# Link Blockchain Documentation

***Release 1.0***

**Jonathan Brown <jbrown@bluedroplet.com>**

**Sep 20, 2017**

# Contents

LINK is an uncontrolled linked data ecosystem living on the blockchain. It is fully public. It cannot be censored and no one can be prevented from participating. It is fully programmable at every level. It is a protocol for interconnected distributed content apps that empower the individual.

The blockchain was activated on Thursday 27th April 2017 09:15:23 AM (UTC).

CHAPTER 1

# Useful links

- Homepage
- Blog
- Twitter
- Reddit
- Gitter
- YouTube
- BitChute
- Block explorer
- Network stats
- Solidity APIs
- Issue tracker
- Logo

CHAPTER 2

Table of contents

## How does it work?

LINK is an Ethereum blockchain. A blockchain is a shared database that has a built-in cryptocurrency that is used to financially incentivize the neutrality of the database. No-one can be in charge of a blockchain.

The first blockchain was called Bitcoin. Pretty much the only thing in the Bitcoin database is how much Bitcoin each account has.

Ethereum takes this concept much further. Computer programs called smart contracts can be uploaded into the blockchain and these programs can then store data in the database that everyone has a copy of.

Many big projects are deployed on the Ethereum blockchain. But this presents a problem. One must assume that every software project has fatal bugs that will need to be fixed once they are discovered. Bitcoin had fatal problems early on that needed to be fixed and so did Ethereum.

Unfortunately, when the projects that are deployed on Ethereum need to be fixed they will have a very hard time because they will need to convince the entire blockchain to deploy their fix. This was the problem with The DAO crowdfunding project. An attacker stole $150m because of a bug in the smart contract. In attempting to fix this problem the Ethereum was split into two blockchains: Ethereum and Ethereum Classic.

The only purpose of the LINK blockchain is to run LINK. This means that whenever there is a problem that can only be fixed with a hard fork, it will not be difficult to convince the LINK community to adopt it.

The cryptocurrency of the LINK blockchain is also called LINK.

### BlobStore

The principle smart contract on the LINK blockchain is blobstore. It uses IPFS as the underlying storage layer. Anyone can store a blob of data and get back a blobId. This data is publically readable to everyone.

BlobStore has a rudimentary revisioning system so blobs can be updated while retaining their blobId and revision history.

### Google Protocol Buffers

In order for the data stored in BlobStore to have meaning, there must be some sort of schema. Google Protocol Buffers (Protobuf) is ideal for this. It has an upgradable schema system. This means that LINK has a system of content type inheritance. A standardized story type could be extended with new fields and still be readable by software that only knows how to read standard stories.

Anyone can make a new type. Example types are user profiles, media metadata, tweets, reddit posts, blog posts, business information, reviews, etc.

Protobuf encodes the data very tightly. This data is then compressed with Google Brotli. This minimizes the cost of storing information on LINK.

### Additional smart contracts

Anyone can deploy additional smart contracts onto the LINK blockchain. An important example would be a feed contract that would provide functionality akin to RSS / Twitter / YouTube.

Other examples would be contracts to vote for content or tag content.

## Cryptofuel

Name: LINK

Code: LINK

Symbol:

BIP44 coin type: 76 / 0x8000004c

All transactions on the LINK Blockchain must be paid for with LINK. Transaction size is measured in "gas". Transactions can be prioritized by paying a higher gas price.

## Issuance

Project revenue of 55 million LINK (equivalent to 5 years of mining) is pre-allocated into the *Revenue* smart contract. It will release it over 2000 days at a rate that decreases every 200 days.

The purpose of the revenue is to provide funds for re-investment into the LINK ecosystem and to provide profit for the developer.

The block reward is the same as Ethereum. 5 LINK per block / 30k LINK per day.

### Issuance rate

### Cumulative issuance

### Post-revenue inflation rate

### Revenue live status

## Configuration

Network ID: 76

Chain ID: 76

Port: 30313

RPC Port: 8645

WS Port: 8646

RPC TLS Port: 8647

```
git clone https://github.com/link-blockchain/link-blockchain.git
```

LINK Blockchain can be synchronized with either Geth or Parity:

### Geth

Install Geth as described here: https://github.com/ethereum/go-ethereum/wiki/Building-Ethereum

```
geth --config link.toml --datadir ~/.link-geth init genesis.json
geth --config link.toml --datadir ~/.link-geth --rpc
# In a separate terminal launch the console.
geth attach ~/.link-geth/geth.ipc
```

### Parity

Install Parity as described here: https://parity.io/parity.html

```
parity --chain link.json --port 30313 --jsonrpc-port 8645 --geth
```

### Netstats

To have your node listed on http://stats.link-blockchain.org/ do the following (requires Node.js):

```
git clone https://github.com/cubedro/eth-net-intelligence-api
cd eth-net-intelligence-api
npm install
sudo npm install pm2 -g
```

Edit app.json:

```
     {
       "NODE_ENV"        : "production",
       "RPC_HOST"        : "localhost",
-      "RPC_PORT"        : "8545",
-      "LISTENING_PORT"  : "30303",
-      "INSTANCE_NAME"   : "",
+      "RPC_PORT"        : "8645",
+      "LISTENING_PORT"  : "30313",
+      "INSTANCE_NAME"   : "MY_NODE_NAME",      // Customize this.
       "CONTACT_DETAILS" : "",
-      "WS_SERVER"       : "wss://rpc.ethstats.net",
-      "WS_SECRET"       : "see http://forum.ethereum.org/discussion/2112/how-to-add-
↪yourself-to-the-stats-dashboard-its-not-automatic",
+      "WS_SERVER"       : "wss://stats.link-blockchain.org",
+      "WS_SECRET"       : "welcometothelinkedworld",
       "VERBOSITY"       : 2
     }
```

Then:

```
pm2 start app.json
```

## Filter bubbles

500 million tweets are published every day, but if you go to Twitter you will only see a small subset of these. Every platform has its own filter system to decide which content you see.

On Twitter you see the latest tweets from people you follow (minus those who have been banned), some sponsored tweets, with a "While you were away..." section at the top generated by unknown means. People are sometimes banned entirely from the platform.

On Reddit, you see content from subreddits you are subscribed to. Typically ordered by an opaque algorithm called "hot". Users can vote, but there is no real way to see how this affects what content is displayed. Large sections of the site will sometimes disappear.

On Medium content creators are sometimes ordered to change their content under threat of it being removed altogether.

Each of these communities is siloed from each other. They control the filter bubble within their platform.

LINK is fully programmable at every level. Anyone can take existing content on the system and develop new ways to select which content each user should see, or even utilize data from another filter bubble, for example analysing existing votes in a new way.

## Light client

Currently to interact with LINK in a fully decentralized way you need to have a fully syncronized node running locally. It can take a long time to sync and will take up a lot of storage space.

This will all change with Ethereum's light client functionaity that is rapidly nearing maturity. Both Geth and Parity will be able to serve light clients and act as light clients themselves.

It will also be possible for Android and iOS devices to have light client functionality.

# Smart contracts

## Revenue

Source

Deployment address: `0x97c7f4f8f0bbf384578a9f5754ae73f37ff49ec2`

Solidity 0.4.10 with optimizations.

See *Issuance* for more information.

## Item Store

Item Store is the principle smart contract system for LINK. All base content is registered with Item Store. IPFS is used as the storage layer.

### Useful links

- Source code
- Issue tracker

### Properties

Item Store has the following properties:

- **Timestamped** The approximate time that every item is published is recorded on the blockchain.
- **World-readable** Every item published can be read by anyone. The only way to avoid this would be to encrypt the item before publishing it.
- **Immutable** While an item can be "retracted", it can never really be deleted because the transaction that created it will be archived for eternity on full nodes.
- **Revisioned** Item Store has a rudimentary revisioning system built-in where an item can have multiple revisions, e.g. for editing posts. More sophisticated revisioning systems can be built on top of Item Store where each item is a revision.
- **Ownership** Each item can have an owner. Only the owner can modify an item, change item settings, or transfer ownership to another address.
- **Configurable** Each item has the following flags that can be set:
  - **Updatable** The contents of the item can be changed. Once disabled this flag cannot be re-enabled.
  - **Enfore Revisions** When updating the item a new revision must be created. It is not possible to retract revisions. Once enabled, this flag cannot be disabled.
  - **Retractable** The item in its entirety can be retracted. This is unaffected by Enforce Revisions. The itemId of a retracted item can never be used again. Once disabled this flag cannot be re-enabled.
  - **Transferable** The item can be transfered to another user (if they accept it), or disowned completely. Once disabled this flag cannot be re-enabled. At creation time items can also be flagged as anonymous to not have an owner associated. An alternative to transferable items is to use a proxy account with transferable ownership as the item owner.

- **Scalable** Only a bare-minimum of information is stored in contract state. IPFS is used for actual content storage.
- **Upgradability** Item Store is an upgradable system. Due to a security vulnerability, new storage system, or gas cost improvements a new Item Store contract may be deployed.
- **Unit tests** Item Store has tests written in Solidity using the Dapp framework.

#### itemId

Each item has a 20 byte itemId that is unique to the contract that generated it. The itemId can be further classified to indicate which contract the item is on.

- **Off-chain** There is considered to be an ordered list of Item Store contracts. For example, the first contract would be #0. The next one would be #1. 20 byte itemIds can be prefixed to indicate which contract the item is in, or this can be assumed from context.

  Client software should have a hard coded whitelist of Item Store contracts that its knows how to read from.
- **On-chain** New Item Store contracts register with the Item Store registration contract. When an itemId is stored in a smart contract, it must be stored with the 12 byte Item Store contractId. This way the smart contract can look up the address of any Item Store contract in the registration contract. This is essential to future-proof smart contracts that need to communicate with Item Store contracts.

#### Deployments

All current deployments of Item Store contracts on LINK were compiled with Solidity 0.4.16 with optimizations enabled.

Git hash: `b517451720bcb59d8f731974dfebf80d2ade57d1` link

ItemStoreRegistry contract address: `0x3ea61ec502f053ee89b906fdc3abe2e5f9eb4a7f`

#### Item Store #0

IPFS with SHA256 hash.

contract address: `0x2af783f7d5a607098057ce715b55a82e707a7cab`

BlobStore contractId: `0xb639900808b2633054720f33`

### Mixin Registry

Source

Deployment address: `0x99be57fca49270b69306cdbe89df416d769dbd06`

Solidity 0.4.16 with optimizations.

Every item of content stored on LINK is composed of one or more mixins. Examples of potential content types are tweets, comments, blog posts, media metadata and user profiles.

LINK has a hierarchical system of mixins. There is a root mixin that all other mixins are derived from.

Each mixin is identified by an integer, the mixinId. The mixinId for each mixin is issued by the Mixin Registry smart contract. The mixinId of the root mixin is 0.

When creating a new mixin, the addMixin() method must be called. `parent` must be the mixinId of the mixin that is being extending. `uri` should be a link to an immuntable commit in a repo that is derived from the parent mixin.

# Obtain LINK

In order to publish content on the LINK network, or execute any kind of transaction, you must spend some LINK. LINK can be obtained either by mining it or by purchasing it.

You need to have a LINK account. If you do not have one already, go to the LINK console and run `personal.newAccount();`.

Alternatively an Ethereum account can be created with MyEtherWallet and that can be used for LINK.

## Mine LINK

### CPU Mining

CPU mining with Geth is very easy. Run the regular Geth command, but add the `--mine` option. Use `--etherbase` to specify which account will receive the mining rewards. It defaults to using all processor cores. This can be changed using the `--minerthreads` option.

### GPU Mining

### Genoil

Ubuntu 15.10 or Newer. OpenCL only (for AMD cards)

```
sudo apt-get update
sudo apt-get -y install software-properties-common
sudo add-apt-repository -y ppa:ethereum/ethereum
sudo apt-get update
sudo apt-get install git cmake libcryptopp-dev libleveldb-dev libjsoncpp-dev␣
↪libjsonrpccpp-dev libboost-all-dev libgmp-dev libreadline-dev libcurl4-gnutls-dev␣
↪ocl-icd-libopencl1 opencl-headers mesa-common-dev libmicrohttpd-dev build-essential␣
↪-y
git clone https://github.com/Genoil/cpp-ethereum/
cd cpp-ethereum/
mkdir build
cd build
cmake -DBUNDLE=miner ..
make -j8
```

Ubuntu 15.10 or Newer. OpenCL + CUDA (for NVIDIA cards)

```
wget http://developer.download.nvidia.com/compute/cuda/repos/ubuntu1404/x86_64/cuda-
↪repo-ubuntu1404_7.5-18_amd64.deb
sudo dpkg -i cuda-repo-ubuntu1404_7.5-18_amd64.deb
sudo apt-get -y install software-properties-common
sudo add-apt-repository -y ppa:ethereum/ethereum
sudo apt-get update
sudo apt-get install git cmake libcryptopp-dev libleveldb-dev libjsoncpp-dev␣
↪libjsonrpccpp-dev libboost-all-dev libgmp-dev libreadline-dev libcurl4-gnutls-dev␣
↪ocl-icd-libopencl1 opencl-headers mesa-common-dev libmicrohttpd-dev build-essential␣
↪cuda -y
```

```
git clone https://github.com/Genoil/cpp-ethereum/
cd cpp-ethereum/
mkdir build
cd build
cmake -DBUNDLE=cudaminer ..
make -j8
```

CD to the `ethminer` subfolder and run the following command

```
./ethminer -G -F http://localhost:8645
```

#### Claymore

Download from https://github.com/nanopool/Claymore-Dual-Miner/releases

Linux

```
./ethdcrminer64 -epool http://localhost:8645 -allcoins exp
```

Windows

```
EthDcrMiner64.exe -epool http://localhost:8645 -allcoins exp
```

Here is a Claymore on Windows LINK mining tutorial: https://klmoney.wordpress.com/link-blockchain-windows-gpu-mining/

### Purchase LINK

LINK is not yet trading on any exchange.

If you wish to purchase LINK please email purchase@link-blockchain.org. It is currently available at a price of 0.04 USD per LINK. This price is subject to change at any time. Payment can be made via any cryptocurrency.

## Events

### Upcoming

none

### Previous

#### LINK Blockchain: The successor to the World Wide Web

Thursday, August 17, 20:00 - 23:00
Care Hub Cafe
30 Bà Huyn Thanh Quan
H Chí Minh City, Vietnam

### Dojo Bali Cryptocurrency Mastermind (Dojo members only)

Tuesday, August 1, 19:00 - 21:00
Dojo Bali Coworking Space
No. 88 Jalan Batu Mejan
Canggu, Echo Beach
Kuta Utara, Kabupaten Badung
Bali, Indonesia

### Kuala Lumpur Blockchain Evening - video

Friday, July 14, 19:00 - 22:00
Einstein Boardroom
WORQ
Level 3A, Glo Damansara
No. 699, Jalan Damansara
Kuala Lumpur

### Bangkok Satoshi Square

Monday, June 26, 18:30 - 21:30
American Bar & Grill
4/26 Sukhumvit Soi 8
Bangkok

## Contact

If you have any question whatsoever, please email contact@link-blockchain.org

CHAPTER 3

# Indices and tables

- genindex
- modindex
- search