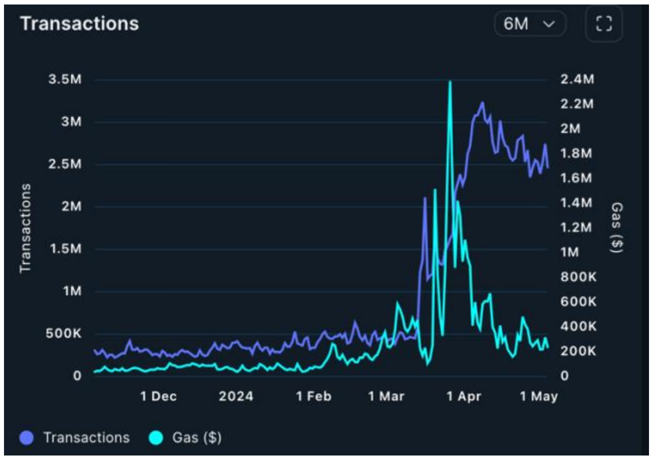
NFT Smart Contract Deployment on Base (using Foundry) 7-May-24

C.G. Belford

Coinbase L2 project Base is rocking. Recent monthly tx volume was > 2x that on the Ethereum blockchain. Base sequencer fees (earned by Coinbase for batching base tx & relaying them to Ethereum blockchain) were the primary revenue driver for "other tx revenue" in the most recent quarterly fin stmt.

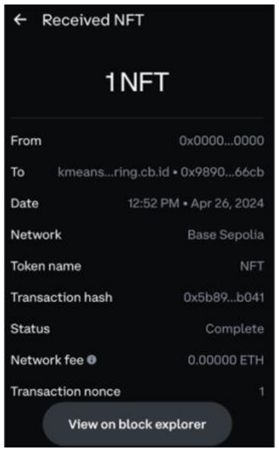
Fig1. Base L2 tx & gas stats (5Nov23 – 4May24)



Source: Nansen

Base is going places. Lets start interacting with it via minting an NFT (ERC721) on Base Sepolia ETH Testnet!

Fig2. Minted NFT hitting our Coinbase Wallet



← Received NFT

1 NFT

From	0x0000...0000
To	kmeans...ring.cb.id • 0x9890...66cb
Date	12:52 PM • Apr 26, 2024
Network	Base Sepolia
Token name	NFT
Transaction hash	0x5b89...b041
Status	Complete
Network fee	0.00000 ETH
Transaction nonce	1

Follow the steps below to mint an NFT.

Step1. Download Coinbase Wallet extension from Chrome Web store

a. https://chrome.google.com/webstore/detail/coinbase-wallet-extension/hnfanknocfeofbddgcijnmhnfnkdnaad?hl=en
b. When finished should see it pinned (blue/white circle) in your chrome browser.

Step2. Get Some Base Sepolia TestNet ETH

a. Need Base Sepolia TestNet ETH (Chain: 84532), not Sepolia TestNet ETH (Chain ID: 11155111) (author's noob mistake, base discord helpdesk was kind 😊).
b. Pretty stable faucet: https://www.ethereum-ecosystem.com/faucets/base-sepolia
c. Enter your Coinbase public address & let the "mining" begin (0.1 ETH s/b enough for starters).

Step3. From terminal - install Foundry (smart contract dev/test/debug tool suite)

a. %curl -L https://foundry.paradigm.xyz | bash
b. If get a libusb not found warning, install it
c. %brew install libusb
d. (In a new terminal): **%foundryup**
e. If successful you will see the 4 main foundryup components (forge, cast, anvil & chisel) installed.

Step4. Create a Project

a. (in terminal) **%mkdir <Your Project Name>**
b. %cd <Your Project Name>
c. %forge init

Step5. Compile a NFT smart contract (ERC721) written in Solidity

a. Add OpenZeppelin contract lib to the Foundry project
b. %forge install openzeppelin/openzeppelin-contracts
c. ....installed openzeppelin-contracts v.5.0.2
d. In Foundry project, dir /src/Counter.sol – delete file contents and replace with:

```
// SPDX-License-Identifier: MIT
pragma solidity ^0.8.23;
import "openzeppelin-contracts/contracts/token/ERC721/ERC721.sol";
contract NFT is ERC721 {
  uint256 public currentTokenId;

  constructor() ERC721("NFT Name", "NFT") {}
  function mint(address recipient) public payable returns (uint256) {
    uint256 newItemId = ++currentTokenId;
    _safeMint(recipient, newItemId);
    return newItemId; }}
```

e. Rename /src/Counter.sol to /src/NFT.sol
f. Delete /tests/Counter.t.sol & /script/Counter.s.sol files
g. Compile NFT contract using foundry/forge: **%forge build**
h. **…**Compiling 12 files…Compiler run successful

Step6. Config Foundry w/Base

a. Import private key to Foundry keystore
b. %cast wallet import deployer –interactive
c. Enter Base Wallet Key & p/w for signing tx
d. ….'deployer' keystore was saved successfully. Address 0x98…
e. Confirm 'deployer' acct setup in Foundry **%cast wallet list**
f. …if successful returns "deployer (Local)".

Step7. Add Base as a Network

a. Create .env file in <Your Project Name> home dir to add Base Network & API key for verifying contract on BaseScan **%vi .env** & add following 3 lines & save.

```
BASE_MAINNET_RPC="https://mainnet.base.org"
BASE_SEPOLIA_RPC="https://sepolia.base.org"
ETHERSCAN_API_KEY="PLACEHOLDER_STRING"
```

```
1. Open vi file: vi .env
2. Press "i" to enter insert
3. Press ESC to exit insert model
4. Press :w! to save
5. Press :q! to exit
```

b. Load env variables **%source .env**

Step8. Deploy NFT Smart Contract to Base Sepolia Testnet

a. %forge create ./src/NFT.sol:NFT --rpc-url $BASE_SEPOLIA_RPC --account deployer
b. If receive "error: a value is required for '--rpc-url <URL>' but none was supplied" , then again run **%source .env**

c. Enter keystore p/w
d. Should now see 3 Deployer Address (YOUR_BASE_WALLET_ADDRESS, DEPLOYED_TO_ADDRESS & Tx hash)

Step9. Perform Call

a. %**cast call <DEPLOYED TO_ADDRESS> --rpc-url $BASE_SEPOLIA_RPC "balanceOf(address)" <YOUR_BASE_WALLET_ADDRESS>**
b. Example: %cast call 0x2117...DA5 --rpc-url $BASE_SEPOLIA_RPC "balanceOf(address)" 0x989...66cb
c. Should receive response =0 in hexadecimal as have deployed NFT contract but no NFTs minted yet & thus acct balance=0:

```
: 0x0000000000000000000000000000000000000000000000000000000000000000
```

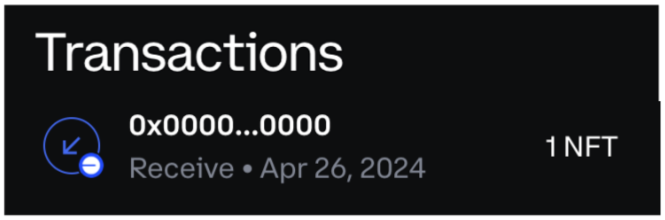
Step10. Sign & Publish Tx

a. Let's call the mint function on the NFT contract deployed above.
b. %**cast send <DEPLOYED_TO_ADDRESS> --rpc-url=$BASE_SEPOLIA_RPC "mint(address)" <YOUR_BASE_WALLET_ADDRESS> --account deployer**
c. Example: %cast send 0x2117...DA5 --rpc-url=$BASE_SEPOLIA_RPC "mint(address)" 0x989...66cb --account deployer
d. If successful Foundry reverts w/info (blockHash, blockNumber, gas stats, status, txHash, etc).
e. To confirm 1 NFT has been minted, run Step9a. %cast call cmd again.
f. Should see response (1 in hexadecimal).

```
0x0000000000000000000000000000000000000000000000000000000000000001
```

1st NFT minted

g. Congrats – you have deployed & minted an NFT smart contract with Foundry.
h. Use BaseScan (Testnet) (https://sepolia.basescan.org) to track your tx activity.
i. Should also see 1 NFT received into your Base Wallet (created in Step1):



Transactions

0x0000...0000
Receive • Apr 26, 2024
1 NFT

Ref: https://docs.base.org/tutorials/deploy-with-foundry/