

# NEAR Lake indexer basic tutorial

Source code for the tutorial [frolvanya/near-lake-raw-printer](#) : source code for the tutorial on how to create an indexer that prints block height and number of shards Recently we have [published a Python version of the NEAR Lake Framework](#) on pypi.org

We want to empower you with a basic tutorial on how to use the Python Package. Let's get started!

## Create a project

Create an indexer project:

```
mkdir near-lake-raw-printer && cd near-lake-raw-printer touch main.py
```

## Install dependencies

Install near-lake-framework

```
pip3 install near-lake-framework
```

## Import near-lake-framework

In the main.py file let's import the necessary dependencies:

```
from near_lake_framework import near_primitives, LakeConfig, streamer
```

We've imported the main function streamer which will be called to actually run the indexer, near\_primitives and LakeConfig type we need to construct.

## Create a config

Add the instantiation of LakeConfig below:

## config

```
LakeConfig . mainnet ( ) config . start_block_height =
```

```
69030747 config . aws_access_key_id = os . getenv ( "AWS_ACCESS_KEY_ID" ) config . aws_secret_key = os . getenv ( "AWS_SECRET_ACCESS_KEY" )
```

Just a few words on the config, function mainnet() has sets3\_bucket\_name, s3\_region\_name for mainnet. You can go to [NEAR Explorer](#) and get the most recent block height to set config.start\_block\_height .

## Starting the stream

Let's call streamer function with the config :

```
stream_handle, streamer_messages_queue = streamer ( config ) while
```

```
True : streamer_message =
```

```
await streamer_messages_queue . get ( ) print ( f"Block # { streamer_message . block . header . height } Shards: { len ( streamer_message . shards ) } " )
```

And an actual start of our indexer in the end of the main.py

## loop

```
asyncio . get_event_loop ( ) loop . run_until_complete ( main ( ) )
```

## All together

```
import asyncio import os
```

```
from near_lake_framework import LakeConfig, streamer, near_primitives
```

async

def

main ( ) : config = LakeConfig . mainnet ( ) config . start\_block\_height =

69030747 config . aws\_access\_key\_id = os . getenv ( "AWS\_ACCESS\_KEY\_ID" ) config . aws\_secret\_key = os . getenv ( "AWS\_SECRET\_ACCESS\_KEY" )

stream\_handle , streamer\_messages\_queue = streamer ( config ) while

True : streamer\_message =

await streamer\_messages\_queue . get ( ) print ( f"Block # { streamer\_message . block . header . height } Shards: { len ( streamer\_message . shards ) } " )

## loop

asyncio . get\_event\_loop ( ) loop . run\_until\_complete ( main ( ) ) That's it. Now we run main.py

python3 main.py You should see something like the following:

Received 400 blocks from S3 Block #69030747 Shards: 4 Block #69030748 Shards: 4 Block #69030749 Shards: 4 Block #69030750 Shards: 4 Block #69030751 Shards: 4 Block #69030752 Shards: 4 Block #69030753 Shards: 4 Block #69030754 Shards: 4 You can stop the indexer by pressing CTRL+C

Credentials To be able to access the data from [NEAR Lake](#) you need to provide credentials. Please, see the [Credentials](#) article You can find the [source code for this tutorial on the GitHub](#) . [Edit this page](#) Last updated on Dec 9, 2023 by gagdiez Was this page helpful? Yes No

[Previous JS basic tutorial](#) [Next NFT Indexer](#)