Substrate

River

Kitties-002

Creating an Event

https://substrate.dev/substrate-collectables-workshop/#/2/creating-an-event

Why Event

在函数结束时发出一个事件，不仅要报告成功，
还要告诉“off-chain world”某些特定的状态转换已经发生。

1- Declaring an Event

```
decl_event!(
    pub enum Event<T>
    where
        <T as system::Trait>::AccountId,
        <T as system::Trait>::Balance
    {
        MyEvent(u32, Balance),
        MyOtherEvent(Balance, AccountId),
    }
);
```

2- Adding an Event Type

```
pub trait Trait: balances::Trait {
    type Event: From<Event<Self>> + Into<<Self as system::Trait>::Event>;
}
```

3- Depositing an Event

```
fn deposit_event<T>() = default;
```

4- Calling deposit_event()

Module

```
let my_value = 1337;
let my_balance = <T::Balance as As<u64>>::sa(1337);
Self::deposit_event(RawEvent::MyEvent(my_value, my_balance));
```

5- Updating lib.rs to Include Events

```
// `lib.rs`
...
impl mymodule::Trait for Runtime {
    type Event = Event;
}
...
```

construct_runtime!

```
MyModule: mymodule::{Module, Call, Storage, Event<T>},
```

You turn

1- // ACTION: Don't forget to update `lib.rs` with the `Event` type

2- // ACTION: Add `support::decl_event` to use the `decl_event!` macro

3- // ACTION: Define your `Event` type here
   //   HINT: It needs these traits: `From<Event<Self>> + Into<<Self as system::Trait>::Event>`

4- // ACTION: Add a `Created` event which includes an `AccountId` and a `Hash`

5- // ACTION: Define your generic `deposit_event<T>()` function

6- // ACTION: Deposit your event here

Event

makeExtrinsicCall: updated status :: {"events":[{"phase":{"ApplyExtrinsic":1},"event":{"index":"0x0500","data":[**"5GrwvaEF5zXb26Fz9rcQpDWS57CtERHpNehXCPcNoHGKutQY","0x4efec02013032b87374f3ce1d4426cccfcc85a1c99c46787899e93b51880b129"**]}},{"phase":{"ApplyExtrinsic":1},"event":{"index":"0x0000","data":[]}}],"status":{"Finalized":"0x7ecc7ad3e6be9876c7207468a75f5ef6c0d6eff0fb4a68c2bf18dc6960a0bf69"}}

makeExtrinsicCall: updated status :: {"events":[{"phase":{"ApplyExtrinsic":1},"event":{"index":"0x0500","data":["**5GrwvaEF5zXb26Fz9rcQpDWS57CtERHpNehXCPcNoHGKutQY**","**0x63e3069af36a3177bfc29f777356f1dbee1f324c5e7a3d4c73a0d5b98bc88f32**"]}},{"phase":{"ApplyExtrinsic":1},"event":{"index":"0x0000","data":[]}}],"status":{"Finalized":"0xbde501a405ea3b08ca6d6691e413d5a41a816d35a60b66c36cfff027fa8c19f1"}}

探索？？？

```
// All code is wrapped within an async closure,
// allowing access to api, hashing, keyring, types, util.
// (async ({ api, hashing, keyring, types, util }) => {
//   ... any user code is executed here ...
// })();

// Make a transfer from Alice to Bob and listen to system events.
// You need to be connected to a development chain for this example to work.
const ALICE = '5GrwvaEF5zXb26Fz9rcQpDWS57CtERHpNehXCPcNoHGKutQY';

// Create a extrinsic, transferring randomAmount units to Bob.
const transfer = api.tx.substratekitties.create();

// Sign and Send the transaction
transfer.signAndSend(ALICE, ({ events = [], status }) => {
  if (status.isFinalized) {
    console.log('Successful transfer of ' + randomAmount + ' with hash ' +
status.asFinalized.toHex());
  } else {
    console.log('Status of transfer: ' + status.type);
  }

  events.forEach(({ phase, event: { index, data } }) => {
    console.log(phase.toString() + ' : ' + data.toString());
  });
});
```

Substrate

River

Thanks