The Move Book

Events are a way to notify off-chain listeners about on-chain events. They are used to emit additional information about the transaction that is not stored - and, hence, can't be accessed - on-chain. Events are emitted by the sui::event module located in the Sui Framework .

Any custom type with the copy and drop abilities can be emitted as an event. Sui Verifier requires the type to be internal to the module.

Events are emitted using the emit function in the sui::event module. The function takes a single argument - the event to be emitted. The event data is passed by value,

The Sui Verifier requires the type passed to the emit function to be internal to the module . So emitting a type from another module will result in a compilation error. Primitive types, although they match the copy and drop requirement, are not allowed to be emitted as events.

Events are a part of the transaction result and are stored in the transaction effects . As such, they natively have the sender field which is the address who sent the transaction. So adding a "sender" field to the event is not necessary. Similarly, event metadata contains the timestamp. But it is important to note that the timestamp is relative to the node and may vary a little from node to node.

Emitting Events

Events are emitted using the emit function in the sui::event module. The function takes a single argument - the event to be emitted. The event data is passed by value,

```bash module book::events;

use sui::coin::Coin; use sui::sui::SUI; use sui::event;

/// The item that can be purchased. public struct Item has key { id: UID }

/// Event emitted when an item is purchased. Contains the ID of the item and /// the price for which it was purchased. public struct ItemPurchased has copy, drop { item: ID, price: u64 }

/// A marketplace function which performs the purchase of an item. public fun purchase(coin: Coin, ctx: &mut TxContext) { let item = Item { id: object::new(ctx) };

// Create an instance of `ItemPurchased` and pass it to `event::emit`.
event::emit(ItemPurchased {
    item: object::id(&item),
    price: coin.value()
});

// Omitting the rest of the implementation to keep the example simple.
abort 0

} ```
```

The Sui Verifier requires the type passed to the emit function to be internal to the module . So emitting a type from another module will result in a compilation error. Primitive types, although they match the copy and drop requirement, are not allowed to be emitted as events.

Events are a part of the transaction result and are stored in the transaction effects . As such, they natively have the sender field which is the address who sent the transaction. So adding a "sender" field to the event is not necessary. Similarly, event metadata contains the timestamp. But it is important to note that the timestamp is relative to the node and may vary a little from node to node.

Event Structure

Events are a part of the transaction result and are stored in the transaction effects . As such, they natively have the sender field which is the address who sent the transaction. So adding a "sender" field to the event is not necessary. Similarly, event metadata contains the timestamp. But it is important to note that the timestamp is relative to the node and may vary a little from node to node.