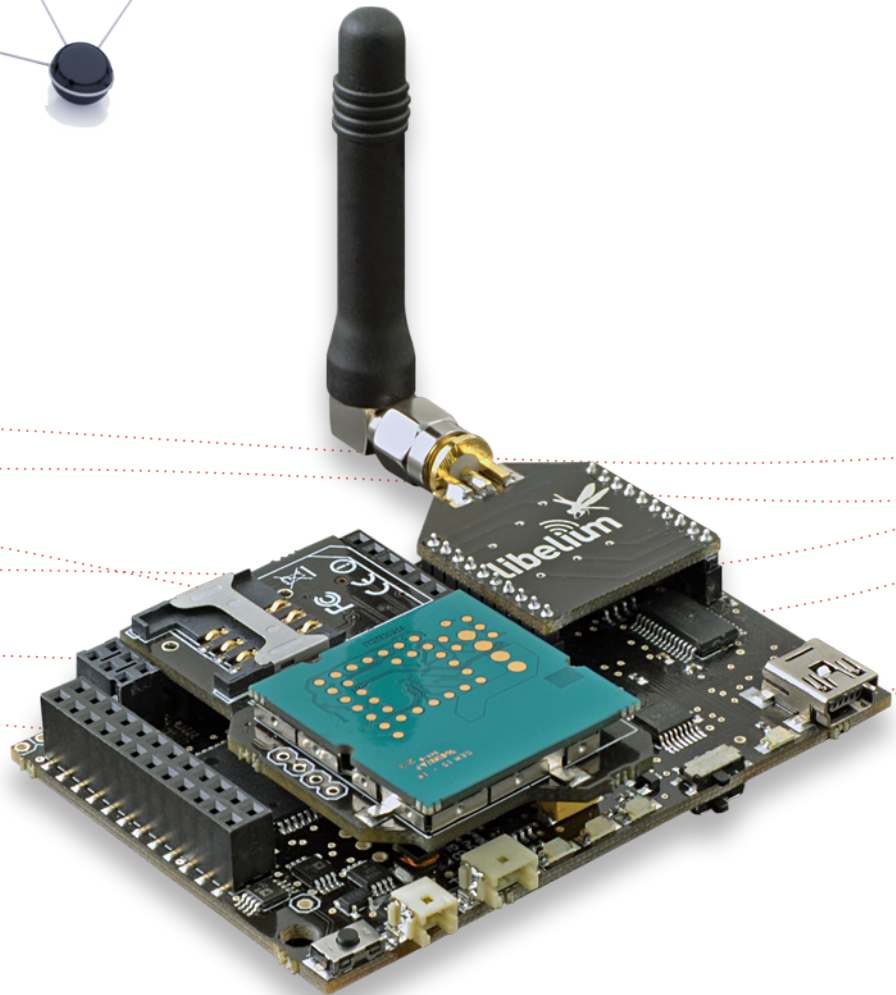
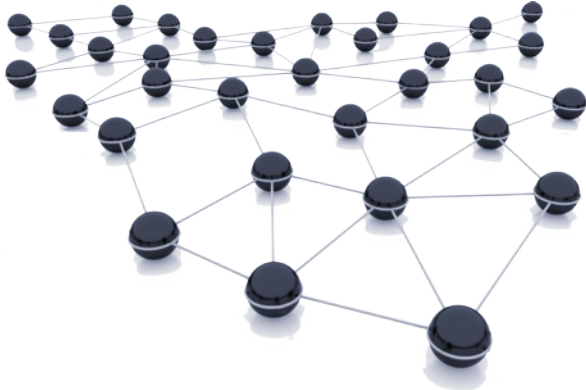


Wasp mote

Quickstart Guide



Document version: v4.2 - 08/2013

© Libelium Comunicaciones Distribuidas S.L.

INDEX

1. Introduction	3
2. General and safety information	4
3. Waspote's Hardware Setup	5
3.1. Batteries.....	5
3.2. Antennas	6
3.3. Modules.....	6
3.4. Sensor Boards.....	7
3.5. SD card.....	7
4. Waspote IDE: Download and Installation	8
4.1. Linux	8
4.2. Windows.....	9
4.3. Mac.....	9
5. Receiving Frames from Waspote	10
5.1. In Waspote Gateway	10
5.2. In Meshlium	11
6. Compiling a New Program	12
7. Uploading a New Program to Waspote.....	15
8. How to Use the Developer's Guides	18

1. Introduction

The aim of this Guide is to introduce the user to WaspMote in a practical way.

WaspMote is a complex device and a learning process must be completed. Libelium offers many Guides and examples which will help the developer.

The present Guide was created in the hope of helping the developer in the very first steps with WaspMote. We advice to follow this Guide when the user wants to start the learning process. The last chapter will try to plan a learning process, proposing further steps.

2. General and safety information

Software:

- Upload code only using Waspote IDE. If a different IDE is used, Waspote can be damaged and can become unresponsive. This use is not covered under warranty.
- Do not unplug any connector while uploading code. Waspote can become unresponsive. This use is not covered under warranty.
- Do not connect or disconnect any connector while Waspote is ON. Waspote can become unstable or unresponsive, and internal parts can be damaged. This fact is not covered under warranty.

Hardware:

- Do not submerge Waspote in liquids.
- Do not place nodes on places or equipment where it could be exposed to shocks and/or big vibrations.
- Do not expose Waspote to temperatures below -10°C or above 50°C.
- Do not power Waspote with other power sources than the original provided by Libelium. Voltage and current maximum ratings can be exceeded, stopping Waspote working and voiding warranty.
- Do not connect any sensor on the solar panel connector and also do not connect the solar panel to any of sensor connectors. Waspote can be damaged and warranty void.
- Do not connect any sensor not provided by Libelium.
- Do not place Waspote where water can reach the device.
- For more information: <http://www.libelium.com>

DO NOT TRY TO RECHARGE THE NON-RECHARGEABLE BATTERY. IT MAY EXPLODE AND CAUSE INJURIES AND DESTROY THE EQUIPMENT. DEVICES WITH NON-RECHARGEABLE BATTERIES MUST BE PROGRAMMED THROUGH THE USB CABLE WITHOUT THE BATTERIES CONNECTED. PLEASE DOUBLE CHECK THIS CONDITION BEFORE CONNECTING THE USB. DO NOT CONNECT EITHER UNDER ANY CIRCUMSTANCE THE SOLAR PANEL TO A DEVICE WITH A NON-RECHARGEABLE BATTERY AS IT MAY EXPLODE AND CAUSE INJURIES AND DESTROY THE EQUIPMENT.

3. Waspote's Hardware Setup

Check the next basic points about the hardware configuration:

3.1. Batteries

Connect the battery to Waspote. Remember you have to charge the batteries at least for 24h before start using Waspote.

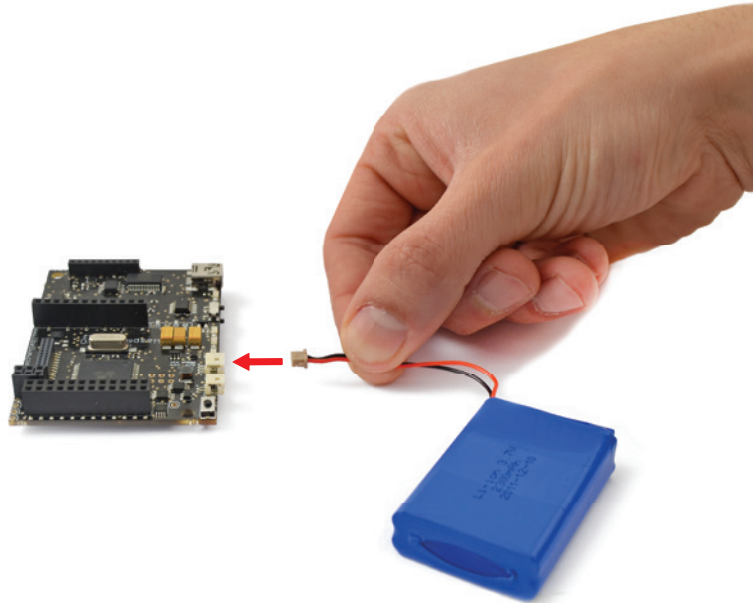


Figure 1: Connect the battery

DO NOT TRY TO RECHARGE THE NON-RECHARGEABLE BATTERY. IT MAY EXPLODE AND CAUSE INJURIES AND DESTROY THE EQUIPMENT. DEVICES WITH NON-RECHARGEABLE BATTERIES MUST BE PROGRAMMED THROUGH THE USB CABLE WITHOUT THE BATTERIES CONNECTED. PLEASE DOUBLE CHECK THIS CONDITION BEFORE CONNECTING THE USB. DO NOT CONNECT EITHER UNDER ANY CIRCUMSTANCE THE SOLAR PANEL TO A DEVICE WITH A NON-RECHARGEABLE BATTERY AS IT MAY EXPLODE AND CAUSE INJURIES AND DESTROY THE EQUIPMENT.

3.2. Antennas

Plug the antennas to the modules they need to.

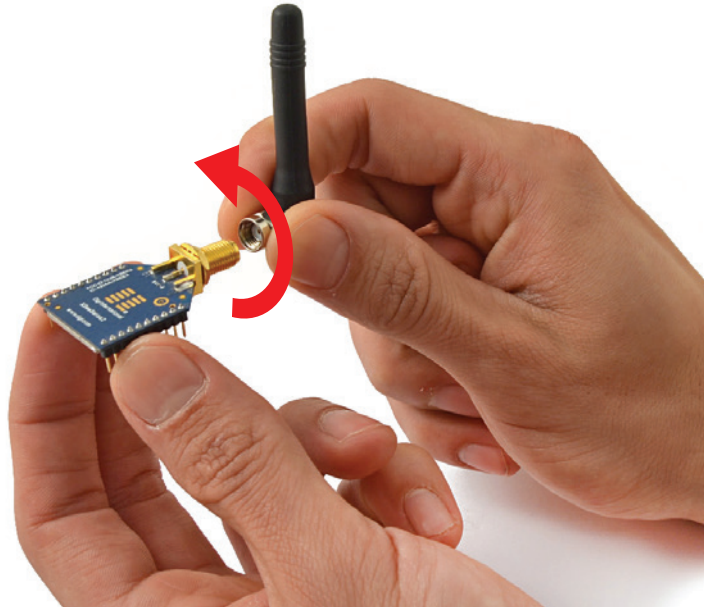


Figure 2: Plug the antenna to a XBee module

3.3. Modules

Place the module to be used in the corresponding socket in Waspote.

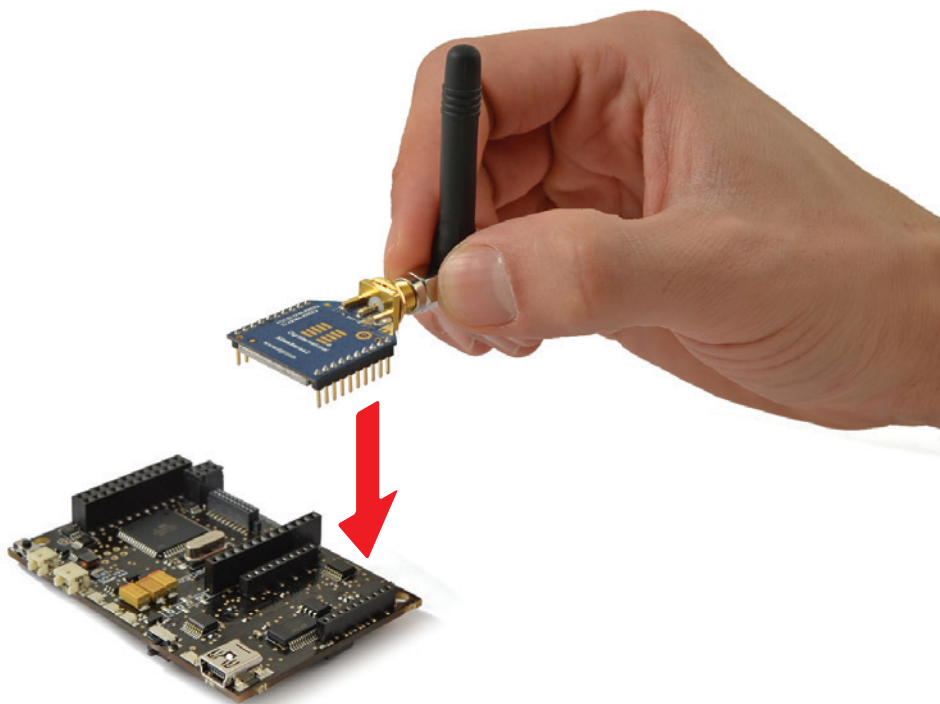


Figure 3: Plug a XBee module

3.4. Sensor Boards

Place the sensor board to be used in Waspote.

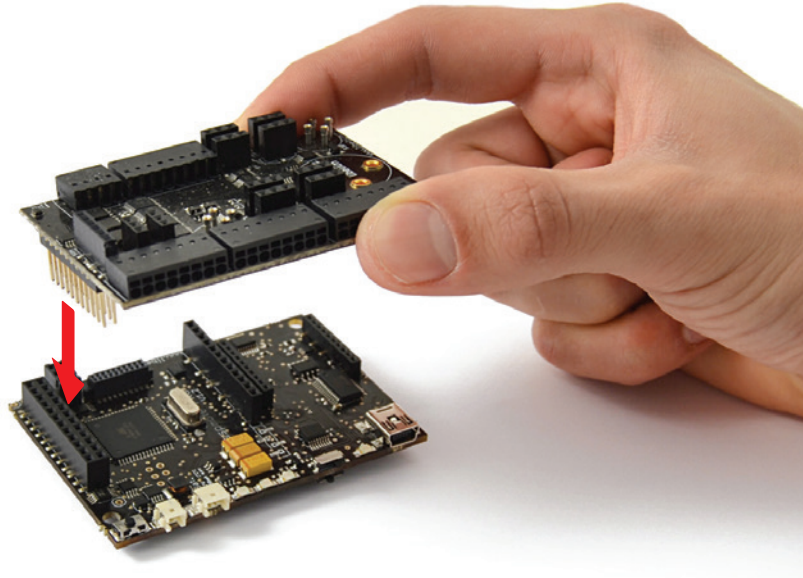


Figure 4: Plug a Sensor Board

3.5. SD card

Insert the SD card into Waspote.

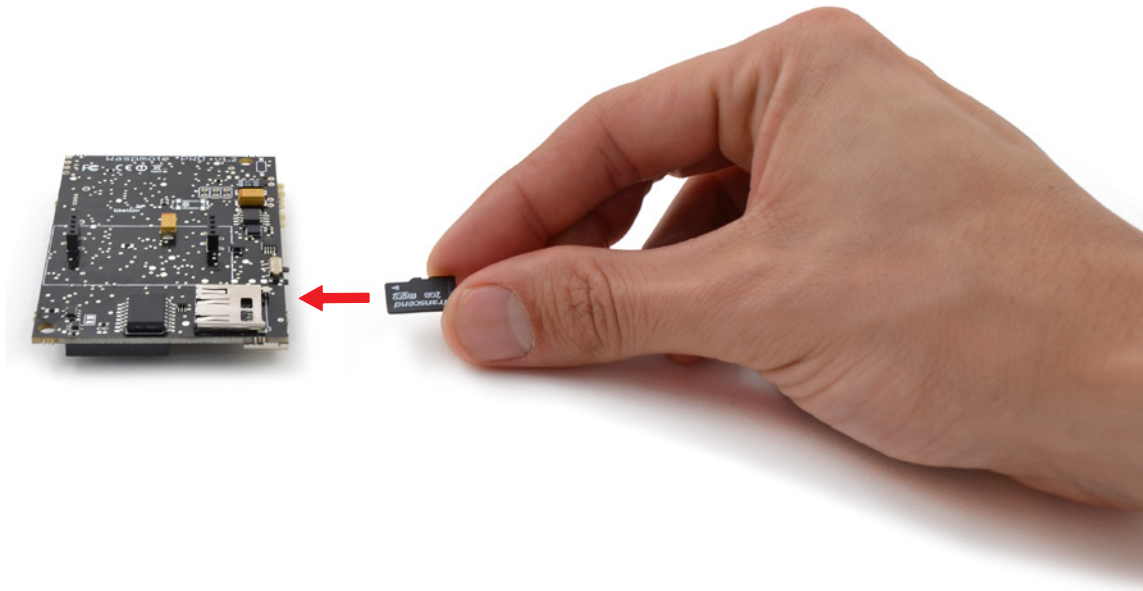


Figure 5: Insert a SD card

4. Wasmote IDE: Download and Installation

The first step is to install the **Wasmote-IDE** (Integrated Development Environment) used to program Wasmote. This IDE can be found on:

<http://www.libelium.com/development/wasmote>

4.1. Linux

To be able to correctly compile and use Wasmote it is necessary to install some packets related with the version of the compiler for Atmel microcontrollers and Java environment.

a) Installing Java

The first step is to install the necessary version of the Java environment. We can use the Synaptic package manager or a terminal.

Using the Synaptic package manager, we must look for the “sun-java6-jre” package and install it.

Using the terminal we must use the apt-get command in the following way:

```
$ sudo apt-get install sun-java6-jre
```

b) Installing AVR-GCC Compiler

The next step is to install the necessary version of the avr-gcc compiler to be able to program the Wasmote ATMEGA 1281 microcontroller. We can use the Synaptic package manager or a terminal.

Using Synaptic we must look for the “gcc-avr” package and install it. Using the terminal we must use the apt-get command in the following way:

```
$ sudo apt-get install gcc-avr
```

c) Installing lib-avc Library

The next step is to install the necessary version of the lib-avc library. We can use the Synaptic package manager or a terminal.

Using Synaptic we must look for the “lib-avc” package and install it. Using the terminal we must use the apt-get command in the following way:

```
$ sudo apt-get install avr-libc
```

d) Running Wasmote IDE

Wasmote installation entails unzipping the file downloaded in the previous step to the chosen folder. Once the downloaded file has been unzipped, the file called **Wasmote** must be run to launch the IDE.

4.2. Windows

a) Installing Waspote

The next step is to unzip the downloaded file to the chosen folder. This folder includes the drivers needed in the next step to install the USB and FTDI converter.

b) Connecting a Waspote board

When connecting a Waspote board using the mini-USB connector, the message "New device found" will appear. A window will open for the installation of this device.

Select the option "Not right now" and press the 'Next' button.

Next select the path where the drivers for the FTDI converter are. These drivers are in the folder where Waspote was unzipped.

Then proceed to the installation of the FTDI converter drivers, which shows the following message when finished.

Once installation is finished, the message 'New device found' will appear, referring to the USB. The same process carried out for the FTDI converter must now be followed, choosing the same options in all the windows. The path for the drivers is the same as that previously specified.

Once this installation is finished, a message will appear indicating the correct installation of the USB.

Once both devices are correctly installed, the port on which the Waspote board has been installed will appear in the "Device Administrator".

4.3. Mac

a) Installing Waspote

The next step is to unzip the downloaded file to the chosen folder. The drivers needed in the next step to install the FTDI converter are found in this folder.

b) Installing FTDI converter drivers

Waspote requires the installation of the FTDI converter drivers. These drivers are found in the downloaded file.

Once the drivers are installed for the FTDI converter, the Waspote board can be connected and the system will recognize it correctly.

5. Receiving Frames from Wasmote

5.1. In Wasmote Gateway

Wasmote comes preconfigured from factory with a program which lets you check the right operation of the device. This program sends standard frames to the Gateway. It is called `wasp_pro_start_program_XXX.pde` and depends on the used XBee protocol. It is also available inside the IDE.

Steps:

1. Install the drivers and the serial monitor software on the computer.
2. Connect the antennas and the rest of the desired components to Wasmote and Wasmote Gateway.
3. Plug Wasmote Gateway to the USB port on the computer.
4. Launch the serial monitor application and set the next parameters:
 - USB port: 115200bps
 - 8bits
 - 1 bit stop
 - no parity setting
5. Connect the batteries to the Wasmotes.
6. Switch Wasmotes to the ON position.

When the program starts, it executes sequentially these actions:

- State 1 – Leds ON for 5 seconds
- State 2 – Leds blinking for 3 seconds
- State 3 – Sending messages

State 1 and 2 are only executed once (when program starts) whereas state 3 will loop indefinitely every second (if we reset Wasmote, the program starts again).

Every packet contains a message with sensor data formatted as Wasmote Data Frame. The sensor fields added to the frame are: Accelerometer values, RTC internal temperature value, and battery level. In the case the XBee is not using DigiMesh protocol, then the MAC address is added (because of length constraints). For further information, please check the Wasmote Data Frame Guide in:

<http://www.libelium.com/development/wasmote/documentation/programming>

Example:

```
~\0x00I\0x90\0x00}3\0xa2\0x00@z\0xcb\0x92\0xd8\0xd3\0x02<=>\0x80\0x03#35689722##7#ACC:80;10;987#IN_  
TEMP:22.50#BAT:93#\0xb4
```

Initially there are some hexadecimal characters, which belong to the XBee API frame, followed by the message. In the above example the message is:

```
<=>\0x80\0x03#35689722##7#ACC:80;10;987#IN_TEMP:22.50#BAT:93#
```

5.2. In Meshlium

When you buy a kit containing Meshlium and Waspote, your Waspotes already come configured to send frames to the Gateway. Later, once the user has developed the code for transmitting to Gateway, he can switch to Meshlium.

Before send frames to Meshlium, We recommend you read and study networking guides for your module:

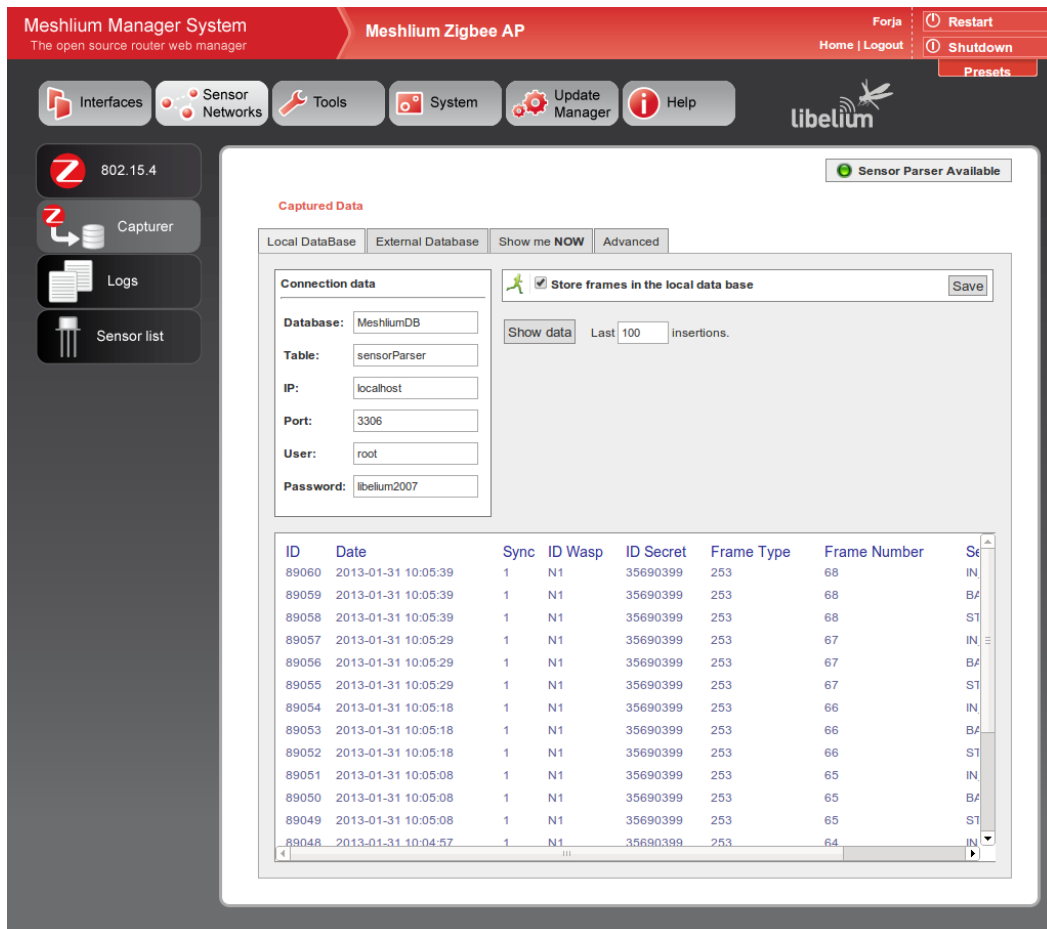
- Waspote 802.15.4 Networking Guide
- Waspote 868MHz Networking Guide
- Waspote 900MHz Networking Guide
- Waspote Digimesh Networking Guide
- Waspote ZigBee Networking Guide

Meshlium will receive the sensor data sent by Waspote using the XBee radio and it will store the frames in the Local Database. That can be done in an automatic way now thanks to the new **Sensor Parser**.

The **Sensor Parser** is a new feature for Meshlium (version 3.0.5 or older). It is a new software system which is able to do the following tasks in an easy and transparent way:

- receive frames from XBee (with the Data Frame format)
- parse these frames
- store the data in local Database
- synchronize the local Database with an external Database

Besides, the user can add his own sensors.



Meshlium Manager System
The open source router web manager

Meshlium Zigbee AP

Forja | Home | Logout | Restart | Shutdown | Presets

Interfaces | Sensor Networks | Tools | System | Update Manager | Help

libelium

Sensor Parser Available

Captured Data

Local DataBase | External Database | Show me NOW | Advanced

Connection data

Database: MeshliumDB

Table: sensorParser

IP: localhost

Port: 3306

User: root

Password: libelium2007

☒ Store frames in the local data base

Show data | Last 100 | Insertions.

ID	Date	Sync	ID Wasp	ID Secret	Frame Type	Frame Number	Status
89060	2013-01-31 10:05:39	1	N1	35690399	253	68	IN
89059	2013-01-31 10:05:39	1	N1	35690399	253	68	BA
89058	2013-01-31 10:05:39	1	N1	35690399	253	68	ST
89057	2013-01-31 10:05:29	1	N1	35690399	253	67	IN
89056	2013-01-31 10:05:29	1	N1	35690399	253	67	BA
89055	2013-01-31 10:05:29	1	N1	35690399	253	67	ST
89054	2013-01-31 10:05:18	1	N1	35690399	253	66	IN
89053	2013-01-31 10:05:18	1	N1	35690399	253	66	BA
89052	2013-01-31 10:05:18	1	N1	35690399	253	66	ST
89051	2013-01-31 10:05:08	1	N1	35690399	253	65	IN
89050	2013-01-31 10:05:08	1	N1	35690399	253	65	BA
89049	2013-01-31 10:05:08	1	N1	35690399	253	65	ST
89048	2013-01-31 10:04:57	1	N1	35690399	253	64	IN

© Libelium Comunicaciones Distribuidas S.L. | Terms of use

6. Compiling a New Program

To use the Waspmote-IDE compiler we must run the executable script called 'Waspmote', which is in the folder where the compiler has been installed.

Waspmote is divided into 4 main parts which can be seen in the following figure.

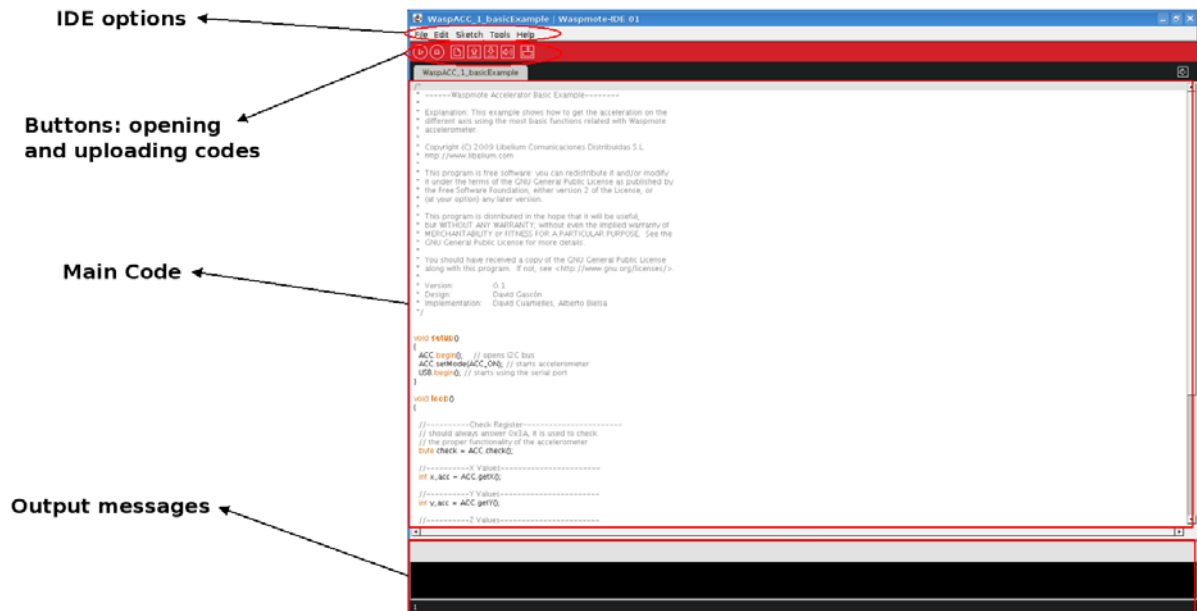


Figure 6: IDE – Waspmote parts

- The first part is the menu which allows configuration of general parameters such as the selected serial port.
- The second part is a button menu which allows verification, opening, saving or loading the selected code on the board.
- The third part contains the main code which will be loaded to Waspmote.
- The fourth part shows us the possible compilation and load errors, as well as the success messages if the process is carried out satisfactorily.

The Waspmote-IDE buttons panel allows certain functions to be carried out such as opening a previously saved code, creating a new one or loading the code on the board. The following figure shows the panel and the functions of each button.

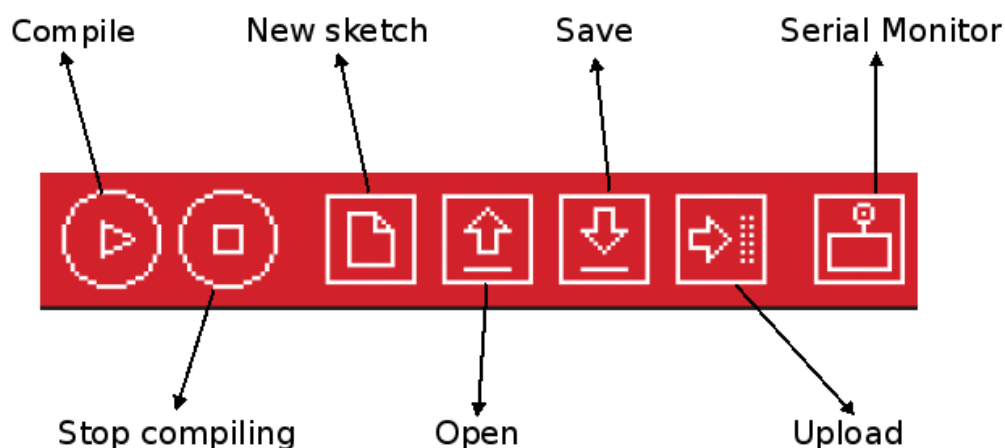


Figure 7: IDE – Waspmote panel of buttons

Once the program has been opened correctly some configuration changes must be made so that the programs load correctly in Waspmote.

In the **'Tools/Board'** tab the Waspmote board must be selected. This refers to the API selected.

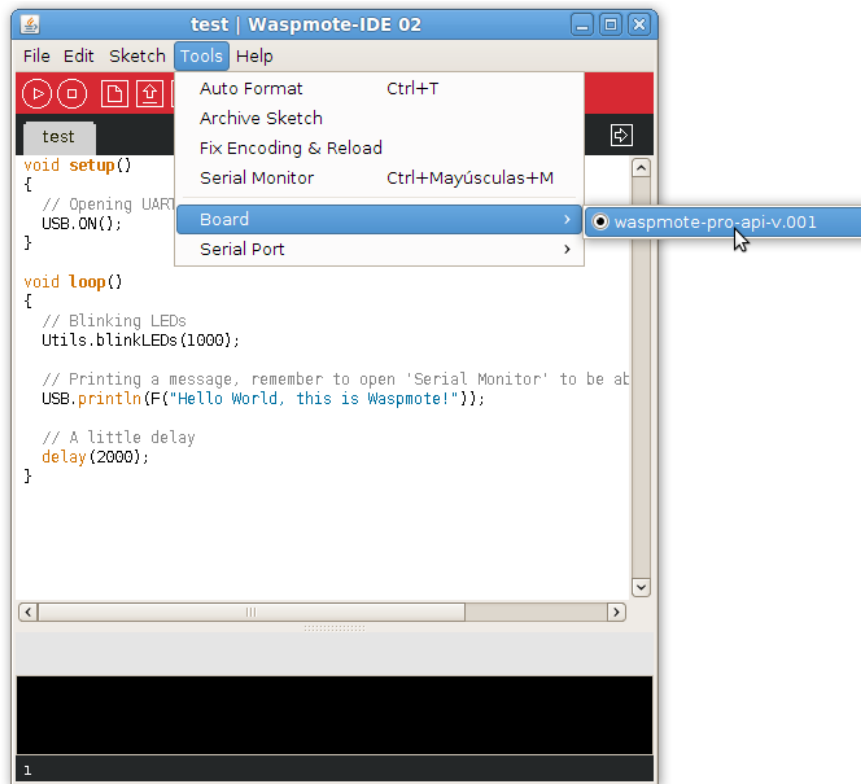


Figure 8: Select API

In the **'Tools/Serial Port'** tab, the USB to which Waspmote has been connected to the computer must be selected.

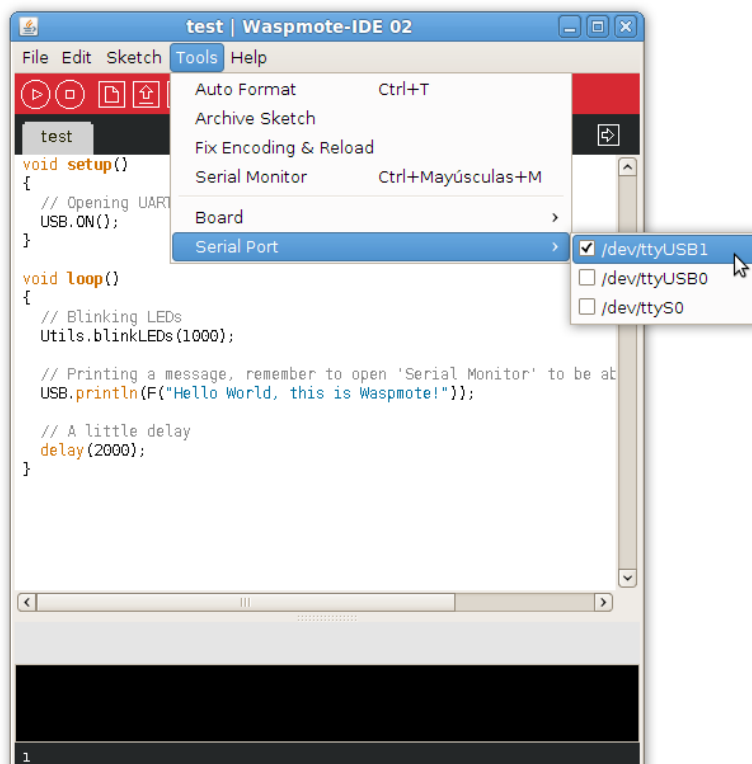


Figure 9: Select USB port

Once these 2 parameters have been configured we can load a program onto Wasmote. The process will be explained using a very simple example. A series of examples for learning and familiarizing yourself with the Wasmote environment have been included in the downloaded file that contains the compiler.

The simplest example is the file called 'test.pde'. In this example the text string "Hello World!" appears on the screen. The example shows how to load a program onto Wasmote and how to show information on the screen.

The next step is to configure the folder where the created programs are going to be saved. In the Wasmote-IDE this folder is called '**sketchbook**' and can be configured by accessing the '**File/Preferences**' tab. Clicking on this tab will open a new window where the location of the sketchbook can be indicated. Once the sketchbook folder path is indicated, the downloaded test program must be saved in this folder.

Wasmote-IDE must be closed so that the changes and the newly saved program in the sketchbook folder are reflected.

Run Wasmote again and open the downloaded test program by clicking on '**Open**'.

Select the 'test.pde' file in the path where it has been unzipped and open it. As can be seen, it is a very simple code which lights up a LED every 3 seconds and writes "Hello World!" on the screen.

The next step is to load the program onto Wasmote. To do this Wasmote must be connected to the computer through the USB and the button 'upload' must be clicked. Then, it will start compiling the program. When the program has been compiled correctly, a message will appear on the lower part of the window indicating this event. Conversely, if a fault occurs, red messages will appear indicating the bugs in the code. When compiling is over, the code will be loaded onto Wasmote.

When the program has been loaded correctly, a message appears in the Wasmote window indicating '**Done Uploading**'.

Conversely, if some problem occurs during loading, red messages will appear indicating the failures.

Once this program is loaded onto the board, the loaded code will run as was explained in the Architecture and System chapter.

Note: The Gateway is just a UART-USB bridge. This means that the Gateway cannot be programmed and no code can not be uploaded. Its function is to pass data from the XBee to the USB, and vice-versa.

7. Uploading a New Program to Wasmote

The following steps must be done each time we must upload code, always:

Step 1: Switch Wasmote ON (in the image, move the switch to the left).

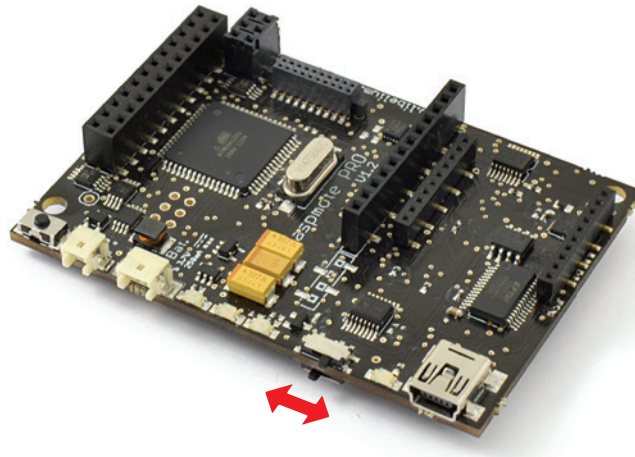


Figure 10: Switch Wasmote ON

Step 2: Connect Wasmote to your PC through the USB cable. Open the Wasmote's IDE and select the proper Board and Serial Port with in the "Tools" menu.

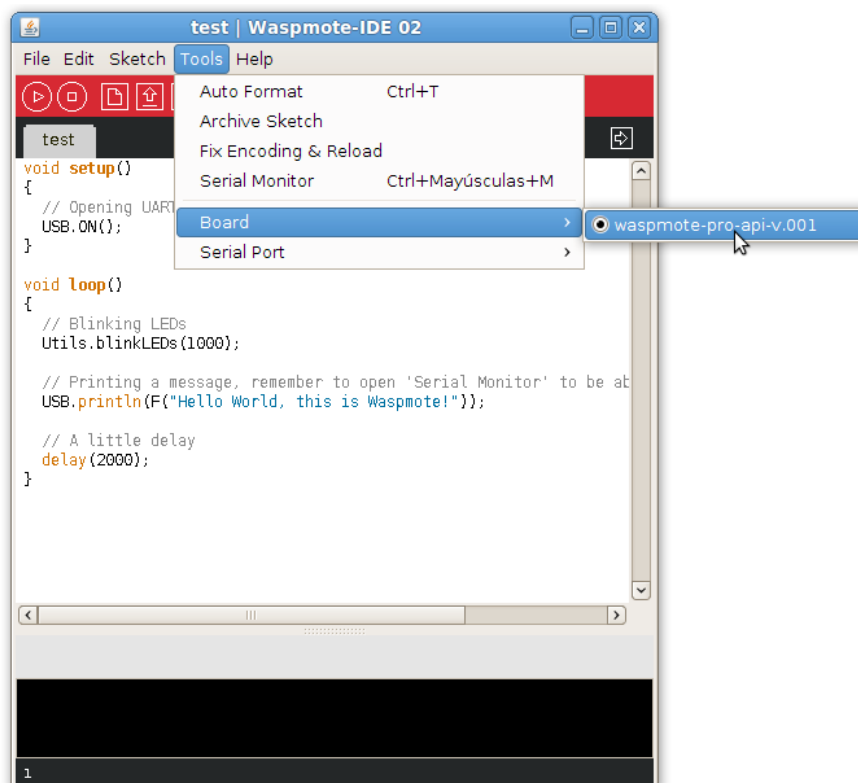


Figure 11: Select API

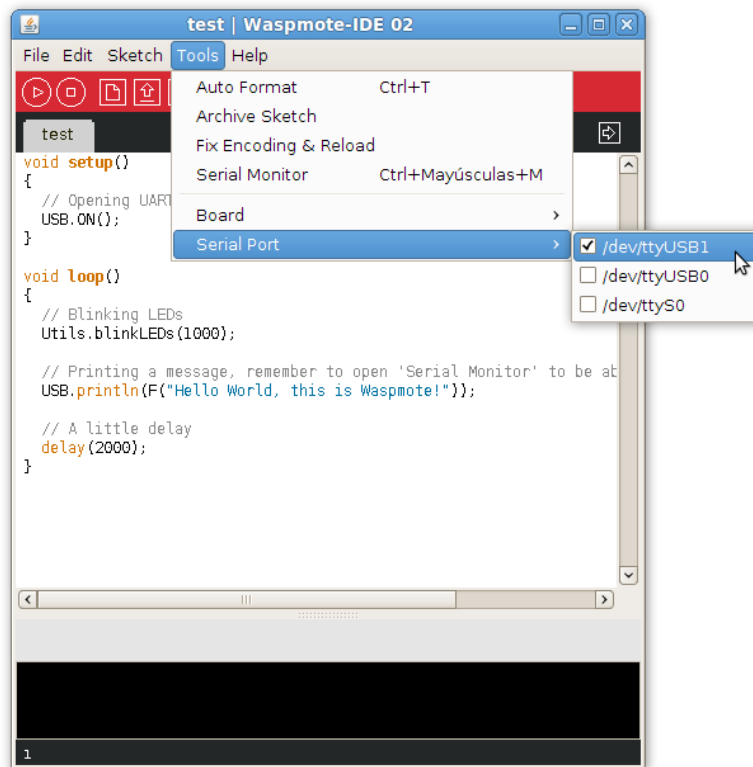


Figure 12: Select USB port

Step 3: Prepare your code for Waspmote. In our case, go to the template of the “hello_world” or copy and paste the text in the sketch.

Step 4: Save the sketch (the IDE has a button for that), for example with the name “hello_world”, and check the IDE states “Done Saving”.

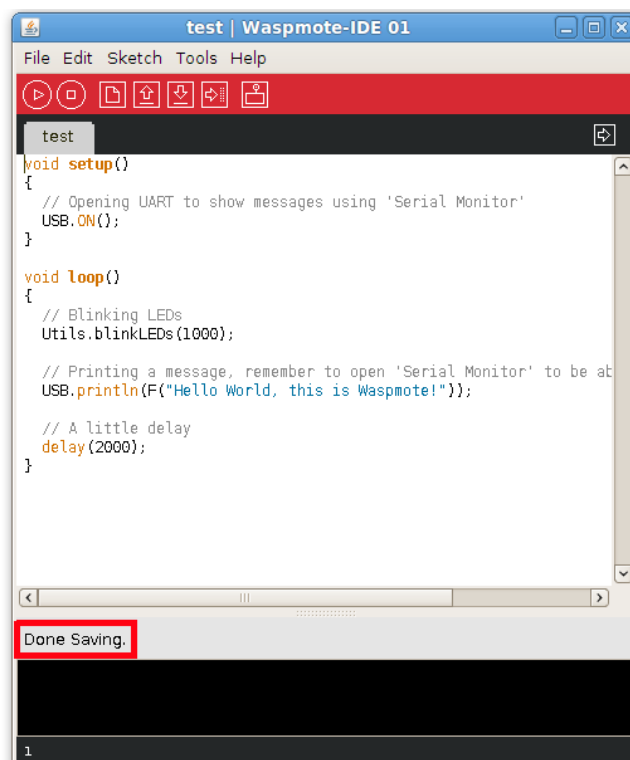


Figure 13: Save your program

Step 5: Compile the code (the IDE has a button for that), and check there are no errors or warnings. The IDE should say “Done Compiling”.

Step 6: Upload the code to Waspmote: click the “Upload” button and wait a few seconds until the process ends; check there are no error messages, just “Done uploading”.



Figure 14: Upload your program



Figure 15: Program uploading done

8. How to Use the Developer's Guides

Congratulations, if you arrived here, you completed the Quick Start Guide and now you know the basics about WaspMote.

Your next steps depend on what kind of project you want to develop. Normally, a program for WaspMote is composed of 4 parts:

1. configuration (RTC, sensors, communications module)
2. read sensor(s)
3. communications (XBee, GPRS, ...)
4. enter sleep mode

As a first step, you should read the WaspMote Technical Guide. It is a good introduction to WaspMote because it talks about all the features, modules and boards. Make sure you read the chapters about the features you need to use thoroughly.

RTC management or power/sleep features should be present in almost any type of project, so it is a good idea to start with them. Besides, they are pretty easy to control. You should read the RTC Guide and the Power Guide carefully. For any feature, module or board, there is a good number of examples, make sure you execute them in your WaspMote while you read the related guide. This will help to clear up things and you can learn concepts by doing and not just by reading. In any case, examples are arranged in increasing order of difficulty. This means you should start from the lower examples and then move to the advanced examples.

Reading sensors is also easy and it must be one of the first steps. The appropriate Sensor Guide must be read and then you can execute the related examples.

There are optional features or modules which should be studied now in the case you plan to use them: SD, accelerometer, EEPROM, etc.

We suggest to leave the communication part for the last steps, since it may be the most difficult part. Until that moment, it is a good idea to transmit your data via USB for the first steps of your development. Remember you can print as many messages as you want, so you can print helpful intermediate messages. Make sure you learn about the Frame, we advice to send frames in the official Frame format with your communication modules.

Once you arrived here, you can try to do your first complete program, combining RTC, power, sensors and communications. There are different combined examples which will show how to do it in a successful way.

Do not forget to read the Programming Guide as you advance in the project development. This Guide contains atomic tips which are useful for real-life projects. Some of the ideas shown there will help you to build successful projects.

A Gateway is always recommended. We believe it is better to start receiving frames with a Gateway, and once you feel comfortable with it, you can switch to Meshlium in the case you have one. Meshlium can be seen as an advanced Gateway. You can start by reading the Meshlium Quick Start Guide, and after that you can study the complete Meshlium Technical Guide.

If you want to use 2 communication radios, you should study them separately. First one of them; once you completed the development, continue with the 2nd one. Once you know both of them, you can try both of them in the same program. There are combined examples with 2 radios which may help.

Interruptions are one advanced feature so you can leave that for the last part of the learning process.

If you need OTA features, it is a complicated feature too and it could be a good moment now to start reading the OTA Guide and execute the examples.