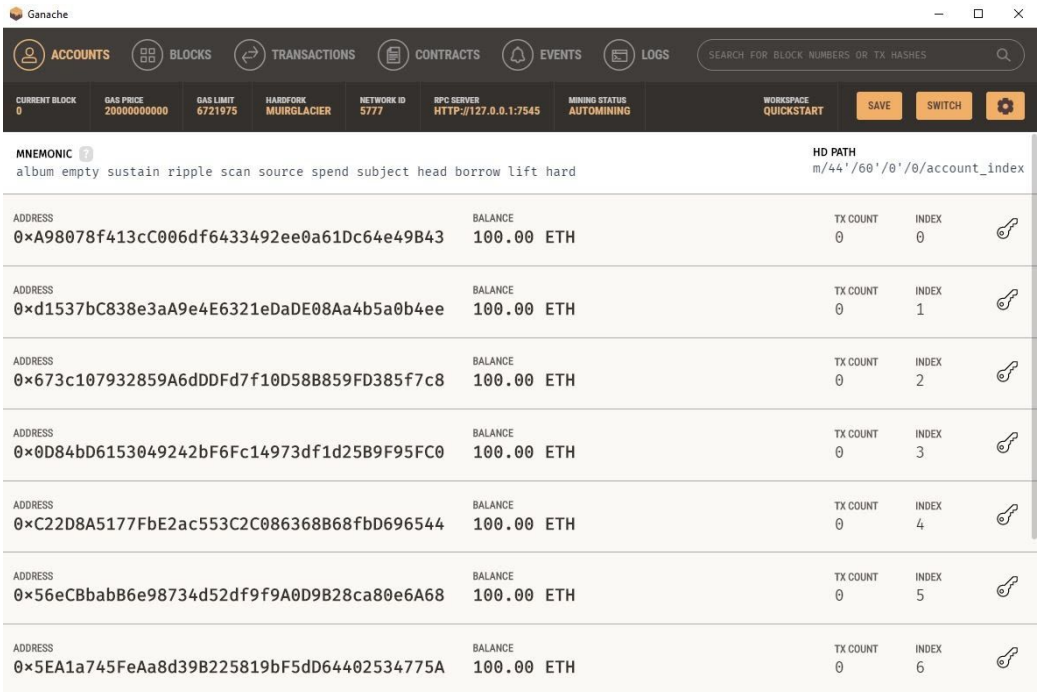
Usability Test Report

This Report Will Focus On the Solidity Remix IDE with A focus on a beginners users experience

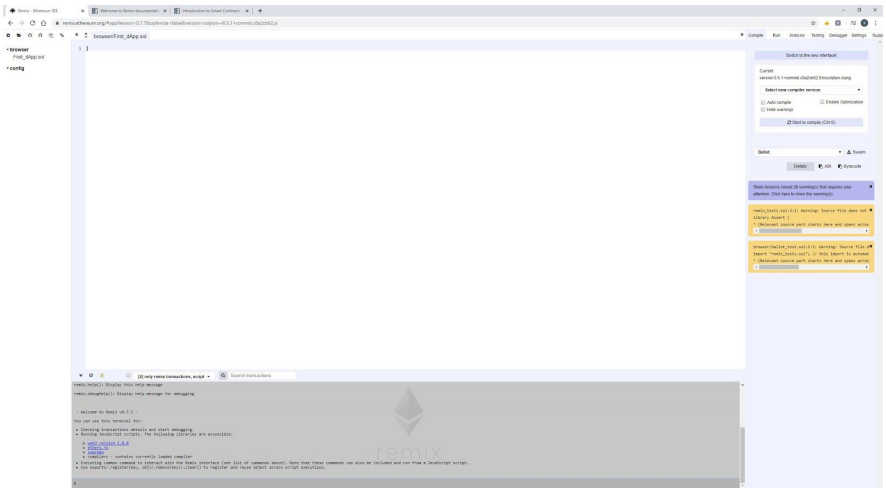
The Cognitive Walkthrough

1. Open up an instance of google Chrome.
2. Navigate to the Remix.Ethereum IDE.
3. Open up a second tab and navigate to Remix IDE Documentation.
4. Open up a third tab and navigate to Solidity Smart Contract Documentation.
5. Because we are builidng a dApp and running it with the Remix IDE rather then using a local machine framework, we will need a few other frameworks to get our application up and running.
    a. The first Item you will need to download is Ganache. (A local block chain development tool). Once the ganache tool is downloaded you should have an account page like below.



    b. The second item you will need is a blockchain web browser. We will use MetaMask for this (Download MetaMask) documentation for MetaMask can be found Here (tutorial to set up MetaMask can be found Here).

6. Using the Remix IDE hit the plus button in the top right hand corner to create a new file called First_dApp.sol (You should now have something similar to the image below.)



7. dApp type: the type of dApp that this walk through will focus on is Election application. Please see [Voting Guide Solidity.]

Example Contract code enter into the Remix IDE

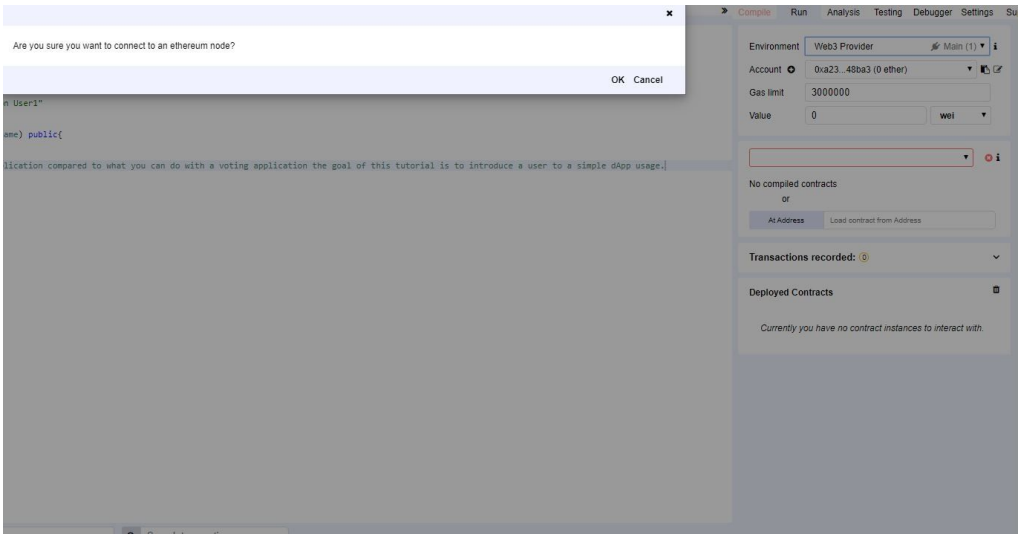
```
pragma solidity ^0.4.2;

contract VotingSystem {
    string public User_Canidate_Name;

    function Election () public {
        User_Canidate_Name = "Mike";
    }

    function Candidate (string _name) public {
        User_Canidate_Name = _name;
    }
}
```

NOTE: MAKE SURE YOU SET THE COMPILER TO VERSION 0.4.26

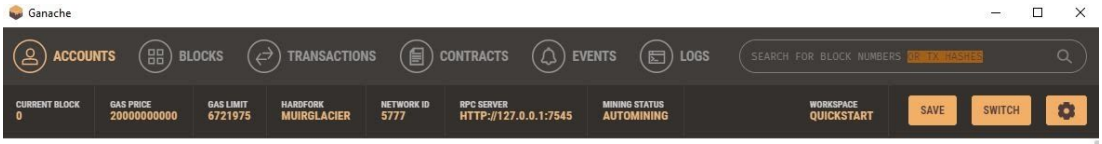
//This is an extremely simple application compared to what you can do with a voting application the goal of this tutorial is to introduce a user to simple dApp usage.

Make sure to compile the code!

8. Using the remix IDE navigate to the run tab and change the environment to web3 Provider (See image below). Hit ok!

a. 

9. Next you will be asked to enter in the local host address the address can be found in Ganache. (The address is the last 4 digits of the RPC server field).

a. 

10.

11. Now you should be able to deploy your smart contract. Under Run select your contract via the name you gave it in your sol file. Once you have selected the correct contract name hit the deploy button. After this is completed you should see that the contract has deployed under “Deployed Contracts”.
    a. Note: In order to complete this soft dApp you must now build a client side application to communicate with the deployed blockChain we have created.
12. For this you will need web3, html pages, css and javascript.
13. Key steps are initialization of the web3 library itself. Web3 will communicate with MetaMask.
14. Pass web3 the Contract Abi at the specific contract address.

    Example code:

    myContractAbi = [Json object that can be found in remix ide under compile and in details copy the abi and paste it here.]

    MyContractAddress = (Copy contract address under the deployed tab)

    Var Election = web3.eth.contract (myContractAbi).at(MyContractAddress);

15. Program the rest of your desired application however you would like it to work. We now have complete communication between the remix IDE contract and our deployed smart contract. You can display the candidate name with javascript calls such as:

```
Election.User_Canidate_Name(function(err,User_Canidate_Name){
        $('#User_Canidate_Name').html(User_Canidate_Name);
});
```

Evaluation On Usability

After conducting my cognitive walk through I will now evaluate the usability of building a smart contract and deploying a dApp while using the remix IDE and a users local machine.This evaluation will be done from a beginners perspective on how to get all of the required moving parts up and running. The metrics I have decided to use for my usability testing consist of Effectiveness, Efficiency, Satisfaction, Time spent on Task, Number of Errors, and Mental effort.

Scale:(0 = extremely low usability all the way to 5 = extremely usable)

Evaluating the Deployment of our Smart Contract

This part of the evaluation will focus on tasks (1-11) of our cognitive walkthrough. Tasks 1-4 are relatively straightforward tasks and require a low cost in terms of mental effort. These tasks can be done with perfect scores on the Usability metrics I proved above. Task 5 is where it gets a bit more complicated. I found solidity documentation on how to use the remix IDE a bit confusing. This is mainly due to the lack of information it feels like Solidity provides to users while setting up the environment field of the IDE (Environment set up). When a user has to go through the process of hooking up MetaMask, Ganache in order to use the web3 provider option(Needed to create and deploy an dApp with remix) it becomes a bit unclear how to go about this. I personally needed to find resources outside of the solidity documentation inorder to complete these tasks. In terms of rating this task I would say Effectiveness = 3, Efficiency = 3, Satisfaction = 5, Time spent on Task = 10-15min, Number of Errors = 0, and Mental effort = 4. Tasks 6 and 7 can easily be completed with perfect scores. I had no problems when building the smart contract in Remix. Solidity provides amazing documentation on how to build smart contracts and set them up. Task 8 and 9 I found to be very easy as well but in terms of rating there are some errors that can occur here if you didn't set up MetaMask, and Ganache properly. So for a rating I would say Effectiveness = 4, Efficiency = 4, Satisfaction =5, Time spent on Task= minimal if set up was correct if not you must do some backtracking, Number of Errors = 2, and Mental effort = depends. Regarding tasks 9 - 11 these tasks are very straight forward and are explained in the

REMIX IDE documentation that solidity provides I would say if you make it past tasks 6 and 7 these steps should have perfect usability scores.

Final thoughts: Connecting a smart contract up with applications outside of the REMIX IDE can be rather difficult but is rewarding. I dont think its the easiest task for a beginner but anyone who wants to become a blockchain developer should learn about the process. I wish solidity provided a bit more documentation about tools such as ganache or MetaMask making the beginner user experience a bit more digestible.

Building The dApp

This part of the evaluation will focus on tasks (12-15) of our cognitive walkthrough. Evaluation of this section will be rather difficult because it's not really explained in Solidities documentation. In the REMIX IDE documentation they do provide information to users about things such as Abi and storage of the contract addresses. But in terms of building the dApp and how to go about hooking these things up I had to look outside of the provided documentation. Rating this section will be difficult because Solidities documentation does provide a robust amount of information to users. The only problem is this information is difficult to work with if you are a beginner. Scoring wise I would give this section of the walkthrough scores of: Effectiveness =3 , Efficiency = 3, Satisfaction = 3, Time spent on Task = 60+min, Number of Errors = 20+, and Mental effort = a lot depending on your skill level.

Final thoughts: I wish solidity had a streamlined guide on how to attach a smart contract to a web app. With soldities current documentation on how to go about this process it is definitely not user friendly to entry level users.