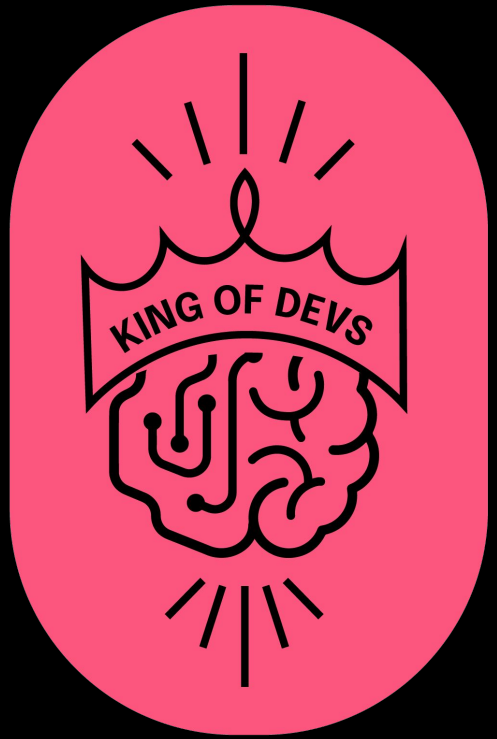
KING OF DEVS

KING OF



WORKSHOP DAY 2

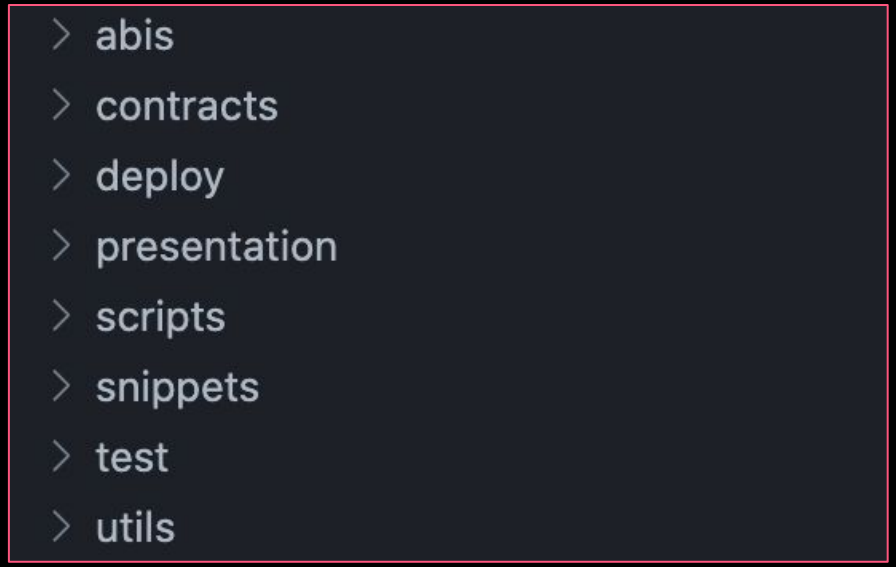
Deployment with Hardhat deploy

Hardhat deploy

- hardhat plugin
    - Deploy secuencial de contratos
        - namedAccounts
            - Deploy con dependencias

Hardhat deploy

1. Crear carpeta deploy



Hardhat deploy

2. Nombra tus cuentas en el archivo hardhat.config.js

```
namedAccounts: {
    deployer: 0,
    admin: 1,
    minter: 2,
    user: 3,
},
```

Hardhat deploy

3. Deploy de tu primer contrato

```
const func: DeployFunction = async function (hre: HardhatRuntimeEnvironment) {
  const {deployments, getNamedAccounts} = hre;
  const {deploy} = deployments;

  const {deployer} = await getNamedAccounts();

  printInfo(`\n Deploying ${ContractName} contract...`);
  const unlockTime = getUnlockTime();

  const LockV8Result = await deploy(ContractName, {
    args: [unlockTime],
    contract: ContractName,
    from: deployer,
    skipIfAlreadyDeployed: false,
    value: ethers.utils.parseEther('1'),
    autoMine: true,
  });

  const lockV8Address = LockV8Result.address;

  printDeploySuccessful(ContractName, lockV8Address);

  return true;
};
```

Hardhat deploy

4. Deploy del segundo contrato dependiente del primero

```
const version = 'v0.0.0';
const ContractName = 'StoppablePattern';

const func: DeployFunction = async function (hre: HardhatRuntimeEnvironment) {
  const {deployments, getNamedAccounts} = hre;
  const {deploy} = deployments;

  const {deployer} = await getNamedAccounts();

  printInfo(`\n Deploying ${ContractName} contract...`);

  const LockV8 = await ethers.getContract("LockV8");

  const StoppableResult = await deploy(ContractName, {
    args: [deployer,LockV8.address],
    contract: ContractName,
    from: deployer,
    skipIfAlreadyDeployed: false,
    autoMine: true,
  });

  const stoppableAddress = StoppableResult.address;

  printDeploySuccessful(ContractName, stoppableAddress);

  return true;
};

export default func;
const id = ContractName + version;
func.tags = [id, version];
func.id = id;
func.dependencies = ["LockV8"];
```

GRACIAS