import { Wallet, utils } from "zksync-web3";

import \* as ethers from "ethers";

import { HardhatRuntimeEnvironment } from "hardhat/types";

import { Deployer } from "@matterlabs/hardhat-zksync-deploy";

// An example of a deploy script that will deploy and call a simple contract.

export default async function (hre: HardhatRuntimeEnvironment) {

console.log(`Running deploy script for the Greeter contract`);

// Initialize the wallet.

const wallet = new Wallet("<WALLET-PRIVATE-KEY>");

// Create deployer object and load the artifact of the contract you want to deploy.

const deployer = new Deployer(hre, wallet);

const artifact = await deployer.loadArtifact("Greeter");

// Estimate contract deployment fee

const greeting = "Hi there!";

const deploymentFee = await deployer.estimateDeployFee(artifact, [greeting]);

// OPTIONAL: Deposit funds to L2

// Comment this block if you already have funds on zkSync.

const depositHandle = await deployer.zkWallet.deposit({

to: deployer.zkWallet.address,

token: utils.ETH\_ADDRESS,

amount: deploymentFee.mul(2),

});

// Wait until the deposit is processed on zkSync

await depositHandle.wait();

// Deploy this contract. The returned object will be of a `Contract` type, similarly to ones in `ethers`.

// `greeting` is an argument for contract constructor.

const parsedFee = ethers.utils.formatEther(deploymentFee.toString());

console.log(`The deployment is estimated to cost ${parsedFee} ETH`);

const greeterContract = await deployer.deploy(artifact, [greeting]);

//obtain the Constructor Arguments

console.log("constructor args:" + greeterContract.interface.encodeDeploy([greeting]));

// Show the contract info.

const contractAddress = greeterContract.address;

console.log(`${artifact.contractName} was deployed to ${contractAddress}`);

// Verify contract programmatically

//

// Contract MUST be fully qualified name (e.g. path/sourceName:contractName)

const contractFullyQualifedName = "contracts/Greeter.sol:Greeter";

const verificationId = await hre.run("verify:verify", {

address: contractAddress,

contract: contractFullyQualifedName,

constructorArguments: [greeting],

bytecode: artifact.bytecode,

});

console.log(`${contractFullyQualifedName} verified! VerificationId: ${verificationId}`)

}