INPUT | OUTPUT

Validadores parametrizados

Validadores parametrizados: ¿Que son?

- Son validadores de Plutus que reciben **información adicional** usada en la lógica del validador, lo que los hace más flexibles y reutilizables.

- Estas funciones **aceptan parámetros para crear validadores únicos** según sus datos de entrada.

- Al aplicar **diferentes parámetros**, se generan **validadores distintos**, cada uno con su propio hash y dirección específicos.

Validadores parametrizados: Ejemplo

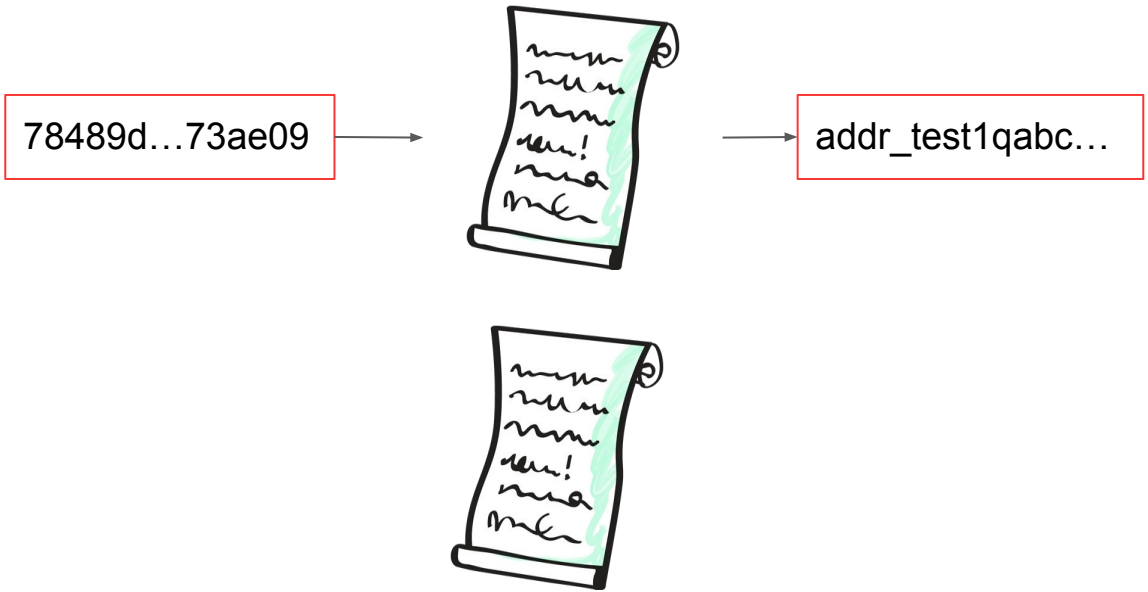
Digamos que tenemos un script que recibe una Public Key Hash como parámetro:

Validadores parametrizados: Ejemplo

Le proveemos un Public Key Hash como parámetro y obtenemos el script final a partir del cual se calcula la dirección del script:

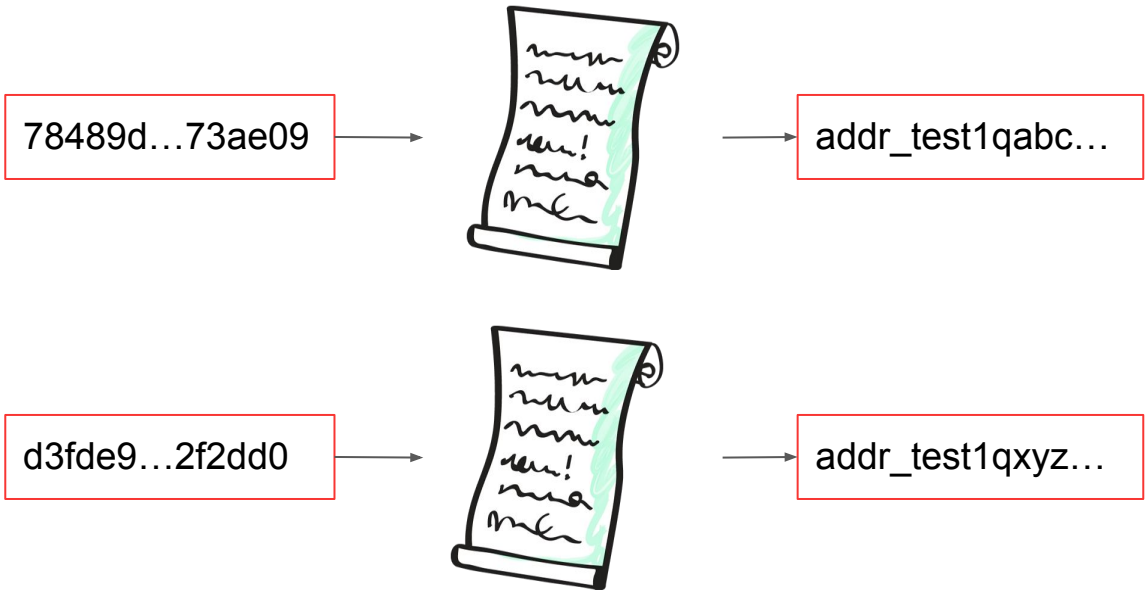
Validadores parametrizados: Ejemplo

Ahora, tenemos el mismo script previo a la aplicación del PKH:



Validadores parametrizados: Ejemplo

Al proveerle un parámetro (PKH) diferente, la dirección cambia completamente:



Validadores parametrizados: Ejemplo sin parametrizar

```
validator {
  fn signed(beneficiary: Hash<Blake2b_224, VerificationKey>, _r: Void, ctx: ScriptContext) -> Bool {
    let ScriptContext {
      transaction: Transaction { extra_signatories, .. },
      purpose,
    } = ctx
    when purpose is {
      Spend(_) -> {
          list.has(extra_signatories, beneficiary)
      }
      _ -> False
    }
}
```

Validadores parametrizados: Ejemplo parametrizado

```
validator (beneficiary: Hash<Blake2b_224, VerificationKey>){
  fn signed(_d: Void, _r: Void, ctx: ScriptContext) -> Bool {
    let ScriptContext {
      transaction: Transaction { extra_signatories, .. },
      purpose,
    } = ctx
    when purpose is {
      Spend(_) -> {
          list.has(extra_signatories, beneficiary)
      }
      _ -> False
    }
}
```

Validadores parametrizados: Código MeshJS

Para poder aplicar parámetros tenemos la función `applyParamsToScript` de MeshJS:

```
applyParamsToScript(codigoDeScript, [param1,param2, ...])
```

*Al usar applyParamsToScript , hay que proveer los parámetros en el mismo orden y no es necesario hacer applyCborEncoding

Validadores parametrizados: ¿Para qué?

- **Si se pone en el datum:** Puede cambiar en una transacción que actualice el datum

- **Si se hardcodea:** Hay que cambiar el validador. Lo que significa que hay que recompilar, re-testear, y re-auditar el código.


Fin
Preguntas?
INPUT | OUTPUT