▣Testing Runes: Getting Started Guide▣

Last updated 4/1/2024

Disclaimer:

Runes have not yet officially launched on Bitcoin. They are not scheduled to go-live until the halving later in April 2024 on block 840,000. This guide is current as of April 1st, 2024 and is for testing purposes only. Changes may still happen in the codebase that affect this guide before go-live, so check back for updates if you're running into any issues.

Prerequisites - Software required:

- A fully indexed **Bitcoin Core** node: https://bitcoin.org/en/download (v25) or https://bitcoincore.org/en/download/ (v26)
- **Ord v0.17.1**:
  Download: https://github.com/ordinals/ord/releases/tag/0.17.1
  Build instructions: https://github.com/ordinals/ord/blob/master/README.md#building

Required reading:

- Runes docs: https://docs.ordinals.com/runes.html
- Batch inscribing and yaml spec: https://docs.ordinals.com/guides/batch-inscribing.html?highlight=postage#example-batchyaml

Sources:

- Ordinals Coding Club (3/20/24) https://www.youtube.com/watch?v=Abqk-am5IGo&t=4714s
- Ordinals Coding Club (3/20/24) https://www.youtube.com/live/cDIHGLJvPh4?si=HXNIObLJwKkop90S

Troubleshooting:

Need help? Ask in the Ordicord - the official ord Discord

Corrections? Ordicord: @cryptoni9n

Etching a mintable Rune

1) First, run the command `ord env` to spin up a regtest bitcoind and ord instance. This will create a new folder called ‘env’ on disk that will be used as your data directory for regtest. It will also start a regnet ordinals explorer at localhost:9001. The output of the command is below - note that it also provides example syntax for both regtest instances. Leave this window running and perform commands in a new window.

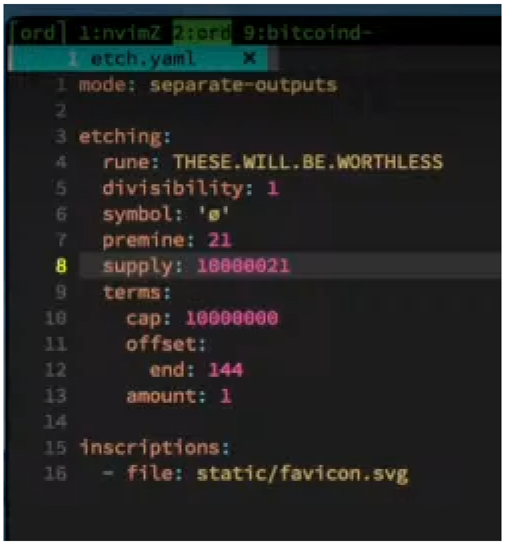
```
C:\Users\nine\ord\target\release>ord env
Listening on http://0.0.0.0:9001
`ord` server URL:
http://127.0.0.1:9001
Example `bitcoin-cli` command:
bitcoin-cli -datadir=env getblockchaininfo
Example `ord` command:
C:\Users\nine\ord\target\release\ord.exe --datadir env wallet balance
```

2) The act of creating a new rune and configuring its metadata is called ‘etching’. We etch the rune using a YAML file called test.yaml, (see step 3 for yaml setup). Etching uses a commit/reveal scheme so after sending this commit transaction, we must wait 6 blocks before the reveal transaction becomes available, when the commit “matures”. This wait time helps protect the commit transaction while it is in the mempool. The etching commit tx is accepted by ord when it returns the message “**Waiting for rune commitment to mature…**”

```
ord --regtest --cookie-file env\regtest\.cookie --data-dir env wallet batch --batch F:\example.yaml
--fee-rate 2
```

3) Below is the template for the test.yaml file. It contains a mode, etching, and inscriptions section. A rune etching and inscription can be created at the same time or separately. If an etching and inscription are created in the same transaction, the inscription will appear on the rune page as a logo and the two will be permanently linked. See the template for the YAML file below:

```
mode: separate-outputs

etching:
  rune: <13+ character rune name goes here, can use periods to separate words>
  divisibility: <# of subunits in superunit in powers of 10, 0 is not subdividable, see chart
on pg 7>
  premine: <total number of runes to be minted to you during the etch>
  supply: <mandatory; the (amount*cap + premine); the total number of runes that can be
minted>
  symbol: <single character rune symbol>
  terms:
    cap: <the total number of mint tx allowed>
    offset:
      end: <relative number of blocks before etch is unmintable>
    amount: <amount minted during each mint tx>

inscriptions:
 - file: <path to image to be inscribed alongside rune etching; image will be seen on rune
page, and vice versa, optional>
```

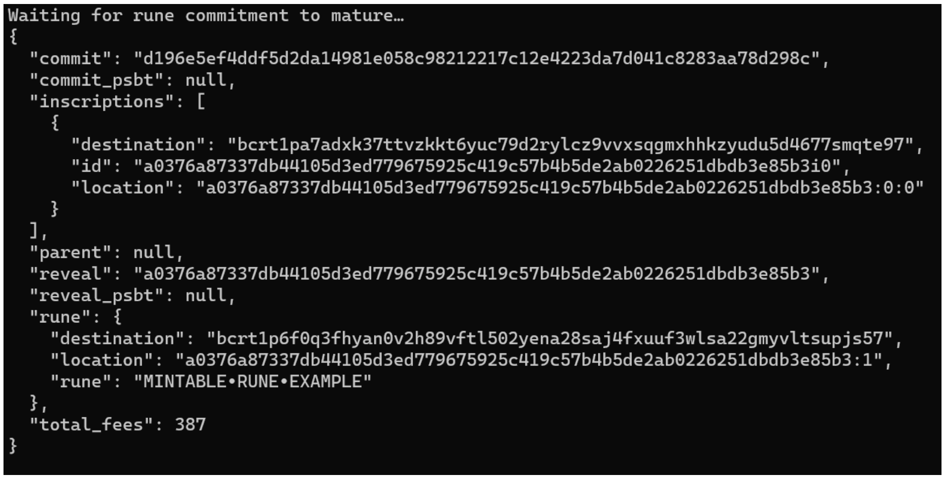


```
mode: separate-outputs

etching:
  rune: THESE.WILL.BE.WORTHLESS
  divisibility: 1
  symbol: 'ø'
  premine: 21
  supply: 10000021
  terms:
    cap: 10000000
    offset:
      end: 144
    amount: 1

inscriptions:
  - file: static/favicon.svg
```

4) In regtest, we must manually mine blocks by first creating a wallet receive address (`ord wallet receive`) and then generating the block rewards to that address. This will need to be done in a separate command window from the waiting commit tx. Use the following command:

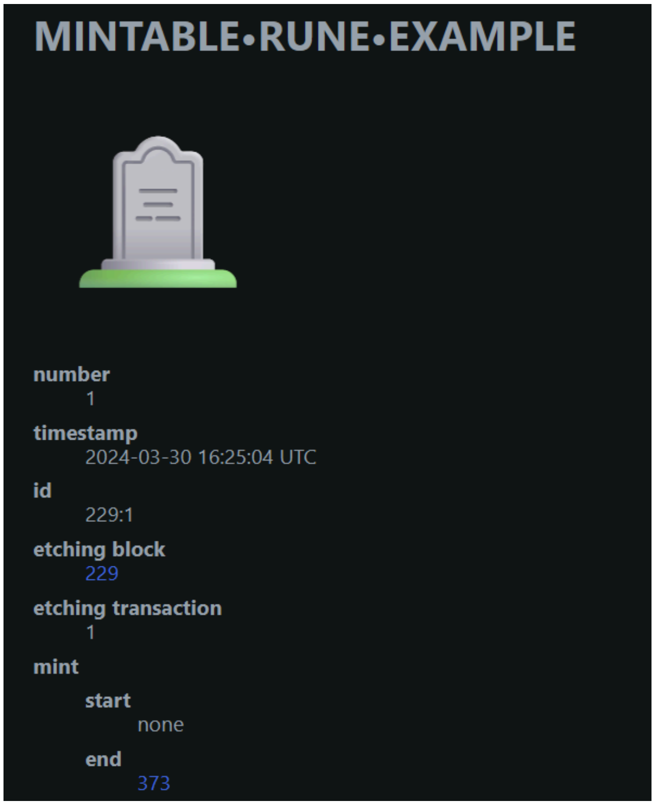
```
bitcoin-cli -datadir=env generatetoaddress <number_of_blocks> <ADDRESS>
```

5) After you've mined 6 blocks, the reveal transaction will be in the mempool. Mine one more block for the transaction to be finished. You'll know you've mined enough blocks when the commit tx window returns the reveal tx metadata:



```
Waiting for rune commitment to mature…
{
  "commit": "d196e5ef4ddf5d2da14981e058c98212217c12e4223da7d041c8283aa78d298c",
  "commit_psbt": null,
  "inscriptions": [
    {
      "destination": "bcrt1pa7adxk37ttvzkkt6yuc79d2rylcz9vvxsqgmxhhkzyudu5d4677smqte97",
      "id": "a0376a87337db44105d3ed779675925c419c57b4b5de2ab0226251dbdb3e85b3i0",
      "location": "a0376a87337db44105d3ed779675925c419c57b4b5de2ab0226251dbdb3e85b3:0:0"
    }
  ],
  "parent": null,
  "reveal": "a0376a87337db44105d3ed779675925c419c57b4b5de2ab0226251dbdb3e85b3",
  "reveal_psbt": null,
  "rune": {
    "destination": "bcrt1p6f0q3fhyan0v2h89vftl502yena28saj4fxuuf3wlsa22gmyvltsupjs57",
    "location": "a0376a87337db44105d3ed779675925c419c57b4b5de2ab0226251dbdb3e85b3:1",
    "rune": "MINTABLE•RUNE•EXAMPLE"
  },
  "total_fees": 387
}
```

6) Now, you can check for your new rune on your regtest ordinal explorer, which is at http://localhost:9001/runes



MINTABLE•RUNE•EXAMPLE

number
1

timestamp
2024-03-30 16:25:04 UTC

id
229:1

etching block
229

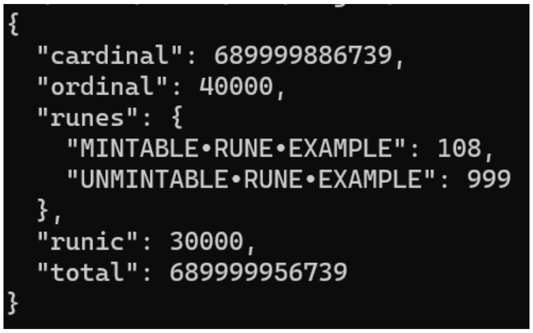
etching transaction
1

mint

start
none

end
373

7) Running the command `ord wallet balance` will now show you your wallet's runes balances.

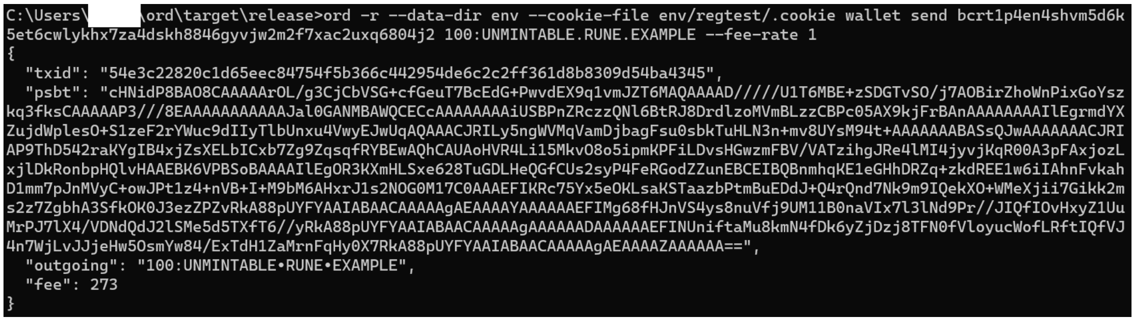


```
{
  "cardinal": 689999886739,
  "ordinal": 40000,
  "runes": {
    "MINTABLE•RUNE•EXAMPLE": 108,
    "UNMINTABLE•RUNE•EXAMPLE": 999
  },
  "runic": 30000,
  "total": 689999956739
}
```

8) In order to send runes to a <RECEIVE_ADDRESS>, you can run the command:

```
ord --cookie-file <COOKIE_PATH> --data-dir <ENV PATH> wallet send <RECEIVE_ADDRESS>
<AMOUNT>:<RUNENAME> --fee-rate 1
```

*Note: if the rune you’re sending is indivisible (divisibility=0) then sending any amount will send all of the runes for that etched rune.



```
C:\Users\[redacted]\ord\target\release>ord -r --data-dir env --cookie-file env/regtest/.cookie wallet send bcrt1p4en4shvm5d6k
5et6cwlykhx7za4dskh8846gyvjw2m2f7xac2uxq6804j2 100:UNMINTABLE.RUNE.EXAMPLE --fee-rate 1
{
  "txid": "54e3c22820c1d65eec84754f5b366c442954de6c2c2ff361d8b8309d54ba4345",
  "psbt": "cHNidP8BAO8CAAAAArOL/g3CjCbVSG+cfGeuT7BcEdG+PwvdEX9q1vmJZT6MAQAAAAD/////U1T6MBE+zSDGTvSO/j7AOBirZhoWnPixGoYsz
kq3fksCAAAAAP3///8EAAAAAAAAAAAJal0GANMBAWQCECcAAAAAAAAiUSBPnZRczzQNl6BtRJ8DrdlzoMVmBLzzCBPc05AX9kjFrBAnAAAAAAAAIlEgrmdYX
ZujdWplesO+S1zeF2rYWuc9dIIyTlbUnxu4VwyEJwUqAQAAACJRILy5ngWVMqVamDjbagFsu0sbkTuHLN3n+mv8UYsM94t+AAAAAAAABASsQJwAAAAAAACJRI
AP9ThD542raKYgIB4xjZsXELbICxb7Zg9ZqsqfRYBEwAQhCAUAoHVR4Li15MkvO8o5ipmKPFiLDvsHGwzmFBV/VATzihgJRe4lMI4jyvjKqR00A3pFAxjozL
xjlDkRonbpHQlvHAAEBK6VPBSoBAAAAIlEgOR3KXmHLSxe628TuGDLHeQGfCUs2syP4FeRGodZZunEBCEIBQBnmhqKE1eGHhDRZq+zkdREE1w6iIAhnFvkah
D1mm7pJnMVyC+owJPt1z4+nVB+I+M9bM6AHxrJ1s2NOG0M17C0AAAEFIKRc75Yx5eOKLsaKSTaazbPtmBuEDdJ+Q4rQnd7Nk9m9IQekXO+WMeXjii7Gikk2m
s2z7ZgbhA3SfkOK0J3ezZPZvRkA88pUYFYAAIABAACAAAAAgAEAAAAYAAAAAAEFIMg68fHJnVS4ys8nuVfj9UM11B0naVIx7l3lNd9Pr//JIQfIOvHxyZ1Uu
MrPJ7lX4/VDNdQdJ2lSMe5d5TXfT6//yRkA88pUYFYAAIABAACAAAAAgAAAAAADAAAAAAEFINUniftaMu8kmN4fDk6yZjDzj8TFN0fVloyucWofLRftIQfVJ
4n7WjLvJJjeHw5OsmYw84/ExTdH1ZaMrnFqHy0X7RkA88pUYFYAAIABAACAAAAAgAEAAAAZAAAAAA==",
  "outgoing": "100:UNMINTABLE•RUNE•EXAMPLE",
  "fee": 273
}
```

9) Mine another block (see step 4 for syntax) in order to send the Runes. Note, only one block is necessary because we are not etching a new rune, just sending some to another address.

10) Now that the runes have been sent, check your wallet balance again and note that the amount has been decremented.

Etching a non-mintable Rune

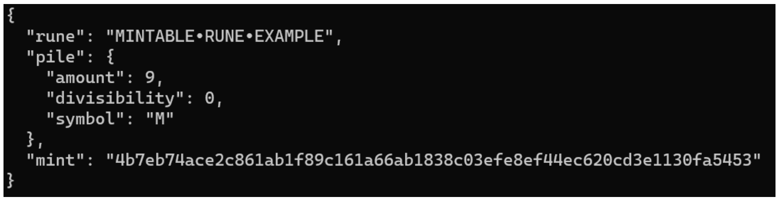
1) There are only minor differences in the etching process when it comes to etching an unmintable rune. These changes will go into the test.YAML from step 3. The template for the mintable test.YAML file is below:

```
mode: separate-outputs

etching:
  rune: <13+ char rune name, use periods to separate words if you like>
  divisibility: <0 = indivisible, see divisibility chart on pg 7>
  premine: <mandatory; number to mint to yourself at etch time>
  symbol: <single digit rune symbol>
  supply: <should equal the premine for an unmintable rune>

inscriptions:
  - file: <path to inscription file, optional>
```

Minting a Rune

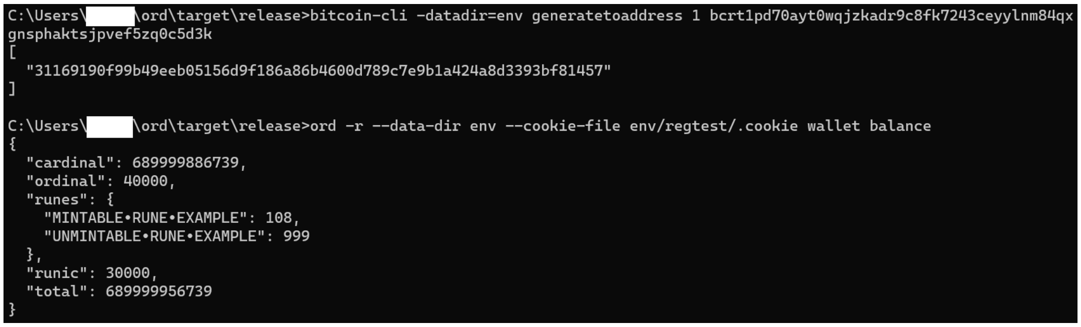
1) Minting a rune is done with the following command:

```
ord --cookie-file <COOKIE_PATH> --data-dir <ENV_PATH> wallet mint --rune <RUNENAME> --fee rate 1
```



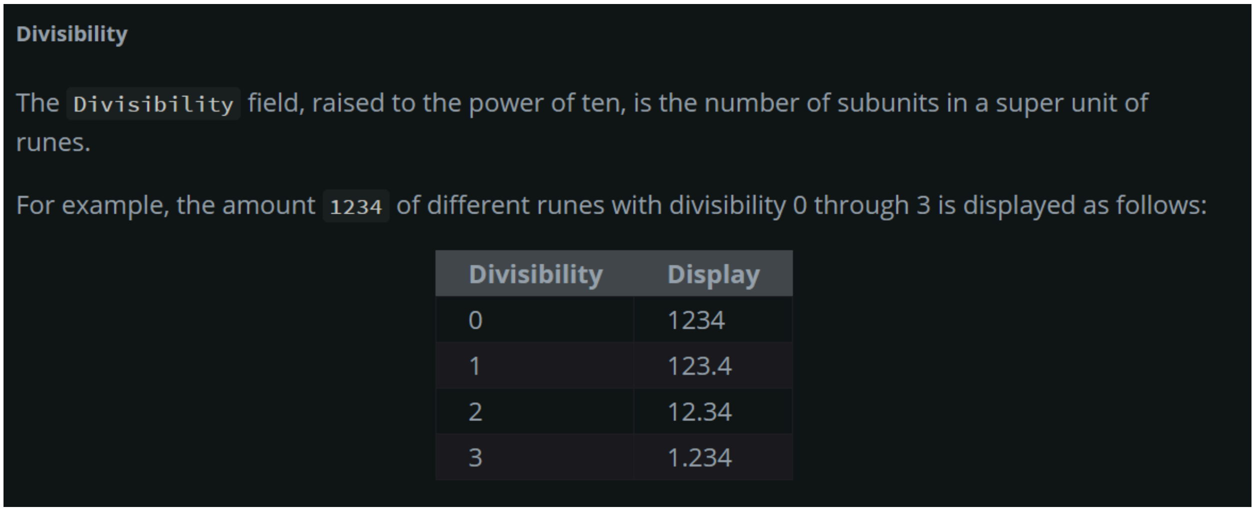
```
{
  "rune": "MINTABLE•RUNE•EXAMPLE",
  "pile": {
    "amount": 9,
    "divisibility": 0,
    "symbol": "M"
  },
  "mint": "4b7eb74ace2c861ab1f89c161a66ab1838c03efe8ef44ec620cd3e1130fa5453"
}
```

2) Be sure to mine another block and then check your wallet balance to confirm your receipt of minted runes.



```
C:\Users\[redacted]\ord\target\release>bitcoin-cli -datadir=env generatetoaddress 1 bcrt1pd70ayt0wqjzkadr9c8fk7243ceyylnm84qx
gnsphaktsjpvef5zq0c5d3k
[
  "31169190f99b49eeb05156d9f186a86b4600d789c7e9b1a424a8d3393bf81457"
]

C:\Users\[redacted]\ord\target\release>ord -r --data-dir env --cookie-file env/regtest/.cookie wallet balance
{
  "cardinal": 689999886739,
  "ordinal": 40000,
  "runes": {
    "MINTABLE•RUNE•EXAMPLE": 108,
    "UNMINTABLE•RUNE•EXAMPLE": 999
  },
  "runic": 30000,
  "total": 689999956739
}
```

Divisibility Chart



Divisibility

The `Divisibility` field, raised to the power of ten, is the number of subunits in a super unit of runes.

For example, the amount `1234` of different runes with divisibility 0 through 3 is displayed as follows:

Divisibility	Display
0	1234
1	123.4
2	12.34
3	1.234

Glossary:

Rune: shitcoins on Bitcoin; a digital commodity
Etch: a transaction that creates and defines a new Rune and its properties
Mint: a transaction that allocates a number of runes from an etched Rune
Cenotaph: an etching or minting that is in error becomes invalid and burned
Edict: (not covered yet) the transfer of runes to another address at etch time