Account Abstraction on Starknet

A glimpse into the future of crypto UX

Martín Triay
OpenZeppelin

Section 1

What is Account Abstraction anyway?

Smart Accounts

- Smart Contracts that behave as accounts
  - Verify transactions
  - Hold assets
  - Call other contracts

- Examples
  - Gnosis Safe
  - Argent
  - Instadapp

When a Smart Account can pay for a transaction, we call it Account Abstraction

That's basically it

Contracts that pay for transactions.

I could have said it earlier.

¯_(ツ)_/¯

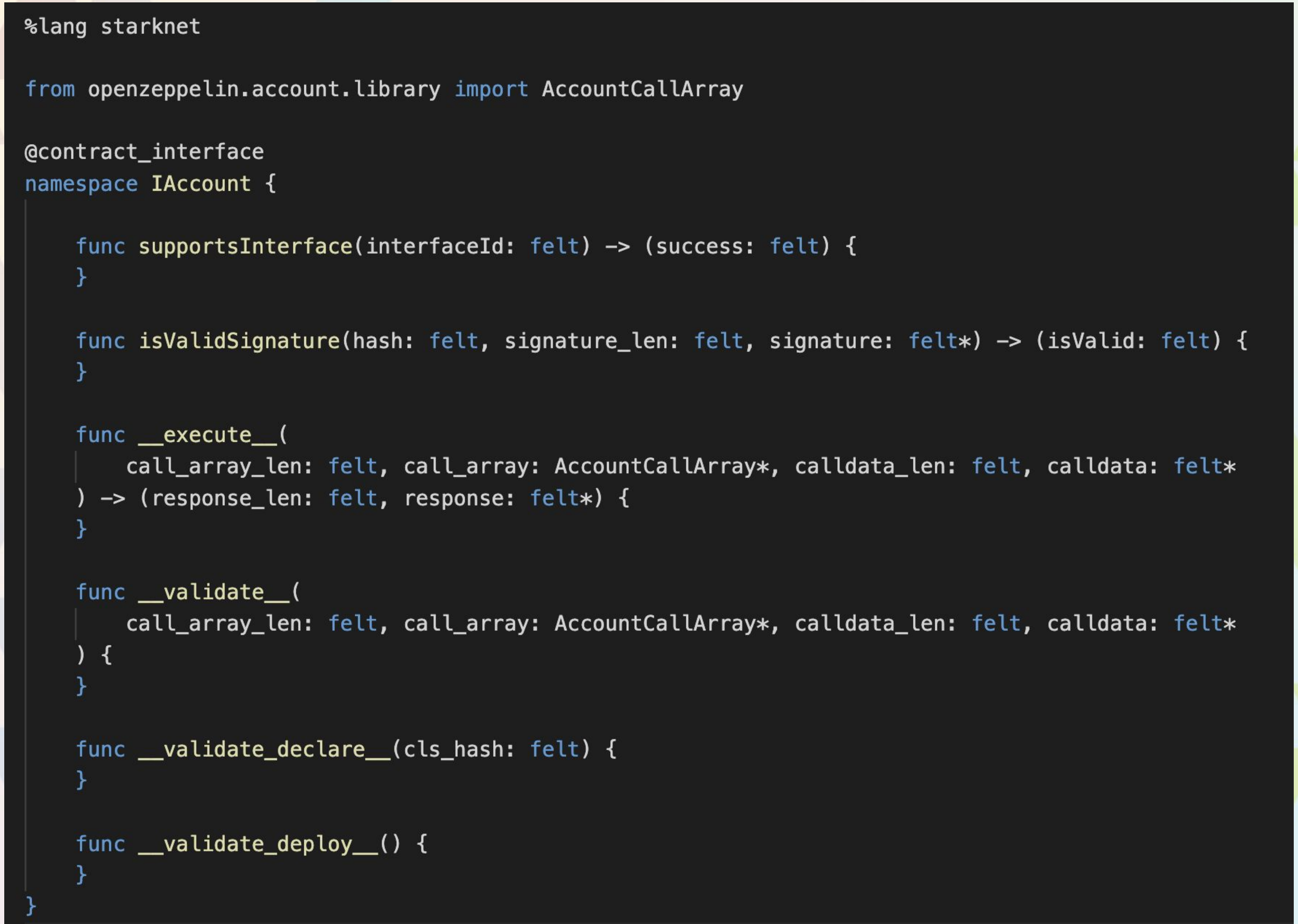
What can we do with them?

- Custom tx validation schemes
  - Ethereum signatures? Bitcoin signatures? Both?
  - Multisig?
  - Only valid on Wednesdays?
- Key rotation
- Guardians
- Social recovery
- Session keys

Section 2

How does it look in StarkNet

Account Interface



```
%lang starknet

from openzeppelin.account.library import AccountCallArray

@contract_interface
namespace IAccount {

    func supportsInterface(interfaceId: felt) -> (success: felt) {
    }

    func isValidSignature(hash: felt, signature_len: felt, signature: felt*) -> (isValid: felt) {
    }

    func __execute__(
        call_array_len: felt, call_array: AccountCallArray*, calldata_len: felt, calldata: felt*
    ) -> (response_len: felt, response: felt*) {
    }

    func __validate__(
        call_array_len: felt, call_array: AccountCallArray*, calldata_len: felt, calldata: felt*
    ) {
    }

    func __validate_declare__(cls_hash: felt) {
    }

    func __validate_deploy__() {
    }
}
```

Two step execution flow

__validate__

- arbitrary logic to determine whether a transaction is valid
- cannot read other contracts storage → anti-spam
- sequencers check this function and funds availability before accepting a transaction

__execute__

- what you would expect

Counterfactual deployments make it easy

1. Calculate the address before deploying
2. Send funds to the address
3. The contract pays for the tx if it passes **__validate_deploy__**
4. Contract deployed ✨

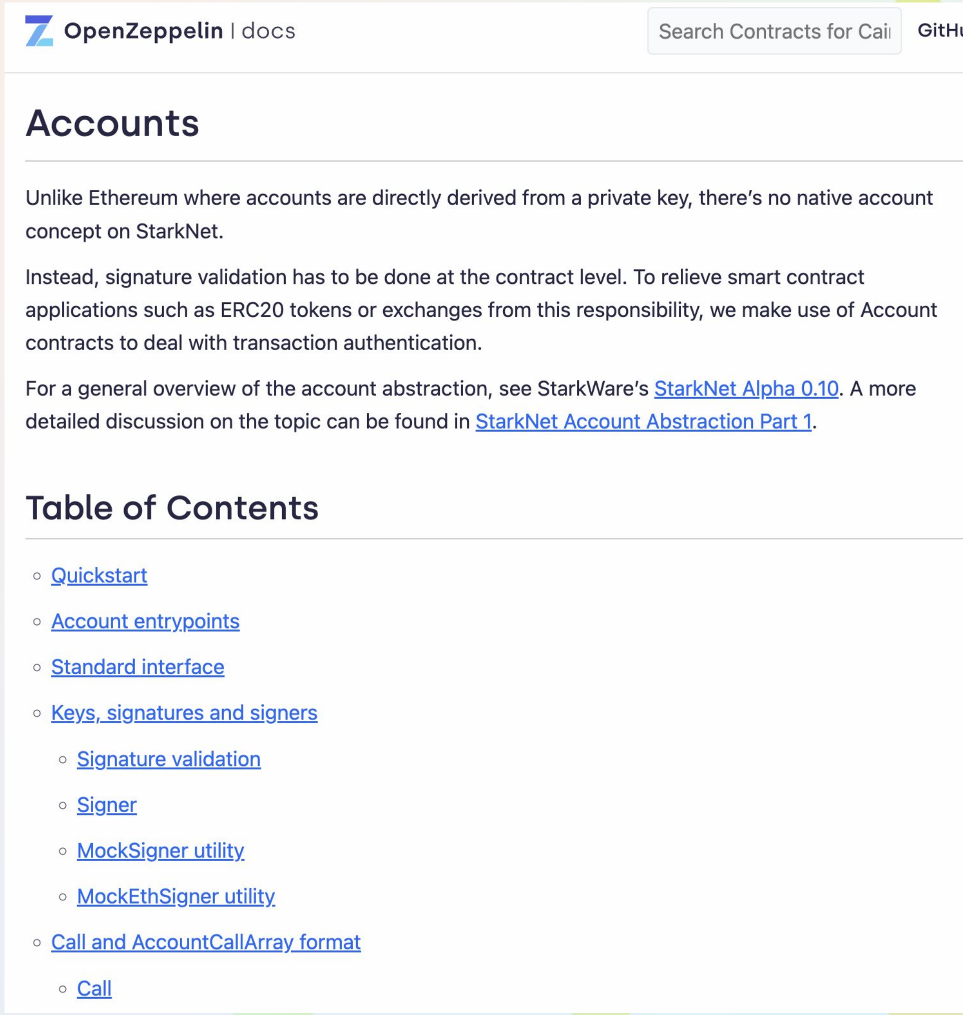
Accounts need to be deployed

OpenZeppelin Contracts (for Cairo)

Flavors

- Account (vanilla)
- EthAccount
- Account library
  (to implement custom accounts)

https://docs.openzeppelin.com/contracts-cairo/accounts


OpenZeppelin | docs

Search Contracts for Cai

GitHu

Accounts

Unlike Ethereum where accounts are directly derived from a private key, there's no native account concept on StarkNet.

Instead, signature validation has to be done at the contract level. To relieve smart contract applications such as ERC20 tokens or exchanges from this responsibility, we make use of Account contracts to deal with transaction authentication.

For a general overview of the account abstraction, see StarkWare's StarkNet Alpha 0.10. A more detailed discussion on the topic can be found in StarkNet Account Abstraction Part 1.

Table of Contents

Account Abstraction means huge improvements in onboarding, user experience, and security.

Thank you!

Martín Triay

OpenZeppelin

marto@openzeppelin.com

@martriay