Marco Walz / 26-08-2022

æternity

Universe Two Hæckathon

How to make an NFT using AEX-141

What to expect?

What to expect?

- **No minting UI / Launchpad and marketplaces available yet**

  - this is why Hackathons exist, right? ☺
- **GitHub repository with a detailed README**

  - https://github.com/aeternity/aex141-nft-collection-example
- **Step-by-step guide how to ...**

  - ... (programmatically) create an NFT collection
  - ... mint NFTs
  - ... use CLI to fetch NFT data directly from contract state by calling various entrypoints

æternity

Agenda

Agenda

1. NFT Standards - why are they needed?

2. ERC-721 vs. AEX-141

3. Dealing with Metadata – a first proposal

4. Sample Collection: „Apes stepping into the Metaverse“
   - Deployment of contract
   - Minting of NFTs
   - Fetching NFT metadata

5. What to build? ☺

6. Social Channels - Get in touch with us! 💬

æternity

(NFT) Standards

Why are they needed?

(NFT) Standards – Why are they needed?

Strict rules for everybody to follow …

> … are defined a **standard definition**:
- Fungible Tokens (ERC-20 on EVM based chains, AEX-9 on æternity)
- Non-Fungible-Tokens (ERC-721, ERC-1155 on EVM based chains, AEX-141 on æternity)
- …

> … can be **extended** by more complex standards that build upon other standards

> … **needed for platforms**:
- marketplaces
- minting solutions / launchpads
- …

> … **useful for developers** to know that their implementation will be supported by platforms if they follow the rules

æternity

ERC-721
VS.
AEX-141

ERC-721

IERC721

Required interface of an ERC721 compliant contract.

FUNCTIONS

`balanceOf(owner)`

`ownerOf(tokenId)`

`safeTransferFrom(from, to, tokenId)`

`transferFrom(from, to, tokenId)`

`approve(to, tokenId)`

`getApproved(tokenId)`

`setApprovalForAllowner, _approved)`

`isApprovedForAll(owner, operator)`

`safeTransferFrom(from, to, tokenId, data)`

`supportsInterface(interfaceId)` IERC165

EVENTS

`Transfer(from, to, tokenId)`

`Approval(owner, approved, tokenId)`

`ApprovalForAll(owner, operator, approved)`

IERC721Receiver

Interface for any contract that wants to support safeTransfers from ERC721 asset contracts.

FUNCTIONS

`onERC721Received(operator, from, tokenId, data)`

AEX-141

```
contract interface IAEX141 =
    datatype metadata_type = URL | IPFS | OBJECT_ID | MAP
    datatype metadata = MetadataIdentifier(string) | MetadataMap(map(string, string))

    record meta_info =
        { name: string
        , symbol: string
        , base_url: option(string)
        , metadata_type : metadata_type }

    datatype event
        = Transfer(address, address, int)
        | Approval(address, address, int, string)
        | ApprovalForAll(address, address, string)

    entrypoint aex141_extensions : () => list(string)
    entrypoint meta_info : () => meta_info
    entrypoint metadata : (int) => option(metadata)
    entrypoint balance : (address) => option(int)
    entrypoint owner : (int) => option(address)
    stateful entrypoint transfer : (address, address, int, option(string)) => unit
    stateful entrypoint approve : (address, int, bool) => unit
    stateful entrypoint approve_all : (address, bool) => unit
    entrypoint get_approved : (int) => option(address)
    entrypoint is_approved : (int, address) => bool
    entrypoint is_approved_for_all : (address, address) => bool
```

```
contract interface IAEX141NFTReceiver =
    entrypoint on_nft_received : (address, address, int, option(string)) => bool
```

https://github.com/aeternity/AEXs/blob/master/AEXS/aex-141.md

æternity

Dealing with Metadata

Dealing with metadata

> **avoid using URLs** that point to **traditional websites**
  > data can get **lost**
  > data can be **manipulated** (if no proper checks to verify metadata are defined/provided)

> store your (external) metadata on a **distributed storage (e.g. IPFS)** whenever possible
  > data can also get "lost" if nobody pins the file in the distributed storage
  > data can easily "recovered" by pinning it again
  > hash of the data never changes ☺

> **first proposal** how to deal with **metadata of NFT collections in AEX-141**
  > new extension will be proposed (name to be defined ;-)) where following data will be used using metadata_type "MAP":

- `name` the name of the NFT
- `description` the description of the NFT
- `media_type` the media type of the NFT
  - e.g. `NONE`, `IMAGE`, `AUDIO`, `VIDEO`, `3D_GLB`, `PDF`, ... (every value is allowed right now, but ideally we define the best route to go based on discussions within the community)
- `media_url`
  - e.g. `ipfs://`, `ar://` (also not specified yet, but it might make sense to go this route)
  - we recommend to **AVOID** using centralized URLs as this property must be immutable
- `immutable_attributes`
  - a JSON string representing `map<string, object>` so that basically everything is possible to be represented here
  - these attributes are **immutable** and it should **NOT** be possible to change them
- `mutable_attributes` (not demonstrated in this example yet)
  - a JSON string representing `map<string, object>` so that basically everything is possible to be represented here
  - mutable attributes are especially interesting for game items, where you can for example collect experience points for the NFTs
  - of course any usecase is possible

æternity

Sample Collection

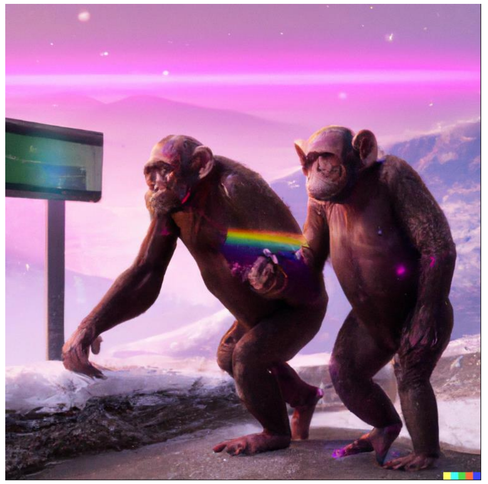
„Apes stepping into the Metaverse“

Sample Collection: „Apes stepping into the Metaverse“

> Example available on **GitHub**
  > https://github.com/aeternity/aex141-nft-collection-example

> Images **created with AI** using DALL-E 2
> Images **pinned on IPFS** using Pinata

> Collection created on Testnet
  > contract: ct_Fv9d66QTjr4yon9GEuMRc2B5y7Afy4to1ATaoYmpUTbN6DYiP
  > amount of minted NFTs: 8



Sample Collection: „Apes stepping into the Metaverse“

1. Define metadata

```
{
    "name": "Apes stepping into the Metaverse",
    "symbol": "ASITM",
    "nfts": [
        {
            "name": "Walking on the ladder",
            "description": "They are escaping from earth and stepping into the metaverse!",
            "media_type": "IMAGE",
            "media_url": "ipfs://QmfCr586aHFVk6p2WhTC1Kvcaps24Mtny2CLB5bsTT9MvZ",
            "immutable_attributes": {
                "apes_count": 2,
                "moon_visible": true
            },
            "mutable_attributes": {
                "retries": 0
            }
        },
        {
            "name": "The path to heaven",
            "description": "Heaven or metaverse? We don't care!",
            "media_type": "IMAGE",
            "media_url": "ipfs://Qmef7Xrh1YTgQqXbr86o3TrFoya9ZLk1RVdXMga1JjEjnm",
            "immutable_attributes": {
                "apes_count": 3,
                "moon_visible": false
            },
            "mutable_attributes": {
                "retries": 0
            }
        },
        {...
        },
```

2. Deploy contract

> provide meta_info for the collection
> provide metadata for each NFT (read from json file)

```
const CONTRACT = './contracts/MintableMutableNFT.aes';
const source = utils.getContractContent(CONTRACT);
const fileSystem = utils.getFilesystem(CONTRACT);

const contract = await aeSdk.getContractInstance({ source, fileSystem });

// deploy
await contract.deploy([
    collectionMetadata.name,
    collectionMetadata.symbol
]);
console.log(`Contract successfully deployed!`);
console.log(`Contract address: ${contract.deployInfo.address}`);
console.log(`Tx-Hash: ${contract.deployInfo.txData.hash}`);
console.log(`Gas used: ${contract.deployInfo.result.gasUsed}`);
console.log(`----------------------------------------------------------------------------------------------------`);
console.log(`----------------------------------------------------------------------------------------------------`);

// mint
for(let i=0; i<collectionMetadata.nfts.length; i++) {
    const nftMetadataMapStringValues = new Map(Object.entries(collectionMetadata.nfts[i]).map(([k, v]) => [k, String(v)]));
    const tx = await contract.methods.mint(
        senderAddress,
        {'MetadataMap': [nftMetadataMapStringValues]}
    );
    console.log(`Minted '${nftMetadataMapStringValues.get('name')}' with id '${tx.decodedResult}'`);
    console.log(`Tx-Hash: ${tx.hash}`);
    console.log(`Gas used: ${tx.result.gasUsed}`);
    console.log(`----------------------------------------------------------------------------------------------------`);
    console.log(`----------------------------------------------------------------------------------------------------`);
};
```

Sample Collection: „Apes stepping into the Metaverse“

Console output

```
Deploying with account: ak_QVSUoGrJ31CVxWpvgvwQ7PUPFgnvWQouUgsDBVoGjuT7hjQYW
==> Adding include to filesystem: core/utils.aes
==> Adding include to filesystem: core/interfaces.aes
Contract successfully deployed!
Contract address: ct_2tw26RwgNADrpuCnrQWKPBH87bPxuRbLR1KLccS9ZJTUMMj4z8
Tx-Hash: th_2vXnGY3GB7ieWRYAXiiEzFJNtn5WbMSVoLEsagEuUn9bgi1vHe
Gas used: 974
--------------------------------------------------------------------------------
--------------------------------------------------------------------------------
Minted 'Walking on the ladder' with id '1'
Tx-Hash: th_2d5iaRa2DkgJb6ABSt5ea6TcM1FVB2EW6dx7FRU9XMWi1J4n9e
Gas used: 14499
--------------------------------------------------------------------------------
--------------------------------------------------------------------------------
Minted 'The path to heaven' with id '2'
Tx-Hash: th_BPiUgq2aqm7rTmhb68DW2vEhReWda3mioxeFBjPfxGtLnkAtg
Gas used: 14615
--------------------------------------------------------------------------------
--------------------------------------------------------------------------------
Minted 'Still sitting in the jungle' with id '3'
Tx-Hash: th_2KummuRWbQVPv6vitcQg1ymUiQ731GvFYMPKYxCNMF4GYqDeeH
Gas used: 14925
--------------------------------------------------------------------------------
--------------------------------------------------------------------------------
Minted 'We almost made it!' with id '4'
Tx-Hash: th_bSmhMu9zba3t9mwZf4ts1a9Rks7mDxcMRriR4mA1DPfwGgiXs
Gas used: 14555
--------------------------------------------------------------------------------
--------------------------------------------------------------------------------
Minted 'I'm in!' with id '5'
Tx-Hash: th_bwWPhicZjxgP7BRSjoXgT6dzWzkWBNFZEAPTws8se5nMdwPzy
Gas used: 14547
--------------------------------------------------------------------------------
--------------------------------------------------------------------------------
```

Sample Collection: „Apes stepping into the Metaverse“

Use the **CLI** to read data from contract using the ACI provided in the repository
> https://github.com/aeternity/aepp-cli-js

meta_info

```
marc0olo@DESKTOP-B1QLEDL:~/git/aeternity/aex141-nft-collection-example$ aecli contract call sample-wallet --contractAddress ct_2tw2
6RwgNADrpuCnrQWKPBH87bPxuRbLR1KLccS9ZJTUMMj4z8 --contractAci aex-141-ACI.json --callStatic meta_info
✓ Enter your password … ****
----------------------Call info----------------------
Contract address ____________________ ct_2tw26RwgNADrpuCnrQWKPBH87bPxuRbLR1KLccS9ZJTUMMj4z8
Gas price ___________________________ 1000000000
Gas used ____________________________ 8128
Return value (encoded) ______________ cb_S4FBcGVzIHN0ZXBwaW5nIGludG8gdGhlIE1ldGF2ZXJzZRVBU0lUTa+CAAEAP6+EAAAAAAM/LMdBvA==
Return value (decoded) ______________ {"name":"Apes stepping into the Metaverse","symbol":"ASITM","metadata_type":{"MAP":[]}}
```

Returning

```
{
  "name":"Apes stepping into the Metaverse",
  "symbol":"ASITM",
  "metadata_type":{"MAP":[]}
}
```

balance (of ak_QVSUoGrJ31CVxWpvgvwQ7PUPFgnvWQouUgsDBVoGjuT7hjQYW)

```
marc0olo@DESKTOP-B1QLEDL:~/git/aeternity/aex141-nft-collection-example$ aecli contract call sample-wallet --contractAddress ct_2tw2
6RwgNADrpuCnrQWKPBH87bPxuRbLR1KLccS9ZJTUMMj4z8 --contractAci aex-141-ACI.json --callStatic balance '["ak_QVSUoGrJ31CVxWpvgvwQ7PUPFg
nvWQouUgsDBVoGjuT7hjQYW"]'
✓ Enter your password … ****
----------------------Call info----------------------
Contract address ____________________ ct_2tw26RwgNADrpuCnrQWKPBH87bPxuRbLR1KLccS9ZJTUMMj4z8
Gas price ___________________________ 1000000000
Gas used ____________________________ 2081
Return value (encoded) ______________ cb_r4IAAQEbED5vR8I=
Return value (decoded) ______________ 8
```

Returning
8

metadata *(of NFT with id 1)*

```
marc0olo@DESKTOP-B1QLEDL:~/git/aeternity/aex141-nft-collection-example$ aecli contract call sample-wallet --contractAddress ct_2tw2
6RwgNADrpuCnrQWKPBH87bPxuRbLR1KLccS9ZJTUMMj4z8 --contractAci aex-141-ACI.json --callStatic metadata '[1]'
✓ Enter your password … ****
----------------------Call info----------------------
Contract address ____________________ ct_2tw26RwgNADrpuCnrQWKPBH87bPxuRbLR1KLccS9ZJTUMMj4z8
Gas price ___________________________ 1000000000
Gas used ____________________________ 122364
Return value (encoded) ______________ cb_r4IAAQEbr4IBAQEbLwYRbmFtZVVXYWxraW5nIG9uIHRoZSBsYWRkZXIlbWVkaWFfdXJs1WlwZnM6Ly9RbWZDcj
U4NmFIRlZrNnAyV2hUQzFLdmNhcHMyNE10bnkyQ0xCNWJzVFQ5TXZaKW1lZGlhX3R5cGUVSU1BR0UtZGVzY3JpcHRpb271VGhleSBhcmUgZXNjYXBpbmcgZnJvbSBlYXJ0a
CBhbmQgc3RlcHBpbmcgaW50byB0aGUgbWV0YXZlcnNlIUltdXRhYmxlX2F0dHJpYnV0ZXM9W29iamVjdCBPYmplY3RdUWltbXV0YWJsZV9hdHRyaWJ1dGVzPVtvYmplY3Qg
T2JqZWN0XZHeF1I=
Return value (decoded) ______________ {"MetadataMap":[{}]}
```

Return value here currently doesn‘t show the actual medata. Check following issue:
- https://github.com/aeternity/aepp-cli-js/issues/204

In the tests provided in the example repo the actual metadata is verified, see:
- https://github.com/aeternity/aex141-nft-collection-example/blob/master/test/mintableMutableNFT.js#L83

æternity

What to build?

What to build?

Minting Solution / Launchpad

Marketplace

Auction platform

Fractional Ownership

Social campaign airdrops verified by Oracles

... whatever else you want!

æternity

Social Channels

Social Channels 💬

> Forum

> Discord

> Twitter

> Telegram (Official)
> Telegram (Universe Two)

> Website

And don‘t forget to sign up for the Hæckathon!

👉 https://universe-two-haeckathon.devpost.com

THANKS

æternity