Creating a Local Blockchain

-Tanishka Yagneshwar, 2025

taneez/local-blockchain-setup

We'll use Hardhat, a standard and powerful Ethereum development environment with a local blockchain network that is perfect for this task. We'll write the contracts in Solidity and use Ethers.js (included with Hardhat) to interact with them concurrently from a script.

Project Goal:

1. Create a simple Counter and Transfer smart contract in Solidity.
2. Deploy this contract to a local Hardhat Network instance.
3. Write a Node.js script using Ethers.js to send multiple increment() transactions concurrently for Counter and multiple deposit() transactions concurrently for Transfer.
4. Verify the final count.

Phase 1: Setting up the Hardhat Project

1. **Prerequisites:**
   - Node.js (v16 or later recommended) and npm installed. Check with node -v and npm -v.
   - A terminal or command prompt.
2. **Create Project Directory:**
   mkdir concurrent-execution-transactions

   cd concurrent-execution-transactions

3. **Initialize npm:**
   npm init -y
4. **Install Hardhat:**
   npm install --save-dev hardhat
5. **Initialize Hardhat Project:**
   npx hardhat
6. Select "Create a JavaScript project" (or TypeScript if you prefer, but this guide uses JS).
   - Accept the default project root.
   - Say yes (y) to adding a .gitignore.
   - Say yes (y) to installing the sample project's dependencies (@nomicfoundation/hardhat-toolbox). This toolbox includes Ethers.js, Chai, Mocha, and other essentials.

Phase 2: Creating the Smart Contract

Delete Sample Contract: Remove the default Lock.sol file inside the contracts/ directory.

```
    # On Linux/macOS
rm contracts/Lock.sol
# On Windows
del contracts\Lock.sol
```

1. **Create Counter.sol & Transfer.sol:** Create new files named Counter.sol and Transfer.sol inside the contracts/ directory.
2. **Add the Counter and Transfer Contract codes from GitHub**
3. **Compile the Contracts:** Run the compile task to make sure everything is okay and to generate artifacts.
   npx hardhat compile

Phase 3: Writing the Interaction Script

Delete Sample Script: Remove the default deploy.js (or deploy.ts) file inside the scripts/ directory or create the scripts directory if not already present.

```
    # On Linux/macOS
rm scripts/deploy.js
# On Windows
del scripts\deploy.js
```

Copy the required files inside scripts from the github code.

Phase 4: Running the Script

Execute the Script: Run the script using the Hardhat runner, targeting the local Hardhat Network.

```
    npx hardhat run scripts/concurrentIncrement.js --network hardhat
```

Expected Output:

You will see output similar to this (addresses and hashes will differ):

```
Deploying Counter contract...
Counter deployed to: 0x5FbDB2315678afecb367f032d93F642f64180aa3

Initial count: 0
```

Preparing to send 5 increment transactions concurrently...
- Preparing tx from signer 0 (0xf39F...)
- Preparing tx from signer 1 (0x7099...)
- Preparing tx from signer 2 (0x3C44...)
- Preparing tx from signer 3 (0x90F7...)
- Preparing tx from signer 4 (0x15d3...)

Sending 5 transactions...
All transaction requests sent to the node.

Waiting for all transactions to be mined...
- Waiting for tx: 0x.....
- Waiting for tx: 0x.....
- Waiting for tx: 0x.....
- Waiting for tx: 0x.....
- Waiting for tx: 0x.....

All transactions have been mined!

Final count: 5

✅ Success! The final count matches the expected count.

Setting up the localhost address

Modify the hardhat.config.js file to the following to setup the localhost server

```
// hardhat.config.js
require("@nomicfoundation/hardhat-toolbox");

module.exports = {
  solidity: "0.8.20", // Or your version
  networks: {
    hardhat: {
      // No URL needed here if using the implicit in-memory node for
single-threaded tasks
    },
    localhost: {
      url: "http://127.0.0.1:8545",
```

```
      // accounts are automatically picked up from the 'npx hardhat node'
instance
    }
  }
};
```

Start the Local Hardhat Node (Terminal 1)
Open a new terminal window or tab. In this terminal, navigate to your project directory and start a standalone Hardhat node:

```
Unset
cd path/to/your/concurrent-execution-transactions # Navigate to
your project root
npx hardhat node
```

You should see output similar to this, indicating the node is running and listening:

```
Unset
Started HTTP and WebSocket JSON-RPC server at
http://127.0.0.1:8545/

Accounts
========
... (List of 20 accounts with addresses and private keys) ...
```

.
Keep this terminal window open. The node needs to be running while you execute your script.

Run Your Program Code (Terminal 2)
Open a second terminal window or tab. Navigate to your project directory again. Now, execute your benchmark script, specifically targeting the localhost network you defined:

```
Unset
cd path/to/your/concurrent-execution-transactions # Navigate to
your project root
npx hardhat run scripts/runWorkerBenchmark.js --network localhost
```