Dexr

Decentralized Cryptocurrency Exchange Platform

Getting Started Guide

Table of Contents

Getting Started

1.1 Prerequisites

- Active internet connection
- Installation of .NET Framework 4.6 (https://www.microsoft.com/en-my/download/details.aspx?id=48130)

1.2 Supported OS

Windows

- Windows 7 SP1
- Windows 8.1
- Windows 10 Anniversary Update (version 1607) or later versions
- Windows Server 2008 R2 SP1 (Full Server or Server Core)
- Windows Server 2012 SP1 (Full Server or Server Core)
- Windows Server 2012 R2 (Full Server or Server Core)
- Windows Server 2016 or later versions (Full Server, Server Core, or Nano Server)

Note: Linux and MacOS support will be available later once the project is converted to use .NET Core framework.

1.3 Node Installation on Windows OS

Download and extract the contents of dexr-win-x86-x64.zip to a folder named “dexr-client”. No installation required.

1.4 Node Configuration

Open the dexr-client folder. The **config.json** file stores general settings for the node. Open the config.json file in your text editor to configure the following settings:

```
{
  "ChainFolderName": "Chain",
  "ServerHostType": "localhost",
  "ServerPort": 8080,
  "SeedList": [
    "http://localhost:8080"
  ],
  "System": {
    "MemCacheSizeMB": 1024
  }
}
```

Settings	Description
ChainFolderName	The folder name to store all chain data. Will be automatically created in the same folder as the application.
ServerHostType	Values: localhost | default | localhost_https | default_https

	localhost: Runs the node on local machine using http protocol. default: Runs the node on the public IP address using http protocol. localhost_https: Runs the node on local machine using https protocol. default_https: Runs the node on the public IP address using https protocol.
ServerPort	The port number to use for incoming traffic.
SeedList	The list of seed nodes in which to connect to the network. A minimum of one seed node is required.
MemCacheSizeMB	The amount of memory the application is allowed to use for caching and performance indexing.

Note: If you are only running the node on a single machine, you can set the ServerHostType to localhost. To run multiple nodes, duplicate the dexr-client folder and modify each individual config.json file so that each node will have a different ServerPort setting. Consensus will then run on the same machine with multiple nodes. It is also possible to run consensus with only one node.

Open the dexr-client folder. The **deploy.json** file stores settings for creating a new chain. Open the deploy.json file in your text editor to configure the following settings:

```
{
  "NativeTokenName": "DEXR Exchange Token",
  "NativeTokenSymbol": "DXR",
  "InitialSupply": "100000000",
  "Decimals": 8
}
```

Settings	Description
NativeTokenName	The name of the native token.
NativeTokenSymbol	The native token symbol.
InitialSupply	The circulating supply for the native token – fully redeemed upon block creation.
Decimals	The number of decimals for the native token.

For developer reference: Application embedded configuration are found in the **ConstantConfig.cs** file in the Dexr pre-compiled source code. The ConstantConfig.cs file can be found in the following location:

DEXR.Core/Configuration/ConstantConfig.cs

```
/// <summary>
/// Interval between each block in seconds
/// </summary>
public const int BlockInterval = 15;

/// <summary>
/// The network fee rate per TransactionsCount/NetworkFeeUnit
/// Eg: 5000 Transactions in a block with NetworkFeeUnit 1000 would yield
/// MinimumFee = 0.0002 * (5000/1000) = 0.001
/// </summary>
public const decimal NetworkFeeRate = 0.0002M;

/// <summary>
/// The network fee unit per Transactions
/// </summary>
public const long NetworkFeeUnit = 1000;

/// <summary>
/// The maximum number of background threads to use when
/// broadcasting a message to connected nodes
/// </summary>
public const int BroadcastThreadCount = 4;

/// <summary>
/// Determines whether the node will verify transactions signature
/// before pushing the transaction to queue. Disable this setting only
/// for debugging reasons.
/// </summary>
public const bool EnableSignatureVerification = false;
```

Using the Command Line Interface (CLI)

Run the dexr.exe application to launch the Command Line Interface (CLI). The CLI is the starting point for interacting with the Dexr blockchain.

2.1 Creating a new wallet

```
> create wallet my_wallet.json
```

Enter the above command to create a new wallet file named “my_wallet.json”. You may use any name for your wallet file so long as it is of .json extension. The wallet file will be created in the same folder as the Dexr application.

2.2 Opening a wallet file

```
> open wallet my_wallet.json
```

Enter the above command to open the wallet file named “my_wallet.json”. You must open a wallet to perform certain actions like creating a new chain and starting consensus. Only one wallet can be opened at a time.

2.3 Creating the blockchain

```
> create new chain
```

Enter the above command to start a new chain and create the genesis block. Do this only for the first time when there is no network yet. You must open a wallet first. The opened wallet address will be the owner of the new chain.

2.4 Joining the consensus network

```
> start consensus
```

Enter the above command to join the network and start consensus. This action will also start your node’s API server. You will need to allow incoming and outgoing connections on your firewall. You must open a wallet first. The opened wallet balance is used for the proof-of-stake consensus.

2.5 Start API Server

```
> start server
```

Enter the above command to start the API server on your node. This action is not necessary if you have already started consensus.

2.6 Deleting a chain local

```
> delete chain
```

Enter the above command to delete the chain folder on your node.

2.7 Show current block height

```
> show height
```

Enter the above command to show the current block height synced on your node.

Transactions (API)

Transactions are sent to the network via HTTP web requests posted to any node connected on the network. To learn about all available endpoints, see the full API documentation.

3.1 Creating a new token

```
POST transaction/create-token
```

See the Create Token endpoint for details and example.

3.2 Making a token transfer

```
POST transaction/transfer
```

See the Transfer endpoint for details and example.

3.3 Placing limit orders

```
POST transaction/limit-order
```

See the Limit Order endpoint for details and example.

3.4 Placing market orders

```
POST transaction/market-order
```

See the Market Order endpoint for details and example.

3.5 Cancelling orders

```
POST transaction/cancel-order
```

See the Cancel Order endpoint for details and example.

Browsing the Blockchain (API)

The built in indexing allows you to browse the blockchain and receive summaries instantaneously. Requests are sent via HTTP web request to any single node connected on the network. To learn about all available endpoints, see the full API documentation.

4.1 View current block height

```
GET view/height
```

See the View Height endpoint for details and example.

4.2 View individual blocks

```
POST view/blocks
```

See the View Blocks endpoint for details and example.

4.3 View pending transactions in queue for next block

```
GET view/pending-transactions
```

See the View Pending Transactions endpoint for details and example.

4.4 View all tokens

```
GET view/tokens
```

See the View All Tokens endpoint for details and example.

4.5 View wallet balance

```
POST view/wallet-balance
```

See the View Wallet Balance endpoint for details and example.

4.6 View exchange orders

```
POST view/orders
```

See the View Orders endpoint for details and example.

4.7 View current network fee

```
POST view/network-fee
```

See the View Network Fee endpoint for details and example.