



Bloque 3 · Despliegue del MMDV Wine Token en la Blockchain (Sepolia)



Del código a la red: cómo desplegar, verificar y mintear tu primer token de vino en Ethereum.

☒ 1. Punto de partida: qué vamos a hacer en este bloque

Hasta ahora hemos trabajado sobre conceptos y diseño. En el **Bloque 1** descubrimos qué significa tokenizar un activo real. Aprendimos que tokenizar es representar valor en una blockchain, y que ese valor puede ser cualquier cosa: desde una obra de arte hasta un lote de vino. En el **Bloque 2**, dimos el siguiente paso: aprendimos cómo se crea un token fungible (ERC-20), diseñamos el **MMDV Wine Token (MMWT)** y entendimos su lógica interna: cap, supply, funciones esenciales, eventos de trazabilidad y estructura técnica.

Ahora comienza la parte más emocionante del viaje: convertir lo que hasta ahora ha sido teoría en un activo real viviendo en una blockchain pública. En otras palabras:

En este bloque vamos a **desplegar la primera versión real del MMDV Wine Token** en la red de pruebas Sepolia, verificaremos que el contrato existe, y crearemos los primeros tokens reales que representarán un lote concreto de vino tokenizado.

Este bloque es el puente entre la idea y la ejecución. Despues de completarlo, cualquier alumno podrá decir: “He desplegado mi propio token en Ethereum. Puedo verlo en Etherscan y puedo demostrar su despliegue en la blockchain”

Qué aprenderás concretamente en este bloque

Aprenderás a:

- Preparar el entorno mínimo para trabajar con la blockchain (wallet, red de pruebas y ETH de prueba).
- Utilizar Remix, una herramienta accesible desde el navegador, para cargar, compilar y desplegar contratos inteligentes.
- Conectar Remix con tu wallet en la red Sepolia.
- Desplegar el contrato del MMWT.
- Verificar el contrato en Etherscan para que sea público, transparente y accesible.
- Añadir el token a tu wallet para visualizar tus tokens.
- Realizar tu *primer mint* real de un lote de vino tokenizado.
- Interpretar el resultado en blockchain: eventos, balances, transacciones y trazabilidad.



Y lo más importante: **entenderás cada paso, por qué se hace y qué significa.**

2. Preparación del entorno: los cimientos del despliegue

Antes de trabajar con un contrato real, necesitamos preparar el entorno mínimo que nos permita interactuar con la blockchain de forma segura y sencilla. Si este es tu primer contacto con Ethereum, esta fase es esencial.

2.1 La wallet: Metamask

Metamask es una extensión para navegador que funciona como tu “billetera digital”. No solo guarda claves: te permite **firmar transacciones**, conectar con aplicaciones descentralizadas (DApps) y gestionar sus tokens.

Para configurarlo:

1. Descarga la extensión desde la web oficial de Metamask.
2. Crea tu wallet, guarda tu frase semilla en un lugar seguro fuera del ordenador.
3. Una vez dentro, verás tu dirección pública (empieza por 0x...), que será tu identidad en Ethereum.

En principio, Metamask se conecta a la red principal de Ethereum, pero para este proyecto usaremos la red de pruebas Sepolia.

2.2 La red: Sepolia

Sepolia es una **testnet oficial de Ethereum**, una red pública donde:

- las transacciones funcionan como en la red principal,
- pero el ETH que se usa es de prueba y no tiene valor.

Usarla nos permite experimentar sin riesgos.

Para activarla:

1. Abre Metamask.
2. En el selector de redes, elige “**Sepolia**”.
 - Si no aparece, tendrás que añadirla manualmente o permitir a Metamask hacerlo por ti.

Una vez activa, Metamask mostrará un balance en ETH... probablemente cero al principio.

2.3 ETH de prueba

Cada interacción con Ethereum tiene un coste en “gas”, incluso en una testnet.

Para desplegar el contrato y mintear tokens necesitamos ETH de prueba.

La forma de conseguirlo es mediante un faucet.

Un faucet es una web que “regala” pequeñas cantidades de ETH de prueba. Basta con pegar tu dirección de Sepolia, pasar un pequeño captcha y recibirás ETH en segundos.

Verás cómo tu Metamask pasa a mostrar algo así como 0.1 SepoliaETH.

Y con esto ya tienes lo necesario para entrar en el mundo real de los contratos.

3. Las herramientas clave del despliegue: Remix y Etherscan

Para poder desplegar el contrato necesitamos dos herramientas principales: **Remix** y **Etherscan**. Este bloque será la primera vez que las toquemos de verdad, así que las vamos a presentar como lo haríamos con alguien que nunca ha trabajado con blockchain.

Remix es un entorno de desarrollo accesible desde cualquier navegador. No hace falta instalar programas ni configurar nada complejo. Desde Remix podremos abrir el contrato, compilarlo y desplegarlo directamente en Ethereum (en nuestro caso, en la red de pruebas Sepolia). Es, por decirlo así, el “taller” donde nace el token.

Etherscan, por otro lado, es la puerta de cristal. Es la herramienta que nos permite ver todo lo que sucede dentro de la blockchain. Cuando despleguemos el contrato, Etherscan será la página donde podremos comprobar que existe, ver sus transacciones, visualizar el código y hasta ejecutar funciones desde su interfaz.



The screenshot shows the Remix IDE on the left and the Etherscan interface on the right. In Remix, under the 'DEPLOY & RUN' tab, the environment is set to Sepolia, the account has 2.618136565³ ETH, and the value is set to 0 wei. A green 'Deploy' button is visible. In the Etherscan interface, the 'Contract Verification' section shows a green checkmark next to 'Contract Verified'. It provides details about the contract creator (0xD0Ebb167Ba44225F8B1B26AAA4...), the block number (453064), the transaction hash (#458084), and the contract name (MMDVWineToken (ERC-20 (Solidity))). The 'Metadata' tab is selected, showing the compiler as Solidity (0.8.25) and the license as MIT.

4. Cargar el contrato en Remix y compilarlo

Una vez que entramos en Remix, lo primero que vemos es una interfaz con varios paneles. A la izquierda tenemos un explorador de archivos vacío; en el centro, el editor de texto; y a la derecha, las herramientas para compilar y desplegar.

El primer paso consiste en crear un archivo nuevo. Lo llamamos MMDVWineToken.sol, y dentro pegamos el contrato que construimos en el Bloque 2. Ese contrato contiene exactamente lo que necesitamos para comenzar: las funciones del token ERC-20, los eventos de trazabilidad, el cap y los permisos de propietario.

Cuando el archivo está listo, pasamos a compilarlo. La compilación es simplemente la forma que tiene Remix de verificar que el código está bien escrito y que no hay errores que impidan su funcionamiento. Seleccionamos la misma versión de Solidity que aparece en la cabecera del contrato y pulsamos compilar. Si hay un check verde, significa que el contrato está listo para ir a la blockchain.



```
// SPDX-License-Identifier: MIT
pragma solidity ^0.8.24;

import "openzeppelin/contracts/token/ERC20/extensions/ERC20Capped.sol";
import "openzeppelin/contracts/token/ERC20/extensions/ERC20Burnable.sol";
import "openzeppelin/contracts/token/ERC20/extensions/ERC20Mintable.sol";
import "openzeppelin/contracts/token/ERC20/extensions/ERC20Pausable.sol";

contract MMDVWineTokenV3 is ERC20, ERC20Capped, ERC20Burnable, ERC20Mintable, Ownable {
    // Cap EDUCATIVO: 1.000.000 tokens ENEROES (1 token = 1 unidad de vino)
    uint256 public constant MAX_SUPPLY = 1_000_000;

    // Precios simbólico educación: 10 "tokens" por token (no usamos euros on-chain)
    uint256 public constant BASE_PRICE_UNITS = 10;

    // Guardar cuándo compró cada dirección y cuándo se puso a holder
    struct Holding {
        uint256 amount;
        uint256 firstPurchaseAt;
    }

    mapping(address => Holding) public holdings;

    constructor() payable gas limit 3000000
        ERC20("MMDV Wine Token V3", "MWT3")
        ERC20Capped(MAX_SUPPLY)
        ERC20Burnable()
        Ownable()
    {
    }

    /// @notice Truncate 8 decimales: 1 token = 1 unidad completa de vino
    function decimals() public pure override returns (uint8) {
        return 0;
    }
}
```

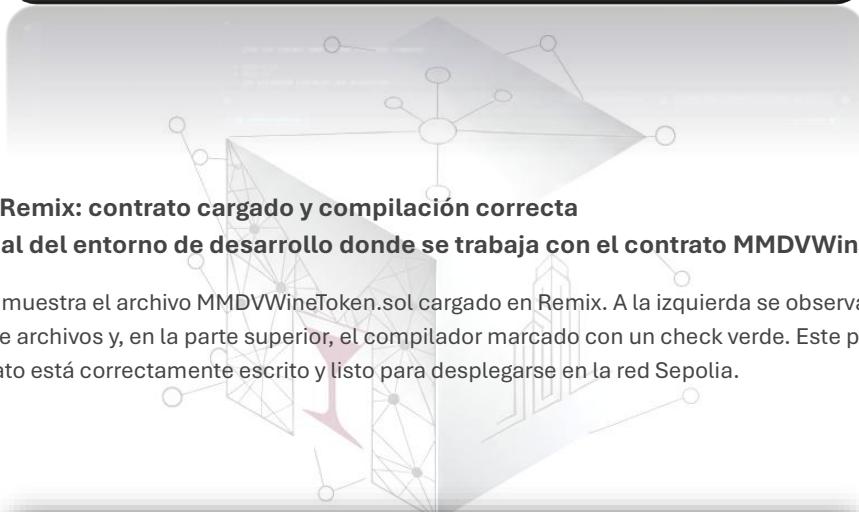


Imagen 2 - Remix: contrato cargado y compilación correcta
(captura real del entorno de desarrollo donde se trabaja con el contrato MMDVWineToken.sol)

Esta imagen muestra el archivo MMDVWineToken.sol cargado en Remix. A la izquierda se observa el explorador de archivos y, en la parte superior, el compilador marcado con un check verde. Este paso confirma que el contrato está correctamente escrito y listo para desplegarse en la red Sepolia.

The screenshot shows the Etherscan interface for the MMDV Wine Token V5 (MWT5) contract. The top navigation bar includes Sepolia Testnet, Home, Blockchain, Tokens, NFTs, and More. The main page displays the following information:

- ERC-20**:
 - Overview**: MAX TOTAL SUPPLY 7,300 MWT5, HOLDERS 4, TOTAL TRANSFERS 7.
 - Market**: ONCHAIN MARKET CAP 0, CIRCULATING SUPPLY MARKET CAP 0.
 - Other Info**: TOKEN CONTRACT (WITH 0 DECIMALS) 0x9e913dEadC1f9c25a2Adc7BAD793cEf72Cd02dC2.
- Transfers**: A table showing 7 transactions found:

Transaction Hash	Method	Block	Age	From	To	Amount
0x1d575819e5...	Educational...	9633993	2 days ago	0x00000000...00000000	0xC9d05cdB...2612a9197	5,000
0xbe8f357baf2...	Transfer	9633972	2 days ago	0xC9d05cdB...2612a9197	0x02043aC4...013c37cEa	1,000
0x1a170e5684f...	Owner Buyback	9630676	2 days ago	0x6Ba3EB96...A9E48A2DC	0xC9d05cdB...2612a9197	100
0x1685f7606ee...	Cancel Withdrawal	9628199	2 days ago	0x3764B983...C2cc746d6	0xC9d05cdB...2612a9197	100
0xb32e347452...	Educational...	9627657	3 days ago	0x00000000...00000000	0x3764B983...C2cc746d6	500
0xad2bd4744b...	Educational...	9627652	3 days ago	0x00000000...00000000	0x6Ba3EB96...A9E48A2DC	800
0x8c0719662c...	Educational...	9627646	3 days ago	0x00000000...00000000	0xC9d05cdB...2612a9197	1,000
- Downloads**: Download Page Data, First, <, Page 1 of 1, >, Last.
- Information**: A note states: "A token is a representation of an on-chain or off-chain asset. The token page shows information such as price, total supply, holders and transfers and social links. Learn more about this page in our Knowledge Base." There is also a link to "Download CSV Export".
- Footer**: Powered by Ethereum, Back to Top.



Imagen 3 - Etherscan: contrato desplegado y primeros movimientos

(captura real de la página pública del token del proyecto en Sepolia)

Vista real del token MMDV Wine Token en Etherscan (Sepolia). Aquí pueden verse el suministro total, los holders, las transacciones y la dirección oficial del contrato. Esta pantalla confirma que el token ya está vivo en la blockchain y que cada operación queda registrada de forma transparente y permanente.

5. Conectar Remix con Metamask en la red Sepolia

Con el contrato ya compilado, toca conectarlo con nuestra wallet. Esto es fundamental, porque para desplegar el contrato necesitamos que Remix utilice nuestra cuenta en Sepolia para firmar la transacción.

En la sección “Deploy & Run” de Remix seleccionamos la opción “Injected Provider – MetaMask”. Automáticamente aparecerá una ventana de Metamask pidiendo permiso para conectarse. La aceptamos y comprobamos que estamos en la red **Sepolia**.

A partir de aquí, Remix reconoce nuestra dirección de Sepolia y nos muestra nuestro balance de ETH de prueba. Ese balance será el que pague el gas necesario para desplegar el contrato.

6. El despliegue: el contrato pasa de papel a blockchain

Este es el momento clave del bloque. Hasta ahora todo lo que hemos hecho vivía en el navegador. El despliegue es el paso en el que el contrato se “graba” de forma permanente en la blockchain. A partir de ese instante ya no depende de nosotros: pasa a ser un objeto autónomo y público, accesible para cualquiera.

En Remix, con la conexión a Metamask ya activa, seleccionamos el contrato MMDVWineToken en el menú desplegable. Si nuestro contrato tiene un constructor —como es el caso— tendremos que introducir tres datos: el nombre del token, el símbolo y el cap máximo. Estos valores se escriben una sola vez y no pueden modificarse después del despliegue.

Después de introducirlos, pulsamos “Deploy”. Metamask abre una ventana mostrando el coste estimado de gas. Confirmamos y esperamos unos segundos. Cuando la transacción se confirma, Remix muestra una nueva sección llamada “Deployed Contracts” donde aparece nuestro contrato con una dirección asociada. Esa dirección es su ubicación en la blockchain.

7. Comprobar el contrato en Etherscan: transparencia total

Con la dirección del contrato copiada, abrimos Etherscan (versión Sepolia) y la pegamos en la barra de búsqueda. Aparece inmediatamente la página pública del contrato. Allí podemos ver:

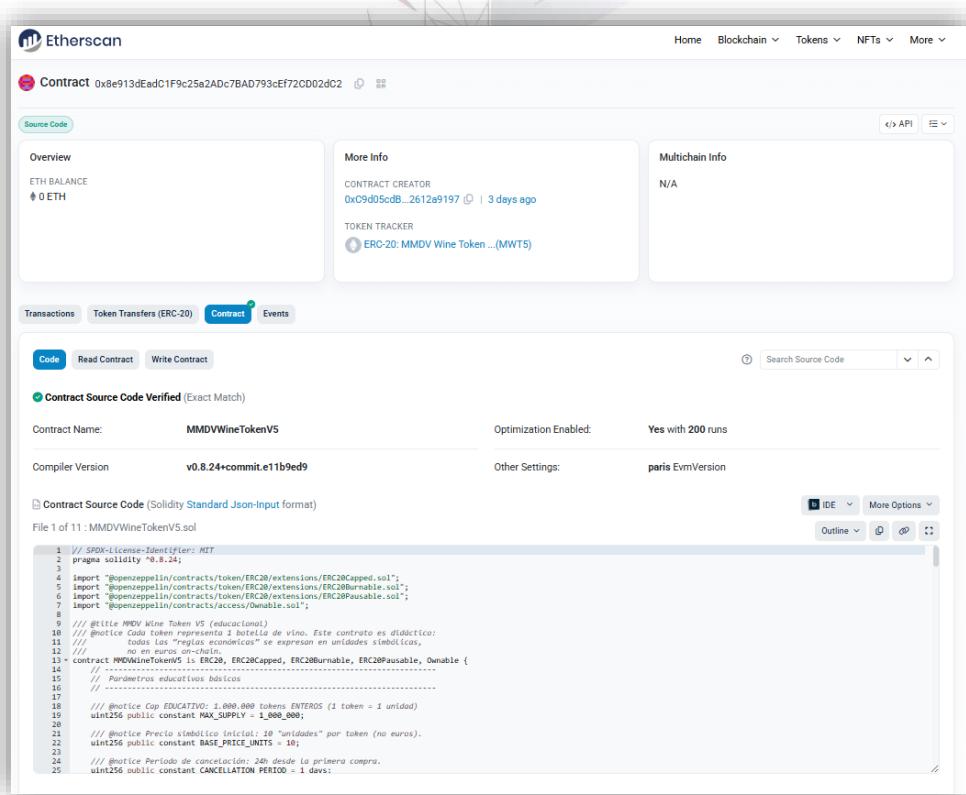
- la transacción de creación,
- el bloque donde se incluyó,
- la dirección que lo desplegó (nuestra wallet),
- y el estado de la transacción.

Esto es importante porque demuestra la esencia de la blockchain: **todo lo que ocurre queda registrado de forma permanente y transparente**.

Aquí es también donde más adelante verificaremos el contrato, escribiremos funciones y veremos los eventos.

Contrato MMDV Wine Token (Sepolia):

<https://sepolia.etherscan.io/address/0x8e913dEadC1F9c25a2ADc7BAD793cEf72CD02dC2#code>



The screenshot shows the Etherscan interface for the MMDV Wine Token V5 contract. The top navigation bar includes Home, Blockchain, Tokens, NFTs, and More. The main content area has tabs for Overview, More Info, Multichain Info, Transactions, Token Transfers (ERC-20), Contract (selected), and Events. The Contract tab shows the contract address (0x8e913dEadC1F9c25a2ADc7BAD793cEf72CD02dC2) and a green badge indicating "Contract Source Code Verified (Exact Match)". Below this, it lists the contract name (MMDVWineTokenV5), compiler version (v0.8.24+commit.e11b9ed9), and optimization settings (Yes with 200 runs). The source code tab shows the Solidity code for the contract, which includes imports from OpenZeppelin and defines the MMDVWineTokenV5 contract as an ERC20 token with specific parameters like MAX_SUPPLY and BASE_PRICE_INITS.

```

1 // SPDX-License-Identifier: MIT
2 pragma solidity ^0.8.24;
3
4 import "@openzeppelin/contracts/token/ERC20/extension/ERC20Capped.sol";
5 import "@openzeppelin/contracts/token/ERC20/extensions/ERC20Burnable.sol";
6 import "@openzeppelin/contracts/token/ERC20/extensions/ERC20Pausable.sol";
7 import "@openzeppelin/contracts/access/Ownable.sol";
8
9 /// @title MMDV Wine Token V5 (educational)
10 /// @notice Este contrato es un token de vino. Este contrato es didáctico;
11 ///         solo las "reglas económicas" se expresan en unidades simbólicas,
12 ///         no en euros on-chain.
13 ///         El contrato MMDVWineTokenV5 es ERC20, ERC20Capped, ERC20Burnable, ERC20Pausable, Ownable {
14     // Parámetros educativos básicos
15     //
16     //
17     // @notice Cap EDUCATIVO: 1.000.000 tokens ENTEROS (1 token = 1 unidad)
18     uint256 public constant MAX_SUPPLY = 1_000_000;
19
20     // @notice Precio simbólico inicial: 10 "unidades" por token (no euros).
21     uint256 public constant BASE_PRICE_INITS = 10;
22
23     // @notice Período de cancelación: 24h desde la primera compra.
24     uint256 public constant CANCELLATION_PERIOD = 1 days;
25

```



Imagen 4 — Contrato MMDV Wine Token (MWT) verificado en la blockchain

(Captura real del contrato del MMDV Wine Token desplegado en la red de pruebas Sepolia).

En esta vista puede observarse el estado de verificación del código fuente, el nombre exacto del contrato (MMDWTokenV5), la versión del compilador utilizado, la dirección pública del creador y la información general del token.

Esta página confirma que el contrato está activo, accesible públicamente y que su código coincide exactamente con el que fue compilado y desplegado. Desde aquí es posible consultar las funciones, ejecutar operaciones como mint o redeem y revisar todos los eventos emitidos por el contrato.

8. Verificar el contrato en Etherscan

Una vez desplegado, la blockchain sabe que el contrato existe, pero no sabe qué hace. Para que cualquiera pueda leer su código, necesitamos “verificar y publicar” el contrato.

Desde la pestaña “Contract” de Etherscan accedemos a la opción “Verify and Publish”. Allí indicamos:

- la versión exacta de Solidity,
- el tipo de compilación,
- la licencia,

y pegamos el código fuente.

Cuando Etherscan confirma la verificación, la página del contrato se transforma: aparecen nuevas pestañas con el código abierto, funciones de lectura y de escritura, y la posibilidad de interactuar directamente con el contrato sin necesidad de usar Remix.

The screenshot shows the Etherscan interface for the MMDV Wine Token (MWT) contract. At the top, it displays the contract address: 0x6f13d8e0c1f9c2fa24fb7ba07793e7f72040c2. Below this, there are tabs for "Overview", "Transactions", "Token Transfers (ERC20)", and "Events". The "Code" tab is active, showing the Solidity source code for the contract. The code defines a struct for a wine, a mapping of addresses to balances, and several functions: constructor, mint, burn, transfer, and transferFrom. The "More Info" section provides details about the contract, including its deployment date (2021-09-19T07:47:58Z), developer (0x6f13d8e0c1f9c2fa24fb7ba07793e7f72040c2), and the name "MMDV Wine Token - (MWT)". The "Blockchain Info" section shows the current balance as 0.00 ETH. The bottom of the page includes a note about the contract being deployed to a revert context.



Imagen 5 - Lectura del contrato MMDV Wine Token en Etherscan

(Read Contract)

Vista real de la pestaña “Read Contract” en Etherscan, donde pueden consultarse las variables y funciones públicas del contrato MMDV Wine Token (MWT).

Aquí el alumno puede observar valores clave como MAX_SUPPLY, decimals, totalSupply, los parámetros internos del token y los datos asociados a cada dirección.

Esta interfaz permite entender cómo se consulta información directamente desde la blockchain sin necesidad de usar herramientas de desarrollo.

9. Añadir el token MWT a Metamask

Aunque el contrato esté activo, si no añadimos manualmente el token, nuestra wallet no lo mostrará. En Metamask seleccionamos “Importar tokens”, introducimos la dirección del contrato y Metamask completa automáticamente el símbolo y los decimales. Después de confirmar, aparece el token “MWT” en nuestra lista de activos, inicialmente con un saldo de cero.

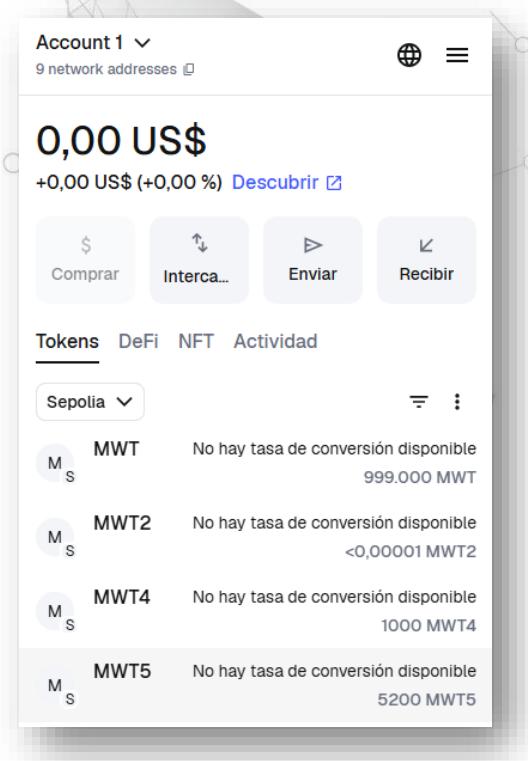


Imagen 6 - Tokens MMDV añadidos a Metamask en la red Sepolia

(Vista real de Metamask mostrando distintos tokens MMDV añadidos manualmente durante el proceso de pruebas y despliegues (MWT, MWT2, MWT4, MWT5)

Esta imagen ilustra cómo aparecen los tokens ERC-20 en la wallet una vez que se ha importado la dirección del contrato, permitiendo visualizar su balance de forma sencilla.



Es una referencia útil para que el alumno reconozca el aspecto exacto de un token personalizado dentro de Metamask y confirme que la importación se ha realizado correctamente.

⌚ 10. Creación del primer lote: nuestro primer mint real

Ahora que el contrato está desplegado y verificado, y el token añadido a nuestra wallet, llega uno de los momentos más especiales de todo el bloque: crear el primer lote de vino tokenizado.

Accedemos a la pestaña “Write Contract” en Etherscan y conectamos nuestra wallet. Dentro encontramos la función mint. Ese formulario nos pide cuatro datos:

- la dirección que recibirá los tokens,
- la cantidad en unidades base (multiplicada por 10^{18}),
- el batchId (por ejemplo, “MWT-RIOJA-2024-L01”),
- y una nota descriptiva del lote.

Al pulsar “Write” Metamask nos pide confirmar la operación. Lo hacemos y, unos segundos después, aparece en Etherscan un evento con toda la información del lote recién creado. Es un momento muy potente porque el lote queda registrado para siempre en la blockchain.

The screenshot shows the Etherscan interface for the Sepolia Testnet. The top navigation bar includes "Home", "Blockchain", "Tokens", "NFTs", and "More". Below the navigation is a search bar and a network status indicator. The main content area is titled "Contract 0x8e913dEadC1F9c25a2AdC7BAD793cEf72CD02d02". It features three tabs: "Source Code", "More Info", and "Multichain Info". The "More Info" tab displays the "CONTRACT CREATOR" as 0xC9d0ScDB_2612a9197 (3 days ago) and the "TOKEN TRACKER" as ERC-20: MMDV Wine Token ... (MWT5). Below these tabs are buttons for "Transactions", "Token Transfers (ERC-20)", "Contracts", and "Events". The "Contracts" tab is currently active. A sub-section titled "Code" shows the Solidity code for the mint function, which includes steps like approve, burn, burnFrom, cancelWithin24h, educationalMint, and the final mint step with parameters "to" and "amount". The "amount" field is set to 1000. At the bottom of this sub-section is a large blue "Write" button.



Imagen 7 - Ejecución del primer mint del MMDV Wine Token en Etherscan

(Captura real de la pestaña “Write Contract” en Etherscan, mostrando la función `educationalMint` con los campos `to` y `amount` llenados)

Desde esta interfaz el usuario puede interactuar directamente con el contrato inteligente y crear un nuevo lote de tokens en la blockchain. Esta operación queda registrada públicamente una vez se firma con la wallet conectada.

The screenshot shows a transaction details page on Etherscan. The transaction hash is 0x1a12d01367723c6eeb78fe8982071ce001b57ed5bcd0a0570b95111f53be466. It was successful, included in block 9648061, and occurred 25 seconds ago on Nov-17-2025 at 11:11:12 AM UTC. The transaction transferred 1,000 MWT5 from 0xC9d05cdBfE0b2611A0b70364DF6c43d2612a9197 to 0x8e913dEadC1F9c25a2Adc7BAD793cEf72CD02dC2. The value transferred was 0 ETH, with a transaction fee of 0.000069799500418797 ETH and a gas price of 1.500000009 Gwei (0.00000000150000009 ETH). The transaction is identified as an ERC-20 token transfer.

Imagen 8 - Transacción de mint confirmada en la blockchain

(Resultado real de la transacción en la que se crearon y asignaron 1.000 MMDV Wine Tokens (MWT5) a la dirección del usuario.)

La pantalla muestra el hash de la transacción, el bloque donde fue incluida, el estado “Success” y los detalles completos del movimiento del token. Esta información confirma que el mint se ejecutó correctamente y que el nuevo lote quedó registrado de forma permanente en la red Sepolia.

11. Los problemas reales que surgieron y cómo los resolvimos

Trabajar con blockchain por primera vez siempre implica encontrarse con pequeñas situaciones inesperadas. No son fallos del alumno ni del sistema, sino parte natural del proceso. En este proyecto, al desplegar y empezar a interactuar con el contrato del MMDV Wine Token, aparecieron varios de esos momentos. Documentarlos es fundamental, porque son exactamente los que cualquier persona verá en su primer contacto con una red como Sepolia.



Aquí tienes, de forma clara y detallada, todo lo que nos sucedió y qué aprendimos de cada situación.

11.1. Cuando el primer mint “fallaba” sin fallar realmente

Uno de los primeros momentos de confusión llegó al intentar mintear los primeros tokens. La función estaba bien, Metamask abría su ventana, la transacción se firmaba... pero algo no cuadraba: la cantidad no se reflejaba correctamente.

La razón era sencilla pero clave para entender cómo funciona un token ERC-20:

la cantidad que introducimos no representa un token directo, sino “unidades base”.

Si el token utiliza 18 decimales —como el nuestro—, entonces un “1” no equivale a 1 token, sino a 1 unidad mínima, una cifra insignificante para efectos prácticos.

Es decir, para crear 1 token completo debíamos introducir:

1×10^{18}

Y para crear 10 tokens:

10×10^{18}

Cuando esto no se hace, la transacción técnicamente puede ejecutarse, pero la cantidad recibida es tan pequeña que parece “que no se ha mintido nada”. Ese fue, literalmente, el primer aprendizaje real: los tokens no operan en números humanos, sino en números matemáticamente exactos.

Aprendizaje:

Antes de mintear, siempre hay que convertir la cantidad a unidades base según los decimales del token.



11.2. La falta de gas: cuando Metamask dice “insufficient funds”

Otro de los momentos habituales fue encontrarnos con el mensaje de Metamask indicando que no había suficiente ETH para pagar la transacción. Al principio sorprende, porque trabajamos en una red de pruebas y “no estamos pagando nada real”. Sin embargo, incluso en Sepolia es necesario tener una pequeña cantidad de ETH para cubrir el coste del gas.

Este gas no es un “pago” como tal, sino la forma de compensar a la red por ejecutar la operación. Deployar un contrato, mintear tokens, transferir... todo requiere gas, aunque sea muy poco en red de pruebas.

En nuestro caso, un par de transacciones consumieron más gas del esperado y nos quedamos momentáneamente sin saldo. El faucet nos solucionó la vida: volvimos a pedir ETH de prueba y todo siguió adelante.

Aprendizaje:

En blockchain, aunque sea testnet, siempre hay que vigilar el saldo de ETH antes de interactuar con el contrato.

11.3. El mint imposible: intentar crear más tokens de los permitidos

En uno de los intentos iniciales, probamos a mintear una cantidad muy alta “para probar” y la transacción falló con un mensaje de error: execution reverted.

La explicación estaba en la propia naturaleza del MMDV Wine Token: es un token **capado**. Tiene un límite máximo de tokens que pueden existir en toda su historia, definido en el cap.

Cuando se intenta superar ese límite, el contrato lo rechaza automáticamente. No hay forma de saltarse la restricción, ni modificando el código ni aprobando manualmente la transacción. El sistema está diseñado precisamente para prevenir emisiones indebidas.

Aprendizaje:

El cap es un mecanismo de seguridad que protege el proyecto. Si se intenta superarlo, el contrato simplemente no lo permite.



11.4. Cuando Metamask no mostraba el token (y parecía que el mint no había funcionado)

Otro momento clásico: después de ejecutar el mint, Etherscan confirmaba la transacción, los eventos del lote aparecían como registrados, pero Metamask seguía mostrando “0 MWT”.

El problema no estaba en el mint, sino en que Metamask todavía no conocía el token. Hasta que no añadimos manualmente el token personalizado, la wallet no puede mostrarlo.

Además, en algún momento los decimales confusos hicieron que Metamask no completara automáticamente la configuración y tuviéramos que introducirlos manualmente (18 decimales).

En cuanto añadimos:

- la dirección del contrato,
- el símbolo del token,
- y los decimales,

el saldo apareció inmediatamente.

Aprendizaje:

El éxito del mint se comprueba siempre en Etherscan. Metamask es una visualización, no una auditoría.

11.5. La sensación de “¿ha ido bien o no?” por el retraso de la red

Hubo momentos en los que Remix daba la operación como ejecutada, pero Etherscan tardaba unos segundos en mostrar la transacción. Ese pequeño intervalo genera la sensación de que “algo ha fallado”. Lo mismo ocurre cuando Metamask tarda en actualizar el saldo o en reflejar el token.

Esto no es un error: simplemente la red necesita unos segundos para procesar la transacción y extenderla por los nodos.

Aprendizaje:

En blockchain, lo que importa es lo que dice Etherscan. La interfaz puede tardar, pero la red siempre responde.



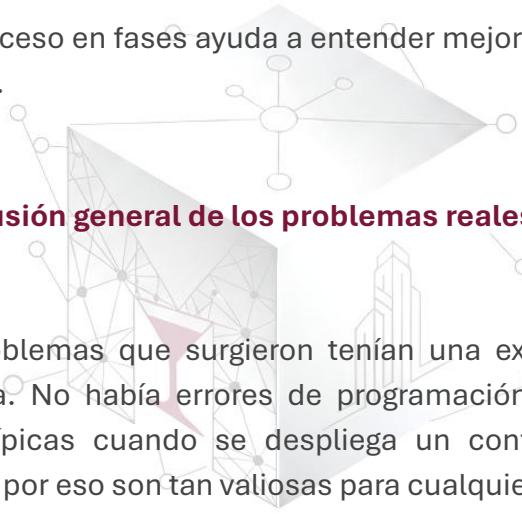
11.6. La función “redeem”, aún pendiente para el siguiente bloque

Aunque el contrato ya estaba desplegado y funcional, decidimos no ejecutar todavía la función redeem, que destruye tokens asociados a un lote. Guardamos esta operación para el Bloque 4, donde la veremos con más contexto, más calma y con una interfaz visual que ayudará a entenderla mejor.

Esto también forma parte del aprendizaje: no es necesario probar todas las funciones de un contrato en el mismo momento en que se despliega. Es mejor avanzar por etapas, validar cada paso y consolidar conocimientos.

Aprendizaje:

Separar el proceso en fases ayuda a entender mejor cada parte del ciclo de vida del token.



11.7. Conclusión general de los problemas reales

Todos los problemas que surgieron tenían una explicación lógica y una solución clara. No había errores de programación ni fallos graves: eran situaciones típicas cuando se despliega un contrato por primera vez. Precisamente por eso son tan valiosas para cualquier alumno.

Este bloque no solo enseña a desplegar un contrato; enseña a interpretar la blockchain, a leer mensajes de error, a revisar transacciones y, sobre todo, a confiar en el proceso.

Porque la blockchain siempre registra la verdad técnica de lo que ocurre.

12. Conclusiones del Bloque 3 y transición hacia el Bloque 4

El Bloque 3 marca un antes y un después en el proyecto. Hasta ahora habíamos trabajado con conceptos, ideas y diseño; pero aquí, por primera vez, hemos visto cómo todo eso se convierte en algo real. El MMDV Wine Token ya no es un ejercicio teórico ni un esquema conceptual: es un contrato vivo, verificable y trazable en la blockchain de Ethereum (red Sepolia).



Hemos recorrido un proceso completo de principio a fin. Primero preparamos el entorno, asegurándonos de que contábamos con las herramientas básicas para interactuar con la red. Después cargamos y compilamos el contrato en Remix, conectamos nuestra wallet a Sepolia y dimos el paso más simbólico del bloque: el despliegue. Desde ese momento, el contrato quedó inscrito en la blockchain de forma pública y permanente, accesible para cualquier persona en el mundo.

Una vez desplegado, aprendimos a verificarlo en Etherscan, a entender cada pestaña y a ver cómo la transparencia es parte esencial del ecosistema blockchain. Añadimos el token a Metamask para visualizarlo correctamente y ejecutamos el primer mint real del proyecto: el primer lote de vino tokenizado, registrado en un evento público y asociado a una dirección concreta. Con ese lote nació oficialmente el MMDV Wine Token como activo digital.

También fue importante detenernos en los obstáculos. Los problemas nos enseñaron más que cualquier explicación teórica. Aprendimos a interpretar errores de gas, a trabajar con decimales, a respetar el cap del contrato y a confiar en que Etherscan es siempre la fuente principal de verdad en blockchain. Son vivencias que no se pueden aprender solo con teoría.

Este bloque demuestra que la tokenización no es una idea abstracta: es un proceso concreto, técnico, con pasos claros y resultados visibles. Y también demuestra que cualquier persona, incluso sin experiencia previa en programación, puede entenderlo y ejecutarlo si sigue una guía adecuada.

Con esto, dejamos atrás la fase de creación del token y entramos en la siguiente etapa: **interactuar con él como herramienta funcional**.

Qué veremos en el Bloque 4



En el próximo bloque completaremos el ciclo natural del token:

- veremos la función redeem en acción,
- conectaremos el contrato con una pequeña interfaz visual (una mini-dApp),
- y mostraremos cómo cualquier usuario puede interactuar con el token sin necesidad de usar Remix o Etherscan.

Es decir, pasaremos de la capa técnica a la capa de experiencia de usuario. El objetivo será mostrar que un token, además de existir en la blockchain, puede integrarse en una aplicación sencilla y accesible, el paso previo a cualquier proyecto del mundo real.



Con eso cerraremos el capítulo técnico del MMDV Wine Token y dejaremos el proyecto listo para que cualquier alumno pueda reproducirlo, adaptarlo o utilizarlo como base para ideas propias.

“La tecnología no cobra vida en la teoría... sino en el momento en que la grabas en la cadena.”

