HASHGRAPH POE- DOCUMENTATION

INTRODUCTION

This project intends to use the concept of Proof Of Existence. This POE application can store strings on the hashgraph network, search for a particular string in the same network and return the hash code of the string, search for a particular hash code and retrieve the corresponding idea stored in the network. Also, the application checks for the string in the network before adding it so that there is no duplicacy of ideas.

ABOUT HASHGRAPH

Hashgraph is said to be a more robust system blockchain. Its consensus algorithm provides a new platform for distributed consensus. Some of the attributes commonly used to refer or describe Blockchain are distributed, transparent, consensus-based, transactional and flexible. Hashgraph bears all these features. However, it is a data structure and consensus algorithm is much faster, fairer, and more secure than blockchain. It is described as the future of distributed ledger technology. It uses two special techniques to achieve fast, fair and secure consensus.

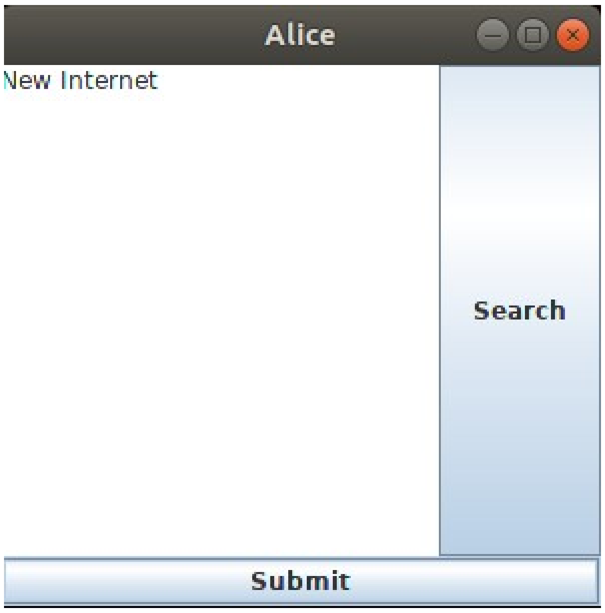
1.Gossip about Gossip
2.Virtual Voting

Gossip about Gossip basically means attaching a small additional amount of information to this Gossip, which are two hashes containing the last two people talked to. Using this information, a Hashgraph can be built and regularly updated when more information is gossiped, on each node.
Once the Hashgraph is ready, it is easy to know what a node would vote, since we are aware of information that each node has and when they knew it. This data can thus be used as an input to the voting algorithm and to find which transactions have reached consensus quickly.

Despite focusing on private, enterprise settings, Hashgraph has enjoyed plenty of public interest. Swirlds released a SDK that allows anybody to experiment with the (closed source) Hashgraph consensus library. Although Hashgraph core is written in Java and LISP, the SDK makes it possible to build applications with any JVM language (Java, Scala, etc). Community members have also used the whitepaper to develop their own implementations in Python, Go, and of course, JavaScript. Each of these libraries follows the whitepaper closely. Their simplicity and similarity hint at what many technical people love about Hashgraph: it is elegant, lends itself to fun visualizations, and its correctness is easy to prove.

FUNCTIONS

1. **Add String-** In the java GUI that pops up when you run the code, you can type in the string and press the “Submit” button. If the idea already exists in the network, it will show a convenient message saying “Your idea already exists”. If not, your idea will be added in the network and the hash of the string will be shown in the GUI window.



2. **Seach String-** If you want to find out if an idea exists in the network, you can just type the particular string in the text area and click on “Search”. If your idea is present, the text area will show the hash of the string, if not, then a message will be displayed saying “Sorry, we could not find your idea”.

3. **Search Hash-** If you want to search for the hash of a string and want to obtain the corresponding string, you may type the hash in the text area and click on “Search”, this will retrieve the string and display it, if unable to find, it will show “Sorry, we could not find your idea”.

HOW TO SETUP

Step 1: Download Swirlds SDK

Download and unzip Swirlds SDK from its website :
https://www.swirlds.com/download/

Step 2: Download and Install JAVA SDK

Download Java SDK according to your PC’s architect from

http://www.oracle.com/technetwork/java/javase/downloads/jdk8-downloads-2133151.html

Step 3 Download JCE

Go to the following link http://www.oracle.com/technetwork/java/javase/downloads/jce8-download-2133166.html to download Java Cryptography Extension.

Replace the local_policy.jar and US_export_policy.jar files in your respective security folder in your jre path.

Step 6 Download and Install Eclipse

Go to the link https://www.eclipse.org/downloads/ to download latest Eclipse IDE tool for developing Java application, after installing,

1. Copy the Swirlds SDK folder in your Eclipse Workspace Directory
2. In eclipse, select “File”, then select “Import”.
3. Select “Maven”, “Existing Maven Projects”, Then “Next”.
4. In “Root Directory”, select the sub-directory under Eclipse workspace directory, where you have copied your SDK file. The window will auto scan existing project under the directory. Select “Finish”, Eclipse will import all Swirlds example projects.

Now you have obtained all the demos created by Swirlds which run on hashgraph.

Step 7 Copy the new code into the java file

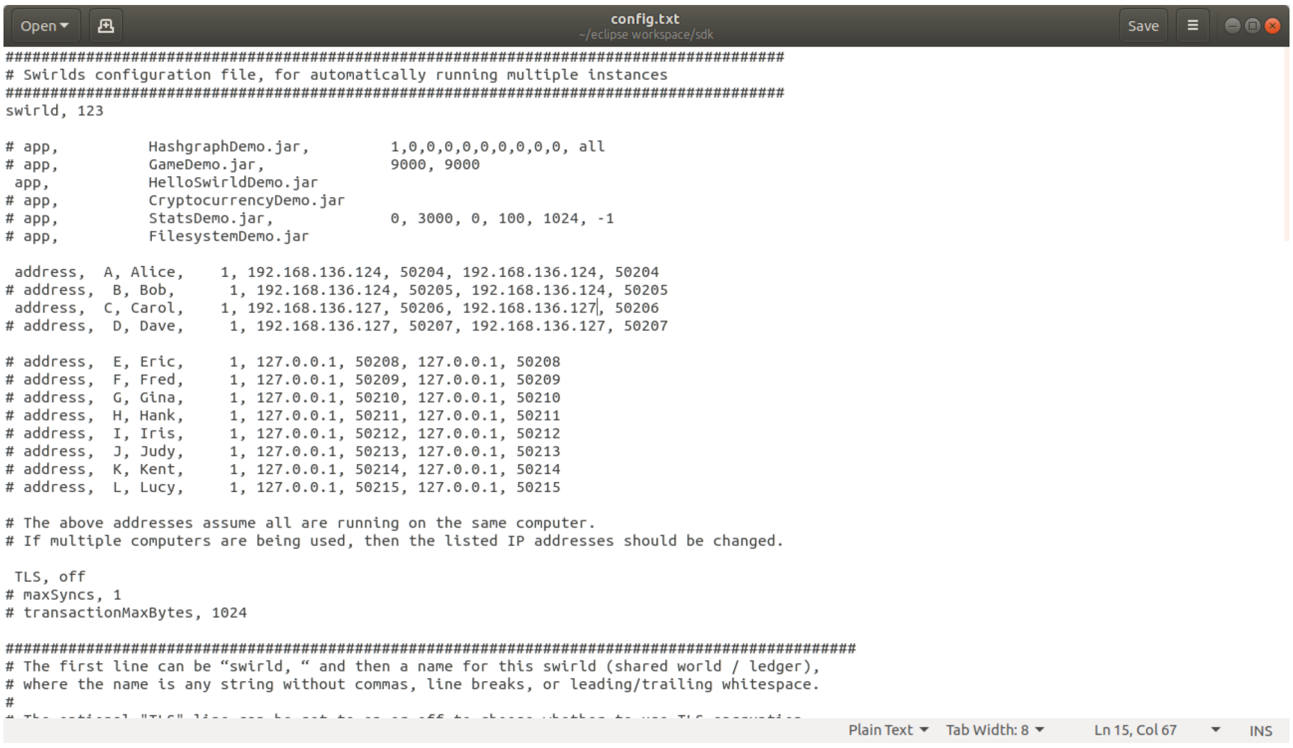
1. In the Project Explorer, dropdown “HelloSwirldDemo”, then dropdown “src”, then dropdown “(default package)”, and open the file “HelloSwirldDemoMain.java”.

2. Replace the already present code with the new code(POE) you downloaded.

Step 8 Make Changes in the config.txt file

1. Open the SDK folder and then open the config.txt file.

2. Uncomment the HelloSwirldDemo.jar line if it is commented.



```
########################################################################################
# Swirlds configuration file, for automatically running multiple instances
########################################################################################
swirld, 123

# app,          HashgraphDemo.jar,         1,0,0,0,0,0,0,0,0,0, all
# app,          GameDemo.jar,              9000, 9000
 app,           HelloSwirldDemo.jar
# app,          CryptocurrencyDemo.jar
# app,          StatsDemo.jar,             0, 3000, 0, 100, 1024, -1
# app,          FilesystemDemo.jar

 address,  A, Alice,    1, 192.168.136.124, 50204, 192.168.136.124, 50204
# address,  B, Bob,      1, 192.168.136.124, 50205, 192.168.136.124, 50205
 address,  C, Carol,    1, 192.168.136.127, 50206, 192.168.136.127, 50206
# address,  D, Dave,     1, 192.168.136.127, 50207, 192.168.136.127, 50207

# address,  E, Eric,     1, 127.0.0.1, 50208, 127.0.0.1, 50208
# address,  F, Fred,     1, 127.0.0.1, 50209, 127.0.0.1, 50209
# address,  G, Gina,     1, 127.0.0.1, 50210, 127.0.0.1, 50210
# address,  H, Hank,     1, 127.0.0.1, 50211, 127.0.0.1, 50211
# address,  I, Iris,     1, 127.0.0.1, 50212, 127.0.0.1, 50212
# address,  J, Judy,     1, 127.0.0.1, 50213, 127.0.0.1, 50213
# address,  K, Kent,     1, 127.0.0.1, 50214, 127.0.0.1, 50214
# address,  L, Lucy,     1, 127.0.0.1, 50215, 127.0.0.1, 50215

# The above addresses assume all are running on the same computer.
# If multiple computers are being used, then the listed IP addresses should be changed.

 TLS, off
# maxSyncs, 1
# transactionMaxBytes, 1024

################################################################################################
# The first line can be “swirld, “ and then a name for this swirld (shared world / ledger),
# where the name is any string without commas, line breaks, or leading/trailing whitespace.
#
```

3. Provide the IP addresses of the users you want to be a part of the hashgraph network. Uncomment the respective lines. As shown in the picture above, two computers are being used as Alice and Carol and their mentioned IP addresses.

Step 9 Compile and run

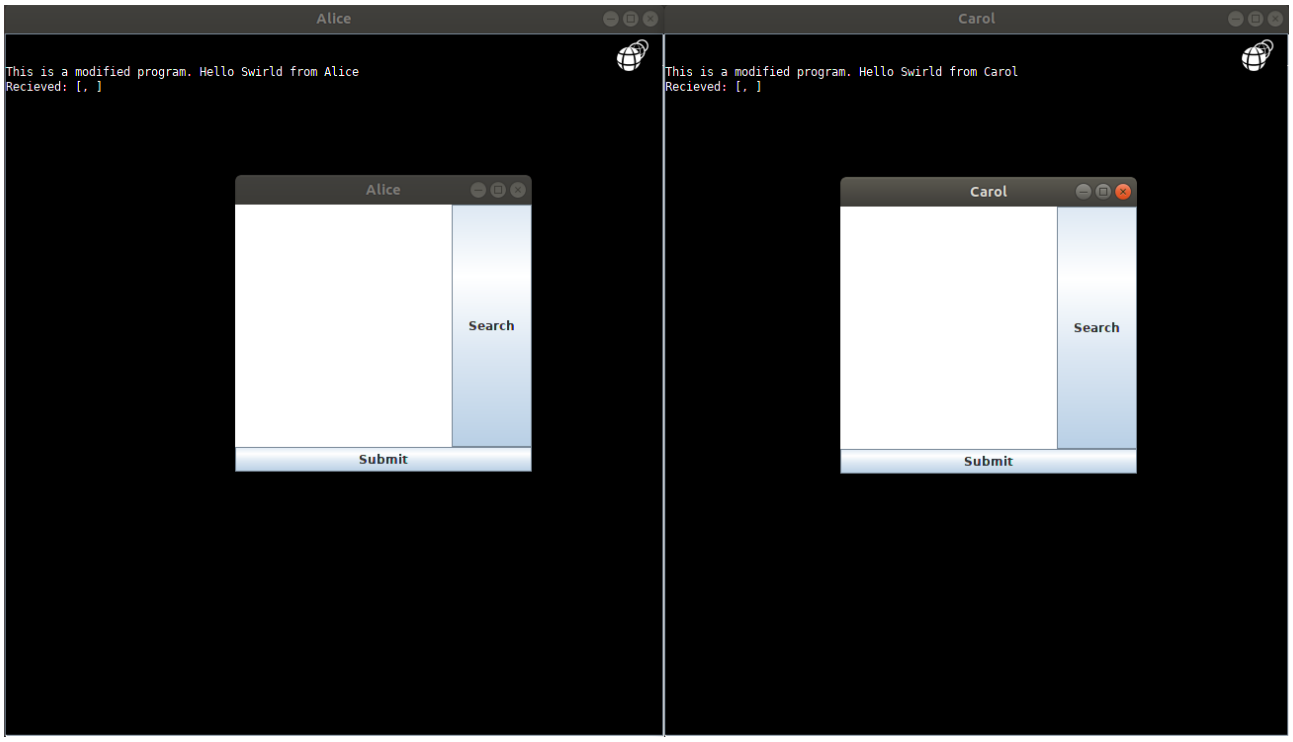
1. Now expand the project HelloSwirldDemo, right click file “pom.xml”, then select “Run As” -> “Mavern Install”. This step would compile the HelloSwirldDemo example.

2. Select the green bug icon or the green array icon, then select “Run Configuration” option.

3. Next, select tab “Arguments”, choose “Others”, then use “File Systems” to select SDK directory under workspace, then select “Apply”, then “Run”.

This should open the corresponding windows of the users. If you are using different computers, then the code should be run simultainously in the machines.

A black screen window appears for each user, which shows the receives string in the network. A simple java GUI window opens on top of the black screen window which allows you to add strings, search for them, etc.



```
This is a modified program. Hello Swirld from Alice
Recieved: [, ]
```

```
This is a modified program. Hello Swirld from Carol
Recieved: [, ]
```

It will look somewhat like in the picture above(same PC).

Now the 'POE using Hashgraph' is completely setup in your local computer(same IP addresses for both Alice and Carol).

To close the Application, you may just close all the windows, this will stop your program.

<u>UNDERSTANDING THE CODE</u>

In layman's terms, the application takes a string as input(in the text area), converts it into bytes and passes it to the function 'createTransaction' which belongs to the interface platform. This string, along with its hash gets added to the hashgraph network. Now, each of the users receive the information and if there is a general consensus(super majority), the string gets approved.

The users also have the ability to search for a string or a hash. The exact string is to be typed because the algorithm uses basic text matching to search for the string in the network. In the same way, the same hash can be typed and searched.