COINSOLIDATION
By OpebQbit.

EX change tensions

COINsolidation.

Tallinn, Estonia. (E-residency)

White Paper.

version 1.0.0

December 2020.

COINsolidation International merged with www.OpenQbit.com for the cooperation of quantum mechanics based technology (Quantum Security & Quantum Computing). This merger allows to use, share and reengineer the technology developed by OpenQbit Inc. (Estonia, E-residency)

Content

1. Introduction.

Currently, mergers of companies are up to date, whether it is for an economic, technological or market good.

We present the first model of crypto fusion or crypto-tokens that offers a backup between two cryptomonads, tokens or a mixture of these, based on an algorithm to create a consolidated address that is used and generated in the COINsolidation environment.

We created three types of consolidated addresses.

The **CUA** (Consolidated Universal Address) is used to consolidate and create a new (active) token to be used by the user. The combination can be of three types: Cryptocurrency-Cryptocurrency, Cryptocurrency-Token or Token-Token. In the case of the CUA it is formed by a Token-Token relationship.

HAC (Hibric Address Consolidated) is used when we need to consolidate an address referring to a cryptocurrency and/or token and a normal asset transfer address.

The **DAC** (Dual Address Consolidated) is used to manage and consolidate two normal addresses from the same blockchain or from two different technologies, they are simple addresses for asset transfer.

Let's start by looking at the advantages of the CUA.

A CUA address is formed by the address of the COINsolidation token (static address) and an additional token known as the "Colored Coin" (variable address). In this case we can see that the CUA addresses will always be formed by addresses of some kind of combination of assets (Cryptcoins or tokens).

In our case, when we consolidate the COINsolidation token and an OAP token we will know it as the "CUA genesis" or **CUAG (Consolidated Universal Address Genesis).**

El token COINsolidation esta creado en el ***blockchain Ethereum*** y usa el standard ERC20 (Ethereum Request for Comments 20).

The "Colored coin" token is based on and created by the ***Bitcoin blockchain and*** uses the Open Assest Protocol **(OAP)** standard.

Let's start reviewing what the potential and benefit of consolidating addresses is.

I. For the users that create a CUA it will be possible to create a token (**OAP**) that can be customized by the user who created the CUA, the user will have the possibility to have his own token or crypto active so that he can use it in the creation, support or expansion of his business(es), in a simple and easy way he will have an asset in the world of crypto-tokens.

II. For companies that create a CUA, they may have a token (**OAP) that they** can use to create value in their supply chain or use the asset in liquidity transactions based on the economic support of their company assets and liabilities.

III. For existing cryptomonads and tokens, by creating a CUA, they will be able to use their address that identifies their asset and by consolidating it with the token (**OAP**) they will be able to grow their demand by offering their current and future investors their own token for their users.

Example of **CUAG**, we have the respective addresses of two different Blockchain:

Bitcoin- Token address - (OAP).

akXma4vqxvmEqnVAKSM953wYsnjNBhN3GM7

Address Ethereum - Token COINsolidation - (ERC20).

0x8390f8abb8fd8ad3bf8457db59f2ed75e015d303

Applying an algorithm to consolidate the previous addresses we obtain the CUA address.

cua50d0615d303k8X3m9a04fv8qaxbvbm8Efqdn8VaAdK3SbMf985435w7Ydsbn5j9NfB2heNd37G5Me70

* For more details of the algorithm refer to paragraph 7.- "Algorithm for the creation of a consolidated universal address".

We have as a result a single direction that represents two different technologies from two different directions consolidated in a single direction.

We reflect this in the field of profitability and financial expansion in a simple and straightforward way by investing in one of the tokens that integrates our CUA immediately you will get a token based on the Bitcoin blockchain (OAP).

Now let's look at two guidelines that we also put forward in COINsolidation for the world of cryptomontages and/or tokes.

COINsolidation token is the project to consolidate addresses and have an immediate support in obtaining a custom token to use in the growth of each user in the world of cryptomontages. The project was born in 2018 with a group of engineers and financiers interested in merging the financial and technological sectors, to take advantage of investment funds and occupy innovative technology such as quantum computing to give security to assets, as well as the purpose of using tools that could be available to everyone.

After an evaluation of several development possibilities we chose the option of the visual programming methodology Blockly this methodology is based on using extensions or modules (programs in java programming language) with simple but powerful functionality to expand the business in crypto active for anyone, to achieve this we must cover the following points:

- ✓ a.- Immediate financial ROI for users, investors and assets by being able to create a non tangible asset (personal tokens) for the exclusive use of the creator and user of the (CUA).
- ✓ b.- We use the advantages of joining two blockchains at user's selection to grow current and future investments in the crypto active market using the (CUA).
- ✓ c.- Facilitating the administration of separate addresses by consolidating them in (DAC).
- ✓ d.- Create and use security based on Quantum Computing.

The challenge began in the creation of the technology of extensions sufficiently modular in functionality and "size" this last one was the challenge of the development team of COINsolidation since the extensions that are used in the Blockly methodology and systems of this type (AppInventor, AppyBuilder, Thunkable, Kondular, etc) are usually extensions (programs) created that do not exceed 100k - 300k bytes, with the restrictions that have in their size the task of creating extensions for use in the current Blockchain were virtually impossible due to the libraries used in their creation exceed between 10MB and 35MB these sizes for the current tools Blockly systems are not functional to use them.

The team had to create, adapt and minimize programming methodology and libraries in order to obtain the extensions with the optimal functionality, security and size.

After almost two years of development and testing we have completed the first blockchain "beta" using extensions for Blockly including the consensus algorithm "Proof of Quantum" using quantum security for the exchange of crytomonies.

Currently we have a proprietary blockchain that has been released for "Beta" testing and at the end of 2021 we will be launching the production version for information distribution. Currently our COINsolidation Token is based on the Ethereum and Bitcoin blockchains, the latter for the creation of custom tokens for users.

2. Security Quantum Computing.

How does quantum computing work? (2)

The digital transformation is generating changes 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.

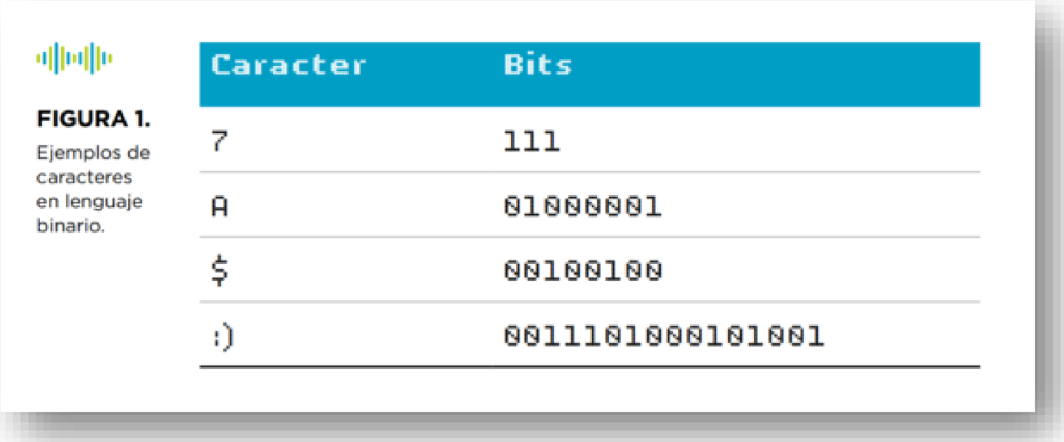


FIGURA 1. Ejemplos de caracteres en lenguaje binario.

Caracter	Bits
7	111
A	01000001
$	00100100
:)	0011101000101001

To understand how quantum computers work, it is helpful to first explain how the computers we use every day, which we will refer to in this document as digital or classic computers, work. They, like other electronic devices such as tablets or cell 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 states zero and one of the bits correspond to electrical current that circulates, or not, through microscopic pieces 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.

In a more simplified way, 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. As an example, figure 1 shows the binary writing of some characters. Now that we have an idea of how current 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 now- that like bits can be in the low level, which corresponds to a low excitation or energy state defined as 0, or in the high level, which corresponds to a higher excitation or energy state 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. This phenomenon is known as quantum superposition and is natural in quantum systems.

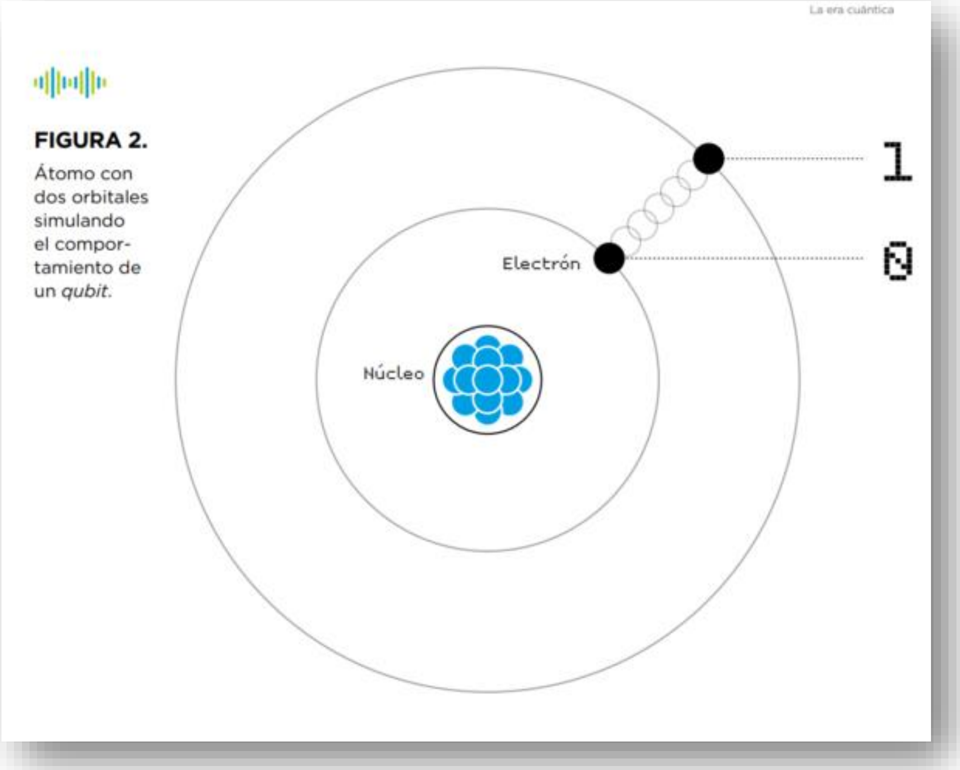


FIGURA 2. Átomo con dos orbitales simulando el comportamiento de un *qubit.*

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 interlacing to provide much greater processing power than the classics. It is important to indicate 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 implies. Let's imagine that we are in Bogotá and we want to know which is 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 classical 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, although 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 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 one known as 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 are and the scarcity of developers and people with technical profiles mean that convergence is still a pending task.

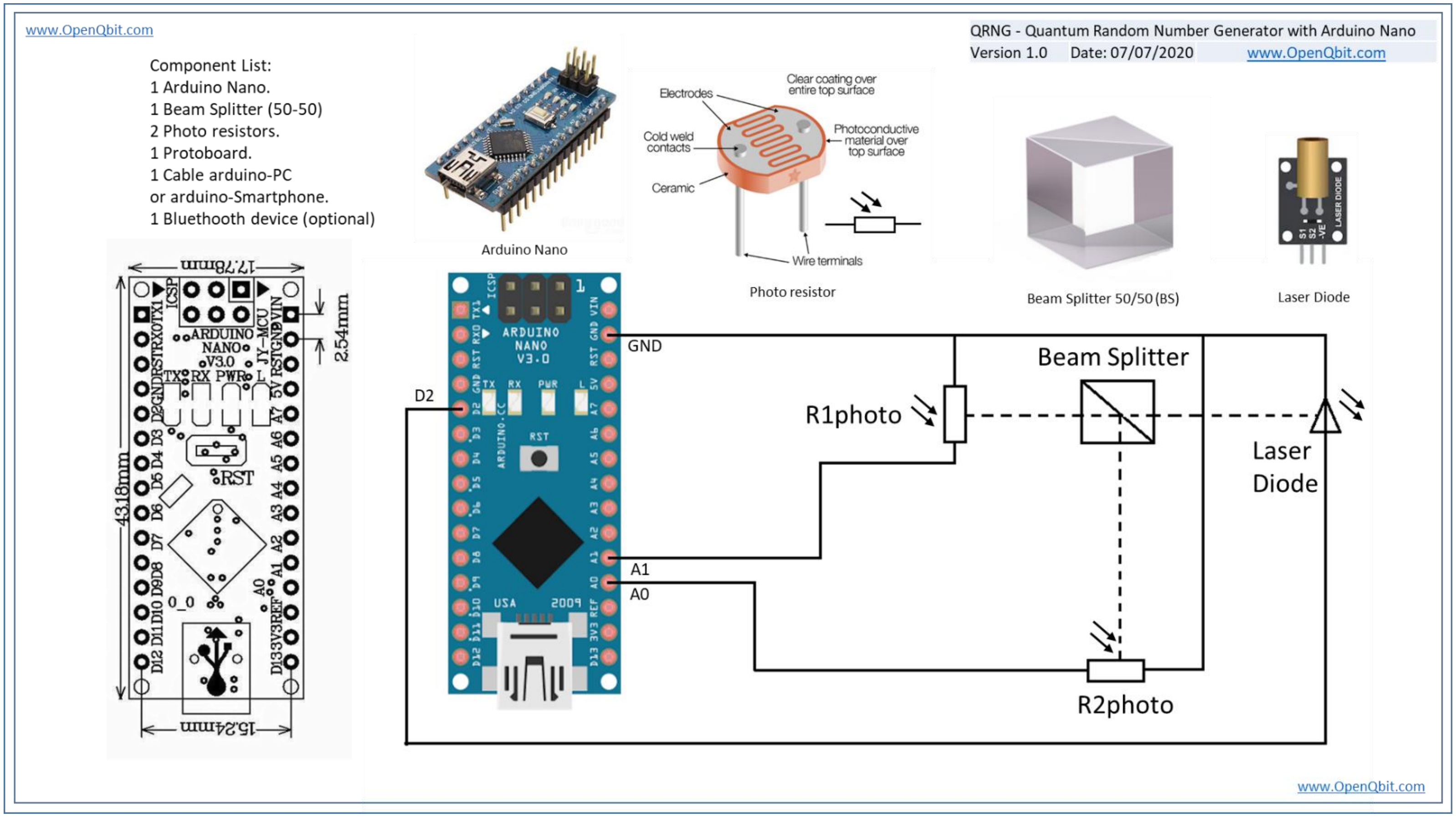
Quantum technologies, because of their disruptive potential, are expected not only to converge with all these new technologies, but to have a transversal influence on practically all of them. Quantum computing will threaten the authentication, exchange and secure storage of data, having a greater impact on those technologies in which cryptography has a more relevant role, such as cyber security or blockchain, and a lesser 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 like IBM and Rigetti.

3. Creation of a "Hardware" device of a QRNG (Quantum Random Number Generator).

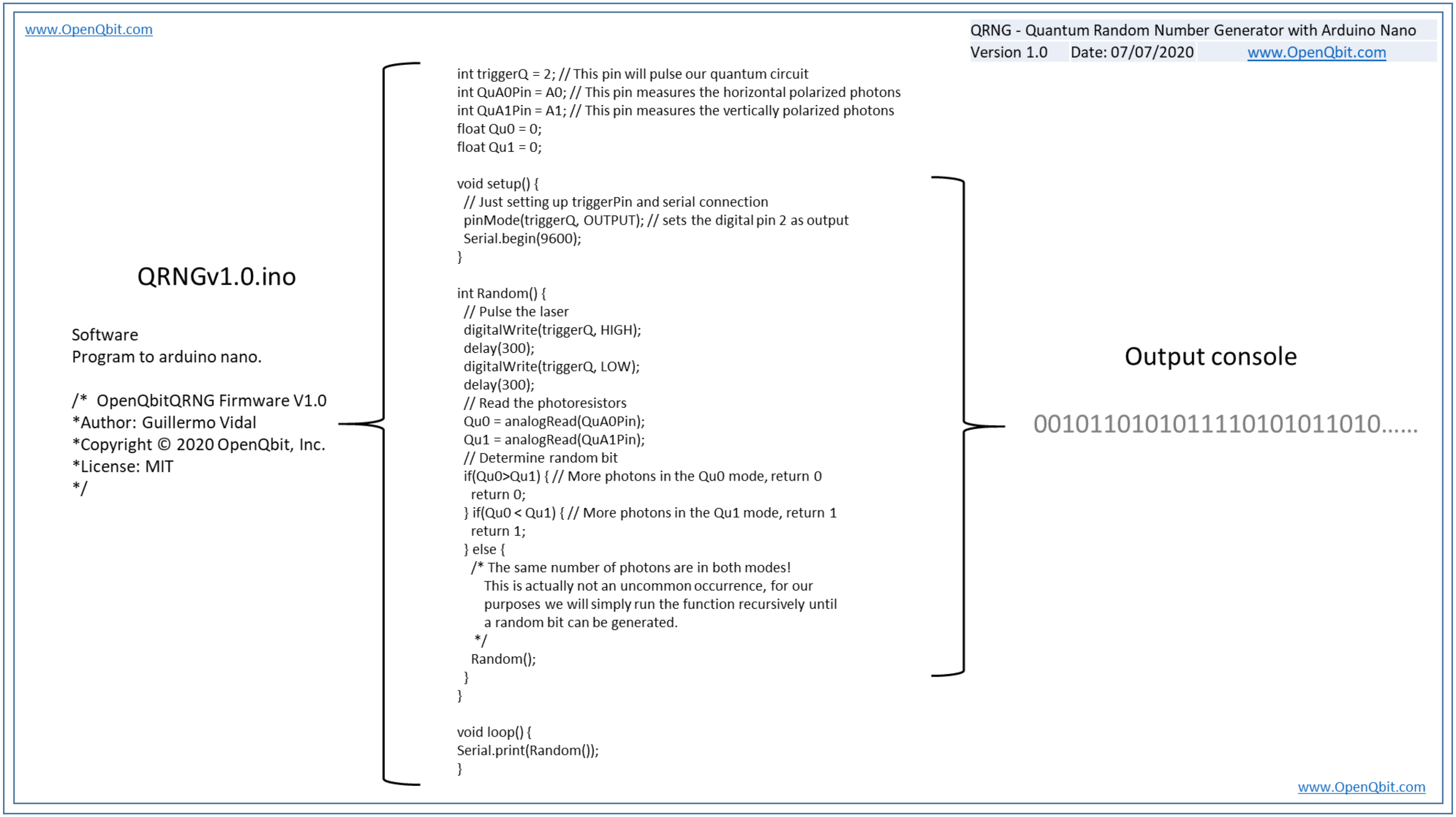
We will now create a physical "Hardware" device to generate Quantum Random Numbers (QRNG) with inexpensive components that can be easily assembled at home and cost approximately $35 USD.



www.OpenQbit.com

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

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)



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.

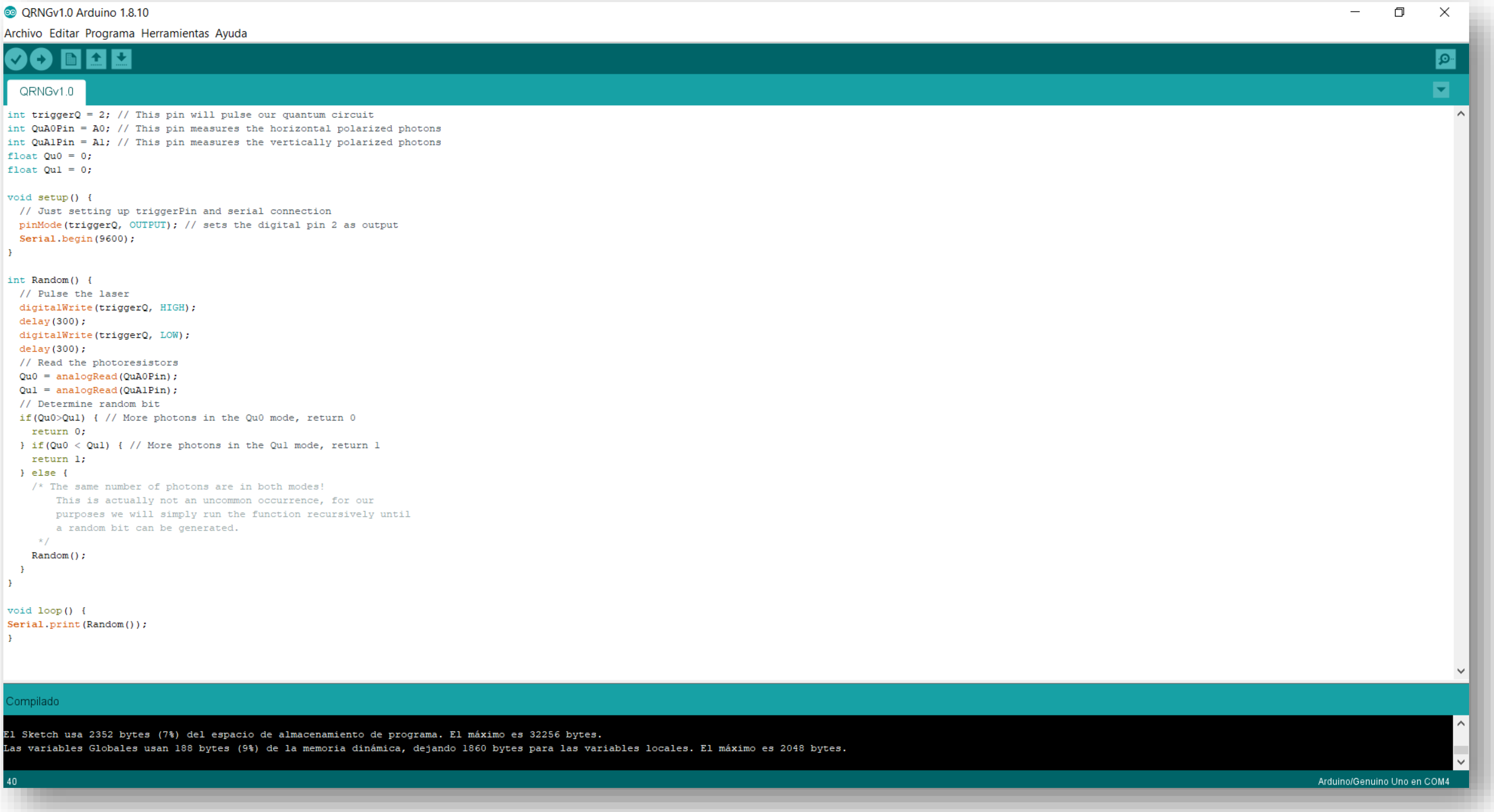
/* OpenQbitQRNG Firmware V1.0
*Author: Guillermo Vidal
*Copyright © 2020 OpenQbit, Inc.
*License: MIT
*/

```
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......
```

Compiling the program QRNGv10.ino and uploading to arduino nano….



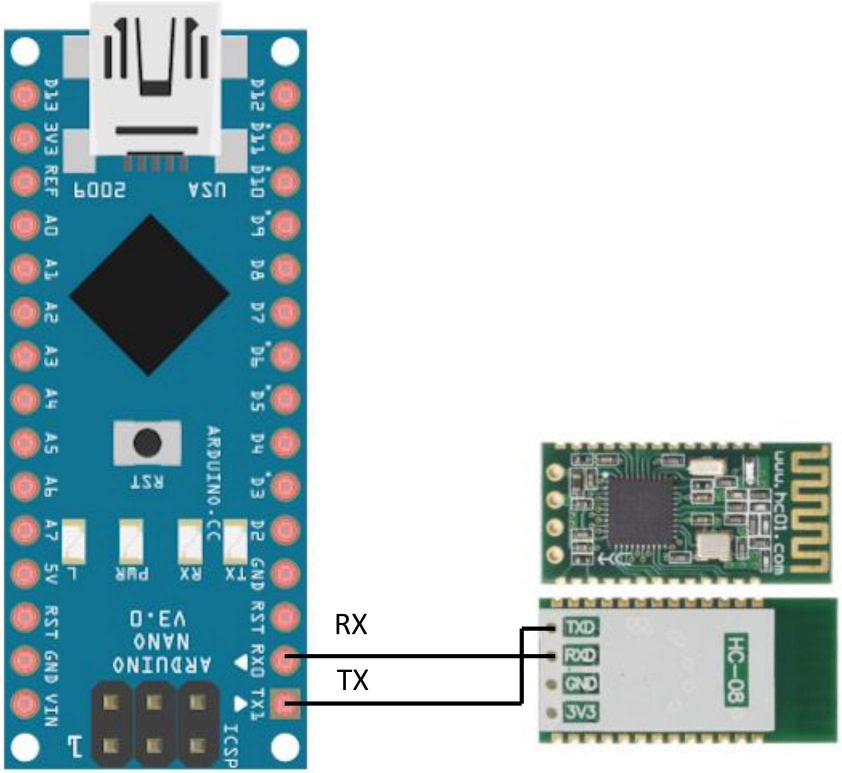
```
int triggerQ = 2; // This pin will pulse our quantum circuit
int QuA0Pin = A0; // This pin measures the horizontal polarized photons
int QuA1Pin = A1; // This pin measures the vertically polarized photons
float Qu0 = 0;
float Qu1 = 0;

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());
}
```

Compilado

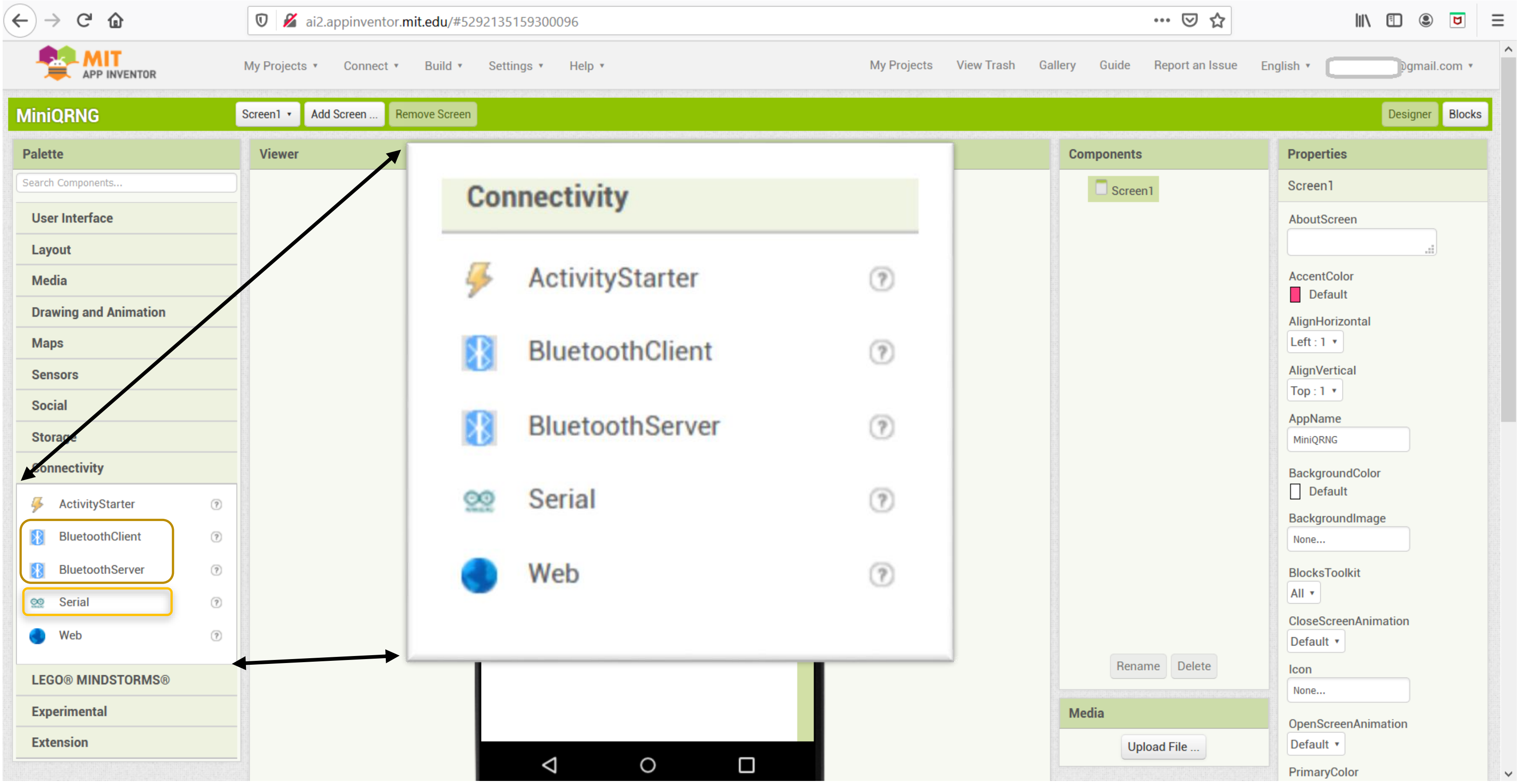
```
El Sketch usa 2352 bytes (7%) del espacio de almacenamiento de programa. El máximo es 32256 bytes.
Las variables Globales usan 188 bytes (9%) de la memoria dinámica, dejando 1860 bytes para las variables locales. El máximo es 2048 bytes.
```

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

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



To connect App Inventor with Arduino you can use the following components Serial or Bluetooth:



AboutScreen

AccentColor
Default

AlignHorizontal
Left : 1 ▾

AlignVertical
Top : 1 ▾

AppName
MiniQRNG

BackgroundColor
Default

BackgroundImage
None...

BlocksToolkit
All ▾

CloseScreenAnimation
Default ▾

Icon
None...

OpenScreenAnimation
Default ▾

PrimaryColor

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 type system may be through similar bluetooth online.

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

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

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

To review the whole project of design and use of QRNG (Quantum Random Number Generator) extensions. Review the user manual at:

https://github.com/COINsolidation/UserGuide

4. What is Proof of Quantum (PQu)?

PoQu. - "Proof of Quantum" is a consensus algorithm developed for Mini BlocklyChain and COINsolidation, this test is a variant of the Proof of Work (PoW) that works as follows.

The Test of Quantum (PoQu) at startup is executed with the same algorithm of the "Test of Work" (PoW) is based on putting the processor of the device (PC, Server, Tablet or Mobile Phone) to work to obtain a string of characters that is a mathematical puzzle called "hash".

Remember that a "hash" is an algorithm or mathematical process that when introducing a phrase or some type of digital information such as text files, program, image, video, sound or other diverse type of digital information gives us as a result an alphanumeric character that represents the digital signature that represents it in a unique and non-repeatable way of the data, the hash algorithm is unidirectional, this means that when entering a data to obtain its signature "hash" its reverse process can not be performed, to have a signature "hash" we can not know what information was obtained this property gives us a security advantage to process the information we send over the Internet. How does it work? Imagine sending any type of information through non-secure channels and accompany that information with its respective "hash origin", the receiver when receiving the information can get the "hash" of the information received we will call it "destination hash" and check it with the "hash origin" if both "hashes" are the same we can confirm that the information has not been altered in the channel that was sent, is just an example where this type of information security process is currently used.

Currently there are different types of algorithms or hash processes that differ in the level of security. The most used or known are: MD5, SHA256 and SHA512.

Example of SHA256:

We have a chain or phrase as follows: "Mini BlocklyChain is modular"

If we apply a SHA256 hash to the previous string it will give us the next hash.

f41af7e61c3b02fdd5e5c612302b62a2dd52fcb38f9de97cb2afd827e8804db8

The above alphanumeric string is the signature that represents the sentence in the above example

For more example we can use the site on the internet:

https://emn178.github.io/online-tools/sha256.html

In the case of the "Test Work" (PoW) algorithm, it works by using computing power to obtain a predefined hash.

Let's imagine that we have the previous "hash" that we took from the chain "Mini BlocklyChain is modular".

f41af7e61c3b02fdd5e5c612302b62a2dd52fcb38f9de97cb2afd827e8804db8

To this "hash" in its beginning we put the parameter of difficult that is simply to put zeros "0" in the beginning, that is to say if we say that the difficulty is of 4 it will have "0000" + "hash" to this we will call him "hash seed".

0000 f41af7e61c3b02fdd5e5c612302b62a2dd52fcb38f9de97cb2afd827e8804db8

Now taking into account that we know the input information that is the string: "Mini BlocklyChain is modular" we add at the end of the string a number starting from zero "0" and we take out its hash to this we will call it "hash nonce":

f41af7e61c3b02fdd5e5c612302b62a2dd52fcb38f9de97cb2afd827e8804db80

We took out "hash nonce":

7529f3ad273fc8a9eff12183f8d6f886821900750bb6b59c1504924dfd85a7c8

Then we make a comparison of the new "hash nonce" with the "hash seed" if they are equal the node that first finds the equality will win the execution of processing the current transaction. As we can see this process is based on probability and computational force that the device has which gives the "Proof of Work" test a consensus equity for all nodes.

If the "seed hash" does not coincide with the "hash nonce", it increases the difficulty by one and the "hash nonce" is taken out again, the number that is being increased we will call it "nonce" number, it is compared with the "seed hash" until they coincide or are equal.

As we can see the number "nonce" or increase is the one that will help to obtain the "hash" of equality.

Based on the "Test Work" (PoW) algorithm, the Quantum Test (PoQu) algorithm is based on obtaining the number "nonce" as PoW does and using a minimum level difficulty ranging from 1 to 5, this serves only to the mobile device to gain the right to be a candidate to win the consensus.

The Quantum Test (PoQu), is activated when the cell phone has finished the minimum PoW and wins the pass to obtain a probability number in the QRNG system.

The QRNG (Quantum Random Number Generator) is a random quantum number generator, this system is based on generating true random numbers based on quantum mechanics is the safest system today to generate such numbers. For more details see "Quantum Computing Security" in index 3.

COINsolidation can implement both minimum PoW and PoQu concession types.

The PoQu test is based on obtaining the number "nonce". This number in the PoQu test is known as the "Magic Number" and with this the "Peer to Peer" system will confirm if the number is correct and then a random number will be obtained with the COINsolidation QRNG server pool. This random number will be registered in all the nodes, a list will be created containing ((Node Addition) /2)) +1 and from this list the one with the highest percentage of probability to be the winner candidate of the consensus (PoQu) will be chosen and it will execute the current transaction queue.

The PoQu algorithm also uses NIST (National Institute of Standards and Technology) testing to assure us that the random numbers in the QRNG are truly random numbers.

https://nvlpubs.nist.gov/nistpubs/legacy/sp/nistspecialpublication800-22r1a.pdf

In COINsolidation we have implemented a block for PoW and a block for PoQu. These blocks use a hash type: SHA256 for free use, for commercial use we have a SHA512 and other hashes as required.

For more details on the concept of HASH see:

https://es.wikipedia.org/wiki/Funcion_hash

NOTE: The Test of Work (PoW) used in cell phones can only use a maximum difficulty of 5 since the mathematical processing of these devices is not dedicated like servers or PCs. The algorithm (PoW) is only used to obtain the opportunity to obtain your pass or permission to enter the Quantum Random Number Generator (QRNG) system and with it to execute the Test of Quantum (PoQu) algorithm. See use of (PoQu) in Mini BlockyChain:

https://github.com/openqbit-diy/MiniBlocklyChain

5. Algorithm for the creation of a consolidated universal address (CUA)

Our CUA (Consolidated Universal Address) addresses are created using the following algorithm:

Step 1.- Identifiers are removed from the respective addresses, they are the alphanumeric characters that identify the address from which blockchain were created.

Bitcoin- Token address - (OAP).

akXma4vqxvmEqnVAKSM953wYsnjNBhN3GM7

Address Ethereum - Token COINsolidation - (ERC20).

0x9d08c0ac0f2fdf078c883db6fa617b15776e4b41

Step 2.- The SHA512(address String-Text) of each address without its initial identifier is obtained by removing the "a" from A1 and the "0x" from A2 and taking the last two characters of each operation hash symbolized with "U". Verification numbers.

U(SHA512(kXma4vqxvmEqnVAKSM953wYsnjNBhN3GM7)) = 50

U(SHA512(9d08c0ac0f2fdf078c883db6fa617b15776e4b41)) = fb

Step 3.- The characters of each address are concatenated one by one starting from the address that has less characters that compose it, in case of having the same amount of characters the concatenation can start from any address.

Address 1 = A10 [0], A11 [1], A12 [2], A13 [3], A14 [4] ……… A1N[n], A1N+1[n+1].

Address 2 = A20 [0], A21 [1], A22 [2], A23 [3], A24 [4] ……… A2N[n], A2N+1[n+1].

Concatenation of addresses:

A10 [0] + A20 [0] + A10 [1] + A20 [1] + A11 [2] + A22 [2] + …. A1N+1 [n+1] + A2N+2 [n+1]

**Last characters that cannot be concatenated are put at the beginning of the string.

6e4b41k9Xdm0a84cv0qaxcv0mfE2qfndVfA0K7S8Mc985833wdYbs6nfjaN6B1h7Nb31G5M777

Step 4.- The number of characters that could be concatenated in step 3 is added to the beginning of the string resulting from step 3.

66e4b41k9Xdm0a84cv0qaxcv0mfE2qfndVfA0K7S8Mc985833wdYbs6nfjaN6B1h7Nb31G5M777

Step 5.- The two pairs of verifiers from step 2 of each direction are concatenated at the beginning of the chain resulting from step 3 in the same order A1 + A2.

50fb66e4b41k9Xdm0a84cv0qaxcv0mfE2qfndVfA0K7S8Mc985833wdYbs6nfjaN6B1h7Nb31G5M777

Step 6.- The CUAG (Consolidated Universal Address Genesis) identification is integrated at the beginning of the address created in step 5.

cua50fb66e4b41k9Xdm0a84cv0qaxcv0mfE2qfndVfA0K7S8Mc985833wdYbs6nfjaN6B1h7Nb31G5M777

**In the case of Bitcoin and Ethereum address consolidation it will give an address consisting of 82 hexadecimal characters.

6. Algorithm for dual consolidated address (DAC) and (HAC).

The creation of a DAC is the same as the CUA, the difference is that in as DACs are used to consolidate normal addresses to receive transactions, these addresses do not represent any cryptomoneda or token.

Step 1.- Identifiers are removed from the respective addresses, they are the alphanumeric characters that identify the address from which blockchain were created.

18gYNA9c2G9X8HZ8QxWLpLXZauAxFnsJbe (Bitcoin Address)

0x5d2Acdb34c279Aa6d1e94a77F7b18aB938BFb2bB (Dirección Ethereum)

Step 2.- The SHA512(address String-Text) of each address without its initial identifier is obtained by removing the "1" from A1 and the "0x" from A2 and taking the last two characters of each operation hash symbolized with "U". Verifier numbers.

U(SHA512(8gYNA9c2G9X8HZ8QxWLpLXZauAxFnsJbe)) = 48

U(SHA512(5d2Acdb34c279Aa6d1e94a77F7b18aB938BFb2bB)) = f3

Step 3.- The characters of each address are concatenated one by one starting from the address that has less characters that compose it, in case of having the same amount of characters the concatenation can start from any address.

Address 1 = A10 [0], A11 [1], A12 [2], A13 [3], A14 [4] ……… A1N[n], A1N+1[n+1].

Address 2 = A20 [0], A21 [1], A22 [2], A23 [3], A24 [4] ……… A2N[n], A2N+1[n+1].

Concatenation of addresses:

A10 [0] + A20 [0] + A10 [1] + A20 [1] + A11 [2] + A22 [2] + ... A1N+1 [n+1] + A2N+2 [n+1]

**Last characters that cannot be concatenated are put at the beginning of the string.

8BFb2bB85gdY2NAAc9dcb23G49cX287H9ZA8aQ6xdW1Lep9L4XaZ7a7uFA7xbF1n8saJBb9e3

Step 4.- The number of characters that could be concatenated in step 3 is added to the beginning of the string resulting from step 3.

78BFb2bB85gdY2NAAc9dcb23G49cX287H9ZA8aQ6xdW1Lep9L4XaZ7a7uFA7xbF1n8saJBb9e3

Step 5.- The two pairs of verifiers from step 2 of each direction are concatenated at the beginning of the chain resulting from step 3 in the same order A1 + A2.

48f378BFb2bB85gdY2NAAc9dcb23G49cX287H9ZA8aQ6xdW1Lep9L4XaZ7a7uFA7xbF1n8saJBb9e3

Step 6.- The DAC (Dual Address Consolidated) Identification is integrated at the beginning of the address created in step 5.

dac48f378BFb2bB85gdY2NAAc9dcb23G49cX287H9ZA8aQ6xdW1Lep9L4XaZ7a7uFA7xbF1n8saJBb9e3

**In the case of Bitcoin and Ethereum address consolidation it will give an address consisting of 81 hexadecimal characters.

For the case of HAC (Hibric Address Consolidated) applies in the previous what varies are the addresses that are used, in this case we will use an address that represents an asset (Cryptomoneda or token) and a normal standard address of asset transfers of some type of blockchain.

NOTE: The size of the CUA, HAC and DAC addresses may vary in each case depending on the addresses that make them up.

Project and solution by COINsolidation.

Currently there are different types of Blockchain oriented to assets of different characteristics, this leads to have an infinite number of types of addresses of daily use to have to keep a comprehensive control to avoid making transfer errors.

On the other hand, the world of cryptomoney and tokens is limited to financial experts or in their case experts in blockchain technology, so it is difficult for the average person to venture into the creation of their own cryptomoney or token.

We have solved the two previous problems in COINsolidation by making the following points and / or tools that we have created.

For the point of control of different blockchain addresses, we create an algorithm where it consolidates (joins) two or more addresses in their different combinations resulting in a single address of type CUA, HAC and/or DAC.

With this solution, instead of sending two addresses from the same or different blockchain, only one consolidated address will be used.

For the second problem we have used the programming methodology called Blockly, this is a visual tool where no great knowledge of programming is needed and any average person or company will be able to create their own applications without having to invest expensive development teams, time and money.

We have created the extensions (modules) to just install and use them to create mobile applications, in 15 minutes. Example your own crypto currency exchange or develop your own currency (token) in minutes. All of the above using state of the art data security called PQC (Post-Quantum Cryptography).

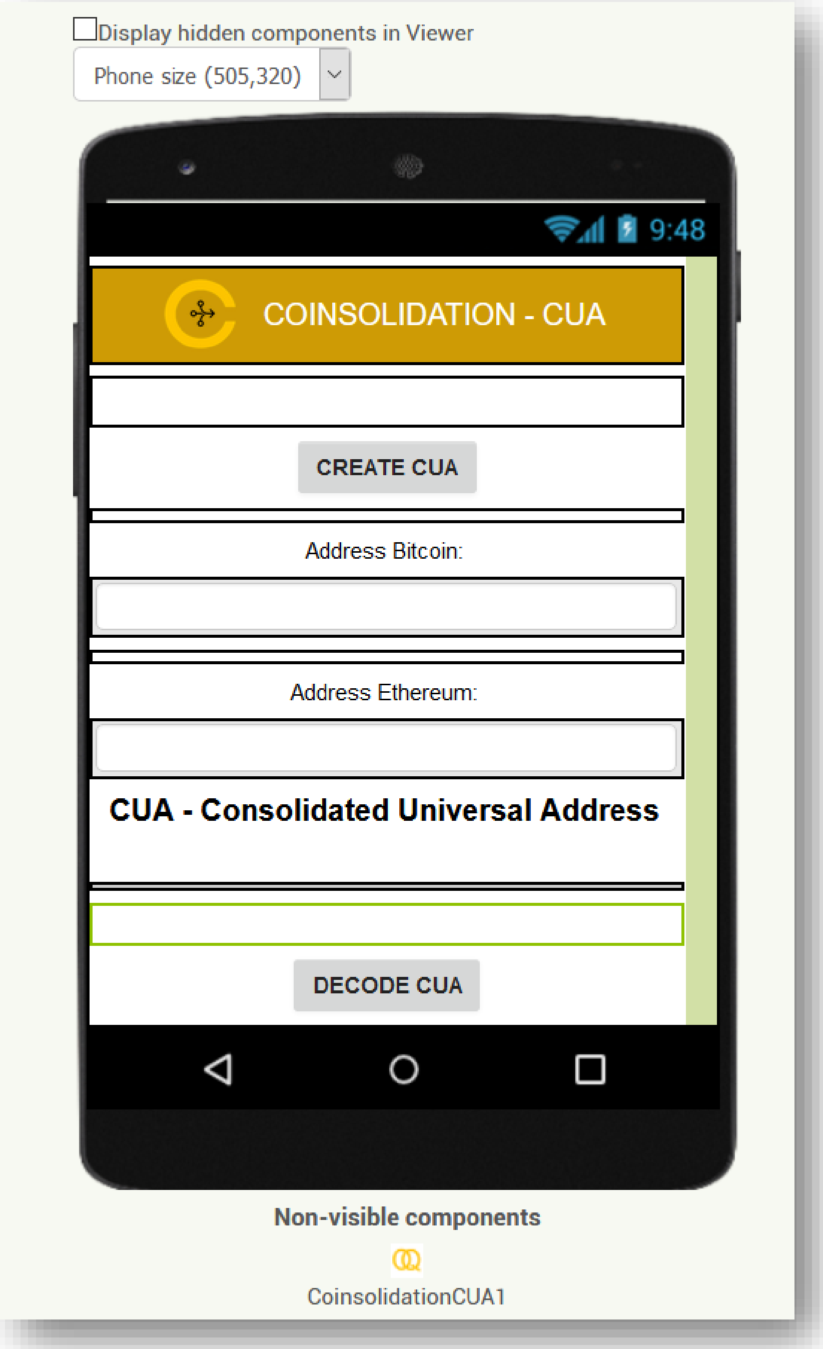
Just install the extensions on any free tool such as Appventor, AppyBuider, Thunkable, Kondular or others and in minutes you can enter the world of cryptomonies and token creation all in the palm of your hand.

Finally COINsolidation is creating the use of low-cost quantum security (software and hardware) that can be used in the protection of computer data at home. Currently, technologies based on quantum computing and security have a high cost only corporations with high financial level can create and use them. However, in COINsolidation we believe that new technologies should be available to everyone, the fairness of use of the Blockchain and Quantum Computing should be for everyone, we create free software (cryptomonics) and low-cost hardware (quantum security).

7. Creation of App CUA (Consolidated Universal Address) in 15 minutes.

*App for Bitcoin and Ethereum coins (BTC-ETH)

Design screen 5 minutes in https://appinventor.mit.edu/



□Display hidden components in Viewer

Phone size (505,320)

Address Bitcoin:

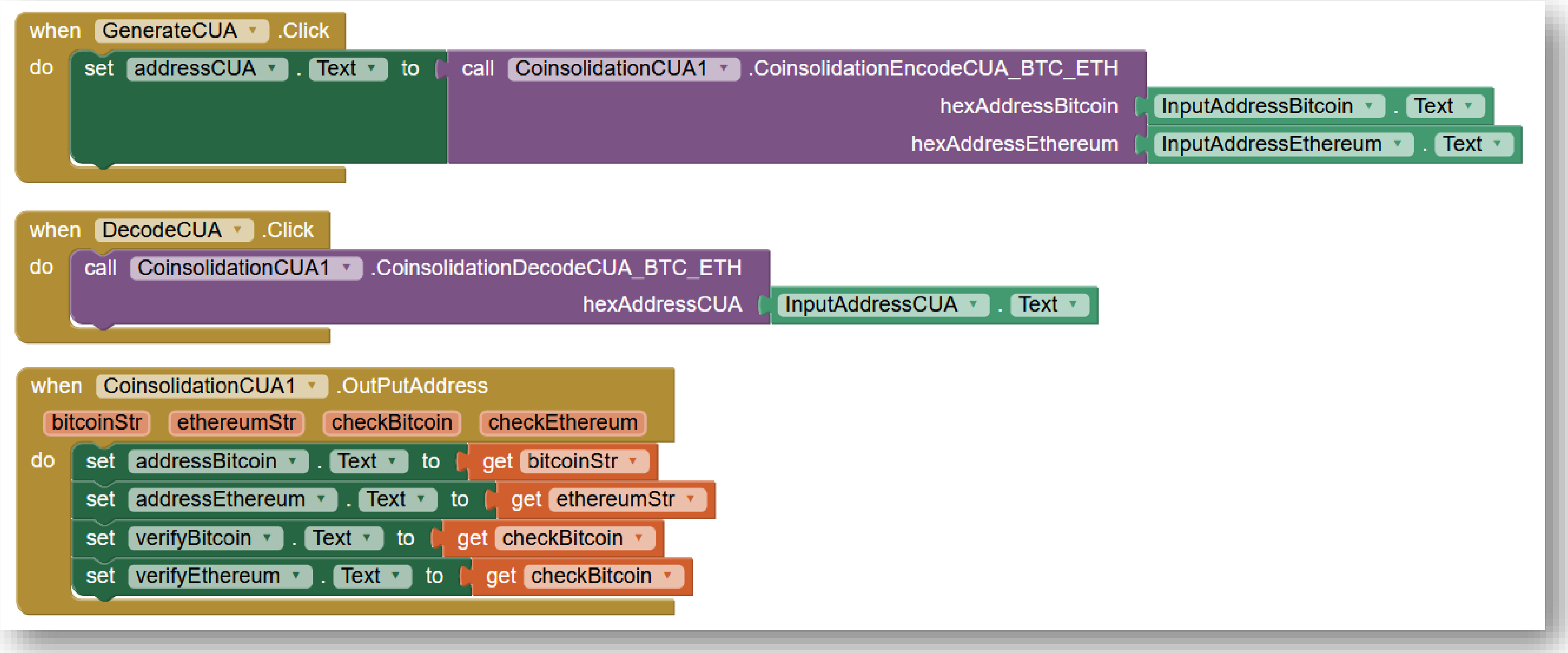
Address Ethereum:

CUA - Consolidated Universal Address

Non-visible components

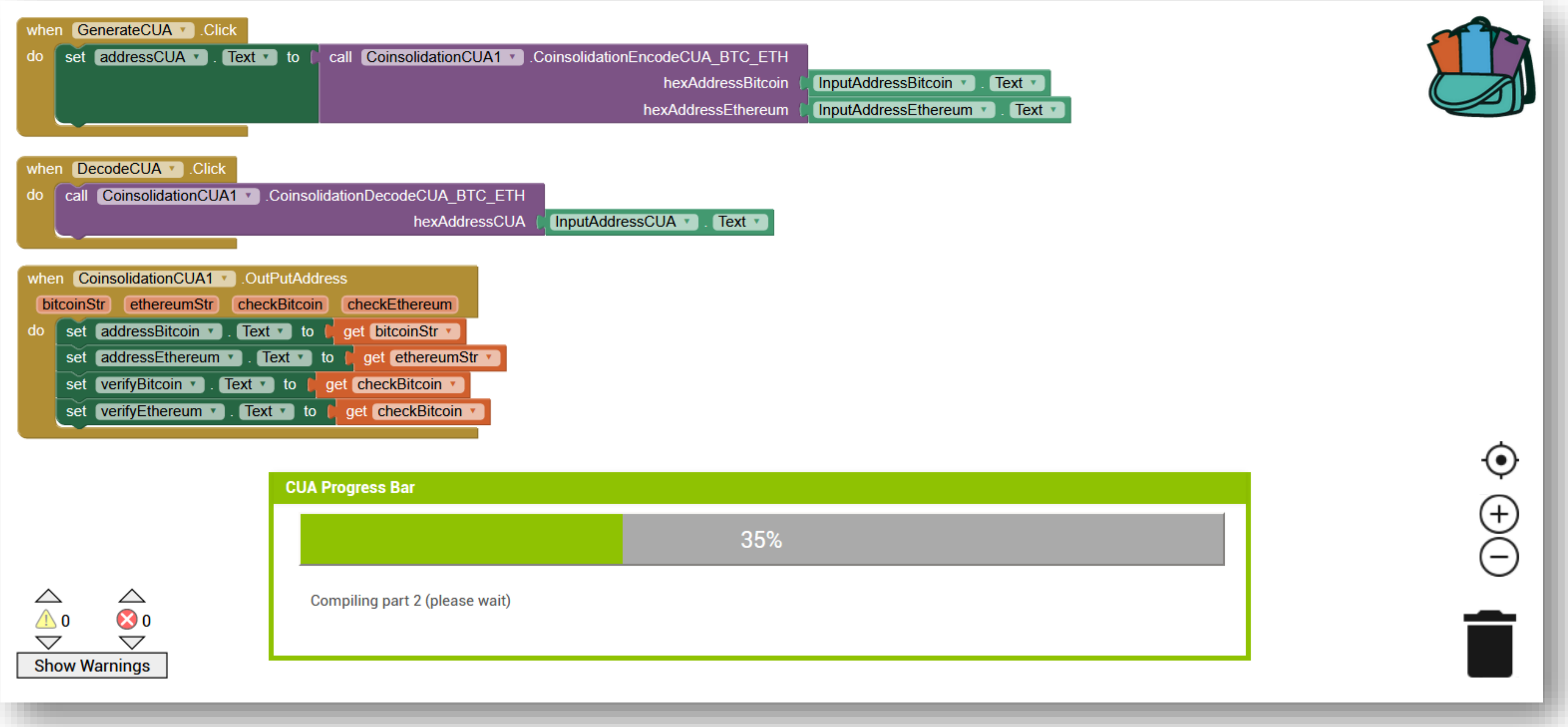
CoinsolidationCUA1

Use of extension **CoinsolidatioCUA.AIX** (5 minutes).



```
when GenerateCUA .Click
do  set addressCUA . Text to  call CoinsolidationCUA1 .CoinsolidationEncodeCUA_BTC_ETH
                                                 hexAddressBitcoin   InputAddressBitcoin . Text
                                                 hexAddressEthereum  InputAddressEthereum . Text

when DecodeCUA .Click
do  call CoinsolidationCUA1 .CoinsolidationDecodeCUA_BTC_ETH
                                      hexAddressCUA  InputAddressCUA . Text

when CoinsolidationCUA1 .OutPutAddress
 bitcoinStr  ethereumStr  checkBitcoin  checkEthereum
do  set addressBitcoin . Text to  get bitcoinStr
    set addressEthereum . Text to  get ethereumStr
    set verifyBitcoin . Text to  get checkBitcoin
    set verifyEthereum . Text to  get checkBitcoin
```

We create the application in **Menu > Build > App (provide QR code for .apk) - (5** minutes)



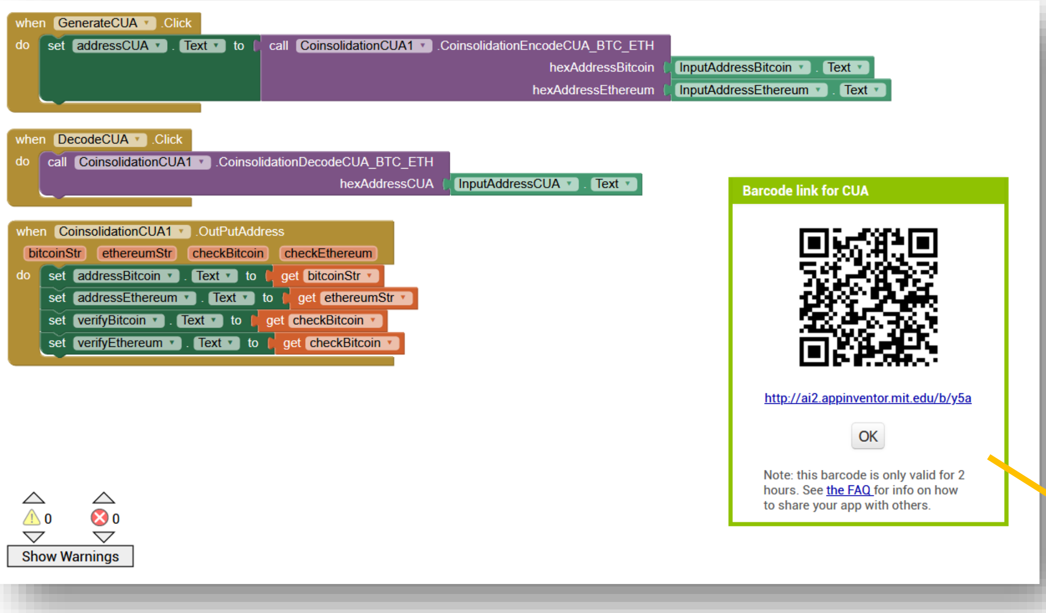
```
when GenerateCUA .Click
do  set addressCUA . Text to  call CoinsolidationCUA1 .CoinsolidationEncodeCUA_BTC_ETH
                                                     hexAddressBitcoin  InputAddressBitcoin . Text
                                                     hexAddressEthereum InputAddressEthereum . Text

when DecodeCUA .Click
do  call CoinsolidationCUA1 .CoinsolidationDecodeCUA_BTC_ETH
                                  hexAddressCUA  InputAddressCUA . Text

when CoinsolidationCUA1 .OutPutAddress
 bitcoinStr  ethereumStr  checkBitcoin  checkEthereum
do  set addressBitcoin . Text to  get bitcoinStr
    set addressEthereum . Text to  get ethereumStr
    set verifyBitcoin . Text to  get checkBitcoin
    set verifyEthereum . Text to  get checkBitcoin
```

CUA Progress Bar

35%

Compiling part 2 (please wait)

We installed the application on the cell phone from the QR using AppInventor's Android application (MIT AI2 Companion) - https://play.google.com/store/apps/details?id=edu.mit.appinventor.aicompanion3



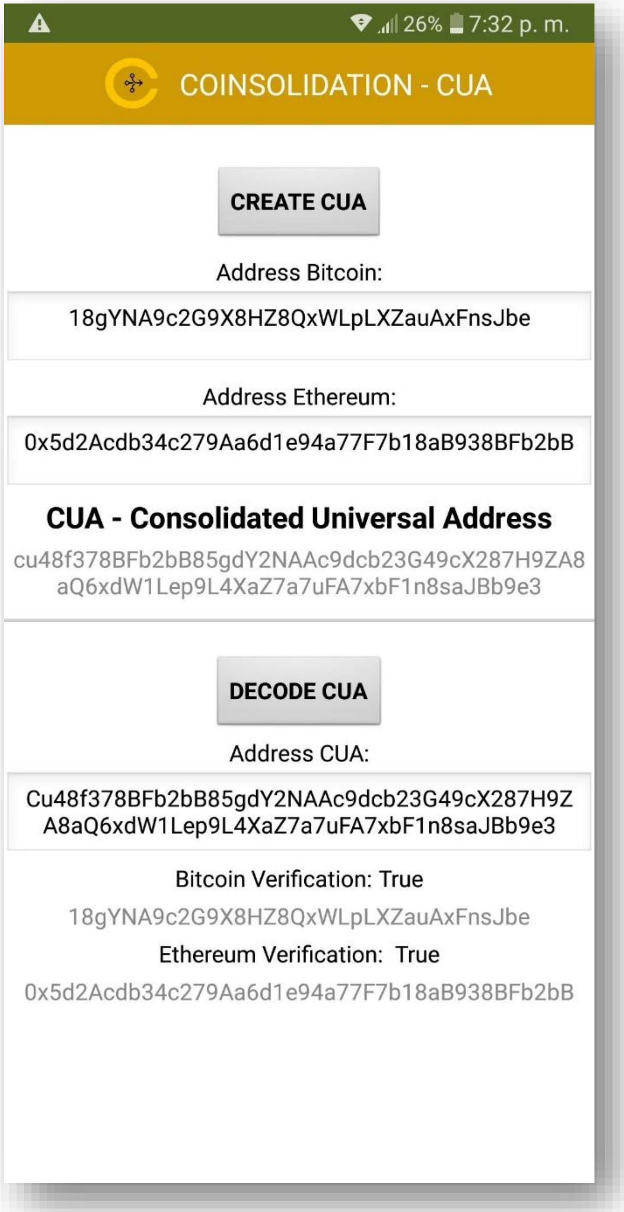
```
when GenerateCUA .Click
do  set addressCUA . Text to  call CoinsolidationCUA1 .CoinsolidationEncodeCUA_BTC_ETH
                                              hexAddressBitcoin   InputAddressBitcoin . Text
                                              hexAddressEthereum  InputAddressEthereum . Text

when DecodeCUA .Click
do  call CoinsolidationCUA1 .CoinsolidationDecodeCUA_BTC_ETH
                                  hexAddressCUA  InputAddressCUA . Text

when CoinsolidationCUA1 .OutPutAddress
 bitcoinStr  ethereumStr  checkBitcoin  checkEthereum
do  set addressBitcoin . Text to  get bitcoinStr
    set addressEthereum . Text to  get ethereumStr
    set verifyBitcoin . Text to  get checkBitcoin
    set verifyEthereum . Text to  get checkBitcoin
```

Barcode link for CUA

http://ai2.appinventor.mit.edu/b/y5a

OK

Note: this barcode is only valid for 2 hours. See the FAQ for info on how to share your app with others.



Address Bitcoin:

18gYNA9c2G9X8HZ8QxWLpLXZauAxFnsJbe

Address Ethereum:

0x5d2Acdb34c279Aa6d1e94a77F7b18aB938BFb2bB

CUA - Consolidated Universal Address

cu48f378BFb2bB85gdY2NAAc9dcb23G49cX287H9ZA8aQ6xdW1Lep9L4XaZ7a7uFA7xbF1n8saJBb9e3

Address CUA:

Cu48f378BFb2bB85gdY2NAAc9dcb23G49cX287H9ZA8aQ6xdW1Lep9L4XaZ7a7uFA7xbF1n8saJBb9e3

Bitcoin Verification: True

18gYNA9c2G9X8HZ8QxWLpLXZauAxFnsJbe

Ethereum Verification: True

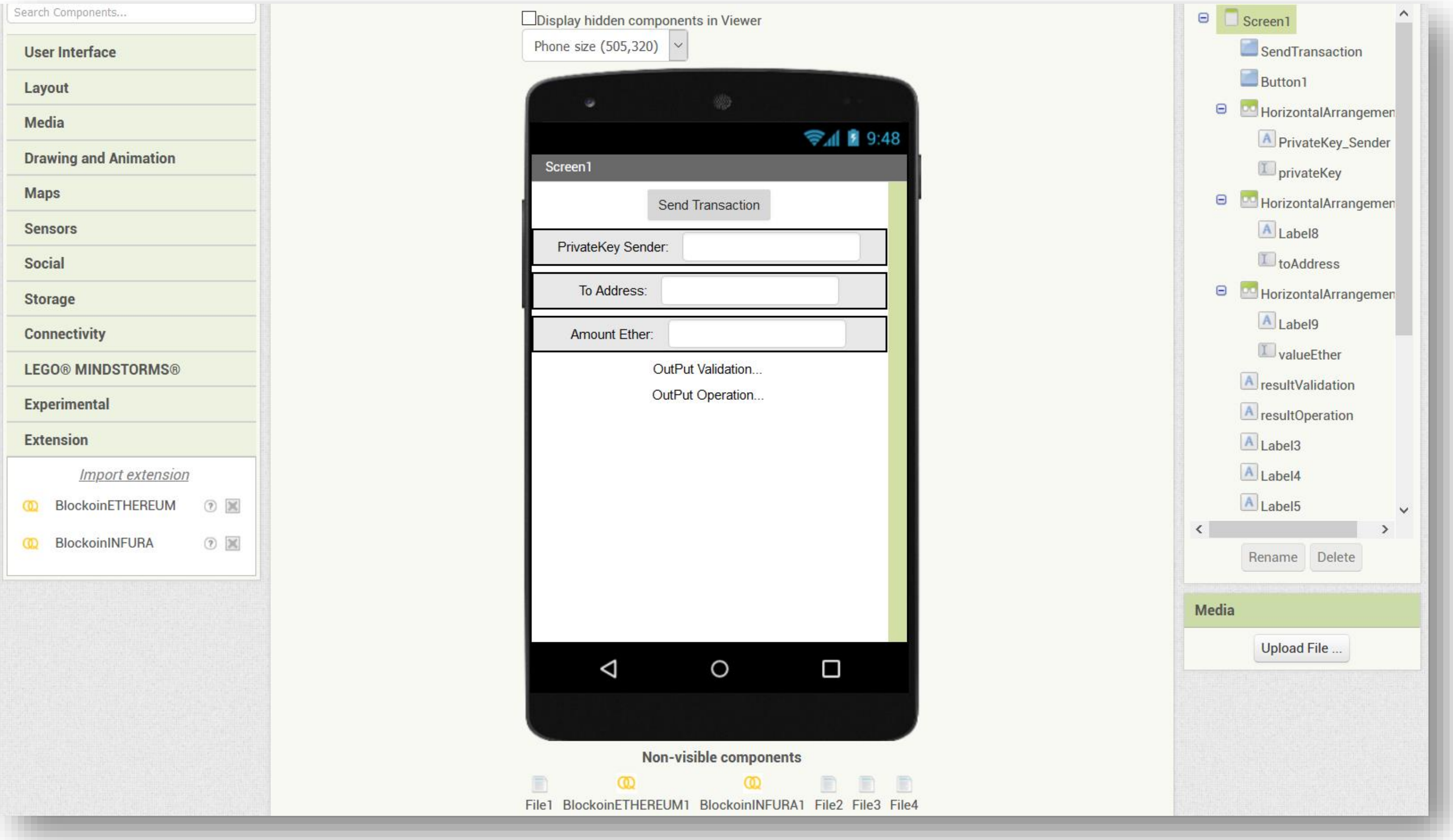
0x5d2Acdb34c279Aa6d1e94a77F7b18aB938BFb2bB

NOTE: The APK file application ready to be installed is located in the following repository: https://github.com/COINsolidation/App

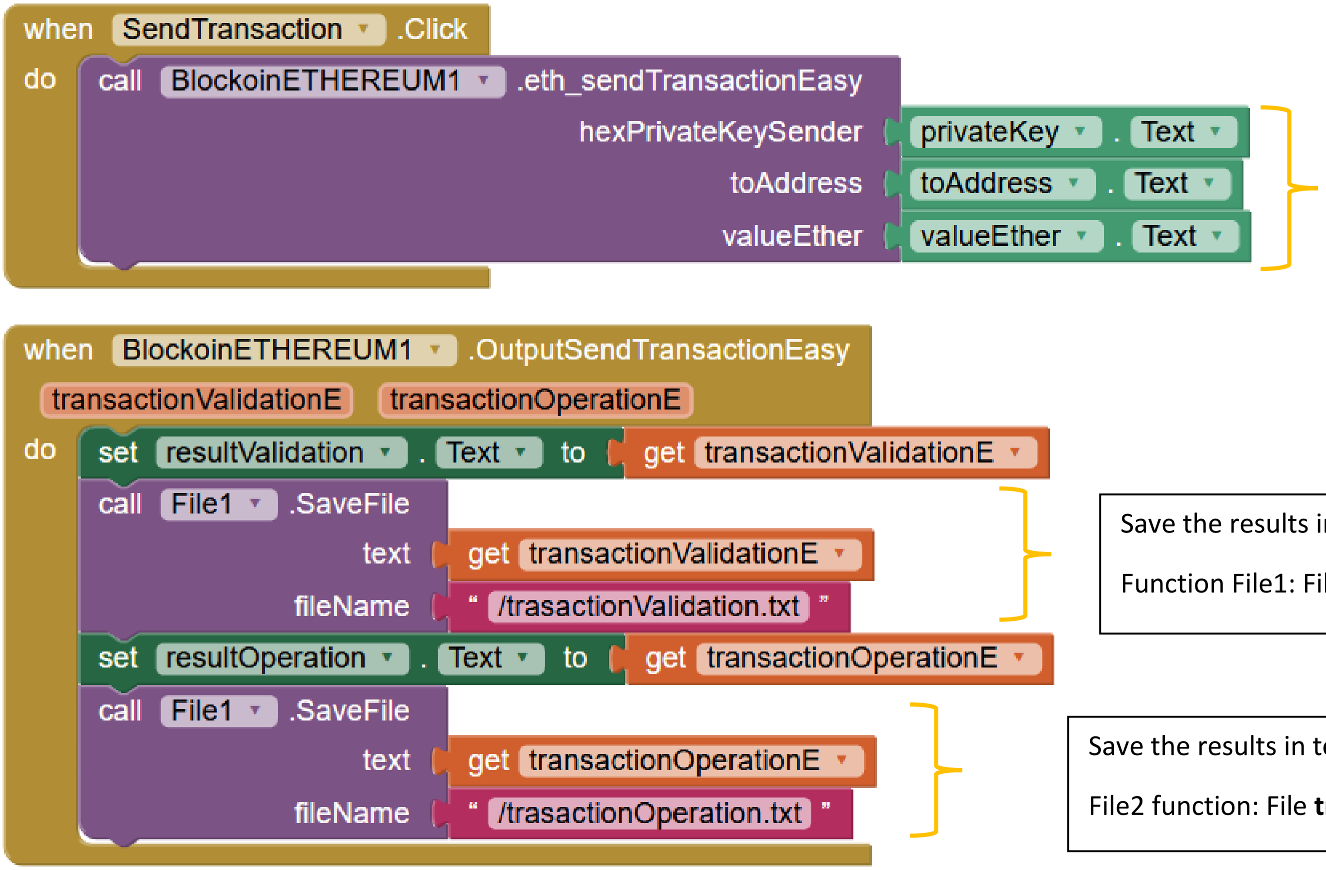
To review the Java Code for CUA extension generation and to implement a consolidated universal address generation algorithm review the Annex "Code for CUA algorithm" or consult the code link: https://github.com/COINsolidation/source

8. Create your Ethereum crypto currency exchange in Android in only 15 minutes.

Design in App Inventor (Screen). - **5 minutes.**



Function (**eth_SendTransactionEasy)** and event (**OutPutSendTransactionEasy**) blocks. - **5 minutes**



```
when SendTransaction .Click
do  call BlockoinETHEREUM1 .eth_sendTransactionEasy
                       hexPrivateKeySender  privateKey . Text
                                 toAddress  toAddress . Text
                                valueEther  valueEther . Text

when BlockoinETHEREUM1 .OutputSendTransactionEasy
  transactionValidationE  transactionOperationE
do  set resultValidation . Text to  get transactionValidationE
    call File1 .SaveFile
                   text  get transactionValidationE
               fileName  " /trasactionValidation.txt "
    set resultOperation . Text to  get transactionOperationE
    call File1 .SaveFile
                   text  get transactionOperationE
               fileName  " /trasactionOperation.txt "
```

Input data:

PrivateKey: Primary key to the sender's address.

toAddress: Hexadecimal address of the receiver.

valueEther: Give the amount of Ether that will be sent.

Save the results in text files:

Function File1: File **trasactionValidation.txt**

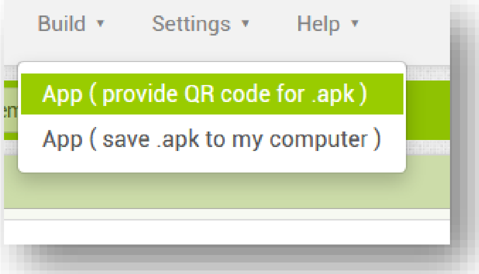
Save the results in text files:

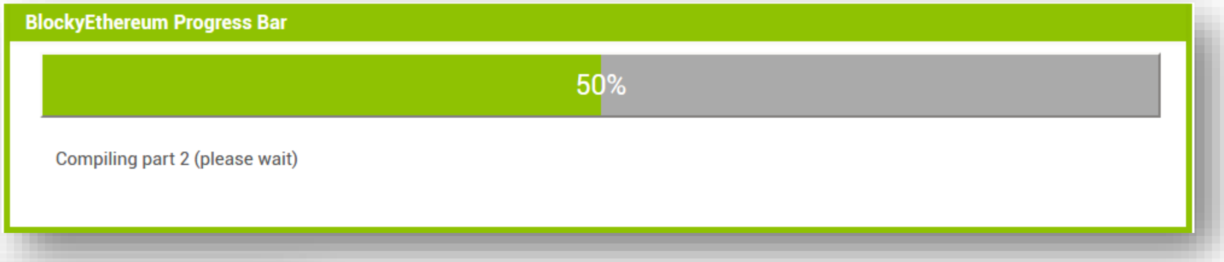
File2 function: File **trasactionValidation.txt**

**More details see User Guide Ethereum Exchange (EEE) Extension in the repository: https://github.com/COINsolidation/userguide

**Repositorio de extensiones COINSolidation: https://github.com/coinsolidation/Extesions-Cryptocurrencies o OpenQbit (Blockchain & Quantum Computing) https://github.com/openqbit-diy

We compile, generate APK file to install it on the Android device. - **5 minutes**

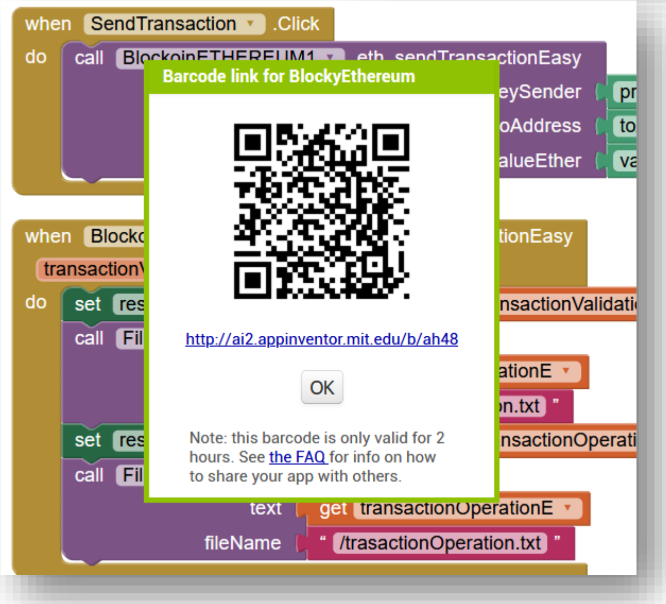

Build ▾
Settings ▾
Help ▾
App (provide QR code for .apk)
App (save .apk to my computer)



BlockyEthereum Progress Bar

50%

Compiling part 2 (please wait)


when SendTransaction .Click
do call BlockoinETHEREUM1 . eth_sendTransactionEasy
eySender
oAddress
alueEther
when Blockc
tionEasy
transaction
do set res
nsactionValidati
call Fil
ationE
n.txt
set res
nsactionOperati
call Fil
text get transactionOperationE
fileName " /trasactionOperation.txt "

Barcode link for BlockyEthereum

http://ai2.appinventor.mit.edu/b/ah48

OK

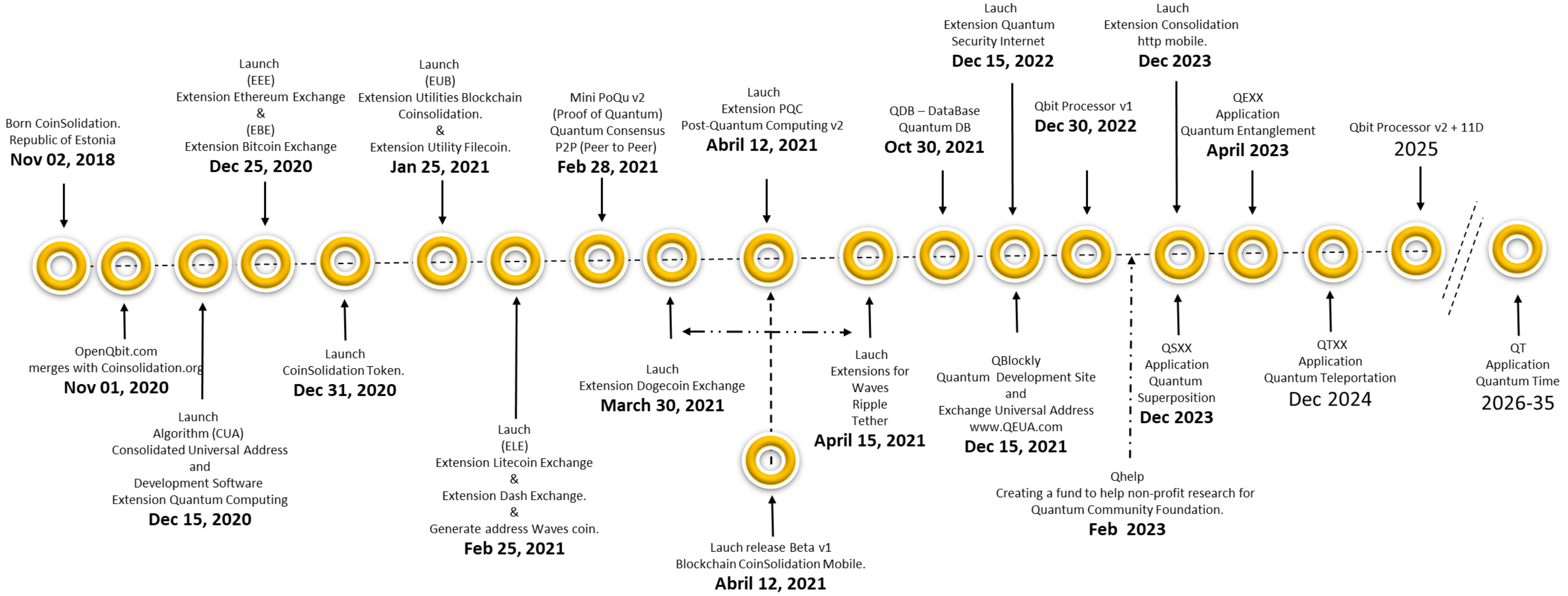
Note: this barcode is only valid for 2 hours. See the FAQ for info on how to share your app with others.

NOTE: When the transaction is executed it will take approximately 6 to 8 seconds to release the "Send Transaction" button. Due to the connection time with the Ethereum network.

9. Roadmap COINsolidation.

ROADMAP

oinSolidation + OpenQbit



Born CoinSolidation.
Republic of Estonia
Nov 02, 2018

Launch
(EEE)
Extension Ethereum Exchange
&
(EBE)
Extension Bitcoin Exchange
Dec 25, 2020

Launch
(EUB)
Extension Utilities Blockchain
Coinsolidation.
&
Extension Utility Filecoin.
Jan 25, 2021

Mini PoQu v2
(Proof of Quantum)
Quantum Consensus
P2P (Peer to Peer)
Feb 28, 2021

Lauch
Extension PQC
Post-Quantum Computing v2
Abril 12, 2021

QDB – DataBase
Quantum DB
Oct 30, 2021

Lauch
Extension Quantum
Security Internet
Dec 15, 2022

Qbit Processor v1
Dec 30, 2022

Lauch
Extension Consolidation
http mobile.
Dec 2023

QEXX
Application
Quantum Entanglement
April 2023

Qbit Processor v2 + 11D
2025

OpenQbit.com
merges with Coinsolidation.org
Nov 01, 2020

Launch
Algorithm (CUA)
Consolidated Universal Address
and
Development Software
Extension Quantum Computing
Dec 15, 2020

Launch
CoinSolidation Token.
Dec 31, 2020

Lauch
(ELE)
Extension Litecoin Exchange
&
Extension Dash Exchange.
&
Generate address Waves coin.
Feb 25, 2021

Lauch
Extension Dogecoin Exchange
March 30, 2021

Lauch release Beta v1
Blockchain CoinSolidation Mobile.
Abril 12, 2021

Lauch
Extensions for
Waves
Ripple
Tether
April 15, 2021

QBlockly
Quantum Development Site
and
Exchange Universal Address
www.QEUA.com
Dec 15, 2021

Qhelp
Creating a fund to help non-profit research for
Quantum Community Foundation.
Feb 2023

QSXX
Application
Quantum
Superposition
Dec 2023

QTXX
Application
Quantum Teleportation
Dec 2024

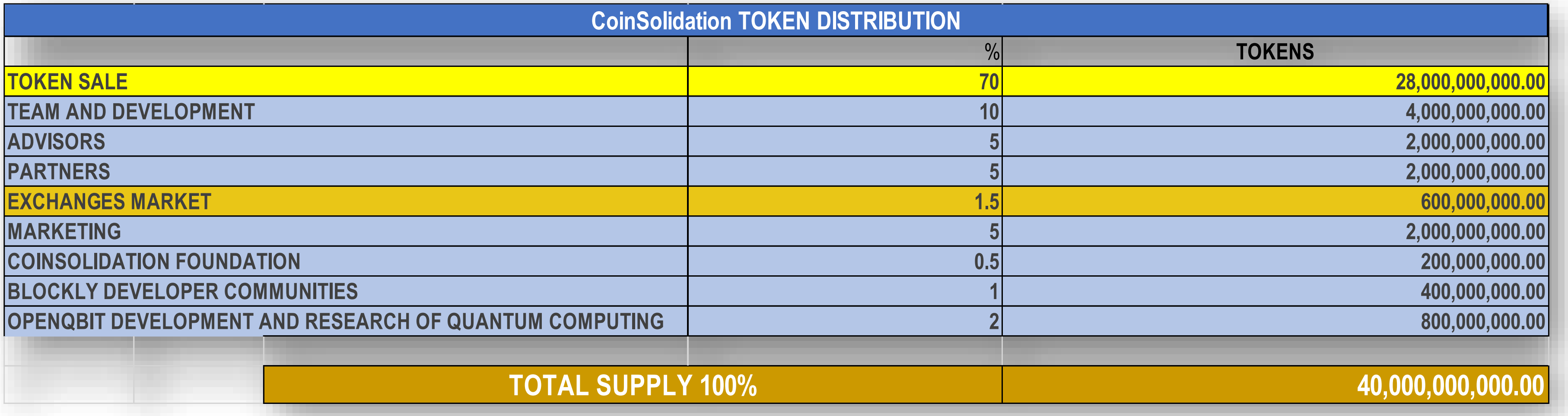
QT
Application
Quantum Time
2026-35

*OpenQbit.com merges with COINsolidation.org (Nov 01, 2020) / OpenQbit specializes in Quantum Computing and Security Quantum.
*Quantum processor version 1 will be using basic quantum logic gates for home use.

10.COINsolidation Token (CUAG) - ICO DISTRIBUTION PLAN.

The ICO is divided into three stages:

The private sale	$ 0.01 USD	(30/Dec 2020 - 30/Jan 2021)	HARD CAPITAL:	$ 280,000,000.00 USD
ICO FIRST PHASE	$ 0.01 USD	(31/Jan 2021 - 28/Feb 2021)	SOFT CAPITAL:	$ 10,000,000 USD
ICO SECOND PHASE	$ 0.15 USD	(1/Mar 2021 - 31/Mar 2021)		



CoinSolidation TOKEN DISTRIBUTION		
	%	TOKENS
TOKEN SALE	70	28,000,000,000.00
TEAM AND DEVELOPMENT	10	4,000,000,000.00
ADVISORS	5	2,000,000,000.00
PARTNERS	5	2,000,000,000.00
EXCHANGES MARKET	1.5	600,000,000.00
MARKETING	5	2,000,000,000.00
COINSOLIDATION FOUNDATION	0.5	200,000,000.00
BLOCKLY DEVELOPER COMMUNITIES	1	400,000,000.00
OPENQBIT DEVELOPMENT AND RESEARCH OF QUANTUM COMPUTING	2	800,000,000.00
TOTAL SUPPLY 100%		40,000,000,000.00

0x9d08c0ac0f2fdf078c883db6fa617b15776e4b41	**COINsolidation TOKEN**
0xbbF57DE98c59B4C304C9d15BC5FAb01304aeCD97	**ICO ADDRESS**
0xa646c054394f85257E18D56Cf5c6b5E603447470	**COINSOLIDATION OPERATION ADDRESS**

11.General characteristics of the COINsolidation token:

Created by: Lugu Samaya.
Name: COINsolidation
Symbol: CUAG - (Consolidated Universal Address Genesis).
Type: NFT
Total tokens created: 40,000,000,000.00
Number of decimals: 18
Launching Country: Estonia
Official Site: www.COINsolidation.org
Company: COINsolidation International.
Launch date: December 30, 2020
Consensus Algorithm: PQu (Proof of Quantum)
Address algorithm: Consolidated Universal Address (CUA).
Security used: PQC (Post-Quantum Cryptography) based on quantum computing.
Technological proposal: Extensions for Blockly systems to use cryptomonads and implementation of quantum security.

Technology partnerships or agreements (merger):

Company: OpenQbit Inc.
Industry: Quantum computing and PQC (Post-Quantum Cryptography).
Official website: www.OpenQbit.com

12. Basic concepts applied in Blockchain platforms.

What is a blockchain?

The blockchain is generally associated with Bitcoin and other crypto-currencies, but these are just the tip of the iceberg since it is not only used for digital money, but can be used for any information that may have a value for users and/or companies. This technology, which has its origins in 1991, when Stuart Haber and W. Scott Stornetta described the first work on a chain of cryptographically secured blocks, was not noticed until 2008, when it became popular with the arrival of bitcoin. But currently its use is being demanded in other commercial applications and is projected to grow in the medium future in several markets, such as financial institutions or the Internet of Things IoT among other sectors.

The blockchain, better known by the term blockchain, is a single record, agreed upon and distributed over several nodes (electronic devices such as PCs, smartpohones, tablets, etc) in a network. In the case of crypto-currencies, we can think of it as the accounting book where each of the transactions is recorded.

Its operation can be complex to understand if we go deeper into the internal details of its implementation, but the basic idea is simple to follow.

It is stored in each block:

1.- a number of valid records or transactions,

2.- information concerning that block,

3 .- its link with the previous block and the next block through the hash of each block –un unique code that would be like the fingerprint of the block.

Therefore, **each block** has a **specific and unmovable place within the chain,** since each block contains information from the hash of the previous block. The entire string is stored on each node in the network that makes up the blockchain, so **an exact copy of the string is stored on all network participants**.

What is an address or account within the blockchain Ethereum platform?

It is a string of 42 characters in the Ethereum platform that represent a number in hexadecimal base, where the assets defined in the Ethereum will be deposited or sent. In other blockchain platforms the number of characters of the account or address can be different, for example:

0x5d2Acdb34c279Aa6d1e94a77F7b18aB938BFb2bB

What is a kryptomoney?

It is a digital or virtual currency designed to function as a medium of exchange. It uses cryptography (digital security) to secure and verify transactions, as well as to control the creation of new units of a particular crypto currency.

What is a token?

Tokens are digital assets that can be used within the ecosystem of a given project.

The main distinction between tokens and crypto-currencies is that the former require another blockchain platform (not their own) to function. Ethereum is the most common platform for creating tokens, mainly due to its intelligent contract function. The tokens created on the Ethereum blockchain are generally known as ERC-20 tokens although there are other more specialized types of tokens such as the ERC-721 token used mainly for collectible assets (cards, use in video games, works of art, etc.).

What is an Exchange?

A crypto-currency exchange is the meeting point where exchanges of crypto-currencies take place in exchange for fiat money or other crypto-currencies. In these online exchange houses the market price is generated which marks the value of the cryptomonies based on supply and demand.

What is Exchange Rates?

These are the rates of the value of an Ether or other crypto-currency in the circulation currencies of each country. For example, at the day of the creation of this manual, an Ether has a value in US dollars of $430.94

What is a transaction?

It is the execution or transfer of some type of non tangible asset that can be given a pre-established value within the Ethereum system and that can later be changed to a tangible value for a company or person.

What is txHash?

It is a hexadecimal number that helps to track the result in detail of each transaction.

What types of transactions are there?

You have two types, one is the transaction "offline" this creates without the need to have connection to the main network of Ethereum can be stored until you choose to connect to the network of Ethereum and release the transaction, have the advantage of security because the entire transaction is processed offline which prevents any anomaly that could be in the network connection. The other transaction is the "online" one which always needs to be connected to the internet with the security advantages and disadvantages that it brings.

What is a Blockchain address?

An address or account is composed of three parts, the address, the public key and the private key, these two keys are a string of numbers and characters in hexadecimal format that are used to send and receive (active) or ether (digital currency).

The primary key should never be shared with anyone as it is the one that authorizes the release of the balance (signs the transactions) held in the account.

The public key is known to the public and is shared with anyone as it is the reference to confirm that the transaction is correct both in terms of value and to whom it is sent.

Examples of Ethereum network management components:

```
{
 "private": "429a043ea6393b358d3542ff2aab9338b9c0ed928e35ec0aed630b93adb14a1c",
 "public": "049b4b7e72701a09d3ee09165bba460f2549494a9d9fd7a95aaac57c2827eac162fd9e105b2461cd6594ca8ca6a8daf10fe982f918be1b0060c87db9cfbcd289a8",
 "address": "88ab6dcecc3603c7042f4334fc06db8e8d7062d5"
}
```

13. What is Blockly programming?

Blockly is a **visual programming methodology** composed by 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 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. It is based on the JavaScript language and was developed by the Google company and MIT.

Blockly is a form of programming where you don't need any background in any kind of computer language, this is because it is only 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 cell 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 to create.

14. Annex "Code for CUA algorithm".

Reference to Github: https://github.com/coinsolidation/source

15. Terms.

Terms and conditions of use see on the site of www.coinsolidation.org or https://github.com/coinsolidation/Terms

Support with commercial use.
support@coinsolidation.org

Commercial use of blockchain sales.
sales@coinsolidation.org

Legal information and licensing questions or concerns.
legal@coinsolidation.org

Social networks:

Twitter: https://twitter.com/ecoinsolidation

Facebook: https://www.facebook.com/coinsolidation