DigiByte Integration Guide

V1.6

DigiByte

BLOCKCHAIN

www.digibyte.io

Table of Contents

Thank you for your interest in DigiByte!

DigiByte is a 6-year old UXTO blockchain that started in 2014 with a focus on speed, scalability and security.

DigiByte is proudly not an ICO, and the small 0.5% pre-mine was given away completely to community members in full in the first 30 days of creation, in order to encourage adoption and full-node downloads.

DigiByte has no founders reward or block-fee, and although the founder is still actively engaged in DigiByte, any DigiByte held by him or other developers was obtained through purchasing on an Exchange, or mining on their own, just as any and all other users would. In addition, there is no individual that controls any significant portion of the circulating supply, such as the founder or developers. DigiByte is highly decentralized and fairly distributed.

DigiByte has a sound reputation both for its forward thinking and for its rapid responses to changing conditions. DigiByte was the first major UTXO blockchain to implement SegWit, several weeks ahead of both Litecoin / Bitcoin. Similarly, DigiByte was the first non-Bitcoin blockchain to implement a fix for CVE-2018-17144.

DigiByte is also the first major UTXO blockchain to implement the Dandelion privacy protocol with the 7.17 release.

DigiByte pioneered the DigiShield difficulty adjustment algorithm that is used in dozens of other blockchains, such as Ethereum, Zcash, Dogecoin and more. DigiByte was also the first blockchain to switch from a single algorithm to multi-algorithm for the increased security that 5x mining algorithms provides. DigiByte will also be pursuing ASIC-resistant algorithms throughout 2020 / 2021 through the likes of GPU and CPU-focused mining algorithms. DigiByte recently implemented Odocrypt in 2019: an FPGA-focused algorithm created specifically for DigiByte that changes every 10 days making ASIC creation pointless. Another solid first.

DigiByte mining is also highly distributed and the most decentralized PoW in the world. Unlike X16R, DigiBytes MultiAlgo implementation is not a selection of hashing algorithms in a pseudo-random manner, but rather the 5x algorithms each actively compete against each other to mine 20% of all blocks. This also further encourages the immediate processing of all transactions, due to other hardware vendors and types being used otherwise across all algorithms.

Finally, we want to welcome you to the global DigiByte community, and to thank you for taking the time to integrate with DigiByte. Our vast community is incredibly passionate about the DigiByte blockchain and we are sure that you will be too.

DigiByte Technical Specifications

Denominations

Plural	**DigiBytes**
Ticker symbol	**DGB**
Currency symbol	**Ɗ (Unicode: U+018A)**
Precision	**10^{-8}**

Subunits

mDGB (miliDigiByte)	**1/1000**
µDGB (microDigiByte)	**1/1000000**
ɗSats (digiSatoshi, Digis)	**1/100000000**

Development

Original author(s)	**Jared Tate**
Implementation(s)	**DigiByte Protocol** **DigiByte Core** **DigiAssets Protocol**
Initial release	**1.0 / 10 January 2014**
Latest release	**7.17.2 / 3 May 2019**
Development status	**Active (Working Product)**
Website	**DigiByte.io**

Ledger

Ledger start	January 10th, 2014
Genesis Block Hash	"USA Today: 10/Jan/2014, Target: Data stolen from up to 110M customers"
Ledger type	Public, Decentralized, UTXO based, Multi-Algorithm
Timestamping scheme	Proof-of-work (Partial hash inversion)
Hash function	Five individual SHA256, Scrypt, Odocrypt, Skein & Qubit
Issuance schedule	Decentralized (block reward) Initially Ɗ72,000 per block, 1% reduced every monthly
Block time	15 seconds, (75 seconds per Algo)
Algorithm block share	20% Block Share per Algorithm (5 Algorithms x 20% for total Block Share)
Difficulty retarget	Every 1 Block, 5 Separate Difficulties, 1 For each Mining Algo
Block size and capacity	Max block size 1MB. 1066 TPS with 4x SegWit scaling

Block explorer	**digiexplorer.info, chainz.cryptoid.info, ccore.online & more...**
Supply limit	**Ɗ21,000,000,000 (Supply limit reached in 2035)**
Address formats	**"D" prefixed legacy addresses** **"S" prefixed p2sh SegWit compatible / MultiSig addresses** **"dgb1" prefixed bech32 native SegWit addresses**
SegWit support	**Yes! First major Altcoin to successfully activate SegWit (April 2017)**

Where should we download the DigiByte source code from?

You can find the source code at:

https://github.com/DigiByte-Core/DigiByte

Previously the code was at digibyte/digibyte however this has been moved to an organization as of early 2020 in order to best accommodate the growth of the project and in best-keeping with industry expectations.

What are the standard ports for DigiByte integration?

DigiByte uses the following ports:

RPC Port: 14022
P2P Port: 12024
Testnet RPC: 14023
Testnet P2P: 12026

All ports are TCP. While you do not need to have them publicly exposed / forwarded, doing-so does not present any security risk and allows your node to further contribute to the security of the DigiByte network by being discoverable by other nodes, as well as minimizing any possibility of a Sybil / isolation attack.

What is the DigiByte Magic Number?

DigiBytes Magic Number is:

0xfac3b6da

What is the derivation path for DigiByte?

DigiByte legacy (D-prefix) addresses use a value of: **m/44'/20'/0'/0/0**

DigiByte SegWit (S-prefix) addresses use a value of: **49**

DigiByte Bech32 (dgb1-prefix) addresses use a value of: **84**

What should integrators be aware of regarding Dandelion?

Dandelion is implemented in to DigiByte Core 7.17.2, alongside Odocrypt.
With Dandelion the transactions are put into the "**stempool**", and the "**mempool**" second once the transaction has flowered.

If you are checking "**gettxout**" or similar for a transaction, please be aware it will not show immediately if you are using Dandelion.

You can alternatively disable dandelion using "**-disabledandelion=1**" as a launch flag, however, we recommend only using this as a last-resort.

What should integrators be aware of regarding Odocrypt?

Odocrypt is a new and unique hashing algorithm that morphs itself every 10 days, and will be replacing the Myr-Groestl algorithm from block 9,100,000 as part of DigiByte Core 7.17.2

Odocrypt was designed specifically to be ASIC resistant, due to these changes every 10 days, making the notion of creating an ASIC redundant as a company would effectively be creating a FPGA.

Odocrypt is already live on the DigiByte testnet and testing against it can be performed immediately.

Please keep in mind the change of epoch occurs daily at UTC+0 on testnet, where it occurs every 10 days on mainnet.

You can find more information about it at:

https://github.com/digibyte-core/digibyte/blob/7.17.2/src/crypto/odocrypt.cpp

Alternatively, you can find more information on the wiki:

https://www.dgbwiki.com/index.php?title=Odocrypt

Do you have any good practices in integration with your wallet for deposits and withdrawals?

We recommend a minimum of **6** confirmations **for deposits** / **withdrawals** (90 seconds), however as your monitor the value of transactions being deposited, we recommend **increasing this up to 50 for higher-value deposits**, in order to account for the SHA256 & Scrypt algorithms, in the unlikely event of a collusion from some of the larger BTC / LTC mining pools.
Notifications usually occur within 1-2 seconds for non-Dandelion transactions, with block creation remaining unaffected at every 15 seconds.

If possible, use either **Bech32** or **SegWit** addresses, and **batch-process transactions**. DigiByte expects to add Schnorr Signatures in 2020, and these will be recommended for further transaction space-saving.

Assuming that we will have 10 million users, how many wallets do you recommend we should have? Should we divide wallets / servers for different tasks, e.g. address generating, deposit handling, withdrawals?

This genuinely depends on the capacity of the server, and the optimization of the software integrating with it. Look at your Bitcoin server utilization for a good indication for you to base your DigiByte node from, however keep in mind that DigiByte will consume ~4GB RAM without any additional software interfacing with it so considerations should be made based around this.

What parameters are recommended during installation / running a wallet?

You can connect to more than just the default number of nodes to ensure a geographical distribution, specifically adding the following to your digibyte.conf:

addnode=seed1.digibyte.io
addnode=seed2.digibyte.io
addnode=seed3.digibyte.io
addnode=seed.digibyte.io
addnode=seed.digibyteprojects.com
addnode=digihash.co
addnode=digiexplorer.info
addnode=seed.digibyteguide.com
addnode=explorer-1.us.digibyteservers.io

However, in order to ensure a connection to as many nodes as possible as a “best practice”, you can also use something such as:

maxconnections=300

What are the required hardware resources to run a single node?

A single node can happily seed the DigiByte blockchain on any *64-bit CPU with 4GB RAM and 40GB HDD at present (Early 2020)*. The DigiByte Core wallet will run on almost any Windows, Linux or OSX computer. However, this is probably less than ideal for an **Enterprise-grade production environment.**

When interfacing with the blockchain, most of the time you will be IO-bound through drive-access. Any recent SSD will suffice, but a spinning platter HDD is not recommended. Running off a mid-range Intel Core i5 CPU will allow you to fully sync the DigiByte blockchain in approximately 4hours. Additional services such as Insight APIs will likely take a further ~24hours to sync on top.

While 4GB of RAM is currently the bare minimum recommended for a Linux VM, 8GB is the least that you should consider for deploying in a production environment, with 16GB being ideal if you are utilizing additional services, such as the Insight API services.

Also, you should consider future growth of the blockchain, where it now consumes approx. 25GB (including indexes), future growth requirements would suggest you actively monitor whatever server you run this on for drive-space usage etc.

Once your server is in production, you may also want to look at regularly clearing / logrotating the debug.log file in ~/.digibyte/ . Please keep in mind that doing-so will severely limit your troubleshooting abilities down the line, should an issue arise.

Also, digibyted can and should always be run as *a non-privileged user* from within their home directory, and sudo access is not required at any time.

Where are the 32-bit binaries?

As of 2018, 32-bit builds are no longer supported due to the number of blocks exceeding the addressable memory-space of 32-bit architecture.

DigiByte runs purely on **64-bit** for the time being.

You can read more about this here:

https://github.com/digibyte/digibyte/issues/144

What are the practices to backup / restore wallet?

As per Bitcoin, *backing up the private keys or wallet.dat* is sufficient, though care should be taken with encryption etc.

We recommend where possible you use a **2-of-3 MultiSig or similar setup.**

Are there any known restrictions we should know about?

Since early 2018, DigiByte has been in the process of changing newly generated wallet addresses from the legacy "**D**" prefix to "**dgb1**" as part of bech32 support, and from "**3**" to "**S**" for **SegWit** addresses. Legacy address formats will continue to be supported indefinitely, so please allow for capacity to send to both. As of DigiByte Core 8.19.0 we will generate Bech32 addresses by default, so support is a priority.

As of **DigiByte Core 6.16.2**, DigiByte has implemented upstream Bitcoin Core 0.17 pre-release RPC formats. Common calls that exchanges / pools need to know about is that **getinfo** has been replaced by:

- **getblockchaininfo**
- **getnetworkinfo**
- **getwalletinfo**
- **getmininginfo**

In addition, **signrawtransaction** has been split in to two calls:

- **signrawtransactionwithkey**
- **signrawtransactionwithwallet**

signrawtransactionwithkey requires private keys to be passed in and does not use the wallet for any signing. **signrawtransactionwithwallet** uses the wallet to sign a raw transaction and does not have any parameters to take private keys.

We strongly advise also being aware of this where integrators are using the RPC calls directly, this call will no doubt be deprecated across other wallets going forward too as they bring themselves up to speed with the Bitcoin Core codebase.

Where existing products and services are using **signrawtransaction**, you should simply use **signrawtransactionwithwallet** in its place.

In addition, if you've been making use of the **ismine** value in **validateaddress**, you'll want to instead call **getaddressinfo**, as the result is being returned there.

The **accounts** RPC call is also being changed; we recommend viewing the release notes from Bitcoin Core:

https://github.com/bitcoin/bitcoin/blob/master/doc/release-notes/release-notes-0.17.0.md

Do you have a sample digibyte.conf that you would recommend?

Certainly, though customizing the configuration file is unneeded to get the node up and running, many opt to further customize it for their interactions with it using something such as:

```
# Place this config in the following path:
# ~/.digibyte/digibyte.conf
server=1
daemon=1
txindex=1
rpcallowip=127.0.0.1
maxconnections=300
addnode=seed1.digibyte.io
addnode=seed2.digibyte.io
addnode=seed3.digibyte.io
addnode=seed.digibyte.io
addnode=seed.digibyteprojects.com
addnode=digihash.co
addnode=digiexplorer.info
addnode=seed.digibyteguide.com
addnode=explorer-1.us.digibyteservers.io
```

Do you have any additional APIs we can use to integrate with the DigiByte Blockchain?

We recommend you run an Insight blockchain explorer service, then use the APIs documented here:

https://github.com/DigiByte-Core/insight-api

A detailed setup guide is available on our Wiki, with step-by-step instructions available here:

https://www.dgbwiki.com/index.php?title=Running_your_own_Insight_explorer

What are the advantages of running a full node?

We recommend all exchanges / wallets / service-providers run their own node they can use for API calls, especially if they are expecting any sort of decent volume. This can be run inside a virtualized environment without any additional configuration requirements when doing-so.

Running your own node will allow the best possible performance for your product. Although there are tens of thousands of active DigiByte nodes, very few run the Insight API on top of it for 3rd-party integration.

In addition, DigiByte was not an ICO and cannot sustain all of the server needs for the entire community directly.

Do you have a running testnet?

Yes, please set:

testnet=1

In your digibyte.conf

The testnet operates on:

TCP 12026.

Who should we contact for additional technical support?

All of our developers and most of our community operate out of Telegram:

https://t.me/DigiByteDevelopers

In addition, the community member who provided this document to you should also be able to put you in touch with somebody who can assist with technical queries via email.

Should English not be your preferred language, please be sure to mention that, as our global community is very multi-lingual.

Can you confirm details for the DigiByte logo that we should be using?

You can download the DigiByte logo in a variety of formats from:

https://github.com/DigiByte-Core/digibyte-logos

DigiByte logos guide:

https://github.com/DigiByte-Core/digibyte-logos/blob/master/Logos%20Bitmap/Guide.png

DigiByte Dark Blue color code:

#002352

DigiByte Light Blue color code:

#0066cc

Please ensure you are writing **D**igi**B**yte with capital **D** & **B** and that you are using the updated Logo described **here**, which was released in October 2017 as this is different to what is on the first post of the BitcoinTalk announcement thread.

It is recommended that you use the Unicode character **Ɗ (U+018A)** when using DigiByte as a currency symbol.

What is the download link for the official Wallet?

The Wallet can be downloaded from:

https://github.com/DigiByte-Core/digibyte/releases

You may also build from the Master branch, as this is usually the latest stable-release branch. Changes are committed to their own branch, which is tagged for release, and then merged in to Master.

Any service should ideally follow best-practice of building from source, however the binaries are naturally available for convenience.

Do you have a Docker image for running a full node?

Yes, this can be found at:

https://github.com/DigiByte-Core/digibyte-docker

Who is the best contact for updates / what is the best way to keep informed about new releases?

We recommend subscribing to the GitHub repository:

https://github.com/DigiByte-Core/digibyte/releases

To remain informed of any updates.

You may also want to pass on any escalation / contact details of your own to whoever provided you with this document, in the case of any high priority issues such as CVE-2018-17144. There are members of the community who have previously contacted exchanges, pool operators and other service providers to inform them of such priority upgrades. With planned algorithm swaps occurring throughout 2020, remaining up-to-date should be a priority.

If you downloaded this document yourself, please reach out to the community through the Developer Channel on Telegram at:

https://t.me/DigiByteDevelopers

Do you have any 3rd parties who can run the infrastructure for us?

Yes, DigiByte is supported at **NOWNodes** and **GetBlock**.

For more information please visit:

https://nownodes.io

https://getblock.io