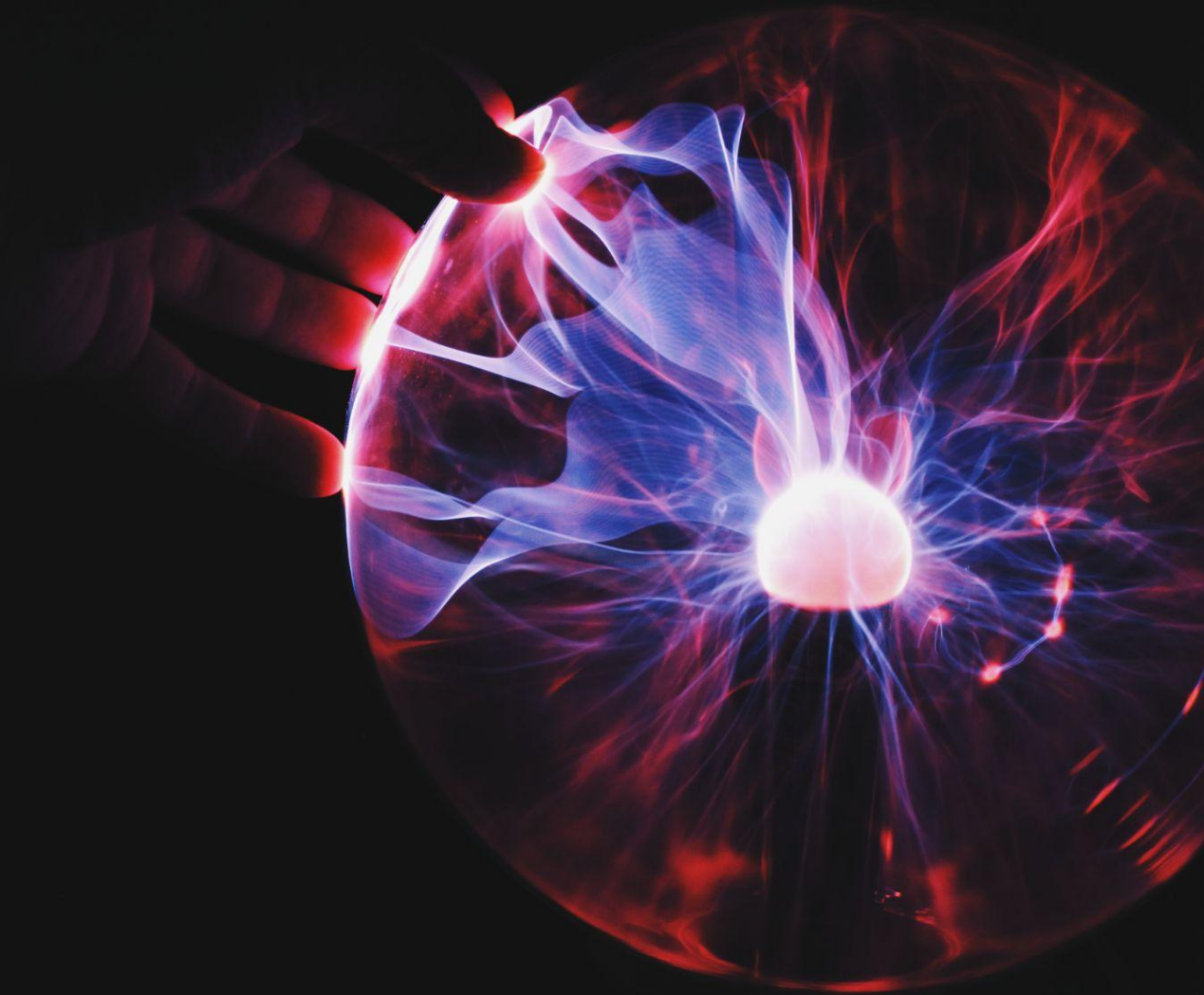


INPUT | OUTPUT

Minting Policy



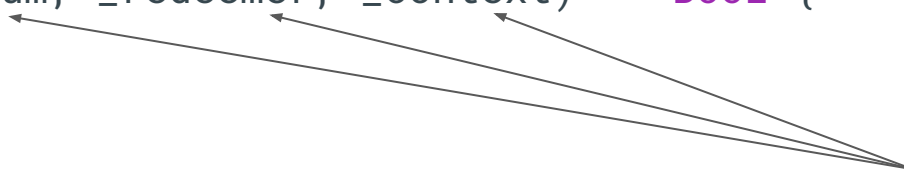
+ MINTING POLICIES _

Minting Policy: Con Aiken

A diferencia de los Spend Validators en Aiken, los Minting scripts no usan el Datum como parámetro:

```
validator {  
  fn gift(_datum, _redeemer, _context) -> Bool {  
    True  
  }  
}
```

El Spend validator
necesita **Datum**,
Redeemer y
ScriptContext



```
validator {  
  fn gift(_redeemer, _context) -> Bool {  
    True  
  }  
}
```

Minting Policy: Con Aiken

A diferencia de los Spend Validators en Aiken, los Minting scripts no usan el Datum como parámetro:

```
validator {  
  fn gift(_datum, _redeemer, _context) -> Bool {  
    True  
  }  
}
```

```
validator {  
  fn gift(_redeemer, _context) -> Bool {  
    True  
  }  
}
```

El Mint validator solo necesita **Redeemer** y **ScriptContext**



Minting Policy: Partes de un Token

Un token está compuesto por dos partes: **Policy Id + Token Name**

Por ejemplo:

33bf36b2bb5e3328ddc1a3a0303c196f3964fe6b99ced17634206d32 + 746F6B656E

Minting Policy: Partes de un Token

Un token está compuesto por dos partes: **Policy Id + Token Name**

Por ejemplo:

33bf36b2bb5e3328ddc1a3a0303c196f3964fe6b99ced17634206d32 + 746F6B656E



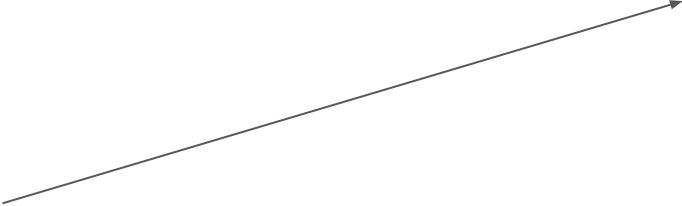
La Policy Id es el Hash del código en Plutus utilizado para crear el token.

Minting Policy: Partes de un Token

Un token está compuesto por dos partes: **Policy Id + Token Name**

Por ejemplo:

33bf36b2bb5e3328ddc1a3a0303c196f3964fe6b99ced17634206d32 + 746F6B656E



El Token Name es el nombre del token representado en formato hexadecimal

Minting Policy: Partes de un Token

Un token está compuesto por dos partes: **Policy Id + Token Name**

Por ejemplo:

33bf36b2bb5e3328ddc1a3a0303c196f3964fe6b99ced17634206d32 + 746F6B656E

Hash

String to Hex

+

Token

```
validator {  
  fn gift(_datum, _redeemer, _context) -> Bool {  
    True  
  }  
}
```


Minting Policy: Utilidades con MeshJS

Funciones útiles para trabajar con Minting policies:

`resolvePlutusScriptHash(Script)`: Función para obtener Policy Id dado un Script de plutus.

`mintAsset(Script, Mint, Redeemer)`: Método de la clase Transaction para minar tokens.

`mint(quantity, policy, name)`: Método de la clase MeshTxBuilder para minar tokens.

`mintPlutusScriptV2()`: Especifica que el Script de Plutus corresponde a la versión 2.

`mintRedeemerValue(redeemer)`: Especifica el valor de redeemer para ejecutar el código de Plutus

`mintingScript(Script)`: Especifica el código de Plutus que se va a ejecutar.

Minting Policy: Tipos de reglas

Al igual que los mint validators se pueden programar diferentes reglas para minar tokens y estos pueden ser mezcladas con reglas para gastar UTxOs de Spend validators. Entre algunos ejemplos aquí están los más comunes:

- Cuántos tokens se pueden crear
- Cuál nombre deben de tener los tokens
- Cuándo se pueden crear los tokens
- Quién puede crear los tokens
- ¿Se debe correr un Spend Validator para crear un token?

Minting Policy: Ejercicios

- **Token firmado:** Crear un script que permita crear un token solamente cuando una persona específica firme la transacción.
- **Token firmado y selección de token name:** Crear un script que permita crear un token solamente cuando una persona específica firme la transacción y además el nft tiene un nombre específico.

Notas:

- Los Scripts deben ser parametrizados
- Escribir código on-chain y off-chain

NFT: ¿Qué son?

Por sus siglas, un **NFT** (Non-Fungible Token) son tokens que no pueden ser reemplazados y son únicos e irrepetibles. Sólomente se puede crear uno de cada NFT. Esto se puede lograr por medio de diferentes técnicas utilizando código de Plutus (Aiken).



NFT: ¿Para qué sirven?

Los NFTs pueden tener varios usos por su naturaleza irrepetible. Por ejemplo puede significar la propiedad de algún objeto físico, los derechos de autor de alguna obra o la autoridad para poder realizar una acción dentro de la blockchain, entre otros usos.



NFT: ¿Cómo se logra crear un NFT?

Una de las técnicas más utilizadas en Cardano para la creación de NFTs es el uso de Validadores parametrizados:

- **UTxO como parámetro:** Se verifica que en la transacción en la que se crea el NFT, este UTxO de parámetro sea gastado, esto nos asegura de que el este no será gastado en otra ocasión.
- **Solo crear un token:** En la lógica del script se estipula que se quiere minar solamente un token. Asegurándonos de que este no se repita.
- **Token name:** También se puede decidir por el nombre del token. (Opcional)

Si se trata de crear un nuevo NFT con el mismo Script se debe utilizar un nuevo UTxO por lo tanto el Policy ID va a cambiar.

Fin

Preguntas?



INPUT | OUTPUT