Loyalty Tokens

Using the Sui Closed-Loop Token standard, you can create tokens that are valid only for a specific service, like an airline that wants to grant tokens to frequent flyers to purchase tickets or upgrades.

The following example demonstrates the creation of a loyalty token that bearers can use to make purchases in a digital gift shop.

The Loyalty Token example illustrates a loyalty token that is created with the Closed Loop Token standard. If you were to implement this example, the Admin would send LOYALTY tokens to the users of your service as a reward for their loyalty. The example creates a GiftShop where holders can spend LOYALTY tokens to buy Gift s.

The loyalty.move source file contains the examples::loyalty module code that creates the loyalty token. The module includes the one-time witness (OTW) that creates the coin (with the same name as the module, LOYALTY), possesses only the drop ability, and has no fields. These are the characteristics of a OTW, which ensures the LOYALTY type has a single instance.

The init function of the module uses the LOYALTY OTW to create the token. All init functions run one time only at the package publish event. The initializer function makes use of the OTW LOYALTY type defined previously in its call to create_currency . The function also defines a policy, sending both the policy capability and treasury capability to the address associated with the publish event. The holder of these transferrable capabilities can mint new LOYALTY tokens and modify their policies.

The LOYALTY minting function is called reward_user . As mentioned previously, the holder of the TreasuryCap can call this function to mint new loyalty tokens and send them to the desired address. The function uses the token::mint function to create the token and token::transfer to send it to the intended recipient.

Finally, the example includes a buy_a_gift function to handle the redemption of LOYALTY tokens for Gift types. The function ensures the gift price matches the number of loyalty tokens spent, then uses the token::spend function to handle the treasury bookkeeping.

Toggle display of the complete source for this example, including comments, or use the link in the Related links section to view the project source on GitHub.

loyalty.move

Example

The Loyalty Token example illustrates a loyalty token that is created with the Closed Loop Token standard. If you were to implement this example, the Admin would send LOYALTY tokens to the users of your service as a reward for their loyalty. The example creates a GiftShop where holders can spend LOYALTY tokens to buy Gift s.

The loyalty.move source file contains the examples::loyalty module code that creates the loyalty token. The module includes the one-time witness (OTW) that creates the coin (with the same name as the module, LOYALTY), possesses only the drop ability, and has no fields. These are the characteristics of a OTW, which ensures the LOYALTY type has a single instance.

The init function of the module uses the LOYALTY OTW to create the token. All init functions run one time only at the package publish event. The initializer function makes use of the OTW LOYALTY type defined previously in its call to create_currency . The function also defines a policy, sending both the policy capability and treasury capability to the address associated with the publish event. The holder of these transferrable capabilities can mint new LOYALTY tokens and modify their policies.

The LOYALTY minting function is called reward_user . As mentioned previously, the holder of the TreasuryCap can call this function to mint new loyalty tokens and send them to the desired address. The function uses the token::mint function to create the token and token::transfer to send it to the intended recipient.

Finally, the example includes a buy_a_gift function to handle the redemption of LOYALTY tokens for Gift types. The function ensures the gift price matches the number of loyalty tokens spent, then uses the token::spend function to handle the treasury bookkeeping.

Toggle display of the complete source for this example, including comments, or use the link in the Related links section to view the project source on GitHub.

loyalty.move

Related links