進階概念

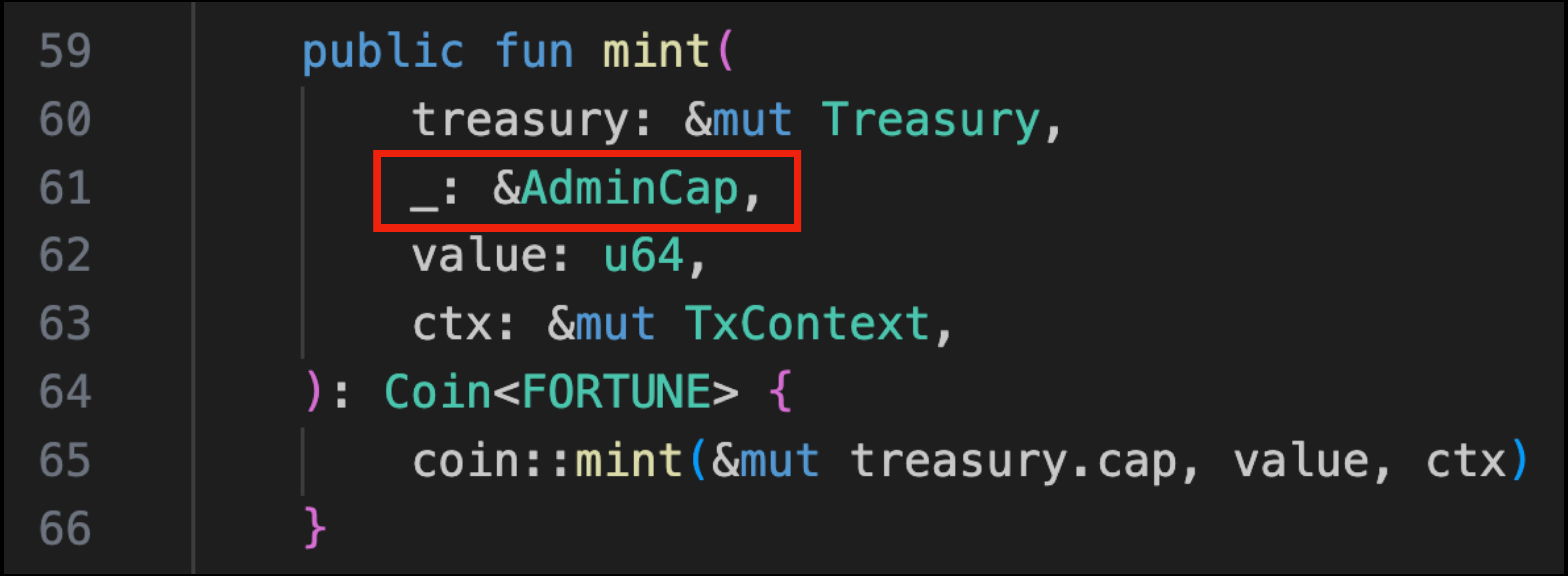
Justa from Bucket Protocol

Bucket

大綱

- 權限控制 (capability)
- OTW (One Time Witness)
- 物件表現 (displayer)
- Hot Potato

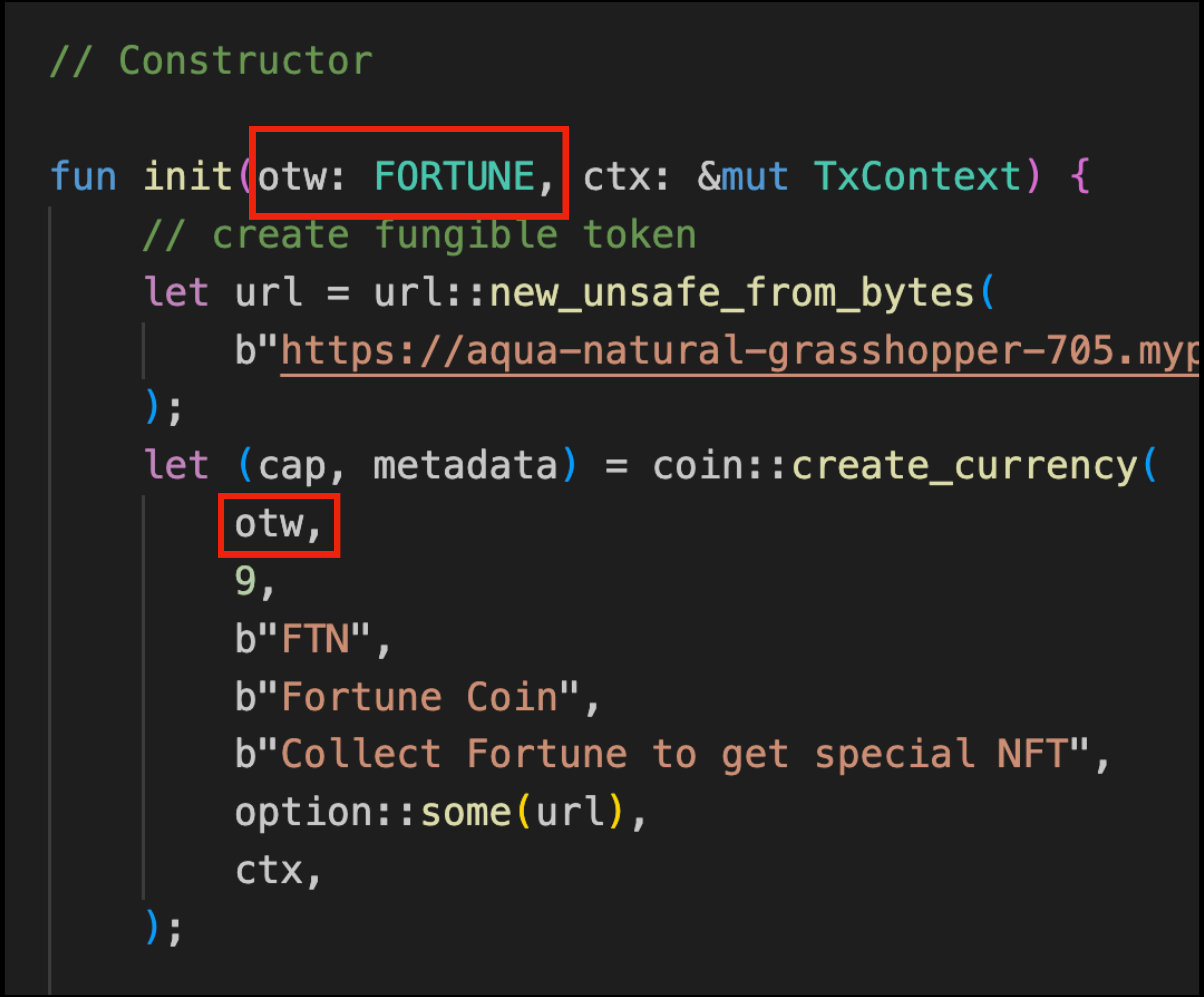
權限控制 (capability)



```
public fun mint(
    treasury: &mut Treasury,
    _: &AdminCap,
    value: u64,
    ctx: &mut TxContext,
): Coin<FORTUNE> {
    coin::mint(&mut treasury.cap, value, ctx)
}
```

OTW (One Time Witness)

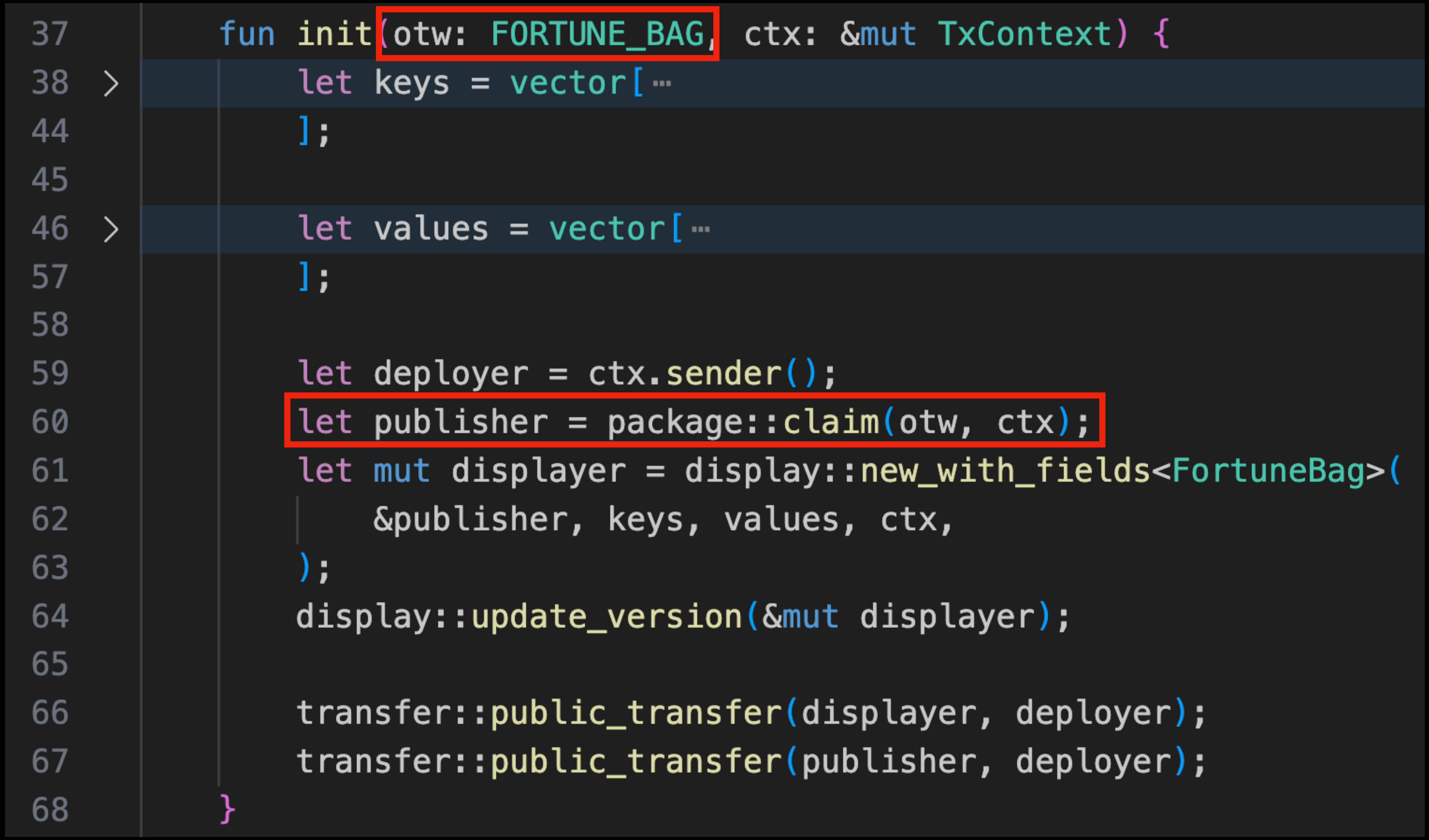
sui::coin::create_currency



```
// Constructor

fun init(otw: FORTUNE, ctx: &mut TxContext) {
    // create fungible token
    let url = url::new_unsafe_from_bytes(
        b"https://aqua-natural-grasshopper-705.myp
    );
    let (cap, metadata) = coin::create_currency(
        otw,
        9,
        b"FTN",
        b"Fortune Coin",
        b"Collect Fortune to get special NFT",
        option::some(url),
        ctx,
    );
```

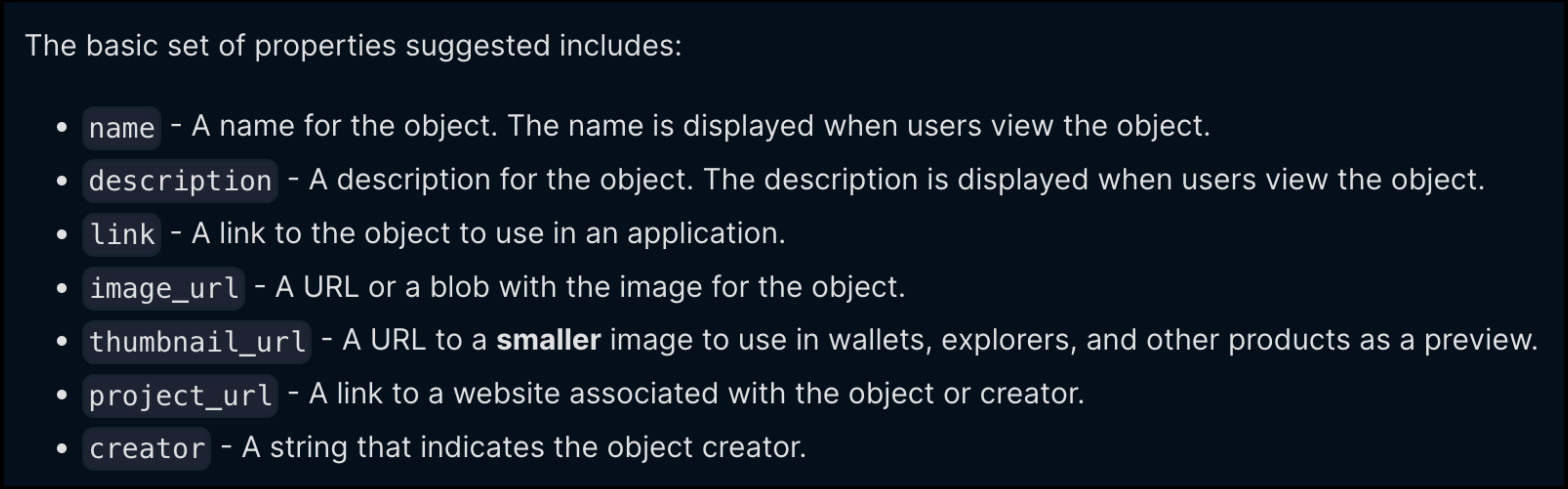
OTW (One Time Witness)

sui::package::claim



```
fun init(otw: FORTUNE_BAG, ctx: &mut TxContext) {
    let keys = vector[⋯
    ];

    let values = vector[⋯
    ];

    let deployer = ctx.sender();
    let publisher = package::claim(otw, ctx);
    let mut displayer = display::new_with_fields<FortuneBag>(
        &publisher, keys, values, ctx,
    );
    display::update_version(&mut displayer);

    transfer::public_transfer(displayer, deployer);
    transfer::public_transfer(publisher, deployer);
}
```

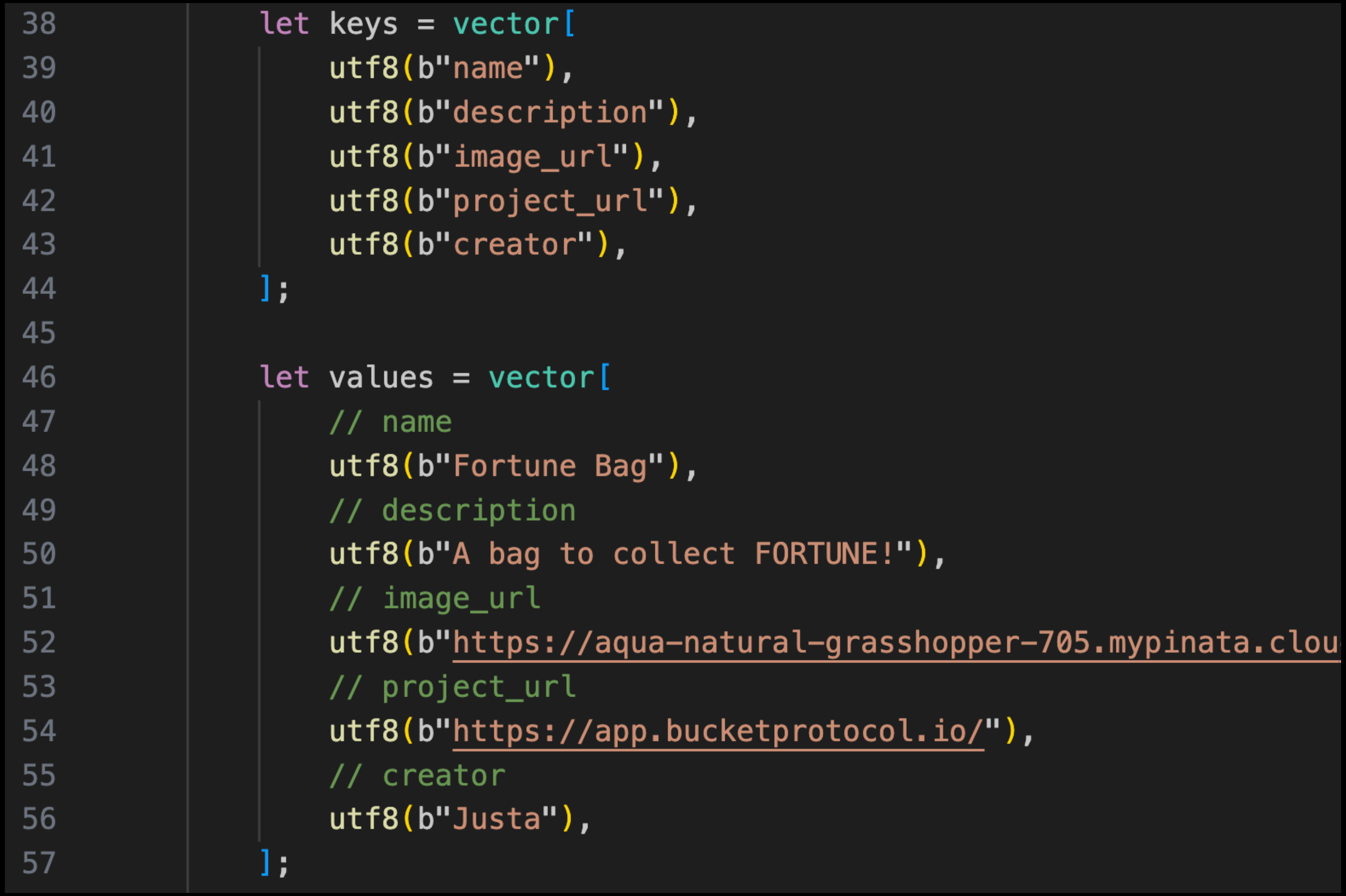
物件表現 (displayer)



The basic set of properties suggested includes:

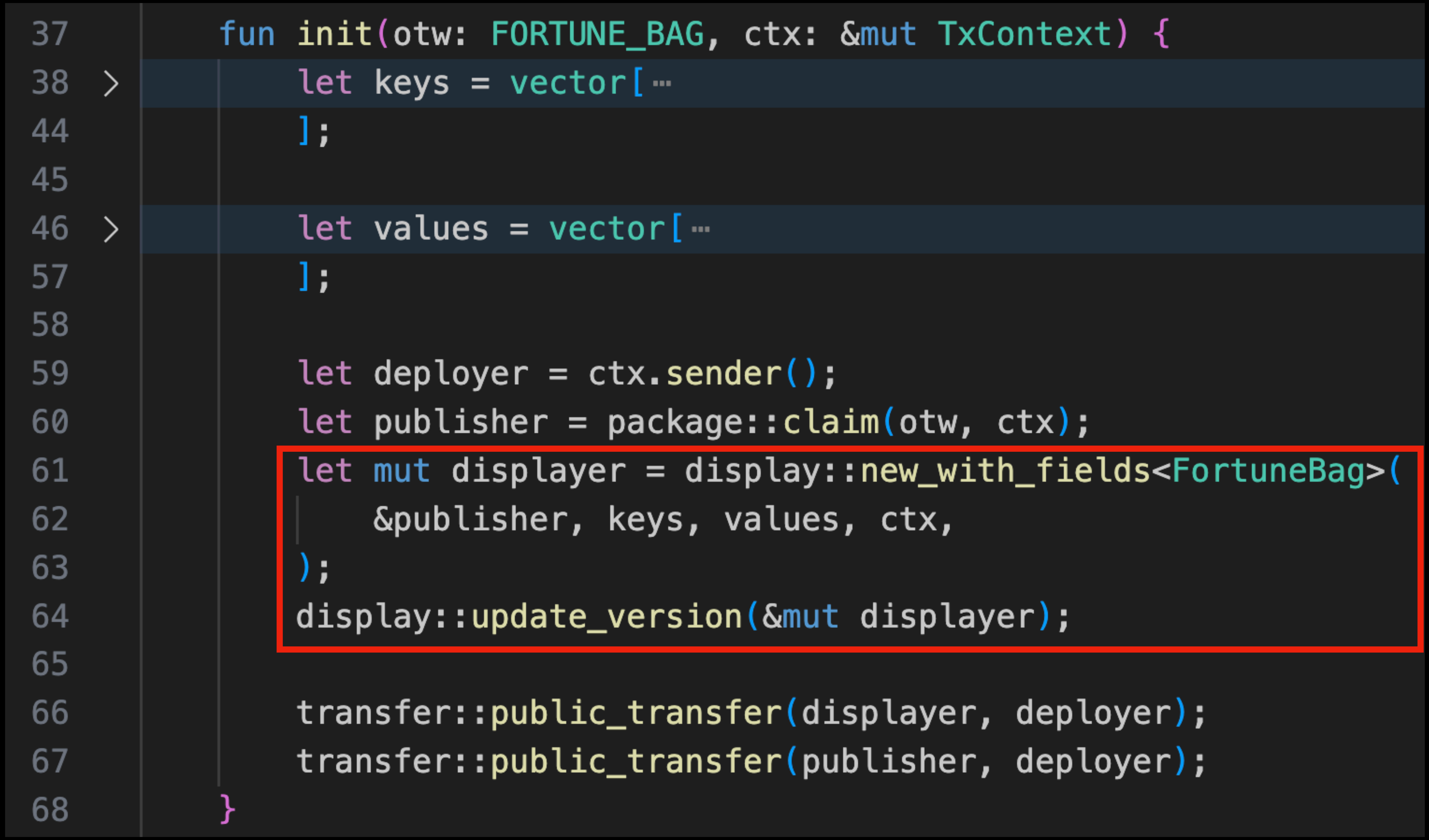
- `name` - A name for the object. The name is displayed when users view the object.
- `description` - A description for the object. The description is displayed when users view the object.
- `link` - A link to the object to use in an application.
- `image_url` - A URL or a blob with the image for the object.
- `thumbnail_url` - A URL to a **smaller** image to use in wallets, explorers, and other products as a preview.
- `project_url` - A link to a website associated with the object or creator.
- `creator` - A string that indicates the object creator.

物件表現 (displayer)



```
    let keys = vector[
        utf8(b"name"),
        utf8(b"description"),
        utf8(b"image_url"),
        utf8(b"project_url"),
        utf8(b"creator"),
    ];

    let values = vector[
        // name
        utf8(b"Fortune Bag"),
        // description
        utf8(b"A bag to collect FORTUNE!"),
        // image_url
        utf8(b"https://aqua-natural-grasshopper-705.mypinata.clou
        // project_url
        utf8(b"https://app.bucketprotocol.io/"),
        // creator
        utf8(b"Justa"),
    ];
```

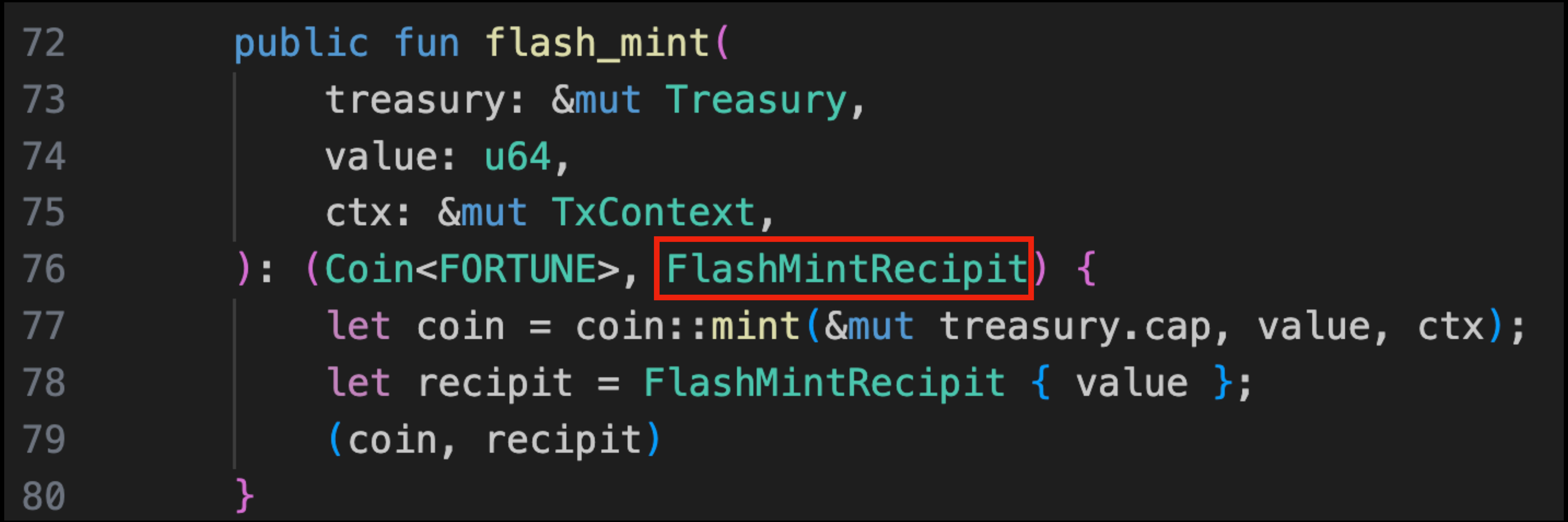
物件表現 (displayer)



```
fun init(otw: FORTUNE_BAG, ctx: &mut TxContext) {
    let keys = vector[⋯
    ];

    let values = vector[⋯
    ];

    let deployer = ctx.sender();
    let publisher = package::claim(otw, ctx);
    let mut displayer = display::new_with_fields<FortuneBag>(
        &publisher, keys, values, ctx,
    );
    display::update_version(&mut displayer);

    transfer::public_transfer(displayer, deployer);
    transfer::public_transfer(publisher, deployer);
}
```

Hot Potato

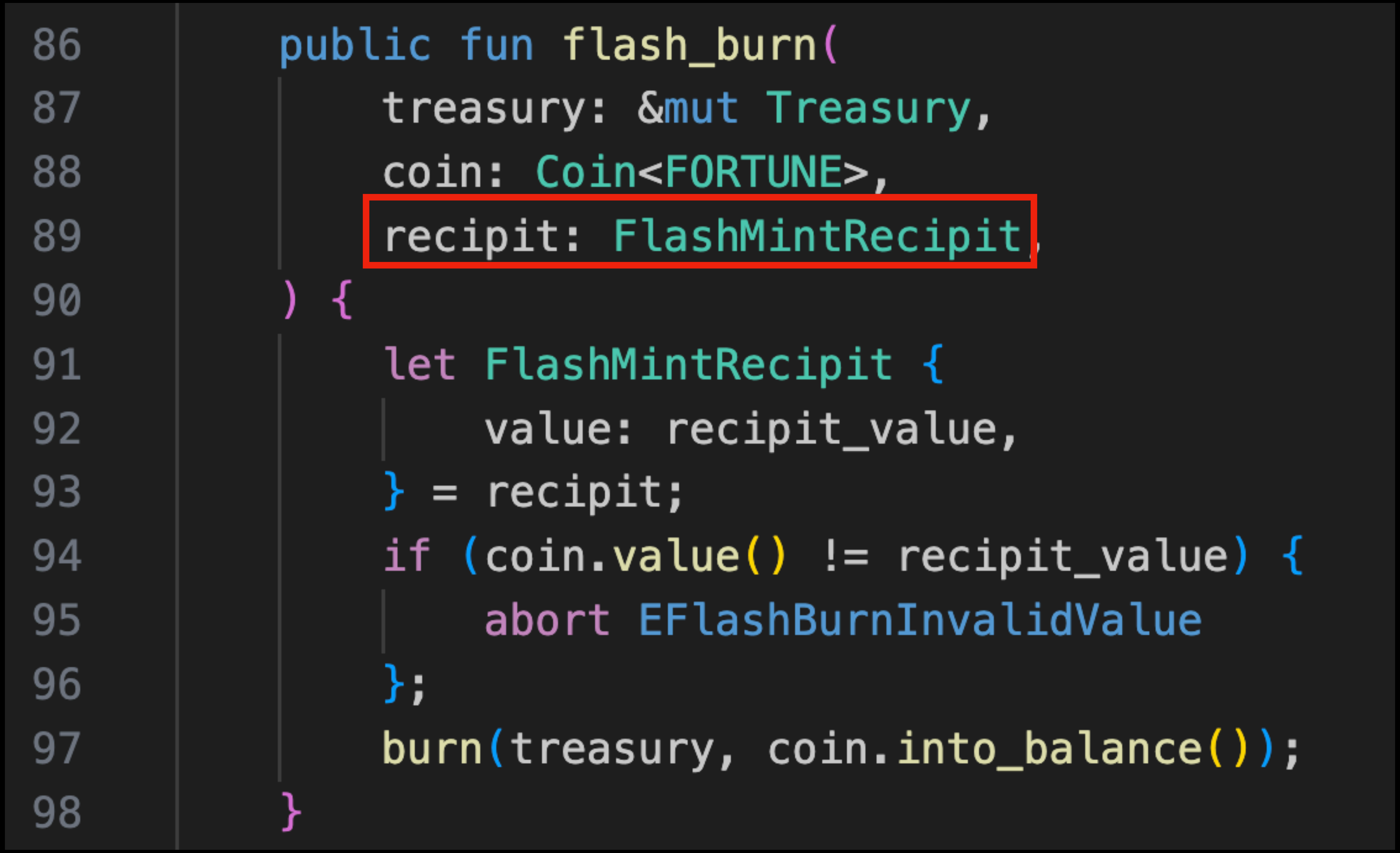
create



```
public fun flash_mint(
    treasury: &mut Treasury,
    value: u64,
    ctx: &mut TxContext,
): (Coin<FORTUNE>, FlashMintRecipit) {
    let coin = coin::mint(&mut treasury.cap, value, ctx);
    let recipit = FlashMintRecipit { value };
    (coin, recipit)
}
```

Hot Potato

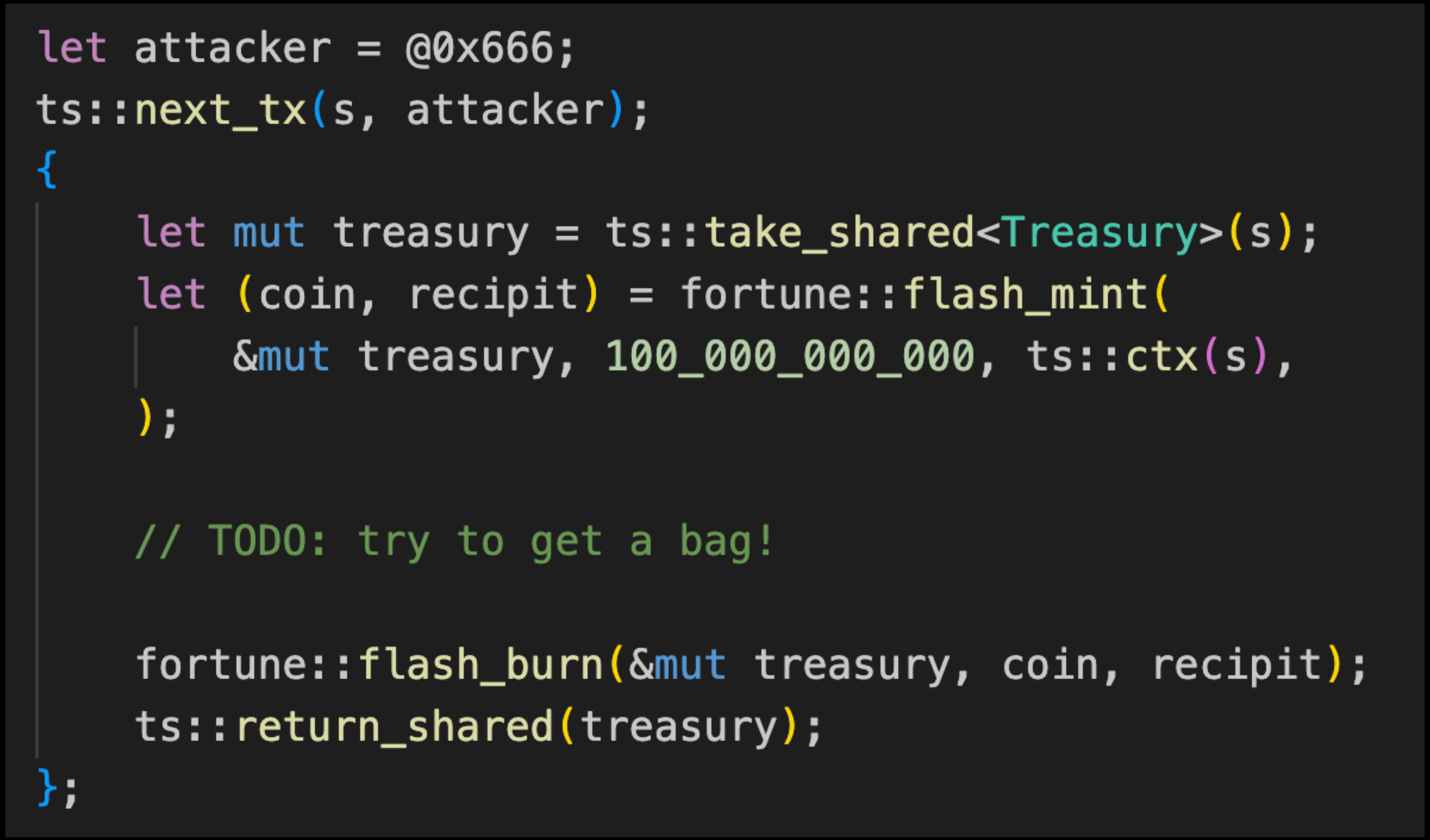
consume



```
    public fun flash_burn(
        treasury: &mut Treasury,
        coin: Coin<FORTUNE>,
        recipit: FlashMintRecipit,
    ) {
        let FlashMintRecipit {
            value: recipit_value,
        } = recipit;
        if (coin.value() != recipit_value) {
            abort EFlashBurnInvalidValue
        };
        burn(treasury, coin.into_balance());
    }
```

CTF

Get a bag without the need of Fortune coin



```
let attacker = @0x666;
ts::next_tx(s, attacker);
{
    let mut treasury = ts::take_shared<Treasury>(s);
    let (coin, recipit) = fortune::flash_mint(
        &mut treasury, 100_000_000_000, ts::ctx(s),
    );

    // TODO: try to get a bag!

    fortune::flash_burn(&mut treasury, coin, recipit);
    ts::return_shared(treasury);
};
```