IMoveObject

This interface is implemented by types that represent a Move object on-chain (A Move value whose type has `key`).

```
interface IMoveObject {
  contents: MoveValue
  hasPublicTransfer: Boolean!
  display: [DisplayEntry!]
  dynamicField(
    name: DynamicFieldName!
  ): DynamicField
  dynamicObjectField(
    name: DynamicFieldName!
  ): DynamicField
  dynamicFields(
    first: Int
    after: String
    last: Int
    before: String
  ): DynamicFieldConnection!
}
```

Fields

`IMoveObject.contents` ● `MoveValue` **object**

Displays the contents of the Move object in a JSON string and through GraphQL types. Also provides the flat representation of the type signature, and the BCS of the corresponding data.

`IMoveObject.hasPublicTransfer` ● `Boolean!` **non-null** **scalar**

Determines whether a transaction can transfer this object, using the TransferObjects transaction command or `sui::transfer::public_transfer`, both of which require the object to have the `key` and `store` abilities.

`IMoveObject.display` ● `[DisplayEntry!]` **list** **object**

The set of named templates defined on-chain for the type of this object, to be handled off-chain. The server substitutes data from the object into these templates to generate a display string per template.

`IMoveObject.dynamicField`.`DynamicField` **object**

Access a dynamic field on an object using its name. Names are arbitrary Move values whose type have `copy`, `drop`, and `store`, and are specified using their type, and their BCS contents, Base64 encoded.

Dynamic fields on wrapped objects can be accessed by using the same API under the Ownertype.

`IMoveObject.dynamicField.name`.`DynamicFieldName!` **non-null** **input**

`IMoveObject.dynamicObjectField`.`DynamicField` **object**

Access a dynamic object field on an object using its name. Names are arbitrary Move values whose type have `copy`, `drop`, and `store`, and are specified using their type, and their BCS contents, Base64 encoded. The value of a dynamic object field can also be accessed off-chain directly via its address (e.g. using `Query.object`).

Dynamic fields on wrapped objects can be accessed by using the same API under the Owner type.

`IMoveObject.dynamicObjectField.name`.`DynamicFieldName!` **non-null** **input**

`IMoveObject.dynamicFields`.`DynamicFieldConnection!` **non-null** **object**

The dynamic fields and dynamic object fields on an object.

Dynamic fields on wrapped objects can be accessed by using the same API under the Owner type.

`IMoveObject.dynamicFields.first`.`Int` **scalar**

`IMoveObject.dynamicFields.after`.`String` **scalar**

`IMoveObject.dynamicFields.last`.`Int` **scalar**

`IMoveObject.dynamicFields.before`.`String` **scalar**

Implemented By

`Coin` **object**.`CoinMetadata` **object**.`MoveObject` **object**.`StakedSui` **object**.`SuinsRegistration` **object**

Edit this page