

DECENTRALISED APPLICATIONS

ON

TURTLENETWORK



POWERED BY TURTLENETWORK

REVISION HISTORY

Revision History		
Revision	Date	Description
1	12/4/2020	First revision
2	12/7/2020	Some corrections

CONTENTS

REVISION HISTORY 2

ACKNOWLEDGEMENT 3

WHAT IS A DAPP? 4

LANGUAGE FOR DAPPS ON
TURTLENETWORK 4

HOW TO DEPLOY MY DAPP ON
TURTLENETWORK BLOCKCHAIN 4

A DAPP SCRIPT EXAMPLE 11

ACKNOWLEDGEMENT

*First and foremost, I would like to sincerely thank **TurtleNetwork** team for their wonderful work that brought **TurtleNetwork** into existence a few years ago, and most importantly their positive encouragement and a warm spirit to help us in almost every thing related to the blockchain technology.*

*I also want to extend my thanks to **Polarity** team for their valuable contribution in my humble guide, and for their effort to bridge the the gap between the blockchain technology and the real world usage.*

*Last but not least, I would like also to thank our wonderful **TurtleNetwork community** for their continuous support.*

WHAT IS A DAPP?

DApp refers to a decentralized application or distributed application. Basically, it looks like your application downloaded from App Store or Google Play, except it is a decentralized entity built on top of a particular blockchain or any other peer-to-peer (P2P) network. Decentralization is the core concept of the blockchain industry. In the dApp context, it means that no one in the entire world can manage or control a dApp.

Traditional application

Frontend (the visual appearance of an application) – API – centralized database

Decentralized application

Frontend (the visual appearance of an application) – smart contract – blockchain

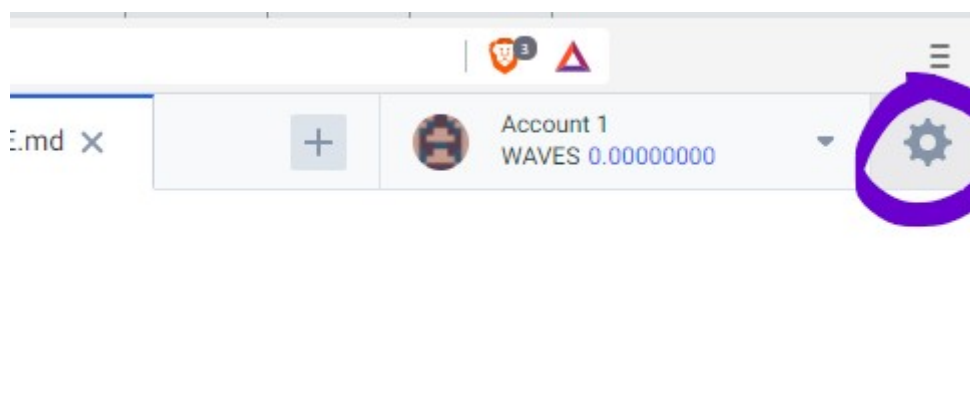
LANGUAGE FOR DAPPS ON TURTLENETWORK

To develop Dapps on the TurtleNetwork blockchain, we use **RIDE** language. RIDE is a blockchain scripting language, which enables **'smart'** blockchain transactions. The execution result is predicated on certain logic, realised using RIDE scripts and deployed on the blockchain. The goal of RIDE's architecture is to create **a native on-chain computation layer** which is as close to the general blockchain architecture (full data synchronisation) as possible.

HOW TO WRITE AND PUBLISH A DAPP ON TOP OF THE TURTLENETWORK

To write and publish your first dapp on Turtlenetwork blockchain, use the following steps :

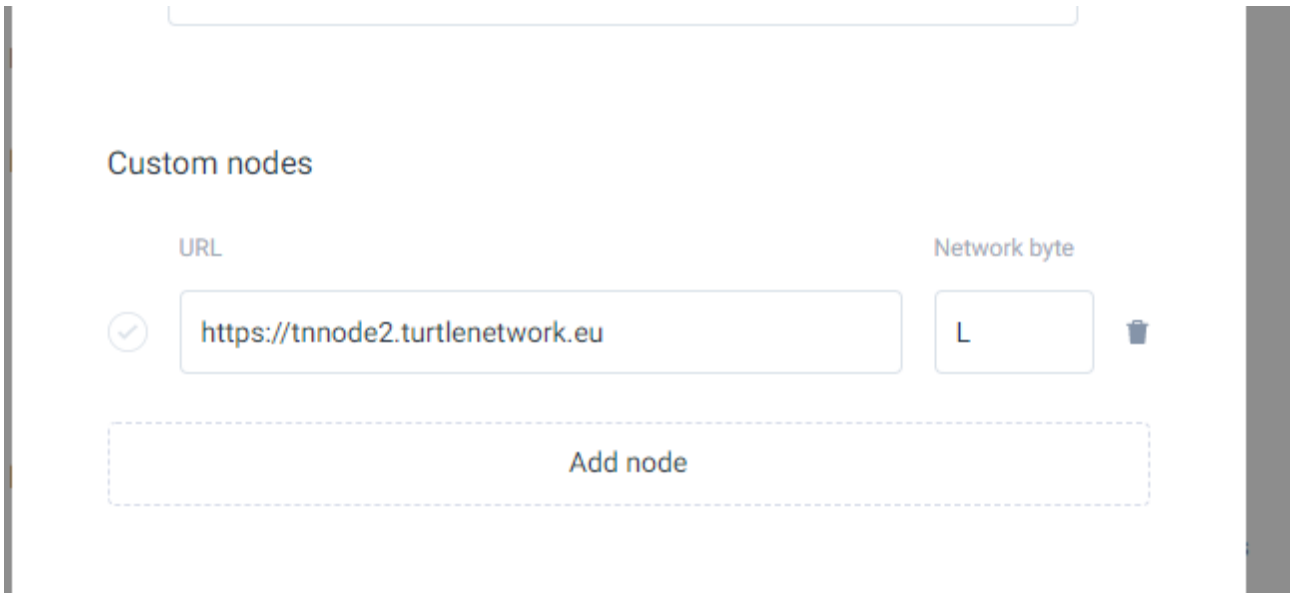
- Open the online waves IDE: <https://waves-ide.com/>
- Click on the setting button as shown in the pic bellow :



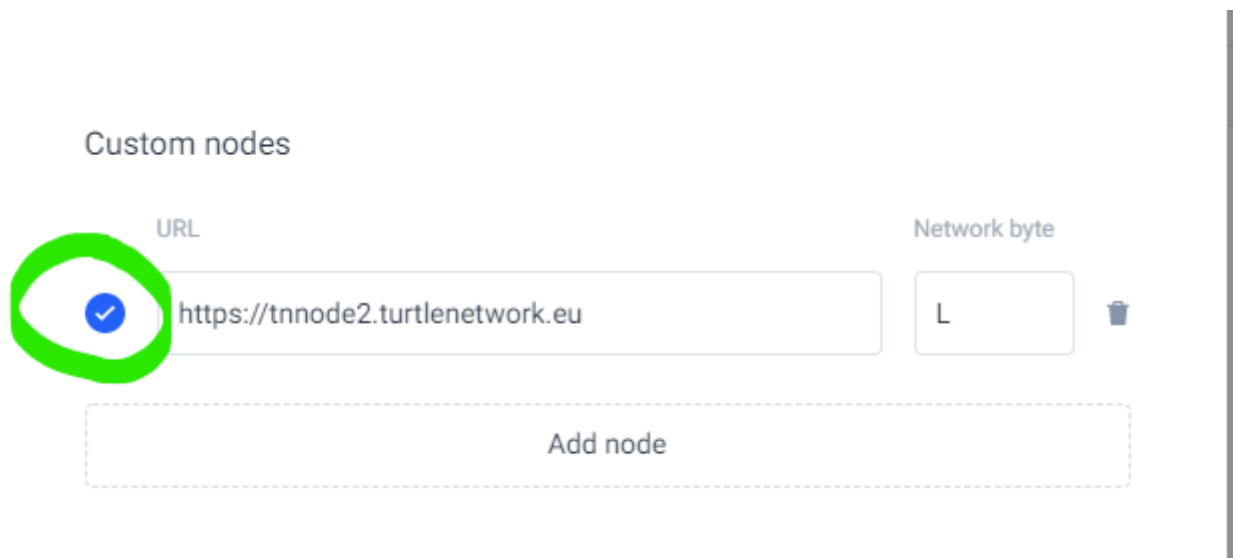
- Enter the TurtleNetwork custom node as shown on the pic below :

URL : <https://tnnode2.turtlenetwork.eu>

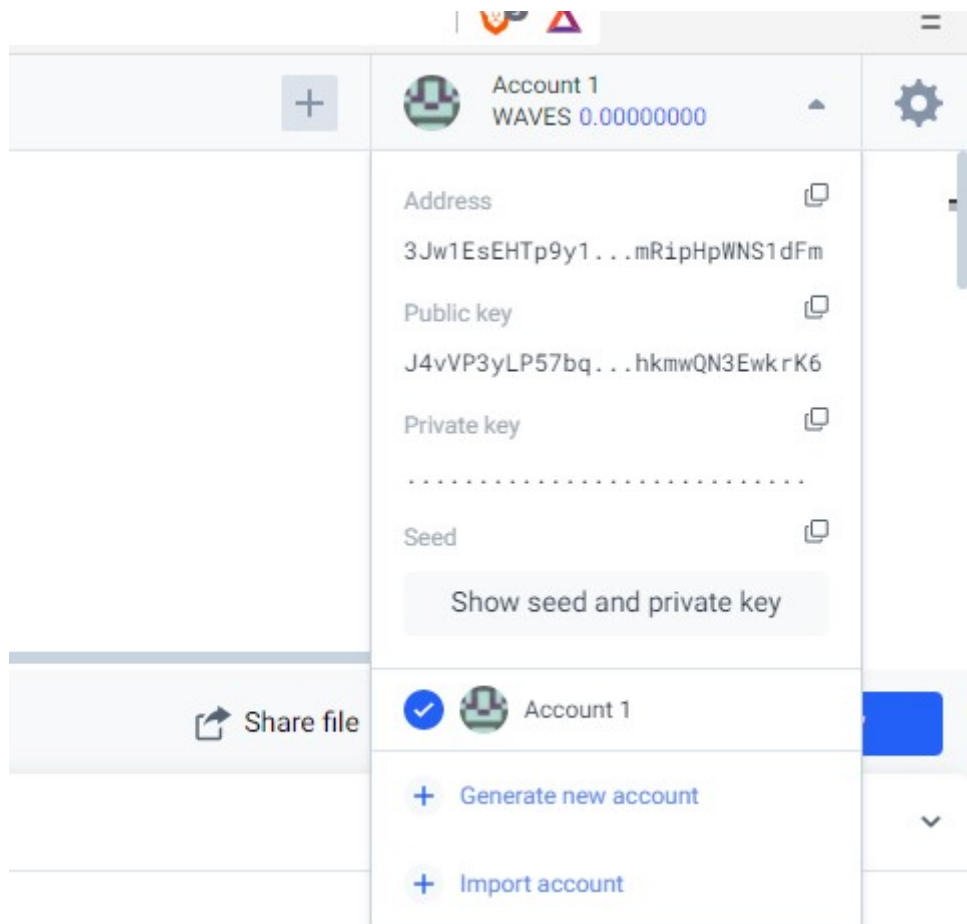
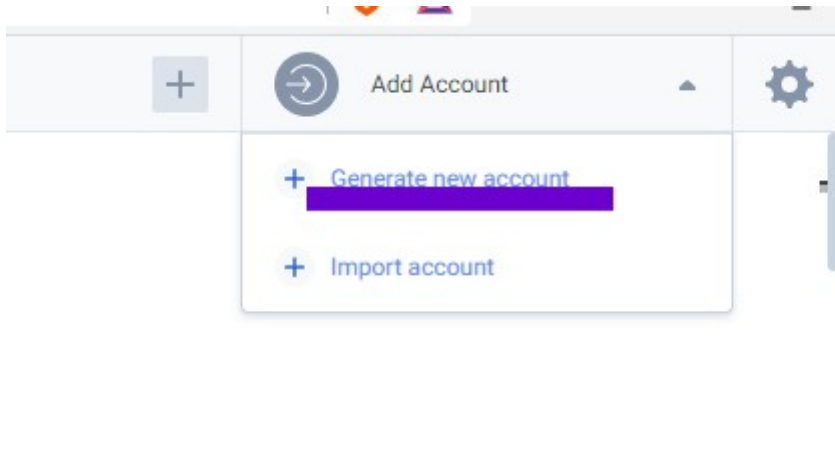
Network byte : L



- Then activate it as your default custom node as shown below :



- After setting up the custom node, you will have to generate a new Turtlenetwork account as shown below and fill it with some TN (minimum 1 TN to cover the publishing fee)



- Then click on add sign :

Sign and publish ✕

```
1 {
2   "type": 13,
3   "version": 2,
4   "chainId": 76,
5   "fee": 1000000,
6   "timestamp": 1607124198228,
7   "proofs": [],
8   "script": "base64:AAIDAAAAAAAAAYIARIAEgAAAAACAQAAApyYW5kb21pemVyAAAAAQAAAApbnYEAAAACGxhc3RQbGF5BAAA
9 }
```

Sign with: IDE Account ▼ Account: Account 1 ▼ Proof index: 1 ▼ **+ Add sign**

- Before publishing your dapp, change version 2 to 1 and fee 1000000 to 100000000

~~version 2~~ : **version 1** and ~~fee 1000000~~ : **fee 100000000**

Sign and publish
✕

```

1  {
2    "type": 13,
3    "version": 2,
4    "senderPublicKey": "J4vVP3yLP57bqCEXiVJUQNaW4qw6HShhkMwQN3EwkrK6",
5    "chainId": 76,
6    "fee": 1000000,
7    "timestamp": 1607124198228,
8    "proofs": [
9      "4T8bitJtVv9dbPAEBzG8X6ZtwFK9HZLpR6trJU2LD53nrbkooM4GxVg9ur5bXrkuDwghuJHc5WmHwJjrbuoknPur"
10   ],
11   "id": "GETK5GGp6DXh9UiJ4zzKN4YV1pzUVYX1iBJMwaupAUwG",
12   "script": "base64:AAIDAAAAAAAAAYIARIAEgAAAAACAQAAAApyYw5kb21pemVyAAAAAQAAAAAnpbnYEAACGxhc3RQbGF5BAAA
13 }
                
```

Sign with	Account	Proof index	
IDE Account ▾	Account 1 ▾	2 ▾	✔ Sign added


- At the end click on publish button to put your dapp on the blockchain :

```
1 {
2   "type": 13,
3   "version": 1,
4   "senderPublicKey": "J4vVP3yLP57bqCEXiVJUQNaW4qw6HShhkmwQN3EwkrK6",
5   "chainId": 76,
6   "fee": 10000000,
7   "timestamp": 1607190291434,
8   "proofs": [
9     "4caMys1yuCvS4gCNFsGaMxsVWRDaVnyGotyco5TeYnyEE8ejAPZ4zfme3jPzZzyakVyjin9C2mtmg91WKKtEjcs"
10  ],
11  "id": "3ib19DE7VESz8Y7NrtVDfriTQmNYQe3ULqKLFhmTf6YF",
12  "script": "base64:AAIEAAAAAAAAAAQIAhIAAAAAAAAAAAEAAAABaQEAAAAEY2FsAAAAAAAAAAAAAABWFzc2V0CQAEQwAAAAcAAAA
13 }
```

Sign with: IDE Account | Account: Account 1 | Proof index: 2 | + Add sign

Cancel | **Publish**

SUCCESS

 Tx has been sent. ID:
6RtsQqKo1bbKWDUd6BPRz5TQ4Q8d1grPTU
wmqfXAnr8q

Waves IDE on GitHub

Congratulations !
Your dapp is on the blockchain now,

A DAPP SCRIPT EXAMPLE

here is how a dapp script looks like :

```
{-# STDLIB_VERSION 4 #-}
{-# CONTENT_TYPE DAPP #-}
{-# SCRIPT_TYPE ACCOUNT #-}

@Callable(i)
func call() = {
  let asset = Issue("Asset", "", 1, 0, true, unit, 0)
  let assetId = asset.calculateAssetId()

  # Script execution results
  # More details in docs: https://docs.wavesplatform.com/en/ride/functions/callable-function#callable-functions-in-standard-library-v4
  [
    BinaryEntry("bin", base58'), # base16, base58, base64 or any other ByteVector values
    BooleanEntry("bool", true),
    IntegerEntry("int", 1),
    StringEntry("str", ""),
    DeleteEntry("str"),
    asset,
    Reissue(assetId, 1, false),
    Burn(assetId, 1),
    ScriptTransfer(i.caller, 1, assetId)
  ]
}

@Verifier(tx)
func verify() = sigVerify(tx.bodyBytes, tx.proofs[0], tx.senderPublicKey)
```



Do not use your own TN wallet to publish a Dapp, you might lose your assets, use a separate account for that !