Minting Reference

This document is a reference to the Minting module of the Myria Core SDK. The module contains information about Myria mint transactions.

Interfaces

`MintERC721Params`

Data structure passed to createMintTransactionERC721() method, which contains required data to create a new ERC721 mint transaction.

```
interface MintERC721Params {
    starkKey: string;
    contractAddress: string;
    uri: string;
    tokenId: string;
    description?: string;
    fees: [
        {
            percentage: number;
            receiptAddress: string;
            feeType: FeeType;
        }
    ];
}
```

Attributes

- `starkKey` - Stark Key, has to start with `0x`
- `contractAddress` - contract address used to withdraw assets to the Ethereum network
- `uri` - URL path to the mintable asset metadata
- `tokenId` - unique identifier of a given asset, should be an incremental numeric value starting from 1
- `description` - description of a given asset
- `fees` - an array of fees details for a given asset. FeeType accepts: `ROYALTY`, `TAKER`, `MAKER`, `PROTOCOL`

`MintERC721Response`

Data structure returned by the createMintTransactionERC721() method.

```
interface MintERC721Response {
    status: string;
    data: any;
}
```

Attributes

- `status` - response status: success, fail
- `data` - response data

`GetMintedTransactionParams`

Data structure passed to getMintTransactionList() method, which contains required parameters to get details of a given minted transaction.

```
interface GetMintedTransactionParams {
    transactionId: number;
}
```

Attributes

- `transactionId` - unique id of a minted transaction

`GetMintedTransactionResponse`

Data structure passed to getMintTransactionList() method, which contains required data to get details of the minted asset.

```
interface GetMintedTransactionResponse {
    status: string;
    data: {
        data: string;
        updatedAt: number;
        vaultId: number;
        transactionStatus: string;
        createdAt: number;
        starkKey: string;
```

```
        transactionId: number;
        transactionType: string;
        batchId: number;
        quantizedAmount: string;
        assetId: string;
    };
}
```

Attributes

- `status` - response status: success, failure
- `data` - response data
- `updatedAt` - date of the latest transaction update
- `vaultId` - vault id
- `transactionStatus` - transaction status
- `createdAt` - date of transaction creation
- `starkKey` - Stark Key, has to start with `0x`
- `transactionId` - unique transaction id
- `transactionType` - transaction type
- `batchId` - batch id
- `quantizedAmount` - quantized amount
- `assetId` - unique asset id

`GetMintedAssetsParams`

Data structure passed to getMintedAssetByStarkKey() method, which contains required parameters to get a list of minted assets.

```
interface GetMintedAssetsParams {
    starkKey: string;
}
```

Attributes

- `starkKey` - Stark Key, has to start with `0x`

`GetMintedAssetsResponse`

Data structure returned by the getMintedAssetByStarkKey() method.

```
interface GetMintedStarkKeyResponse {
    status: string;
}
```

Attributes

- `status` - response status: success, fail
- `data` - response data

Methods

`createMintTransactionERC721()`

Creates a mint transaction of an ERC721 asset.

```
createMintTransactionERC721(data: MintERC721Params):
Promise<MintERC721Response | undefined>;
```

Parameters

- MintERC721Params object

Returns

Returns an object with response call status and the `data` object that contains details of the mint transaction.

Example

Typescript

```
import { MintingManager, MintERC721Params, MintERC721Response, FeeType,
EnvTypes } from "myria-core-sdk";

(async (): Promise<void> => {
  // STAGING or PRODUCTION
  const env = EnvTypes.STAGING;
```

```
  const mintingManager: MintingManager = new MintingManager(env);

  const params: MintERC721Params = {
    starkKey: "STARK_KEY",
    contractAddress: "CONTRACT_ADDRESS",
    uri: "TOKEN_URI",
    tokenId: "TOKEN_ID",
    description: "DESCRIPTION",
    royalties: [
      {
        percentage: 10,
        receiptAddress: "ROYALTY_RECIPIENT_ADDRESS",
      },
    ],
  };

  const mintTransactionResponse: MintERC721Response | undefined =
    await mintingManager.createMintTransactionERC721(params);
})();
```

> **INFO**
>
> Currently, royalties aren't supported.

Response

▸ MintERC721Response

`getMintTransactionList()`

Returns details of the minted asset by a given transaction id.

```
getMintTransactionList(data: GetMintedTransactionParams):
Promise<GetMintedTransactionResponse>;
```

Parameters

- GetMintedTransactionParams object

Returns

Returns an object with response call status and the `data` object that contains details of the minted asset.

Example

Typescript

```
import { MintingManager, GetMintedTransactionParams,
GetMintedTransactionResponse, EnvTypes } from "myria-core-sdk";

(async (): Promise<void> => {
  // STAGING or PRODUCTION
  const env = EnvTypes.STAGING;

  const mintingManager: MintingManager = new MintingManager(env);

  const params: GetMintedTransactionParams = {
    transactionId: TRANSACTION_ID,
  };

  const mintTransactionResponse: GetMintedTransactionResponse |
undefined =
    await mintingManager.getMintTransactionList(params);
})();
```

Response

▸ GetMintedTransactionResponse

`getMintedAssetByStarkKey()`

Returns a list of minted assets by a given Stark Key.

```
getMintedAssetByStarkKey(data: GetMintedAssetsParams):
Promise<GetMintedAssetsResponse>;
```

Parameters

- GetMintedAssetsParams object

Returns

Returns an object with response call status and the `data` object that contains a list of minted assets by a given Stark Key.

Example

Typescript

```
import { MintingManager, GetMintedTransactionParams,
GetMintedTransactionResponse, EnvTypes } from "myria-core-sdk";

(async (): Promise<void> => {
  // STAGING or PRODUCTION
  const env = EnvTypes.STAGING;

  const mintingManager: MintingManager = new MintingManager(env);

  const params: GetMintedAssetsParams = {
    starkKey: "STARK_KEY",
  };

  const mintStarkKeyResponse: GetMintedAssetsResponse | undefined =
    await mintingManager.getMintedAssetByStarkKey(params);
})();
```

Response

▸ GetMintedAssetsResponse