

NFT + DIGITAL ASSETS REINVENTED

Fast, cheap, green. Purpose-built for media.
Powered by Theta native chain.



CREACIÓN DE NFT EN
THETA NETWORK



Índice

Fuente y versión

Pasos y requisitos previos

PASO 1. Organizamos las pestañas

PASO 2. Abrimos y preparamos el IDE Remix

PASO 3. Copiamos y pegamos el código del contrato

PASO 4. Compilamos el contrato

PASO 5. Accedemos a nuestra Wallet para crear el NFT

PASO 6. Validamos nuestro NFT

Lectura de nuestro Smart Contract

Enviar el NFT a otra wallet

Comprobación del envío del NFT

FUENTE Y VERSIÓN

Información extraída de la explicación de Jieyi Long Cofounder / CTO en la presentación del proyecto Theta Labs en Google Developer Student Club PoliMi el 15 de abril de 2021.

<https://www.youtube.com/watch?v=fmJMcGAg2HU> – Minuto: 25:53

Código fuente de Github de Theta Labs: <https://github.com/thetatoken>



theta-nft-demo

Demo project for creating and transferring an NFT token

● Solidity 0 0 2 0 Updated on 9 Apr

Última actualización del código fuente el 9 de abril de 2021, versión del contrato solidity **0.6.2**

La versión 0.6.2 del contrato para crear un NFT puede cambiar en un futuro, pero los pasos a seguir serán los mismos a menos que Theta Labs indique otra cosa.

Versión 1 del tutorial para la creación de un NFT en Theta Network.

Documento creado el 4 de julio de 2021 por Sersi @sersi29

El autor de este tutorial no se hace responsable de cualquier inconveniente o error que pueda ocurrir en la creación del NFT ni en la posible pérdida de Tfuel gastado en el proceso. Este tutorial se ha realizado siguiendo los pasos del tutorial de Jieyi Long, CTO de Theta Labs.

Wallet utilizada de prueba: 0x868a92be1fc3ea61b4e3456be4cc9c6815b643e1



PASOS Y REQUISITOS PREVIOS ANTES DE CREAR NUESTRO NFT

Para la creación de un NFT necesitaremos:

1. Una Wallet de Theta, si no tenemos ninguna la creamos desde la página oficial de Theta Network:

<https://wallet.thetatoken.org/>

2. Tener al menos 20 Tfuel + Tfuel EXTRA en nuestra Wallet.

Tabla de costes de la red Theta (**segunda columna**), actualizada el 11 de junio de 2021 sacada del artículo:

<https://medium.com/theta-network/mainnet-3-0-update-tfuel-burning-and-transaction-fee-increase-f1aa03-b9459f>

	Theta tx fees (current)		Theta tx fees (proposed)		Ethereum
	in TFUEL	in USD	in TFUEL	in USD	
Send transaction	0.000001	\$0.00	0.3	\$0.12	\$1.49
Deploy smart contract	5	\$1.94	20	\$7.74	\$155.44
Interact with smart contract	0.25	\$0.10	1	\$0.39	\$14.18

3. El archivo con el que queremos crear el NFT, formatos admitidos (JPG, PNG, GIF).

Los contratos de Theta son compatibles con los de Ethereum, con lo cual, es posible que también se admitan otros formatos como SVG, MP4, WEBM, MP3, WAV, OGG, GLB o GLTF, ya que en OpenSea sí se permiten, pero no hemos probado estos casos.

¡IMPORTANTE! El archivo ha de subirse previamente a algún servidor que tengamos o alguna página como <https://giphy.com> donde nos podemos crear una cuenta y subir nuestro GIF allí. También podemos subirlo a cualquier cuenta que tengamos de Wordpress o similar para obtener una URL del archivo que se utilizará más adelante.

4. Un IDE (Entorno de Desarrollo Integrado) donde compilar el contrato del NFT. Podemos utilizar Remix desde nuestro navegador accediendo esta dirección:

<https://remix.ethereum.org/>

5. El código fuente del contrato que podemos encontrar en esta dirección:

<https://github.com/thetatoken/theta-nft-demo/blob/main/contracts/nft.sol>



Si tenemos una **Wallet de Theta** con más de **20 Tfuel** y la **URL de nuestro archivo**, podemos comenzar.

PASO 1. Organizamos las pestañas.

Abrimos en nuestro navegador 5 pestañas con las siguientes URL:

Pestaña 1: La URL de nuestro archivo para crear el NFT

Pestaña 2: <https://github.com/thetatoken/theta-nft-demo/blob/main/contracts/nft.sol>

Pestaña 3: <https://remix.ethereum.org/>

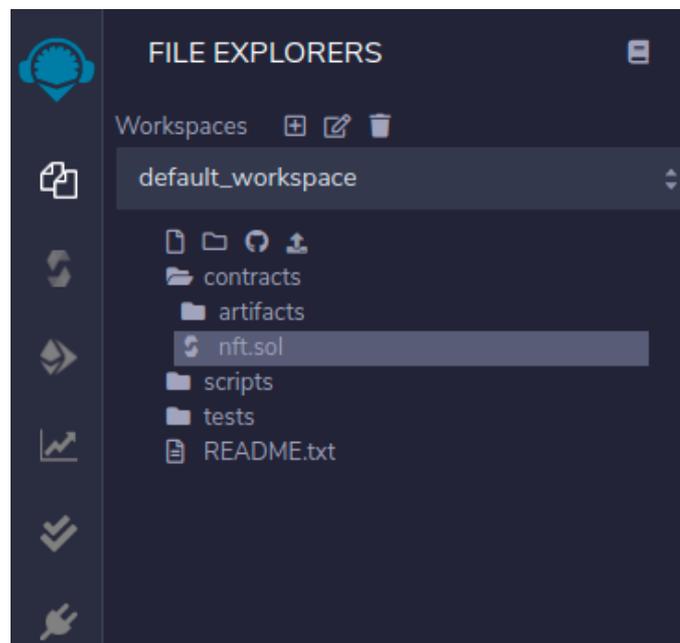
Pestaña 4: <https://wallet.thetatoken.org/>

Pestaña 5: <https://explorer.thetatoken.org/>

En la **pestaña 1** tenemos nuestro **archivo**, en la **pestaña 2** el **contrato del NFT**, en la **pestaña 3** el **IDE Remix**, en la **pestaña 4** nuestra **Wallet** y en la **pestaña 5** el **Explorador de Theta**.

PASO 2. Abrimos y preparamos el IDE Remix.

Vamos a la pestaña 3 donde tenemos el IDE Remix y nos fijamos que tengamos en la parte izquierda una estructura de archivos similar a la de la imagen. Puede que por defecto tengamos algunos archivos ya creados que podemos eliminar. Nos creamos un archivo que se llame **nft.sol** dentro de la carpeta **contracts**.



PASO 3. Copiamos y pegamos el código del contrato.

Cogemos el código fuente del contrato de la pestaña 2 y lo pegamos dentro del archivo nft.sol de la pestaña 3.

¡IMPORTANTE! Asegúrate de haber seleccionado el código hasta abajo del todo, esta versión 0.6.2 del contrato tiene 702 líneas, puede que en futuras versiones del código tenga diferente número de líneas.



```

FILE EXPLORERS
Workspaces
default_workspace
contracts
artifacts
nft.sol
scripts
tests
README.txt

1 // SPDX-License-Identifier: MIT
2 //
3 // TNT-721 Non-Fungible Token Implementation based on the OpenZeppelin Lib
4 //
5
6 pragma solidity ^0.6.2;
7
8 abstract contract Context {
9     function _msgSender() internal view virtual returns (address payable) {
10         return msg.sender;
11     }
12
13     function _msgData() internal view virtual returns (bytes memory) {
14         this; // silence state mutability warning without generating bytecode - see https://github.com/ethereum/solidity/issues/2691
15         return msg.data;
16     }
17 }
18
19 library SafeMath {
20     function add(uint256 a, uint256 b) internal pure returns (uint256) {
21         uint256 c = a + b;
22         require(c >= a, "SafeMath: addition overflow");
23     }
24
25     function sub(uint256 a, uint256 b) internal pure returns (uint256) {
26         return sub(a, b, "SafeMath: subtraction overflow");
27     }
28
29     function sub(uint256 a, uint256 b, string memory errorMessage) internal pure returns (uint256) {
30         require(b <= a, errorMessage);
31         uint256 c = a - b;
32         return c;
33     }
34
35     function mul(uint256 a, uint256 b) internal pure returns (uint256) {
36         // Gas optimization: this is cheaper than requiring 'a' not being zero, but the
37         // benefit is lost if 'b' is also tested.
38         // See: https://github.com/OpenZeppelin/openzeppelin-contracts/pull/522
39         if (a == 0) {
40             return 0;
41         }
42         uint256 c = a * b;
43         require(c / a == b, "SafeMath: multiplication overflow");
44     }
45
46     function div(uint256 a, uint256 b) internal pure returns (uint256) {
47         return div(a, b, "SafeMath: division by zero");
48     }
49
50     function div(uint256 a, uint256 b, string memory errorMessage) internal pure returns (uint256) {
51         require(b > 0, errorMessage);
52         uint256 c = a / b;
53         require(c * b <= a, errorMessage);
54     }
55 }

```

PASO 4. Compilamos el contrato.



Para ello vamos al segundo icono del menú de la izquierda y seleccionamos las opciones que hay en la imagen:

Seleccionamos la versión del **COMPILADOR** (en este caso la versión 0.6.2 igual que la del contrato pero puede cambiar en futuras versiones). Podemos ver la versión del contrato en la parte superior del código fuente:

```

1 // SPDX-License-Identifier:
2 //
3 // TNT-721 Non-Fungible Tok
4 //
5
6 pragma solidity ^0.6.2;|
7
8 abstract contract Context {

```

LANGUAGE: Solidity

EVM VERSION: compiler default

Activamos la opción de **“Enable optimization” 200**. Esta opción es para generar menos código y ahorrarnos costes de gas.

Cuando lo tengamos le damos al botón azul de **“Compile”**.



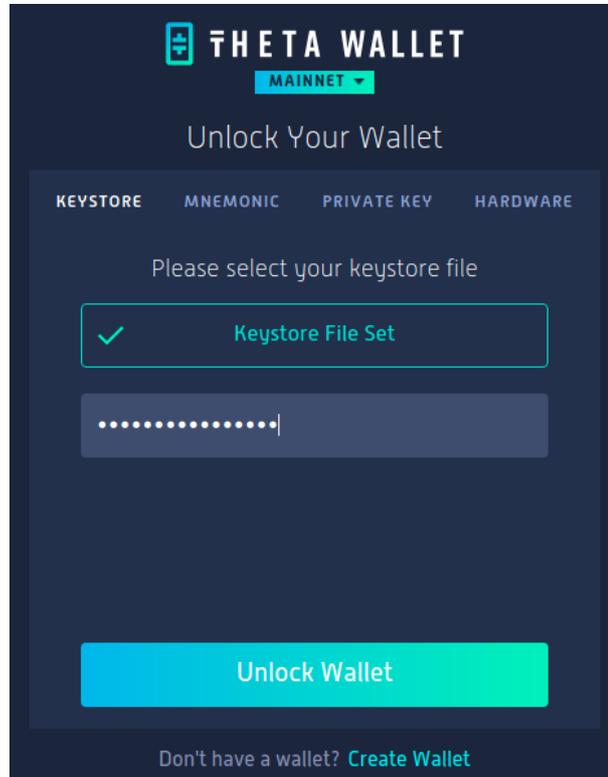
El proceso dura unos segundos y nos aparecerá en el icono del menú de la izquierda un check en verde y en la parte de abajo unas nuevas opciones. En el primer desplegable seleccionamos **CoolINFT**.



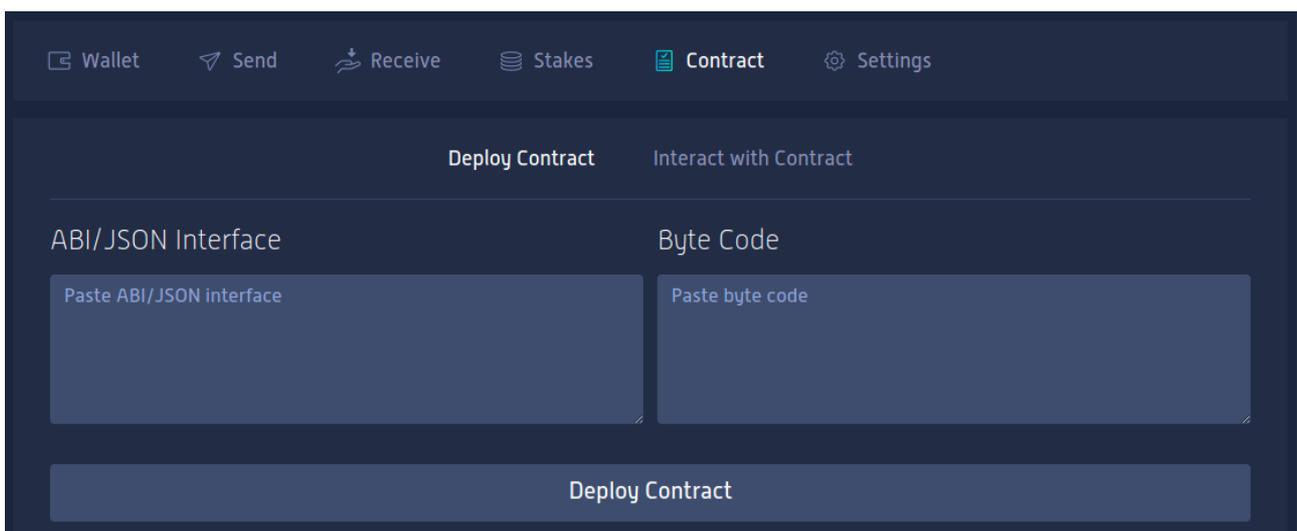


PASO 5. Accedemos a nuestra Wallet para crear el NFT.

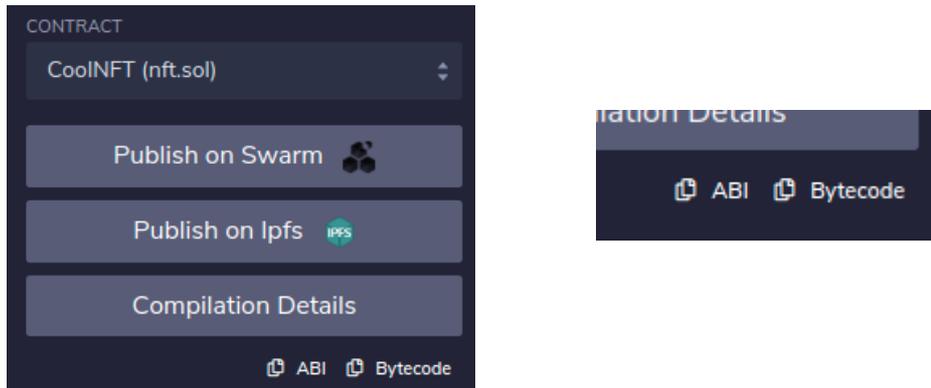
Nos aseguramos que tenemos seleccionada la opción de **“Mainnet”** en la parte superior.



Dentro de nuestra Wallet vamos al apartado de **“Contract”** y la primera pestaña **“Deploy Contract”**. Tenemos 2 campos donde pegaremos 2 textos que copiaremos de la pestaña 3 del IDE Remix.

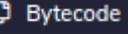


Vamos a Remix y en las nuevas opciones que nos han aparecido en la izquierda cuando hemos compilado el contrato, abajo del todo tenemos lo siguiente:

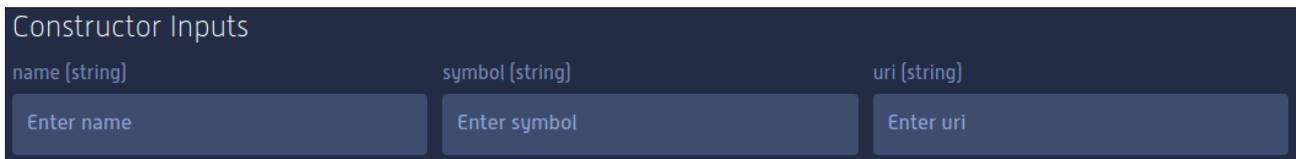


Nos aseguramos que tenemos seleccionada la opción “CoolNFT”.

 Hacemos clic sobre el icono “ABI” para copiar un texto que se ha generado al compilar el contrato, y lo pegamos en el campo “ABI / JSON Interface” de nuestra Wallet.

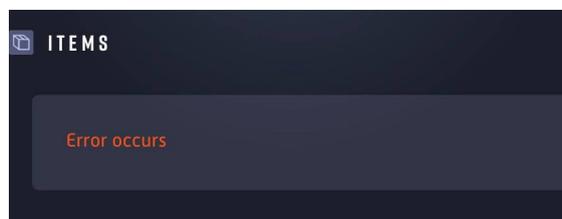
 Hacemos lo mismo con el icono de “Bytecode” y lo pegamos en el campo “Byte Code”.

Al pegar los textos nos aparecerán 3 campos nuevos:



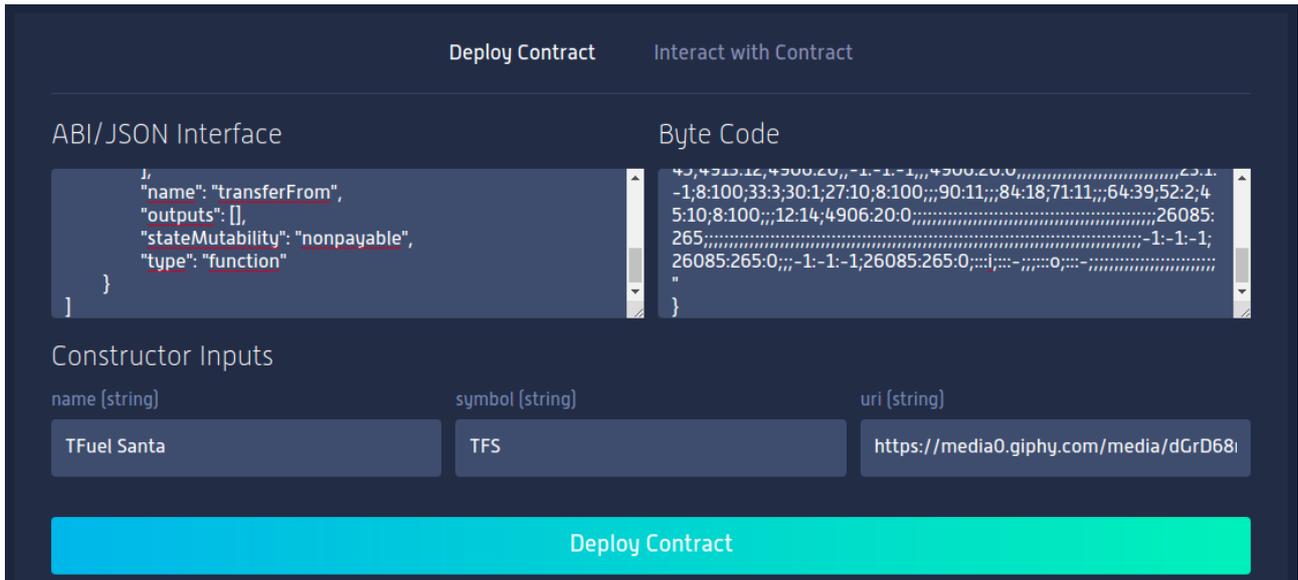
En el primer campo “name” pondremos el nombre que queramos darle a nuestro NFT, en el segundo campo “symbol” una abreviatura del nombre o símbolo que queramos darle al NFT y en el campo “uri” pegaremos la URL de nuestra imagen con la que queremos crear el NFT.

¡IMPORTANTE! Asegúrate de incluir el **https://** delante de la URL, si no, se creará mal el NFT. No dará ningún error al crearlo, pero cuando lo queramos ver, en lugar de ver el NFT nos aparecerá el siguiente mensaje en el explorador de Theta:



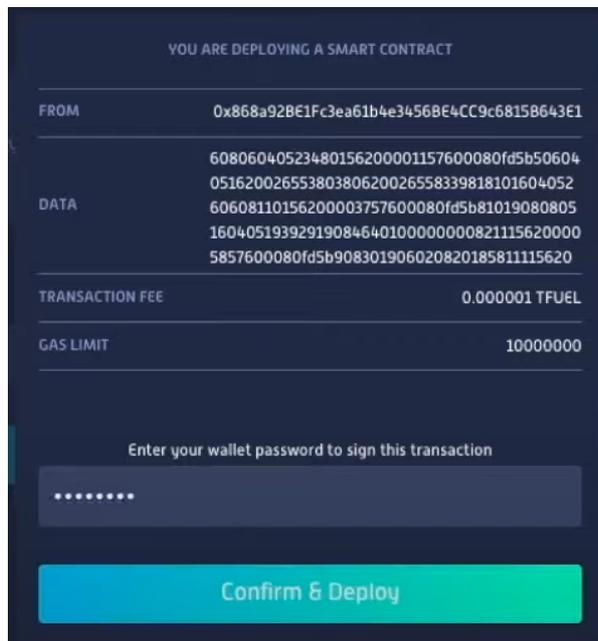


Ha de quedarnos algo similar a esto:



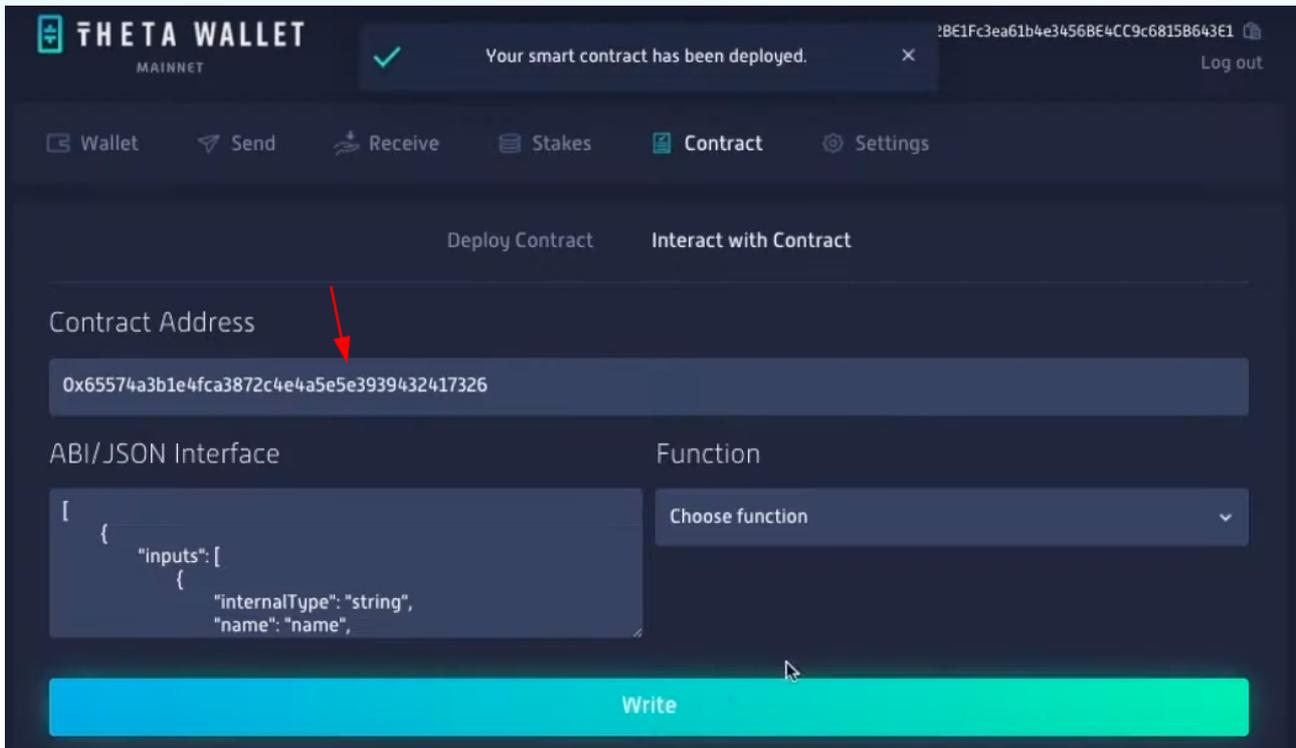
Cuando lo tengamos le damos al botón azul de **“Deploy Contract”**.

Nos aparecerá una ventana donde tenemos que poner nuestra contraseña (la misma que hemos puesto cuando hemos accedido a la Wallet), y le damos a **“Confirm & Deploy”**. Puede tardar unos 15 segundos aproximadamente.



* El TRANSACTION FEE puede que sea más elevado a como aparece en la imagen, es del vídeo del tutorial que se creó antes de la subida de los Fees.

Si todo sale bien, nos aparecerá el siguiente mensaje en la parte de arriba: **“Your smart contract has been deployed”**.



Automáticamente nos lleva a la segunda pestaña de **“Interact with Contract”** donde nos muestra la dirección de nuestro contrato (la que selecciona la flecha roja), y en el desplegable de la derecha nos da una serie de funciones que podemos hacer con nuestro contrato.

PASO 6. Validamos nuestro NFT.

Copiamos la dirección de nuestro contrato de NFT, vamos a la pestaña 5 de nuestro navegador donde tenemos el explorador de Theta, lo pegamos en el campo de arriba a la derecha y le damos al icono de la lupa.





Nos llevará a nuestro Smart Contract:

The screenshot shows the 'ACCOUNT DETAIL' page for a Theta account. At the top, the account address is `Ox65574a3b1e4fca3872c4e4a5e5e3939432417326`. The balance is `0 Theta [$0 USD]` and `0 TFuel [$0 USD]`. The sequence number is `0`.

Below the account details is a 'Transactions' section. A single transaction is listed with the following details:

TYPE	TXN HASH	BLOCK	AGE	FROM	TO	VALUE
Smart Contract	<code>0x393632f4016d0d6...</code>	9882023	6 minutes	<code>0x868a92be1fc3ea6...</code>	<code>0x0000000000000000...</code>	0 Theta [\$0 USD] 0 TFuel [\$0 USD]

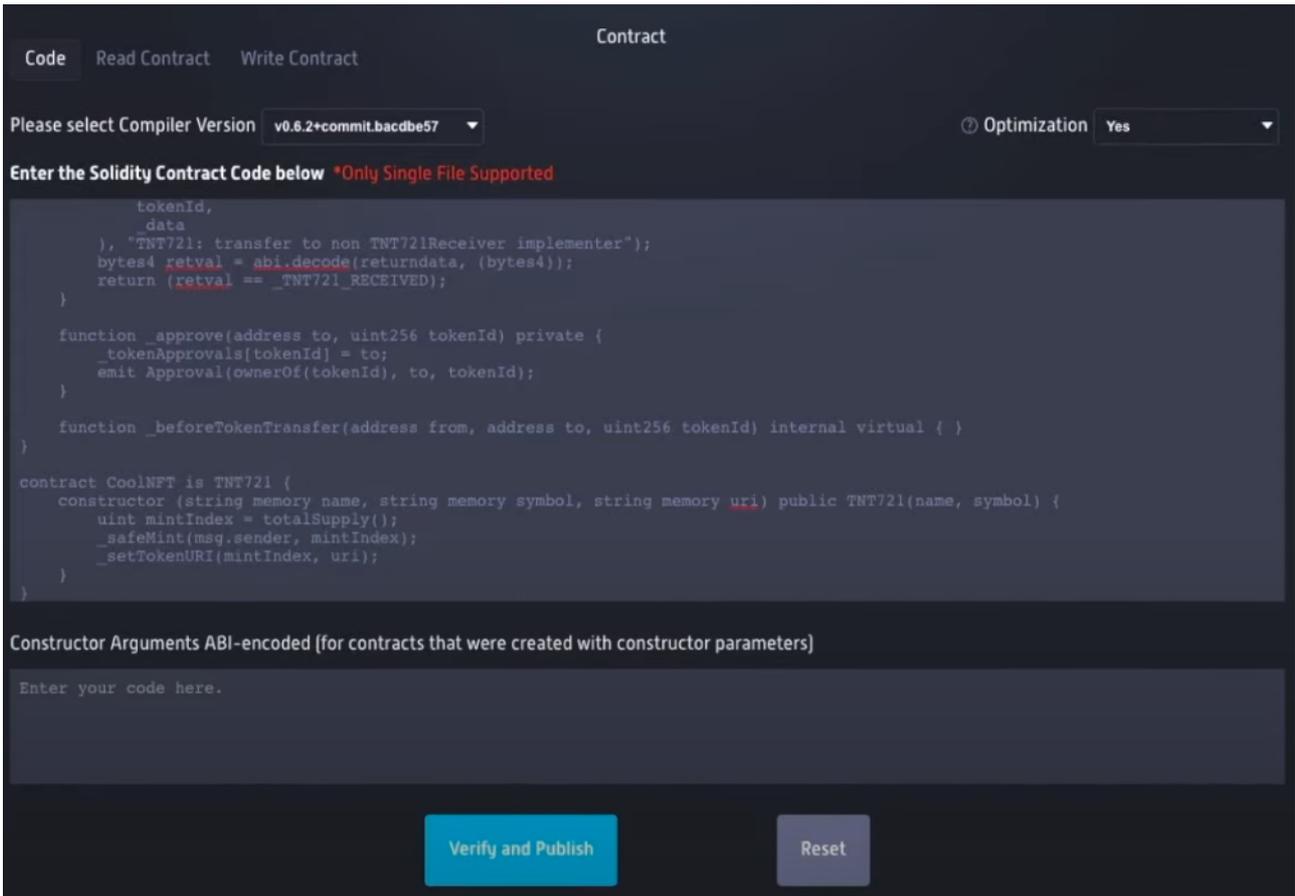
Below the transactions is a 'Contract' section with tabs for 'Code', 'Read Contract', and 'Write Contract'. The 'Code' tab is selected. It features a dropdown menu for 'Please select Compiler Version' (currently set to '[Please select]') and a dropdown for 'Optimization' (currently set to 'No'). Below these is a text input field with the placeholder 'Enter your code here.' and a red warning message: 'Enter the Solidity Contract Code below *Only Single File Supported'.

Ahora tenemos que volver a la pestaña 2 donde tenemos el código fuente del contrato (el mismo que hemos copiado y pegado cuando hemos compilado el código fuente en Remix), seleccionarlo de nuevo y pegarlo en el campo donde pone **“Enter your code here.”**

Seleccionamos la versión de nuestro compilador en el desplegable **“Please select Compiler Version”** **v0.6.2+commit.bacdbe57** y la **“Optimization”** la ponemos en **Yes**, para optimizar el código y reducir el gas.

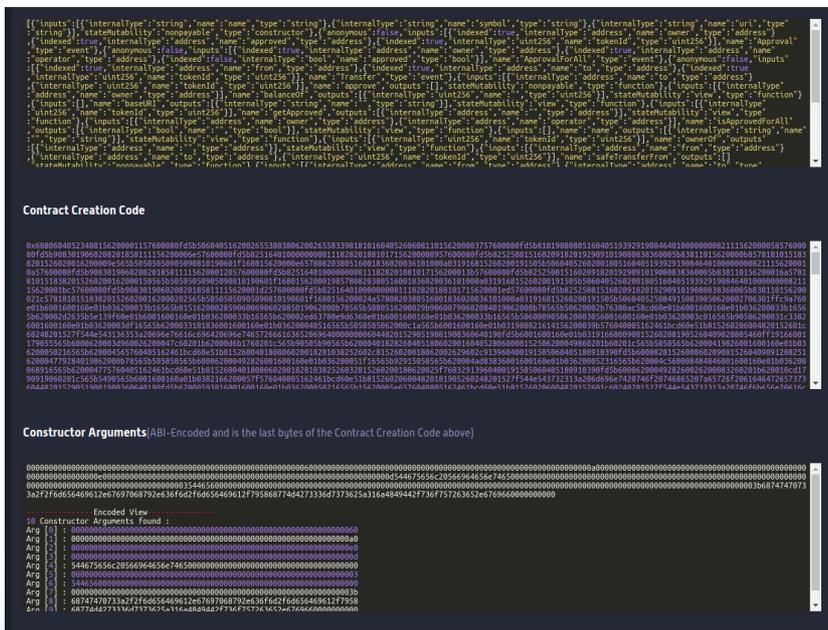


Nos ha de quedar algo similar a como está en la imagen:



Cuando lo tenemos le damos al botón azul de “Verify and Publish”. El proceso dura aproximadamente unos 10 segundos.

Si todo sale bien ya tendremos nuestro NFT creado y nos mostrará la siguiente salida como se muestra en la siguiente imagen.





Para visualizar nuestro NFT solamente debemos hacer clic sobre el número de la transacción **TXN HASH**

Contract

Code Read Contract Write Contract

Contract Source Code Verified

Contract Name: CoolNFT Optimization Enabled: Yes with 200 runs
Compiler Version: v0.6.2+commit.bacdbe57 Other Settings: default evmVersion

Contract Source Code (Solidity)

```
1 - /**
2 -  *Submitted for verification at thetadoken.org on 2021-06-21
3 -  */
4 -  *SPDX-License-Identifier: MIT
5 -  *
6 -  *TNT-721 Non-Fungible Token Implementation based on the OpenZeppelin Lib
7 -  *
8 -  */
9 -  pragma solidity ^0.6.2;
10 -
11 -  abstract contract Context {
12 -    function _msgSender() internal view virtual returns (address payable) {
13 -      return msg.sender;
14 -    }
15 -
16 -    function _msgData() internal view virtual returns (bytes memory) {
17 -      hint: // please state mutability warning without generating bytecode - see https://github.com/ethereum/solidity/issues/2691
18 -      return msg.data;
19 -    }
20 -  }
21 -
22 -  library SafeMath {
23 -    function add(uint256 a, uint256 b) internal pure returns (uint256) {
24 -      uint256 c = a + b;
25 -      require(c >= a, "SafeMath: addition overflow");
26 -
27 -      return c;
28 -    }
29 -
30 -    function sub(uint256 a, uint256 b) internal pure returns (uint256) {
31 -      return sub(a, b, "SafeMath: subtraction overflow");
32 -    }
33 -
34 -    function sub(uint256 a, uint256 b, string memory errorMessage) internal pure returns (uint256) {
35 -      require(b <= a, errorMessage);
36 -      uint256 c = a - b;
37 -
38 -      return c;
39 -    }
40 -
41 -    function mul(uint256 a, uint256 b) internal pure returns (uint256) {
42 -      // Gas optimization: this is cheaper than computing 'a' not being zero, but the
```

Transactions

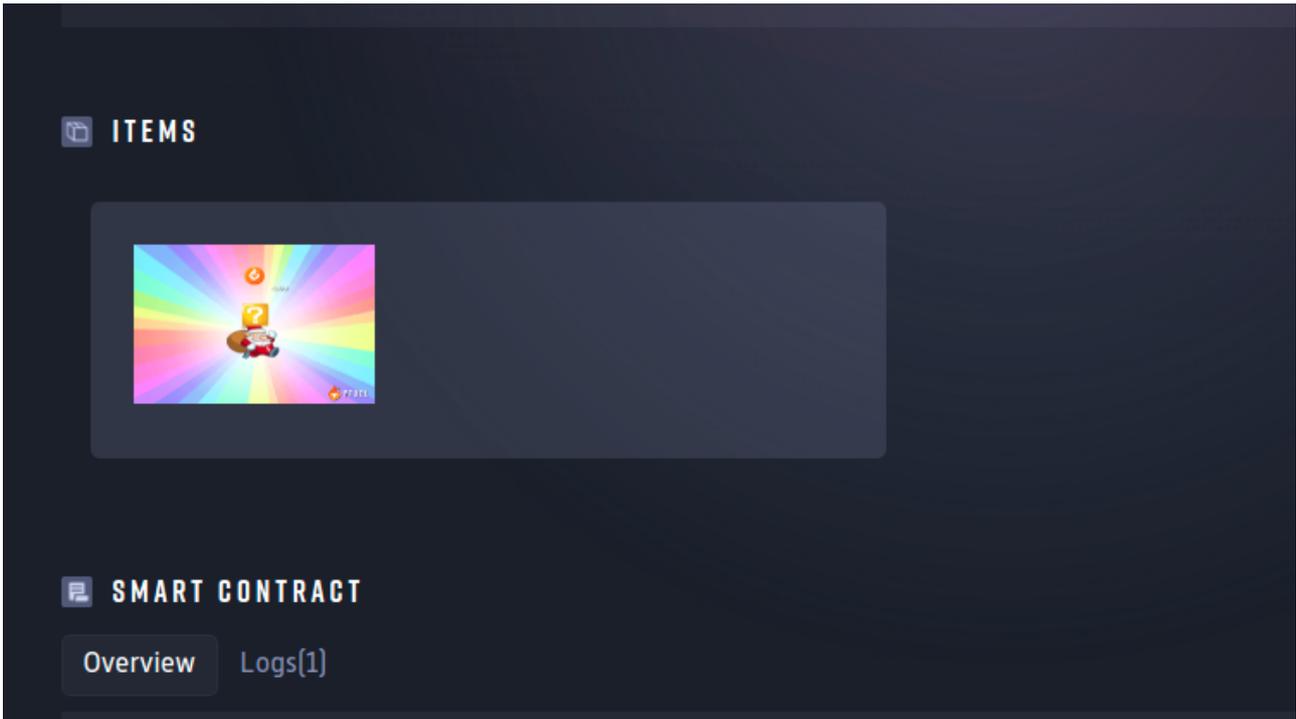
Display 1 selected types TxS

TYPE	TXN HASH	BLOCK	AGE	FROM	TO	VALUE
Smart Contract	0x393632f4016d0d6...	9882023	6 minutes	0x868a92be1fc3ea6...	0x0000000000000000...	0Theta [\$0 USD] 0TFuel [\$0 USD]

Contract

Code Read Contract Write Contract

Contract Source Code Verified

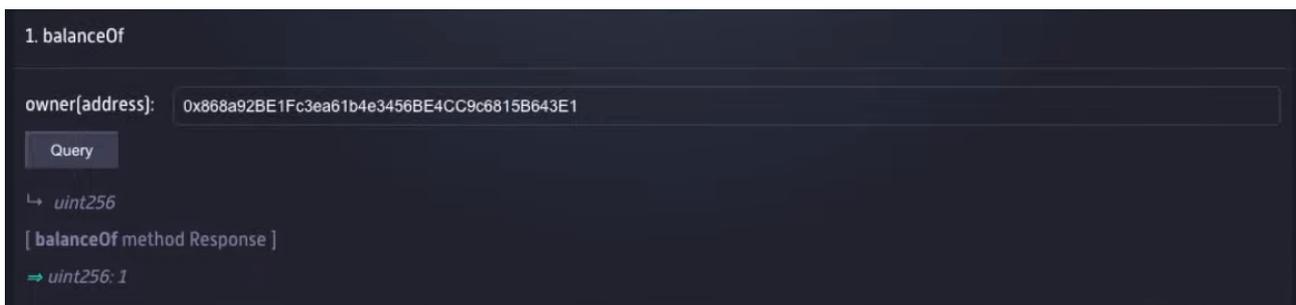


LECTURA DE NUESTRO SMART CONTRACT

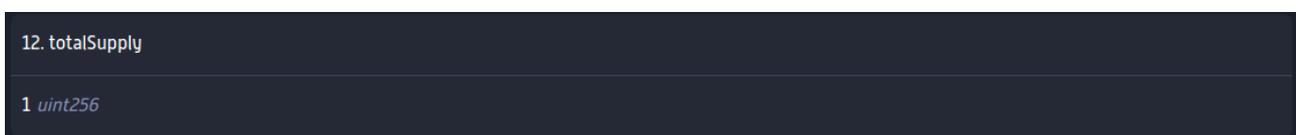
Podemos realizar varias consultas de nuestro smart contract.

VER EL NÚMERO DE NFTs CREADOS

Por defecto se crea solamente **1**. Para ello vamos a la pestaña de **“Read contract”** y en el campo **1 “balanceOf”**, en **“owner(address)”** pegamos la dirección de nuestra Wallet (no del smart contract), y le damos al botón de **“Query”**. Nos da como salida → **uint256: 1**



También podemos ver el supply o suministro total de ese NFT en el campo número **12 “totalSupply”**.





VER EL NOMBRE DEL NFT

En el campo número **5** “**name**” nos muestra el nombre.

```
5. name
TFuel Santa string
```

VER EL PROPIETARIO DEL NFT

Por defecto los propietarios seremos nosotros, pero si le enviamos ese NFT a otra persona, dejaremos de serlo. Para ver el propietario actual vamos al campo número **6** “**ownerOf**” y en “**tokenId(uint256)**” escribimos un **0** que es el valor del **token ID**, y le damos al botón de “**Query**”. Nos mostrará un mensaje:

→ **address: 0x868...** que es la wallet del propietario de ese NFT.

```
6. ownerOf
tokenId(uint256): 0
Query
↳ address
[ ownerOf method Response ]
⇒ address: 0x868a92be1fc3ea61b4e3456be4cc9c6815b643e1
```

VER EL SÍMBOLO DEL NFT

En el campo número **8** “**symbol**” nos muestra el símbolo que le hemos puesto al NFT.

```
8. symbol
TFS string
```

VER LA URL DEL ARCHIVO UTILIZADO PARA CREAR EL NFT

En el campo número **11** “**tokenUri**” escribimos en “**tokenId(uint256)**” un **0**. Nos mostrará la URL que le pusimos a la hora de crear el NFT.

→**string: https://media0.giphy.com/media/dGrD68rMnOev0udvFR/giphy.gif**

```
11. tokenURI
tokenId(uint256): 0
Query
↳ string
[ tokenURI method Response ]
⇒ string: https://media0.giphy.com/media/dGrD68rMnOev0udvFR/giphy.gif
```



ENVIAR EL NFT A OTRA WALLET

Para enviar el NFT creado a otra wallet ya sea nuestra o de otra persona a la que se lo queramos regalar, hemos de ir a nuestra wallet, al apartado de “**Contract**” y “**Interact with Contract**”.

¡IMPORTANTE! Si lo hacemos justo después de crear el NFT, es posible que ya esté relleno el campo de “**ABI/JSON Interface**” de haber seguido el proceso previo, pero si lo queremos enviar en cualquier otro momento o si salimos y entramos de nuestra wallet, el campo de “**ABI/JSON Interface**” nos aparecerá vacío, entonces deberemos **repetir los pasos 3 y 4** de compilar el código fuente del Smart Contract para obtener el texto del código “**ABI**” y pegarlo en este campo.

En el campo “**Function**” seleccionaremos la opción de “**transferFrom**”.

Nos aparecerán unos campos en la parte de abajo “**Function Inputs**” que rellenaremos:

from(address): Pondremos la dirección de nuestra wallet.

to(address): Pondremos la dirección de la wallet a la que se lo queramos enviar.

tokenId(uint256): El ID del token que es 0.

THETA WALLET MAINNET My Wallet: 0x868a928E1Fc3ea61b4e34568E4CC9c6815B643E1 Log out

Wallet Send Receive Stakes Contract Settings

Deploy Contract Interact with Contract

Contract Address

0x65574a3b1e4fca3872c4e4a5e5e3939432417326

ABI/JSON Interface Function

```
[ { "inputs": [ { "internalType": "string", "name": "name",
```

transferFrom

Function Inputs

from [address] to [address] tokenId [uint256]

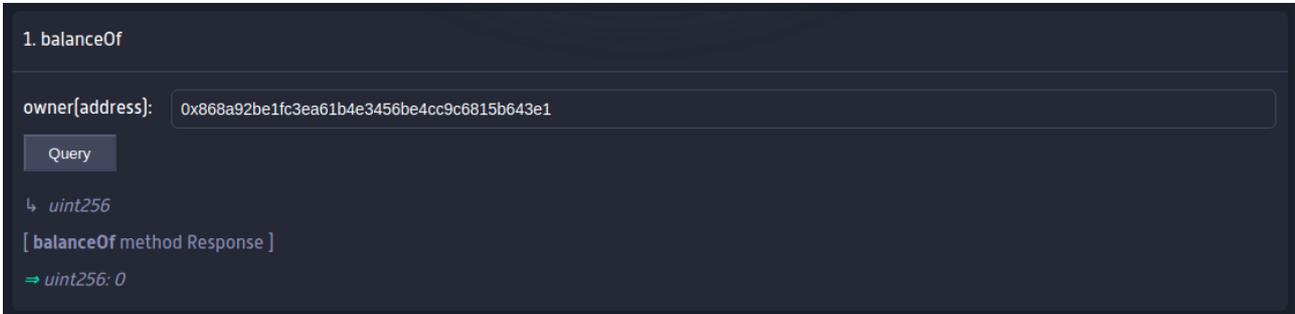
0x868a928E1Fc3ea61b4e34568E4CC9c681 0x91999B5003EFDA57FBF801AC65EF2423 0

Write

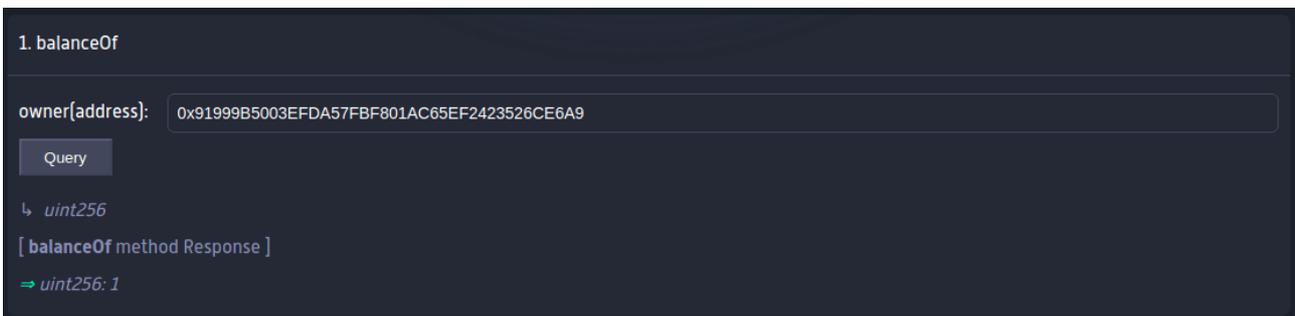


VER EL NÚMERO DE NFTs DE UN SMART CONTRACT DE LA WALLET

Si ahora realizamos la consulta del campo número 1, si ponemos la dirección de la wallet que creó el NFT, la nuestra, nos dará como salida → **uint256: 0**. Eso quiere decir que de ese Smart Contract poseemos 0 NFTs.



Sin embargo, si ponemos la wallet a la que hemos enviado el NFT nos indicará que tiene 1 NFT de ese Smart Contract.



VER EL PROPIETARIO DEL NFT

Si comprobamos el propietario del NFT en el campo 6, ahora nos aparecela wallet a la que se lo hemos enviado.

