Universidad del Istmo de Guatemala
Facultad de Ingenieria
Ing. en Sistemas
Analisis, diseño y fabricación de Sistemas
Prof. Ernesto Rodriguez - erodriguez@unis.edu.gt

Hoja de trabajo #3/#4

Fecha de entrega: 11 de Mayo, 2018 - 11:59pm
Modalidad: Parejas o individual

Instrucciones: Realizar cada uno de los ejercicios siguiendo sus respectivas instrucciones. El trabajo debe ser entregado a traves de Github, en su repositorio del curso, colocado en una carpeta llamada "Hoja de trabajo 3". Al menos que la pregunta indique diferente, todas las respuestas a preguntas escritas deben presentarse en un documento formato pdf, el cual haya sido generado mediante Latex. Los ejercicios de programación deben ser colocados en una carpeta llamada "Programas", la cual debe colocarse dentro de la carpeta correspondiente a esta hoja de trabajo.

Recursos

- Parity: Un nodo de Ethereum.
- Solidity: Documentación del lenguaje para *smart contracts* Solidity.
- Ethereum Ropsten Faucet: Fuente de Ether (Ropsten) para el testnet. Permite obtener Ethereum (Ropsten) sin tener que minarlo.

Descripción

El objetivo de esta tarea es desarrollar una applicación que utilize la tecnologia de blockchains para funcionar. Dicha aplicación tendra dos componentes: (1) El programa que correra sobre el blockchain *Ethereum* y (2) una aplicación cliente (desktop, mobil, web o consola) que permitira visualizar de forma intuitiva el estado actual de la aplicación almacenado en el blockchain.

Descripción

Describir una aplicación que sera fabricada utilizando la tecnologia de blockchains. En dicha descripción debe incluir:

- Descripción breve de la applicación
- Listado de los casos de uso de la applicación cliente
- Descripción del estado que se almacenara en el blockchain

- Descripción de las funciones que modificaran dicho estado
- Descripción de los eventos que se generaran al modificar el estado.

Aplicación de Blockchain

Implementar mediante *Smart Contracts* la logica de la aplicación utilizando el lenguaje *Solidity*. Se recomienda utilizar el cliente *Parity* para comunicarse con el blockchain de *Ethereum*. Tambien se recomienda conectarse al *test net* (Ropsten) de *Ethereum* ya que en el *test net* no tiene costo correr una aplicación. Para conectarse a Ropsten utilizando *Parity* se puede utilizar el comando:

```
> parity --chain ropsten
```

Media vez inicie *Parity*, se puede interactuar con el mediante su interfaz web acediendo al sitio `http://127.0.0.1:8180/`. Media vez acede a este sitio, puede navegar a la seccion de "Parity Wallet" para crear una billetera de *Ethereum*. Luego de crear la billetera, puede acceder al sitio para abonar su billetera con *Ethereum*.

La sección de *Contracts* de la billetera permite crear *smart contracts* y subirlos al blockchain de *Ethereum*.

Para aprender más sobre *Solidity* y como crear *smart contracts*, puede visitar los sitios:

- `https://www.ethereum.org/token`
- `https://medium.com/@bleev.in.tech/how-to-deploy-a-smart-contract-to-ethereum-testnet-e`

Nota: A *Parity* le tomara algunas horas sincronizarse con el blockchain de *Ethereum*.

Aplicacón Cliente

Támbien se debe implementar una aplicación cliente que permita visualizar de forma intuitiva la información almacenada en el blockchain relacionada a la aplicación de la sección anterior. Puede utilizar la plataforma integrada de *Parity* (`https://wiki.parity.io/Development-Overview`) para desarrollar su aplicación o un lenguaje externo (como Java) y conectarse al blockchain de *Ethereum* utilizando un adaptador como Ethereumj. La mayoria de lenguajes populares de programación tienen adaptadores para conectarse al blockchain de *Ethereum*.