## Building my first dApp

**This is the steps on how to create a front end, deploy a Solidity smart contract, and connect them together.**

**App that simply reads and writes a value to the blockchain**

**It about a 3 stages**

* Create a basic HTML web page
* Create a basic Solidity smart contract
* Connect the web page with the smart contracts using Ethers.js.

**Tools used**

* Metamask to access your wallet and connect it with the app
* Remix IDE to write,compile and deploy the smart contracts using solidity
* Ether.js for interacting with the Ethereum Blockchain and its ecosystem usign js

**Process**

1. Downloading and install Metamask.
2. Request some Goerli Tesnet Ether from a faucet loaded into your Metamask Wallet.
3. Create your webpage (html/css/js).

<body>

<div>

<h1>This is my dApp!</h1>

<p>Here we can set or get the mood:</p>

<label for="mood">Input Mood:</label> <br />

<input type="text" id="mood" />

<button onclick="getMood()">Get Mood</button>

<button onclick="setMood()">Set Mood</button>

</div>

</body>

1. Creating the smart contract and deploye it to the blockchain using solidity language and Remix IDE

//specify the version of solidity

// SPDX-License-Identifier: MIT

pragma solidity ^0.8.7;

/// a simple set and get function for mood defined:

//define the contract

contract MoodDiary{

//create a variable called mood

string mood;

//create a function that writes a mood to the smart contract

function setMood(string memory \_mood) public{

mood = \_mood;

}

//create a function the reads the mood from the smart contract

function getMood() public view returns(string memory){

return mood;

}

}

Deploy the contract on the Goerli Testnet.

* Make sure your Metamask is connected to the Goerli Testnet.
* Make sure you select the right compiler version to match the solidity contract. (In the compile tab)
* Compile the code using the "Solidity Compiler" tab. Note that it may take a moment to load the compiler
* Deploy the contract under the "Deploy and Run Transactions" tab
* Under the Deployed Contracts section, you can test out your functions on the Remix Run tab to make sure your contract works as expected!

1. Connect Your Webpage to Your Smart Contract using ether.js

<script

src="https://cdn.ethers.io/lib/ethers-5.2.umd.min.js"

type="application/javascript"

></script>

<script>

const MoodContractAddress="YOUR CONTRACT ADDRESS" // you can copie it from the feploy and run transaction tab in remix IDE in the deployed contracts section

const MoodContractABI = [] // JSON OF THE CONTRACT ABI , you can get it from the solidity compiler at the buttom

let MoodContract;

let signer;

//Next, Define an ethers provider. In our case it is Goerli:

const provider = new ethers.providers.Web3Provider(

window.ethereum,

"goerli"

);

//Request access to the user's wallet and connect the signer to your metamask account

//(we use [0] as the default), and define the contract object using your contract address, ABI, and signer

provider.send("eth\_requestAccounts", []).then(() => {

provider.listAccounts().then((accounts) => {

signer = provider.getSigner(accounts[0]);

MoodContract = new ethers.Contract(

MoodContractAddress,

MoodContractABI,

signer

);

});

});

//Create asynchronous functions to call your smart contract functions

async function getMood() {

const getMoodPromise = MoodContract.getMood();

const Mood = await getMoodPromise;

console.log(Mood);

}

async function setMood() {

const mood = document.getElementById("mood").value;

const setMoodPromise = MoodContract.setMood(mood);

await setMoodPromise;

}

</script>

1. Test Your Work Out!

* Got your web server up? Go to <http://127.0.0.1:3000/> in your browser to see your page!
* Test your functions and approve the transactions as needed through Metamask. Note block times are ~15 seconds... so wait a bit to read the state of the blockchain
* See your contract and transaction info via <https://goerli.etherscan.io/>
* Open a console (Ctrl + Shift + i) in the browser to see the magic happen as you press those buttons