# 423.https://stackoverflow.com/questions/71129884/hardhat-issue-unpredictable-gas-limit-in-a-script-within-ethereum-development-of

**T:**Hardhat issue UNPREDICTABLE\_GAS\_LIMIT in a script within ethereum development of ERC-721

**Q:**I have a problem that I really despair of and I think it has maybe to do with an "wait for confirmation?" issue, but I don't know how to activate this.  
  
Issue is, that I get as result of an request against a function of my ERC-721 contract following message in a deployment script:  
  
reason: 'cannot estimate gas; transaction may fail or may require manual gas limit',code: 'UNPREDICTABLE\_GAS\_LIMIT',  
  
I think it maybe has to do with that the commands are happening to fast after each other because, when I proceed the following steps in a hardhad console manually (npx hardhat console --network rinkeby), everthing works without this error (even with that <>.confirmations are always 0, what I don't understand)  
  
I have the following commands in a deploy.jsI run with:  
  
npx hardhat run --network rinkeby scripts/deploy.js  
  
async function main() { const BBA = await hre.ethers.getContractFactory("mycontract"); const bba = await BBA.deploy(); await bba.deployed(); console.log("mycontract deployed to:", bba.address); await bba.safeMint("0x123...", 0, 'https://gateway.pinata.cloud/ipfs/url'); await bba.lockToken(0, "1644598343"); console.log(await bba.getLock(0)); ...}  
  
WARN: THIS PARAGRAPH CONTAINS TAG: [CODE]   
  
With the call await bba.getLock(0) I get theError: cannot estimate gas; transaction may fail or may require manual gas limit (error={"name":"ProviderError","code":-32000,"\_isProviderError":true}, method="call", transaction= ...  
  
I build a 1:1 ERC-721 based on openzeppelin contract wizardhttps://docs.openzeppelin.com/contracts/4.x/wizard.  
  
In this contract I added a simple function:  
  
 mapping(uint256 => uint256) private \_locks; function lockToken( uint256 targetTokenId, uint256 since, ) public { \_locks[targetTokenId] = since; } function getLock(uint256 targetTokenId) public view returns (uint256) { return \_locks[targetTokenId]; }  
  
WARN: THIS PARAGRAPH CONTAINS TAG: [CODE]   
  
I use INFURA with following hardhat.config.js:  
  
const { projectId, mnemonic } = require('./secrets.json');module.exports = { rinkeby: { url: "https://rinkeby.infura.io/v3/" + projectId, accounts: { mnemonic: mnemonic }, confirmations: 2, gas: 2100000, gasPrice: 8000000000, saveDeployments: true } }, solidity: { version: "0.8.4", settings: { evmVersion: "byzantium", optimizer: { enabled: true, runs: 1500, } } }};  
  
WARN: THIS PARAGRAPH CONTAINS TAG: [CODE]   
  
Anyone could guide me, to the reason of that problem?  
  
Thanks in advance.

1 **Answer**

**A1:**after additional research and testing the solution seems very simple.  
  
I just needed to declare a const for the function calls and then needed to call a wait() call on these.  
  
As I read, then the procedure waits for the call to be mined.  
  
I'm still unsure how I can command to wait for two confirmations, but at already fixes the GAS LIMIT issue.  
  
const mint = await bba.safeMint("0x123...", 0, 'https://gateway.pinata.cloud/ipfs/url');await mint.wait();const locking= await bba.lockToken(0, "1644598343");await locking.wait();console.log(await bba.getLock(0));  
  
WARN: THIS PARAGRAPH CONTAINS TAG: [CODE]