

BALLOT SCREENSHOTS

The image displays two screenshots of the Remix IDE interface, showing the deployment and execution of a Solidity contract named `Ballot`.

Top Screenshot:

- Left Panel (DEPLOY & RUN TRANSACTIONS):** Shows the deployment of the `Ballot` contract. The `delegate` function is selected, and the `address to` field is set to `0xAb8483F64d9C6d1EcF...`. The `vote` function is also shown with a value of `1`. The `proposals` field is set to `1`. The `voters` field is set to `1`. The `winningPropo...` field is empty.
- Right Panel (Contract Code):** Shows the Solidity code for the `Ballot` contract. The code includes a `contract Ballot` definition with a `struct Voter` and a `struct Proposal`. The `Ballot` contract has a `vote` function that increments the `voteCount` for a given proposal.
- Bottom Panel (Transaction Log):** Shows the transaction log. The first transaction is a successful deployment of the `Ballot` contract. The second transaction is a pending `Ballot.vote` transaction.

Bottom Screenshot:

- Left Panel (DEPLOY & RUN TRANSACTIONS):** Shows the deployment of the `Ballot` contract. The `delegate` function is selected, and the `address to` field is set to `0xAb8483F64d9C6d1EcF...`. The `vote` function is also shown with a value of `1`. The `proposals` field is set to `1`. The `voters` field is set to `1`. The `winningPropo...` field is empty.
- Right Panel (Contract Code):** Shows the Solidity code for the `Ballot` contract. The code includes a `contract Ballot` definition with a `struct Voter` and a `struct Proposal`. The `Ballot` contract has a `vote` function that increments the `voteCount` for a given proposal.
- Bottom Panel (Transaction Log):** Shows the transaction log. The first transaction is a successful deployment of the `Ballot` contract. The second transaction is a failed `Ballot.vote` transaction. The error message is: "The transaction has been reverted to the initial state. Reason provided by the contract: '5 Minute Time Limit is Over'." The transaction hash is `0x609...e62e6`.

HELLO WORLD SCREENSHOT

The screenshot displays the Remix IDE interface, which is used for developing, deploying, and interacting with smart contracts. The interface is divided into several panels:

- Left Panel (Deploy & Run Transactions):** This panel contains controls for deploying and running transactions. It includes a **Deploy** button, a checkbox for **Publish to IPFS**, and a section for **Deployed Contracts**. Under this section, a contract named **HELLOWORLD AT 0x652...BA595 (MI)** is listed. Below the contract name, there are two buttons: **storeNumber** (with a value of 15) and **retrieveNumber**. At the bottom of this panel, there is a **Low level interactions** section with a **CALLDATA** input field and a **Transact** button.
- Top Panel (Code Editor):** This panel shows the Solidity code for the **zkpHelloWorld.sol** contract. The code is as follows:

```
1 // SPDX-License-Identifier: MIT
2
3 pragma solidity 0.8.13;
4
5
6 contract HelloWorld{
7
8     //establish variable with type and name
9     int input;
10
11     //function to store the numbers, with _input to signify the input in the function
12     function storeNumber(int _input) public {
13         input = _input;
14     }
15
16     //function to retrieve the number from the newly altered variable
```
- Bottom Panel (Transaction Log):** This panel shows the execution history of the contract. It includes a search bar and a list of transactions. The first transaction is a **transact to HelloWorld.storeNumber pending ...**. The second transaction is a **[vm] from: 0xAb8...35cb2 to: HelloWorld.storeNumber(int256) 0x652...ba595 value: 0 wei data: 0x813...0000f** with a **logs: 0 hash: 0x50d...6a776** and a **call to HelloWorld.retrieveNumber**. The third transaction is a **[call] from: 0xAb8483F64d9C6d1EcF9b849Ae677dD3315835cb2 to: HelloWorld.retrieveNumber()** with a **data: 0xa00...9491b**. Each transaction has a **Debug** button next to it.