Decentralized Systems

Smart Contract (Standards for tokens)

Blockchain tokens

- **Digital representations of assets** that exist on a blockchain
  - can represent different assets, from cryptocurrencies to digital representations of physical assets, ownership rights, or even access to specific services
- **Secured through cryptographic techniques**, they are tamper-resistant and guarantee the integrity of the transactions and ownership
- Built thanks to standard **smart contracts**

ERC-20

ERC-20 Tokens (2015)

- ERC-20 tokens
  - blockchain-based **assets** that have value and can be sent and received
  - known as **fungible tokens**, ERC-20 tokens are **identical** from one another
  - **represent virtually anything**, e.g., lottery tickets, points on an online platform, skills for a character in a game, fiat currency, etc.

ERC-20 Tokens

- They are indistinguishable, interchangeable, divisible
- Can be **transferred** from accounts
- It is possible to
  - get the **current token balance** of an account
  - find out **how much** of a token's entire supply **is available**
  - know if **certain quantity** of tokens from one account **can be spent by a third-party account**

ERC-20 Tokens

Creating tokens

Handling transactions

Keeping track of balances

ERC-20 Tokens

Creating tokens

Handling transactions

Keeping track of balances

Be careful that buggy smart contracts, and therefore buggy tokens, once published will remain on the blockchain forever

ERC-20 Tokens

Ethereum Improvement Proposals

All Core Networking Interface ERC Meta Informational

Standards Track: ERC

ERC-20: Token Standard

Authors Fabian Vogelsteller <fabian@ethereum.org>, Vitalik Buterin <vitalik.buterin@ethereum.org>

Created 2015-11-19

Table of Contents

https://eips.ethereum.org/EIPS/eip-20

Simple Summary

A standard interface for tokens.

ERC-20 Tokens

- The ERC-20 standard implements an **API for tokens**

- **Optional**
  - name
  - symbol
  - decimals

- **Required**
  - totalSupply
  - balanceOf
  - transfer
  - transferFrom
  - approve
  - allowance

- **Not standard**
  - mint
  - burn

Only the owner can create (mint) or destroy (burn) tokens

ERC-20 Tokens

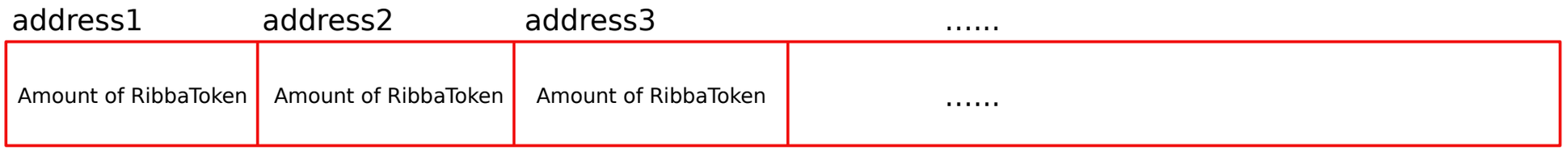
- Go to Etherscan https://etherscan.io/tokens
- Check some smart contract
- Even without any real ETH, you can make calls (Read Contract)

Build your token!

- RibbaToken: 
- Following the standard, a token has
  - A totalSupply (mandatory)
  - A name (optional), e.g. RibbaToken
  - A symbol (optional), e.g. RIB
  - A decimal value (optional)

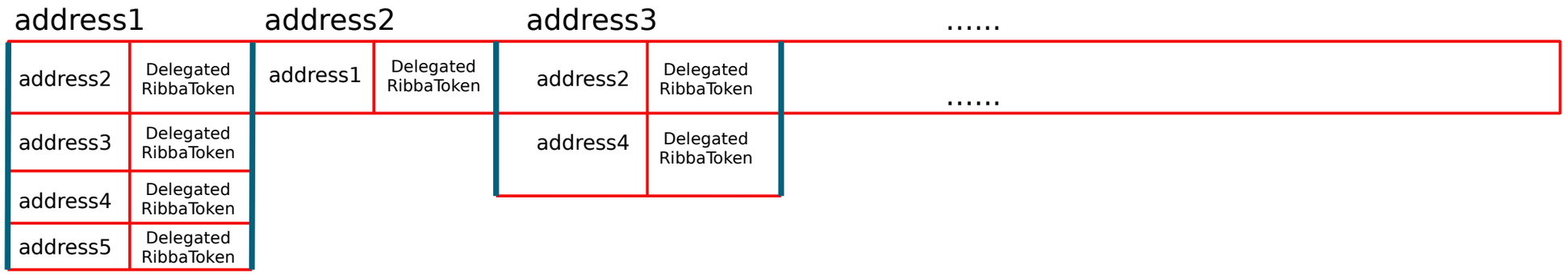
Build your token!

- RibbaToken:  Transfer
  - function **balanceOf**(address _owner) constant returns (uint256 balance);
  - function **transfer**(address _to, uint256 _value) returns (bool success);
  - event **Transfer**(address indexed _from, address indexed _to, uint256 _value);
  - balanceOf uses a state variable (mapping)



Build your token!

- RibbaToken:  Delegated transfer
  - function **approve**(address _spender, uint256 _value) returns (bool success);
  - function **transferFrom**(address _from, address _to, uint256 _value) returns (bool success);
  - function **allowance**(address _owner, address _spender) constant returns (uint256 remaining);
  - event **Approval**(address indexed _owner, address indexed _spender, uint256 _value);
  - allowance uses a state variable (mapping of mapping)

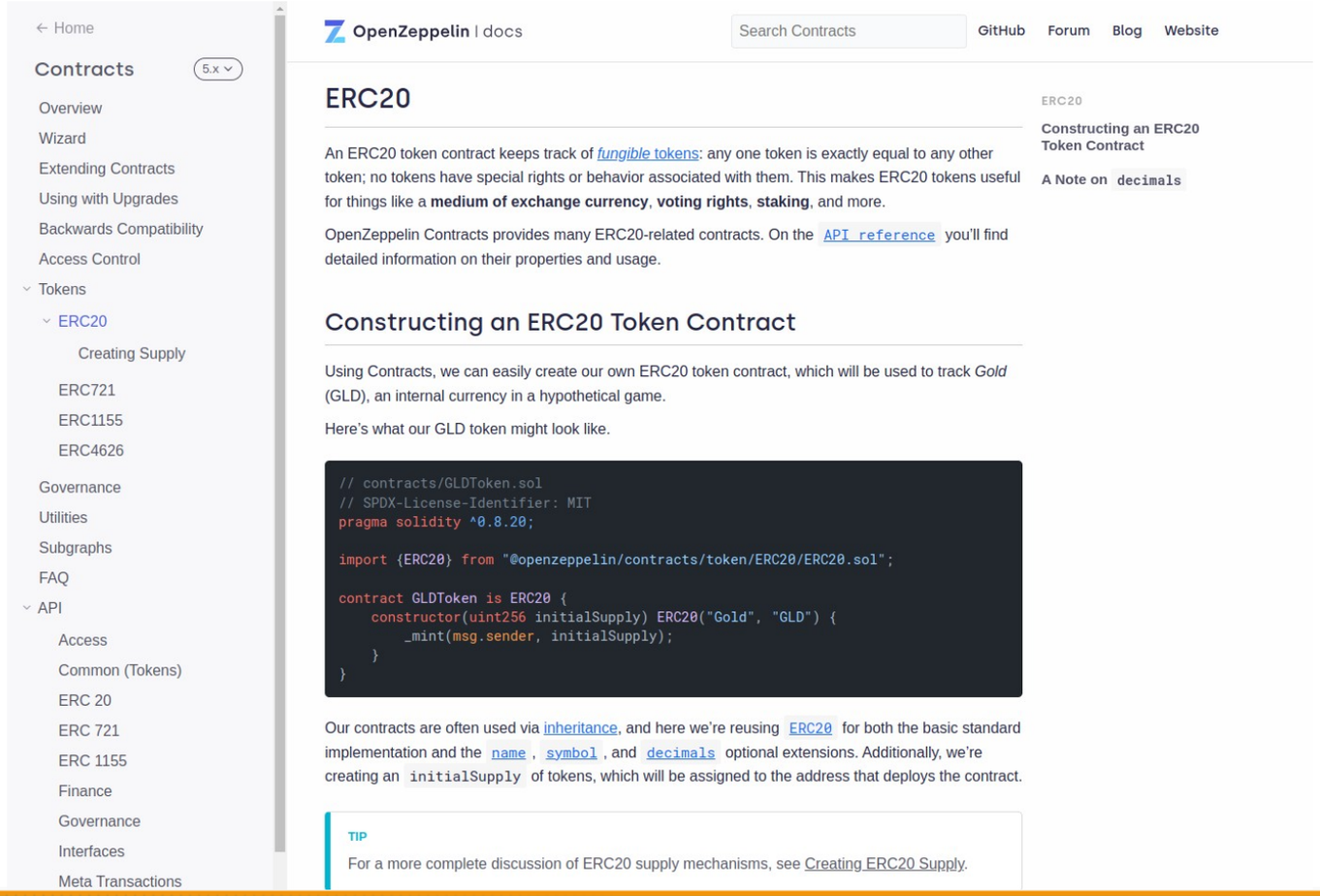


Build your token!

- RibbaToken:  Events
  - **event** Transfer(address indexed _from, address indexed _to, uint256 _value);
  - **emit** Transfer(msg.sender, _to, _value);
- Signal that “something” (a transaction) took place
- Stored in the blockchain, they are a good way of broadcasting relevant information
- Cannot be accessed by other smart contracts, but only from external applications

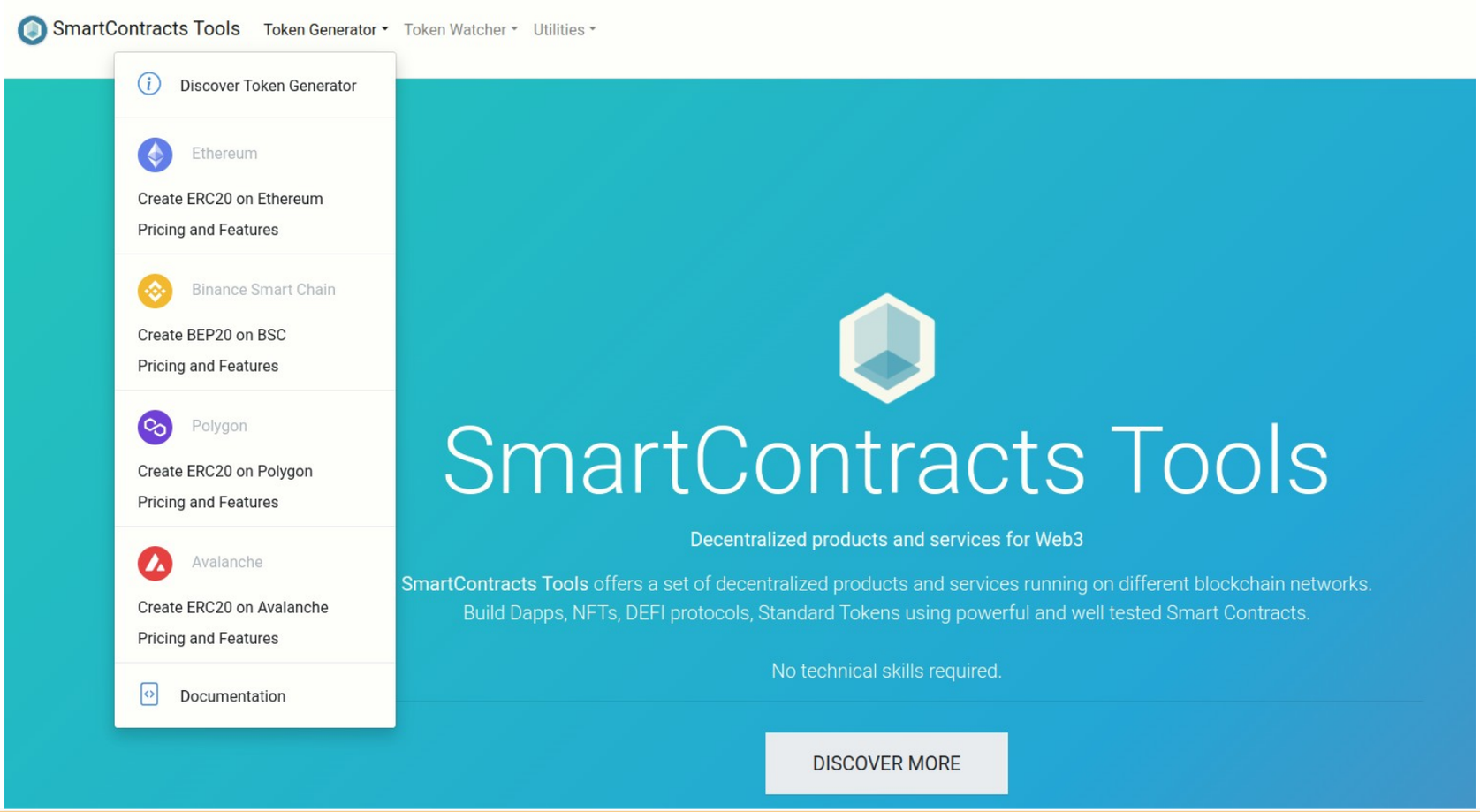
Build your token!

- You can use libraries, see for example: https://docs.openzeppelin.com/contracts/2.x/erc20



Build your token!

- Or online tokens generators, see for example: https://www.smartcontracts.tools/




ERC-721

ERC-721 Tokens (2018)

- Standard that describes how to build **non-fungible** or unique tokens (**NFT**), they are the **digital collectibles of the 21st century**
- NFT ownership is recorded on public blockchains, allowing **anybody to easily verify** their legitimacy and ownership at any moment
  - Defined in EIP-721: https://eips.ethereum.org/EIPS/eip-721

ERC-721 Tokens

- An NFT token represents **ownership of “something”**, for example an image
  - function ownerOf(uint256 _tokenId) external view returns (address);
- NFT files are too large to be stored directly on the blockchain
- They are stored “somewhere” else and the files’ locations (among other details) are stored in the NFT metadata

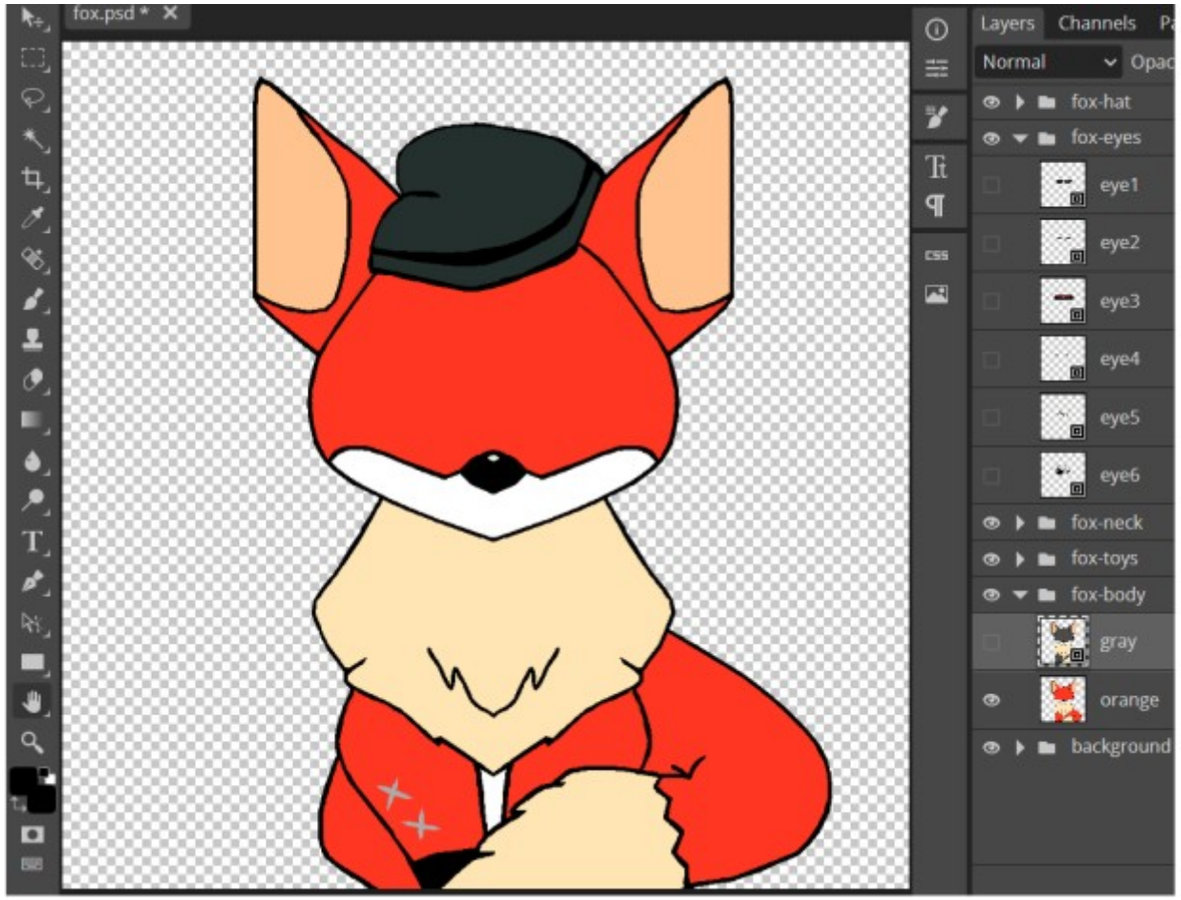
ERC-721 Tokens

- **Metadata** are **JSON documents** that contain various **information**
  - NFT’s name
  - description
  - link to the file
- **Traits**, like
  - background color, clothing, facial expressions, accessories
  - rarity levels like common, rare, epic, legendary
  - strength, speed, other powers

ERC-721 Tokens

- In generative NFT collections, traits are often combined algorithmically
  - A base set of traits is predefined and a computer program mixes and matches them to create thousands of unique NFTs, each with its own trait combination
- NFT metadata are the input of the smart contract
- Despite NFTs and metadata are **stored off-chain**, they are **minted** on-chain

Build your collection




```
const layerConfigurations = [
  {
    growEditionSizeTo: 3000,
    layersOrder: [
      { name: "Background" },
      { name: "fox-body" },
      { name: "fox-eyes" },
      { name: "fox-hat" },
      { name: "fox-toys" },
      { name: "fox-neck" },
    ],
  },
];
```

Build your collection


```
"attributes": [
  {
    "trait_type": "Background",
    "value": "bgcolor3"
  },
  {
    "trait_type": "fox-body",
    "value": "white"
  },
  {
    "trait_type": "fox-eyes",
    "value": "eye5"
  },
  {
    "trait_type": "fox-hat",
    "value": "hat6"
  },
  {
    "trait_type": "fox-neck",
    "value": "neck3"
  },
  {
    "trait_type": "fox-toys",
    "value": "toy1"
  }
],
```


ERC-1155 Tokens

- Token standards like ERC-20 and ERC-721 require a separate contract for each token type or collection
- With ERC-1155 it is possible to have **multiple tokens in the same contract** and also to add new ones later on
- **Less contracts** means **less space** in the blockchain (less gas)
- More complex to implement, are becoming more and more popular

ERC-1155 Tokens

- ERC-1155
  - **enables batch transfers**, which allow multiple tokens to be sent in a single transaction
  - **supports atomic swaps**, allowing users to trade one type of token for another in a single, trustless transaction
- Especially useful for gaming, where users might want to exchange one type of item for another, without relying on an intermediary

- See https://www.youtube.com/watch?v=FkUn86bH34M