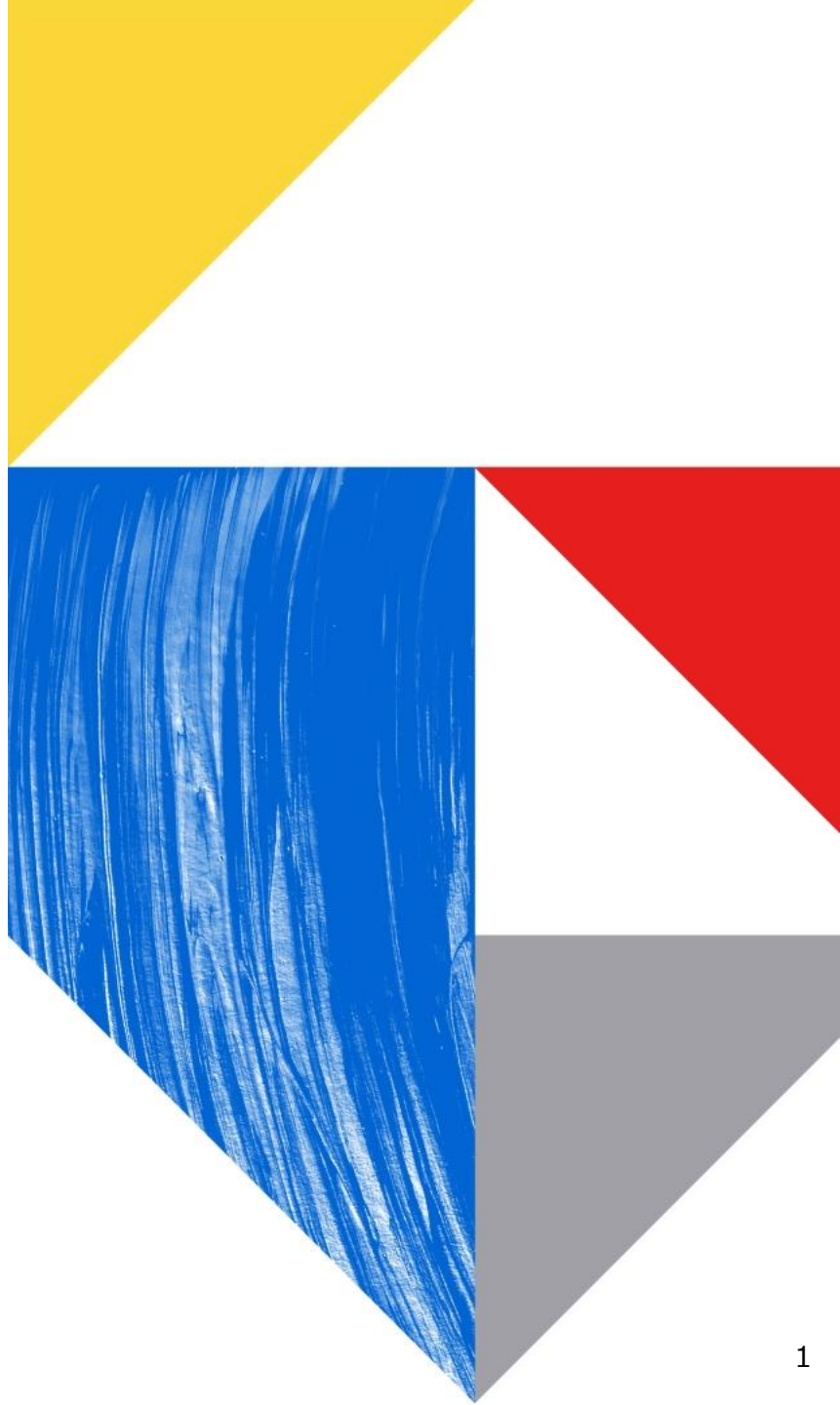
NFTについて



概要

NFT(Non-Fungible Token)は、イーサリアム上で構築できる代替不可能なトークンとして、ゲームや暗号資産などで利用されています。しかしビジネス用途ではイーサリアムの利用自体が難しい場合が多くあります。そこで、同様のトークンをBC+上でも実現し、（これもNFTと呼ぶことにします）エンタープライズ領域でのNFTの活用を広くすすめていきたいと考えています。

本家イーサリアムのNFTはスマートコントラクトとして実装されています。 ERC721と呼ばれる標準的なインターフェスが定義されており、NFTの開発者がそれに準拠してスマートコントラクトを作成することにより、アプリケーションからの利用を容易にしています。これと同様に、DNCWARE Blockchain+(BC+)においても、NFTの標準的なインタフェースを定義し、それに準拠する形でNFTをスマートコントラクトとして実装します。

本書では、下記のドキュメントについての概要や利用方法を説明していきます。
・NFTインタフェース定義： BC+におけるNFTの標準的なインタフェースを定義したものです。
・NFTサンプルコード：上記インタフェース定義に準拠したNFTコントラクトのサンプルコードです。
・NFTサンプルコード利用例：NFTサンプルコードの利用例をノートブック形式で説明した文書です。

なお、コインについても、NFTと同様にインタフェース定義とサンプルコードがあります。

ファイル一覧

インタフェース定義

- nft-spec.html:NFTに関するインタフェース定義
- coin-spec.html:コイン(Fungible Token)に関するインタフェース定義

サンプルコード

- nft100.mjs:NFTコントラクトの最小限の実装を行ったサンプルコード
- nft200.mjs:上記に加え、mint/burnと転送履歴を実装したサンプルコード
- coin100.mjs:コインコントラクトの最小限の実装を行ったサンプルコード

利用例

- nft100-usage1.html:nft100の各機能の確認例
- nft100-usage2.html:nft100でコントラクトにapproveする例
- nft200-usage1.html:nft200の各機能の確認例
- coin100-usage1.html:coin100の各機能の確認例
- coin100-usage2.html:coin100でコントラクトにapproveする例

NFTインタフェース定義の見かた

HTML形式のファイルとして提供されます。ブラウザで開いて見ることができます。

BC+におけるNFTの標準的なインタフェースを定義しています。

章立てはたとえば下記のようになっています。
・用語の説明
・NFTコントラクトのプロパティ
・NFTコントラクトの動作
・NFTコントラクトの各機能の動作
　⇒１５の各機能の仕様
・onNFTReceivedの定義
・特記事項
・独自の機能拡張

DNCWARE Blockchain+におけるNFTインタフェース定義（

用語の説明

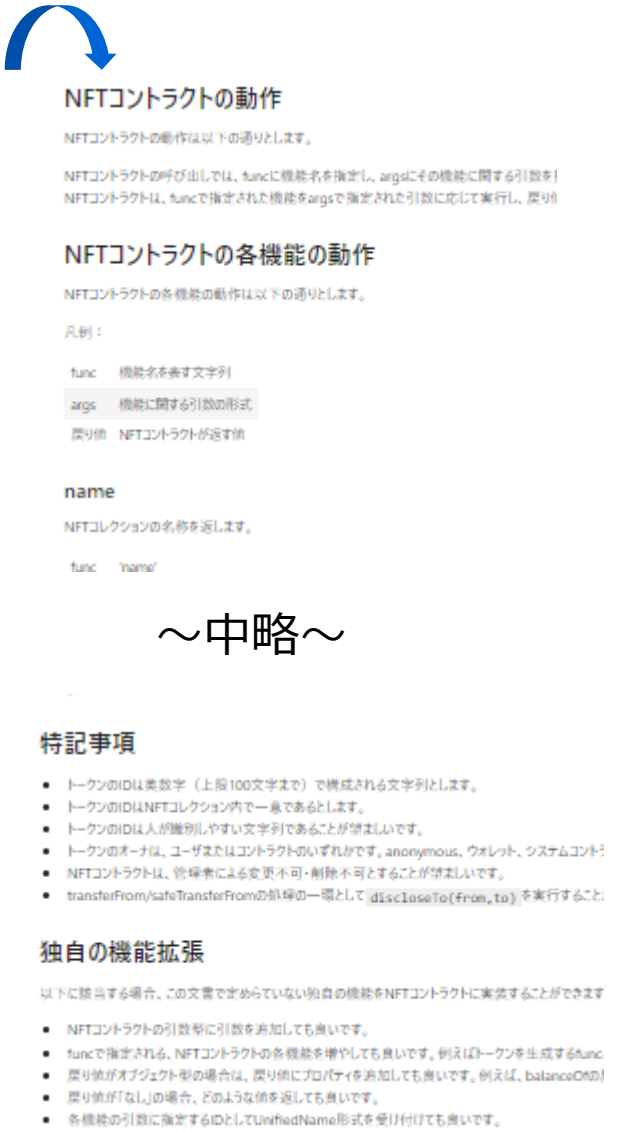
用語	説明
NFT	Non Fungible Tokenの略称です。 基本的には非代替性のトークンのとつのとつのことを指しますが、意味が抽象的に広がる場合もありま
トークン	ブロックチェーン上に記録されたデジタル情報です。デジタル資産としての価値を持つ場合があります。 代替性の有無によって種類が分けられ、非代替性の性質をもつトークンはNFTと呼ばれます。
NFTコレクション	同じテーマやコンセプトに基づいた一連の非代替性トークンの集合です。 NFTコレクションに含まれるトークンは、その中でユニークなIDを持ちます。
NFTコントラクト	NFTコレクションを実現した、ブロックチェーン上のスマートコントラクトです。 NFTコントラクトをブロックチェーンにデプロイすることで、NFTコレクションが実現されます。

NFTコントラクトのプロパティ

NFTコントラクトのプロパティの設定は以下の通りとします。

プロパティ		設定値などの説明
name（オブジェクト名）		（任意）
description（説明）		（任意）
argtypes（引数型）		{ func: "string", args: "any" }
mask（制限）	inaccessible	無効（実行可）
	inaccessible as subcontract	無効（サブコントラクトとして呼び出し可）
	inaccessible by delegation	無効（delegationによる呼び出し可）
	nonrecording	無効（トランザクションを記録する）
code		（インターフェース定義に準拠した任意のコード）
accessible_to（実行許可先リスト）		（NFTの流通範囲に合わせて設定）
disclosed_to（履歴開示先リスト）		（NFTの取引履歴等の開示範囲を設定）

なお、１つのスマートコントラクトで１つのNFTコレクションを実現します。
つまり、NFTコレクションとNFTコントラクトの対応は１対１です。



NFTコントラクトの動作

NFTコントラクトの動作は以下の通りとします。

NFTコントラクトの呼び出しでは、funcに機能名を指定し、argsにその機能に関する引数を
NFTコントラクトは、funcで指定された機能をargsで指定された引数に応じて実行し、戻り

NFTコントラクトの各機能の動作

NFTコントラクトの各機能の動作は以下の通りとします。

凡例：

func	機能名を表す文字列
args	機能に関する引数の形式
戻り値	NFTコントラクトが返す値

name

NFTコレクションの名称を返します。

func 'name'

～中略～

特記事項

- トークンのIDは英数字（上限100文字まで）で構成される文字列とします。
- トークンのIDはNFTコレクション内で一意であるとします。
- トークンのIDは人が識別しやすい文字列であることが望ましいです。
- トークンのオーナは、ユーザまたはコントラクトのいずれかです。anonymous、ウォレット、システムコント
- NFTコントラクトは、管理者による変更不可・削除不可とすることが望ましいです。
- transferFrom/safeTransferFromの振舞の一環として `discloseTo(from,to)` を実行すること

独自の機能拡張

以下に該当する場合、この文書で定められていない独自の機能をNFTコントラクトに実装することができます

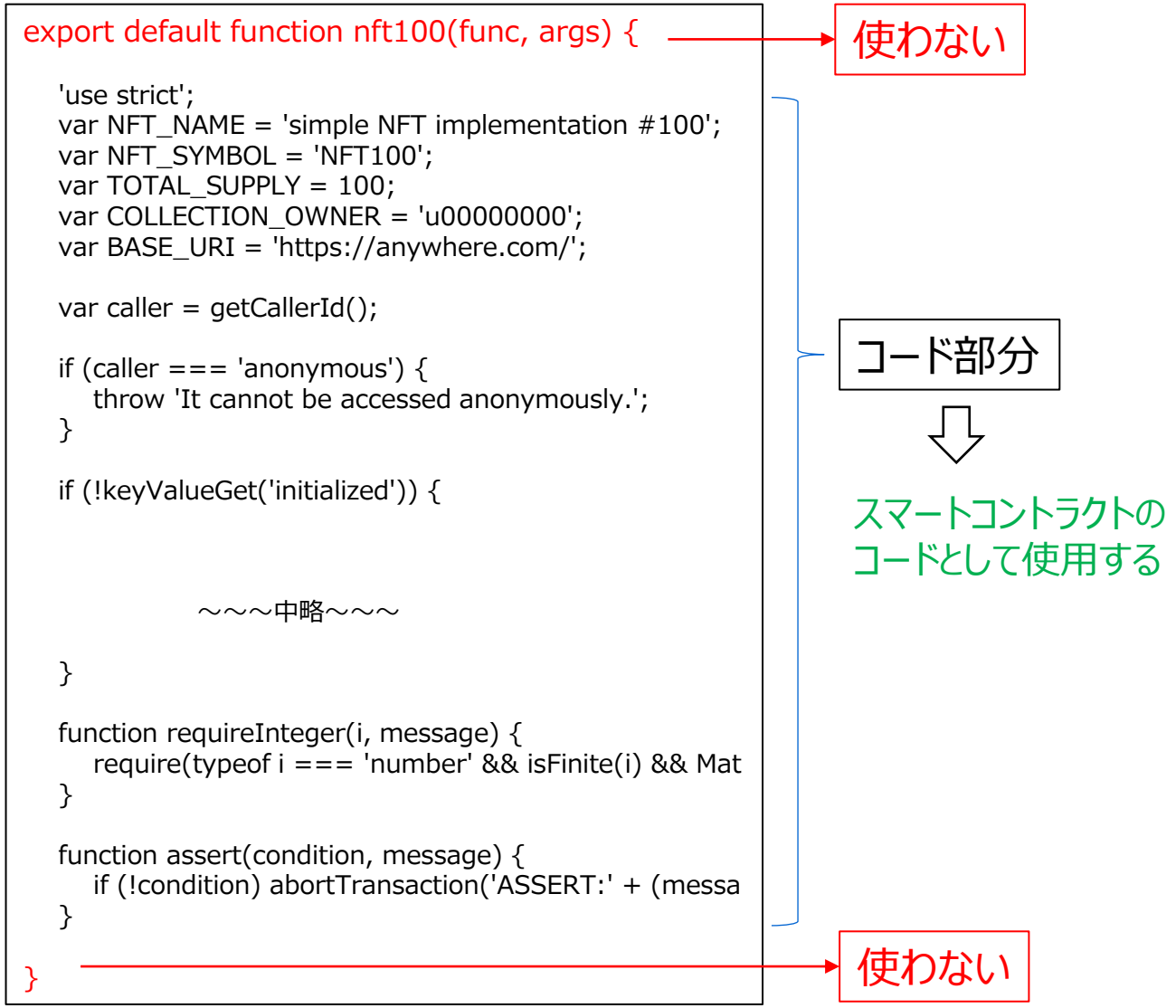
- NFTコントラクトの引数形に引数を追加しても良いです。
- funcで指定される、NFTコントラクトの各機能を増やしても良いです。例えばトークンを生成するfunc
- 戻り値がオブジェクト形の場合は、戻り値にプロパティを追加しても良いです。例えば、balanceOfの
- 戻り値が「なし」の場合、どのような値を返しても良いです。
- 各機能の引数に指定するIDとしてUnifiedName形式を受け付けても良いです。

NFTサンプルコードの見かた

NFTサンプルコードは、NFTコントラクト（スマートコントラクト）のコードのサンプルです。javascriptのモジュール形式のファイルとして提供され、任意のエディタ（メモ帳、VSCodeなど）で開いて見ることができます。

javascriptモジュールの形式(*.mjs)となっていますが、これは便宜上の形式であり、DNCWARE Blockchain+のスマートコントラクトとしてデプロイした場合のみ正しく動作するコードです。

なお、スマートコントラクトのコードとして使用するのは、exportしているfunctionのコード部分のみです。ファイルの最初の行と最後の行は使いません。（右図参照）



NFTサンプルコード利用例の見かた

HTMLファイルとして提供され、ブラウザで開いて見ることができます。

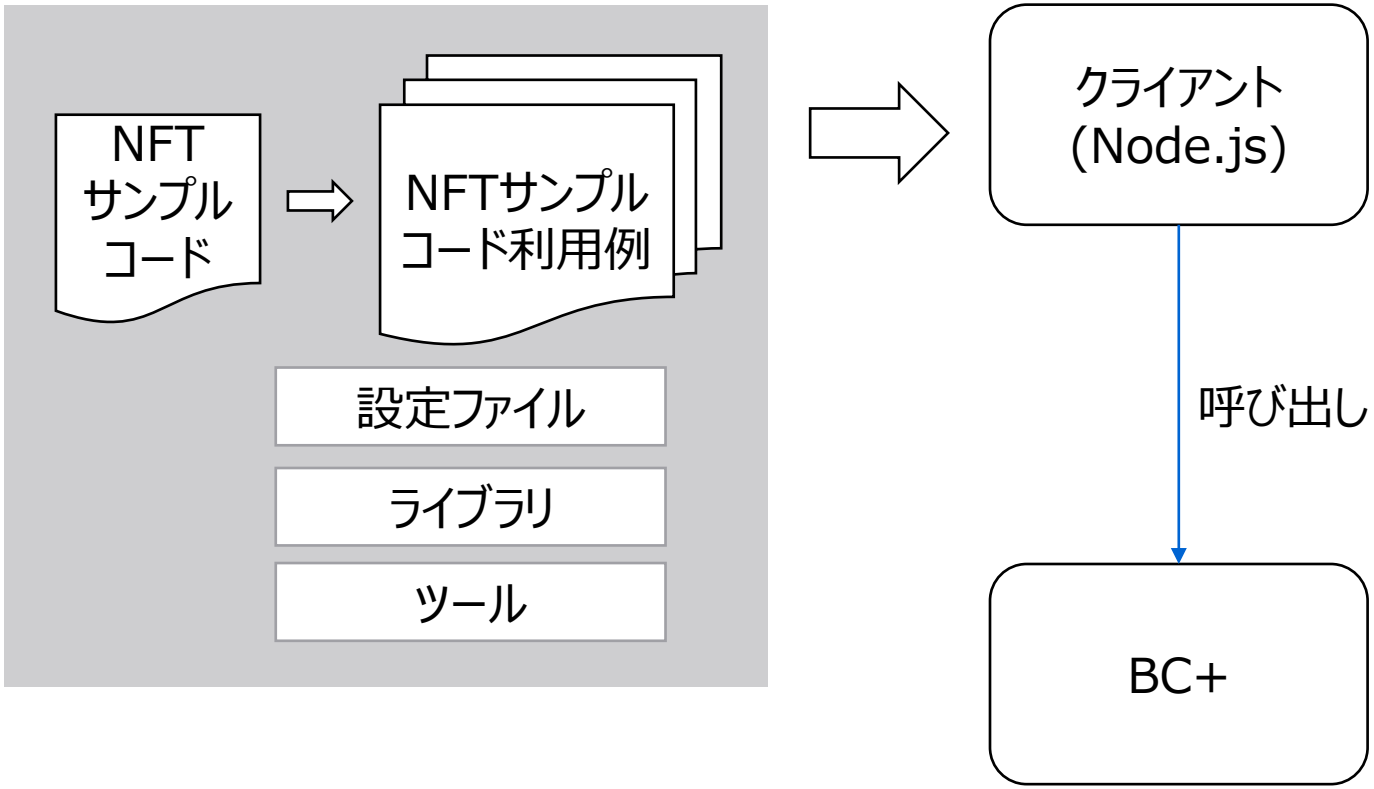
右のようなノートブック形式となっており、説明つきでサンプルコードとその実行結果が示されています。

実行は、上から下へと進みます。

コードセルの中にコードそのものが示されています。コードセルの前にその説明、後ろにその実行結果が示されています。（右図参照）

NFTサンプルコード利用例の動作環境

NFTサンプルコード利用例は、他のサンプルコードと共通の動作環境を前提としています。その詳細については、サンプルコード全般の説明を参照してください。



本書では、これ以降、こちらの利用例を使った説明はしません。（プログラミングベースの方法であるため）
代わりに、本書では、より手軽に実施できる、管理画面のGUIを使った利用方法について説明していきます。

ブロックチェーンへの導入方法

ブロックチェーンへの導入方法　ー 概要

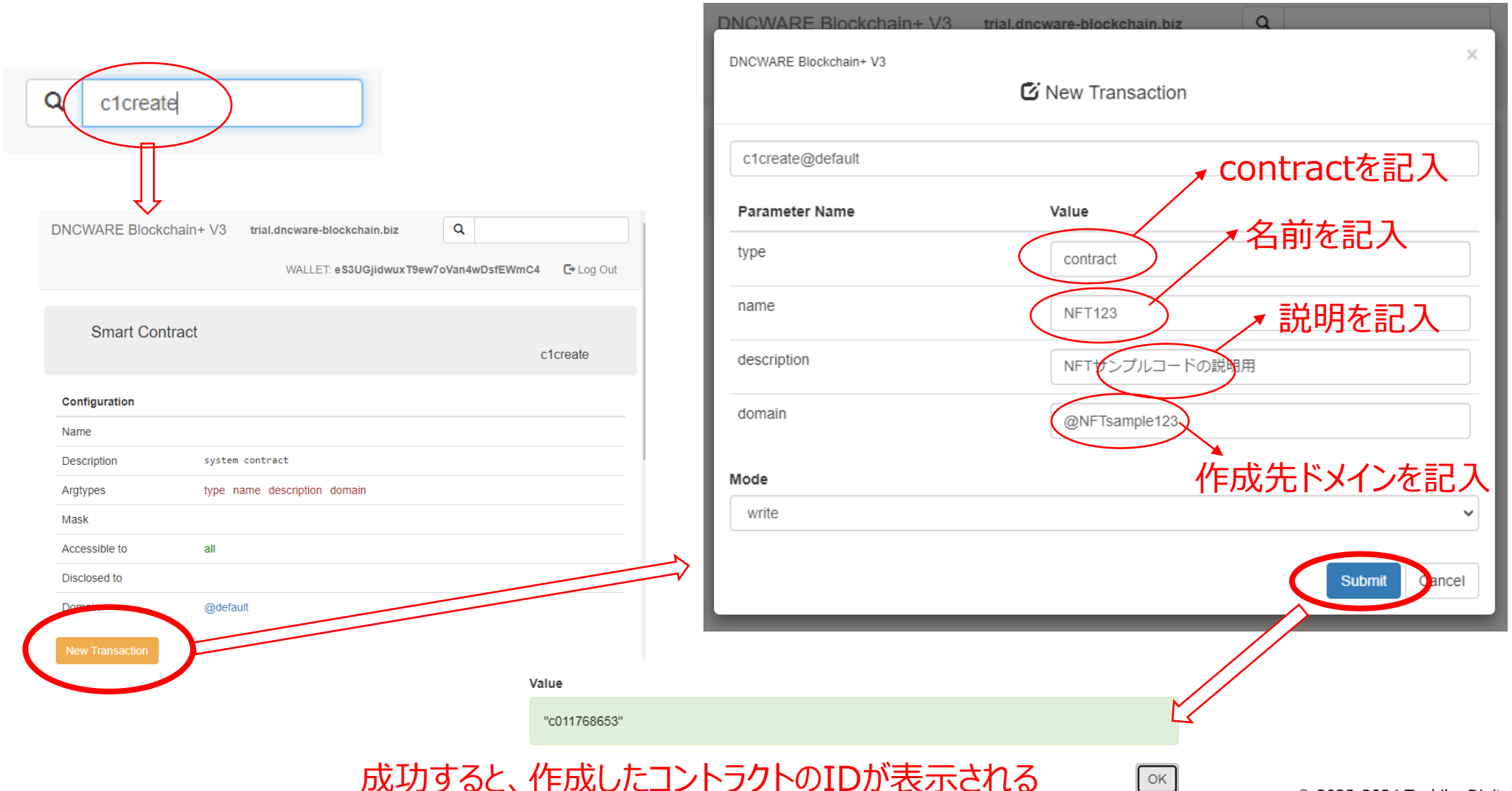
ブロックチェーンへのNFTの導入方法の概要を説明します。
ここでは、自身で管理するドメインにNFTを導入する方法を説明します。
事前に、「NFTインタフェース定義」の「用語の説明」を理解しておいてください。

手順
(1) 管理画面にドメイン管理者のウォレットでログインし、ブロックチェーンに接続する。
(2) ドメイン内にNFT用のスマートコントラクトを作成する。
(3) NFTコントラクトのプロパティを設定する。（アクセス権限を除く）
(4) 「NFTサンプルコード」からコードをコピペし、カスタマイズする。
(5) 動作の確認をする。
(6) アクセス権限をNFTの流通範囲に合わせて設定する。
(7)（必要に応じて） NFTコントラクトを書き換え不可に設定する。

以下、手順の(2)から順に説明していきます。

ドメイン内にNFT用のスマートコントラクトを作成する

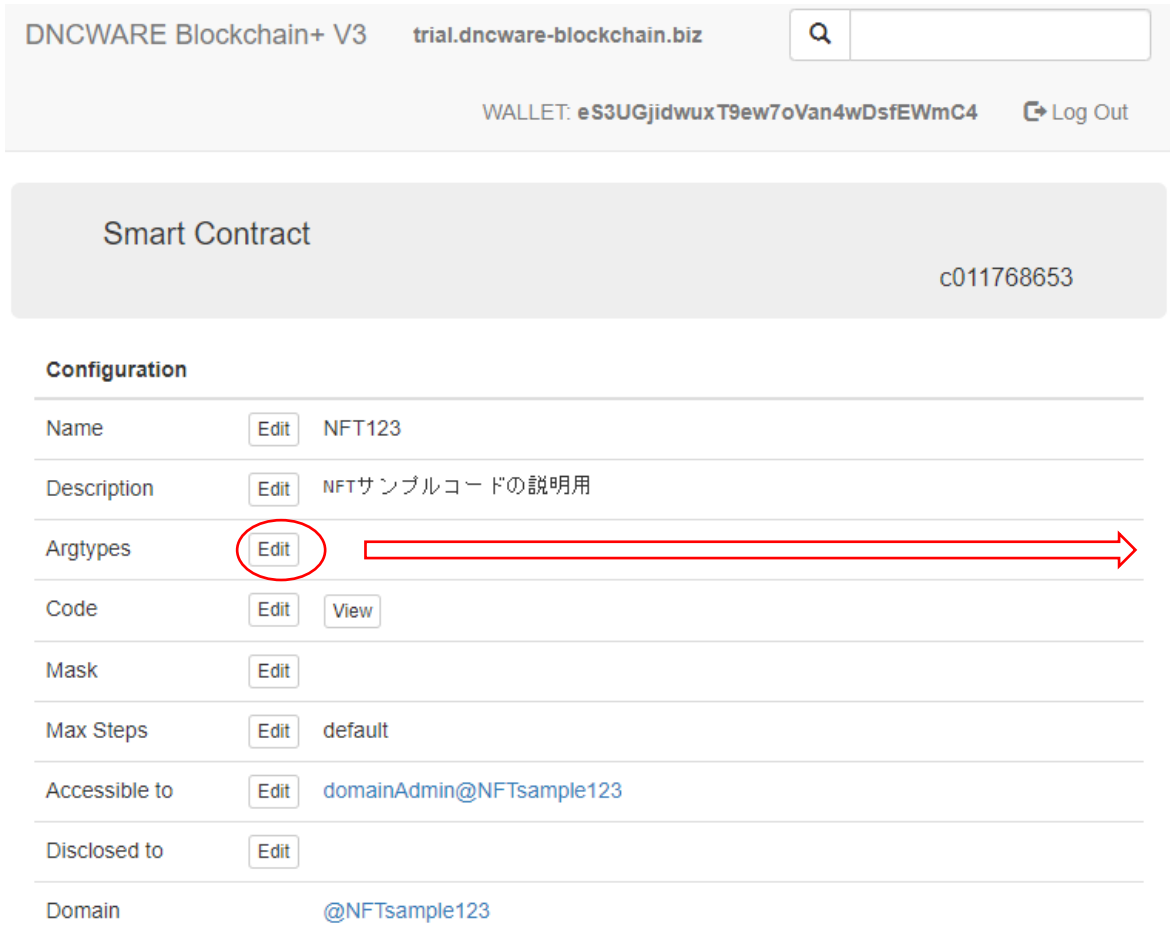
c1createにより、NFTコントラクト用のスマートコントラクトを作成します。

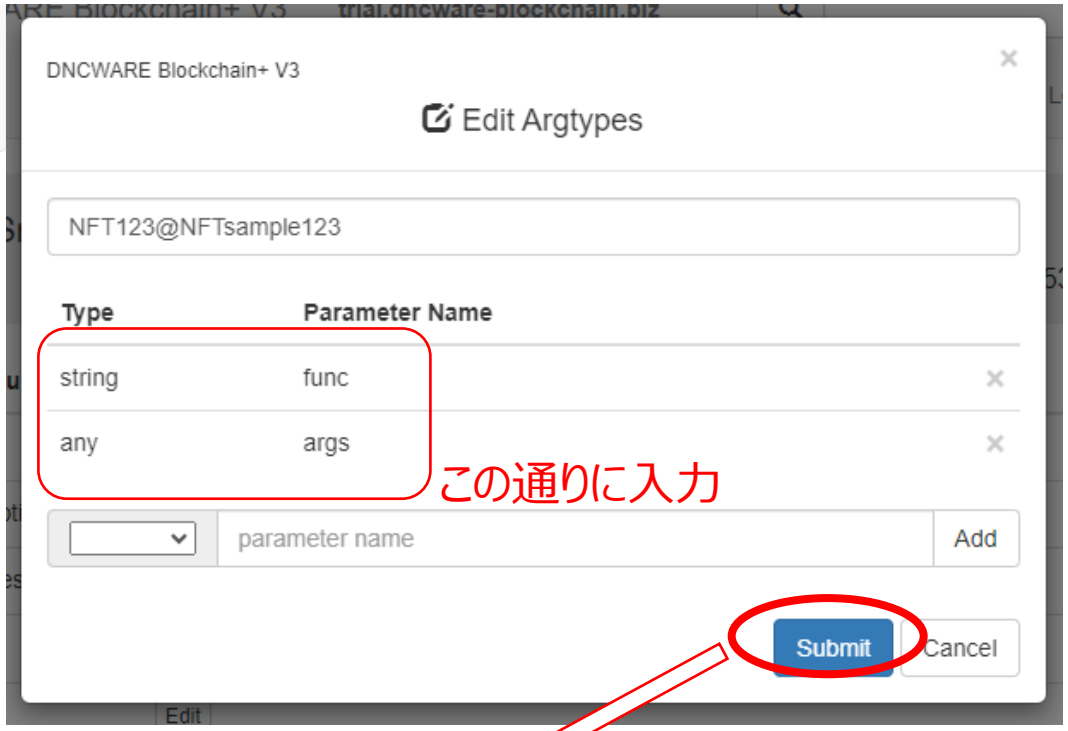


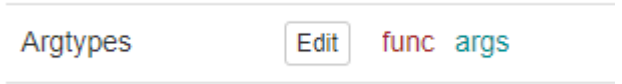
成功すると、作成したコントラクトのIDが表示される

NFTコントラクトのプロパティを設定する

作成したコントラクトの画面を開き、
「NFTインタフェース定義」の「NFTコントラクトのプロパティ」に合わせて設定します。（アクセス権限を除く）
ここで実際に設定変更が必要なのはArgtypesの項目だけです。

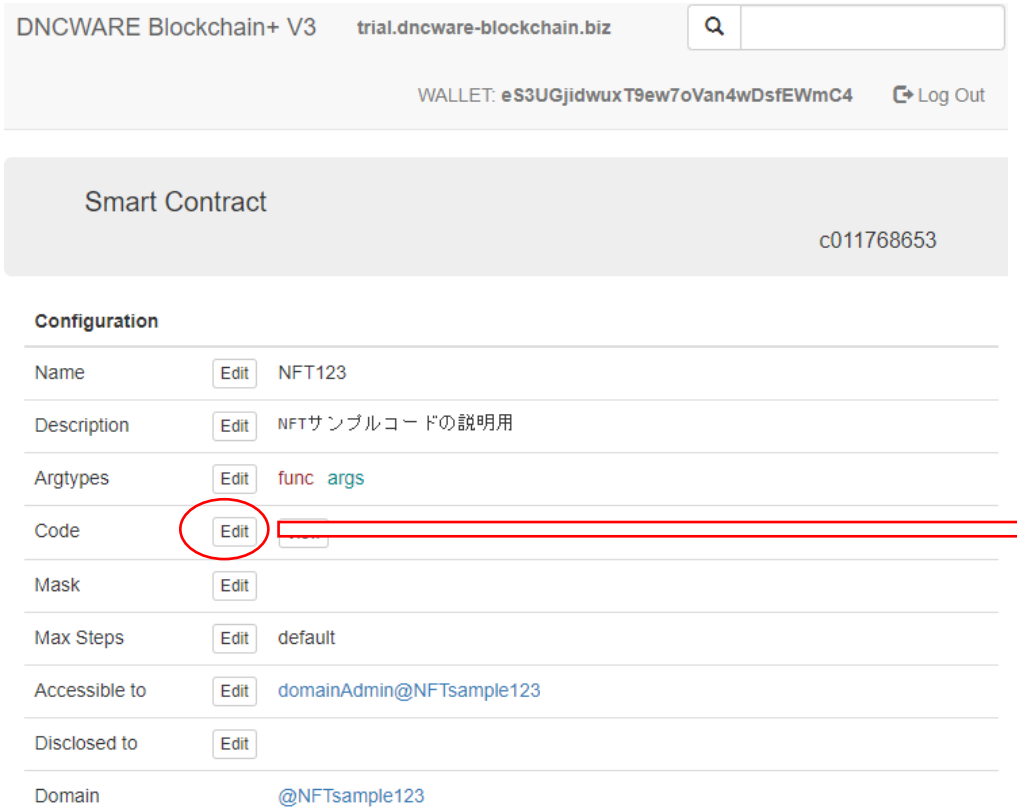






「NFTサンプルコード」からコードをコピペし、カスタマイズする

NFTコントラクトのコードは独自に作成しても良いですが、
ここでは NFTサンプルコード(nft100.mjs)からコードをコピー＆ペーストし、冒頭部分をカスタマイズします。
特にINITIAL_OWNERは必ず変更してください。



カスタマイズの詳細

NFTサンプルコード(nft100.mjs)の冒頭部分のカスタマイズの詳細について説明します。

var NFT_NAME = '********';
NFTコレクションの名称を設定します。

var NFT_SYMBOL = '********';
NFTコレクションの略称を設定します。

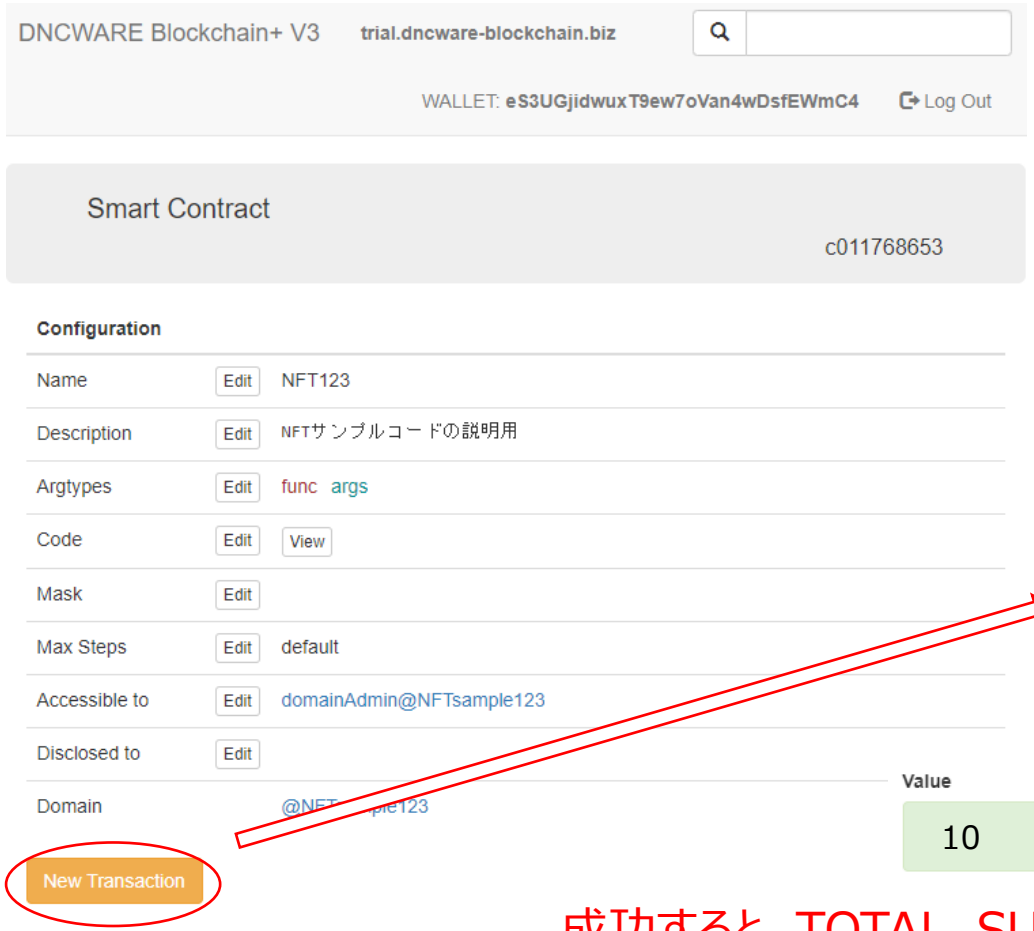
var TOTAL_SUPPLY = ***;
NFTコレクションに含まれるトークンの総数を設定します。（100以下の数を指定します）

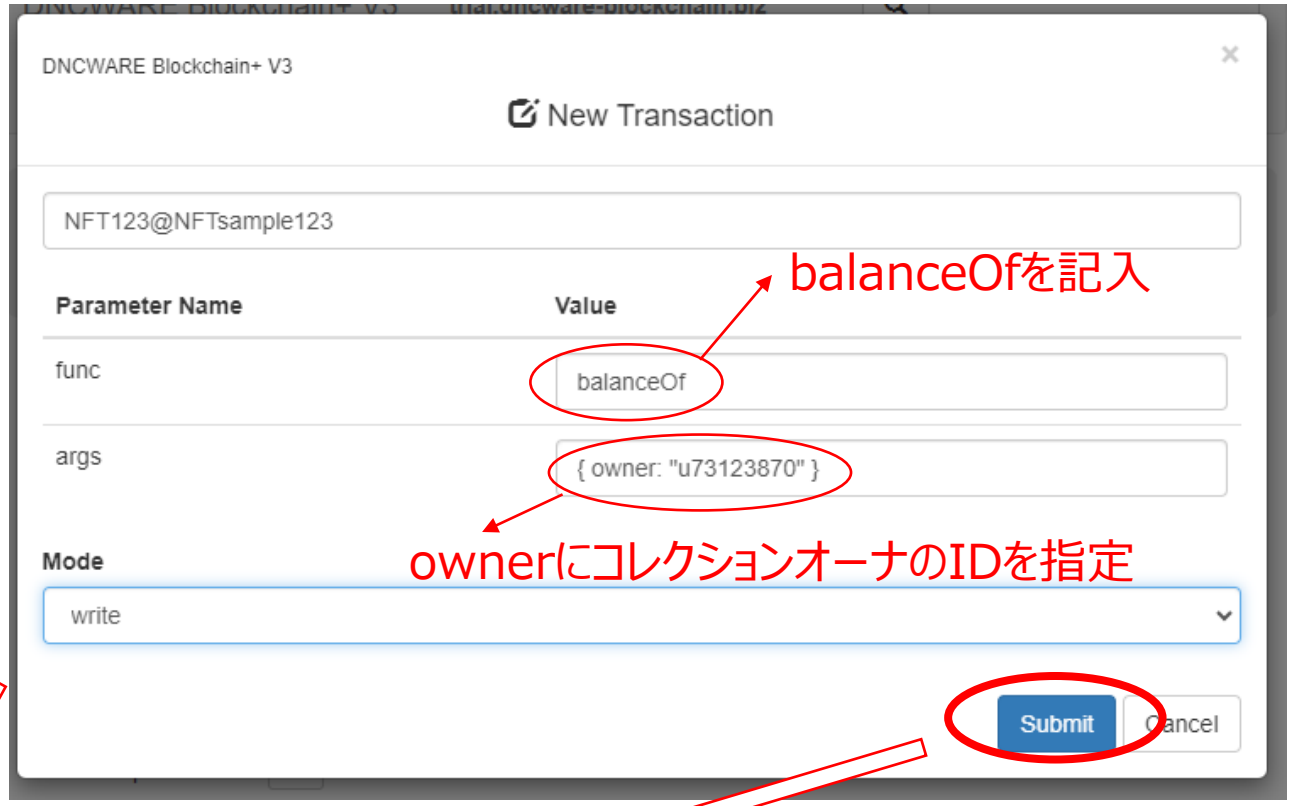
var INITIAL_OWNER = '********';
NFTコレクションの初期オーナのIDを設定します。
例えば、ドメイン管理者（自分）のユーザIDに設定します。

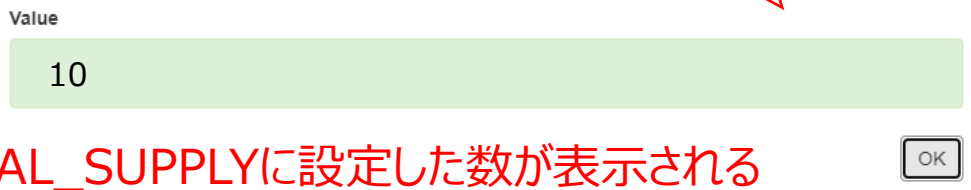
var BASE_URI = '********';
メタデータの場所を設定します。（URIが実際に存在するかをコード上ではチェックしません。）
BASE_URI+tokenIDの形でtokenURIが決まります。

動作の確認をする

例えば、コレクションオーナのbalanceOfを確認します。



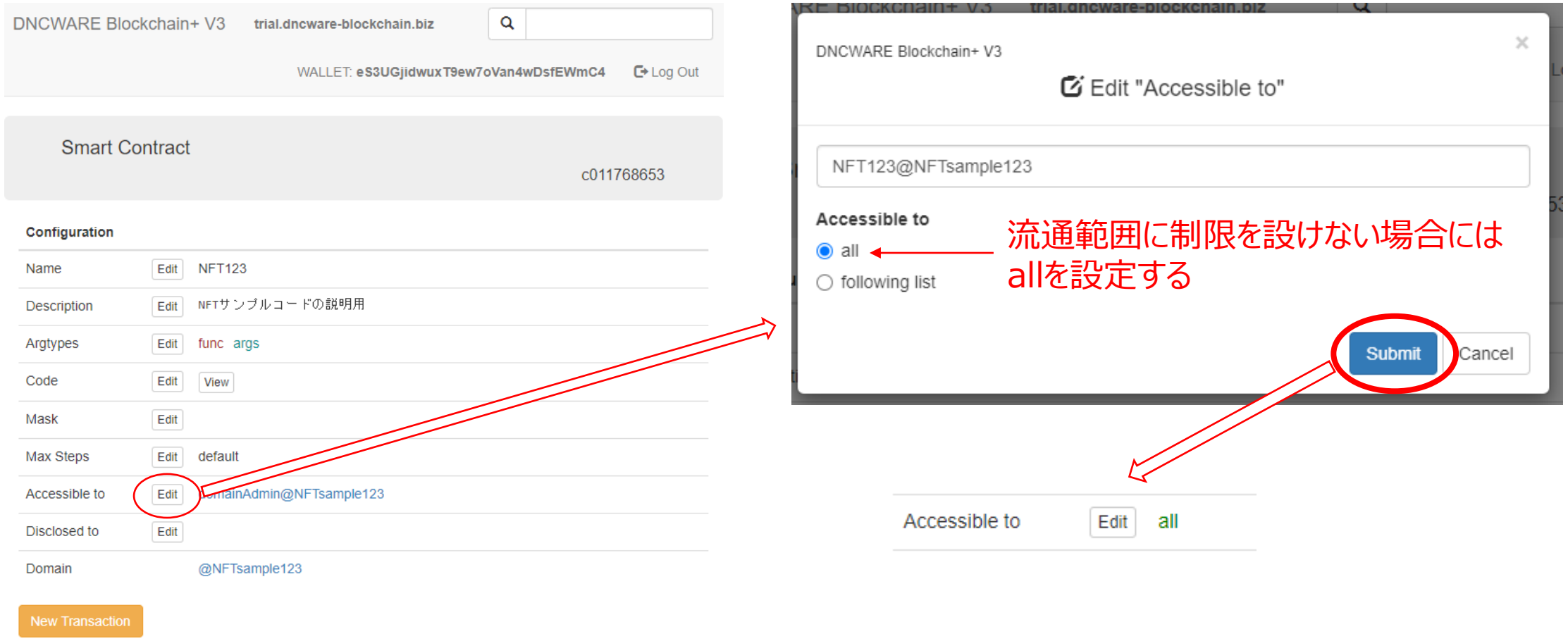




成功すると、TOTAL_SUPPLYに設定した数が表示される

アクセス権限をNFTの流通範囲に合わせて設定する

アクセス権限を設定します。これにより、NFTの流通が可能になります。



NFTコントラクトを書き換え不可に設定する

もともとのイーサリアム上のNFTコントラクトは、デプロイ後のコードの書き換えが不可となっています。将来にわたって誰もコードを変更することができない性質は、非中央集権的な特徴の一つと考えられています。
DNCWARE　Blockchain＋でも、NFTコントラクトのコードの変更を不可に設定することができます。そして、ひとたび変更不可に設定すると、元に戻すことはできません。これにより、（必要に応じて）イーサリアムと同様の性質のNFTを作成できます。

変更不可に設定する方法について、ここでは２通りの方法を説明します。

（１）ドメインの管理権限(managed_by)を空欄にする。（管理者がいない状態にする）
　コントラクトのプロパティの変更のためにはドメインの管理権が必要なため、管理権限をすべて削除すれば、当然コントラクトのコードは変更不可になります。また、管理権限の変更もその管理権が必要なため、だれも管理権を取得できなくなっています。そのため、将来にわたって、そのドメイン内のオブジェクトを変更できなくなります。

（２）c1freeze機能を使ってコントラクトを変更不可に設定する。（バージョン3.3から導入される機能）
　管理権限とは関係なしに、オブジェクトごとにプロパティの変更を不可にする機能がc1freezeです。NFTコントラクトをc1freezeすることにより、 NFTコントラクトのコードの変更を不可に設定することができます。この場合でも、将来にわたって、NFTコントラクトの全プロパティを変更できなくなります。

使用例

典型的には、NFTコントラクトはアプリケーションプログラムからAPIによって操作されます。それに近い形での説明は、別に「NFTサンプルコード利用例」で示しています。アプリ開発の実践としてはそちらを参照してください。

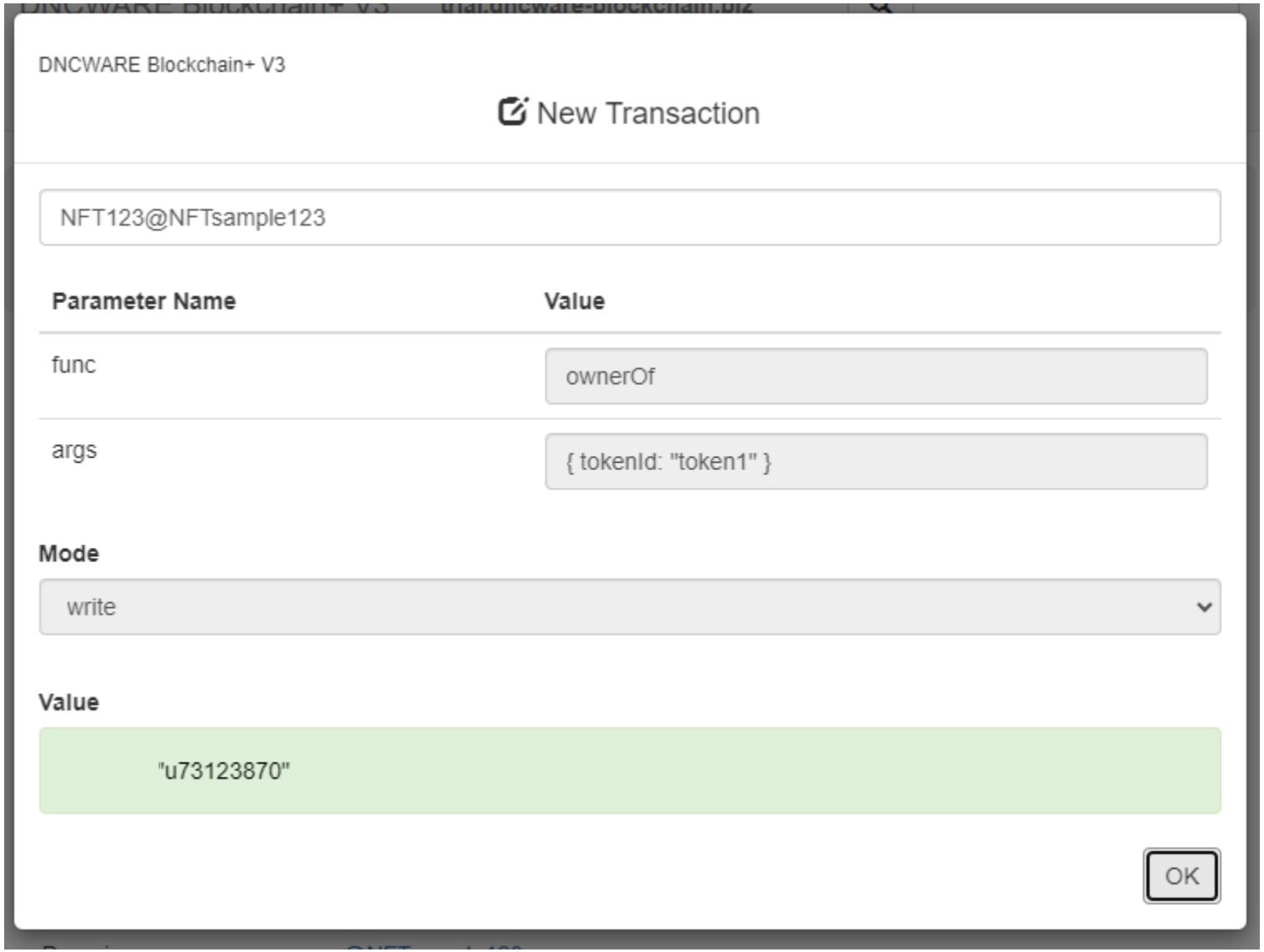
一方、ここではNFTコントラクトの理解を深める目的で、管理画面を使った利用例を簡単に説明します。

ownerOfの実施例

トークンのオーナを問い合わせる例

token1のオーナを問い合わせた結果、ユーザID:u73123870がオーナであると返ってきました。

※なお、以降の実施例では、この管理画面にログインしているユーザが、トークンのオーナであるu73123870と同一であることを暗黙のうちに仮定しています。

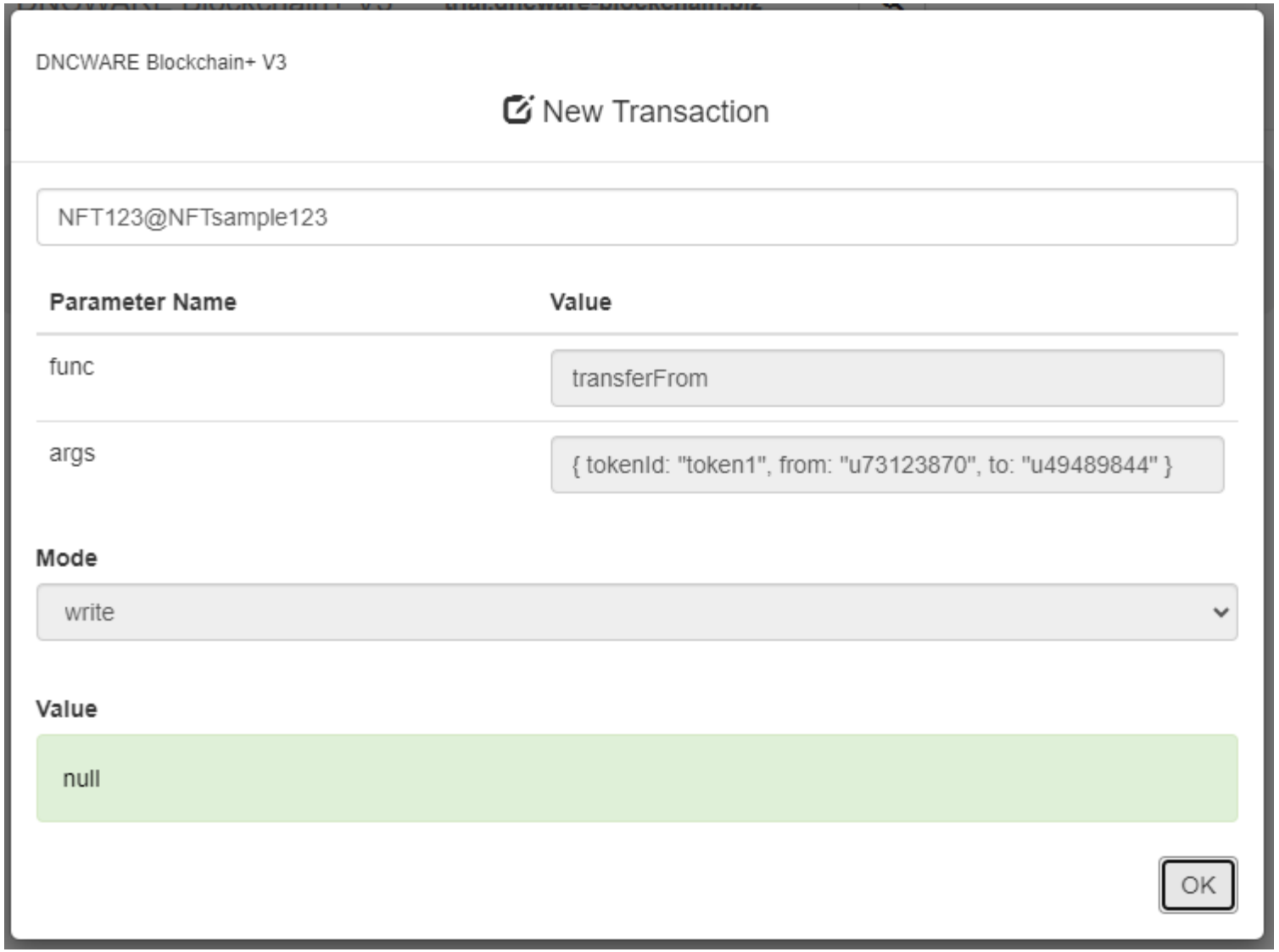


transferFromの実施例

トークンを転送する例

トークンの転送とは、すなわち、トークンのオーナを変更するということです。

token1を、
ユーザID:u73123870から
ユーザID:u49489844へ
転送を要求したところ、成功しました。
(Valueの背景が緑は成功を意味します)

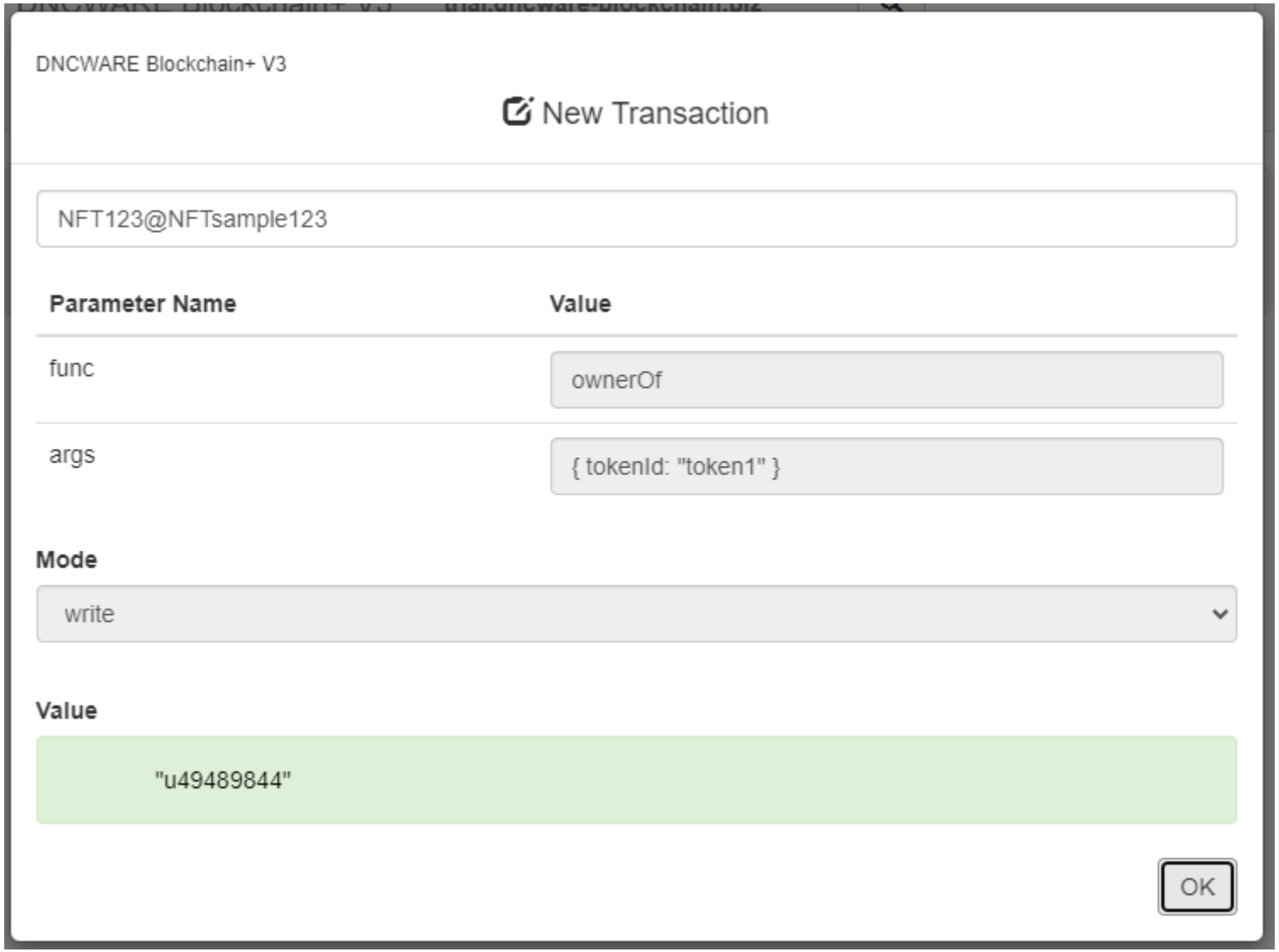


ownerOfの実施例(転送後）

トークンの転送後、トークンのオーナを問い合わせる例

token1のオーナを問い合わせた結果、ユーザID:u49489844がオーナであると返ってきました。

オーナが転送先に変更されていることが分かります。

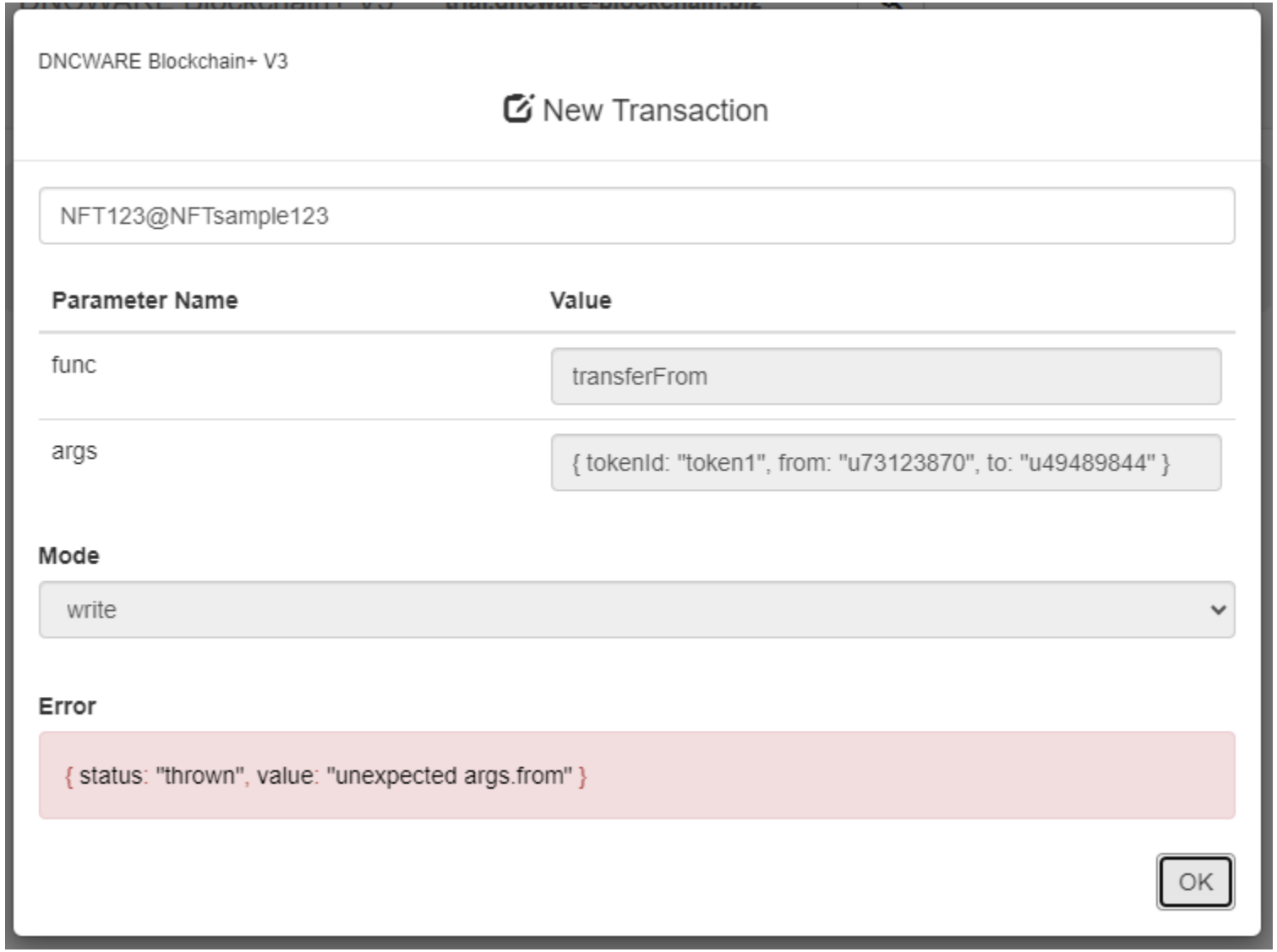


transferFromの実施例（転送後）

トークンの転送に失敗する例１

（前回と同じ転送要求）
token1を、
ユーザID:u73123870から
ユーザID:u49489844へ
転送を要求したところ、失敗しました。
(Valueの背景が赤は失敗を意味します)

なぜなら、fromに指定したユーザと、トークンのオーナが一致しないからです。



transferFromの実施例（転送後）

トークンの転送に失敗する例２

（前回と逆向きの転送要求）
token1を、
ユーザID:u49489844から
ユーザID:u73123870へ
転送を要求したところ、失敗しました。
(Valueの背景が赤は失敗を意味します)

なぜなら、管理画面にログインしたユーザ(caller)と、トークンのオーナが一致しないからです。

※なお、ownerとcallerが一致しない場合でも、委任(approve等)があれば、転送できる場合もあります。詳細は、「インタフェース定義」のtransferFromの項を参照してください。