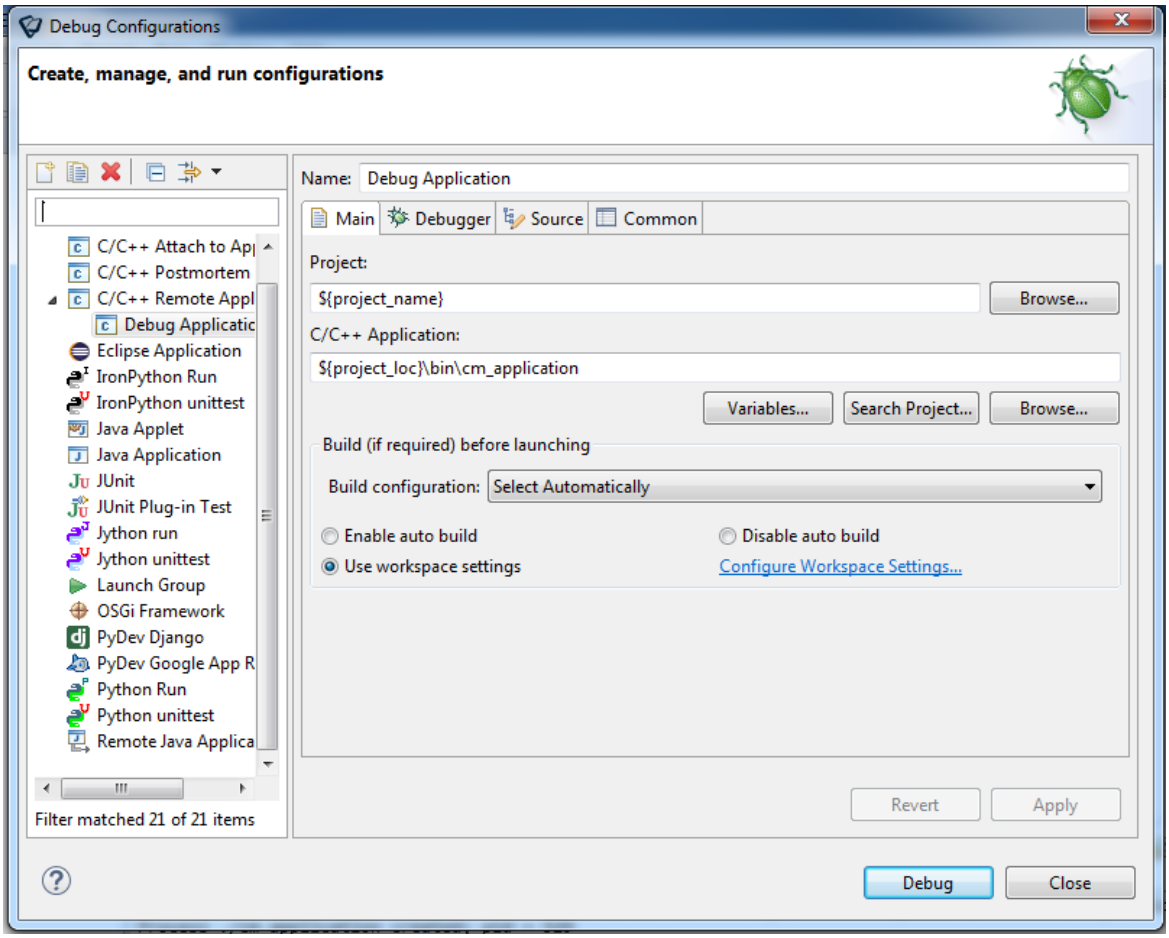


3. Expand the Program section, and then select **Run Server**.
4. Click **Run**.

   The GDB server runs on the module with the application that you built. You can see the messages in the Console view.



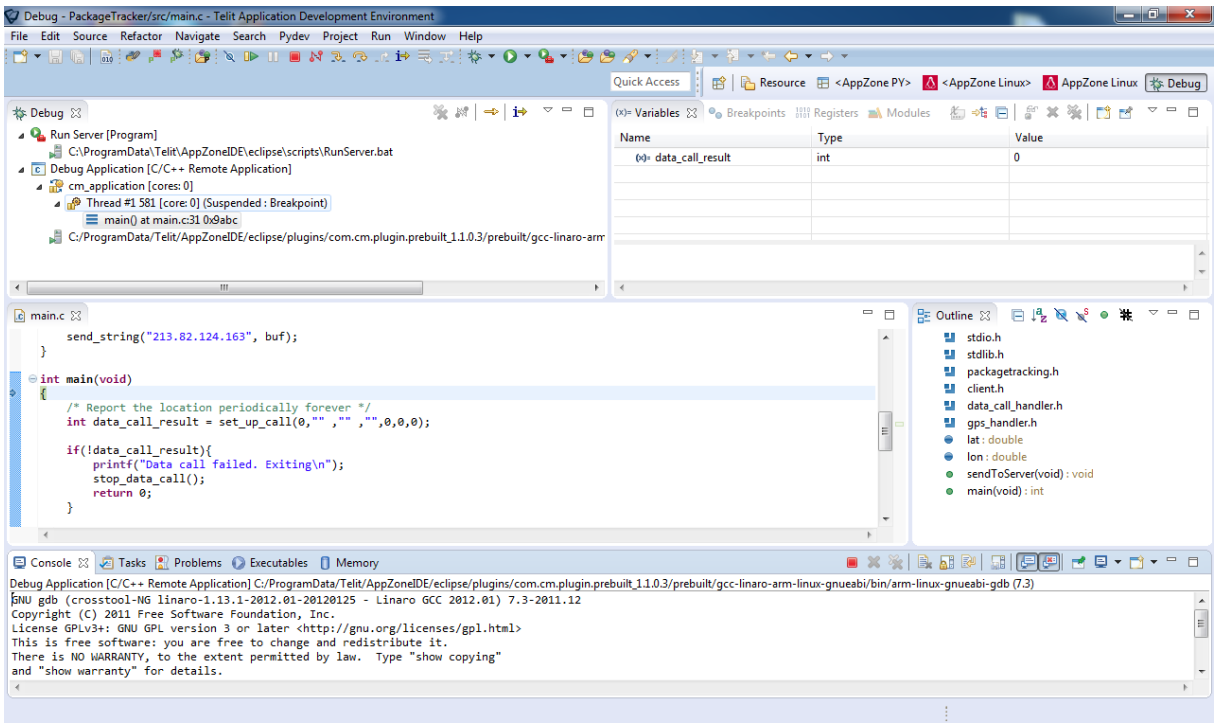
5. After running the server on the module, in the toolbar click **Debug Application** () **> Debug Configurations**. The Debug Configurations window opens.



6. Select **C/C++ Remote Applications > Debug Application**, and then click **Debug**.
7. To open the Debug perspective, click **Yes**.

   You can open the debug perspective at any time.

   The following example shows the Debug perspective:



9. RUN AND CHANGE SCRIPTS MANUALLY

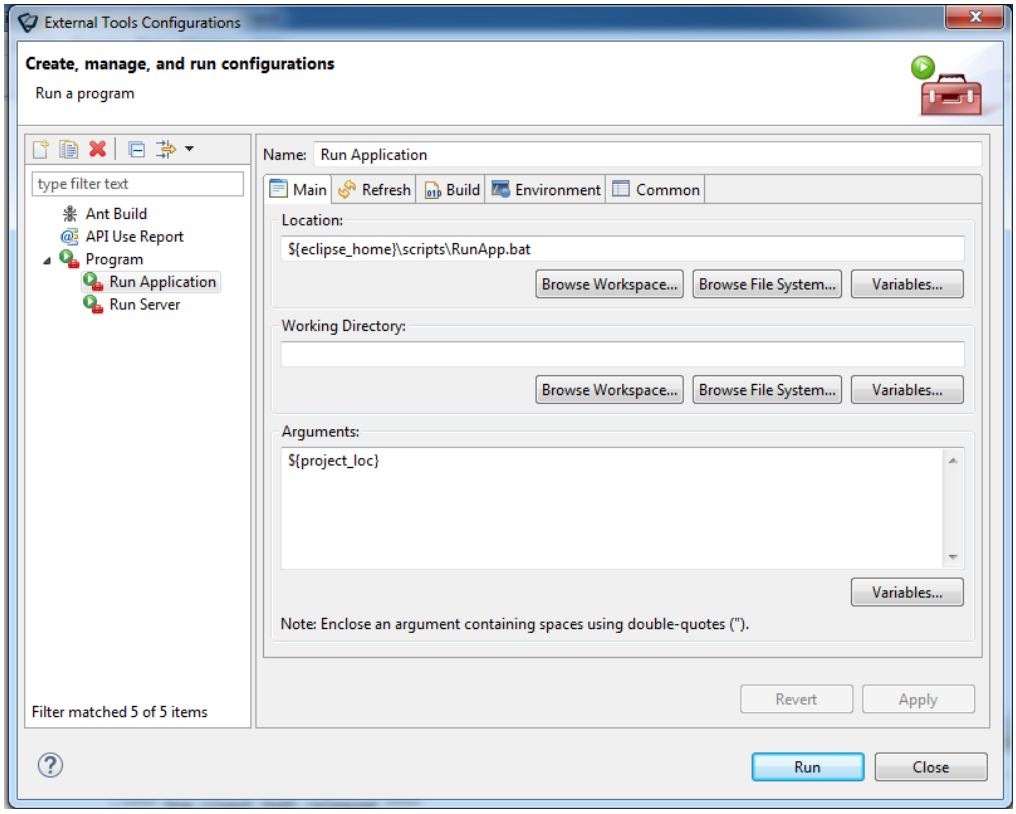
You can change the default configuration of the scripts that AppZone Linux uses.

9.1. Configure the RunApp script

After you modify the RunApp script, you can use to load and run applications from Eclipse as described in Download and run an application

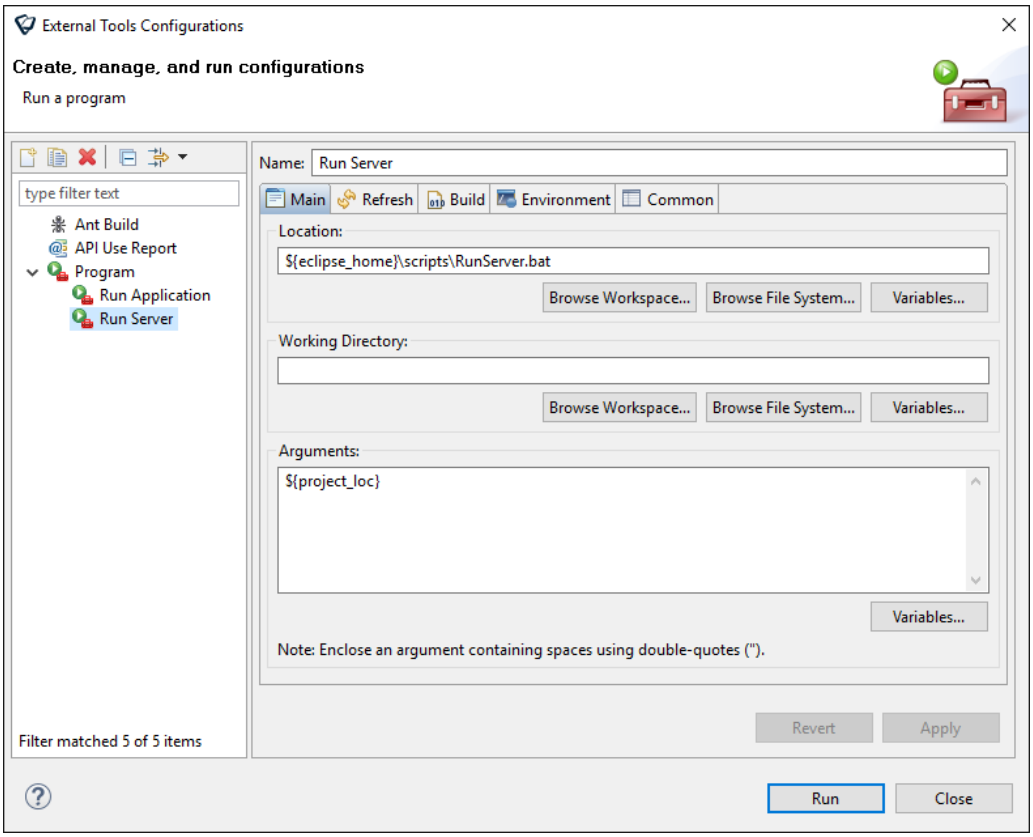
To configure the RunApp script:

1. In the Toolbar click **Run Application** (), and then select **External Tools Configuration**. The External Tools Configuration window opens.



2. In the left pane, right-click **Program** and then select **New**.

   A new configuration is added to the Program list.



3. In the Name filed, which is located above the tabs, field type a name for your script.
4. In the Location field type the path and name for the new script that you created.

   For example, on Windows: ${eclipse_home}/RunApp/RunApp.bat.
5. In the Arguments, click **Variables**. Select project_loc from the list and then click **OK**.
6. Click **Apply** to save the configuration that you just created.
7. Click **Close**.

9.2. Download and run an application

To download and run an application on the module:

1. Open a Command window (in a Windows environment) or a Terminal (in a Linux environment).
2. To copy the application executable file to the module, type the following command:

   adb push <*destination of the executable file*> <*path in module*>

   where:

   - *destination of the executable file* – location of the cm_application file in your development environment.

     In a Windows environment the file, by default, is located in the following folder: workspace/<*project name*>/bin

     In a Linux environment the file, by default, is located in the following folder: workspace/<*project name*>/lib

     To verify the eclipse project path, in the Eclipse Project Explorer pane, right-click the project name and the select Properties.

   - *path in module* – The path to which you want to copy the file in the module. The default folder to be used in the module's file system is /usr.

The following example on Windows copies the cm_application file to the usr folder:

```
C:\>adb push Telit\workspace\MyFirstApp\bin\cm_application /usr
889 KB/s (45546 bytes in 0.050s)
```

3. To enter adb shell mode adb shell and navigate between the files in the module, type the following command:

   adb shell

   The following example shows how to enter the adb shell on Windows:

   ```
   C:\>adb shell
   / #
   ```

4. Navigate to the usr folder to which you copied the application by typing the following command:

   ```
   cd usr
   ```

   For example:

   ```
   / # cd usr
   /usr #
   ```

5. Change the permissions of the application by typing the following command:

   ```
   chmod 777 <application name>
   ```

   Where *application name* is the name the executable file that you copied to the module.

   The following example shows how to run an application with the default name of cm_application:

   ```
   /usr # chmod 777 cm_application
   ```

6. To run the application, navigate to the location of the executable file that you copied, and type the following command:

   ```
   ./<application name>
   ```

   Where *application name* is the name of the executable file that you copied to the module.

   The following example shows how to run an application with the default name of cm_application:

   ```
   /usr # ./cm_application
   *** Telit Connection Manager Application - In Main Function ***
   ```

After the application is executed, the application output is printed to the adb shell.

9.3. Configure the application auto-start option

To configure the module to run an application automatically each time it starts:

1. Open a Command window (in a Windows environment) or a Terminal (in a Linux environment).
2. Navigate to the application executable file folder.

   The default path is: C:\ProgramData\Telit\AppZoneIDE\eclipse\workspace\<*application name*>\bin
3. Copy the application executable file to the /usr/bin folder on the module by typing the following command:

   adb push cm_application /usr/bin
4. Navigate to the location of the start_app_script init script.

   The default path is: C:\ProgramData\Telit\AppZoneIDE\eclipse\scripts
5. Copy the script to the /etc/init.d folder on module by typing the following command:

   adb push start_app_script /etc/init.d
6. Open the adb shell by typing the following command:

   adb shell

7. Navigate to the etc/init.d folder on the module by typing the following command:

   cd etc/init.d

8. Change the permissions of the script by typing the following command:

   chmod 755 start_app_script

9. To configure the script to run automatically on startup, type the following command:

   update-rc.d start_app_script start 45 2 3 4 5 . stop 80 0 1 6 .

9.4. Disable the application auto-start option

To disable the auto-start option so that the module does not run the Telit application automatically each time it starts:

1. Navigate to the location of the AppZone Linux script.

   The default path is: C:\ProgramData\Telit\AppZoneIDE\eclipse\scripts

2. Double-click the DisableApplicationAutoRun script.

To disable the auto run option manually:

1. Open the adb shell by typing the following command:

   adb shell

2. Navigate to the etc/init.d folder on the module by typing the following command:

   cd etc/init.d

3. Change the permissions of the script by typing the following command:

   chmod 755 start_app_script

4. To disable the auto run option in the script , type the following command:

   update-rc.d start_app_script remove

9.5. MCM logging

Logging is implemented in B021 for LE920.

The logging is based on the configuration specified in the /data/mcm-core/log_config.txt file.

If the file does not exist, you must create it manually.

To find the location for the file:

1. Open the ADB shell.
2. Navigate to the following location:

   C:\ProgramData\Telit\ConnectionManager\Configurations\platform-tools

3. Enter the following commands:

```
ADB SHELL
CD /data/mcm-core
```

   /Logging is defined based on the value in the file:

   - 0 - disabled
   - 1- console
   - 2 - file

By default, logging is saved in a file.

NOTE:

If you change the logging option, you must restart the MCM client for the change to take effect.

When logging in FILE is enabled, a log directory is created in /data/mcm-core/, and a file with a date and time is created for every MCM session. There can be a maximum of three log files at a time. If there are more than 3 log files, the oldest files are deleted.

10. SAMPLE APPLICATIONS

Telit provides sample applications.

10.1. Load an application to the module

You can load an existing application from Eclipse. For information, see Download and run an application.

To load the application from a Linux terminal:

1. Open a terminal.
2. Enter the following command to navigate to the application lib folder:

   ```
   cd ~/workspace/SampleApp/lib
   ```

3. Enter the following command to copy the application to the module's usr folder:

   ```
   adb push cm_application /usr
   ```

4. Enter the following command to enter the module's Linux system:

   ```
   adb shell
   ```

5. Enter the following command to enter to navigate to the application folder:

   ```
   cd usr
   ```

6. To run the application, enter:

   ```
   ./cm_application
   ```

10.2. Home security

The application simulates a home security system.

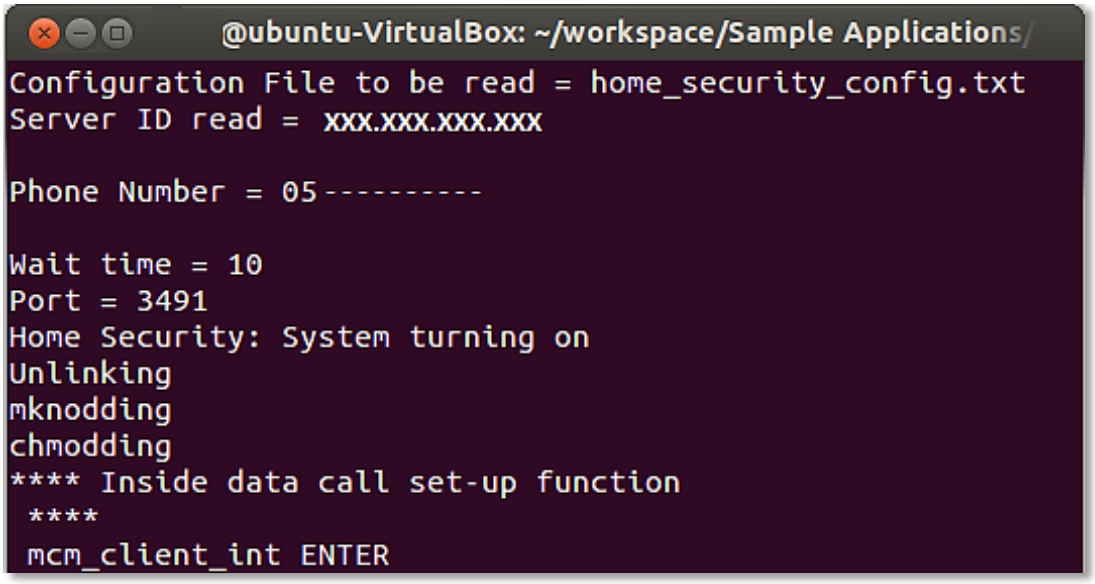
The 'home_security_config.txt' file contains the details of the application server, the phone number for emergency, and the timeout for disabling the alarm.

The application receives certain security events, such as: Detected movement, opened door, and turn on/off alarm.

Based on the received event, the application sends the information to the server, performs a voice call, or sets the alarm.

Note:

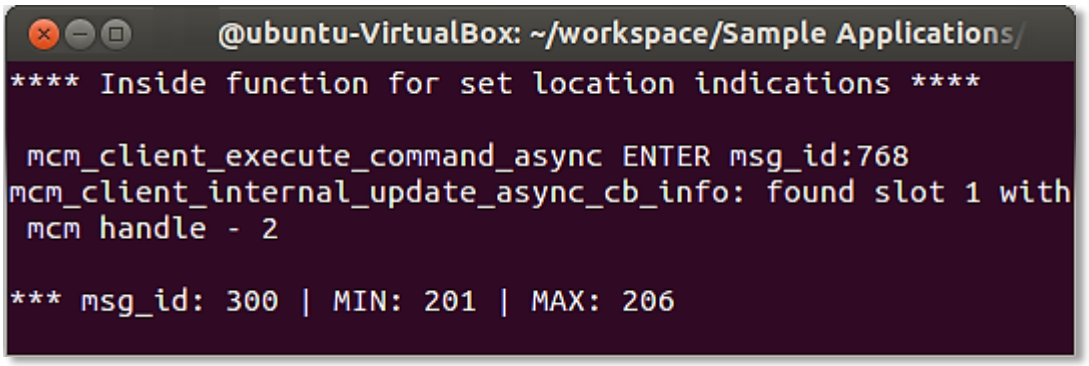
To activate this example application, you must copy both the executable file and the 'home_security_config.txt' file into the module's application folder.



10.3. Package tracker

This application simulates a package tracking system.

The package location is monitored by the application and its coordinates are sent periodically to a server.



10.4. Smart meter

This application simulates a gas metering system.

This application uses the following files:

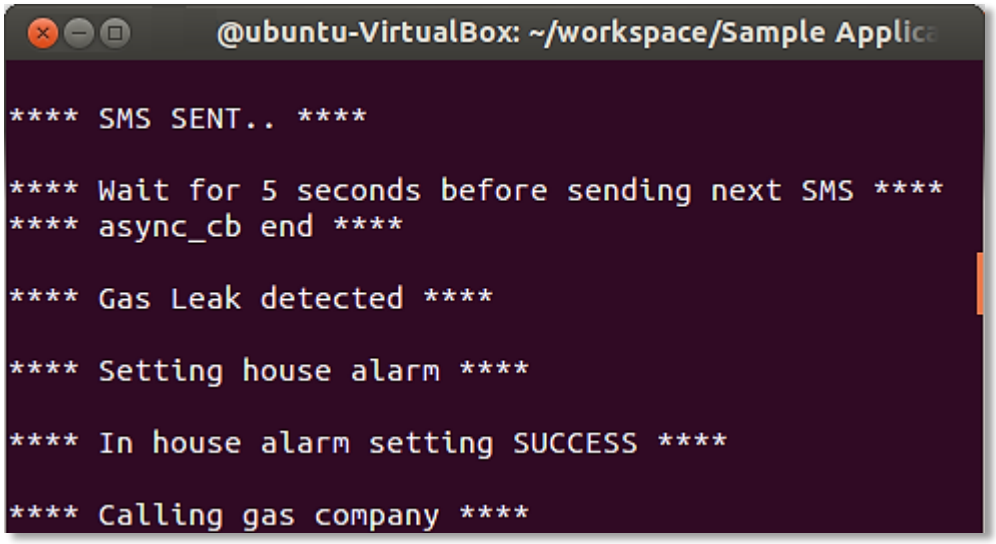
Note:

You must copy both these files into the module's application folder for the application to work properly.

- smart_meter_config.txt – File contains the phone number of the application server or operator, the phone number for emergency, and the interval between the application operations.
- smart_meter_values_text.txt – File contains the detected meter values.

The application monitors the gas meter level and sends SMS messages with the meter level to the server or operator.

If a gas leak is detected, the application performs an emergency call to the gas company.

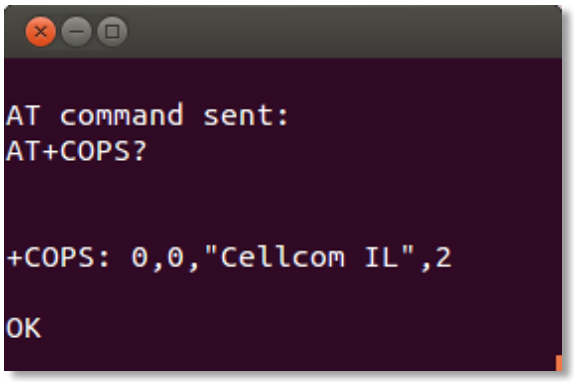


10.5. AT-Commands

This application simulates sending AT-Commands and the received response.

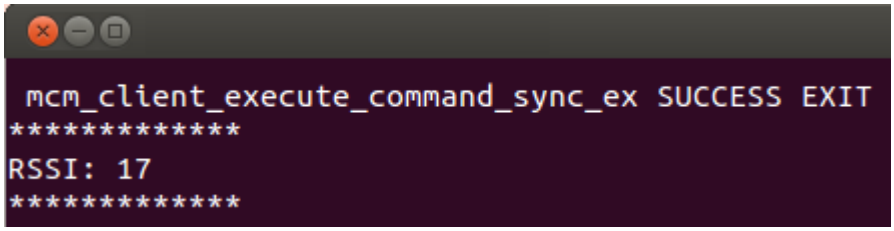
The application contains a definition a default AT-Command (AT+COPS?).

The result is displayed in the terminal.



10.6. RSSI

This application simulates sending AT+CSQ AT-Command and presents the RSSI in the terminal every 5 seconds.




Design Review
120°
180°
Process
Validation
240°
RF/EMC Pre-Certification
Module Selection
60°
300°
Approvals
Support
360°
Value Added Services & Connectivity
Enterprise Integration &
Application Enablement
m2mair