Mini QRNG

By OpenQbit

Installation, configuration & administration.

# User manual

version 1.0 Beta

June 2020.

## Content

1. Introduction. ....3
2. What is Blockly programming? ....4
3. What is Termux? ....4
4. What is Mini QRNG? ....4
5. Storage configuration within Termux. ....8
6. SSH (Secure Shell) server installation. ....9
7. SSH server configuration on mobile phone (smartphone). ....10
8. Ambientes Blockly (App Inventor, AppyBuilder y Thunkable). ....16
9. Definition and use of blocks in Mini QRNG. ....18
10. Creation of "Hardware" device of a QRNG. ....29
11. Annex "OpenQbit Quantum Computing". ....35
12. Licensing and use of software. ....40

## 1. Introduction.

Today's cryptography is based on random number sequences. The pseudo-random number generators currently in use appear to provide random bit sequences, but in reality these bit sequences have certain patterns, so there is a risk of being hacked or otherwise manipulated to make the information travel over public and pirated networks.

The integration of sources of physical entropy in random number generators is the most common method to overcome this security threat. However, classical physics is causal, so the unpredictability of a bit sequence generated with classical physics cannot be proven.

Quantum physics, on the other hand, is essentially random. The numbers generated by a quantum random number generator (QRNG) cannot be predicted: QRNG is demonstrably unpredictable. So, if a quantum random number generator is used in a security system, even a fast supercomputer with fast arithmetic operation cannot predict the random bit sequences used by this security system.

Quantum physics uses methods based on a fundamental concept called entropy.

TYPES OF ENTROPY

There are two general types of entropy sources that can be measured to generate true random numbers. The first type includes a physical process that is difficult or impossible to measure or too computationally intensive to predict, or both. This is a source of caoticentropy. A common example known to most people is a lottery machine. A set of sequentially numbered balls are placed in a chamber and constantly mixed together by rotating the chamber or blowing air through the chamber. Several of the balls are allowed to fall out of the chamber and the numbers marked on the balls represent the lottery draw. The draw is random due to the large number of interactions between the balls and the camera resulting in a rapidly increasing number of possible movements of each ball. Not only is the complexity of these interactions extremely high, there is no apparent way to accurately observe or measure all the internal variables of the balls, the camera and the airflow. A second, very different type of source of entropy is quantum mechanics. Many microscopic particles or waves, such as photons, electrons, and protons have quantum mechanical properties that include rotation, polarization, position, and momentum. Given the proper configuration to produce these particles, the specific values of their rotation or polarization, for example, are not only unknown and theoretically unpredictable, but are physically determined until a measurement is made.

## 2. What is Blockly programming?

Blockly is a visual programming language composed of a simple set of commands that we can combine as if they were the pieces of a puzzle. It is a very useful tool for those who want to learn how to program in an intuitive and simple way or for those who already know how to program and want to see the potential of this type of programming.

Blockly is a form of programming where you don't need any background in any kind of computer language, this is because it is just joining graphic blocks as if we were playing lego or a puzzle, you just need to have some logic and that's it!

Anyone can create programs for mobile phones (smartphones) without messing with those programming languages difficult to understand, just put together blocks in a graphical way in a simple, easy and fast way to create.

## 3. What is Termux?

Termux is an Android terminal emulator and a Linux environment application that works directly without the need for routing or configuration. A minimal base system is automatically installed.

We will use Termux for its stability and easy installation and management, however, you can use an installed environment of Ubuntu Linux for Android.

In this Linux environment you will have the "core" of the communication processes of the MiniQRNG.

## 4. What is Mini QRNG?

Mini QRNG is Software and Hardware that includes three technological solutions to create QRNGs (Quantum Random Number Generators). Classified as follows:

a.- QRNG API. - Quantum random number generator obtained from external servers.

b.- MiniQRNG / Software. - Random quantum number generator obtained using the physical properties (quantum) of the mobile phone camera.

c.- MiniQRNG / Hardware. - Quantum number generator obtained using hardware based on the quantum physics properties of a laser. Later on we will tell you how to build it at home at a low cost.

1. Installation and configuration of Termux Terminal.

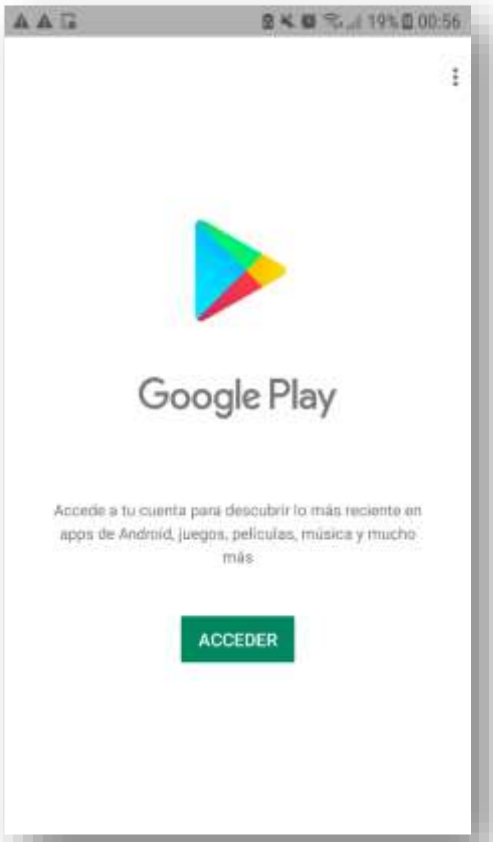
First we need a Linux environment since every Android system is based on Linux for security and flexibility in tools, we will use the "Termux" terminal that contains that environment where we will install the tool(s) that will help us to create QRNGs.

Termux is a Linux emulator where we will install the necessary packages to create quantum numbers.

One of the main advantages of using Termux is that you can install programs without having to "rotate" the mobile phone (Smartphone). This ensures that no manufacturer's warranty is lost due to this installation.

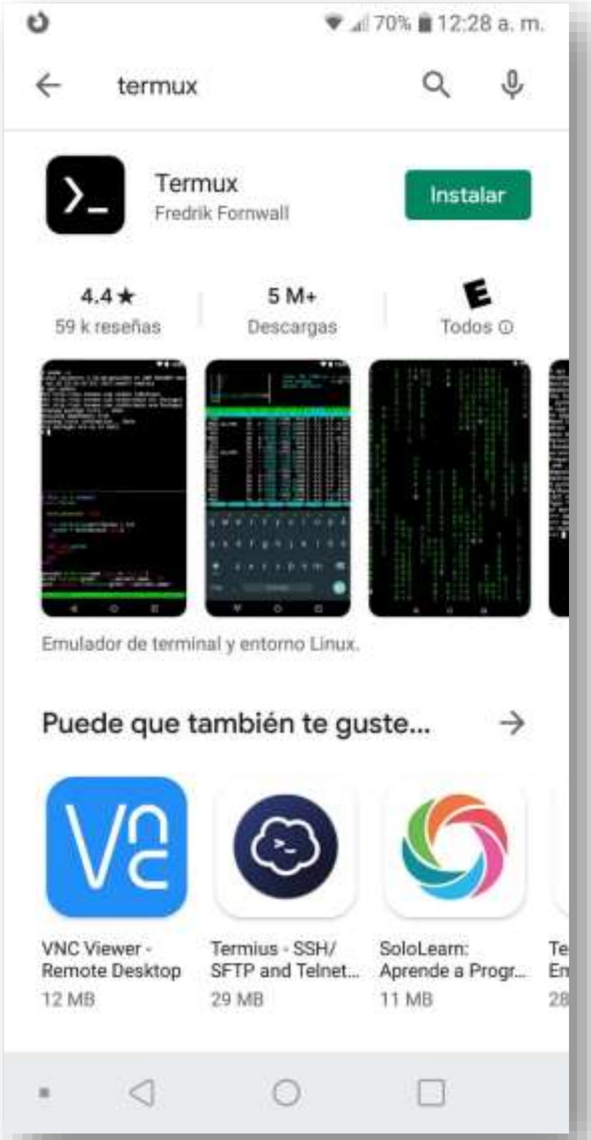
Termux installation.

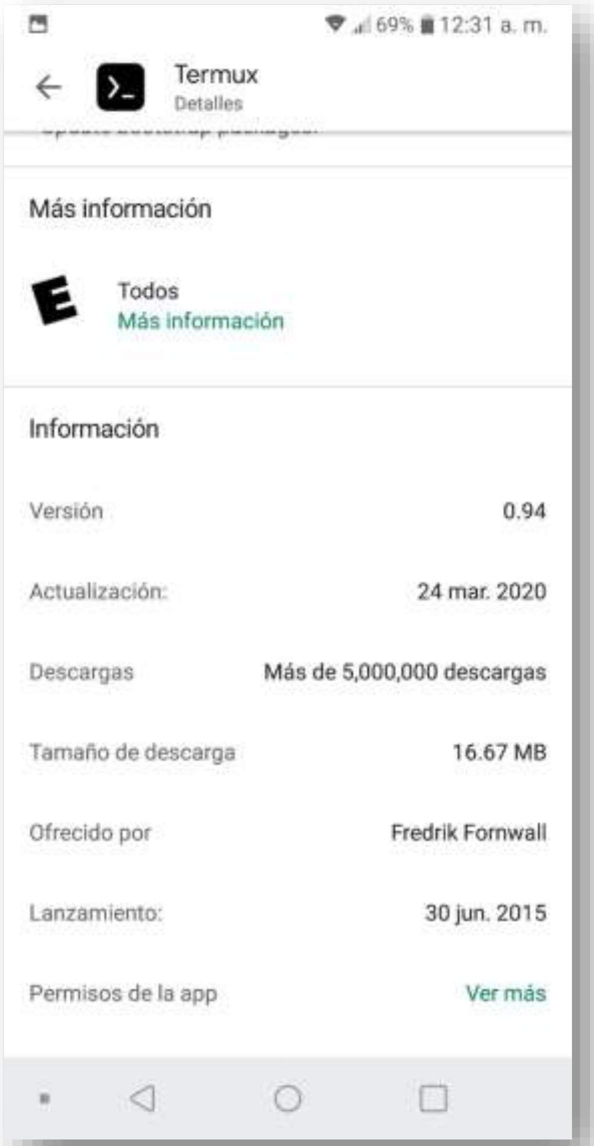
From your mobile, go to the Google Pay icon application (play.google.com).



Accede a tu cuenta para descubrir lo más reciente en apps de Android, juegos, películas, música y mucho más

Search by application "Termux", select it and start the installation process.


70% 12:28 a. m.
termux
Termux
Fredrik Fornwall
Instalar
4.4★
59 k reseñas
5 M+
Descargas
Todos
Emulador de terminal y entorno Linux.
Puede que también te guste...
VNC Viewer - Remote Desktop
12 MB
Termius - SSH/ SFTP and Telnet...
29 MB
SoloLearn: Aprende a Progr...
11 MB



Información

| | |
|---|---|
| Versión | 0.94 |
| Actualización: | 24 mar. 2020 |
| Descargas | Más de 5,000,000 descargas |
| Tamaño de descarga | 16.67 MB |
| Ofrecido por | Fredrik Fornwall |
| Lanzamiento: | 30 jun. 2015 |
| Permisos de la app | Ver más |

## Start of the Termux application.

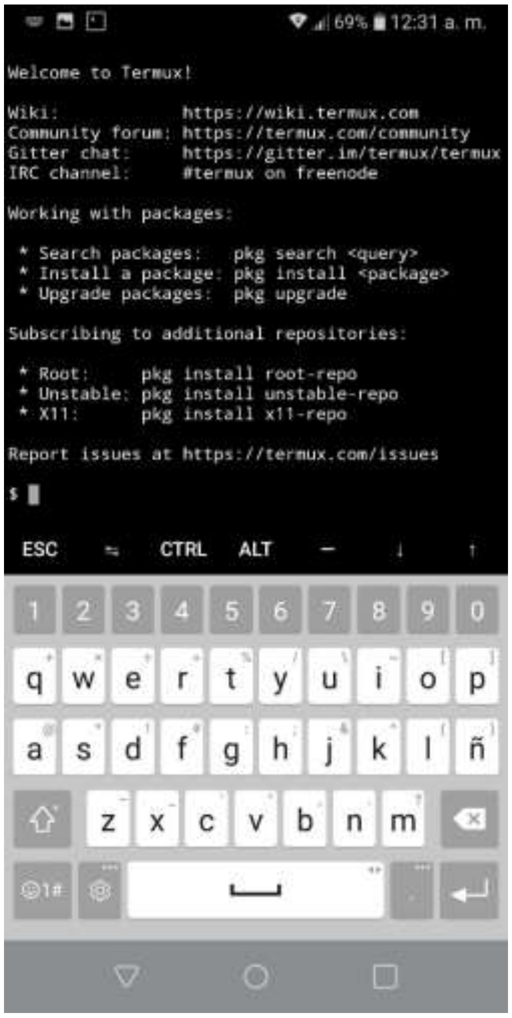
After starting we will have to execute the following two commands to perform updates of the Linux operating system emulator:

$ apt update

$ apt upgrade

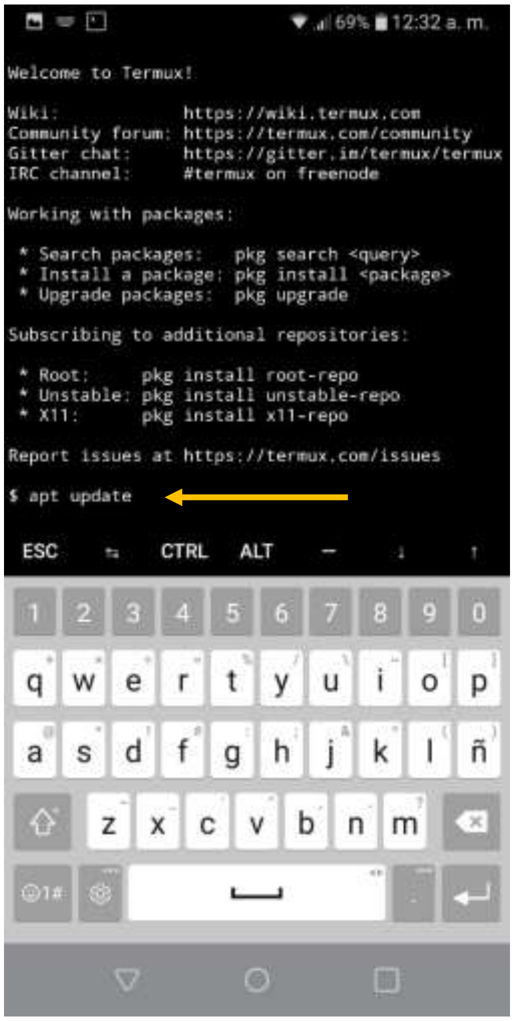
Confirm all options Y(Yes)...

Termux



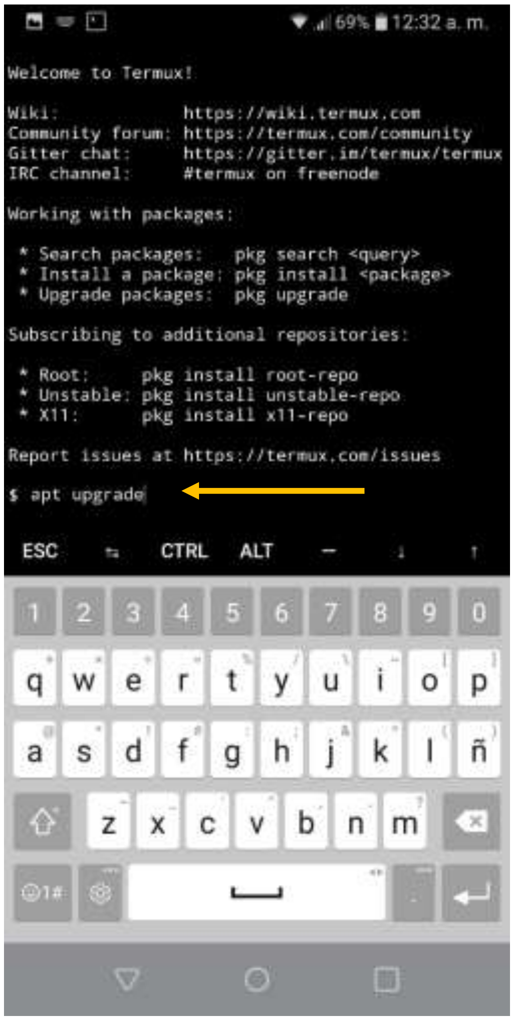
```
Welcome to Termux!

Wiki:            https://wiki.termux.com
Community forum: https://termux.com/community
Gitter chat:     https://gitter.im/termux/termux
IRC channel:     #termux on freenode

Working with packages:

 * Search packages:   pkg search <query>
 * Install a package: pkg install <package>
 * Upgrade packages:  pkg upgrade

Subscribing to additional repositories:

 * Root:     pkg install root-repo
 * Unstable: pkg install unstable-repo
 * X11:      pkg install x11-repo

Report issues at https://termux.com/issues

$
```

Home $ apt update



```
Welcome to Termux!

Wiki:            https://wiki.termux.com
Community forum: https://termux.com/community
Gitter chat:     https://gitter.im/termux/termux
IRC channel:     #termux on freenode

Working with packages:

 * Search packages:   pkg search <query>
 * Install a package: pkg install <package>
 * Upgrade packages:  pkg upgrade

Subscribing to additional repositories:

 * Root:     pkg install root-repo
 * Unstable: pkg install unstable-repo
 * X11:      pkg install x11-repo

Report issues at https://termux.com/issues

$ apt update
```

$ apt upgrade



```
Welcome to Termux!

Wiki:            https://wiki.termux.com
Community forum: https://termux.com/community
Gitter chat:     https://gitter.im/termux/termux
IRC channel:     #termux on freenode

Working with packages:

 * Search packages:   pkg search <query>
 * Install a package: pkg install <package>
 * Upgrade packages:  pkg upgrade

Subscribing to additional repositories:

 * Root:     pkg install root-repo
 * Unstable: pkg install unstable-repo
 * X11:      pkg install x11-repo

Report issues at https://termux.com/issues

$ apt upgrade
```

## 5. Storage configuration within Termux.

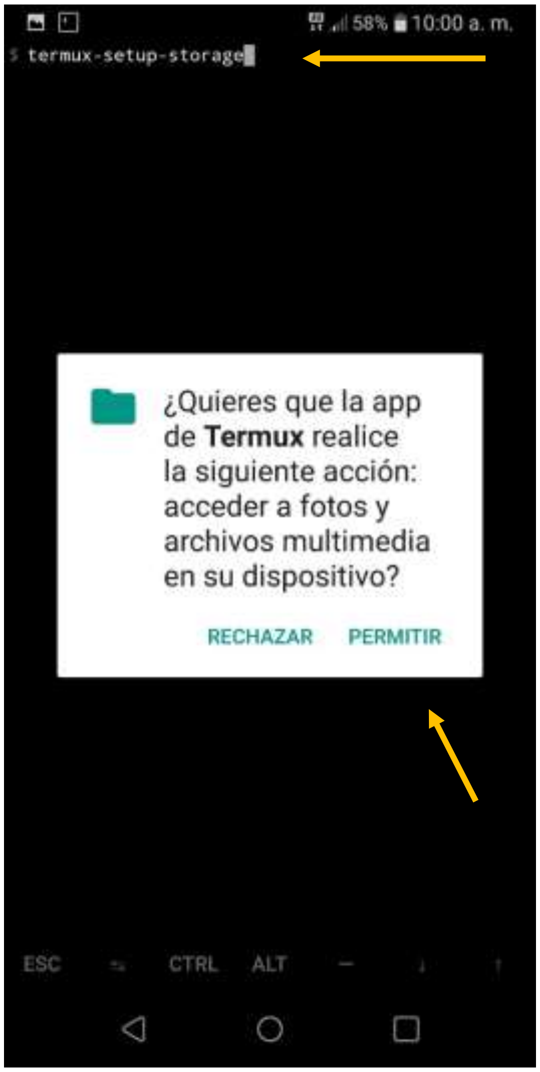
After you have updated and upgraded the Termux system, we will start configuring how to view the internal storage of the phone in the Termux system. This will help you to be able to exchange information between Termux and our information in the phone.

This can be done simply and quickly by running the following command on a Termux terminal.

**$ termux-setup-storage**

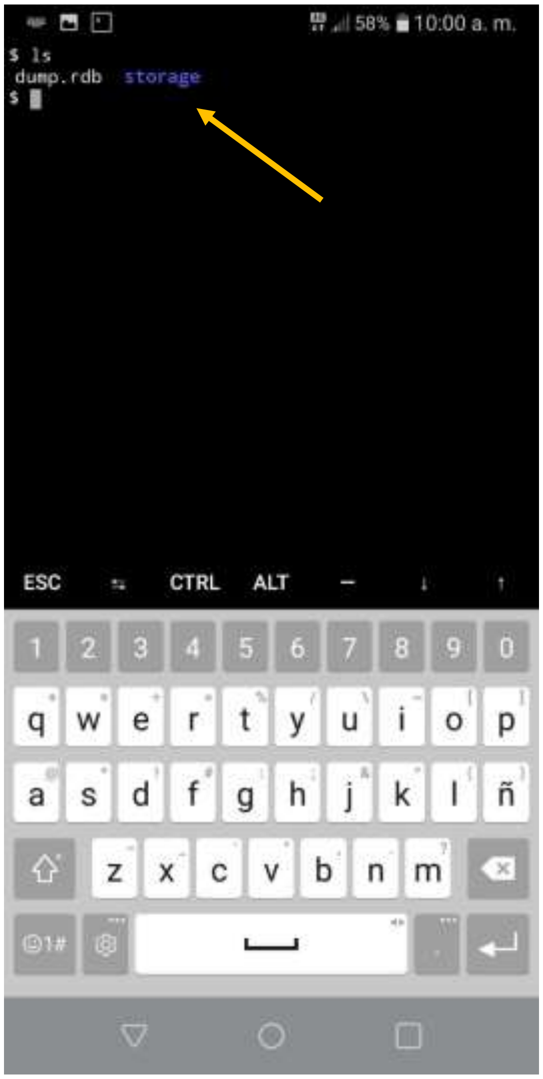
When you execute the previous command, a window appears asking you to confirm the creation of a virtual **storage** (directory) in Termux. We verify by giving the command:

**$ ls**



```
$ termux-setup-storage
```

¿Quieres que la app de **Termux** realice la siguiente acción: acceder a fotos y archivos multimedia en su dispositivo?



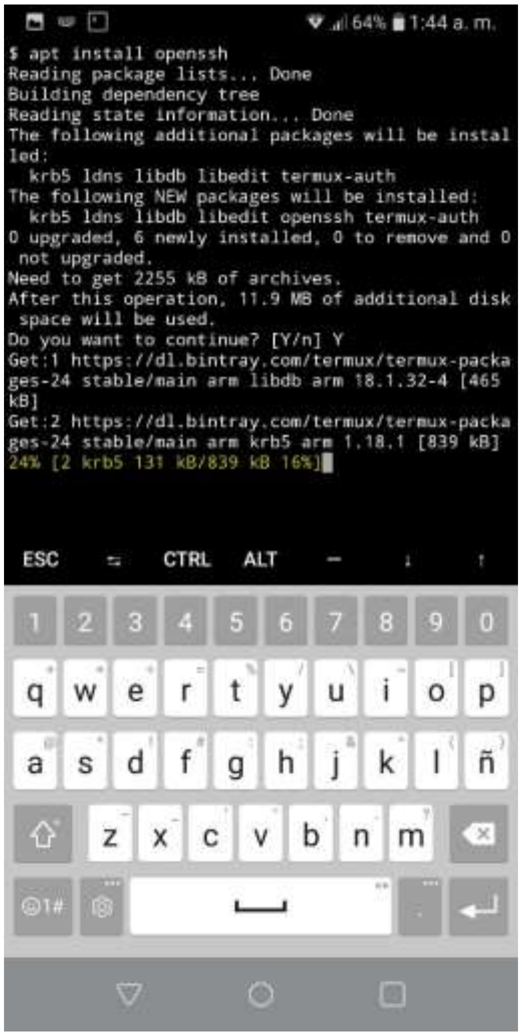
```
$ ls
dump.rdb  storage
$
```

## 6. SSH (Secure Shell) server installation.

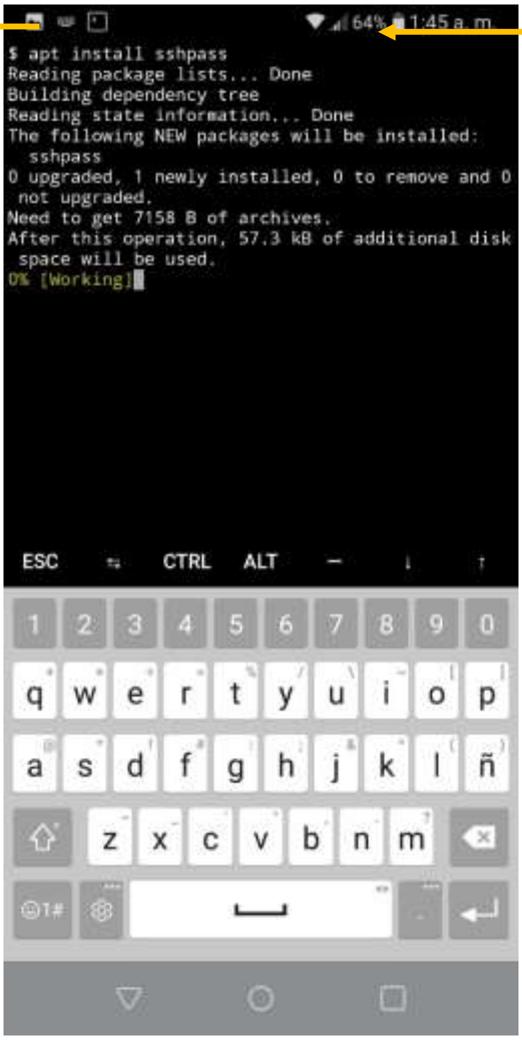
$ apt install openssh

$ apt install sshpass

$ apt install openssh $ apt install sshpass



```
$ apt install openssh
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following additional packages will be instal
led:
  krb5 ldns libdb libedit termux-auth
The following NEW packages will be installed:
  krb5 ldns libdb libedit openssh termux-auth
0 upgraded, 6 newly installed, 0 to remove and 0
 not upgraded.
Need to get 2255 kB of archives.
After this operation, 11.9 MB of additional disk
 space will be used.
Do you want to continue? [Y/n] Y
Get:1 https://dl.bintray.com/termux/termux-packa
ges-24 stable/main arm libdb arm 18.1.32-4 [465
kB]
Get:2 https://dl.bintray.com/termux/termux-packa
ges-24 stable/main arm krb5 arm 1.18.1 [839 kB]
24% [2 krb5 131 kB/839 kB 16%]
```



```
$ apt install sshpass
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following NEW packages will be installed:
  sshpass
0 upgraded, 1 newly installed, 0 to remove and 0
 not upgraded.
Need to get 7158 B of archives.
After this operation, 57.3 kB of additional disk
 space will be used.
0% [Working]
```

We have finished with the installation of the communication network for localhost SSH server on mobile Smartphone.

## 7. SSH server configuration on mobile phone (smartphone).

We will enable the SSH server in the mobile phone to be able to connect from our PC to the mobile phone and to be able to work in a faster and more comfortable way, also it will serve us to check that the service of the SSH server in the mobile phone works correctly since we will use it in the communication with Mini QRNG.

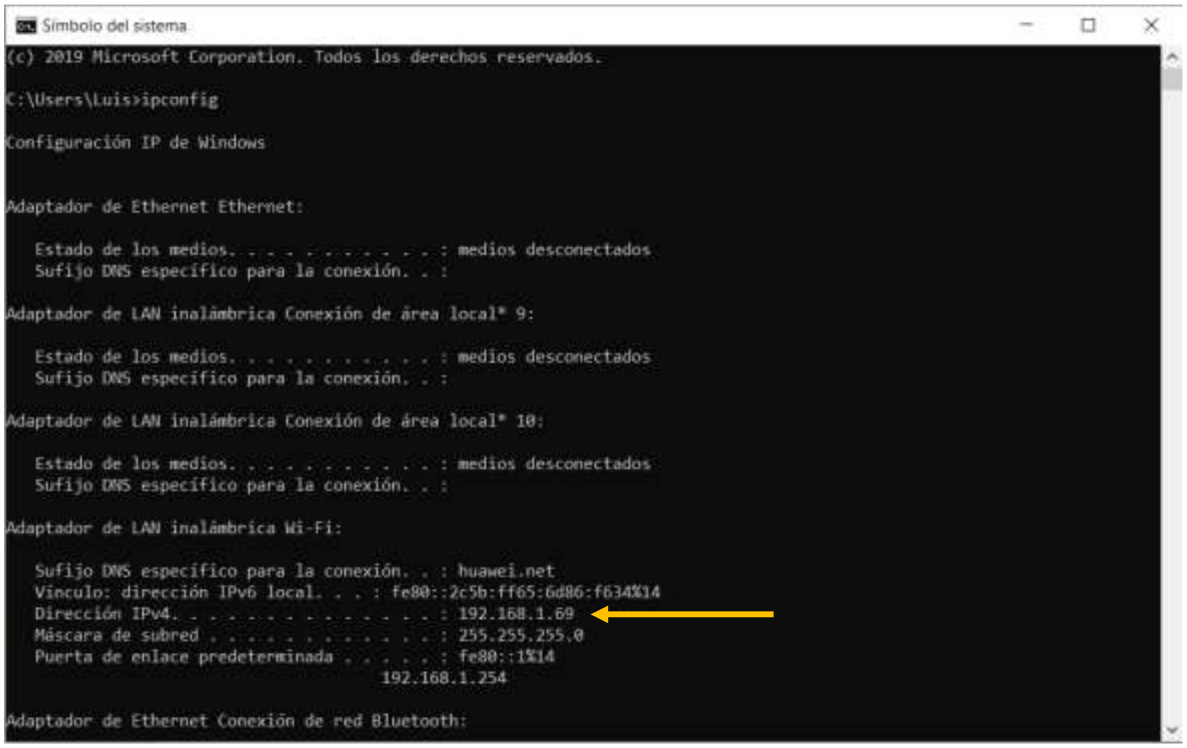
The first thing we have to do is connect the mobile and the PC to the **same WiFi network** so that they can see each other. The IPs or addresses must be similar to 192.168.XXX.XXX the XXX values are variable numbers that are assigned randomly in each computer.

This example was tested on an LG Q6 mobile phone and a PC with Windows 10 Home.

Check the IP or address that the PC has connected to the WiFi we must open a terminal in Windows.

In the bottom panel where the search magnifier is write cmd and press the Enter key. A terminal will open and in it we write the command:

C:User_Name> ipconfig



```
(c) 2019 Microsoft Corporation. Todos los derechos reservados.

C:\Users\Luis>ipconfig

Configuración IP de Windows


Adaptador de Ethernet Ethernet:

   Estado de los medios. . . . . . . . . . . : medios desconectados
   Sufijo DNS específico para la conexión. . :

Adaptador de LAN inalámbrica Conexión de área local* 9:

   Estado de los medios. . . . . . . . . . . : medios desconectados
   Sufijo DNS específico para la conexión. . :

Adaptador de LAN inalámbrica Conexión de área local* 10:

   Estado de los medios. . . . . . . . . . . : medios desconectados
   Sufijo DNS específico para la conexión. . :

Adaptador de LAN inalámbrica Wi-Fi:

   Sufijo DNS específico para la conexión. . : huawei.net
   Vínculo: dirección IPv6 local. . . : fe80::2c5b:ff65:6d86:f634%14
   Dirección IPv4. . . . . . . . . . . . . . : 192.168.1.69
   Máscara de subred . . . . . . . . . . . . : 255.255.255.0
   Puerta de enlace predeterminada . . . . . : fe80::1%14
                                       192.168.1.254

Adaptador de Ethernet Conexión de red Bluetooth:
```

It will show us the IP assigned to the PC in this is 192.168.1.69 but this is most likely to be different in each case.

NOTE: The address where it says "IPv4 address" should be taken, not to be confused with the Gateway.

Now in the case of the mobile phone in the Termux terminal we must type the following command to know the name of our user that we will use to connect to the SSH server that has our phone, we execute the following command:

```
$ whoami
```

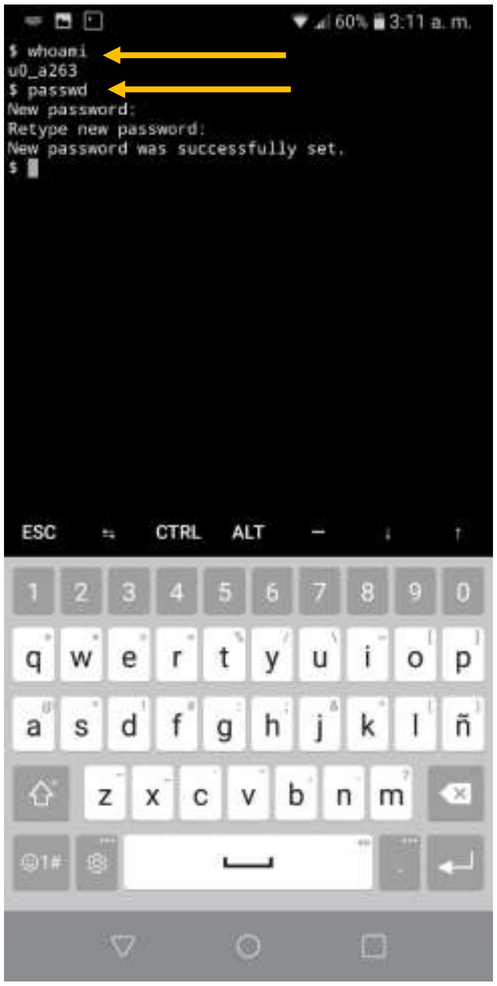
Later we must give a password to this user so we have to execute the following command:

```
$ passwd
```

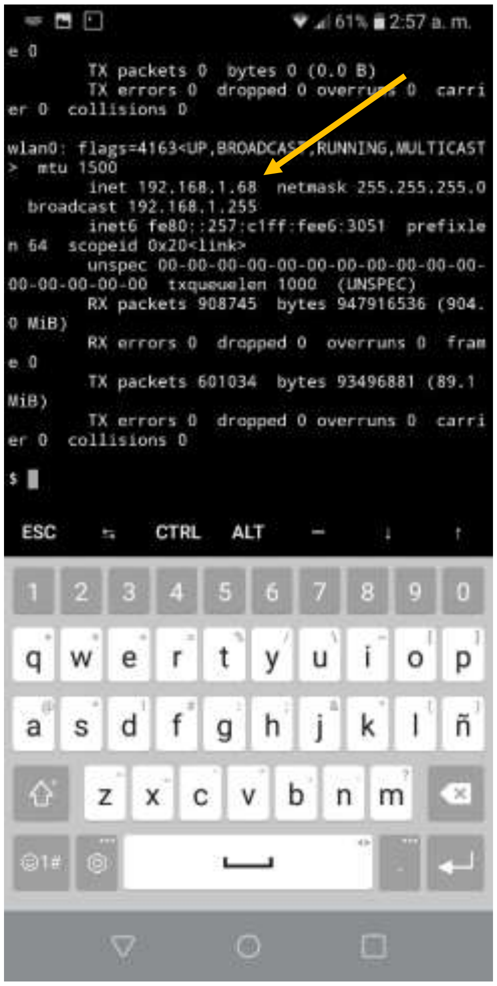
It will ask us to type a password and hit Enter, again it asks us for the password we confirm it and hit Enter, if it has been **successfully "New password was successfully set" in** case of marking an error is possible that the password has not been typed correctly. Perform the procedure again.

And then to know what IP we have in Termux we type the following command, the IP is after the word "**inet**":

```
$ ifconfig -a
```



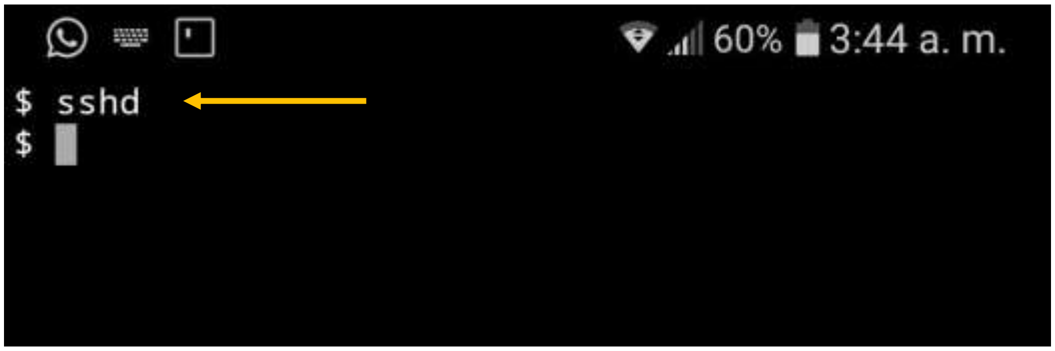
```
$ whoami
u0_a263
$ passwd
New password:
Retype new password:
New password was successfully set.
$ 
```



```
e 0
        TX packets 0  bytes 0 (0.0 B)
        TX errors 0  dropped 0 overruns 0  carri
er 0  collisions 0

wlan0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST
>  mtu 1500
        inet 192.168.1.68  netmask 255.255.255.0
  broadcast 192.168.1.255
        inet6 fe80::257:c1ff:fee6:3051  prefixle
n 64  scopeid 0x20<link>
        unspec 00-00-00-00-00-00-00-00-00-00-00-
00-00-00-00-00  txqueuelen 1000  (UNSPEC)
        RX packets 908745  bytes 947916536 (904.
0 MiB)
        RX errors 0  dropped 0  overruns 0  fram
e 0
        TX packets 601034  bytes 93496881 (89.1
MiB)
        TX errors 0  dropped 0 overruns 0  carri
er 0  collisions 0

$ 
```

Now it's time to start the SSH server service on your phone so you can receive sessions from your PC. We execute the following command in the Termux terminal, this command does not give any result.

$ sshd

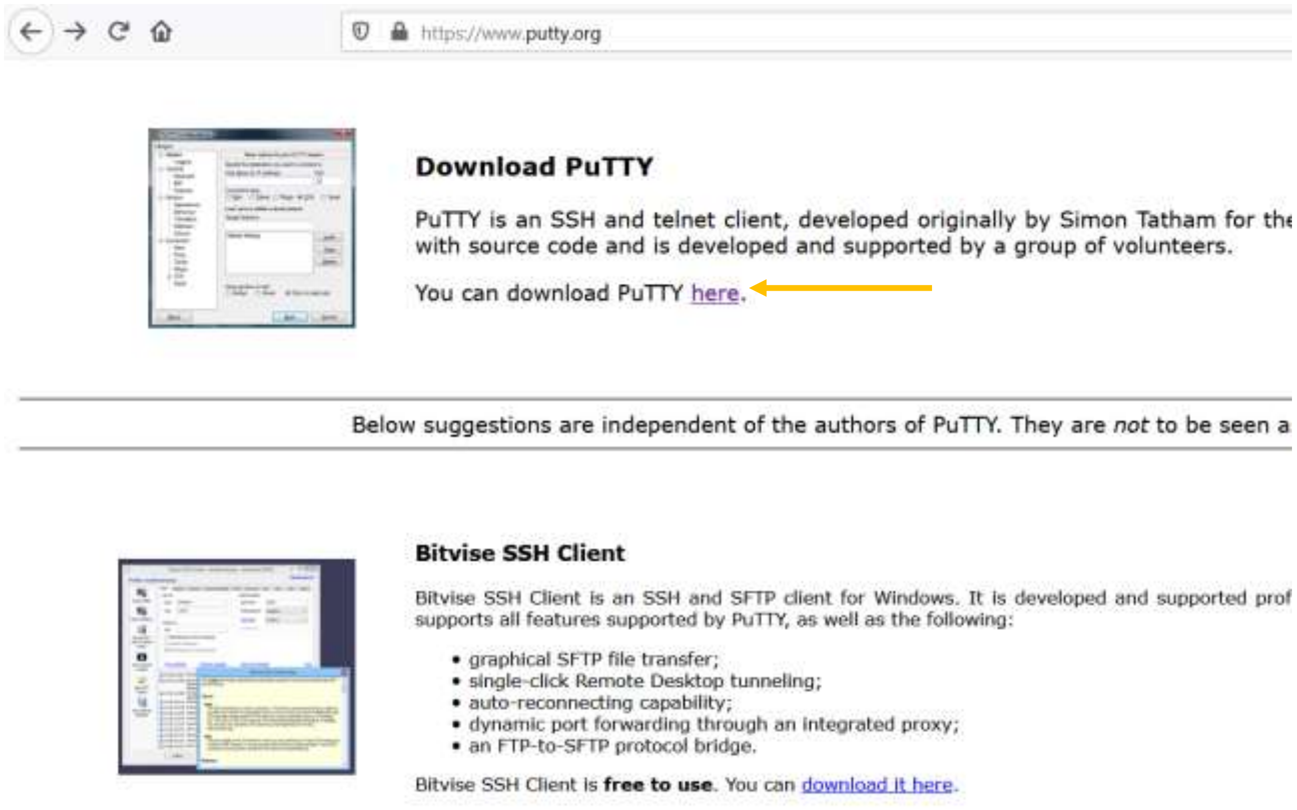


```
$ sshd
$ █
```

Now we will have to install a program on the PC that will communicate with the phone's SSH server from the PC.

We have to go to [https://www.putty.org](https://www.putty.org)

Select where the link "You can download PuTTY here" is



**Download PuTTY**

PuTTY is an SSH and telnet client, developed originally by Simon Tatham for the
with source code and is developed and supported by a group of volunteers.

You can download PuTTY here.

Below suggestions are independent of the authors of PuTTY. They are *not* to be seen a

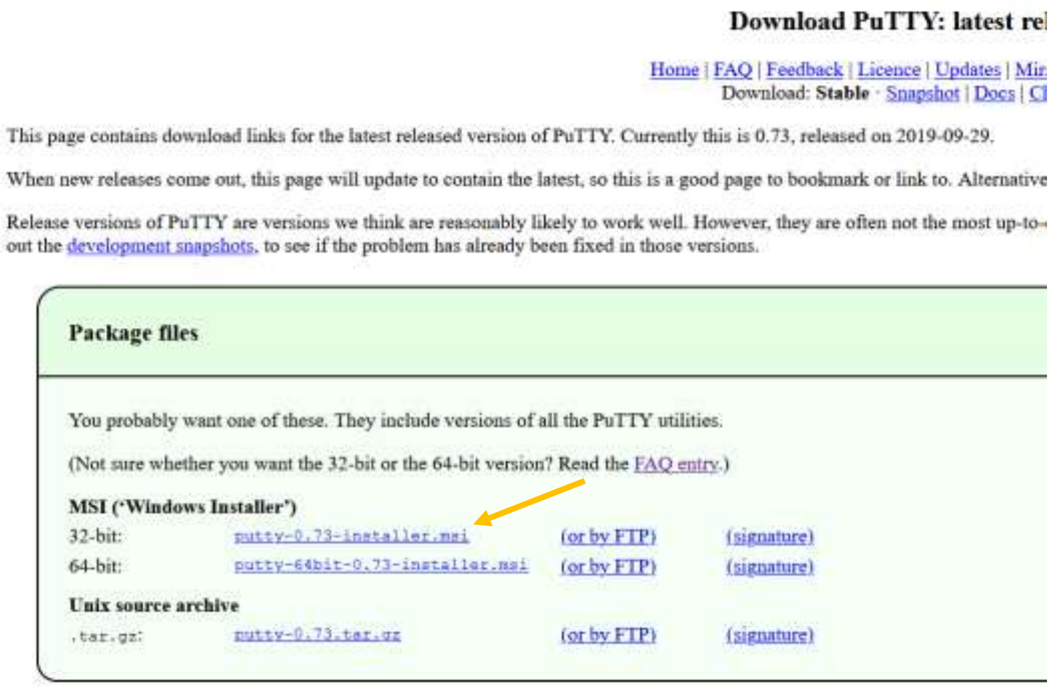
**Bitvise SSH Client**

Bitvise SSH Client is an SSH and SFTP client for Windows. It is developed and supported prof
supports all features supported by PuTTY, as well as the following:

- graphical SFTP file transfer;
- single-click Remote Desktop tunneling;
- auto-reconnecting capability;
- dynamic port forwarding through an integrated proxy;
- an FTP-to-SFTP protocol bridge.

Bitvise SSH Client is **free to use**. You can download it here.

Choose the 32-bit version, it doesn't matter if your system is 64-bit will work fine.



**Download PuTTY: latest rel**

Home | FAQ | Feedback | Licence | Updates | Mir
Download: **Stable** · Snapshot | Docs | Cl

This page contains download links for the latest released version of PuTTY. Currently this is 0.73, released on 2019-09-29.

When new releases come out, this page will update to contain the latest, so this is a good page to bookmark or link to. Alternative

Release versions of PuTTY are versions we think are reasonably likely to work well. However, they are often not the most up-to-
out the development snapshots, to see if the problem has already been fixed in those versions.

**Package files**

You probably want one of these. They include versions of all the PuTTY utilities.

(Not sure whether you want the 32-bit or the 64-bit version? Read the FAQ entry.)

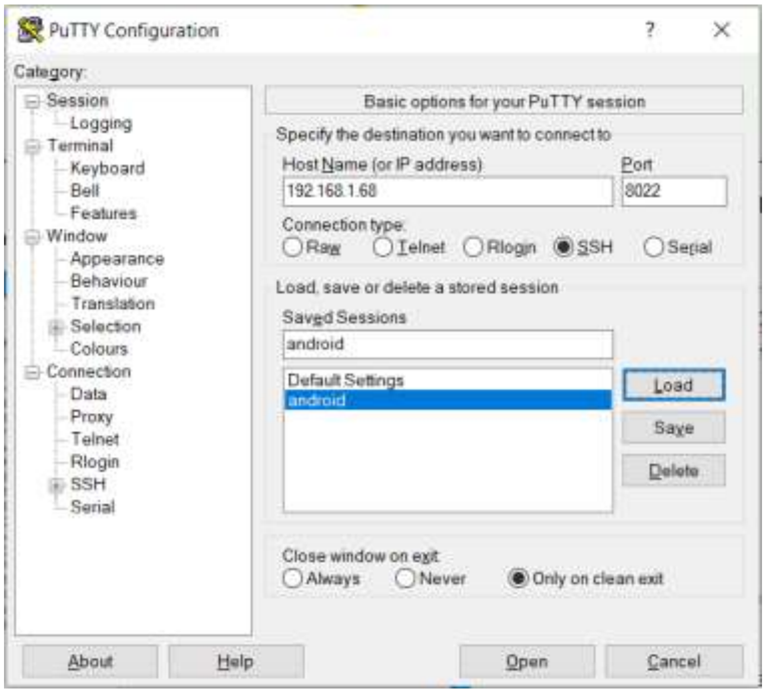
**MSI ('Windows Installer')**

| | | | |
|---|---|---|---|
| 32-bit: | putty-0.73-installer.msi | (or by FTP) | (signature) |
| 64-bit: | putty-64bit-0.73-installer.msi | (or by FTP) | (signature) |

**Unix source archive**

| | | | |
|---|---|---|---|
| .tar.gz: | putty-0.73.tar.gz | (or by FTP) | (signature) |

Once it has been downloaded to your PC, run it and install it with the default options. Then start the PuTTY application.



PuTTY Configuration

Basic options for your PuTTY session

Specify the destination you want to connect to

Host Name (or IP address): 192.168.1.68

Port: 8022

Connection type: ( ) Raw ( ) Telnet ( ) Rlogin (●) SSH ( ) Serial

Load, save or delete a stored session

Saved Sessions: android

Default Settings
android

Close window on exit: ( ) Always ( ) Never (●) Only on clean exit

In this session we will enter the data from our Openssh server that we installed in the mobile phone.

Enter the IP of the mobile phone.

HostName or IP address:

192.168.1.68 (IP example)

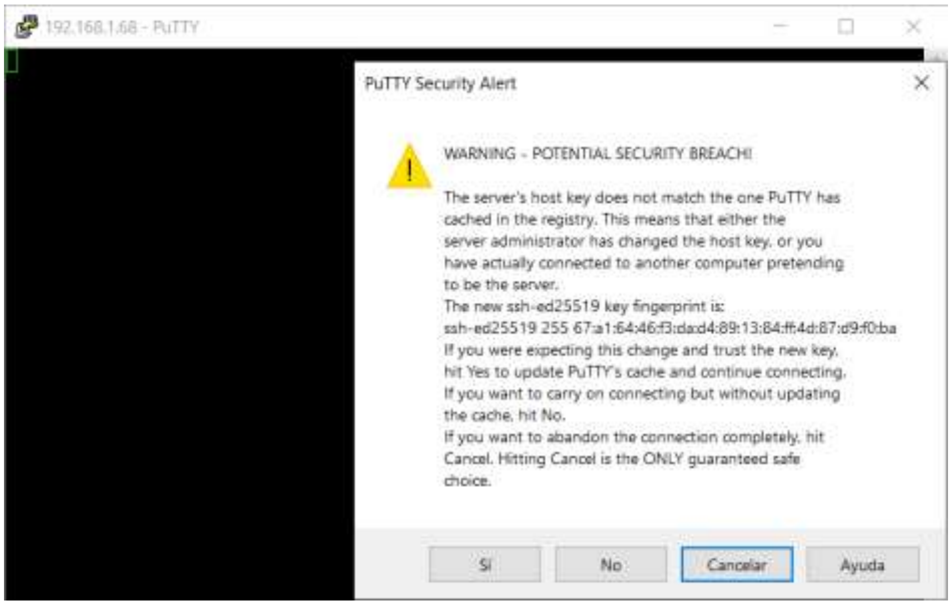
Port:

8022 (Default port of the mobile SSH server).

We can give a name to the session in "Saved Sessions" and click on the Save button.

Later on in the lower part we press to open a connection to the server giving the button "Open".

On the PC when you connect for the first time you **will be asked for confirmation of** the information encryption key by clicking on the "Yes" button.

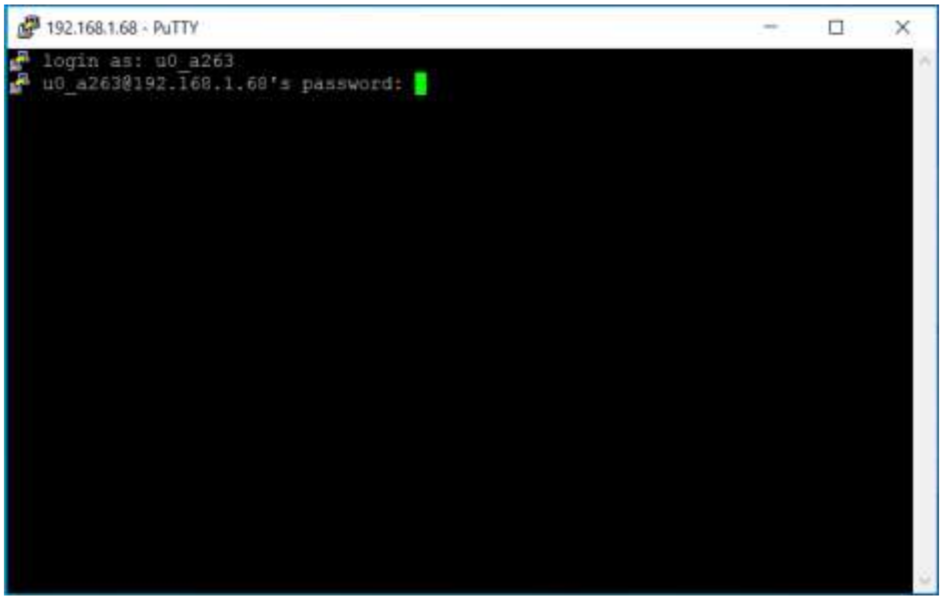


WARNING - POTENTIAL SECURITY BREACH!

The server's host key does not match the one PuTTY has cached in the registry. This means that either the server administrator has changed the host key, or you have actually connected to another computer pretending to be the server.
The new ssh-ed25519 key fingerprint is:
ssh-ed25519 255 67:a1:64:46:f3:da:d4:89:13:84:ff:4d:87:d9:f0:ba
If you were expecting this change and trust the new key, hit Yes to update PuTTY's cache and continue connecting.
If you want to carry on connecting but without updating the cache, hit No.
If you want to abandon the connection completely, hit Cancel. Hitting Cancel is the ONLY guaranteed safe choice.

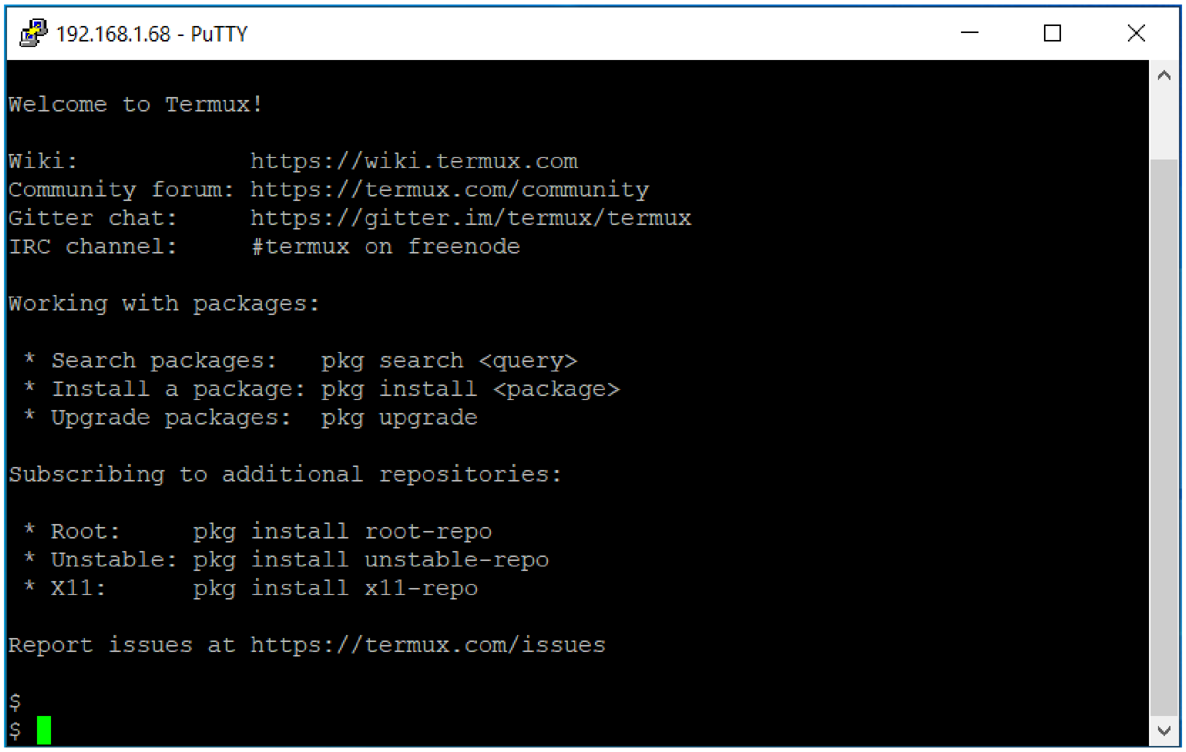
Later on, we will be asked for the user we are going to connect with. We will use the information that we previously obtained (user and password).

In the **Login as:** we must enter our user and give Enter, then we will ask for the password again give the Enter button.



```
login as: u0_a263
u0_a263@192.168.1.68's password:
```

If the data was correct, we will be in an SSH (Secure Shell) session performed from the PC (Client) on the phone (SSH Server).



```
Welcome to Termux!

Wiki:            https://wiki.termux.com
Community forum: https://termux.com/community
Gitter chat:     https://gitter.im/termux/termux
IRC channel:     #termux on freenode

Working with packages:

 * Search packages:   pkg search <query>
 * Install a package: pkg install <package>
 * Upgrade packages:  pkg upgrade

Subscribing to additional repositories:

 * Root:     pkg install root-repo
 * Unstable: pkg install unstable-repo
 * X11:      pkg install x11-repo

Report issues at https://termux.com/issues

$
$
```

**IMPORTANT NOTE:** Remember that the IP (address) of the PC and the IP (address) of the mobile phone connected to the same WiFi will probably be changing every time we disconnect and reconnect so we have to double check what addresses each device has, this will ensure the success of the connection between devices through the phone's SSH server and PC (Client).

## 8. Ambientes Blockly (App Inventor, AppyBuilder y Thunkable).

App Inventor is a software development environment created by Google Labs to build applications for the Android operating system. The user can, visually and from a set of basic tools, link a series of blocks to create the application. The system is free and can be easily downloaded from the web. Applications created with App Inventor are very easy to create because no knowledge of any programming language is required.

All the current environments that use Blockly's technology such as AppyBuilder and Thunkable among others have their free version, their way of use can be through the internet in their different sites or it can also be installed at home.

The blocks that make up the Mini BloclyChain architecture have been tested in App inventor and AppyBuilder but because of their code optimization they should work on the other platforms.

Online versions:

App Inventor.

https://appinventor.mit.edu/

AppyBuilder.

http://appybuilder.com/

Thunkable.

https://thunkable.com/

Version to be installed on your computer (PC):

https://sites.google.com/site/aprendeappinventor/instala-app-inventor

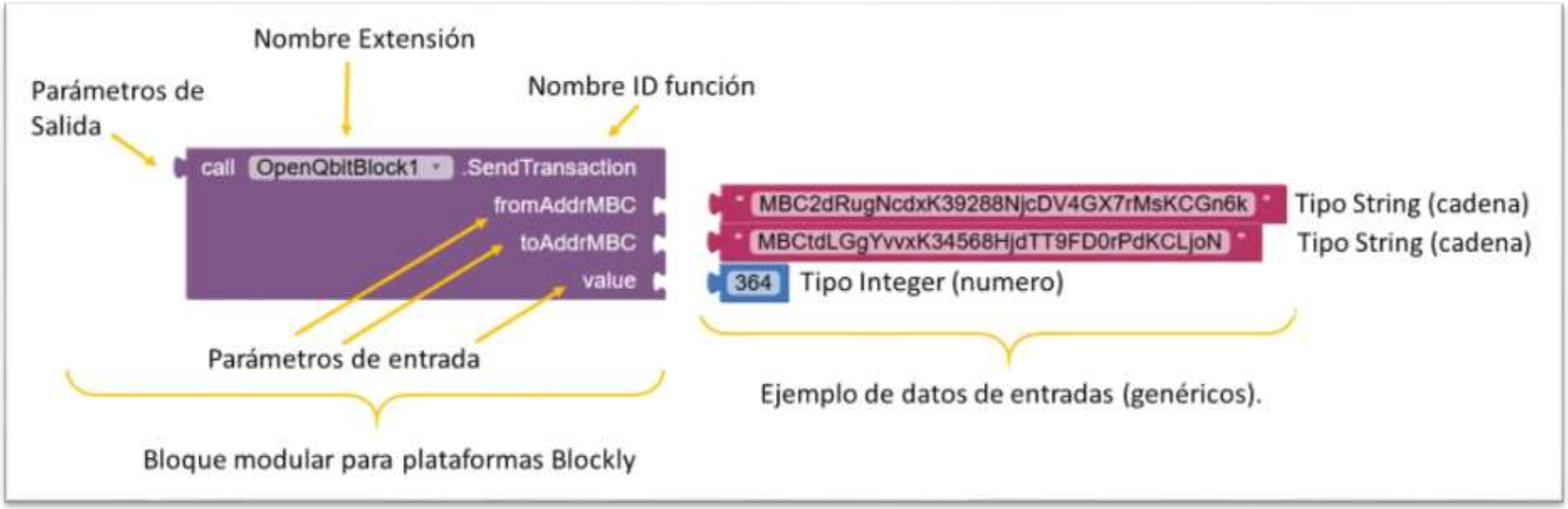
Environment for developers of Blockly blocks.

https://editor.appybuilder.com/login.php

## 9. Definition and use of blocks in Mini QRNG.

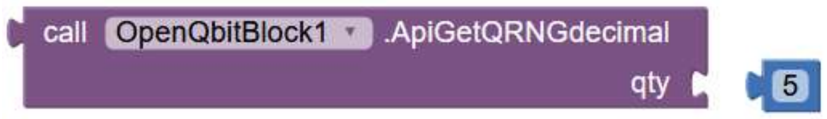
We will start by explaining the distribution of the data that all the blocks will have, their syntax of use and configuration.

In the following example we can see a modular block and its input and output parameters, as well as the types of input data, these data can be of type String (string of characters) or Integer (integer or decimal). We show how it is used and configure it for its proper functioning.


Nombre Extensión
Parámetros de Salida
Nombre ID función
call OpenQbitBlock1 .SendTransaction
fromAddrMBC
toAddrMBC
value
MBC2dRugNcdxK39288NjcDV4GX7rMsKCGn6k
Tipo String (cadena)
MBCtdLGgYvvxK34568HjdTT9FD0rPdKCLjoN
Tipo String (cadena)
364
Tipo Integer (numero)
Parámetros de entrada
Ejemplo de datos de entradas (genéricos).
Bloque modular para plataformas Blockly

Each module block will have its description and will be named in case it has any mandatory or optional dependency(s) of other blocks used as input parameters, the integration process will be announced. Let's start with the blocks of the **OpenQbitQRNGwithSSH** extension.

Block for generating decimal random quantum numbers - (**ApiGetQRNGdecimal**)



Input parameters: **qty <Integer>**

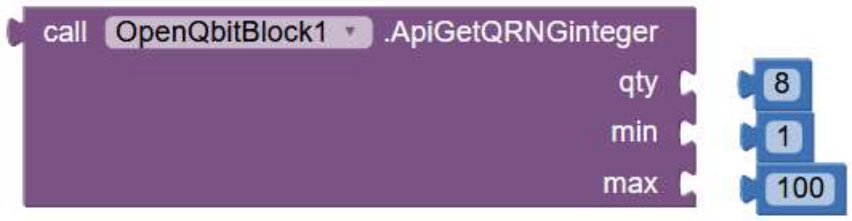
Output Parameters: Gives the amount "qty" of random quantum decimal numbers entered in the input numbers are within the range of 0 and 1 in JSON format.

Example:

qty = 5; output: {"result": [0.5843012986202495, 0.7746497687824652, 0.05951126805960929, 0.1986079055812694, 0.03689783439899279]}

Description: Quantum Random Number Generator (QRNG) API

Block for generating decimal random quantum numbers - (**ApiGetQRNGdecimal**)



Input parameters: **qty <Integer**>, min <Integer>, max <max>

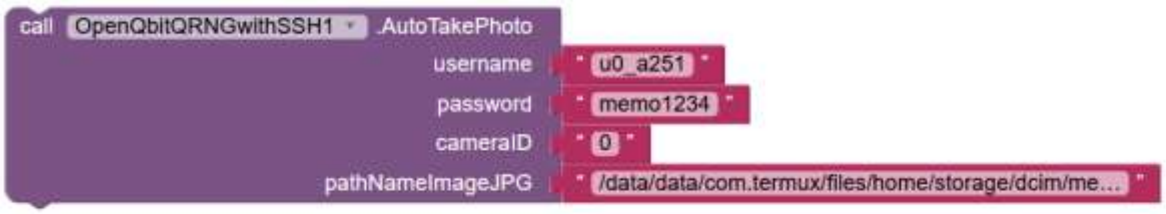
Output Parameters: Gives the amount "qty" of random quantum integers entered in the input the numbers are within the range of min and max in JSON format.

Example:

qty = 8, min = 1, max = 100; output: {"result": [3, 53, 11, 2, 66, 44, 9, 78]}

Description: Quantum Random Number Generator (QRNG) API

Block to take a picture automatically - (**AutoTakePhone**)


call OpenQbitQRNGwithSSH1 .AutoTakePhoto
username "u0_a251"
password "memo1234"
cameraID "0"
pathNameImageJPG "/data/data/com.termux/files/home/storage/dcim/me..."

Input parameters: **username < String**>, **password < String**>, cameraID < String> **,** pathNameImageJPG < **String>.**

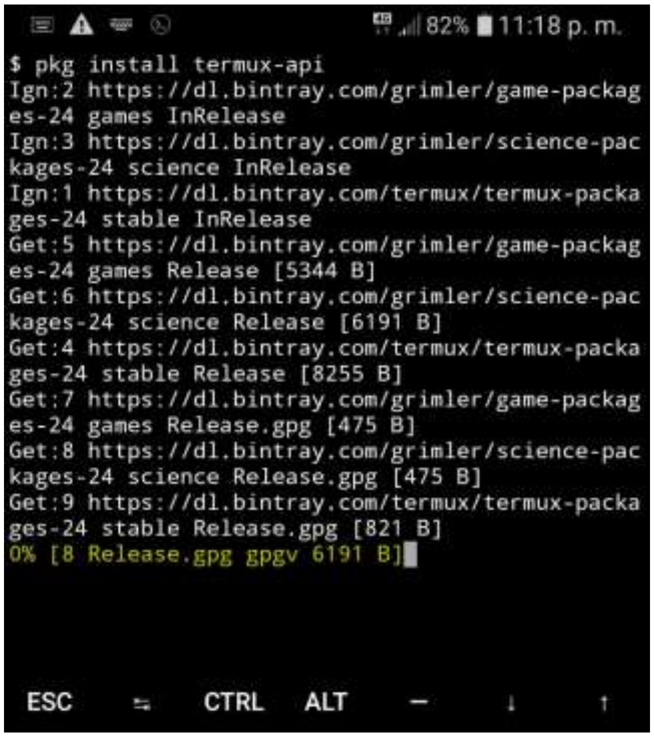
Mandatory dependency: To use this block you must fulfill two software dependencies; install in the Termux terminal the Termux-API module. This module contains the process to automatically take pictures and upload the SSH server that was previously installed.

Output parameters: It delivers a photo (image) in JPG format in the specified path. In the path you must

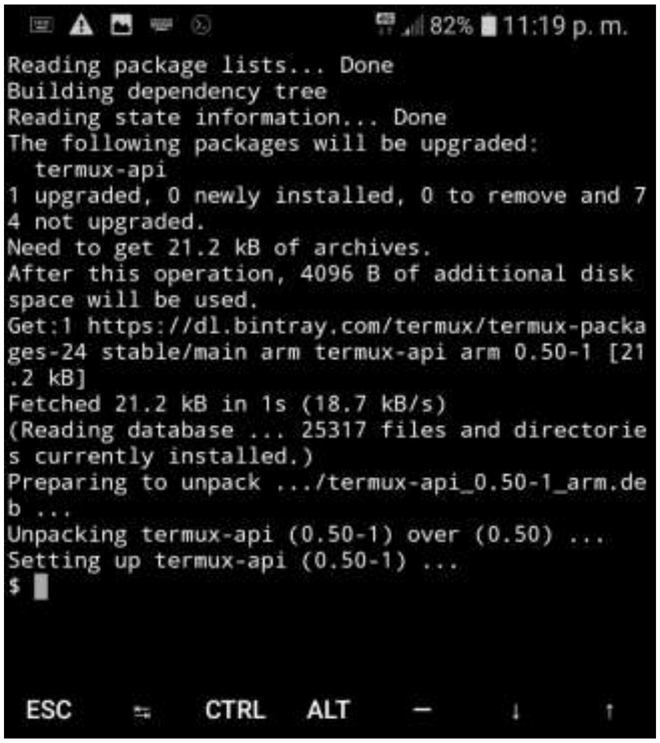
Description: Creates a JPG photo automatically without user intervention.

To install the Termux-API, the following command must be executed in the Termux Terminal:

**$ pkg install termux-api**



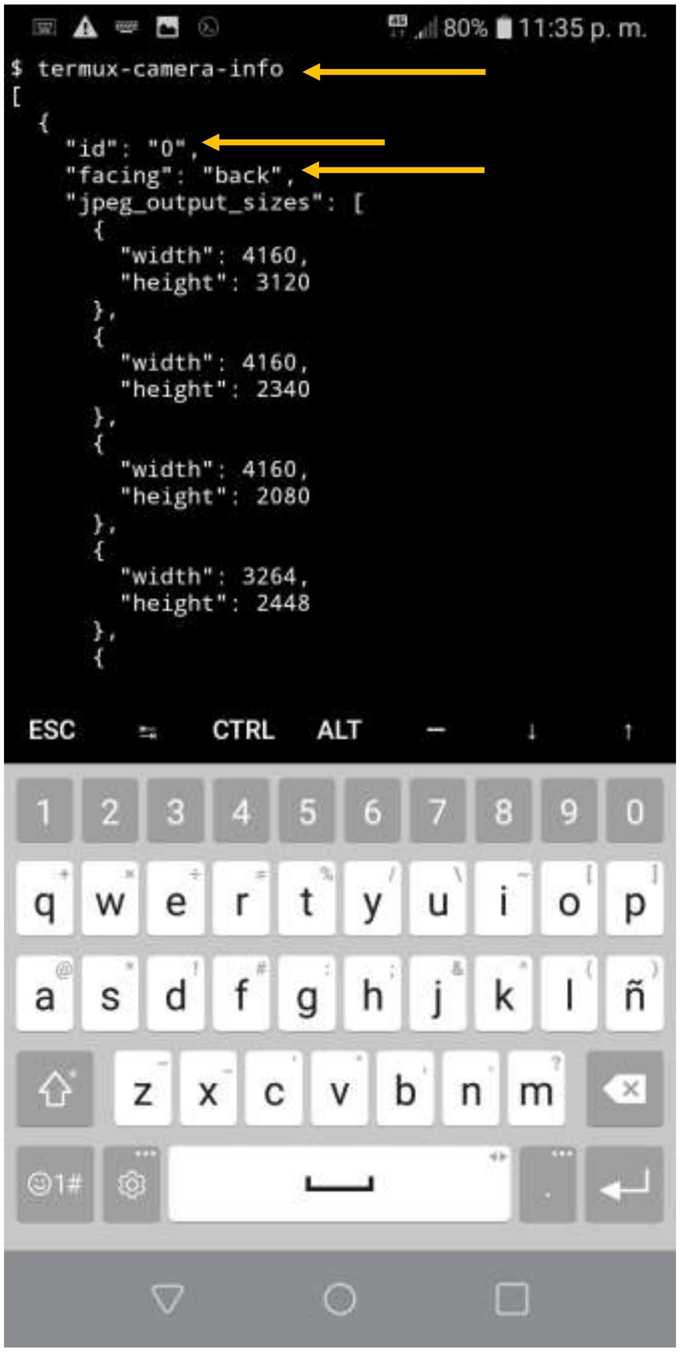
```
$ pkg install termux-api
Ign:2 https://dl.bintray.com/grimler/game-packag
es-24 games InRelease
Ign:3 https://dl.bintray.com/grimler/science-pac
kages-24 science InRelease
Ign:1 https://dl.bintray.com/termux/termux-packa
ges-24 stable InRelease
Get:5 https://dl.bintray.com/grimler/game-packag
es-24 games Release [5344 B]
Get:6 https://dl.bintray.com/grimler/science-pac
kages-24 science Release [6191 B]
Get:4 https://dl.bintray.com/termux/termux-packa
ges-24 stable Release [8255 B]
Get:7 https://dl.bintray.com/grimler/game-packag
es-24 games Release.gpg [475 B]
Get:8 https://dl.bintray.com/grimler/science-pac
kages-24 science Release.gpg [475 B]
Get:9 https://dl.bintray.com/termux/termux-packa
ges-24 stable Release.gpg [821 B]
0% [8 Release.gpg gpgv 6191 B]
```



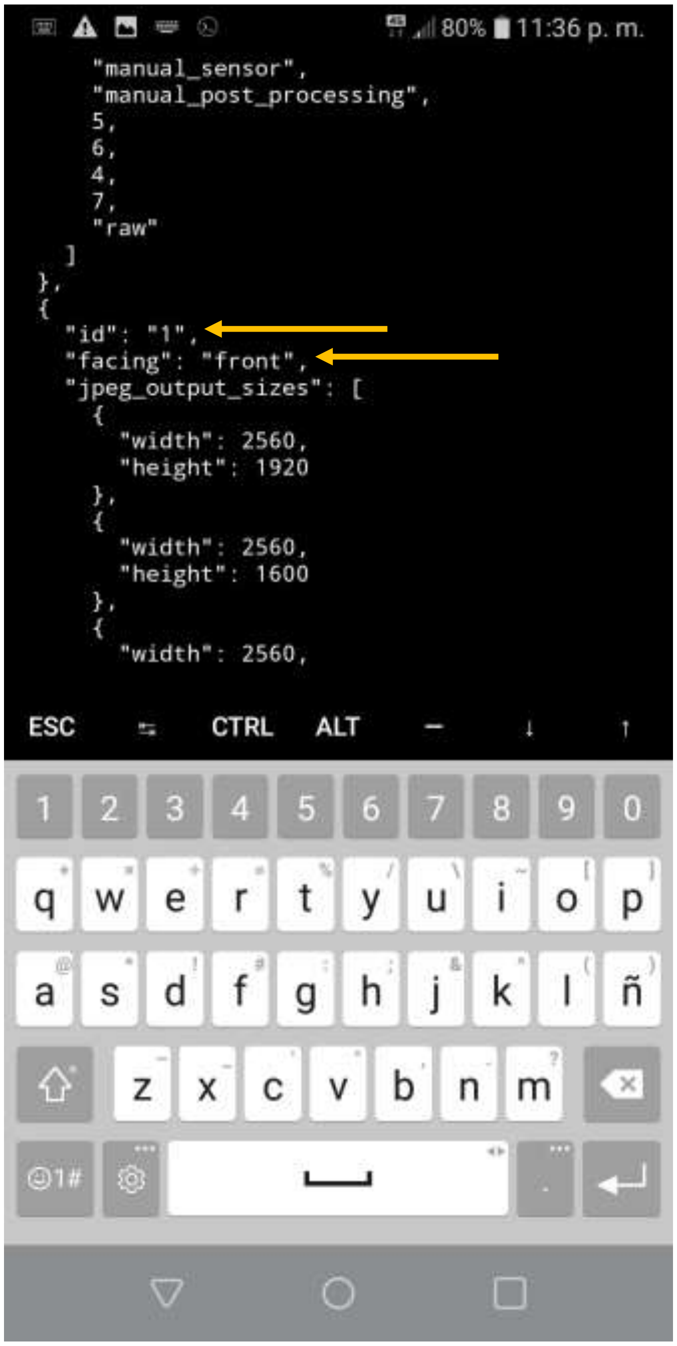
```
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following packages will be upgraded:
  termux-api
1 upgraded, 0 newly installed, 0 to remove and 7
4 not upgraded.
Need to get 21.2 kB of archives.
After this operation, 4096 B of additional disk
space will be used.
Get:1 https://dl.bintray.com/termux/termux-packa
ges-24 stable/main arm termux-api arm 0.50-1 [21
.2 kB]
Fetched 21.2 kB in 1s (18.7 kB/s)
(Reading database ... 25317 files and directorie
s currently installed.)
Preparing to unpack .../termux-api_0.50-1_arm.de
b ...
Unpacking termux-api (0.50-1) over (0.50) ...
Setting up termux-api (0.50-1) ...
$
```

To find out the number, how many and position of IDs (photo lens identifiers) your mobile device (Smartphone) has, execute the following command in the Termux terminal.

$ termux-camera-info



```
$ termux-camera-info
[
  {
    "id": "0",
    "facing": "back",
    "jpeg_output_sizes": [
      {
        "width": 4160,
        "height": 3120
      },
      {
        "width": 4160,
        "height": 2340
      },
      {
        "width": 4160,
        "height": 2080
      },
      {
        "width": 3264,
        "height": 2448
      },
      {
```



```
      "manual_sensor",
      "manual_post_processing",
      5,
      6,
      4,
      7,
      "raw"
    ]
  },
  {
    "id": "1",
    "facing": "front",
    "jpeg_output_sizes": [
      {
        "width": 2560,
        "height": 1920
      },
      {
        "width": 2560,
        "height": 1600
      },
      {
        "width": 2560,
```

In our example the LG Q6 smarpone we used has two IDs "0" on the back and "1" on the front.

Now let's test the API to take a picture using the ID "0" of the rear lens and give it an albitary name in our case test.jpg

Remember that the API only delivers photos in JPG format:

**$ termux-camera-photo -c 0 test.jpg**

The previous command must have created a file with the name test.jpg automatically, if so we can use the block (AutoTakePhoto), do not forget to start our local SSH server with the command: **$ sshd**

**NOTE:** In the variable pathNameImageJPG it should be considered that the path inside the Termux terminal to access the Smartphone storage should be used:

/data/data/com.termux/files/home/storage/dcim/example.jpg

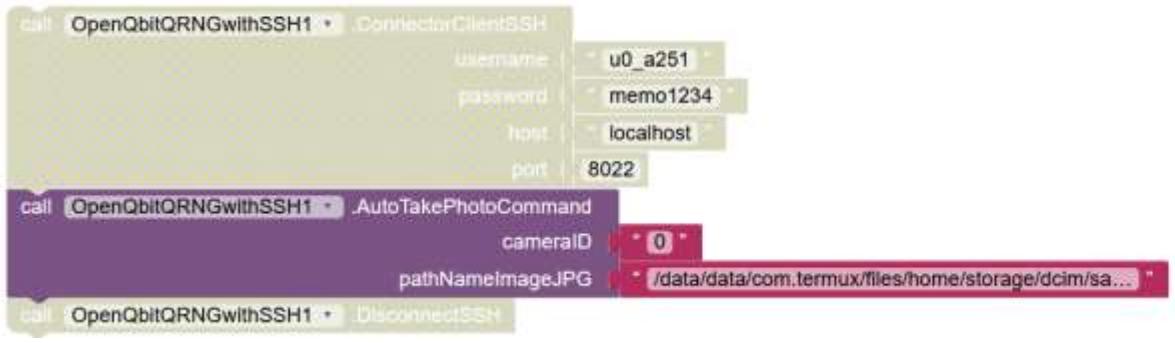
The previous route on android would be the same as:

/mnt/sdcard/dcim/example.jpg

However, we must remember that in the Termux terminal the valid route for viewing the mobile phone power-up must always be considered the default route:

/data/data/com.termux/files/home/storage

Block to take a photo automatically ONLY COMMAND - (**AutoTakePhoneCommand**)



```
call OpenQbitQRNGwithSSH1 .ConnectorClientSSH
                          username  "u0_a251"
                          password  "memo1234"
                          host      "localhost"
                          port      8022
call OpenQbitQRNGwithSSH1 .AutoTakePhotoCommand
                          cameraID          "0"
                          pathNameImageJPG  "/data/data/com.termux/files/home/storage/dcim/sa..."
call OpenQbitQRNGwithSSH1 .DisconnectSSH
```

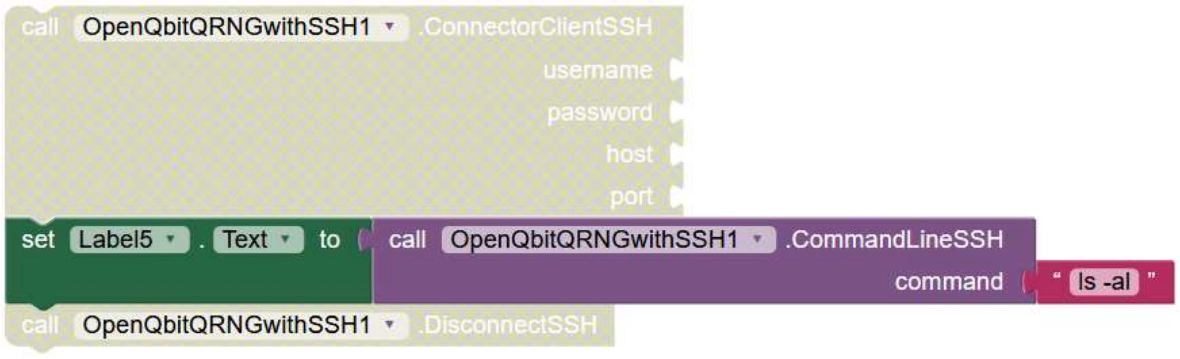
Input parameters: **cameraID <String>**, pathNameImageJPG **<String>**.

Mandatory dependency: Block (**ConnectorClientSSH**), Block (**DisconnectSSH**).

Output parameters: It delivers a photo (image) in JPG format in the specified path.

Description: Creates a JPG photo automatically without user intervention. However, the difference compared to the block (AutoTakePhoto), this block only contains the command to create the photo and you need the block first to connect to the SSH server (**ConnectClientSSH) and** then use the block (**DisconnectSSH**).

Block to execute command in Termux terminal - (**CommandLineSSH**)


call OpenQbitQRNGwithSSH1 .ConnectorClientSSH
username
password
host
port
set Label5 . Text to
call OpenQbitQRNGwithSSH1 .CommandLineSSH
command
" ls -al "
call OpenQbitQRNGwithSSH1 .DisconnectSSH

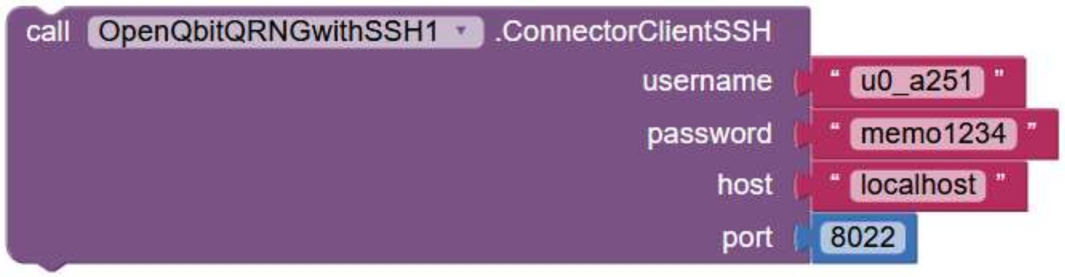
Input parameters: **command <String>**

Mandatory dependency: Block (**ConnectorClientSSH**), Block (**DisconnectSSH**).

Output parameters: Execute the command entered in the Termux terminal.

Description: An entered command is executed and the block is needed first to connect to the SSH server (**ConnectClientSSH) and** then to use the block (**DisconnectSSH**).

Block to connect to a remote or local SSH server - (**ConnectorClientSSH**).

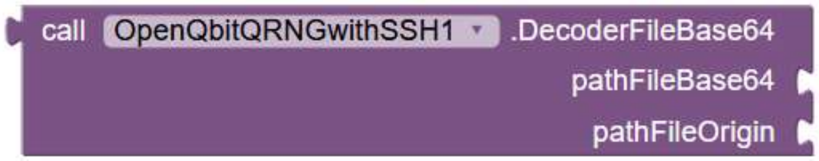


Input parameters: **username** <string>, **password <string>,** host <string>, port<integer>

Output parameters: If the connection with the ssh server of the Termux terminal is successful, it gives us a message; **"Connect SSH"**, if it is not successful, it gives us a **NULL** message.

Description: Communication block to connect the chosen SSH server to the Termux terminal, via SSH (Secure Shell) communication protocol.

Block to decode a file with Base64 algorithm (**DecoderFileBase64**).

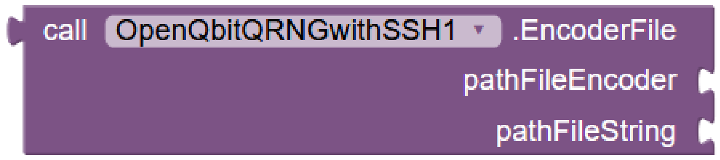


Input parameters: **pathFileBase64 <String**>, pathFileOrigin **<String>**

Output parameters: Source file that was entered in the block (**EncoderFileBase64**)

Description: A Base64 file is converted to the original file that was inserted in the block (**EncoderFileBase64**).

Block converts a file to Base64 format - (**EncoderFile**)

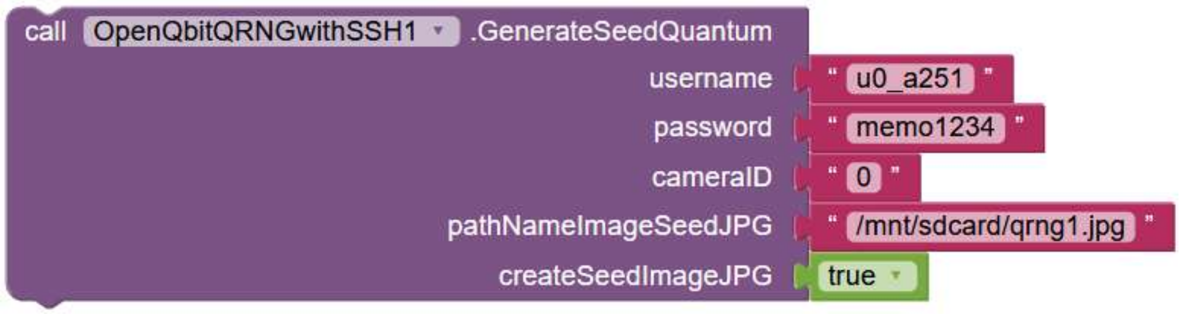


Input parameters: **pathFileOrigin <String>** , pathFileBase64 **<String>**

Output parameters: Base64 encoded file.

Description: Converts a source file of any format to a Base64 file. File names can be arbitrary and chosen by the user.

Block to generate QRNG (Quantum Random Number Generator) - (**GenerateSeedQuantum**)



Input parameters: **username <String>**, **password <String>**, cameraID **<String>,** pathNameImageJPG <String**>. createSeedImageJPG** <Blean>

If the Boolean value is "True" when the block is executed each time a new seed JPG image will be created with the name of the path entered. If the Bolean value is "False" we deactivate the option to take a JPG image (photo) and we can manually indicate where an image is in our selection, which can be of any format.

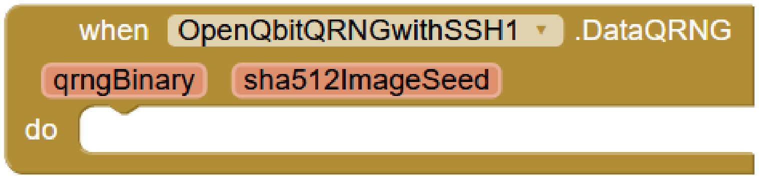
NOTE: The best results for generating a QRNG are based on a "RAW" formatted image. Example DNG format.

Mandatory dependency: The aforementioned Termux-API must be installed in the block (**AutoTakePhoto**).

Output parameters: The event is executed (**DataQRNG**) and gives us two values:

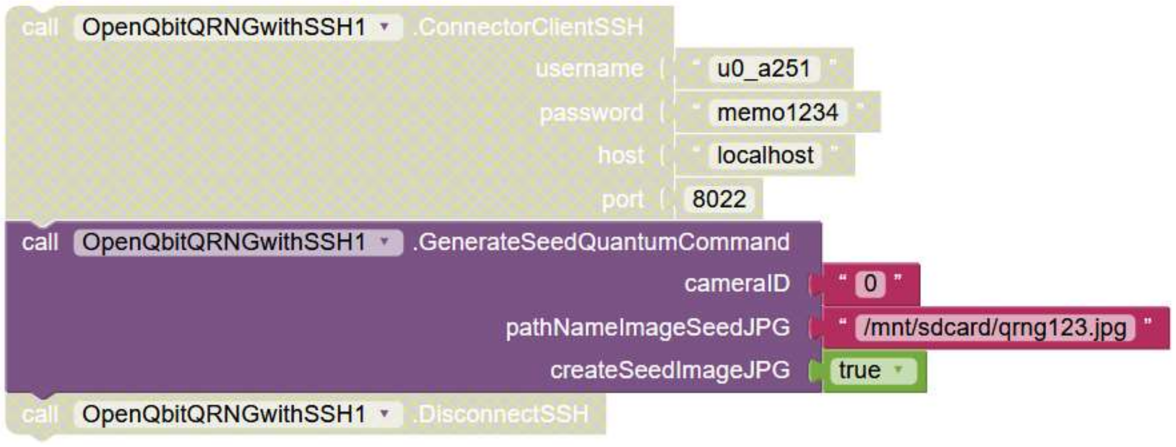
qrngBinary.- A string of random binary numbers

Sha512ImageSeed - Sha512 from the JPG seed image where the random numbers came from.



Description: It generates quantum random numbers (QRNG) through the optical sensor of the mobile phone's camera. The algorithm is based on the capture of random photos and the algorithm is applied to it to deliver a string of binary numbers.

Block to generate QRNG (Quantum Random Number Generator) - (GenerateSeedQuantumCommand).


call OpenQbitQRNGwithSSH1 .ConnectorClientSSH
username " u0_a251 "
password " memo1234 "
host " localhost "
port 8022
call OpenQbitQRNGwithSSH1 .GenerateSeedQuantumCommand
cameraID " 0 "
pathNameImageSeedJPG " /mnt/sdcard/qrng123.jpg "
createSeedImageJPG true
call OpenQbitQRNGwithSSH1 .DisconnectSSH

Input parameters: **cameraID <String>**, pathNameImageJPG **<String>**. createSeedImageJPG <Blean>

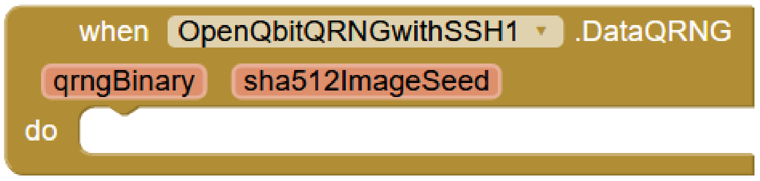
If the Boolean value is "True" when the block is executed each time a new seed JPG image will be created with the name of the path entered. If the Bolean value is "False" we deactivate the option to take a JPG image (photo) and we can manually indicate where an image is in our selection, which can be of any format.

Mandatory dependency: Block (**ConnectorClientSSH**), Block (**DisconnectSSH**).

Output parameters: The event is executed (**DataQRNG**) and gives us two values:

qrngBinary.- A string of random binary numbers

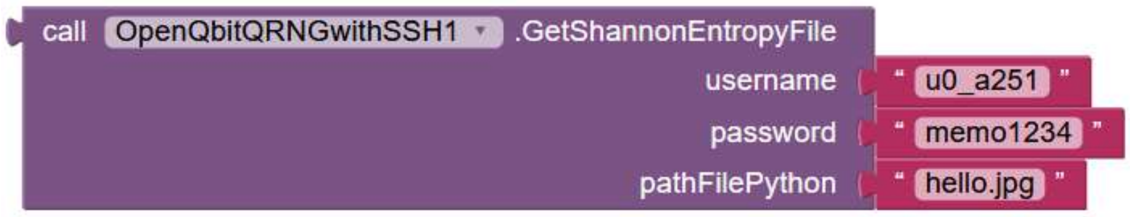
Sha512ImageSeed - Sha512 from the JPG seed image where the random numbers came from.



Description: It generates quantum random numbers (QRNG) through the optical sensor of the mobile phone's camera. The algorithm is based on the capture of random photos and the algorithm is applied to it to deliver a string of binary numbers.

However, the difference compared to the block (**GenerateSeedQuantum),** this block only contains the command to execute the QRNG reaction algorithm and you need the block first to connect to the SSH server (**ConnectClientSSH) and** then use the block (**DisconnectSSH**).

Block to obtain Shannon entropy in an Image (photo) - (**GetShannonEntropyFile**).



Input parameters: **username <String>** , **password <String**>, pathFileImage **<String>**

Mandatory dependency: The Shannon_entropy module needs to be installed in the Termux terminal.

Output Parameters: Delivers the entropy of an image

Example:

Output: 8.94596789873

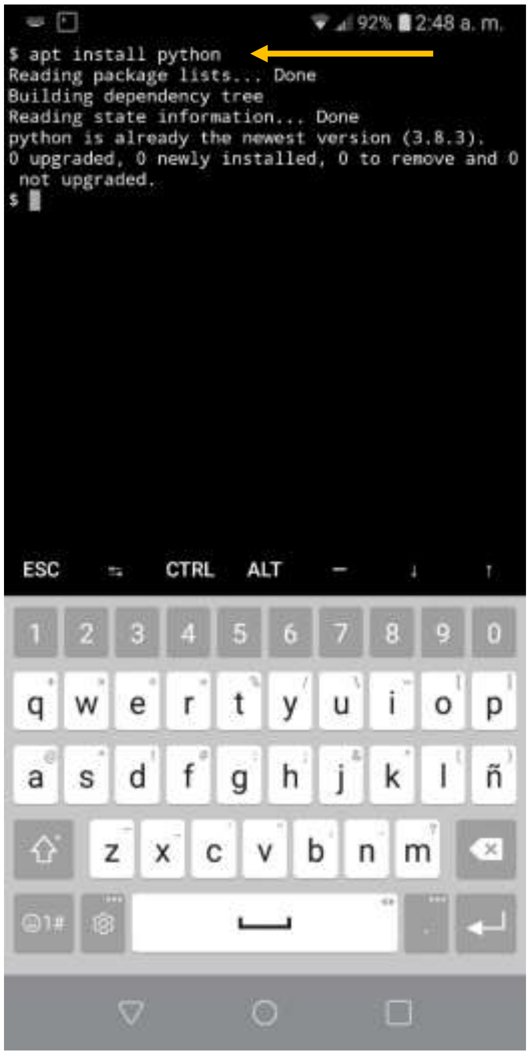
Description: It gives us the entropy of an image. Entropy is the fundamental parameter for the generation of good quality random numbers, the higher the entropy, the better the results.

To install the Shannon entropy module, we first need to install the Python package and then install the Pillow and Shannon_entropy module with the following commands in the Termux terminal.

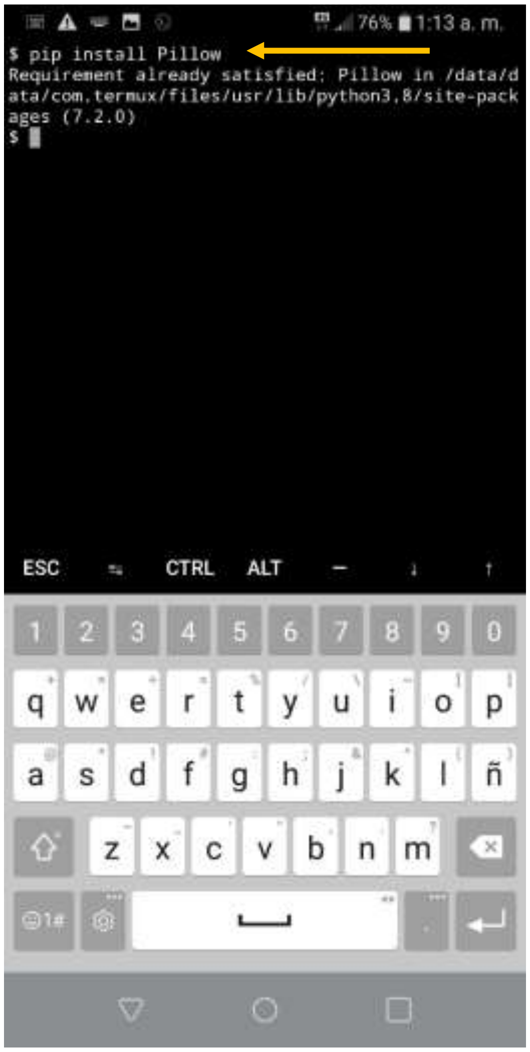
$ apt install Python

$ pip install Pillow

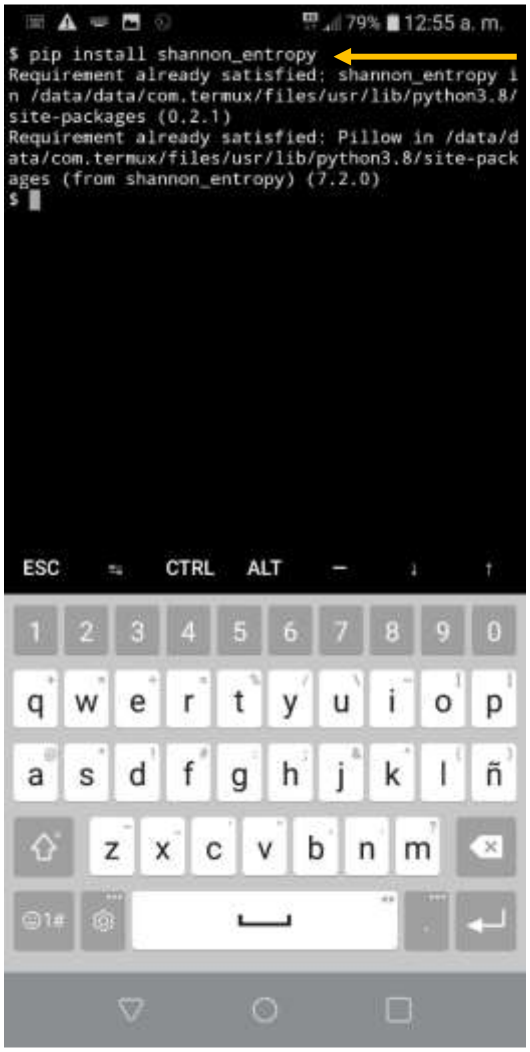
$ pip install Shannon_entropy



```
$ apt install python
Reading package lists... Done
Building dependency tree
Reading state information... Done
python is already the newest version (3.8.3).
0 upgraded, 0 newly installed, 0 to remove and 0
 not upgraded.
$
```



```
$ pip install Pillow
Requirement already satisfied: Pillow in /data/d
ata/com.termux/files/usr/lib/python3.8/site-pack
ages (7.2.0)
$
```



```
$ pip install shannon_entropy
Requirement already satisfied: shannon_entropy i
n /data/data/com.termux/files/usr/lib/python3.8/
site-packages (0.2.1)
Requirement already satisfied: Pillow in /data/d
ata/com.termux/files/usr/lib/python3.8/site-pack
ages (from shannon_entropy) (7.2.0)
$
```

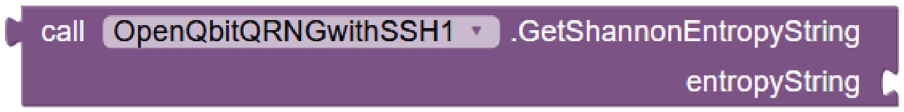
Then we have to create in Termux's "Home" directory the following Python file called "entropy.py" with the following code inside.

```
From PIL import Image

Import math

From Shannon_entropy import *

Import sys

Img=Image.open(sys.argv[1])

print(Shannon_entropy(img))
```

We save the file and have our environment to use with the block (GetShannonEntropyFile).

Tip: In fact, with this Python installation you can create your own programs in this language and run them through the block (ConnectorClientSSH).

Block to obtain Shannon entropy from a string - (GetShannonEntropyString).



Input parameters: entropyString>String>

Output Parameters: Delivers the entropy of a character string.

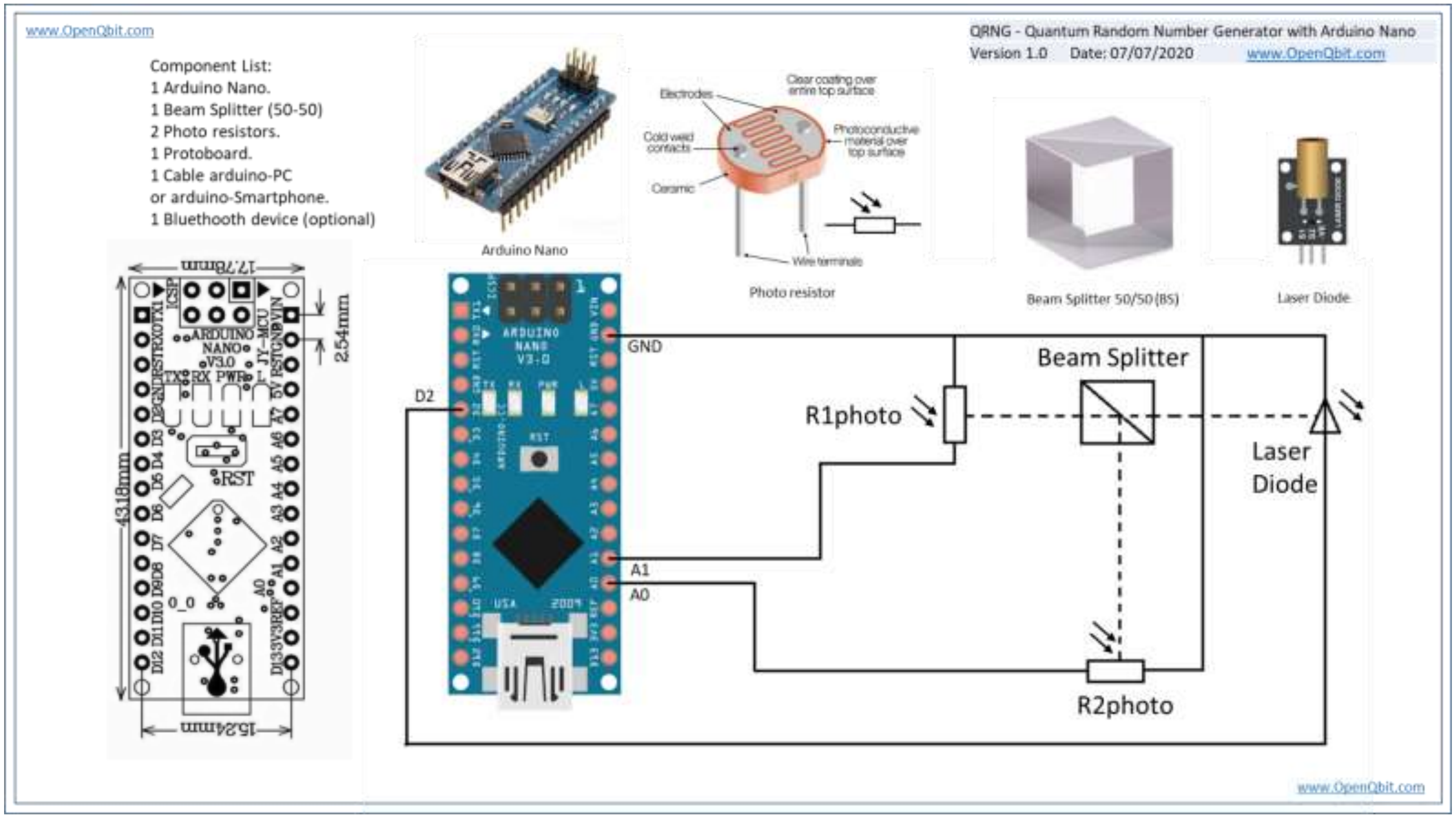
Example:

Output: 5.76002345671

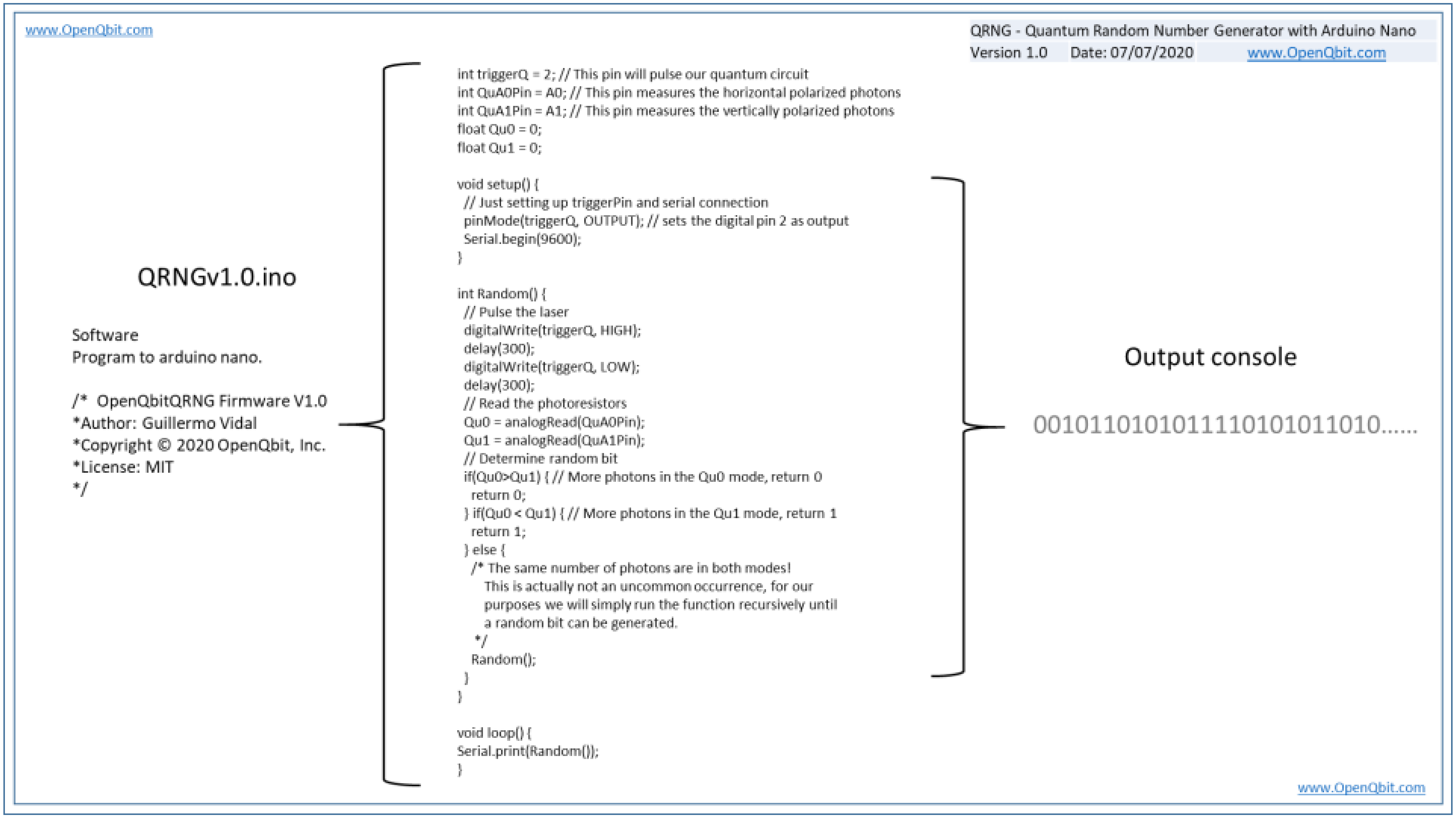
Description: It gives us the entropy of a string of characters. Entropy is the fundamental parameter for the generation of good quality random numbers, the higher the entropy, the better the results.

## 10. Creation of "Hardware" device of a QRNG.

We will now create a physical "Hardware" device to generate Quantum Random Numbers (QRNG) with inexpensive components that can be easily assembled at home.



Component List:
1 Arduino Nano.
1 Beam Splitter (50-50)
2 Photo resistors.
1 Protoboard.
1 Cable arduino-PC
or arduino-Smartphone.
1 Bluethooth device (optional)



www.OpenQbit.com

QRNG - Quantum Random Number Generator with Arduino Nano
Version 1.0 Date: 07/07/2020 www.OpenQbit.com

QRNGv1.0.ino

Software
Program to arduino nano.

/* OpenQbitQRNG Firmware V1.0
*Author: Guillermo Vidal
*Copyright © 2020 OpenQbit, Inc.
*License: MIT
*/

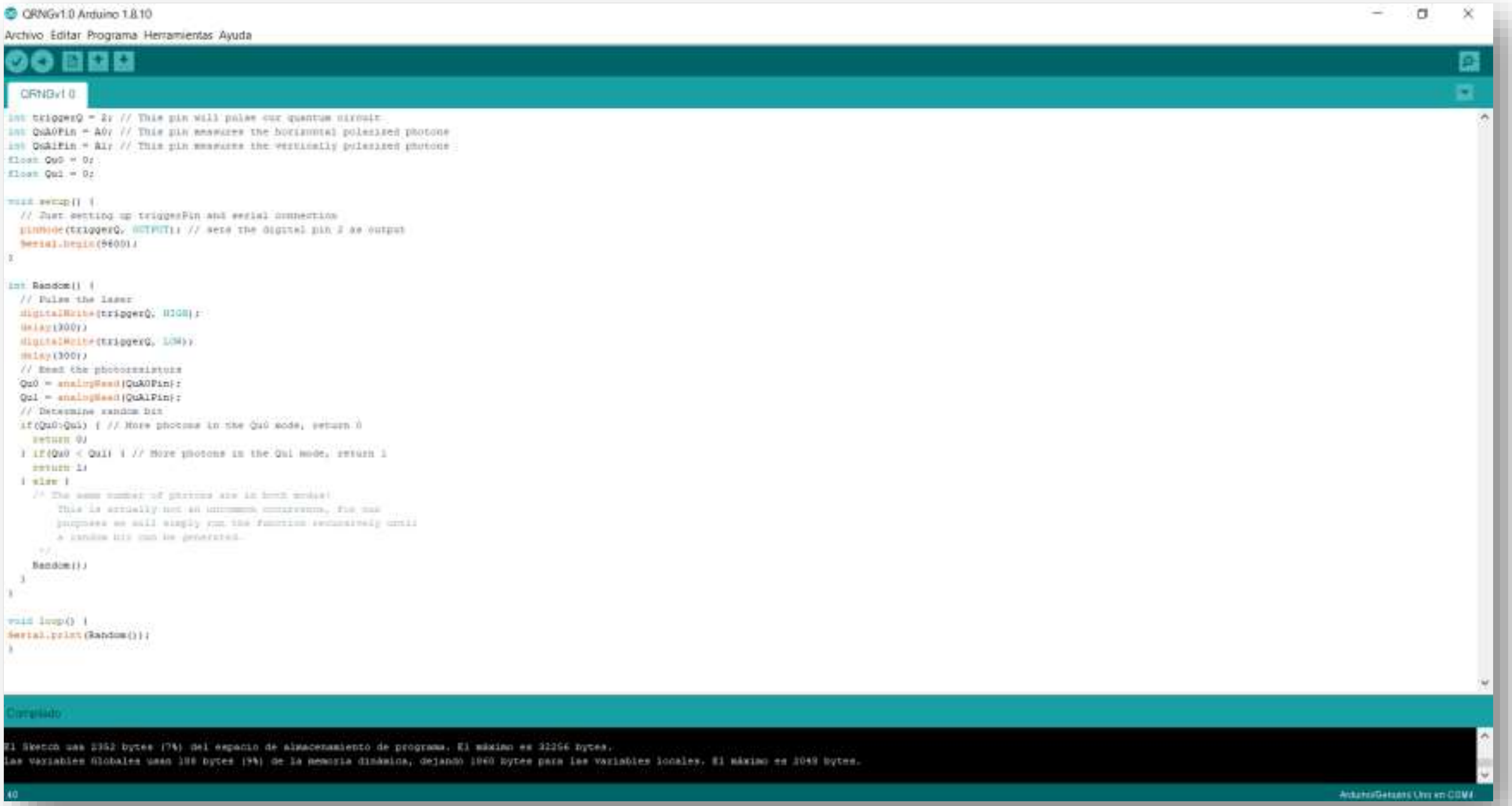
```
int triggerQ = 2; // This pin will pulse our quantum circuit
int QuA0Pin = A0; // This pin measures the horizontal polarized photons
int QuA1Pin = A1; // This pin measures the vertically polarized photons
float Qu0 = 0;
float Qu1 = 0;

void setup() {
 // Just setting up triggerPin and serial connection
 pinMode(triggerQ, OUTPUT); // sets the digital pin 2 as output
 Serial.begin(9600);
}

int Random() {
 // Pulse the laser
 digitalWrite(triggerQ, HIGH);
 delay(300);
 digitalWrite(triggerQ, LOW);
 delay(300);
 // Read the photoresistors
 Qu0 = analogRead(QuA0Pin);
 Qu1 = analogRead(QuA1Pin);
 // Determine random bit
 if(Qu0>Qu1) { // More photons in the Qu0 mode, return 0
  return 0;
 } if(Qu0 < Qu1) { // More photons in the Qu1 mode, return 1
  return 1;
 } else {
  /* The same number of photons are in both modes!
    This is actually not an uncommon occurrence, for our
    purposes we will simply run the function recursively until
    a random bit can be generated.
   */
  Random();
 }
}

void loop() {
Serial.print(Random());
}
```

Output console

```
0010110101011110101011010......
```

www.OpenQbit.com

Compiling the QRNGv10.ino program and uploading to arduous nano...

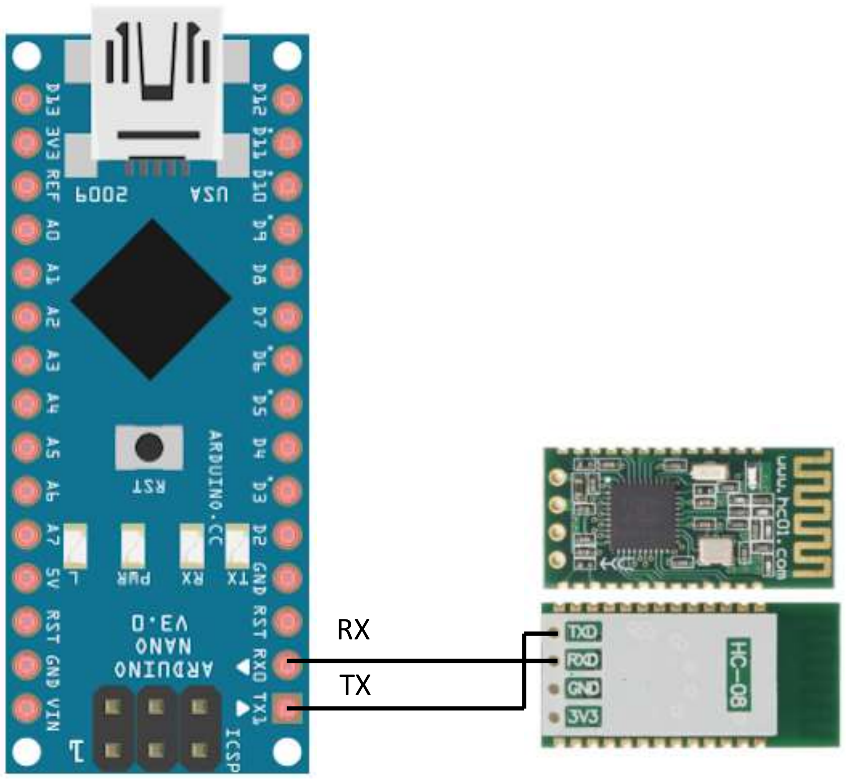


```
int triggerQ = 2; // This pin will pulse our quantum circuit
int QuA0Pin = A0; // This pin measures the horizontal polarized photons
int QuA1Pin = A1; // This pin measures the vertically polarized photons
float Qu0 = 0;
float Qu1 = 0;

void setup() {
  // Just setting up triggerPin and serial connection
  pinMode(triggerQ, OUTPUT); // sets the digital pin 2 as output
  Serial.begin(9600);
}

int Random() {
  // Pulse the laser
  digitalWrite(triggerQ, HIGH);
  delay(300);
  digitalWrite(triggerQ, LOW);
  delay(300);
  // Read the photoresistors
  Qu0 = analogRead(QuA0Pin);
  Qu1 = analogRead(QuA1Pin);
  // Determine random bit
  if(Qu0>Qu1) { // More photons in the Qu0 mode, return 0
    return 0;
  } if(Qu0 < Qu1) { // More photons in the Qu1 mode, return 1
    return 1;
  } else {
    /* The same number of photons are in both modes!
        This is actually not an uncommon occurrence. For our
        purposes we will simply run the function recursively until
        a random bit can be generated.
     */
    Random();
  }
}

void loop() {
Serial.print(Random());
}
```

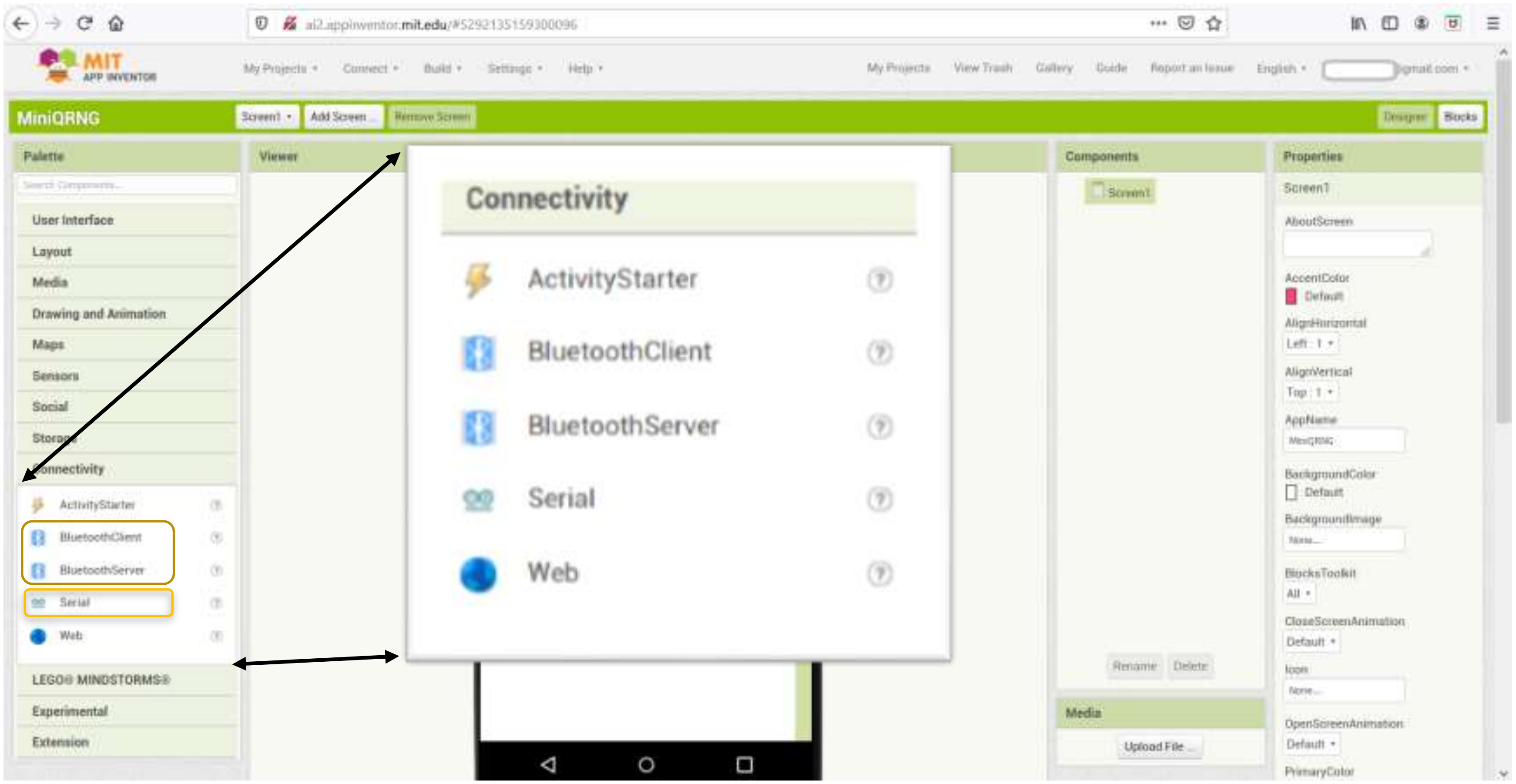
```
El Sketch usa 2352 bytes (7%) del espacio de almacenamiento de programa. El máximo es 32256 bytes.
Las variables Globales usan 188 bytes (9%) de la memoria dinámica, dejando 1860 bytes para las variables locales. El máximo es 2048 bytes.
```

There are two ways to communicate with the arduous nano, one is via the Serial port and the other is via a Bluetooth connection.

For the bluetooth connection is very simple we just have to buy the HC-08 module or a similar one and connect it as follows:



The following Serial or Bluetooth components can be used to connect App Inventor to Arduino:



Now compiled and loaded the program QRNGv10.ino only lack communicate with the arduous nano to save the data (quantum random numbers) these will be in binary format, however, the data obtained can be easily passed to another format such as hexadecimal or decimal depending on the final requirement.

Finally, to see an example of how the serial or Bluetooth connection works, here are some reference links.

Remember that everything is through Blockly programming to be tested with App Inventor this already has blocks for communication with arduino serial or other blockly system may be to similar bluetooth tarves online.

http://kio4.com/appinventor/9A0_bluetooth_RXTX.htm

http://kio4.com/appinventor/index.htm#bluetooth

https://community.appinventor.mit.edu/

## 11.Annex "OpenQbit Quantum Computing".

### How does quantum computing work? (2)

The digital transformation is bringing about change in the world faster than ever before. Would you believe that the digital era is about to end? **Digital literacy** has already been identified as an area where open knowledge and accessible opportunities to learn about technology are urgent to address gaps in social and economic development. Learning from the key concepts of the digital age will become even more critical with the imminent arrival of another new technological wave capable of transforming existing models with amazing speed and power: **quantum technologies**.

In this article, we compare the basic concepts of traditional computing and quantum computing; and we also begin to explore their application in other related areas.

What are quantum technologies?

Throughout history, human beings have developed technology as they have understood how nature works through science. Between 1900 and 1930, the study of some physical phenomena that were not yet well understood gave rise to a new physical theory, **Quantum Mechanics**. This theory describes and explains the functioning of the microscopic world, the natural habitat of molecules, atoms or electrons. Thanks to this theory, not only has it been possible to explain these phenomena, but it has also been possible to understand that subatomic reality works in a completely counter-intuitive, almost magical way, and that in the microscopic world events take place that do not occur in the macroscopic world.

These quantum **properties** include quantum superposition, quantum entanglement and quantum teleportation.

- **Quantum superposition** describes how a particle can be in different states at the same time.
- **Quantum entanglement** describes how two particles as far apart as desired can be correlated in such a way that, when interacting with one, the other is aware of it.
- **Quantum teleportation** uses quantum entanglement to send information from one place to another in space without having to travel through it.

Quantum technologies are based on these quantum properties of the subatomic nature.

In this case, today the understanding of the microscopic world through Quantum Mechanics allows us to invent and design technologies capable of improving people's lives. There are many and very different technologies that use quantum phenomena and some of them, such as lasers or magnetic resonance imaging (MRI), have been with us for more than half a century. However, we are currently witnessing a technological revolution in areas such as quantum computing, quantum information, quantum simulation, quantum optics, quantum metrology, quantum clocks or quantum sensors.

What is quantum computing? First, you have to understand classical computing.

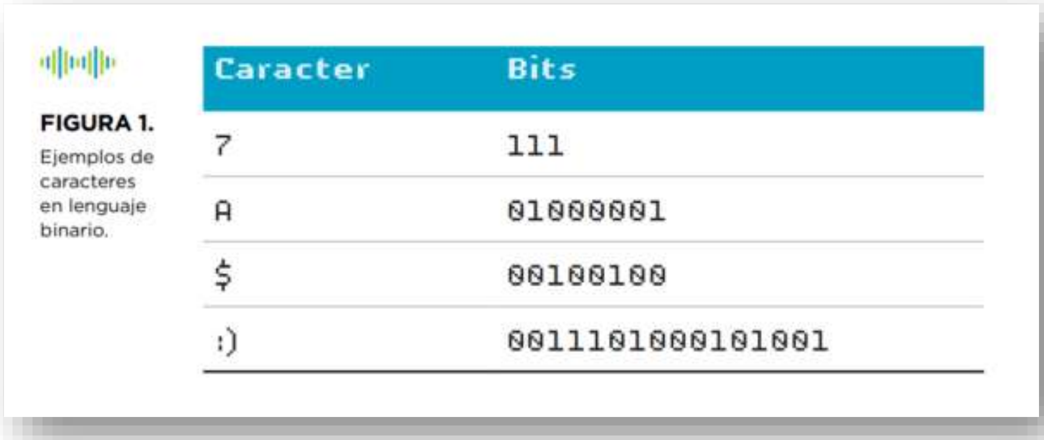


**FIGURA 1.**
Ejemplos de caracteres en lenguaje binario.

| Caracter | Bits |
|---|---|
| 7 | 111 |
| A | 01000001 |
| $ | 00100100 |
| :) | 0011101000101001 |

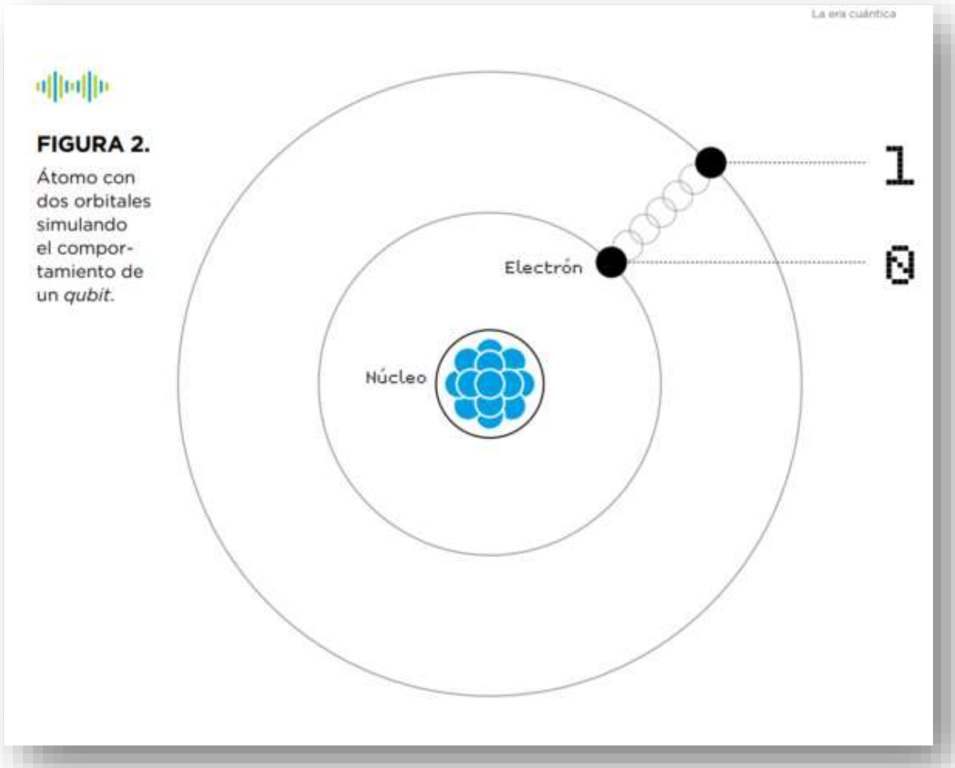
To
understand how quantum computers work, it is convenient to first explain how the computers we use every day, which we will refer to in this document as digital or classic computers, work. These, like the rest of the electronic devices such as tablets or mobile phones, use bits as the fundamental units of memory. This means that programs and applications are encoded in bits, that is, in binary language of zeros and ones. Every time we interact with any of these devices, for example, by pressing a key on the keyboard, strings of zeros and ones are created, destroyed and/or modified inside the computer.

The interesting question is, what are these zeros and ones physically inside the computer? The zero and one states correspond to electrical current that circulates, or not, through microscopic parts called transistors, which act as switches. When no current is flowing, the transistor is "off" and corresponds to bit 0, and when it is flowing it is "on" and corresponds to bit 1.

More simply, it is as if bits 0 and 1 correspond to holes, so that an empty hole is a bit 0 and a hole occupied by an electron is a bit 1. This is why these devices are called electronics. As an example, figure 1 shows the binary writing of some characters. Now that we have an idea of how today's computers work, let's try to understand how quantums work.

## From bits to qubits

The fundamental unit of information in quantum computing is the quantum bit or qubit. Qubits are, by definition, two-level quantum systems -we will see examples here- which, like bits, can be at the low level, which corresponds to a state of low excitation or energy defined as 0, or at the high level, which corresponds to a state of higher excitation or defined as 1. However, and here lies the fundamental difference with classical computing, qubits can also be in any of the infinite intermediate states between 0 and 1, such as a state that is half 0 and half 1, or three quarters of 0 and a quarter of 1.


La era cuántica
FIGURA 2.
Átomo con dos orbitales simulando el compor-tamiento de un *qubit*.
1
0
Electrón
Núcleo

Quantum algorithms, exponentially more powerful and efficient computing

The purpose of quantum computers is to take advantage of these quantum properties of *qubits,* as quantum systems that they are, in order to run quantum algorithms that use overlapping and interleaving to provide much greater processing power than the classics. It is important to point out that the real change of paradigm does not consist in doing the same thing that digital or classical computers do -the current ones- but faster, as it can be read in many articles, but that quantum algorithms allow to perform certain operations in a totally different way that in many cases turns out to be more efficient -that is, in much less time or using much less computational resources-.

Let's look at a concrete example of what this involves. Let's imagine that we are in Bogotá and we want to know the best route to get to Lima from among a million options to get there (N=1,000,000). In order to use computers to find the optimal route we need to digitize 1,000,000 options, which implies translating them into bit language for the classic computer and into *qubits for* the quantum computer. While a classic computer would need to go one by one analyzing all the paths until finding the desired one, a quantum computer takes advantage of the process known as quantum parallelism that allows it to consider all the paths at once. This implies that, while the classical computer needs the order of N/2 steps or iterations, that is, 500,000 attempts, the quantum computer will find the optimal path after only √N operations on the registry, that is, 1,000 attempts.

In the previous case the advantage is quadratic, but in other cases it is even exponential, which means that with n *qubits* we can obtain a computational capacity equivalent to 2n bits. To exemplify this, it is common to count that with about 270 qubits we could have more base states in a quantum computer - more different and simultaneous character strings - than the number of atoms in the universe, which is estimated to be around 280. Another example is that it is estimated that with a quantum computer of between 2000 and 2500 *qubits we* could break practically all the cryptography used today (the so-called public key cryptography).

Why is it important to know about quantum technology?

We are in a moment of digital transformation in which different emerging technologies such as blockchain, artificial intelligence, drones, Internet of things, virtual reality, 5G, 3D printers, robots or autonomous vehicles have more and more presence in multiple fields and sectors. These technologies, called to improve the quality of life of the human being accelerating the development and generating social impact, advance nowadays in a parallel way. Only rarely do we see companies developing products that exploit combinations of two or more of these technologies, such as blockchain and IoT or drones and artificial intelligence. Although they are destined to converge, thus generating an exponentially greater impact, the initial stage of development in which they find themselves and the scarcity of developers and people with technical profiles mean that convergence is still a pending task.

Because of their disruptive potential, quantum technologies are expected not only to converge with all these new technologies, but to have a cross-cutting influence on virtually all of them. Quantum computing will threaten authentication, exchange and secure storage of data, having a major impact on those technologies where cryptography has a more relevant role, such as cyber security or blockchain, and a minor negative impact but also to be considered in technologies such as 5G, IoT or drones.

Do you want to practice quantum computing?

Dozens of quantum computer simulators are already available on the net with different programming languages already in use such as C, C++, Java, Matlab, Maxima, Python or Octave. Also, new languages like Q#, launched by Microsoft. You can explore and play with a virtual quantum machine through platforms such as IBM and Rigetti.

Mini QRNG is created by OpenQbit.com company that focuses on developing quantum computing based technology for different types of sectors both private and public.

***Why Mini QRNG is different from other QRNGs, simply because the system was created to be modular and can be easily assembled at home at a fairly low cost.***

(1) https://blogs.iadb.org/conocimiento-abierto/es/como-funciona-la-computacion-cuantica/

## 12. Licensing and use of software.

Android
https://source.android.com/setup/start/licenses
Termux
https://github.com/termux/termux-app/blob/master/LICENSE.md
Node
https://raw.githubusercontent.com/nodejs/node/master/LICENSE
Python
https://www.python.org/download/releases/2.7/license/
OpenSSH
https://www.openssh.com/features.html
Putty SSH
https://www.chiark.greenend.org.uk/~sgtatham/putty/licence.html
MIT App Inventor 2 Companion and App Inventor Blockly
https://appinventor.mit.edu/about/termsofservice

External extensions:
JSONTOOLs
https://thunkableblocks.blogspot.com/2017/07/jsontools-extension.html

Licensing opensource and commercial versions of the QRNG Mini system see the official website http://www.openqbit.com

Mini QRNG, Mini BlocklyChain, MiniBlockly, BlocklyCode, MiniBlockMiniChain, QBlockly son marcas registradas por OpenQbit.

Mini QRNG is public domain.

All code and documentation in Mini QRNG has been dedicated to the public domain by the authors. All code authors and representatives of the companies they work for have signed affidavits dedicating their contributions to the public domain and the originals of those affidavits are stored in a safe at OpenQbit Mexico's headquarters. Anyone is free to publish, use or distribute the original Mini QRNG (OpenQbit) extensions, either as source code or as compiled binaries, for any purpose, commercial or non-commercial, and by any means.

The previous paragraph applies to the code and documentation deliverable in Mini QRNG those parts of the Mini QRNG library that actually group and ship with a larger application. Some scripts used as part of the compilation process (for example, "configuration" scripts generated by autoconf) may be included in other open source licenses. However, none of these compilation scripts make it into the final QRNG Mini deliverable library, so the licenses associated with those scripts should not be a factor in evaluating your rights to copy and use the QRNG Mini library.

All the deliverable code in Mini QRNG has been written from scratch. No code has been taken from other projects or from the open internet. Each line of code can be traced back to its original author, and all those authors have public domain dedications on file. Therefore, the QRNG Mini code base is clean and not contaminated with code licensed from other open source projects, not open contribution

Mini QRNG is open source, which means you can make as many copies as you want and do what you want with those copies, without limitation. But Mini QRNG is not open source. To keep Mini QRNG in the public domain and ensure that the code is not contaminated with proprietary or licensed content, the project does not accept patches from unknown people. All code in Mini QRNG is original, as it has been written specifically for use by Mini QRNG. No code has been copied from unknown sources on the Internet.

Mini QRNG is in the public domain and does not require a license. However, some organizations want legal proof of their right to use Mini QRNG. The circumstances where this occurs include the following:

- Your company wants compensation for claims of copyright infringement.
- You are using Mini QRNG in a jurisdiction that does not recognize the public domain.
- You are using Mini QRNG in a jurisdiction that does not recognize an author's right to dedicate their work to the public domain.
- You want to have a tangible legal document as evidence that you have the legal right to use and distribute Mini QRNG.
- Your legal department tells you that you must buy a license.

If any of the above circumstances apply to you, OpenQbit, the company that employs all Mini QRNG developers, will sell you a Mini QRNG Title Guarantee. A Title Warranty is a legal document that states that the claimed authors of Mini QRNG are the true authors, and that the authors have the legal right to dedicate the Mini QRNG to the public domain, and that OpenQbit will vigorously defend itself against the licensing claims. All proceeds from the sale of Mini QRNG title warranties are used to fund the continuous improvement and support of Mini QRNG.

Contributed Code

To keep Mini QRNG completely free and royalty free, the project does not accept patches. If you would like to make a suggested change and include a patch as a proof of concept, that would be great. However, don't be offended if we rewrite your patch from scratch. The type of non-commercial or opensource license who uses it in this modality and some similar without purchase of support either individual or corporate use no matter the size of the company will be governed by the following legal premises.

Warranty disclaimer. Unless required by applicable law or agreed to in writing, Licensor provides the Work (and each Contributor provides its Contributions) "AS IS", **WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either** express or implied, including, without limitation, any warranties or conditions of TITLE, NONINFRINGEMENT, MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. You are solely responsible for determining the correct use or redistribution of the Work and for assuming any risks associated with your exercise of permissions under this License.

Any financial or other losses incurred by the use of this software will be borne by the affected party. All legal disputes the parties will submit to courts only in the jurisdiction of Mexico City, country Mexico.

For commercial support, use and licensing an agreement or contract must be established between OpenQbit or its corporate and the interested party.

The terms and conditions of distribution marketing may change without notice, please go to the official website www.openqbit.com to see any changes to support and licensing clauses non-commercial and commercial.

Any person, user, private or public entity of any legal nature or from any part of the world who simply uses the software accepts without conditions the clauses established in this document and those that can be modified at any time in the portal of www.openqbit.com without previous notice and can be applied at the discretion of OpenQbit in non-commercial or commercial use.

Any questions and information about Mini QRNG should be directed to the App Inventor community or to the various Blockly system communities as they are: AppBuilder, Trunkable, etc. and/or to the email opensource@openqbit.com for the demand of questions can take 3 to 5 working days to be answered.

Support with commercial use.
support@openqbit.com

Sales for commercial use.
sales@openqbit.com

Legal information and licensing questions or concerns
legal@openqbit.com