Ether.js Course

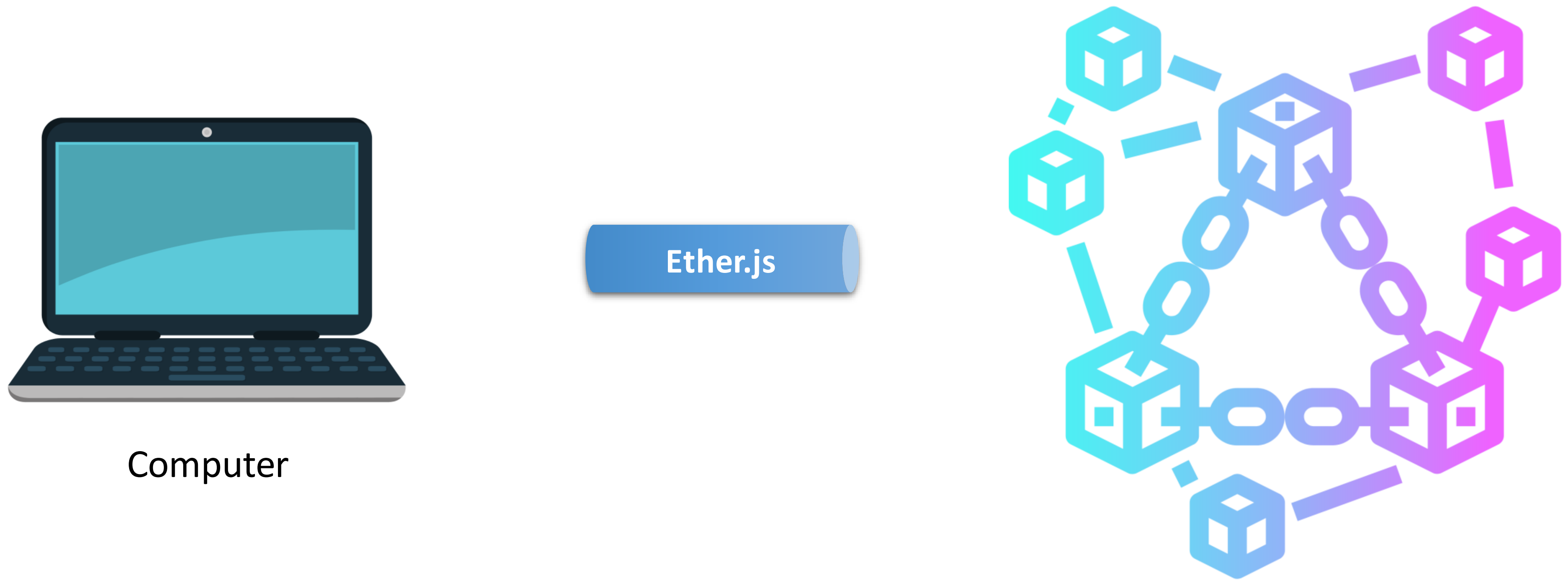
By Code Eater

Ether.js

- What Is Ether.js ?
- Common Terminologies
- Modules
- Quizes
- Interaction With Blockchain
- Interaction With Smart Contract
- Web3.js vs Ether.js

Ether.js

- The ethers.js library aims to be a complete and compact library for interacting with the Ethereum Blockchain and its ecosystem.



Common Terminology

Provider	A Provider (in ethers) is a class which provides an abstraction for a connection to the Ethereum Network. It provides read-only access to the Blockchain and its status.
Signer	A Signer is a class which (usually) in some way directly or indirectly has access to a private key, which can sign messages and transactions to authorize the network to charge your account ether to perform operations.
Contract	A Contract is an abstraction which represents a connection to a specific contract on the Ethereum Network, so that applications can use it like a normal JavaScript object.

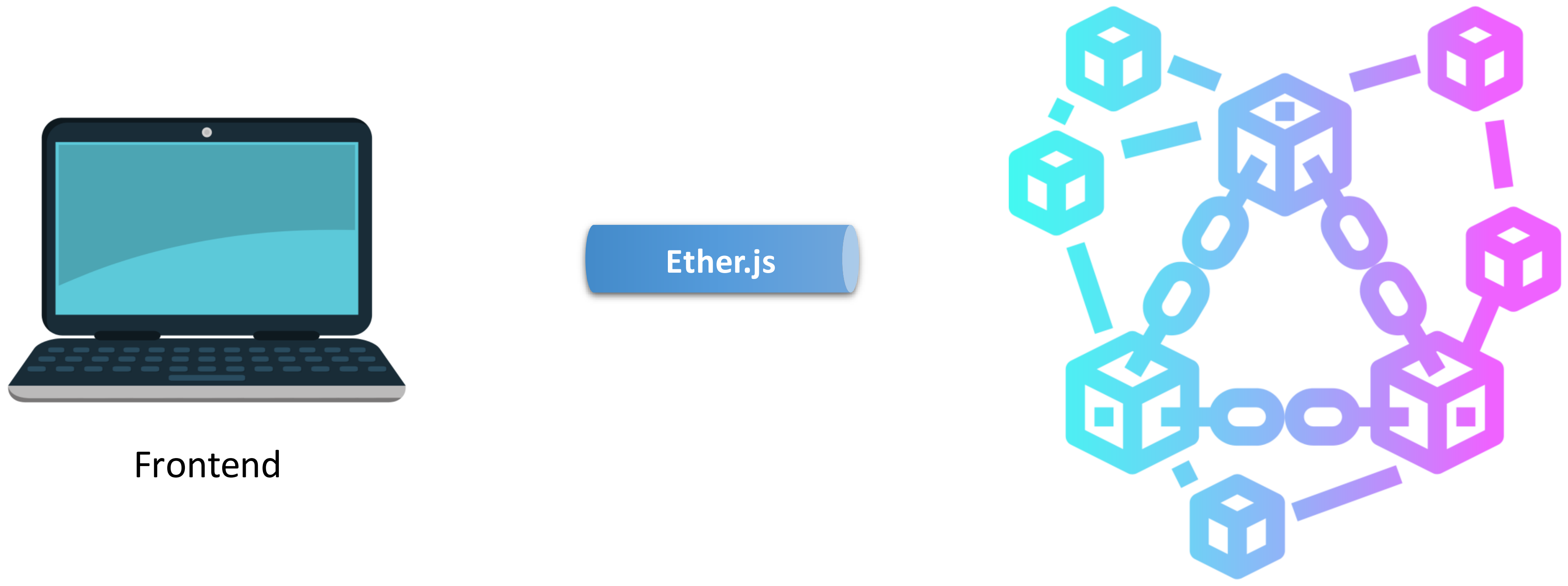
Installation

- npm install --save ethers

Connecting To Blockchain

Ether.js

- The ethers.js library aims to be a complete and compact library for interacting with the Ethereum Blockchain and its ecosystem.



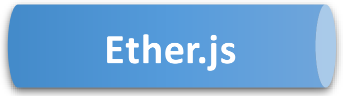
Connecting To Blockchain

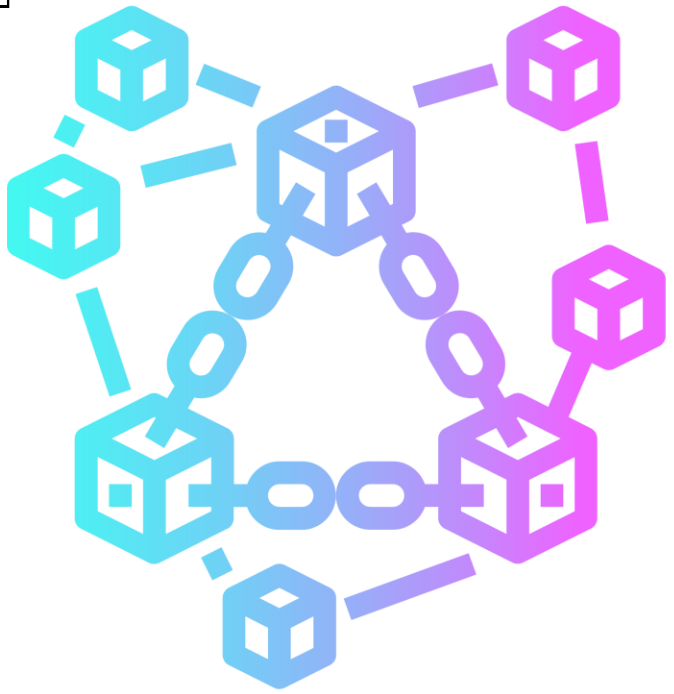
Computer

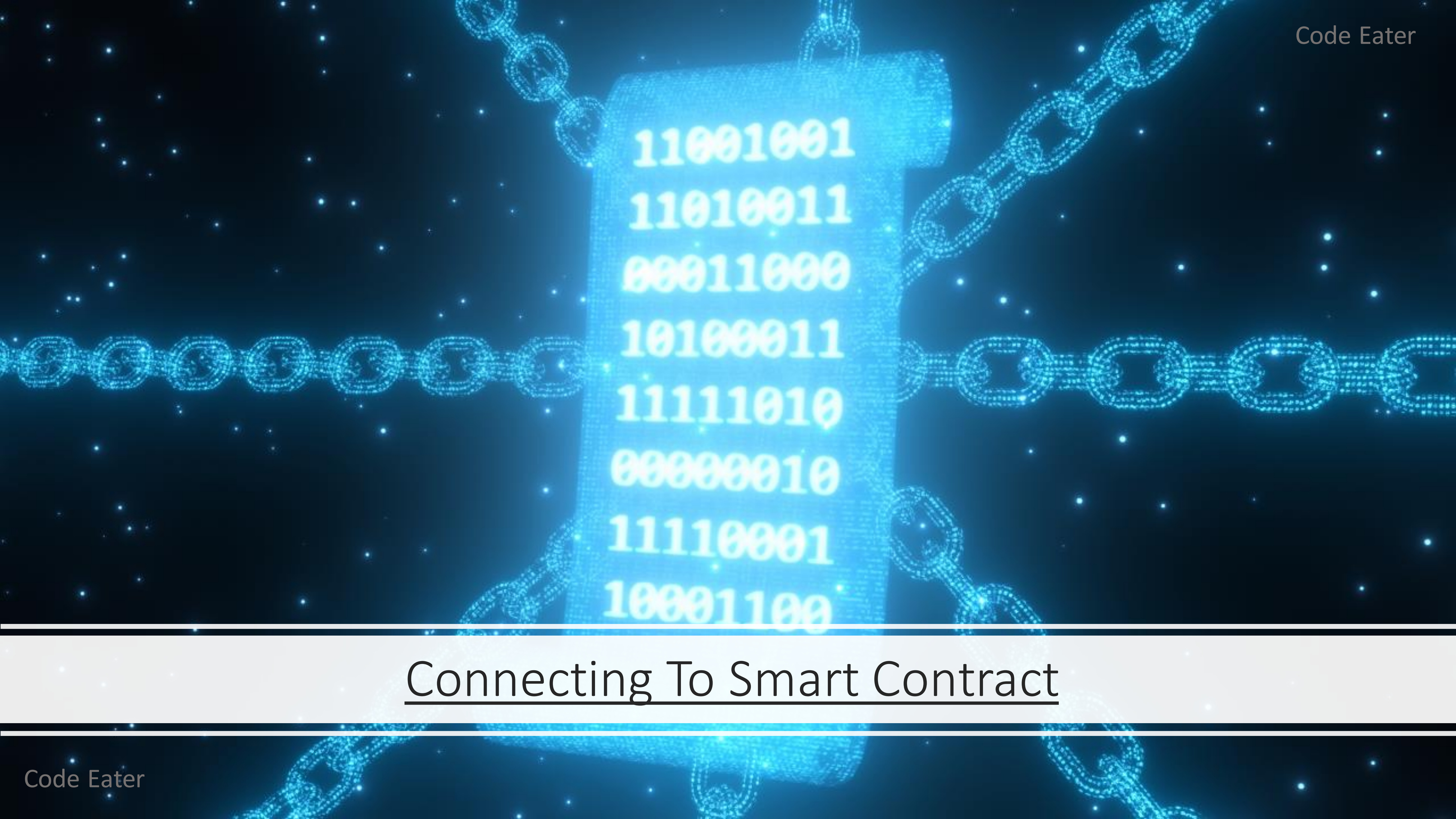
Connecting To Blockchain

Q1) Alternative Of Infura ?

Computer

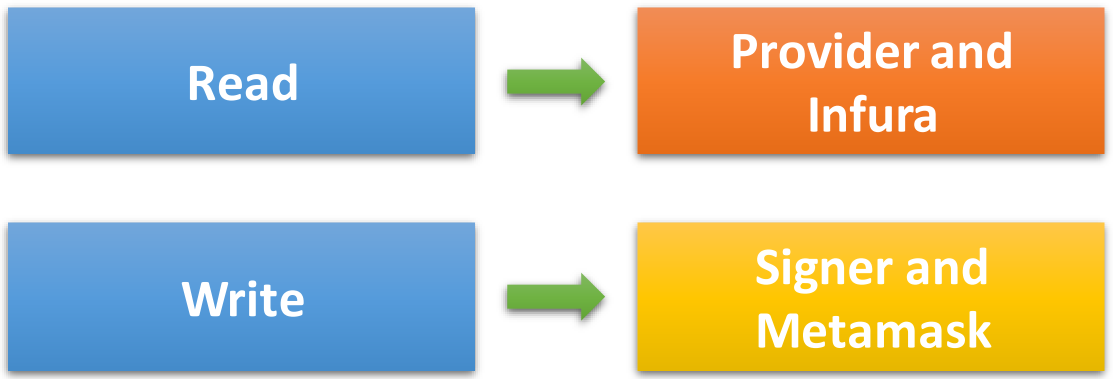




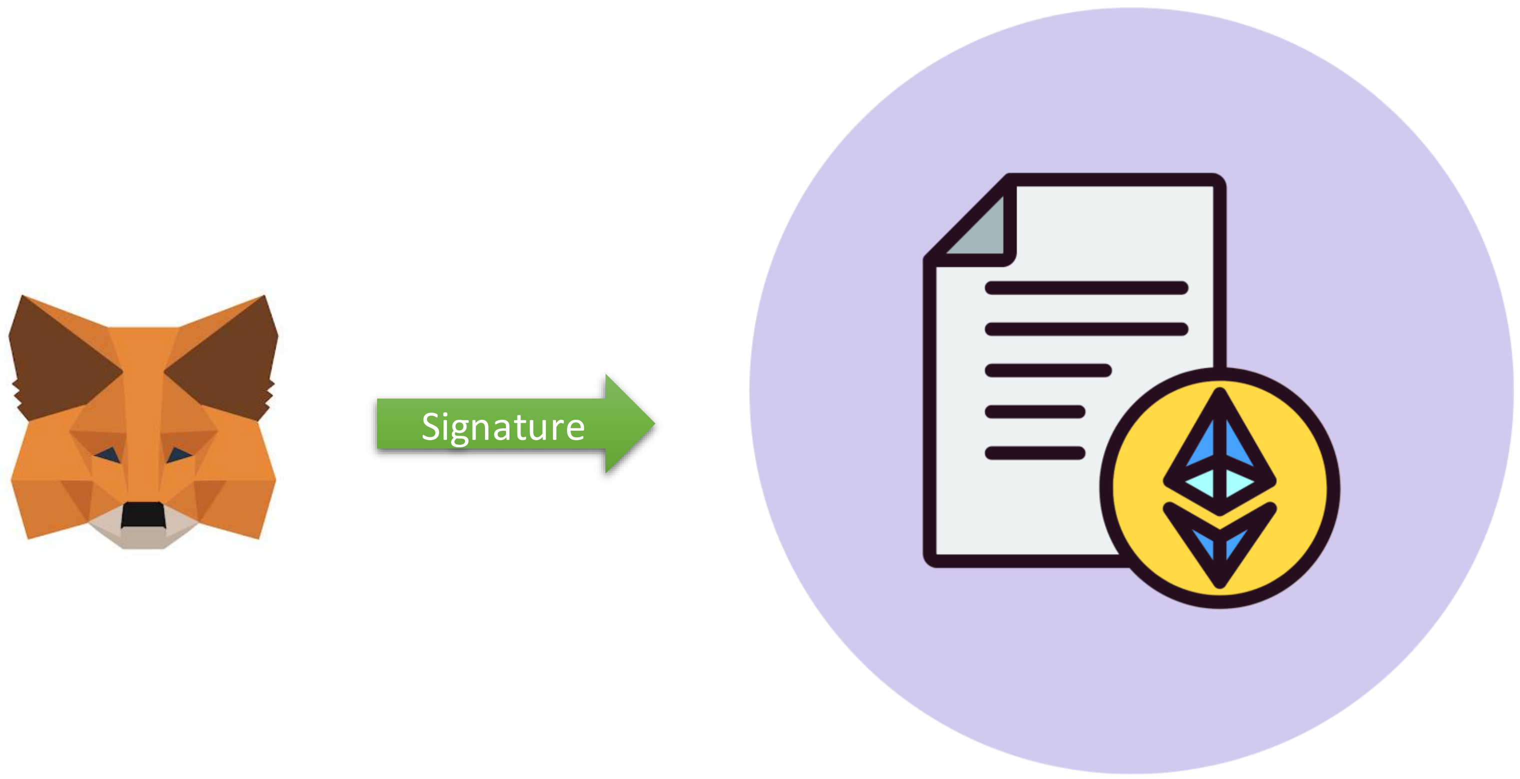


Connecting To Smart Contract

Two Operations

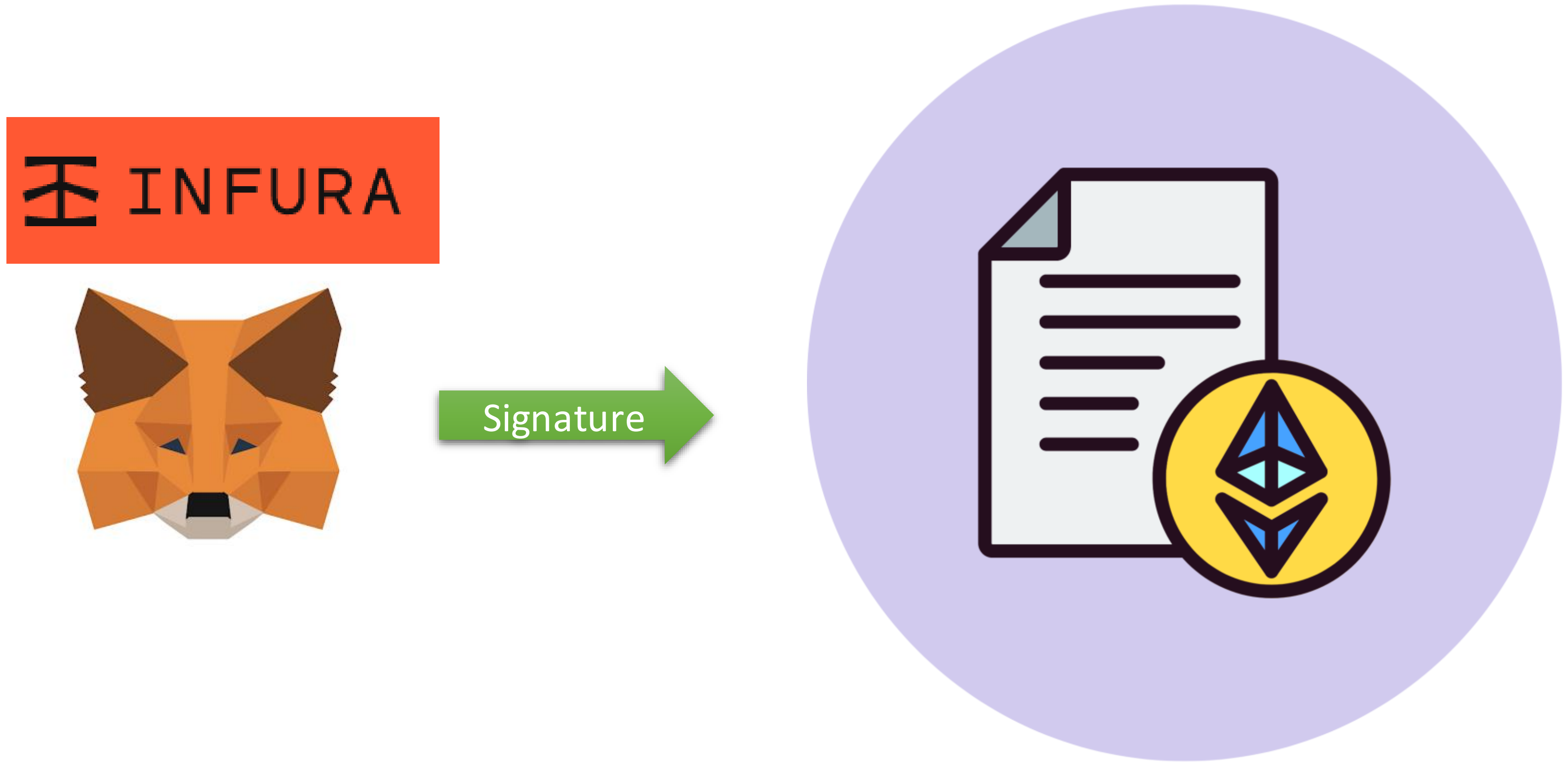


Signer and Metamask



Signer and Metamask

Q) How our computer will interact with Smart Contract without Infura?



Connecting To Metamask

FINAL MACTH

WEB3

VS

WEB3.JS

ETHER.JS

Advantages Of Ether.js Over Web3.js

- Documentation
- Less Bugs
- New Projects
- Well Maintained
- Less Size
- Developer Friendly

Congratulations

Thank You

Please Like and Subsribe :)

Instagram - @codeeater21

LinkedIn - @kshitijWeb3

Notes

Connecting to Ethereum: RPC

```
const provider = new ethers.providers.JsonRpcProvider();
```

If you don't specify a url, Ethers connects to the default Ganache (http://localhost:8545)

```
await provider.send("eth_requestAccounts", []);
```

MetaMask requires requesting permission to connect users accounts

```
const signer = provider.getSigner()
```

The provider also allows signing transactions to send ether and pay to change state within the blockchain. For this, we need the account signer.

Querying the Blockchain

await provider.getBlockNumber()

Look up the current block number

balance = await provider.getBalance("ethers.eth")

Get the balance of an account

ethers.utils.formatEther(balance)

Converts the balance in wei to ether

ethers.utils.parseEther("1.0")

If a user enters a string in an input field, you may need to convert it from ether (as a string) to wei (as a BigNumber)

Connecting to Ethereum: MetaMask

```
const provider = new ethers.providers.Web3Provider(window.ethereum)
```

A Web3Provider wraps a standard Web3 provider, which is what MetaMask injects as window.ethereum into each page.

```
await provider.send("eth_requestAccounts", []);
```

MetaMask requires requesting permission to connect users accounts