ActiveJwk

No description

```
type ActiveJwk {
  iss: String!
  kid: String!
  kty: String!
  e: String!
  n: String!
  alg: String!
  epoch: Epoch
}
```

Fields

ActiveJwk.iss.String! non-null scalar

The string (Issuing Authority) that identifies the OIDC provider.

ActiveJwk.kid.String! non-null scalar

The string (Key ID) that identifies the JWK among a set of JWKs, (RFC 7517, Section 4.5).

ActiveJwk.kty.String! non-null scalar

The JWK key type parameter, (RFC 7517, Section 4.1).

ActiveJwk.e.String! non-null scalar

The JWK RSA public exponent, (RFC 7517, Section 9.3).

ActiveJwk.n.String! non-null scalar

The JWK RSA modulus, (RFC 7517, Section 9.3).

ActiveJwk.alg.String! non-null scalar

The JWK algorithm parameter, (RFC 7517, Section 4.4).

ActiveJwk.epoch.Epoch object

The most recent epoch in which the JWK was validated.

Member Of

`ActiveJwkConnection` **object** . `ActiveJwkEdge` **object**

Edit this page

© 2025 SUI FOUNDATION | DOCUMENTATION DISTRIBUTED UNDER CC BY 4.0

ActiveJwkConnection

No description

```
type ActiveJwkConnection {
  pageInfo: PageInfo!
  edges: [ActiveJwkEdge!]!
  nodes: [ActiveJwk!]!
}
```

Fields

`ActiveJwkConnection.pageInfo`.`PageInfo!` **non-null** **object**

Information to aid in pagination.

`ActiveJwkConnection.edges`.`[ActiveJwkEdge!]!` **non-null** **object**

A list of edges.

`ActiveJwkConnection.nodes`.`[ActiveJwk!]!` **non-null** **object**

A list of nodes.

Member Of

`AuthenticatorStateUpdateTransaction` **object**

Edit this page

© 2025 SUI FOUNDATION | DOCUMENTATION DISTRIBUTED UNDER CC BY 4.0

ActiveJwkEdge

An edge in a connection.

```
type ActiveJwkEdge {
  node: ActiveJwk!
  cursor: String!
}
```

Fields

`ActiveJwkEdge.node`.`ActiveJwk!` **non-null** **object**

The item at the end of the edge

`ActiveJwkEdge.cursor`.`String!` **non-null** **scalar**

A cursor for use in pagination

Member Of

`ActiveJwkConnection` **object**

Edit this page

© 2025 SUI FOUNDATION | DOCUMENTATION DISTRIBUTED UNDER CC BY 4.0

Address

The 32-byte address that is an account address (corresponding to a public key).

```
type Address implements IOwner {
  address: SuiAddress!
  objects(
    first: Int
    after: String
    last: Int
    before: String
    filter: ObjectFilter
  ): MoveObjectConnection!
  balance(
    type: String
  ): Balance
  balances(
    first: Int
    after: String
    last: Int
    before: String
  ): BalanceConnection!
  coins(
    first: Int
    after: String
    last: Int
    before: String
    type: String
  ): CoinConnection!
  stakedSuis(
    first: Int
    after: String
    last: Int
    before: String
  ): StakedSuiConnection!
  defaultSuinsName(
    format: DomainFormat
  ): String
  suinsRegistrations(
    first: Int
    after: String
    last: Int
    before: String
  ): SuinsRegistrationConnection!
  transactionBlocks(
    first: Int
    after: String
    last: Int
```

```
    before: String
    relation: AddressTransactionBlockRelationship
    filter: TransactionBlockFilter
    scanLimit: Int
  ): TransactionBlockConnection!
}
```

Fields

`Address.address`.`SuiAddress!` non-null scalar

`Address.objects`.`MoveObjectConnection!` non-null object

Objects owned by this address, optionally `filter`-ed.

`Address.objects.first`.`Int` scalar

`Address.objects.after`.`String` scalar

`Address.objects.last`.`Int` scalar

`Address.objects.before`.`String` scalar

`Address.objects.filter`.`ObjectFilter` input

`Address.balance`.`Balance` object

Total balance of all coins with marker type owned by this address. If type is not supplied, it defaults to `0x2::sui::SUI`.

`Address.balance.type`.`String` scalar

`Address.balances`.`BalanceConnection!` non-null object

The balances of all coin types owned by this address.

`Address.balances.first`.`Int` scalar

`Address.balances.after`.`String` scalar

`Address.balances.last`.`Int` scalar

`Address.balances.before`.`String` scalar

`Address.coins`.`CoinConnection!` non-null object

The coin objects for this address.

`type` is a filter on the coin's type parameter, defaulting to `0x2::sui::SUI`.

`Address.coins.first`.`Int` scalar

`Address.coins.after`.`String` scalar

`Address.coins.last`.`Int` scalar

`Address.coins.before`.`String` scalar

`Address.coins.type`.`String` scalar

`Address.stakedSuis`.`StakedSuiConnection!` non-null object

The `0x3::staking_pool::StakedSui` objects owned by this address.

`Address.stakedSuis.first`.`Int` scalar

`Address.stakedSuis.after`.`String` scalar

`Address.stakedSuis.last`.`Int` scalar

`Address.stakedSuis.before`.`String` scalar

`Address.defaultSuinsName`.`String` scalar

The domain explicitly configured as the default domain pointing to this address.

`Address.defaultSuinsName.format`.`DomainFormat` enum

`Address.suinsRegistrations`.`SuinsRegistrationConnection!` non-null object

The SuinsRegistration NFTs owned by this address. These grant the owner the capability to manage the associated domain.

`Address.suinsRegistrations.first`.`Int` scalar

`Address.suinsRegistrations.after`.`String` scalar

`Address.suinsRegistrations.last`.`Int` scalar

`Address.suinsRegistrations.before`.`String` scalar

`Address.transactionBlocks`.`TransactionBlockConnection!` non-null object

Similar behavior to the `transactionBlocks` in Query but supporting the additional `AddressTransactionBlockRelationship` filter, which defaults to `SENT`.

`scanLimit` restricts the number of candidate transactions scanned when gathering a page of results. It is required for queries that apply more than two complex filters (on function, kind, sender, recipient, input object, changed object, or ids), and can be at most `serviceConfig.maxScanLimit`.

When the scan limit is reached the page will be returned even if it has fewer than `first` results when paginating forward (`last` when paginating backwards). If there are more transactions to scan, `pageInfo.hasNextPage` (or `pageInfo.hasPreviousPage`) will be set to `true`, and `PageInfo.endCursor` (or `PageInfo.startCursor`) will be set to the last transaction that was scanned as opposed to the last (or first) transaction in the page.

Requesting the next (or previous) page after this cursor will resume the search, scanning the next `scanLimit` many transactions in the direction of pagination, and so on until all transactions in the scanning range have been visited.

By default, the scanning range includes all transactions known to GraphQL, but it can be restricted by the `after` and `before` cursors, and the `beforeCheckpoint`, `afterCheckpoint` and `atCheckpoint` filters.

`Address.transactionBlocks.first`.`Int` scalar

`Address.transactionBlocks.after`.`String` scalar

`Address.transactionBlocks.last`.`Int` scalar

`Address.transactionBlocks.before`.`String` scalar

`Address.transactionBlocks.relation`.`AddressTransactionBlockRelationship` enum

`Address.transactionBlocks.filter`.`TransactionBlockFilter` input

`Address.transactionBlocks.scanLimit`.`Int` scalar

Interfaces

`IOwner` interface

Interface implemented by GraphQL types representing entities that can own objects. Object owners are identified by an address which can represent either the public key of an account or another object. The same address can only refer to an account or an object, never both, but it is not possible to know which up-front.

Returned By

`address` query . `resolveSuinsAddress` query

Member Of

`AddressConnection` object . `AddressEdge` object . `Event` object . `GasInput` object . `Owner` object . `TransactionBlock` object . `Validator` object

Implemented By

`Authenticator` **union**

Edit this page

© 2025 SUI FOUNDATION | DOCUMENTATION DISTRIBUTED UNDER CC BY 4.0

AddressConnection

No description

```
type AddressConnection {
  pageInfo: PageInfo!
  edges: [AddressEdge!]!
  nodes: [Address!]!
}
```

Fields

`AddressConnection.pageInfo.PageInfo!` non-null object

Information to aid in pagination.

`AddressConnection.edges.[AddressEdge!]!` non-null object

A list of edges.

`AddressConnection.nodes.[Address!]!` non-null object

A list of nodes.

Member Of

`Validator` object

Edit this page

© 2025 SUI FOUNDATION | DOCUMENTATION DISTRIBUTED UNDER CC BY 4.0

AddressEdge

An edge in a connection.

```
type AddressEdge {
  node: Address!
  cursor: String!
}
```

Fields

`AddressEdge.node`.`Address!` **non-null** **object**

The item at the end of the edge

`AddressEdge.cursor`.`String!` **non-null** **scalar**

A cursor for use in pagination

Member Of

`AddressConnection` **object**

Edit this page

© 2025 SUI FOUNDATION | DOCUMENTATION DISTRIBUTED UNDER CC BY 4.0

AddressOwner

An address-owned object is owned by a specific 32-byte address that is either an account address (derived from a particular signature scheme) or an object ID. An address-owned object is accessible only to its owner and no others.

```
type AddressOwner {
  owner: Owner
}
```

Fields

`AddressOwner.owner`.`Owner` **object**

Implemented By

`ObjectOwner` **union**

Edit this page

© 2025 SUI FOUNDATION | DOCUMENTATION DISTRIBUTED UNDER CC BY 4.0

AddressTransactionBlockRelationship

The possible relationship types for a transaction block: sent, or received.

```
enum AddressTransactionBlockRelationship {
  SENT
  AFFECTED
}
```

Values

`AddressTransactionBlockRelationship.SENT`

Transactions this address has sent.

`AddressTransactionBlockRelationship.AFFECTED`

Transactions that this address was involved in, either as the sender, sponsor, or as the owner of some object that was created, modified or transfered.

Edit this page

© 2025 SUI FOUNDATION | DOCUMENTATION DISTRIBUTED UNDER CC BY 4.0

Authenticator

An Authenticator represents the access control rules for a ConsensusV2 object.

```
union Authenticator = Address
```

Possible types

`Authenticator.Address` **object**

The 32-byte address that is an account address (corresponding to a public key).

Member Of

`ConsensusV2` **object**

Edit this page

© 2025 SUI FOUNDATION | DOCUMENTATION DISTRIBUTED UNDER CC BY 4.0

AuthenticatorStateCreateTransaction

System transaction for creating the on-chain state used by zkLogin.

```
type AuthenticatorStateCreateTransaction {
  _: Boolean
}
```

Fields

`AuthenticatorStateCreateTransaction._` . `Boolean` **scalar**

A workaround to define an empty variant of a GraphQL union.

Implemented By

`EndOfEpochTransactionKind` **union**

Edit this page

© 2025 SUI FOUNDATION | DOCUMENTATION DISTRIBUTED UNDER CC BY 4.0

AuthenticatorStateExpireTransaction

No description

```
type AuthenticatorStateExpireTransaction {
  minEpoch: Epoch
  authenticatorObjInitialSharedVersion: UInt53!
}
```

Fields

AuthenticatorStateExpireTransaction.**minEpoch**.Epoch **object**

Expire JWKs that have a lower epoch than this.

AuthenticatorStateExpireTransaction.**authenticatorObjInitialSharedVersion**.UInt53! **non-null** **scalar**

The initial version that the AuthenticatorStateUpdate was shared at.

Implemented By

EndOfEpochTransactionKind **union**

Edit this page

© 2025 SUI FOUNDATION | DOCUMENTATION DISTRIBUTED UNDER CC BY 4.0

AuthenticatorStateUpdateTransaction

System transaction for updating the on-chain state used by zkLogin.

```
type AuthenticatorStateUpdateTransaction {
  epoch: Epoch
  round: UInt53!
  newActiveJwks(
    first: Int
    after: String
    last: Int
    before: String
  ): ActiveJwkConnection!
  authenticatorObjInitialSharedVersion: UInt53!
}
```

Fields

`AuthenticatorStateUpdateTransaction.epoch`.`Epoch` **object**

Epoch of the authenticator state update transaction.

`AuthenticatorStateUpdateTransaction.round`.`UInt53!` **non-null** **scalar**

Consensus round of the authenticator state update.

`AuthenticatorStateUpdateTransaction.newActiveJwks`.`ActiveJwkConnection!` **non-null** **object**

Newly active JWKs (JSON Web Keys).

`AuthenticatorStateUpdateTransaction.newActiveJwks.first`.`Int` **scalar**

`AuthenticatorStateUpdateTransaction.newActiveJwks.after`.`String` **scalar**

`AuthenticatorStateUpdateTransaction.newActiveJwks.last`.`Int` **scalar**

`AuthenticatorStateUpdateTransaction.newActiveJwks.before`.`String` **scalar**

`AuthenticatorStateUpdateTransaction.authenticatorObjInitialSharedVersion.UInt53!` **non-null** **scalar**

The initial version of the authenticator object that it was shared at.

Implemented By

`TransactionBlockKind` **union**

Edit this page

© 2025 SUI FOUNDATION | DOCUMENTATION DISTRIBUTED UNDER CC BY 4.0

AvailableRange

Range of checkpoints that the RPC is guaranteed to produce a consistent response for.

```
type AvailableRange {
  first: Checkpoint
  last: Checkpoint
}
```

Fields

`AvailableRange.first.Checkpoint` **object**

`AvailableRange.last.Checkpoint` **object**

Returned By

`availableRange` **query**

Edit this page

© 2025 SUI FOUNDATION | DOCUMENTATION DISTRIBUTED UNDER CC BY 4.0

Balance

The total balance for a particular coin type.

```
type Balance {
  coinType: MoveType!
  coinObjectCount: UInt53
  totalBalance: BigInt
}
```

Fields

Balance.coinType.MoveType! non-null object

Coin type for the balance, such as 0x2::sui::SUI

Balance.coinObjectCount.UInt53 scalar

How many coins of this type constitute the balance

Balance.totalBalance.BigInt scalar

Total balance across all coin objects of the coin type

Member Of

Address object . BalanceConnection object . BalanceEdge object . Coin object . CoinMetadata object . IOwner interface . MoveObject object . MovePackage object . Object object . Owner object . StakedSui object . SuinsRegistration object

Edit this page

© 2025 SUI FOUNDATION | DOCUMENTATION DISTRIBUTED UNDER CC BY 4.0

BalanceChange

Effects to the balance (sum of coin values per coin type) owned by an address or object.

```
type BalanceChange {
  owner: Owner
  coinType: MoveType
  amount: BigInt
}
```

Fields

`BalanceChange.owner.Owner` **object**

The address or object whose balance has changed.

`BalanceChange.coinType.MoveType` **object**

The inner type of the coin whose balance has changed (e.g. `0x2::sui::SUI`).

`BalanceChange.amount.BigInt` **scalar**

The signed balance change.

Member Of

`BalanceChangeConnection` **object** . `BalanceChangeEdge` **object**

Edit this page

© 2025 SUI FOUNDATION | DOCUMENTATION DISTRIBUTED UNDER CC BY 4.0

BalanceChangeConnection

No description

```
type BalanceChangeConnection {
  pageInfo: PageInfo!
  edges: [BalanceChangeEdge!]!
  nodes: [BalanceChange!]!
}
```

Fields

`BalanceChangeConnection.pageInfo.PageInfo!` non-null object

Information to aid in pagination.

`BalanceChangeConnection.edges.[BalanceChangeEdge!]!` non-null object

A list of edges.

`BalanceChangeConnection.nodes.[BalanceChange!]!` non-null object

A list of nodes.

Member Of

`TransactionBlockEffects` object

Edit this page

© 2025 SUI FOUNDATION | DOCUMENTATION DISTRIBUTED UNDER CC BY 4.0

BalanceChangeEdge

An edge in a connection.

```
type BalanceChangeEdge {
  node: BalanceChange!
  cursor: String!
}
```

Fields

`BalanceChangeEdge.node`.`BalanceChange!` **non-null** **object**

The item at the end of the edge

`BalanceChangeEdge.cursor`.`String!` **non-null** **scalar**

A cursor for use in pagination

Member Of

`BalanceChangeConnection` **object**

Edit this page

© 2025 SUI FOUNDATION | DOCUMENTATION DISTRIBUTED UNDER CC BY 4.0

BalanceConnection

No description

```
type BalanceConnection {
  pageInfo: PageInfo!
  edges: [BalanceEdge!]!
  nodes: [Balance!]!
}
```

Fields

`BalanceConnection.pageInfo` . `PageInfo!` **non-null** **object**

Information to aid in pagination.

`BalanceConnection.edges` . `[BalanceEdge!]!` **non-null** **object**

A list of edges.

`BalanceConnection.nodes` . `[Balance!]!` **non-null** **object**

A list of nodes.

Member Of

`Address` **object** . `Coin` **object** . `CoinMetadata` **object** . `IOwner` **interface** . `MoveObject` **object** . `MovePackage` **object** . `Object` **object** . `Owner` **object** . `StakedSui` **object** . `SuinsRegistration` **object**

Edit this page

© 2025 SUI FOUNDATION | DOCUMENTATION DISTRIBUTED UNDER CC BY 4.0

BalanceEdge

An edge in a connection.

```
type BalanceEdge {
  node: Balance!
  cursor: String!
}
```

Fields

`BalanceEdge.node`.`Balance!` **non-null** **object**

The item at the end of the edge

`BalanceEdge.cursor`.`String!` **non-null** **scalar**

A cursor for use in pagination

Member Of

`BalanceConnection` **object**

Edit this page

© 2025 SUI FOUNDATION | DOCUMENTATION DISTRIBUTED UNDER CC BY 4.0

Base64

String containing Base64-encoded binary data.

```
scalar Base64
```

Member Of

Checkpoint object . Coin object . CoinMetadata object . DryRunMutation object . DryRunReturn object . DynamicFieldName input . Event object . IObject interface . MoveModule object . MoveObject object . MovePackage object . MoveValue object . Object object . PublishTransaction object . Pure object . RandomnessStateUpdateTransaction object . StakedSui object . SuinsRegistration object . TransactionBlock object . TransactionBlockEffects object . UpgradeTransaction object . ValidatorCredentials object . verifyZkloginSignature query

Edit this page

© 2025 SUI FOUNDATION | DOCUMENTATION DISTRIBUTED UNDER CC BY 4.0

BigInt

String representation of an arbitrary width, possibly signed integer.

```
scalar BigInt
```

Member Of

Balance object . BalanceChange object . ChangeEpochTransaction object . Coin object . CoinMetadata object . Epoch object . GasCostSummary object . GasInput object . IObject interface . MoveObject object . MovePackage object . Object object . StakedSui object . StakeSubsidy object . StorageFund object . SuinsRegistration object . SystemParameters object . Validator object . ValidatorSet object

Edit this page

© 2025 SUI FOUNDATION | DOCUMENTATION DISTRIBUTED UNDER CC BY 4.0

Boolean

The `Boolean` scalar type represents `true` or `false`.

```
scalar Boolean
```

Member Of

AuthenticatorStateCreateTransaction object • Coin object • CoinDenyListStateCreateTransaction object • CoinMetadata object • dryRunTransactionBlock query • GasCoin object • Immutable object • IMoveObject interface • include directive • MoveFunction object • MoveObject object • MoveStructTypeParameter object • ObjectChange object • PageInfo object • ProtocolConfigFeatureFlag object • RandomnessStateCreateTransaction object • SafeMode object • ServiceConfig object • SharedInput object • SharedObjectDelete object • skip directive • StakedSui object • SuinsRegistration object • ZkLoginVerifyResult object

Edit this page

© 2025 SUI FOUNDATION | DOCUMENTATION DISTRIBUTED UNDER CC BY 4.0

BridgeCommitteeInitTransaction

No description

```
type BridgeCommitteeInitTransaction {
  bridgeObjInitialSharedVersion: UInt53!
}
```

Fields

`BridgeCommitteeInitTransaction.bridgeObjInitialSharedVersion.UInt53!` **non-null** **scalar**

Implemented By

`EndOfEpochTransactionKind` **union**

Edit this page

© 2025 SUI FOUNDATION | DOCUMENTATION DISTRIBUTED UNDER CC BY 4.0

BridgeStateCreateTransaction

No description

```
type BridgeStateCreateTransaction {
  chainId: String!
}
```

Fields

`BridgeStateCreateTransaction.chainId`.`String!` **non-null** **scalar**

Implemented By

`EndOfEpochTransactionKind` **union**

Edit this page

© 2025 SUI FOUNDATION | DOCUMENTATION DISTRIBUTED UNDER CC BY 4.0

chainIdentifier

First four bytes of the network's genesis checkpoint digest (uniquely identifies the network).

```
chainIdentifier: String!
```

Type

`String` **scalar**

The `String` scalar type represents textual data, represented as UTF-8 character sequences. The String type is most often used by GraphQL to represent free-form human-readable text.

Edit this page

© 2025 SUI FOUNDATION | DOCUMENTATION DISTRIBUTED UNDER CC BY 4.0

ChangeEpochTransaction

A system transaction that updates epoch information on-chain (increments the current epoch). Executed by the system once per epoch, without using gas. Epoch change transactions cannot be submitted by users, because validators will refuse to sign them.

This transaction kind is deprecated in favour of `EndOfEpochTransaction`.

```
type ChangeEpochTransaction {
  epoch: Epoch
  protocolVersion: UInt53!
  storageCharge: BigInt!
  computationCharge: BigInt!
  storageRebate: BigInt!
  nonRefundableStorageFee: BigInt!
  startTimestamp: DateTime!
  systemPackages(
    first: Int
    after: String
    last: Int
    before: String
  ): MovePackageConnection!
}
```

Fields

`ChangeEpochTransaction.epoch.Epoch` **object**

The next (to become) epoch.

`ChangeEpochTransaction.protocolVersion.UInt53!` **non-null** **scalar**

The protocol version in effect in the new epoch.

`ChangeEpochTransaction.storageCharge.BigInt!` **non-null** **scalar**

The total amount of gas charged for storage during the previous epoch (in MIST).

`ChangeEpochTransaction.computationCharge.BigInt!` **non-null** **scalar**

The total amount of gas charged for computation during the previous epoch (in MIST).

`ChangeEpochTransaction.storageRebate.BigInt!` non-null scalar

The SUI returned to transaction senders for cleaning up objects (in MIST).

`ChangeEpochTransaction.nonRefundableStorageFee.BigInt!` non-null scalar

The total gas retained from storage fees, that will not be returned by storage rebates when the relevant objects are cleaned up (in MIST).

`ChangeEpochTransaction.startTimestamp.DateTime!` non-null scalar

Time at which the next epoch will start.

`ChangeEpochTransaction.systemPackages.MovePackageConnection!` non-null object

System packages (specifically framework and move stdlib) that are written before the new epoch starts, to upgrade them on-chain. Validators write these packages out when running the transaction.

`ChangeEpochTransaction.systemPackages.first.Int` scalar

`ChangeEpochTransaction.systemPackages.after.String` scalar

`ChangeEpochTransaction.systemPackages.last.Int` scalar

`ChangeEpochTransaction.systemPackages.before.String` scalar

Implemented By

`EndOfEpochTransactionKind` union . `TransactionBlockKind` union

Edit this page

© 2025 SUI FOUNDATION | DOCUMENTATION DISTRIBUTED UNDER CC BY 4.0

Checkpoint

Checkpoints contain finalized transactions and are used for node synchronization and global transaction ordering.

```
type Checkpoint {
  digest: String!
  sequenceNumber: UInt53!
  timestamp: DateTime!
  validatorSignatures: Base64!
  previousCheckpointDigest: String
  networkTotalTransactions: UInt53
  rollingGasSummary: GasCostSummary
  epoch: Epoch
  transactionBlocks(
    first: Int
    after: String
    last: Int
    before: String
    filter: TransactionBlockFilter
    scanLimit: Int
  ): TransactionBlockConnection!
  bcs: Base64
}
```

Fields

`Checkpoint.digest.String!` non-null scalar

A 32-byte hash that uniquely identifies the checkpoint contents, encoded in Base58. This hash can be used to verify checkpoint contents by checking signatures against the committee, Hashing contents to match digest, and checking that the previous checkpoint digest matches.

`Checkpoint.sequenceNumber.UInt53!` non-null scalar

This checkpoint's position in the total order of finalized checkpoints, agreed upon by consensus.

`Checkpoint.timestamp.DateTime!` non-null scalar

The timestamp at which the checkpoint is agreed to have happened according to consensus. Transactions that access time in this checkpoint will observe this timestamp.

`Checkpoint.validatorSignatures`.`Base64!` non-null scalar

This is an aggregation of signatures from a quorum of validators for the checkpoint proposal.

`Checkpoint.previousCheckpointDigest`.`String` scalar

The digest of the checkpoint at the previous sequence number.

`Checkpoint.networkTotalTransactions`.`UInt53` scalar

The total number of transaction blocks in the network by the end of this checkpoint.

`Checkpoint.rollingGasSummary`.`GasCostSummary` object

The computation cost, storage cost, storage rebate, and non-refundable storage fee accumulated during this epoch, up to and including this checkpoint. These values increase monotonically across checkpoints in the same epoch, and reset on epoch boundaries.

`Checkpoint.epoch`.`Epoch` object

The epoch this checkpoint is part of.

`Checkpoint.transactionBlocks`.`TransactionBlockConnection!` non-null object

Transactions in this checkpoint.

`scanLimit` restricts the number of candidate transactions scanned when gathering a page of results. It is required for queries that apply more than two complex filters (on function, kind, sender, recipient, input object, changed object, or ids), and can be at most `serviceConfig.maxScanLimit`.

When the scan limit is reached the page will be returned even if it has fewer than `first` results when paginating forward (`last` when paginating backwards). If there are more transactions to scan, `pageInfo.hasNextPage` (or `pageInfo.hasPreviousPage`) will be set to `true`, and `PageInfo.endCursor` (or `PageInfo.startCursor`) will be set to the last transaction that was scanned as opposed to the last (or first) transaction in the page.

Requesting the next (or previous) page after this cursor will resume the search, scanning the next `scanLimit` many transactions in the direction of pagination, and so on until all transactions in the scanning range have been visited.

By default, the scanning range consists of all transactions in this checkpoint.

`Checkpoint.transactionBlocks.first`.`Int` scalar

`Checkpoint.transactionBlocks.after`.`String` scalar

`Checkpoint.transactionBlocks.last`.`Int` scalar

`Checkpoint.transactionBlocks.before`.`String` scalar

`Checkpoint.transactionBlocks.filter`.`TransactionBlockFilter` input

`Checkpoint.transactionBlocks.scanLimit`.`Int` scalar

`Checkpoint.bcs`.`Base64` scalar

The Base64 serialized BCS bytes of CheckpointSummary for this checkpoint.

Returned By

`checkpoint` query

Member Of

`AvailableRange` object.`CheckpointConnection` object.`CheckpointEdge` object.`TransactionBlockEffects` object

Edit this page

© 2025 SUI FOUNDATION | DOCUMENTATION DISTRIBUTED UNDER CC BY 4.0

CheckpointConnection

No description

```
type CheckpointConnection {
  pageInfo: PageInfo!
  edges: [CheckpointEdge!]!
  nodes: [Checkpoint!]!
}
```

Fields

`CheckpointConnection.pageInfo`.`PageInfo!` **non-null** **object**

Information to aid in pagination.

`CheckpointConnection.edges`.`[CheckpointEdge!]!` **non-null** **object**

A list of edges.

`CheckpointConnection.nodes`.`[Checkpoint!]!` **non-null** **object**

A list of nodes.

Returned By

`checkpoints` **query**

Member Of

`Epoch` **object**

Edit this page

© 2025 SUI FOUNDATION | DOCUMENTATION DISTRIBUTED UNDER CC BY 4.0

CheckpointEdge

An edge in a connection.

```
type CheckpointEdge {
  node: Checkpoint!
  cursor: String!
}
```

Fields

`CheckpointEdge.node`.`Checkpoint!` **non-null** **object**

The item at the end of the edge

`CheckpointEdge.cursor`.`String!` **non-null** **scalar**

A cursor for use in pagination

Member Of

`CheckpointConnection` **object**

Edit this page

© 2025 SUI FOUNDATION | DOCUMENTATION DISTRIBUTED UNDER CC BY 4.0

CheckpointId

Filter either by the digest, or the sequence number, or neither, to get the latest checkpoint.

```
input CheckpointId {
  digest: String
  sequenceNumber: UInt53
}
```

Fields

`CheckpointId.digest`.`String` **scalar**

`CheckpointId.sequenceNumber`.`UInt53` **scalar**

Member Of

`checkpoint` **query**

Edit this page

© 2025 SUI FOUNDATION | DOCUMENTATION DISTRIBUTED UNDER CC BY 4.0

checkpoints

The checkpoints that exist in the network.

```
checkpoints(
  first: Int
  after: String
  last: Int
  before: String
): CheckpointConnection!
```

Arguments

`checkpoints.first.Int` **scalar**

`checkpoints.after.String` **scalar**

`checkpoints.last.Int` **scalar**

`checkpoints.before.String` **scalar**

Type

`CheckpointConnection` **object**

Edit this page

© 2025 SUI FOUNDATION | DOCUMENTATION DISTRIBUTED UNDER CC BY 4.0

Coin

Some 0x2::coin::Coin Move object.

```
type Coin implements IMoveObject, IObject, IOwner {
  address: SuiAddress!
  objects(
    first: Int
    after: String
    last: Int
    before: String
    filter: ObjectFilter
  ): MoveObjectConnection!
  balance(
    type: String
  ): Balance
  balances(
    first: Int
    after: String
    last: Int
    before: String
  ): BalanceConnection!
  coins(
    first: Int
    after: String
    last: Int
    before: String
    type: String
  ): CoinConnection!
  stakedSuis(
    first: Int
    after: String
    last: Int
    before: String
  ): StakedSuiConnection!
  defaultSuinsName(
    format: DomainFormat
  ): String
  suinsRegistrations(
    first: Int
    after: String
    last: Int
    before: String
  ): SuinsRegistrationConnection!
  version: UInt53!
  status: ObjectKind!
  digest: String
  owner: ObjectOwner
```

```
  previousTransactionBlock: TransactionBlock
  storageRebate: BigInt
  receivedTransactionBlocks(
    first: Int
    after: String
    last: Int
    before: String
    filter: TransactionBlockFilter
    scanLimit: Int
  ): TransactionBlockConnection!
  bcs: Base64
  contents: MoveValue
  hasPublicTransfer: Boolean!
  display: [DisplayEntry!]
  dynamicField(
    name: DynamicFieldName!
  ): DynamicField
  dynamicObjectField(
    name: DynamicFieldName!
  ): DynamicField
  dynamicFields(
    first: Int
    after: String
    last: Int
    before: String
  ): DynamicFieldConnection!
  coinBalance: BigInt
}
```

Fields

Coin.address.SuiAddress! non-null scalar

Coin.objects.MoveObjectConnection! non-null object

Objects owned by this object, optionally `filter`-ed.

`Coin.objects.first`.`Int` scalar

`Coin.objects.after`.`String` scalar

`Coin.objects.last`.`Int` scalar

`Coin.objects.before`.`String` scalar

`Coin.objects.filter`.`ObjectFilter` input

Coin.balance.Balance object

Total balance of all coins with marker type owned by this object. If type is not supplied, it defaults to `0x2::sui::SUI`.

`Coin.balance.type`.`String` scalar

`Coin.balances`.`BalanceConnection!` non-null object

The balances of all coin types owned by this object.

`Coin.balances.first`.`Int` scalar

`Coin.balances.after`.`String` scalar

`Coin.balances.last`.`Int` scalar

`Coin.balances.before`.`String` scalar

`Coin.coins`.`CoinConnection!` non-null object

The coin objects for this object.

`type` is a filter on the coin's type parameter, defaulting to `0x2::sui::SUI`.

`Coin.coins.first`.`Int` scalar

`Coin.coins.after`.`String` scalar

`Coin.coins.last`.`Int` scalar

`Coin.coins.before`.`String` scalar

`Coin.coins.type`.`String` scalar

`Coin.stakedSuis`.`StakedSuiConnection!` non-null object

The `0x3::staking_pool::StakedSui` objects owned by this object.

`Coin.stakedSuis.first`.`Int` scalar

`Coin.stakedSuis.after`.`String` scalar

`Coin.stakedSuis.last`.`Int` scalar

`Coin.stakedSuis.before`.`String` scalar

`Coin.defaultSuinsName`.`String` scalar

The domain explicitly configured as the default domain pointing to this object.

`Coin.defaultSuinsName.format`.`DomainFormat` enum

Coin.suinsRegistrations.SuinsRegistrationConnection! non-null object

The SuinsRegistration NFTs owned by this object. These grant the owner the capability to manage the associated domain.

Coin.suinsRegistrations.first.Int scalar

Coin.suinsRegistrations.after.String scalar

Coin.suinsRegistrations.last.Int scalar

Coin.suinsRegistrations.before.String scalar

Coin.version.UInt53! non-null scalar

Coin.status.ObjectKind! non-null enum

The current status of the object as read from the off-chain store. The possible states are: NOT_INDEXED, the object is loaded from serialized data, such as the contents of a genesis or system package upgrade transaction. LIVE, the version returned is the most recent for the object, and it is not deleted or wrapped at that version. HISTORICAL, the object was referenced at a specific version or checkpoint, so is fetched from historical tables and may not be the latest version of the object. WRAPPED_OR_DELETED, the object is deleted or wrapped and only partial information can be loaded."

Coin.digest.String scalar

32-byte hash that identifies the object's contents, encoded as a Base58 string.

Coin.owner.ObjectOwner union

The owner type of this object: Immutable, Shared, Parent, Address

Coin.previousTransactionBlock.TransactionBlock object

The transaction block that created this version of the object.

Coin.storageRebate.BigInt scalar

The amount of SUI we would rebate if this object gets deleted or mutated. This number is recalculated based on the present storage gas price.

Coin.receivedTransactionBlocks.TransactionBlockConnection! non-null object

The transaction blocks that sent objects to this object.

`scanLimit` restricts the number of candidate transactions scanned when gathering a page of results. It is required for queries that apply more than two complex filters (on function, kind, sender, recipient, input object, changed object, or ids), and can be at most `serviceConfig.maxScanLimit`.

When the scan limit is reached the page will be returned even if it has fewer than `first` results when paginating forward (`last` when paginating backwards). If there are more transactions to scan, `pageInfo.hasNextPage` (or `pageInfo.hasPreviousPage`) will be set to `true`, and `PageInfo.endCursor` (or `PageInfo.startCursor`) will be set to the last transaction that was scanned as opposed to the last (or first) transaction in the page.

Requesting the next (or previous) page after this cursor will resume the search, scanning the next `scanLimit` many transactions in the direction of pagination, and so on until all transactions in the scanning range have been visited.

By default, the scanning range includes all transactions known to GraphQL, but it can be restricted by the `after` and `before` cursors, and the `beforeCheckpoint`, `afterCheckpoint` and `atCheckpoint` filters.

`Coin.receivedTransactionBlocks.first`.`Int` scalar

`Coin.receivedTransactionBlocks.after`.`String` scalar

`Coin.receivedTransactionBlocks.last`.`Int` scalar

`Coin.receivedTransactionBlocks.before`.`String` scalar

`Coin.receivedTransactionBlocks.filter`.`TransactionBlockFilter` input

`Coin.receivedTransactionBlocks.scanLimit`.`Int` scalar

`Coin.bcs`.`Base64` scalar

The Base64-encoded BCS serialization of the object's content.

`Coin.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.

`Coin.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.

`Coin.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.

`Coin.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 Owner type.

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

`Coin.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.

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

`Coin.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.

`Coin.dynamicFields.first`.`Int` **scalar**

`Coin.dynamicFields.after`.`String` **scalar**

`Coin.dynamicFields.last`.`Int` **scalar**

`Coin.dynamicFields.before`.`String` **scalar**

`Coin.coinBalance`.`BigInt` **scalar**

Balance of this coin object.

Interfaces

`IMoveObject` **interface**

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

`IObject` interface

Interface implemented by on-chain values that are addressable by an ID (also referred to as its address). This includes Move objects and packages.

`IOwner` interface

Interface implemented by GraphQL types representing entities that can own objects. Object owners are identified by an address which can represent either the public key of an account or another object. The same address can only refer to an account or an object, never both, but it is not possible to know which up-front.

Member Of

`CoinConnection` object . `CoinEdge` object . `MoveObject` object

Edit this page

© 2025 SUI FOUNDATION | DOCUMENTATION DISTRIBUTED UNDER CC BY 4.0

CoinConnection

No description

```
type CoinConnection {
  pageInfo: PageInfo!
  edges: [CoinEdge!]!
  nodes: [Coin!]!
}
```

Fields

CoinConnection.pageInfo.PageInfo! non-null object

Information to aid in pagination.

CoinConnection.edges.[CoinEdge!]! non-null object

A list of edges.

CoinConnection.nodes.[Coin!]! non-null object

A list of nodes.

Returned By

coins query

Member Of

Address object . Coin object . CoinMetadata object . IOwner interface . MoveObject object . MovePackage object . Object object . Owner object . StakedSui object . SuinsRegistration object

Edit this page

© 2025 SUI FOUNDATION | DOCUMENTATION DISTRIBUTED UNDER CC BY 4.0

CoinDenyListStateCreateTransaction

No description

```
type CoinDenyListStateCreateTransaction {
  _: Boolean
}
```

Fields

`CoinDenyListStateCreateTransaction._` . `Boolean` **scalar**

A workaround to define an empty variant of a GraphQL union.

Implemented By

`EndOfEpochTransactionKind` **union**

Edit this page

© 2025 SUI FOUNDATION | DOCUMENTATION DISTRIBUTED UNDER CC BY 4.0

CoinEdge

An edge in a connection.

```
type CoinEdge {
  node: Coin!
  cursor: String!
}
```

Fields

`CoinEdge.node`.`Coin!` **non-null** **object**

The item at the end of the edge

`CoinEdge.cursor`.`String!` **non-null** **scalar**

A cursor for use in pagination

Member Of

`CoinConnection` **object**

Edit this page

© 2025 SUI FOUNDATION | DOCUMENTATION DISTRIBUTED UNDER CC BY 4.0

CoinMetadata

The metadata for a coin type.

```
type CoinMetadata implements IMoveObject, IObject, IOwner {
  address: SuiAddress!
  objects(
    first: Int
    after: String
    last: Int
    before: String
    filter: ObjectFilter
  ): MoveObjectConnection!
  balance(
    type: String
  ): Balance
  balances(
    first: Int
    after: String
    last: Int
    before: String
  ): BalanceConnection!
  coins(
    first: Int
    after: String
    last: Int
    before: String
    type: String
  ): CoinConnection!
  stakedSuis(
    first: Int
    after: String
    last: Int
    before: String
  ): StakedSuiConnection!
  defaultSuinsName(
    format: DomainFormat
  ): String
  suinsRegistrations(
    first: Int
    after: String
    last: Int
    before: String
  ): SuinsRegistrationConnection!
  version: UInt53!
  status: ObjectKind!
  digest: String
  owner: ObjectOwner
```

```
  previousTransactionBlock: TransactionBlock
  storageRebate: BigInt
  receivedTransactionBlocks(
    first: Int
    after: String
    last: Int
    before: String
    filter: TransactionBlockFilter
    scanLimit: Int
  ): TransactionBlockConnection!
  bcs: Base64
  contents: MoveValue
  hasPublicTransfer: Boolean!
  display: [DisplayEntry!]
  dynamicField(
    name: DynamicFieldName!
  ): DynamicField
  dynamicObjectField(
    name: DynamicFieldName!
  ): DynamicField
  dynamicFields(
    first: Int
    after: String
    last: Int
    before: String
  ): DynamicFieldConnection!
  decimals: Int
  name: String
  symbol: String
  description: String
  iconUrl: String
  supply: BigInt
}
```

Fields

`CoinMetadata.address`.`SuiAddress!` non-null scalar

`CoinMetadata.objects`.`MoveObjectConnection!` non-null object

Objects owned by this object, optionally `filter`-ed.

`CoinMetadata.objects.first`.`Int` scalar

`CoinMetadata.objects.after`.`String` scalar

`CoinMetadata.objects.last`.`Int` scalar

`CoinMetadata.objects.before`.`String` scalar

`CoinMetadata.objects.filter`.`ObjectFilter` input

`CoinMetadata.balance`.`Balance` object

Total balance of all coins with marker type owned by this object. If type is not supplied, it defaults to `0x2::sui::SUI`.

`CoinMetadata.balance.type`.`String` scalar

`CoinMetadata.balances`.`BalanceConnection!` non-null object

The balances of all coin types owned by this object.

`CoinMetadata.balances.first`.`Int` scalar

`CoinMetadata.balances.after`.`String` scalar

`CoinMetadata.balances.last`.`Int` scalar

`CoinMetadata.balances.before`.`String` scalar

`CoinMetadata.coins`.`CoinConnection!` non-null object

The coin objects for this object.

`type` is a filter on the coin's type parameter, defaulting to `0x2::sui::SUI`.

`CoinMetadata.coins.first`.`Int` scalar

`CoinMetadata.coins.after`.`String` scalar

`CoinMetadata.coins.last`.`Int` scalar

`CoinMetadata.coins.before`.`String` scalar

`CoinMetadata.coins.type`.`String` scalar

`CoinMetadata.stakedSuis`.`StakedSuiConnection!` non-null object

The `0x3::staking_pool::StakedSui` objects owned by this object.

`CoinMetadata.stakedSuis.first`.`Int` scalar

`CoinMetadata.stakedSuis.after`.`String` scalar

`CoinMetadata.stakedSuis.last`.`Int` scalar

`CoinMetadata.stakedSuis.before`.`String` scalar

`CoinMetadata.defaultSuinsName`.`String` scalar

The domain explicitly configured as the default domain pointing to this object.

`CoinMetadata.defaultSuinsName.format`.`DomainFormat` enum

`CoinMetadata.suinsRegistrations`.`SuinsRegistrationConnection!` non-null object

The SuinsRegistration NFTs owned by this object. These grant the owner the capability to manage the associated domain.

`CoinMetadata.suinsRegistrations.first`.`Int` scalar

`CoinMetadata.suinsRegistrations.after`.`String` scalar

`CoinMetadata.suinsRegistrations.last`.`Int` scalar

`CoinMetadata.suinsRegistrations.before`.`String` scalar

`CoinMetadata.version`.`UInt53!` non-null scalar

`CoinMetadata.status`.`ObjectKind!` non-null enum

The current status of the object as read from the off-chain store. The possible states are: NOT_INDEXED, the object is loaded from serialized data, such as the contents of a genesis or system package upgrade transaction. LIVE, the version returned is the most recent for the object, and it is not deleted or wrapped at that version. HISTORICAL, the object was referenced at a specific version or checkpoint, so is fetched from historical tables and may not be the latest version of the object. WRAPPED_OR_DELETED, the object is deleted or wrapped and only partial information can be loaded."

`CoinMetadata.digest`.`String` scalar

32-byte hash that identifies the object's contents, encoded as a Base58 string.

`CoinMetadata.owner`.`ObjectOwner` union

The owner type of this object: Immutable, Shared, Parent, Address

`CoinMetadata.previousTransactionBlock`.`TransactionBlock` object

The transaction block that created this version of the object.

CoinMetadata.storageRebate.BigInt scalar

The amount of SUI we would rebate if this object gets deleted or mutated. This number is recalculated based on the present storage gas price.

CoinMetadata.receivedTransactionBlocks.TransactionBlockConnection! non-null object

The transaction blocks that sent objects to this object.

`scanLimit` restricts the number of candidate transactions scanned when gathering a page of results. It is required for queries that apply more than two complex filters (on function, kind, sender, recipient, input object, changed object, or ids), and can be at most `serviceConfig.maxScanLimit`.

When the scan limit is reached the page will be returned even if it has fewer than `first` results when paginating forward (`last` when paginating backwards). If there are more transactions to scan, `pageInfo.hasNextPage` (or `pageInfo.hasPreviousPage`) will be set to `true`, and `PageInfo.endCursor` (or `PageInfo.startCursor`) will be set to the last transaction that was scanned as opposed to the last (or first) transaction in the page.

Requesting the next (or previous) page after this cursor will resume the search, scanning the next `scanLimit` many transactions in the direction of pagination, and so on until all transactions in the scanning range have been visited.

By default, the scanning range includes all transactions known to GraphQL, but it can be restricted by the `after` and `before` cursors, and the `beforeCheckpoint`, `afterCheckpoint` and `atCheckpoint` filters.

`CoinMetadata.receivedTransactionBlocks.first.Int` scalar

`CoinMetadata.receivedTransactionBlocks.after.String` scalar

`CoinMetadata.receivedTransactionBlocks.last.Int` scalar

`CoinMetadata.receivedTransactionBlocks.before.String` scalar

`CoinMetadata.receivedTransactionBlocks.filter.TransactionBlockFilter` input

`CoinMetadata.receivedTransactionBlocks.scanLimit.Int` scalar

CoinMetadata.bcs.Base64 scalar

The Base64-encoded BCS serialization of the object's content.

CoinMetadata.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.

CoinMetadata.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.

CoinMetadata.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.

CoinMetadata.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 Owner type.

`CoinMetadata.dynamicField.name`.`DynamicFieldName!` non-null input

CoinMetadata.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.

`CoinMetadata.dynamicObjectField.name`.`DynamicFieldName!` non-null input

CoinMetadata.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.

`CoinMetadata.dynamicFields.first`.`Int` scalar

`CoinMetadata.dynamicFields.after`.`String` scalar

`CoinMetadata.dynamicFields.last`.`Int` scalar

`CoinMetadata.dynamicFields.before`.`String` scalar

`CoinMetadata.decimals.Int` **scalar**

The number of decimal places used to represent the token.

`CoinMetadata.name.String` **scalar**

Full, official name of the token.

`CoinMetadata.symbol.String` **scalar**

The token's identifying abbreviation.

`CoinMetadata.description.String` **scalar**

Optional description of the token, provided by the creator of the token.

`CoinMetadata.iconUrl.String` **scalar**

`CoinMetadata.supply.BigInt` **scalar**

The overall quantity of tokens that will be issued.

Interfaces

`IMoveObject` **interface**

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

`IObject` **interface**

Interface implemented by on-chain values that are addressable by an ID (also referred to as its address). This includes Move objects and packages.

`IOwner` **interface**

Interface implemented by GraphQL types representing entities that can own objects. Object owners are identified by an address which can represent either the public key of an account or another object. The same address can only refer to an account or an object, never both, but it is not possible to know which up-front.

Returned By

`coinMetadata` **query**

Member Of

`MoveObject` **object**

Edit this page

© 2025 SUI FOUNDATION | DOCUMENTATION DISTRIBUTED UNDER CC BY 4.0

coins

The coin objects that exist in the network.

The type field is a string of the inner type of the coin by which to filter (e.g. `0x2::sui::SUI`). If no type is provided, it will default to `0x2::sui::SUI`.

```
coins(
  first: Int
  after: String
  last: Int
  before: String
  type: String
): CoinConnection!
```

Arguments

`coins.first`.`Int` **scalar**

`coins.after`.`String` **scalar**

`coins.last`.`Int` **scalar**

`coins.before`.`String` **scalar**

`coins.type`.`String` **scalar**

Type

`CoinConnection` **object**

Edit this page

© 2025 SUI FOUNDATION | DOCUMENTATION DISTRIBUTED UNDER CC BY 4.0

ConsensusCommitPrologueTransaction

System transaction that runs at the beginning of a checkpoint, and is responsible for setting the current value of the clock, based on the timestamp from consensus.

```
type ConsensusCommitPrologueTransaction {
  epoch: Epoch
  round: UInt53!
  commitTimestamp: DateTime!
  consensusCommitDigest: String
}
```

Fields

`ConsensusCommitPrologueTransaction.epoch.Epoch` **object**

Epoch of the commit prologue transaction.

`ConsensusCommitPrologueTransaction.round.UInt53!` **non-null** **scalar**

Consensus round of the commit.

`ConsensusCommitPrologueTransaction.commitTimestamp.DateTime!` **non-null** **scalar**

Unix timestamp from consensus.

`ConsensusCommitPrologueTransaction.consensusCommitDigest.String` **scalar**

Digest of consensus output, encoded as a Base58 string (only available from V2 of the transaction).

Implemented By

`TransactionBlockKind` **union**

Edit this page

© 2025 SUI FOUNDATION | DOCUMENTATION DISTRIBUTED UNDER CC BY 4.0

ConsensusV2

A ConsensusV2 object is an object that is automatically versioned by the consensus protocol and allows different authentication modes based on the chosen authenticator. (Initially, only single-owner authentication is supported.)

```
type ConsensusV2 {
  startVersion: UInt53!
  authenticator: Authenticator
}
```

Fields

`ConsensusV2.startVersion`.`UInt53!` **non-null** **scalar**

`ConsensusV2.authenticator`.`Authenticator` **union**

Implemented By

`ObjectOwner` **union**

Edit this page

© 2025 SUI FOUNDATION | DOCUMENTATION DISTRIBUTED UNDER CC BY 4.0

DateTime

ISO-8601 Date and Time: RFC3339 in UTC with format: YYYY-MM-DDTHH:MM:SS.mmmZ. Note that the milliseconds part is optional, and it may be omitted if its value is 0.

```
scalar DateTime
```

Member Of

ChangeEpochTransaction object . Checkpoint object . ConsensusCommitPrologueTransaction object . Epoch object . Event object . TransactionBlockEffects object

Edit this page

© 2025 SUI FOUNDATION | DOCUMENTATION DISTRIBUTED UNDER CC BY 4.0

DependencyConnection

No description

```
type DependencyConnection {
  pageInfo: PageInfo!
  edges: [DependencyEdge!]!
  nodes: [TransactionBlock!]!
}
```

Fields

`DependencyConnection.pageInfo`.`PageInfo!` **non-null** **object**

Information to aid in pagination.

`DependencyConnection.edges`.`[DependencyEdge!]!` **non-null** **object**

A list of edges.

`DependencyConnection.nodes`.`[TransactionBlock!]!` **non-null** **object**

A list of nodes.

Member Of

`TransactionBlockEffects` **object**

Edit this page

© 2025 SUI FOUNDATION | DOCUMENTATION DISTRIBUTED UNDER CC BY 4.0

DependencyEdge

An edge in a connection.

```
type DependencyEdge {
  node: TransactionBlock
  cursor: String!
}
```

Fields

`DependencyEdge.node`.`TransactionBlock` **object**

The item at the end of the edge

`DependencyEdge.cursor`.`String!` **non-null** **scalar**

A cursor for use in pagination

Member Of

`DependencyConnection` **object**

Edit this page

© 2025 SUI FOUNDATION | DOCUMENTATION DISTRIBUTED UNDER CC BY 4.0

deprecated

Marks an element of a GraphQL schema as no longer supported.

```
directive @deprecated(
  reason: String = "No longer supported"
) on
  | FIELD_DEFINITION
  | ARGUMENT_DEFINITION
  | INPUT_FIELD_DEFINITION
  | ENUM_VALUE
```

Arguments

`deprecated.reason`.`String` **scalar**

Explains why this element was deprecated, usually also including a suggestion for how to access supported similar data. Formatted using the Markdown syntax, as specified by CommonMark.

Edit this page

© 2025 SUI FOUNDATION | DOCUMENTATION DISTRIBUTED UNDER CC BY 4.0

DisplayEntry

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.

```
type DisplayEntry {
  key: String!
  value: String
  error: String
}
```

Fields

DisplayEntry.key.String! non-null scalar

The identifier for a particular template string of the Display object.

DisplayEntry.value.String scalar

The template string for the key with placeholder values substituted.

DisplayEntry.error.String scalar

An error string describing why the template could not be rendered.

Member Of

Coin object . CoinMetadata object . IMoveObject interface . MoveObject object . Object object . StakedSui object . SuinsRegistration object

Edit this page

© 2025 SUI FOUNDATION | DOCUMENTATION DISTRIBUTED UNDER CC BY 4.0

DomainFormat

No description

```
enum DomainFormat {
  AT
  DOT
}
```

Values

`DomainFormat.AT`

`DomainFormat.DOT`

Edit this page

© 2025 SUI FOUNDATION | DOCUMENTATION DISTRIBUTED UNDER CC BY 4.0

DryRunEffect

No description

```
type DryRunEffect {
  mutatedReferences: [DryRunMutation!]
  returnValues: [DryRunReturn!]
}
```

Fields

`DryRunEffect.mutatedReferences.[DryRunMutation!]` **list** **object**

Changes made to arguments that were mutably borrowed by each command in this transaction.

`DryRunEffect.returnValues.[DryRunReturn!]` **list** **object**

Return results of each command in this transaction.

Member Of

`DryRunResult` **object**

Edit this page

© 2025 SUI FOUNDATION | DOCUMENTATION DISTRIBUTED UNDER CC BY 4.0

DryRunMutation

No description

```
type DryRunMutation {
  input: TransactionArgument!
  type: MoveType!
  bcs: Base64!
}
```

Fields

`DryRunMutation.input`.`TransactionArgument!` **non-null** **union**

`DryRunMutation.type`.`MoveType!` **non-null** **object**

`DryRunMutation.bcs`.`Base64!` **non-null** **scalar**

Member Of

`DryRunEffect` **object**

Edit this page

© 2025 SUI FOUNDATION | DOCUMENTATION DISTRIBUTED UNDER CC BY 4.0

DryRunResult

No description

```
type DryRunResult {
  error: String
  results: [DryRunEffect!]
  transaction: TransactionBlock
}
```

Fields

`DryRunResult.error.String` scalar

The error that occurred during dry run execution, if any.

`DryRunResult.results.[DryRunEffect!]` list object

The intermediate results for each command of the dry run execution, including contents of mutated references and return values.

`DryRunResult.transaction.TransactionBlock` object

The transaction block representing the dry run execution.

Returned By

`dryRunTransactionBlock` query

Edit this page

© 2025 SUI FOUNDATION | DOCUMENTATION DISTRIBUTED UNDER CC BY 4.0

DryRunReturn

No description

```
type DryRunReturn {
  type: MoveType!
  bcs: Base64!
}
```

Fields

`DryRunReturn.type`.`MoveType!` **non-null** **object**

`DryRunReturn.bcs`.`Base64!` **non-null** **scalar**

Member Of

`DryRunEffect` **object**

Edit this page

© 2025 SUI FOUNDATION | DOCUMENTATION DISTRIBUTED UNDER CC BY 4.0

dryRunTransactionBlock

Simulate running a transaction to inspect its effects without committing to them on-chain.

`txBytes` either a `TransactionData` struct or a `TransactionKind` struct, BCS-encoded and then Base64-encoded. The expected type is controlled by the presence or absence of `txMeta`: If present, `txBytes` is assumed to be a `TransactionKind`, if absent, then `TransactionData`.

`txMeta` the data that is missing from a `TransactionKind` to make a `TransactionData` (sender address and gas information). All its fields are nullable.

`skipChecks` optional flag to disable the usual verification checks that prevent access to objects that are owned by addresses other than the sender, and calling non-public, non-entry functions, and some other checks. Defaults to false.

```
dryRunTransactionBlock(
  txBytes: String!
  txMeta: TransactionMetadata
  skipChecks: Boolean
): DryRunResult!
```

Arguments

`dryRunTransactionBlock.txBytes.String!` **non-null** **scalar**

`dryRunTransactionBlock.txMeta.TransactionMetadata` **input**

`dryRunTransactionBlock.skipChecks.Boolean` **scalar**

Type

`DryRunResult` **object**

Edit this page

© 2025 SUI FOUNDATION | DOCUMENTATION DISTRIBUTED UNDER CC BY 4.0

DynamicField

Dynamic fields are heterogeneous fields that can be added or removed at runtime, and can have arbitrary user-assigned names. There are two sub-types of dynamic fields:

1. Dynamic Fields can store any value that has the `store` ability, however an object stored in this kind of field will be considered wrapped and will not be accessible directly via its ID by external tools (explorers, wallets, etc) accessing storage.
2. Dynamic Object Fields values must be Sui objects (have the `key` and `store` abilities, and id: UID as the first field), but will still be directly accessible off-chain via their object ID after being attached.

```
type DynamicField {
  name: MoveValue
  value: DynamicFieldValue
}
```

Fields

`DynamicField.name` . `MoveValue` **object**

The string type, data, and serialized value of the DynamicField's 'name' field. This field is used to uniquely identify a child of the parent object.

`DynamicField.value` . `DynamicFieldValue` **union**

The returned dynamic field is an object if its return type is `MoveObject`, in which case it is also accessible off-chain via its address. Its contents will be from the latest version that is at most equal to its parent object's version

Member Of

`Coin` **object** . `CoinMetadata` **object** . `DynamicFieldConnection` **object** . `DynamicFieldEdge` **object** . `IMoveObject` **interface** . `MoveObject` **object** . `Object` **object** . `Owner` **object** . `StakedSui` **object** . `SuinsRegistration` **object**

Edit this page

© 2025 SUI FOUNDATION | DOCUMENTATION DISTRIBUTED UNDER CC BY 4.0

DynamicFieldConnection

No description

```
type DynamicFieldConnection {
  pageInfo: PageInfo!
  edges: [DynamicFieldEdge!]!
  nodes: [DynamicField!]!
}
```

Fields

DynamicFieldConnection.pageInfo.PageInfo! non-null object

Information to aid in pagination.

DynamicFieldConnection.edges.[DynamicFieldEdge!]! non-null object

A list of edges.

DynamicFieldConnection.nodes.[DynamicField!]! non-null object

A list of nodes.

Member Of

Coin object . CoinMetadata object . IMoveObject interface . MoveObject object . Object object . Owner object . StakedSui object . SuinsRegistration object

Edit this page

© 2025 SUI FOUNDATION | DOCUMENTATION DISTRIBUTED UNDER CC BY 4.0

DynamicFieldEdge

An edge in a connection.

```
type DynamicFieldEdge {
  node: DynamicField!
  cursor: String!
}
```

Fields

`DynamicFieldEdge.node`.`DynamicField!` **non-null** **object**

The item at the end of the edge

`DynamicFieldEdge.cursor`.`String!` **non-null** **scalar**

A cursor for use in pagination

Member Of

`DynamicFieldConnection` **object**

Edit this page

© 2025 SUI FOUNDATION | DOCUMENTATION DISTRIBUTED UNDER CC BY 4.0

DynamicFieldName

No description

```
input DynamicFieldName {
  type: String!
  bcs: Base64!
}
```

Fields

`DynamicFieldName.type` . `String!` non-null scalar

The string type of the DynamicField's 'name' field. A string representation of a Move primitive like 'u64', or a struct type like '0x2::kiosk::Listing'

`DynamicFieldName.bcs` . `Base64!` non-null scalar

The Base64 encoded bcs serialization of the DynamicField's 'name' field.

Edit this page

© 2025 SUI FOUNDATION | DOCUMENTATION DISTRIBUTED UNDER CC BY 4.0

DynamicFieldValue

No description

```
union DynamicFieldValue = MoveObject | MoveValue
```

Possible types

DynamicFieldValue.MoveObject object

The representation of an object as a Move Object, which exposes additional information (content, module that governs it, version, is transferrable, etc.) about this object.

DynamicFieldValue.MoveValue object

Member Of

DynamicField object

Edit this page

© 2025 SUI FOUNDATION | DOCUMENTATION DISTRIBUTED UNDER CC BY 4.0

EndOfEpochTransaction

System transaction that supersedes `ChangeEpochTransaction` as the new way to run transactions at the end of an epoch. Behaves similarly to `ChangeEpochTransaction` but can accommodate other optional transactions to run at the end of the epoch.

```
type EndOfEpochTransaction {
  transactions(
    first: Int
    before: String
    last: Int
    after: String
  ): EndOfEpochTransactionKindConnection!
}
```

Fields

`EndOfEpochTransaction.transactions.EndOfEpochTransactionKindConnection!` non-null object

The list of system transactions that are allowed to run at the end of the epoch.

`EndOfEpochTransaction.transactions.first.Int` scalar

`EndOfEpochTransaction.transactions.before.String` scalar

`EndOfEpochTransaction.transactions.last.Int` scalar

`EndOfEpochTransaction.transactions.after.String` scalar

Implemented By

`TransactionBlockKind` union

Edit this page

© 2025 SUI FOUNDATION | DOCUMENTATION DISTRIBUTED UNDER CC BY 4.0

EndOfEpochTransactionKind

No description

```
union EndOfEpochTransactionKind = ChangeEpochTransaction |
AuthenticatorStateCreateTransaction | AuthenticatorStateExpireTransaction |
RandomnessStateCreateTransaction | CoinDenyListStateCreateTransaction |
BridgeStateCreateTransaction | BridgeCommitteeInitTransaction
```

Possible types

EndOfEpochTransactionKind.ChangeEpochTransaction **object**

A system transaction that updates epoch information on-chain (increments the current epoch). Executed by the system once per epoch, without using gas. Epoch change transactions cannot be submitted by users, because validators will refuse to sign them.

This transaction kind is deprecated in favour of `EndOfEpochTransaction`.

EndOfEpochTransactionKind.AuthenticatorStateCreateTransaction **object**

System transaction for creating the on-chain state used by zkLogin.

EndOfEpochTransactionKind.AuthenticatorStateExpireTransaction **object**

EndOfEpochTransactionKind.RandomnessStateCreateTransaction **object**

EndOfEpochTransactionKind.CoinDenyListStateCreateTransaction **object**

EndOfEpochTransactionKind.BridgeStateCreateTransaction **object**

EndOfEpochTransactionKind.BridgeCommitteeInitTransaction **object**

Member Of

`EndOfEpochTransactionKindConnection` **object** . `EndOfEpochTransactionKindEdge` **object**

Edit this page

© 2025 SUI FOUNDATION | DOCUMENTATION DISTRIBUTED UNDER CC BY 4.0

EndOfEpochTransactionKindConnection

No description

```
type EndOfEpochTransactionKindConnection {
  pageInfo: PageInfo!
  edges: [EndOfEpochTransactionKindEdge!]!
  nodes: [EndOfEpochTransactionKind!]!
}
```

Fields

`EndOfEpochTransactionKindConnection.pageInfo.PageInfo!` **non-null** **object**

Information to aid in pagination.

`EndOfEpochTransactionKindConnection.edges.[EndOfEpochTransactionKindEdge!]!` **non-null** **object**

A list of edges.

`EndOfEpochTransactionKindConnection.nodes.[EndOfEpochTransactionKind!]!` **non-null** **union**

A list of nodes.

Member Of

`EndOfEpochTransaction` **object**

Edit this page

© 2025 SUI FOUNDATION | DOCUMENTATION DISTRIBUTED UNDER CC BY 4.0

EndOfEpochTransactionKindEdge

An edge in a connection.

```
type EndOfEpochTransactionKindEdge {
  node: EndOfEpochTransactionKind!
  cursor: String!
}
```

Fields

`EndOfEpochTransactionKindEdge.node.EndOfEpochTransactionKind!` **non-null** **union**

The item at the end of the edge

`EndOfEpochTransactionKindEdge.cursor.String!` **non-null** **scalar**

A cursor for use in pagination

Member Of

`EndOfEpochTransactionKindConnection` **object**

Edit this page

© 2025 SUI FOUNDATION | DOCUMENTATION DISTRIBUTED UNDER CC BY 4.0

Epoch

Operation of the Sui network is temporally partitioned into non-overlapping epochs, and the network aims to keep epochs roughly the same duration as each other. During a particular epoch the following data is fixed:

- the protocol version
- the reference gas price
- the set of participating validators

```
type Epoch {
  epochId: UInt53!
  referenceGasPrice: BigInt
  validatorSet: ValidatorSet
  startTimestamp: DateTime!
  endTimestamp: DateTime
  totalCheckpoints: UInt53
  totalTransactions: UInt53
  totalGasFees: BigInt
  totalStakeRewards: BigInt
  totalStakeSubsidies: BigInt
  fundSize: BigInt
  netInflow: BigInt
  fundInflow: BigInt
  fundOutflow: BigInt
  protocolConfigs: ProtocolConfigs!
  storageFund: StorageFund
  safeMode: SafeMode
  systemStateVersion: UInt53
  systemParameters: SystemParameters
  systemStakeSubsidy: StakeSubsidy
  liveObjectSetDigest: String
  checkpoints(
    first: Int
    after: String
    last: Int
    before: String
  ): CheckpointConnection!
  transactionBlocks(
    first: Int
    after: String
    last: Int
    before: String
    filter: TransactionBlockFilter
    scanLimit: Int
```

```
  ): TransactionBlockConnection!
}
```

Fields

`Epoch.epochId` . `UInt53!` non-null scalar

The epoch's id as a sequence number that starts at 0 and is incremented by one at every epoch change.

`Epoch.referenceGasPrice` . `BigInt` scalar

The minimum gas price that a quorum of validators are guaranteed to sign a transaction for.

`Epoch.validatorSet` . `ValidatorSet` object

Validator related properties, including the active validators.

`Epoch.startTimestamp` . `DateTime!` non-null scalar

The epoch's starting timestamp.

`Epoch.endTimestamp` . `DateTime` scalar

The epoch's ending timestamp.

`Epoch.totalCheckpoints` . `UInt53` scalar

The total number of checkpoints in this epoch.

`Epoch.totalTransactions` . `UInt53` scalar

The total number of transaction blocks in this epoch.

`Epoch.totalGasFees` . `BigInt` scalar

The total amount of gas fees (in MIST) that were paid in this epoch.

`Epoch.totalStakeRewards` . `BigInt` scalar

The total MIST rewarded as stake.

`Epoch.totalStakeSubsidies` . `BigInt` scalar

The amount added to total gas fees to make up the total stake rewards.

`Epoch.fundSize.BigInt` **scalar**

The storage fund available in this epoch. This fund is used to redistribute storage fees from past transactions to future validators.

`Epoch.netInflow.BigInt` **scalar**

The difference between the fund inflow and outflow, representing the net amount of storage fees accumulated in this epoch.

`Epoch.fundInflow.BigInt` **scalar**

The storage fees paid for transactions executed during the epoch.

`Epoch.fundOutflow.BigInt` **scalar**

The storage fee rebates paid to users who deleted the data associated with past transactions.

`Epoch.protocolConfigs.ProtocolConfigs!` **non-null** **object**

The epoch's corresponding protocol configuration, including the feature flags and the configuration options.

`Epoch.storageFund.StorageFund` **object**

SUI set aside to account for objects stored on-chain, at the start of the epoch. This is also used for storage rebates.

`Epoch.safeMode.SafeMode` **object**

Information about whether this epoch was started in safe mode, which happens if the full epoch change logic fails for some reason.

`Epoch.systemStateVersion.UInt53` **scalar**

The value of the `version` field of `0x5`, the `0x3::sui::SuiSystemState` object. This version changes whenever the fields contained in the system state object (held in a dynamic field attached to `0x5`) change.

`Epoch.systemParameters.SystemParameters` **object**

Details of the system that are decided during genesis.

`Epoch.systemStakeSubsidy.StakeSubsidy` **object**

Parameters related to the subsidy that supplements staking rewards

Epoch.liveObjectSetDigest.String scalar

A commitment by the committee at the end of epoch on the contents of the live object set at that time. This can be used to verify state snapshots.

Epoch.checkpoints.CheckpointConnection! non-null object

The epoch's corresponding checkpoints.

Epoch.checkpoints.first.Int scalar

Epoch.checkpoints.after.String scalar

Epoch.checkpoints.last.Int scalar

Epoch.checkpoints.before.String scalar

Epoch.transactionBlocks.TransactionBlockConnection! non-null object

The epoch's corresponding transaction blocks.

`scanLimit` restricts the number of candidate transactions scanned when gathering a page of results. It is required for queries that apply more than two complex filters (on function, kind, sender, recipient, input object, changed object, or ids), and can be at most `serviceConfig.maxScanLimit`.

When the scan limit is reached the page will be returned even if it has fewer than `first` results when paginating forward (`last` when paginating backwards). If there are more transactions to scan, `pageInfo.hasNextPage` (or `pageInfo.hasPreviousPage`) will be set to `true`, and `PageInfo.endCursor` (or `PageInfo.startCursor`) will be set to the last transaction that was scanned as opposed to the last (or first) transaction in the page.

Requesting the next (or previous) page after this cursor will resume the search, scanning the next `scanLimit` many transactions in the direction of pagination, and so on until all transactions in the scanning range have been visited.

By default, the scanning range consists of all transactions in this epoch.

Epoch.transactionBlocks.first.Int scalar

Epoch.transactionBlocks.after.String scalar

Epoch.transactionBlocks.last.Int scalar

Epoch.transactionBlocks.before.String scalar

Epoch.transactionBlocks.filter.TransactionBlockFilter input

Epoch.transactionBlocks.**scanLimit**.Int scalar

Returned By

epoch query

Member Of

ActiveJwk object . AuthenticatorStateExpireTransaction object . AuthenticatorStateUpdateTransaction object . ChangeEpochTransaction object . Checkpoint object . ConsensusCommitPrologueTransaction object . EpochConnection object . EpochEdge object . RandomnessStateUpdateTransaction object . StakedSui object . TransactionBlock object . TransactionBlockEffects object

Edit this page

Sui

© 2025 SUI FOUNDATION | DOCUMENTATION DISTRIBUTED UNDER CC BY 4.0

EpochConnection

No description

```
type EpochConnection {
  pageInfo: PageInfo!
  edges: [EpochEdge!]!
  nodes: [Epoch!]!
}
```

Fields

`EpochConnection.pageInfo`.`PageInfo!` **non-null** **object**

Information to aid in pagination.

`EpochConnection.edges`.`[EpochEdge!]!` **non-null** **object**

A list of edges.

`EpochConnection.nodes`.`[Epoch!]!` **non-null** **object**

A list of nodes.

Returned By

`epochs` **query**

Edit this page

© 2025 SUI FOUNDATION | DOCUMENTATION DISTRIBUTED UNDER CC BY 4.0

EpochEdge

An edge in a connection.

```
type EpochEdge {
  node: Epoch!
  cursor: String!
}
```

Fields

`EpochEdge.node`.`Epoch!` **non-null** **object**

The item at the end of the edge

`EpochEdge.cursor`.`String!` **non-null** **scalar**

A cursor for use in pagination

Member Of

`EpochConnection` **object**

Edit this page

© 2025 SUI FOUNDATION | DOCUMENTATION DISTRIBUTED UNDER CC BY 4.0

epochs

No description

```
epochs(
  first: Int
  after: String
  last: Int
  before: String
): EpochConnection!
```

Arguments

`epochs.first`.`Int` **scalar**

`epochs.after`.`String` **scalar**

`epochs.last`.`Int` **scalar**

`epochs.before`.`String` **scalar**

Type

`EpochConnection` **object**

Edit this page

© 2025 SUI FOUNDATION | DOCUMENTATION DISTRIBUTED UNDER CC BY 4.0

Event

No description

```
type Event {
  transactionBlock: TransactionBlock
  sendingModule: MoveModule
  sender: Address
  timestamp: DateTime
  contents: MoveValue!
  bcs: Base64!
}
```

Fields

`Event.transactionBlock.TransactionBlock` **object**

The transaction block that emitted this event. This information is only available for events from indexed transactions, and not from transactions that have just been executed or dry-run.

`Event.sendingModule.MoveModule` **object**

The Move module containing some function that when called by a programmable transaction block (PTB) emitted this event. For example, if a PTB invokes A::m1::foo, which internally calls A::m2::emit_event to emit an event, the sending module would be A::m1.

`Event.sender.Address` **object**

Address of the sender of the event

`Event.timestamp.DateTime` **scalar**

UTC timestamp in milliseconds since epoch (1/1/1970)

`Event.contents.MoveValue!` **non-null** **object**

The event's contents as a Move value.

`Event.bcs.Base64!` **non-null** **scalar**

The Base64 encoded BCS serialized bytes of the event.

Member Of

`EventConnection` **object** . `EventEdge` **object**

Edit this page

© 2025 SUI FOUNDATION | DOCUMENTATION DISTRIBUTED UNDER CC BY 4.0

EventConnection

No description

```
type EventConnection {
  pageInfo: PageInfo!
  edges: [EventEdge!]!
  nodes: [Event!]!
}
```

Fields

`EventConnection.pageInfo`.`PageInfo!` **non-null** **object**

Information to aid in pagination.

`EventConnection.edges`.`[EventEdge!]!` **non-null** **object**

A list of edges.

`EventConnection.nodes`.`[Event!]!` **non-null** **object**

A list of nodes.

Returned By

`events` **query**

Member Of

`TransactionBlockEffects` **object**

Edit this page

© 2025 SUI FOUNDATION | DOCUMENTATION DISTRIBUTED UNDER CC BY 4.0

EventEdge

An edge in a connection.

```
type EventEdge {
  node: Event!
  cursor: String!
}
```

Fields

`EventEdge.node`.`Event!` **non-null** **object**

The item at the end of the edge

`EventEdge.cursor`.`String!` **non-null** **scalar**

A cursor for use in pagination

Member Of

`EventConnection` **object**

Edit this page

© 2025 SUI FOUNDATION | DOCUMENTATION DISTRIBUTED UNDER CC BY 4.0

EventFilter

No description

```
input EventFilter {
  sender: SuiAddress
  transactionDigest: String
  emittingModule: String
  eventType: String
}
```

Fields

`EventFilter.sender` . `SuiAddress` **scalar**

Filter down to events from transactions sent by this address.

`EventFilter.transactionDigest` . `String` **scalar**

Filter down to the events from this transaction (given by its transaction digest).

`EventFilter.emittingModule` . `String` **scalar**

Events emitted by a particular module. An event is emitted by a particular module if some function in the module is called by a PTB and emits an event.

Modules can be filtered by their package, or package::module. We currently do not support filtering by emitting module and event type at the same time so if both are provided in one filter, the query will error.

`EventFilter.eventType` . `String` **scalar**

This field is used to specify the type of event emitted.

Events can be filtered by their type's package, package::module, or their fully qualified type name.

Generic types can be queried by either the generic type name, e.g. `0x2::coin::Coin`, or by the full type name, such as `0x2::coin::Coin<0x2::sui::SUI>`.

Member Of

`events` **query**

Edit this page

© 2025 SUI FOUNDATION | DOCUMENTATION DISTRIBUTED UNDER CC BY 4.0

events

Query events that are emitted in the network. We currently do not support filtering by emitting module and event type at the same time so if both are provided in one filter, the query will error.

```
events(
  first: Int
  after: String
  last: Int
  before: String
  filter: EventFilter
): EventConnection!
```

Arguments

`events.first`.`Int` **scalar**

`events.after`.`String` **scalar**

`events.last`.`Int` **scalar**

`events.before`.`String` **scalar**

`events.filter`.`EventFilter` **input**

Type

`EventConnection` **object**

Edit this page

© 2025 SUI FOUNDATION | DOCUMENTATION DISTRIBUTED UNDER CC BY 4.0

executeTransactionBlock

Execute a transaction, committing its effects on chain.

- `txBytes` is a `TransactionData` struct that has been BCS-encoded and then Base64-encoded.
- `signatures` are a list of `flag || signature || pubkey` bytes, Base64-encoded.

Waits until the transaction has reached finality on chain to return its transaction digest, or returns the error that prevented finality if that was not possible. A transaction is final when its effects are guaranteed on chain (it cannot be revoked).

There may be a delay between transaction finality and when GraphQL requests (including the request that issued the transaction) reflect its effects. As a result, queries that depend on indexing the state of the chain (e.g. contents of output objects, address-level balance information at the time of the transaction), must wait for indexing to catch up by polling for the transaction digest using `Query.transactionBlock`.

```
executeTransactionBlock(
  txBytes: String!
  signatures: [String!]!
): ExecutionResult!
```

Arguments

`executeTransactionBlock.txBytes.String!` **non-null** **scalar**

`executeTransactionBlock.signatures.[String!]!` **non-null** **scalar**

Type

`ExecutionResult` **object**

The result of an execution, including errors that occurred during said execution.

Edit this page

© 2025 SUI FOUNDATION | DOCUMENTATION DISTRIBUTED UNDER CC BY 4.0

ExecutionResult

The result of an execution, including errors that occurred during said execution.

```
type ExecutionResult {
  errors: [String!]
  effects: TransactionBlockEffects!
}
```

Fields

`ExecutionResult.errors` . `[String!]` **list** **scalar**

The errors field captures any errors that occurred during execution

`ExecutionResult.effects` . `TransactionBlockEffects!` **non-null** **object**

The effects of the executed transaction. Since the transaction was just executed and not indexed yet, fields including `balance_changes`, `timestamp` and `checkpoint` are not available.

Returned By

`executeTransactionBlock` **mutation**

Edit this page

© 2025 SUI FOUNDATION | DOCUMENTATION DISTRIBUTED UNDER CC BY 4.0

ExecutionStatus

The execution status of this transaction block: success or failure.

```
enum ExecutionStatus {
  SUCCESS
  FAILURE
}
```

Values

`ExecutionStatus.SUCCESS`

The transaction block was successfully executed

`ExecutionStatus.FAILURE`

The transaction block could not be executed

Member Of

`TransactionBlockEffects` **object**

Edit this page

© 2025 SUI FOUNDATION | DOCUMENTATION DISTRIBUTED UNDER CC BY 4.0

Feature

Groups of features served by the RPC service. The GraphQL Service can be configured to enable or disable these features.

```
enum Feature {
  ANALYTICS
  COINS
  DYNAMIC_FIELDS
  NAME_SERVICE
  SUBSCRIPTIONS
  SYSTEM_STATE
  MOVE_REGISTRY
}
```

Values

`Feature.ANALYTICS`

Statistics about how the network was running (TPS, top packages, APY, etc)

`Feature.COINS`

Coin metadata, per-address coin and balance information.

`Feature.DYNAMIC_FIELDS`

Querying an object's dynamic fields.

`Feature.NAME_SERVICE`

SuiNS name and reverse name look-up.

`Feature.SUBSCRIPTIONS`

Transaction and Event subscriptions.

`Feature.SYSTEM_STATE`

Aspects that affect the running of the system that are managed by the validators either directly, or through system transactions.

`Feature.MOVE_REGISTRY`

Named packages service (utilizing dotmove package registry).

Member Of

`ServiceConfig` **object**

Edit this page

© 2025 SUI FOUNDATION | DOCUMENTATION DISTRIBUTED UNDER CC BY 4.0

GasCoin

Access to the gas inputs, after they have been smashed into one coin. The gas coin can only be used by reference, except for with `TransferObjectsTransaction` that can accept it by value.

```
type GasCoin {
  _: Boolean
}
```

Fields

`GasCoin._`.`Boolean` **scalar**

A workaround to define an empty variant of a GraphQL union.

Implemented By

`TransactionArgument` **union**

Edit this page

© 2025 SUI FOUNDATION | DOCUMENTATION DISTRIBUTED UNDER CC BY 4.0

GasCostSummary

Breakdown of gas costs in effects.

```
type GasCostSummary {
  computationCost: BigInt
  storageCost: BigInt
  storageRebate: BigInt
  nonRefundableStorageFee: BigInt
}
```

Fields

`GasCostSummary.computationCost`.`BigInt` **scalar**

Gas paid for executing this transaction (in MIST).

`GasCostSummary.storageCost`.`BigInt` **scalar**

Gas paid for the data stored on-chain by this transaction (in MIST).

`GasCostSummary.storageRebate`.`BigInt` **scalar**

Part of storage cost that can be reclaimed by cleaning up data created by this transaction (when objects are deleted or an object is modified, which is treated as a deletion followed by a creation) (in MIST).

`GasCostSummary.nonRefundableStorageFee`.`BigInt` **scalar**

Part of storage cost that is not reclaimed when data created by this transaction is cleaned up (in MIST).

Member Of

`Checkpoint` **object** . `GasEffects` **object** . `SafeMode` **object**

Edit this page

© 2025 SUI FOUNDATION | DOCUMENTATION DISTRIBUTED UNDER CC BY 4.0

GasEffects

Effects related to gas (costs incurred and the identity of the smashed gas object returned).

```
type GasEffects {
  gasObject: Object
  gasSummary: GasCostSummary
}
```

Fields

`GasEffects.gasObject.Object` **object**

`GasEffects.gasSummary.GasCostSummary` **object**

Member Of

`TransactionBlockEffects` **object**

Edit this page

© 2025 SUI FOUNDATION | DOCUMENTATION DISTRIBUTED UNDER CC BY 4.0

GasInput

Configuration for this transaction's gas price and the coins used to pay for gas.

```
type GasInput {
  gasSponsor: Address
  gasPayment(
    first: Int
    after: String
    last: Int
    before: String
  ): ObjectConnection!
  gasPrice: BigInt
  gasBudget: BigInt
}
```

Fields

GasInput.gasSponsor.Address object

Address of the owner of the gas object(s) used

GasInput.gasPayment.ObjectConnection! non-null object

Objects used to pay for a transaction's execution and storage

GasInput.gasPayment.first.Int scalar

GasInput.gasPayment.after.String scalar

GasInput.gasPayment.last.Int scalar

GasInput.gasPayment.before.String scalar

GasInput.gasPrice.BigInt scalar

An unsigned integer specifying the number of native tokens per gas unit this transaction will pay (in MIST).

GasInput.gasBudget.BigInt scalar

The maximum number of gas units that can be expended by executing this transaction

Member Of

`TransactionBlock` **object**

Edit this page

© 2025 SUI FOUNDATION | DOCUMENTATION DISTRIBUTED UNDER CC BY 4.0

GenesisTransaction

System transaction that initializes the network and writes the initial set of objects on-chain.

```
type GenesisTransaction {
  objects(
    first: Int
    after: String
    last: Int
    before: String
  ): ObjectConnection!
}
```

Fields

`GenesisTransaction.objects`.`ObjectConnection!` **non-null** **object**

Objects to be created during genesis.

`GenesisTransaction.objects.first`.`Int` scalar

`GenesisTransaction.objects.after`.`String` scalar

`GenesisTransaction.objects.last`.`Int` scalar

`GenesisTransaction.objects.before`.`String` scalar

Implemented By

`TransactionBlockKind` **union**

Edit this page

© 2025 SUI FOUNDATION | DOCUMENTATION DISTRIBUTED UNDER CC BY 4.0

Immutable

An immutable object is an object that can't be mutated, transferred, or deleted. Immutable objects have no owner, so anyone can use them.

```
type Immutable {
  _: Boolean
}
```

Fields

`Immutable._`.`Boolean` **scalar**

Implemented By

`ObjectOwner` **union**

Edit this page

© 2025 SUI FOUNDATION | DOCUMENTATION DISTRIBUTED UNDER CC BY 4.0

IMoveDatatype

Interface implemented by all GraphQL types that represent a Move datatype (either structs or enums). This interface is used to provide a way to access fields that are shared by both structs and enums, e.g., the module that the datatype belongs to, the name of the datatype, type parameters etc.

```
interface IMoveDatatype {
  module: MoveModule!
  name: String!
  abilities: [MoveAbility!]
  typeParameters: [MoveStructTypeParameter!]
}
```

Fields

`IMoveDatatype.module` . `MoveModule!` non-null object

The module that the datatype belongs to.

`IMoveDatatype.name` . `String!` non-null scalar

The name of the datatype.

`IMoveDatatype.abilities` . `[MoveAbility!]` list enum

The abilities of the datatype.

`IMoveDatatype.typeParameters` . `[MoveStructTypeParameter!]` list object

The type parameters of the datatype.

Implemented By

`MoveDatatype` object . `MoveEnum` object . `MoveStruct` object

Edit this page

© 2025 SUI FOUNDATION | DOCUMENTATION DISTRIBUTED UNDER CC BY 4.0

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

© 2025 SUI FOUNDATION | DOCUMENTATION DISTRIBUTED UNDER CC BY 4.0

include

No description

```
directive @include(
  if: Boolean!
) on
  | FIELD
  | FRAGMENT_SPREAD
  | INLINE_FRAGMENT
```

Arguments

`include.if`.`Boolean!` **non-null** **scalar**

Edit this page

© 2025 SUI FOUNDATION | DOCUMENTATION DISTRIBUTED UNDER CC BY 4.0

Input

One of the input objects or primitive values to the programmable transaction block.

```
type Input {
  ix: Int!
}
```

Fields

`Input.ix`.`Int!` **non-null** **scalar**

Index of the programmable transaction block input (0-indexed).

Implemented By

`TransactionArgument` **union**

Edit this page

© 2025 SUI FOUNDATION | DOCUMENTATION DISTRIBUTED UNDER CC BY 4.0

Int

The `Int` scalar type represents non-fractional signed whole numeric values. Int can represent values between -(2^31) and 2^31 - 1.

```
scalar Int
```

Member Of

checkpoints query · CoinMetadata object · coins query · epochs query · events query · Input object · MoveModule object · objects query · packages query · packageVersions query · Result object · ServiceConfig object · StakeSubsidy object · SystemParameters object · transactionBlocks query · Validator object · ValidatorSet object

Edit this page

© 2025 SUI FOUNDATION | DOCUMENTATION DISTRIBUTED UNDER CC BY 4.0

IObject

Interface implemented by on-chain values that are addressable by an ID (also referred to as its address). This includes Move objects and packages.

```
interface IObject {
  version: UInt53!
  status: ObjectKind!
  digest: String
  owner: ObjectOwner
  previousTransactionBlock: TransactionBlock
  storageRebate: BigInt
  receivedTransactionBlocks(
    first: Int
    after: String
    last: Int
    before: String
    filter: TransactionBlockFilter
    scanLimit: Int
  ): TransactionBlockConnection!
  bcs: Base64
}
```

Fields

IObject.**version**.UInt53! non-null scalar

IObject.**status**.ObjectKind! non-null enum

The current status of the object as read from the off-chain store. The possible states are: NOT_INDEXED, the object is loaded from serialized data, such as the contents of a genesis or system package upgrade transaction. LIVE, the version returned is the most recent for the object, and it is not deleted or wrapped at that version. HISTORICAL, the object was referenced at a specific version or checkpoint, so is fetched from historical tables and may not be the latest version of the object. WRAPPED_OR_DELETED, the object is deleted or wrapped and only partial information can be loaded.

IObject.**digest**.String scalar

32-byte hash that identifies the object's current contents, encoded as a Base58 string.

IObject.**owner**.ObjectOwner union

The owner type of this object: Immutable, Shared, Parent, Address Immutable and Shared Objects do not have owners.

IObject.previousTransactionBlock.TransactionBlock object

The transaction block that created this version of the object.

IObject.storageRebate.BigInt scalar

IObject.receivedTransactionBlocks.TransactionBlockConnection! non-null object

The transaction blocks that sent objects to this object.

IObject.receivedTransactionBlocks.first.Int scalar

IObject.receivedTransactionBlocks.after.String scalar

IObject.receivedTransactionBlocks.last.Int scalar

IObject.receivedTransactionBlocks.before.String scalar

IObject.receivedTransactionBlocks.filter.TransactionBlockFilter input

IObject.receivedTransactionBlocks.scanLimit.Int scalar

IObject.bcs.Base64 scalar

The Base64-encoded BCS serialization of the object's content.

Implemented By

Coin object . CoinMetadata object . MoveObject object . MovePackage object . Object object . StakedSui object . SuinsRegistration object

Edit this page

© 2025 SUI FOUNDATION | DOCUMENTATION DISTRIBUTED UNDER CC BY 4.0

IOwner

Interface implemented by GraphQL types representing entities that can own objects. Object owners are identified by an address which can represent either the public key of an account or another object. The same address can only refer to an account or an object, never both, but it is not possible to know which up-front.

```
interface IOwner {
  address: SuiAddress!
  objects(
    first: Int
    after: String
    last: Int
    before: String
    filter: ObjectFilter
  ): MoveObjectConnection!
  balance(
    type: String
  ): Balance
  balances(
    first: Int
    after: String
    last: Int
    before: String
  ): BalanceConnection!
  coins(
    first: Int
    after: String
    last: Int
    before: String
    type: String
  ): CoinConnection!
  stakedSuis(
    first: Int
    after: String
    last: Int
    before: String
  ): StakedSuiConnection!
  defaultSuinsName(
    format: DomainFormat
  ): String
  suinsRegistrations(
    first: Int
    after: String
    last: Int
    before: String
```

```
  ): SuinsRegistrationConnection!
}
```

Fields

IOwner.address.SuiAddress! non-null scalar

IOwner.objects.MoveObjectConnection! non-null object

Objects owned by this object or address, optionally `filter`-ed.

IOwner.objects.first.Int scalar

IOwner.objects.after.String scalar

IOwner.objects.last.Int scalar

IOwner.objects.before.String scalar

IOwner.objects.filter.ObjectFilter input

IOwner.balance.Balance object

Total balance of all coins with marker type owned by this object or address. If type is not supplied, it defaults to `0x2::sui::SUI`.

IOwner.balance.type.String scalar

IOwner.balances.BalanceConnection! non-null object

The balances of all coin types owned by this object or address.

IOwner.balances.first.Int scalar

IOwner.balances.after.String scalar

IOwner.balances.last.Int scalar

IOwner.balances.before.String scalar

IOwner.coins.CoinConnection! non-null object

The coin objects for this object or address.

`type` is a filter on the coin's type parameter, defaulting to `0x2::sui::SUI`.

IOwner.coins.first.Int scalar

`IOwner.coins.after`.`String` scalar

`IOwner.coins.last`.`Int` scalar

`IOwner.coins.before`.`String` scalar

`IOwner.coins.type`.`String` scalar

`IOwner.stakedSuis`.`StakedSuiConnection!` non-null object

The `0x3::staking_pool::StakedSui` objects owned by this object or address.

`IOwner.stakedSuis.first`.`Int` scalar

`IOwner.stakedSuis.after`.`String` scalar

`IOwner.stakedSuis.last`.`Int` scalar

`IOwner.stakedSuis.before`.`String` scalar

`IOwner.defaultSuinsName`.`String` scalar

The domain explicitly configured as the default domain pointing to this object or address.

`IOwner.defaultSuinsName.format`.`DomainFormat` enum

`IOwner.suinsRegistrations`.`SuinsRegistrationConnection!` non-null object

The SuinsRegistration NFTs owned by this object or address. These grant the owner the capability to manage the associated domain.

`IOwner.suinsRegistrations.first`.`Int` scalar

`IOwner.suinsRegistrations.after`.`String` scalar

`IOwner.suinsRegistrations.last`.`Int` scalar

`IOwner.suinsRegistrations.before`.`String` scalar

Implemented By

`Address` object . `Coin` object . `CoinMetadata` object . `MoveObject` object . `MovePackage` object . `Object` object . `Owner` object . `StakedSui` object . `SuinsRegistration` object

Edit this page

© 2025 SUI FOUNDATION | DOCUMENTATION DISTRIBUTED UNDER CC BY 4.0

JSON

Arbitrary JSON data.

```
scalar JSON
```

Member Of

`MoveValue` **object**

Edit this page

© 2025 SUI FOUNDATION | DOCUMENTATION DISTRIBUTED UNDER CC BY 4.0

latestPackage

The latest version of the package at `address`.

This corresponds to the package with the highest `version` that shares its original ID with the package at `address`.

```
latestPackage(
  address: SuiAddress!
): MovePackage
```

Arguments

`latestPackage.address.SuiAddress!` **non-null** **scalar**

Type

`MovePackage` **object**

A MovePackage is a kind of Move object that represents code that has been published on chain. It exposes information about its modules, type definitions, functions, and dependencies.

Edit this page

© 2025 SUI FOUNDATION | DOCUMENTATION DISTRIBUTED UNDER CC BY 4.0

Linkage

Information used by a package to link to a specific version of its dependency.

```
type Linkage {
  originalId: SuiAddress!
  upgradedId: SuiAddress!
  version: UInt53!
}
```

Fields

`Linkage.originalId.SuiAddress!` non-null scalar

The ID on-chain of the first version of the dependency.

`Linkage.upgradedId.SuiAddress!` non-null scalar

The ID on-chain of the version of the dependency that this package depends on.

`Linkage.version.UInt53!` non-null scalar

The version of the dependency that this package depends on.

Member Of

`MovePackage` object

Edit this page

© 2025 SUI FOUNDATION | DOCUMENTATION DISTRIBUTED UNDER CC BY 4.0

MakeMoveVecTransaction

Create a vector (possibly empty).

```
type MakeMoveVecTransaction {
  type: MoveType
  elements: [TransactionArgument!]!
}
```

Fields

`MakeMoveVecTransaction.type`.`MoveType` **object**

If the elements are not objects, or the vector is empty, a type must be supplied.

`MakeMoveVecTransaction.elements`.`[TransactionArgument!]!` **non-null** **union**

The values to pack into the vector, all of the same type.

Implemented By

`ProgrammableTransaction` **union**

Edit this page

© 2025 SUI FOUNDATION | DOCUMENTATION DISTRIBUTED UNDER CC BY 4.0

MergeCoinsTransaction

Merges `coins` into the first `coin` (produces no results).

```
type MergeCoinsTransaction {
  coin: TransactionArgument!
  coins: [TransactionArgument!]!
}
```

Fields

`MergeCoinsTransaction.coin`.`TransactionArgument!`

non-null **union**

The coin to merge into.

`MergeCoinsTransaction.coins`.`[TransactionArgument!]!`

non-null **union**

The coins to be merged.

Implemented By

`ProgrammableTransaction` **union**

Edit this page

© 2025 SUI FOUNDATION | DOCUMENTATION DISTRIBUTED UNDER CC BY 4.0

MoveAbility

Abilities are keywords in Sui Move that define how types behave at the compiler level.

```
enum MoveAbility {
  COPY
  DROP
  KEY
  STORE
}
```

Values

`MoveAbility.COPY`

Enables values to be copied.

`MoveAbility.DROP`

Enables values to be popped/dropped.

`MoveAbility.KEY`

Enables values to be held directly in global storage.

`MoveAbility.STORE`

Enables values to be held inside a struct in global storage.

Member Of

`IMoveDatatype` interface . `MoveDatatype` object . `MoveEnum` object . `MoveFunctionTypeParameter` object . `MoveStruct` object . `MoveStructTypeParameter` object . `MoveType` object

Edit this page

© 2025 SUI FOUNDATION | DOCUMENTATION DISTRIBUTED UNDER CC BY 4.0

MoveCallTransaction

A call to either an entry or a public Move function.

```
type MoveCallTransaction {
  package: SuiAddress!
  module: String!
  functionName: String!
  function: MoveFunction
  typeArguments: [MoveType!]!
  arguments: [TransactionArgument!]!
}
```

Fields

`MoveCallTransaction.package`.`SuiAddress!` **non-null** **scalar**

The storage ID of the package the function being called is defined in.

`MoveCallTransaction.module`.`String!` **non-null** **scalar**

The name of the module the function being called is defined in.

`MoveCallTransaction.functionName`.`String!` **non-null** **scalar**

The name of the function being called.

`MoveCallTransaction.function`.`MoveFunction` **object**

The function being called, resolved.

`MoveCallTransaction.typeArguments`.`[MoveType!]!` **non-null** **object**

The actual type parameters passed in for this move call.

`MoveCallTransaction.arguments`.`[TransactionArgument!]!` **non-null** **union**

The actual function parameters passed in for this move call.

Implemented By

`ProgrammableTransaction` **union**

Edit this page

© 2025 SUI FOUNDATION | DOCUMENTATION DISTRIBUTED UNDER CC BY 4.0

MoveData

The contents of a Move Value, corresponding to the following recursive type:

type MoveData = { Address: SuiAddress } | { UID: SuiAddress } | { ID: SuiAddress } | { Bool: bool } | { Number: BigInt } | { String: string } | { Vector: [MoveData] } | { Option: MoveData? } | { Struct: [{ name: string , value: MoveData }] } | { Variant: { name: string, fields: [{ name: string, value: MoveData }], }

```
scalar MoveData
```

Member Of

`MoveValue` **object**

Edit this page

© 2025 SUI FOUNDATION | DOCUMENTATION DISTRIBUTED UNDER CC BY 4.0

MoveDatatype

The generic representation of a Move datatype (either a struct or an enum) which exposes common fields and information (module, name, abilities, type parameters etc.) that is shared across them.

```
type MoveDatatype implements IMoveDatatype {
  module: MoveModule!
  name: String!
  abilities: [MoveAbility!]
  typeParameters: [MoveStructTypeParameter!]
  asMoveEnum: MoveEnum
  asMoveStruct: MoveStruct
}
```

Fields

`MoveDatatype.module` . `MoveModule!` **non-null** **object**

`MoveDatatype.name` . `String!` **non-null** **scalar**

`MoveDatatype.abilities` . `[MoveAbility!]` **list** **enum**

`MoveDatatype.typeParameters` . `[MoveStructTypeParameter!]` **list** **object**

`MoveDatatype.asMoveEnum` . `MoveEnum` **object**

`MoveDatatype.asMoveStruct` . `MoveStruct` **object**

Interfaces

`IMoveDatatype` **interface**

Interface implemented by all GraphQL types that represent a Move datatype (either structs or enums). This interface is used to provide a way to access fields that are shared by both structs and enums, e.g., the module that the datatype belongs to, the name of the datatype, type parameters etc.

Member Of

MoveDatatypeConnection object . MoveDatatypeEdge object . MoveModule object

Edit this page

© 2025 SUI FOUNDATION | DOCUMENTATION DISTRIBUTED UNDER CC BY 4.0

MoveDatatypeConnection

No description

```
type MoveDatatypeConnection {
  pageInfo: PageInfo!
  edges: [MoveDatatypeEdge!]!
  nodes: [MoveDatatype!]!
}
```

Fields

`MoveDatatypeConnection.pageInfo.PageInfo!` non-null object

Information to aid in pagination.

`MoveDatatypeConnection.edges.[MoveDatatypeEdge!]!` non-null object

A list of edges.

`MoveDatatypeConnection.nodes.[MoveDatatype!]!` non-null object

A list of nodes.

Member Of

`MoveModule` object

Edit this page

© 2025 SUI FOUNDATION | DOCUMENTATION DISTRIBUTED UNDER CC BY 4.0

MoveDatatypeEdge

An edge in a connection.

```
type MoveDatatypeEdge {
  node: MoveDatatype!
  cursor: String!
}
```

Fields

`MoveDatatypeEdge.node`.`MoveDatatype!` **non-null** **object**

The item at the end of the edge

`MoveDatatypeEdge.cursor`.`String!` **non-null** **scalar**

A cursor for use in pagination

Member Of

`MoveDatatypeConnection` **object**

Edit this page

© 2025 SUI FOUNDATION | DOCUMENTATION DISTRIBUTED UNDER CC BY 4.0

MoveEnum

Description of an enum type, defined in a Move module.

```
type MoveEnum implements IMoveDatatype {
  module: MoveModule!
  name: String!
  abilities: [MoveAbility!]
  typeParameters: [MoveStructTypeParameter!]
  variants: [MoveEnumVariant!]
}
```

Fields

`MoveEnum.module` . `MoveModule!` non-null object

The module this enum was originally defined in.

`MoveEnum.name` . `String!` non-null scalar

The enum's (unqualified) type name.

`MoveEnum.abilities` . `[MoveAbility!]` list enum

The enum's abilities.

`MoveEnum.typeParameters` . `[MoveStructTypeParameter!]` list object

Constraints on the enum's formal type parameters. Move bytecode does not name type parameters, so when they are referenced (e.g. in field types) they are identified by their index in this list.

`MoveEnum.variants` . `[MoveEnumVariant!]` list object

The names and types of the enum's fields. Field types reference type parameters, by their index in the defining enum's `typeParameters` list.

Interfaces

`IMoveDatatype` interface

Interface implemented by all GraphQL types that represent a Move datatype (either structs or enums). This interface is used to provide a way to access fields that are shared by both structs and enums, e.g., the module that the datatype belongs to, the name of the datatype, type parameters etc.

Member Of

`MoveDatatype` **object** . `MoveEnumConnection` **object** . `MoveEnumEdge` **object** . `MoveModule` **object**

Edit this page

© 2025 SUI FOUNDATION | DOCUMENTATION DISTRIBUTED UNDER CC BY 4.0

MoveEnumConnection

No description

```
type MoveEnumConnection {
  pageInfo: PageInfo!
  edges: [MoveEnumEdge!]!
  nodes: [MoveEnum!]!
}
```

Fields

`MoveEnumConnection.pageInfo.PageInfo!` **non-null** **object**

Information to aid in pagination.

`MoveEnumConnection.edges.[MoveEnumEdge!]!` **non-null** **object**

A list of edges.

`MoveEnumConnection.nodes.[MoveEnum!]!` **non-null** **object**

A list of nodes.

Member Of

`MoveModule` **object**

Edit this page

© 2025 SUI FOUNDATION | DOCUMENTATION DISTRIBUTED UNDER CC BY 4.0

MoveEnumEdge

An edge in a connection.

```
type MoveEnumEdge {
  node: MoveEnum!
  cursor: String!
}
```

Fields

`MoveEnumEdge.node`.`MoveEnum!` **non-null** **object**

The item at the end of the edge

`MoveEnumEdge.cursor`.`String!` **non-null** **scalar**

A cursor for use in pagination

Member Of

`MoveEnumConnection` **object**

Edit this page

© 2025 SUI FOUNDATION | DOCUMENTATION DISTRIBUTED UNDER CC BY 4.0

MoveEnumVariant

No description

```
type MoveEnumVariant {
  name: String!
  fields: [MoveField!]
}
```

Fields

`MoveEnumVariant.name`.`String!` non-null scalar

The name of the variant

`MoveEnumVariant.fields`.`[MoveField!]` list object

The names and types of the variant's fields. Field types reference type parameters, by their index in the defining enum's `typeParameters` list.

Member Of

`MoveEnum` object

Edit this page

© 2025 SUI FOUNDATION | DOCUMENTATION DISTRIBUTED UNDER CC BY 4.0

MoveField

Information for a particular field on a Move struct.

```
type MoveField {
  name: String!
  type: OpenMoveType
}
```

Fields

`MoveField.name`.`String!` **non-null** **scalar**

`MoveField.type`.`OpenMoveType` **object**

Member Of

`MoveEnumVariant` **object** . `MoveStruct` **object**

Edit this page

© 2025 SUI FOUNDATION | DOCUMENTATION DISTRIBUTED UNDER CC BY 4.0

MoveFunction

Signature of a function, defined in a Move module.

```
type MoveFunction {
  module: MoveModule!
  name: String!
  visibility: MoveVisibility
  isEntry: Boolean
  typeParameters: [MoveFunctionTypeParameter!]
  parameters: [OpenMoveType!]
  return: [OpenMoveType!]
}
```

Fields

`MoveFunction.module` . `MoveModule!` non-null object

The module this function was defined in.

`MoveFunction.name` . `String!` non-null scalar

The function's (unqualified) name.

`MoveFunction.visibility` . `MoveVisibility` enum

The function's visibility: `public`, `public(friend)`, or `private`.

`MoveFunction.isEntry` . `Boolean` scalar

Whether the function has the `entry` modifier or not.

`MoveFunction.typeParameters` . `[MoveFunctionTypeParameter!]` list object

Constraints on the function's formal type parameters. Move bytecode does not name type parameters, so when they are referenced (e.g. in parameter and return types) they are identified by their index in this list.

`MoveFunction.parameters` . `[OpenMoveType!]` list object

The function's parameter types. These types can reference type parameters introduce by this function (see `typeParameters`).

`MoveFunction.return`.`[OpenMoveType!]` **list** **object**

The function's return types. There can be multiple because functions in Move can return multiple values. These types can reference type parameters introduced by this function (see `typeParameters`).

Member Of

`MoveCallTransaction` **object** . `MoveFunctionConnection` **object** . `MoveFunctionEdge` **object** . `MoveModule` **object**

Edit this page

© 2025 SUI FOUNDATION | DOCUMENTATION DISTRIBUTED UNDER CC BY 4.0

MoveFunctionConnection

No description

```
type MoveFunctionConnection {
  pageInfo: PageInfo!
  edges: [MoveFunctionEdge!]!
  nodes: [MoveFunction!]!
}
```

Fields

`MoveFunctionConnection.pageInfo.PageInfo!` non-null object

Information to aid in pagination.

`MoveFunctionConnection.edges.[MoveFunctionEdge!]!` non-null object

A list of edges.

`MoveFunctionConnection.nodes.[MoveFunction!]!` non-null object

A list of nodes.

Member Of

`MoveModule` object

Edit this page

© 2025 SUI FOUNDATION | DOCUMENTATION DISTRIBUTED UNDER CC BY 4.0

MoveFunctionEdge

An edge in a connection.

```
type MoveFunctionEdge {
  node: MoveFunction!
  cursor: String!
}
```

Fields

`MoveFunctionEdge.node`.`MoveFunction!` **non-null** **object**

The item at the end of the edge

`MoveFunctionEdge.cursor`.`String!` **non-null** **scalar**

A cursor for use in pagination

Member Of

`MoveFunctionConnection` **object**

Edit this page

© 2025 SUI FOUNDATION | DOCUMENTATION DISTRIBUTED UNDER CC BY 4.0

MoveFunctionTypeParameter

No description

```
type MoveFunctionTypeParameter {
  constraints: [MoveAbility!]!
}
```

Fields

`MoveFunctionTypeParameter.constraints.[MoveAbility!]!`

non-null **enum**

Member Of

`MoveFunction` **object**

Edit this page

© 2025 SUI FOUNDATION | DOCUMENTATION DISTRIBUTED UNDER CC BY 4.0

MoveModule

Represents a module in Move, a library that defines struct types and functions that operate on these types.

```
type MoveModule {
  package: MovePackage!
  name: String!
  fileFormatVersion: Int!
  friends(
    first: Int
    after: String
    last: Int
    before: String
  ): MoveModuleConnection!
  struct(
    name: String!
  ): MoveStruct
  structs(
    first: Int
    after: String
    last: Int
    before: String
  ): MoveStructConnection
  enum(
    name: String!
  ): MoveEnum
  enums(
    first: Int
    after: String
    last: Int
    before: String
  ): MoveEnumConnection
  datatype(
    name: String!
  ): MoveDatatype
  datatypes(
    first: Int
    after: String
    last: Int
    before: String
  ): MoveDatatypeConnection
  function(
    name: String!
  ): MoveFunction
  functions(
    first: Int
    after: String
```

```
    last: Int
    before: String
  ): MoveFunctionConnection
  bytes: Base64
  disassembly: String
}
```

Fields

`MoveModule.package`.`MovePackage!` non-null object

The package that this Move module was defined in

`MoveModule.name`.`String!` non-null scalar

The module's (unqualified) name.

`MoveModule.fileFormatVersion`.`Int!` non-null scalar

Format version of this module's bytecode.

`MoveModule.friends`.`MoveModuleConnection!` non-null object

Modules that this module considers friends (these modules can access `public(friend)` functions from this module).

`MoveModule.friends.first`.`Int` scalar

`MoveModule.friends.after`.`String` scalar

`MoveModule.friends.last`.`Int` scalar

`MoveModule.friends.before`.`String` scalar

`MoveModule.struct`.`MoveStruct` object

Look-up the definition of a struct defined in this module, by its name.

`MoveModule.struct.name`.`String!` non-null scalar

`MoveModule.structs`.`MoveStructConnection` object

Iterate through the structs defined in this module.

`MoveModule.structs.first`.`Int` scalar

`MoveModule.structs.after`.`String` scalar

`MoveModule.structs.last`.`Int` scalar

`MoveModule.structs.before`.`String` scalar

`MoveModule.enum`.`MoveEnum` object

Look-up the definition of a enum defined in this module, by its name.

`MoveModule.enum.name`.`String!` non-null scalar

`MoveModule.enums`.`MoveEnumConnection` object

Iterate through the enums defined in this module.

`MoveModule.enums.first`.`Int` scalar

`MoveModule.enums.after`.`String` scalar

`MoveModule.enums.last`.`Int` scalar

`MoveModule.enums.before`.`String` scalar

`MoveModule.datatype`.`MoveDatatype` object

Look-up the definition of a datatype (struct or enum) defined in this module, by its name.

`MoveModule.datatype.name`.`String!` non-null scalar

`MoveModule.datatypes`.`MoveDatatypeConnection` object

Iterate through the datatypes (enmums and structs) defined in this module.

`MoveModule.datatypes.first`.`Int` scalar

`MoveModule.datatypes.after`.`String` scalar

`MoveModule.datatypes.last`.`Int` scalar

`MoveModule.datatypes.before`.`String` scalar

`MoveModule.function`.`MoveFunction` object

Look-up the signature of a function defined in this module, by its name.

`MoveModule.function.name`.`String!` non-null scalar

`MoveModule.functions`.`MoveFunctionConnection` object

Iterate through the signatures of functions defined in this module.

MoveModule.functions.first.Int scalar

MoveModule.functions.after.String scalar

MoveModule.functions.last.Int scalar

MoveModule.functions.before.String scalar

MoveModule.bytes.Base64 scalar

The Base64 encoded bcs serialization of the module.

MoveModule.disassembly.String scalar

Textual representation of the module's bytecode.

Member Of

Event object . IMoveDatatype interface . MoveDatatype object . MoveEnum object . MoveFunction object . MoveModuleConnection object . MoveModuleEdge object . MovePackage object . MoveStruct object

Edit this page

© 2025 SUI FOUNDATION | DOCUMENTATION DISTRIBUTED UNDER CC BY 4.0

MoveModuleConnection

No description

```
type MoveModuleConnection {
  pageInfo: PageInfo!
  edges: [MoveModuleEdge!]!
  nodes: [MoveModule!]!
}
```

Fields

`MoveModuleConnection.pageInfo` . `PageInfo!` **non-null** **object**

Information to aid in pagination.

`MoveModuleConnection.edges` . `[MoveModuleEdge!]!` **non-null** **object**

A list of edges.

`MoveModuleConnection.nodes` . `[MoveModule!]!` **non-null** **object**

A list of nodes.

Member Of

`MoveModule` **object** . `MovePackage` **object**

Edit this page

© 2025 SUI FOUNDATION | DOCUMENTATION DISTRIBUTED UNDER CC BY 4.0

MoveModuleEdge

An edge in a connection.

```
type MoveModuleEdge {
  node: MoveModule!
  cursor: String!
}
```

Fields

`MoveModuleEdge.node` . `MoveModule!` **non-null** **object**

The item at the end of the edge

`MoveModuleEdge.cursor` . `String!` **non-null** **scalar**

A cursor for use in pagination

Member Of

`MoveModuleConnection` **object**

Edit this page

Sui

© 2025 SUI FOUNDATION | DOCUMENTATION DISTRIBUTED UNDER CC BY 4.0

MoveObject

The representation of an object as a Move Object, which exposes additional information (content, module that governs it, version, is transferrable, etc.) about this object.

```
type MoveObject implements IMoveObject, IObject, IOwner {
  address: SuiAddress!
  objects(
    first: Int
    after: String
    last: Int
    before: String
    filter: ObjectFilter
  ): MoveObjectConnection!
  balance(
    type: String
  ): Balance
  balances(
    first: Int
    after: String
    last: Int
    before: String
  ): BalanceConnection!
  coins(
    first: Int
    after: String
    last: Int
    before: String
    type: String
  ): CoinConnection!
  stakedSuis(
    first: Int
    after: String
    last: Int
    before: String
  ): StakedSuiConnection!
  defaultSuinsName(
    format: DomainFormat
  ): String
  suinsRegistrations(
    first: Int
    after: String
    last: Int
    before: String
  ): SuinsRegistrationConnection!
  version: UInt53!
  status: ObjectKind!
  digest: String
```

```
  owner: ObjectOwner
  previousTransactionBlock: TransactionBlock
  storageRebate: BigInt
  receivedTransactionBlocks(
    first: Int
    after: String
    last: Int
    before: String
    filter: TransactionBlockFilter
    scanLimit: Int
  ): TransactionBlockConnection!
  bcs: Base64
  contents: MoveValue
  hasPublicTransfer: Boolean!
  display: [DisplayEntry!]
  dynamicField(
    name: DynamicFieldName!
  ): DynamicField
  dynamicObjectField(
    name: DynamicFieldName!
  ): DynamicField
  dynamicFields(
    first: Int
    after: String
    last: Int
    before: String
  ): DynamicFieldConnection!
  asCoin: Coin
  asStakedSui: StakedSui
  asCoinMetadata: CoinMetadata
  asSuinsRegistration: SuinsRegistration
}
```

Fields

`MoveObject.address`.`SuiAddress!` **non-null** **scalar**

`MoveObject.objects`.`MoveObjectConnection!` **non-null** **object**

Objects owned by this object, optionally `filter`-ed.

`MoveObject.objects.first`.`Int` **scalar**

`MoveObject.objects.after`.`String` **scalar**

`MoveObject.objects.last`.`Int` **scalar**

`MoveObject.objects.before`.`String` **scalar**

`MoveObject.objects.filter`.`ObjectFilter` input

`MoveObject.balance`.`Balance` object

Total balance of all coins with marker type owned by this object. If type is not supplied, it defaults to `0x2::sui::SUI`.

`MoveObject.balance.type`.`String` scalar

`MoveObject.balances`.`BalanceConnection!` non-null object

The balances of all coin types owned by this object.

`MoveObject.balances.first`.`Int` scalar

`MoveObject.balances.after`.`String` scalar

`MoveObject.balances.last`.`Int` scalar

`MoveObject.balances.before`.`String` scalar

`MoveObject.coins`.`CoinConnection!` non-null object

The coin objects for this object.

`type` is a filter on the coin's type parameter, defaulting to `0x2::sui::SUI`.

`MoveObject.coins.first`.`Int` scalar

`MoveObject.coins.after`.`String` scalar

`MoveObject.coins.last`.`Int` scalar

`MoveObject.coins.before`.`String` scalar

`MoveObject.coins.type`.`String` scalar

`MoveObject.stakedSuis`.`StakedSuiConnection!` non-null object

The `0x3::staking_pool::StakedSui` objects owned by this object.

`MoveObject.stakedSuis.first`.`Int` scalar

`MoveObject.stakedSuis.after`.`String` scalar

`MoveObject.stakedSuis.last`.`Int` scalar

`MoveObject.stakedSuis.before`.`String` scalar

`MoveObject.defaultSuinsName`.`String` scalar

The domain explicitly configured as the default domain pointing to this object.

`MoveObject.defaultSuinsName.format`.`DomainFormat` enum

`MoveObject.suinsRegistrations`.`SuinsRegistrationConnection!` non-null object

The SuinsRegistration NFTs owned by this object. These grant the owner the capability to manage the associated domain.

`MoveObject.suinsRegistrations.first`.`Int` scalar

`MoveObject.suinsRegistrations.after`.`String` scalar

`MoveObject.suinsRegistrations.last`.`Int` scalar

`MoveObject.suinsRegistrations.before`.`String` scalar

`MoveObject.version`.`UInt53!` non-null scalar

`MoveObject.status`.`ObjectKind!` non-null enum

The current status of the object as read from the off-chain store. The possible states are: NOT_INDEXED, the object is loaded from serialized data, such as the contents of a genesis or system package upgrade transaction. LIVE, the version returned is the most recent for the object, and it is not deleted or wrapped at that version. HISTORICAL, the object was referenced at a specific version or checkpoint, so is fetched from historical tables and may not be the latest version of the object. WRAPPED_OR_DELETED, the object is deleted or wrapped and only partial information can be loaded."

`MoveObject.digest`.`String` scalar

32-byte hash that identifies the object's contents, encoded as a Base58 string.

`MoveObject.owner`.`ObjectOwner` union

The owner type of this object: Immutable, Shared, Parent, Address

`MoveObject.previousTransactionBlock`.`TransactionBlock` object

The transaction block that created this version of the object.

MoveObject.storageRebate.BigInt scalar

The amount of SUI we would rebate if this object gets deleted or mutated. This number is recalculated based on the present storage gas price.

MoveObject.receivedTransactionBlocks.TransactionBlockConnection! non-null object

The transaction blocks that sent objects to this object.

`scanLimit` restricts the number of candidate transactions scanned when gathering a page of results. It is required for queries that apply more than two complex filters (on function, kind, sender, recipient, input object, changed object, or ids), and can be at most `serviceConfig.maxScanLimit`.

When the scan limit is reached the page will be returned even if it has fewer than `first` results when paginating forward (`last` when paginating backwards). If there are more transactions to scan, `pageInfo.hasNextPage` (or `pageInfo.hasPreviousPage`) will be set to `true`, and `PageInfo.endCursor` (or `PageInfo.startCursor`) will be set to the last transaction that was scanned as opposed to the last (or first) transaction in the page.

Requesting the next (or previous) page after this cursor will resume the search, scanning the next `scanLimit` many transactions in the direction of pagination, and so on until all transactions in the scanning range have been visited.

By default, the scanning range includes all transactions known to GraphQL, but it can be restricted by the `after` and `before` cursors, and the `beforeCheckpoint`, `afterCheckpoint` and `atCheckpoint` filters.

`MoveObject.receivedTransactionBlocks.first`.`Int` scalar

`MoveObject.receivedTransactionBlocks.after`.`String` scalar

`MoveObject.receivedTransactionBlocks.last`.`Int` scalar

`MoveObject.receivedTransactionBlocks.before`.`String` scalar

`MoveObject.receivedTransactionBlocks.filter`.`TransactionBlockFilter` input

`MoveObject.receivedTransactionBlocks.scanLimit`.`Int` scalar

MoveObject.bcs.Base64 scalar

The Base64-encoded BCS serialization of the object's content.

MoveObject.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.

MoveObject.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.

MoveObject.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.

MoveObject.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 Owner type.

`MoveObject.dynamicField.name`.`DynamicFieldName!` non-null input

MoveObject.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.

`MoveObject.dynamicObjectField.name`.`DynamicFieldName!` non-null input

MoveObject.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.

`MoveObject.dynamicFields.first`.`Int` scalar

`MoveObject.dynamicFields.after`.`String` scalar

`MoveObject.dynamicFields.last`.`Int` scalar

`MoveObject.dynamicFields.before`.`String` scalar

MoveObject.asCoin.Coin object

Attempts to convert the Move object into a `0x2::coin::Coin`.

MoveObject.asStakedSui.StakedSui object

Attempts to convert the Move object into a `0x3::staking_pool::StakedSui`.

MoveObject.asCoinMetadata.CoinMetadata object

Attempts to convert the Move object into a `0x2::coin::CoinMetadata`.

MoveObject.asSuinsRegistration.SuinsRegistration object

Attempts to convert the Move object into a `SuinsRegistration` object.

Interfaces

IMoveObject interface

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

IObject interface

Interface implemented by on-chain values that are addressable by an ID (also referred to as its address). This includes Move objects and packages.

IOwner interface

Interface implemented by GraphQL types representing entities that can own objects. Object owners are identified by an address which can represent either the public key of an account or another object. The same address can only refer to an account or an object, never both, but it is not possible to know which up-front.

Member Of

MoveObjectConnection object . MoveObjectEdge object . Object object . Validator object

Implemented By

DynamicFieldValue union

Edit this page

© 2025 SUI FOUNDATION | DOCUMENTATION DISTRIBUTED UNDER CC BY 4.0

MoveObjectConnection

No description

```
type MoveObjectConnection {
  pageInfo: PageInfo!
  edges: [MoveObjectEdge!]!
  nodes: [MoveObject!]!
}
```

Fields

`MoveObjectConnection.pageInfo.PageInfo!` **non-null** **object**

Information to aid in pagination.

`MoveObjectConnection.edges.[MoveObjectEdge!]!` **non-null** **object**

A list of edges.

`MoveObjectConnection.nodes.[MoveObject!]!` **non-null** **object**

A list of nodes.

Member Of

`Address` **object** . `Coin` **object** . `CoinMetadata` **object** . `IOwner` **interface** . `MoveObject` **object** . `MovePackage` **object** . `Object` **object** . `Owner` **object** . `StakedSui` **object** . `SuinsRegistration` **object**

Edit this page

© 2025 SUI FOUNDATION | DOCUMENTATION DISTRIBUTED UNDER CC BY 4.0

MoveObjectEdge

An edge in a connection.

```
type MoveObjectEdge {
  node: MoveObject!
  cursor: String!
}
```

Fields

`MoveObjectEdge.node` . `MoveObject!` **non-null** **object**

The item at the end of the edge

`MoveObjectEdge.cursor` . `String!` **non-null** **scalar**

A cursor for use in pagination

Member Of

`MoveObjectConnection` **object**

Edit this page

© 2025 SUI FOUNDATION | DOCUMENTATION DISTRIBUTED UNDER CC BY 4.0

MovePackage

A MovePackage is a kind of Move object that represents code that has been published on chain. It exposes information about its modules, type definitions, functions, and dependencies.

```
type MovePackage implements IObject, IOwner {
  address: SuiAddress!
  objects(
    first: Int
    after: String
    last: Int
    before: String
    filter: ObjectFilter
  ): MoveObjectConnection!
  balance(
    type: String
  ): Balance
  balances(
    first: Int
    after: String
    last: Int
    before: String
  ): BalanceConnection!
  coins(
    first: Int
    after: String
    last: Int
    before: String
    type: String
  ): CoinConnection!
  stakedSuis(
    first: Int
    after: String
    last: Int
    before: String
  ): StakedSuiConnection!
  defaultSuinsName(
    format: DomainFormat
  ): String
  suinsRegistrations(
    first: Int
    after: String
    last: Int
    before: String
  ): SuinsRegistrationConnection!
  version: UInt53!
  status: ObjectKind!
  digest: String
```

```
  owner: ObjectOwner
  previousTransactionBlock: TransactionBlock
  storageRebate: BigInt
  receivedTransactionBlocks(
    first: Int
    after: String
    last: Int
    before: String
    filter: TransactionBlockFilter
    scanLimit: Int
  ): TransactionBlockConnection!
  bcs: Base64
  packageAtVersion(
    version: Int!
  ): MovePackage
  packageVersions(
    first: Int
    after: String
    last: Int
    before: String
    filter: MovePackageVersionFilter
  ): MovePackageConnection!
  latestPackage: MovePackage!
  module(
    name: String!
  ): MoveModule
  modules(
    first: Int
    after: String
    last: Int
    before: String
  ): MoveModuleConnection
  linkage: [Linkage!]
  typeOrigins: [TypeOrigin!]
  packageBcs: Base64
  moduleBcs: Base64
}
```

Fields

`MovePackage.address` . `SuiAddress!` non-null scalar

`MovePackage.objects` . `MoveObjectConnection!` non-null object

Objects owned by this package, optionally `filter`-ed.

Note that objects owned by a package are inaccessible, because packages are immutable and cannot be owned by an address.

`MovePackage.objects.first` . `Int` scalar

`MovePackage.objects.after`.`String` scalar

`MovePackage.objects.last`.`Int` scalar

`MovePackage.objects.before`.`String` scalar

`MovePackage.objects.filter`.`ObjectFilter` input

`MovePackage.balance`.`Balance` object

Total balance of all coins with marker type owned by this package. If type is not supplied, it defaults to `0x2::sui::SUI`.

Note that coins owned by a package are inaccessible, because packages are immutable and cannot be owned by an address.

`MovePackage.balance.type`.`String` scalar

`MovePackage.balances`.`BalanceConnection!` non-null object

The balances of all coin types owned by this package.

Note that coins owned by a package are inaccessible, because packages are immutable and cannot be owned by an address.

`MovePackage.balances.first`.`Int` scalar

`MovePackage.balances.after`.`String` scalar

`MovePackage.balances.last`.`Int` scalar

`MovePackage.balances.before`.`String` scalar

`MovePackage.coins`.`CoinConnection!` non-null object

The coin objects owned by this package.

`type` is a filter on the coin's type parameter, defaulting to `0x2::sui::SUI`.

Note that coins owned by a package are inaccessible, because packages are immutable and cannot be owned by an address.

`MovePackage.coins.first`.`Int` scalar

`MovePackage.coins.after`.`String` scalar

`MovePackage.coins.last`.`Int` scalar

`MovePackage.coins.before`.`String` scalar

`MovePackage.coins.type`.`String` scalar

`MovePackage.stakedSuis`.`StakedSuiConnection!` non-null object

The `0x3::staking_pool::StakedSui` objects owned by this package.

Note that objects owned by a package are inaccessible, because packages are immutable and cannot be owned by an address.

`MovePackage.stakedSuis.first`.`Int` scalar

`MovePackage.stakedSuis.after`.`String` scalar

`MovePackage.stakedSuis.last`.`Int` scalar

`MovePackage.stakedSuis.before`.`String` scalar

`MovePackage.defaultSuinsName`.`String` scalar

The domain explicitly configured as the default domain pointing to this object.

`MovePackage.defaultSuinsName.format`.`DomainFormat` enum

`MovePackage.suinsRegistrations`.`SuinsRegistrationConnection!` non-null object

The SuinsRegistration NFTs owned by this package. These grant the owner the capability to manage the associated domain.

Note that objects owned by a package are inaccessible, because packages are immutable and cannot be owned by an address.

`MovePackage.suinsRegistrations.first`.`Int` scalar

`MovePackage.suinsRegistrations.after`.`String` scalar

`MovePackage.suinsRegistrations.last`.`Int` scalar

`MovePackage.suinsRegistrations.before`.`String` scalar

`MovePackage.version`.`UInt53!` non-null scalar

`MovePackage.status`.`ObjectKind!` non-null enum

The current status of the object as read from the off-chain store. The possible states are: NOT_INDEXED, the object is loaded from serialized data, such as the contents of a genesis or system package upgrade transaction. LIVE, the version returned is the most recent for the object, and it is not deleted or wrapped at that version. HISTORICAL, the object was referenced at a specific

version or checkpoint, so is fetched from historical tables and may not be the latest version of the object. WRAPPED_OR_DELETED, the object is deleted or wrapped and only partial information can be loaded."

`MovePackage.digest`.`String` **scalar**

32-byte hash that identifies the package's contents, encoded as a Base58 string.

`MovePackage.owner`.`ObjectOwner` **union**

The owner type of this object: Immutable, Shared, Parent, Address Packages are always Immutable.

`MovePackage.previousTransactionBlock`.`TransactionBlock` **object**

The transaction block that published or upgraded this package.

`MovePackage.storageRebate`.`BigInt` **scalar**

The amount of SUI we would rebate if this object gets deleted or mutated. This number is recalculated based on the present storage gas price.

Note that packages cannot be deleted or mutated, so this number is provided purely for reference.

`MovePackage.receivedTransactionBlocks`.`TransactionBlockConnection!` **non-null** **object**

The transaction blocks that sent objects to this package.

Note that objects that have been sent to a package become inaccessible.

`scanLimit` restricts the number of candidate transactions scanned when gathering a page of results. It is required for queries that apply more than two complex filters (on function, kind, sender, recipient, input object, changed object, or ids), and can be at most `serviceConfig.maxScanLimit`.

When the scan limit is reached the page will be returned even if it has fewer than `first` results when paginating forward (`last` when paginating backwards). If there are more transactions to scan, `pageInfo.hasNextPage` (or `pageInfo.hasPreviousPage`) will be set to `true`, and `PageInfo.endCursor` (or `PageInfo.startCursor`) will be set to the last transaction that was scanned as opposed to the last (or first) transaction in the page.

Requesting the next (or previous) page after this cursor will resume the search, scanning the next `scanLimit` many transactions in the direction of pagination, and so on until all transactions in the scanning range have been visited.

By default, the scanning range includes all transactions known to GraphQL, but it can be restricted by the `after` and `before` cursors, and the `beforeCheckpoint`, `afterCheckpoint` and `atCheckpoint` filters.

`MovePackage.receivedTransactionBlocks.first`.`Int` scalar

`MovePackage.receivedTransactionBlocks.after`.`String` scalar

`MovePackage.receivedTransactionBlocks.last`.`Int` scalar

`MovePackage.receivedTransactionBlocks.before`.`String` scalar

`MovePackage.receivedTransactionBlocks.filter`.`TransactionBlockFilter` input

`MovePackage.receivedTransactionBlocks.scanLimit`.`Int` scalar

`MovePackage.bcs`.`Base64` scalar

The Base64-encoded BCS serialization of the package's content.

`MovePackage.packageAtVersion`.`MovePackage` object

Fetch another version of this package (the package that shares this package's original ID, but has the specified `version`).

`MovePackage.packageAtVersion.version`.`Int!` non-null scalar

`MovePackage.packageVersions`.`MovePackageConnection!` non-null object

Fetch all versions of this package (packages that share this package's original ID), optionally bounding the versions exclusively from below with `afterVersion`, or from above with `beforeVersion`.

`MovePackage.packageVersions.first`.`Int` scalar

`MovePackage.packageVersions.after`.`String` scalar

`MovePackage.packageVersions.last`.`Int` scalar

`MovePackage.packageVersions.before`.`String` scalar

`MovePackage.packageVersions.filter`.`MovePackageVersionFilter` input

`MovePackage.latestPackage`.`MovePackage!` non-null object

Fetch the latest version of this package (the package with the highest `version` that shares this packages's original ID)

`MovePackage.module`.`MoveModule` object

A representation of the module called `name` in this package, including the structs and functions it defines.

`MovePackage.module.name`.`String!` non-null scalar

`MovePackage.modules`.`MoveModuleConnection` object

Paginate through the MoveModules defined in this package.

`MovePackage.modules.first`.`Int` scalar

`MovePackage.modules.after`.`String` scalar

`MovePackage.modules.last`.`Int` scalar

`MovePackage.modules.before`.`String` scalar

`MovePackage.linkage`.`[Linkage!]` list object

The transitive dependencies of this package.

`MovePackage.typeOrigins`.`[TypeOrigin!]` list object

The (previous) versions of this package that introduced its types.

`MovePackage.packageBcs`.`Base64` scalar

BCS representation of the package itself, as a MovePackage.

`MovePackage.moduleBcs`.`Base64` scalar

BCS representation of the package's modules. Modules appear as a sequence of pairs (module name, followed by module bytes), in alphabetic order by module name.

Interfaces

`IObject` interface

Interface implemented by on-chain values that are addressable by an ID (also referred to as its address). This includes Move objects and packages.

`IOwner` interface

Interface implemented by GraphQL types representing entities that can own objects. Object owners are identified by an address which can represent either the public key of an account or another object. The same address can only refer to an account or an object, never both, but it is not possible to know which up-front.

Returned By

latestPackage query . package query . packageByName query

Member Of

MoveModule object . MovePackage object . MovePackageConnection object . MovePackageEdge object . Object object

Edit this page

© 2025 SUI FOUNDATION | DOCUMENTATION DISTRIBUTED UNDER CC BY 4.0

MovePackageCheckpointFilter

Filter for paginating `MovePackage`s that were created within a range of checkpoints.

```
input MovePackageCheckpointFilter {
  afterCheckpoint: UInt53
  beforeCheckpoint: UInt53
}
```

Fields

`MovePackageCheckpointFilter.afterCheckpoint`.`UInt53` **scalar**

Fetch packages that were published strictly after this checkpoint. Omitting this fetches packages published since genesis.

`MovePackageCheckpointFilter.beforeCheckpoint`.`UInt53` **scalar**

Fetch packages that were published strictly before this checkpoint. Omitting this fetches packages published up to the latest checkpoint (inclusive).

Member Of

`packages` **query**

Edit this page

© 2025 SUI FOUNDATION | DOCUMENTATION DISTRIBUTED UNDER CC BY 4.0

MovePackageConnection

No description

```
type MovePackageConnection {
  pageInfo: PageInfo!
  edges: [MovePackageEdge!]!
  nodes: [MovePackage!]!
}
```

Fields

MovePackageConnection.pageInfo.PageInfo! non-null object

Information to aid in pagination.

MovePackageConnection.edges.[MovePackageEdge!]! non-null object

A list of edges.

MovePackageConnection.nodes.[MovePackage!]! non-null object

A list of nodes.

Returned By

packages query . packageVersions query

Member Of

ChangeEpochTransaction object . MovePackage object

Edit this page

© 2025 SUI FOUNDATION | DOCUMENTATION DISTRIBUTED UNDER CC BY 4.0

MovePackageEdge

An edge in a connection.

```
type MovePackageEdge {
  node: MovePackage!
  cursor: String!
}
```

Fields

`MovePackageEdge.node`.`MovePackage!` **non-null** **object**

The item at the end of the edge

`MovePackageEdge.cursor`.`String!` **non-null** **scalar**

A cursor for use in pagination

Member Of

`MovePackageConnection` **object**

Edit this page

© 2025 SUI FOUNDATION | DOCUMENTATION DISTRIBUTED UNDER CC BY 4.0

MovePackageVersionFilter

Filter for paginating versions of a given `MovePackage`.

```
input MovePackageVersionFilter {
  afterVersion: UInt53
  beforeVersion: UInt53
}
```

Fields

`MovePackageVersionFilter.afterVersion`.`UInt53` **scalar**

Fetch versions of this package that are strictly newer than this version. Omitting this fetches versions since the original version.

`MovePackageVersionFilter.beforeVersion`.`UInt53` **scalar**

Fetch versions of this package that are strictly older than this version. Omitting this fetches versions up to the latest version (inclusive).

Member Of

`packageVersions` **query**

Edit this page

© 2025 SUI FOUNDATION | DOCUMENTATION DISTRIBUTED UNDER CC BY 4.0

MoveStruct

Description of a struct type, defined in a Move module.

```
type MoveStruct implements IMoveDatatype {
  module: MoveModule!
  name: String!
  abilities: [MoveAbility!]
  typeParameters: [MoveStructTypeParameter!]
  fields: [MoveField!]
}
```

Fields

`MoveStruct.module` . `MoveModule!` non-null object

The module this struct was originally defined in.

`MoveStruct.name` . `String!` non-null scalar

The struct's (unqualified) type name.

`MoveStruct.abilities` . `[MoveAbility!]` list enum

Abilities this struct has.

`MoveStruct.typeParameters` . `[MoveStructTypeParameter!]` list object

Constraints on the struct's formal type parameters. Move bytecode does not name type parameters, so when they are referenced (e.g. in field types) they are identified by their index in this list.

`MoveStruct.fields` . `[MoveField!]` list object

The names and types of the struct's fields. Field types reference type parameters, by their index in the defining struct's `typeParameters` list.

Interfaces

`IMoveDatatype` interface

Interface implemented by all GraphQL types that represent a Move datatype (either structs or enums). This interface is used to provide a way to access fields that are shared by both structs and enums, e.g., the module that the datatype belongs to, the name of the datatype, type parameters etc.

Member Of

`MoveDatatype` **object** . `MoveModule` **object** . `MoveStructConnection` **object** . `MoveStructEdge` **object**

Edit this page

© 2025 SUI FOUNDATION | DOCUMENTATION DISTRIBUTED UNDER CC BY 4.0

MoveStructConnection

No description

```
type MoveStructConnection {
  pageInfo: PageInfo!
  edges: [MoveStructEdge!]!
  nodes: [MoveStruct!]!
}
```

Fields

`MoveStructConnection.pageInfo` . `PageInfo!` **non-null** **object**

Information to aid in pagination.

`MoveStructConnection.edges` . `[MoveStructEdge!]!` **non-null** **object**

A list of edges.

`MoveStructConnection.nodes` . `[MoveStruct!]!` **non-null** **object**

A list of nodes.

Member Of

`MoveModule` **object**

Edit this page

© 2025 SUI FOUNDATION | DOCUMENTATION DISTRIBUTED UNDER CC BY 4.0

MoveStructEdge

An edge in a connection.

```
type MoveStructEdge {
  node: MoveStruct!
  cursor: String!
}
```

Fields

`MoveStructEdge.node`.`MoveStruct!` **non-null** **object**

The item at the end of the edge

`MoveStructEdge.cursor`.`String!` **non-null** **scalar**

A cursor for use in pagination

Member Of

`MoveStructConnection` **object**

Edit this page

Sui

© 2025 SUI FOUNDATION | DOCUMENTATION DISTRIBUTED UNDER CC BY 4.0

MoveStructTypeParameter

No description

```
type MoveStructTypeParameter {
  constraints: [MoveAbility!]!
  isPhantom: Boolean!
}
```

Fields

`MoveStructTypeParameter.constraints.[MoveAbility!]!` non-null enum

`MoveStructTypeParameter.isPhantom.Boolean!` non-null scalar

Member Of

`IMoveDatatype` interface . `MoveDatatype` object . `MoveEnum` object . `MoveStruct` object

Edit this page

© 2025 SUI FOUNDATION | DOCUMENTATION DISTRIBUTED UNDER CC BY 4.0

MoveType

Represents concrete types (no type parameters, no references).

```
type MoveType {
  repr: String!
  signature: MoveTypeSignature!
  layout: MoveTypeLayout
  abilities: [MoveAbility!]
}
```

Fields

MoveType.repr: String! non-null scalar

Flat representation of the type signature, as a displayable string.

MoveType.signature: MoveTypeSignature! non-null scalar

Structured representation of the type signature.

MoveType.layout: MoveTypeLayout scalar

Structured representation of the "shape" of values that match this type. May return no layout if the type is invalid.

MoveType.abilities: [MoveAbility!] list enum

The abilities this concrete type has. Returns no abilities if the type is invalid.

Returned By

type query . typeByName query

Member Of

Balance object . BalanceChange object . DryRunMutation object . DryRunReturn object . MakeMoveVecTransaction object . MoveCallTransaction object . MoveValue object

© 2025 SUI FOUNDATION | DOCUMENTATION DISTRIBUTED UNDER CC BY 4.0

MoveTypeLayout

The shape of a concrete Move Type (a type with all its type parameters instantiated with concrete types), corresponding to the following recursive type:

type MoveTypeLayout = "address" | "bool" | "u8" | "u16" | ... | "u256" | { vector: MoveTypeLayout } | { struct: { type: string, fields: [{ name: string, layout: MoveTypeLayout }], } } | { enum: [{ type: string, variants: [{ name: string, fields: [{ name: string, layout: MoveTypeLayout }], }] }] }

```
scalar MoveTypeLayout
```

Member Of

MoveType **object**

Edit this page

© 2025 SUI FOUNDATION | DOCUMENTATION DISTRIBUTED UNDER CC BY 4.0

MoveTypeSignature

The signature of a concrete Move Type (a type with all its type parameters instantiated with concrete types, that contains no references), corresponding to the following recursive type:

type MoveTypeSignature = "address" | "bool" | "u8" | "u16" | ... | "u256" | { vector: MoveTypeSignature } | { datatype: { package: string, module: string, type: string, typeParameters: [MoveTypeSignature], } }

```
scalar MoveTypeSignature
```

Member Of

MoveType **object**

Edit this page

© 2025 SUI FOUNDATION | DOCUMENTATION DISTRIBUTED UNDER CC BY 4.0

MoveValue

No description

```
type MoveValue {
  type: MoveType!
  bcs: Base64!
  data: MoveData!
  json: JSON!
}
```

Fields

`MoveValue.type` . `MoveType!` non-null object

The value's Move type.

`MoveValue.bcs` . `Base64!` non-null scalar

The BCS representation of this value, Base64 encoded.

`MoveValue.data` . `MoveData!` non-null scalar

Structured contents of a Move value.

`MoveValue.json` . `JSON!` non-null scalar

Representation of a Move value in JSON, where:

- Addresses, IDs, and UIDs are represented in canonical form, as JSON strings.
- Bools are represented by JSON boolean literals.
- u8, u16, and u32 are represented as JSON numbers.
- u64, u128, and u256 are represented as JSON strings.
- Vectors are represented by JSON arrays.
- Structs are represented by JSON objects.
- Empty optional values are represented by `null`.

This form is offered as a less verbose convenience in cases where the layout of the type is known by the client.

Member Of

Coin object . CoinMetadata object . DynamicField object . Event object . IMoveObject interface . MoveObject object . StakedSui object . SuinsRegistration object

Implemented By

DynamicFieldValue union

Edit this page

© 2025 SUI FOUNDATION | DOCUMENTATION DISTRIBUTED UNDER CC BY 4.0

MoveVisibility

The visibility modifier describes which modules can access this module member. By default, a module member can be called only within the same module.

```
enum MoveVisibility {
  PUBLIC
  PRIVATE
  FRIEND
}
```

Values

`MoveVisibility.PUBLIC`

A public member can be accessed by any module.

`MoveVisibility.PRIVATE`

A private member can be accessed in the module it is defined in.

`MoveVisibility.FRIEND`

A friend member can be accessed in the module it is defined in and any other module in its package that is explicitly specified in its friend list.

Member Of

`MoveFunction` **object**

Edit this page

© 2025 SUI FOUNDATION | DOCUMENTATION DISTRIBUTED UNDER CC BY 4.0

multiGetObjects

Fetch a list of objects by their IDs and versions.

```
multiGetObjects(
  keys: [ObjectKey!]!
): [Object!]!
```

Arguments

`multiGetObjects.keys.[ObjectKey!]!` **non-null** **input**

Type

`Object` **object**

An object in Sui is a package (set of Move bytecode modules) or object (typed data structure with fields) with additional metadata detailing its id, version, transaction digest, owner field indicating how this object can be accessed.

Edit this page

© 2025 SUI FOUNDATION | DOCUMENTATION DISTRIBUTED UNDER CC BY 4.0

Object

An object in Sui is a package (set of Move bytecode modules) or object (typed data structure with fields) with additional metadata detailing its id, version, transaction digest, owner field indicating how this object can be accessed.

```
type Object implements IObject, IOwner {
  address: SuiAddress!
  objects(
    first: Int
    after: String
    last: Int
    before: String
    filter: ObjectFilter
  ): MoveObjectConnection!
  balance(
    type: String
  ): Balance
  balances(
    first: Int
    after: String
    last: Int
    before: String
  ): BalanceConnection!
  coins(
    first: Int
    after: String
    last: Int
    before: String
    type: String
  ): CoinConnection!
  stakedSuis(
    first: Int
    after: String
    last: Int
    before: String
  ): StakedSuiConnection!
  defaultSuinsName(
    format: DomainFormat
  ): String
  suinsRegistrations(
    first: Int
    after: String
    last: Int
    before: String
  ): SuinsRegistrationConnection!
  version: UInt53!
  status: ObjectKind!
```

```
  digest: String
  owner: ObjectOwner
  previousTransactionBlock: TransactionBlock
  storageRebate: BigInt
  receivedTransactionBlocks(
    first: Int
    after: String
    last: Int
    before: String
    filter: TransactionBlockFilter
    scanLimit: Int
  ): TransactionBlockConnection!
  bcs: Base64
  display: [DisplayEntry!]
  dynamicField(
    name: DynamicFieldName!
  ): DynamicField
  dynamicObjectField(
    name: DynamicFieldName!
  ): DynamicField
  dynamicFields(
    first: Int
    after: String
    last: Int
    before: String
  ): DynamicFieldConnection!
  asMoveObject: MoveObject
  asMovePackage: MovePackage
}
```

Fields

`Object.address`.`SuiAddress!` **non-null** **scalar**

`Object.objects`.`MoveObjectConnection!` **non-null** **object**

Objects owned by this object, optionally `filter`-ed.

`Object.objects.first`.`Int` **scalar**

`Object.objects.after`.`String` **scalar**

`Object.objects.last`.`Int` **scalar**

`Object.objects.before`.`String` **scalar**

`Object.objects.filter`.`ObjectFilter` **input**

`Object.balance`.`Balance` **object**

Total balance of all coins with marker type owned by this object. If type is not supplied, it defaults to `0x2::sui::SUI`.

`Object.balance.type`.`String` scalar

`Object.balances`.`BalanceConnection!` non-null object

The balances of all coin types owned by this object.

`Object.balances.first`.`Int` scalar

`Object.balances.after`.`String` scalar

`Object.balances.last`.`Int` scalar

`Object.balances.before`.`String` scalar

`Object.coins`.`CoinConnection!` non-null object

The coin objects for this object.

`type` is a filter on the coin's type parameter, defaulting to `0x2::sui::SUI`.

`Object.coins.first`.`Int` scalar

`Object.coins.after`.`String` scalar

`Object.coins.last`.`Int` scalar

`Object.coins.before`.`String` scalar

`Object.coins.type`.`String` scalar

`Object.stakedSuis`.`StakedSuiConnection!` non-null object

The `0x3::staking_pool::StakedSui` objects owned by this object.

`Object.stakedSuis.first`.`Int` scalar

`Object.stakedSuis.after`.`String` scalar

`Object.stakedSuis.last`.`Int` scalar

`Object.stakedSuis.before`.`String` scalar

`Object.defaultSuinsName`.`String` scalar

The domain explicitly configured as the default domain pointing to this object.

`Object.defaultSuinsName.format`.`DomainFormat` enum

`Object.suinsRegistrations`.`SuinsRegistrationConnection!` non-null object

The SuinsRegistration NFTs owned by this object. These grant the owner the capability to manage the associated domain.

`Object.suinsRegistrations.first`.`Int` scalar

`Object.suinsRegistrations.after`.`String` scalar

`Object.suinsRegistrations.last`.`Int` scalar

`Object.suinsRegistrations.before`.`String` scalar

`Object.version`.`UInt53!` non-null scalar

`Object.status`.`ObjectKind!` non-null enum

The current status of the object as read from the off-chain store. The possible states are: NOT_INDEXED, the object is loaded from serialized data, such as the contents of a genesis or system package upgrade transaction. LIVE, the version returned is the most recent for the object, and it is not deleted or wrapped at that version. HISTORICAL, the object was referenced at a specific version or checkpoint, so is fetched from historical tables and may not be the latest version of the object. WRAPPED_OR_DELETED, the object is deleted or wrapped and only partial information can be loaded."

`Object.digest`.`String` scalar

32-byte hash that identifies the object's current contents, encoded as a Base58 string.

`Object.owner`.`ObjectOwner` union

The owner type of this object: Immutable, Shared, Parent, Address Immutable and Shared Objects do not have owners.

`Object.previousTransactionBlock`.`TransactionBlock` object

The transaction block that created this version of the object.

`Object.storageRebate`.`BigInt` scalar

The amount of SUI we would rebate if this object gets deleted or mutated. This number is recalculated based on the present storage gas price.

`Object.receivedTransactionBlocks.TransactionBlockConnection!` non-null object

The transaction blocks that sent objects to this object.

`scanLimit` restricts the number of candidate transactions scanned when gathering a page of results. It is required for queries that apply more than two complex filters (on function, kind, sender, recipient, input object, changed object, or ids), and can be at most `serviceConfig.maxScanLimit`.

When the scan limit is reached the page will be returned even if it has fewer than `first` results when paginating forward (`last` when paginating backwards). If there are more transactions to scan, `pageInfo.hasNextPage` (or `pageInfo.hasPreviousPage`) will be set to `true`, and `PageInfo.endCursor` (or `PageInfo.startCursor`) will be set to the last transaction that was scanned as opposed to the last (or first) transaction in the page.

Requesting the next (or previous) page after this cursor will resume the search, scanning the next `scanLimit` many transactions in the direction of pagination, and so on until all transactions in the scanning range have been visited.

By default, the scanning range includes all transactions known to GraphQL, but it can be restricted by the `after` and `before` cursors, and the `beforeCheckpoint`, `afterCheckpoint` and `atCheckpoint` filters.

`Object.receivedTransactionBlocks.first.Int` scalar

`Object.receivedTransactionBlocks.after.String` scalar

`Object.receivedTransactionBlocks.last.Int` scalar

`Object.receivedTransactionBlocks.before.String` scalar

`Object.receivedTransactionBlocks.filter.TransactionBlockFilter` input

`Object.receivedTransactionBlocks.scanLimit.Int` scalar

`Object.bcs.Base64` scalar

The Base64-encoded BCS serialization of the object's content.

`Object.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.

`Object.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 Owner type.

`Object.dynamicField.name`.`DynamicFieldName!` non-null input

`Object.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.

`Object.dynamicObjectField.name`.`DynamicFieldName!` non-null input

`Object.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.

`Object.dynamicFields.first`.`Int` scalar

`Object.dynamicFields.after`.`String` scalar

`Object.dynamicFields.last`.`Int` scalar

`Object.dynamicFields.before`.`String` scalar

`Object.asMoveObject`.`MoveObject` object

Attempts to convert the object into a MoveObject

`Object.asMovePackage`.`MovePackage` object

Attempts to convert the object into a MovePackage

Interfaces

`IObject` interface

Interface implemented by on-chain values that are addressable by an ID (also referred to as its address). This includes Move objects and packages.

`IOwner` interface

Interface implemented by GraphQL types representing entities that can own objects. Object owners are identified by an address which can represent either the public key of an account or another object. The same address can only refer to an account or an object, never both, but it is not possible to know which up-front.

Returned By

`multiGetObjects` query . `object` query

Member Of

`GasEffects` object . `ObjectChange` object . `ObjectConnection` object . `ObjectEdge` object . `OwnedOrImmutable` object . `Owner` object . `Receiving` object . `SharedObjectRead` object

Edit this page

© 2025 SUI FOUNDATION | DOCUMENTATION DISTRIBUTED UNDER CC BY 4.0

ObjectChange

Effect on an individual Object (keyed by its ID).

```
type ObjectChange {
  address: SuiAddress!
  inputState: Object
  outputState: Object
  idCreated: Boolean
  idDeleted: Boolean
}
```

Fields

ObjectChange.address.SuiAddress! non-null scalar

The address of the object that has changed.

ObjectChange.inputState.Object object

The contents of the object immediately before the transaction.

ObjectChange.outputState.Object object

The contents of the object immediately after the transaction.

ObjectChange.idCreated.Boolean scalar

Whether the ID was created in this transaction.

ObjectChange.idDeleted.Boolean scalar

Whether the ID was deleted in this transaction.

Member Of

ObjectChangeConnection object . ObjectChangeEdge object

© 2025 SUI FOUNDATION | DOCUMENTATION DISTRIBUTED UNDER CC BY 4.0

ObjectChangeConnection

No description

```
type ObjectChangeConnection {
  pageInfo: PageInfo!
  edges: [ObjectChangeEdge!]!
  nodes: [ObjectChange!]!
}
```

Fields

`ObjectChangeConnection.pageInfo.PageInfo!` non-null object

Information to aid in pagination.

`ObjectChangeConnection.edges.[ObjectChangeEdge!]!` non-null object

A list of edges.

`ObjectChangeConnection.nodes.[ObjectChange!]!` non-null object

A list of nodes.

Member Of

`TransactionBlockEffects` object

Edit this page

© 2025 SUI FOUNDATION | DOCUMENTATION DISTRIBUTED UNDER CC BY 4.0

ObjectChangeEdge

An edge in a connection.

```
type ObjectChangeEdge {
  node: ObjectChange!
  cursor: String!
}
```

Fields

`ObjectChangeEdge.node`.`ObjectChange!` **non-null** **object**

The item at the end of the edge

`ObjectChangeEdge.cursor`.`String!` **non-null** **scalar**

A cursor for use in pagination

Member Of

`ObjectChangeConnection` **object**

Edit this page

© 2025 SUI FOUNDATION | DOCUMENTATION DISTRIBUTED UNDER CC BY 4.0

ObjectConnection

No description

```
type ObjectConnection {
  pageInfo: PageInfo!
  edges: [ObjectEdge!]!
  nodes: [Object!]!
}
```

Fields

`ObjectConnection.pageInfo.PageInfo!` non-null object

Information to aid in pagination.

`ObjectConnection.edges.[ObjectEdge!]!` non-null object

A list of edges.

`ObjectConnection.nodes.[Object!]!` non-null object

A list of nodes.

Returned By

`objects` query

Member Of

`GasInput` object . `GenesisTransaction` object

Edit this page

© 2025 SUI FOUNDATION | DOCUMENTATION DISTRIBUTED UNDER CC BY 4.0

ObjectEdge

An edge in a connection.

```
type ObjectEdge {
  node: Object!
  cursor: String!
}
```

Fields

`ObjectEdge.node`.`Object!` **non-null** **object**

The item at the end of the edge

`ObjectEdge.cursor`.`String!` **non-null** **scalar**

A cursor for use in pagination

Member Of

`ObjectConnection` **object**

Edit this page

© 2025 SUI FOUNDATION | DOCUMENTATION DISTRIBUTED UNDER CC BY 4.0

ObjectFilter

Constrains the set of objects returned. All filters are optional, and the resulting set of objects are ones whose

- Type matches the `type` filter,
- AND, whose owner matches the `owner` filter,
- AND, whose ID is in `objectIds` OR whose ID and version is in `objectKeys`.

```
input ObjectFilter {
  type: String
  owner: SuiAddress
  objectIds: [SuiAddress!]
  objectKeys: [ObjectKey!]
}
```

Fields

`ObjectFilter.type.String` scalar

Filter objects by their type's `package`, `package::module`, or their fully qualified type name.

Generic types can be queried by either the generic type name, e.g. `0x2::coin::Coin`, or by the full type name, such as `0x2::coin::Coin<0x2::sui::SUI>`.

`ObjectFilter.owner.SuiAddress` scalar

Filter for live objects by their current owners.

`ObjectFilter.objectIds.[SuiAddress!]` list scalar

Filter for live objects by their IDs.

`ObjectFilter.objectKeys.[ObjectKey!]` list input

Filter for live objects by their ID and version. NOTE: this input filter has been deprecated in favor of `multiGetObjects` query as it does not make sense to query for live objects by their versions. This filter will be removed with v1.42.0 release.

Member Of

`objects` query

Edit this page

© 2025 SUI FOUNDATION | DOCUMENTATION DISTRIBUTED UNDER CC BY 4.0

ObjectKey

No description

```
input ObjectKey {
  objectId: SuiAddress!
  version: UInt53!
}
```

Fields

`ObjectKey.objectId`: `SuiAddress!` **non-null** **scalar**

`ObjectKey.version`: `UInt53!` **non-null** **scalar**

Member Of

`multiGetObjects` **query** . `ObjectFilter` **input**

Edit this page

© 2025 SUI FOUNDATION | DOCUMENTATION DISTRIBUTED UNDER CC BY 4.0

ObjectKind

No description

```
enum ObjectKind {
  NOT_INDEXED
  INDEXED
}
```

Values

`ObjectKind.NOT_INDEXED`

The object is loaded from serialized data, such as the contents of a transaction that hasn't been indexed yet.

`ObjectKind.INDEXED`

The object is fetched from the index.

Member Of

`Coin` object . `CoinMetadata` object . `IObject` interface . `MoveObject` object . `MovePackage` object . `Object` object . `StakedSui` object . `SuinsRegistration` object

Edit this page

© 2025 SUI FOUNDATION | DOCUMENTATION DISTRIBUTED UNDER CC BY 4.0

ObjectOwner

The object's owner type: Immutable, Shared, Parent, or Address.

```
union ObjectOwner = Immutable | Shared | Parent | AddressOwner | ConsensusV2
```

Possible types

ObjectOwner.Immutable object

An immutable object is an object that can't be mutated, transferred, or deleted. Immutable objects have no owner, so anyone can use them.

ObjectOwner.Shared object

A shared object is an object that is shared using the 0x2::transfer::share_object function. Unlike owned objects, once an object is shared, it stays mutable and is accessible by anyone.

ObjectOwner.Parent object

If the object's owner is a Parent, this object is part of a dynamic field (it is the value of the dynamic field, or the intermediate Field object itself), and it is owned by another object.

Although its owner is guaranteed to be an object, it is exposed as an Owner, as the parent object could be wrapped and therefore not directly accessible.

ObjectOwner.AddressOwner object

An address-owned object is owned by a specific 32-byte address that is either an account address (derived from a particular signature scheme) or an object ID. An address-owned object is accessible only to its owner and no others.

ObjectOwner.ConsensusV2 object

A ConsensusV2 object is an object that is automatically versioned by the consensus protocol and allows different authentication modes based on the chosen authenticator. (Initially, only single-owner authentication is supported.)

Member Of

Coin object . CoinMetadata object . IObject interface . MoveObject object . MovePackage object . Object object . StakedSui object . SuinsRegistration object

Edit this page

© 2025 SUI FOUNDATION | DOCUMENTATION DISTRIBUTED UNDER CC BY 4.0

ObjectRef

No description

```
input ObjectRef {
  address: SuiAddress!
  version: UInt53!
  digest: String!
}
```

Fields

ObjectRef.address: SuiAddress! non-null scalar

ID of the object.

ObjectRef.version: UInt53! non-null scalar

Version or sequence number of the object.

ObjectRef.digest: String! non-null scalar

Digest of the object.

Member Of

TransactionMetadata input

Edit this page

© 2025 SUI FOUNDATION | DOCUMENTATION DISTRIBUTED UNDER CC BY 4.0

objects

The objects that exist in the network.

```
objects(
  first: Int
  after: String
  last: Int
  before: String
  filter: ObjectFilter
): ObjectConnection!
```

Arguments

`objects.first`.`Int` **scalar**

`objects.after`.`String` **scalar**

`objects.last`.`Int` **scalar**

`objects.before`.`String` **scalar**

`objects.filter`.`ObjectFilter` **input**

Type

`ObjectConnection` **object**

Edit this page

© 2025 SUI FOUNDATION | DOCUMENTATION DISTRIBUTED UNDER CC BY 4.0

oneOf

Indicates exactly one field must be supplied and this field must not be `null`.

```
directive @oneOf on INPUT_OBJECT
```

Edit this page

© 2025 SUI FOUNDATION | DOCUMENTATION DISTRIBUTED UNDER CC BY 4.0

OpenMoveType

Represents types that could contain references or free type parameters. Such types can appear as function parameters, in fields of structs, or as actual type parameter.

```
type OpenMoveType {
  signature: OpenMoveTypeSignature!
  repr: String!
}
```

Fields

`OpenMoveType.signature`.`OpenMoveTypeSignature!` **non-null** **scalar**

Structured representation of the type signature.

`OpenMoveType.repr`.`String!` **non-null** **scalar**

Flat representation of the type signature, as a displayable string.

Member Of

`MoveField` **object** . `MoveFunction` **object**

Edit this page

© 2025 SUI FOUNDATION | DOCUMENTATION DISTRIBUTED UNDER CC BY 4.0

OpenMoveTypeSignature

The shape of an abstract Move Type (a type that can contain free type parameters, and can optionally be taken by reference), corresponding to the following recursive type:

type OpenMoveTypeSignature = { ref: ("&" | "&mut")?, body: OpenMoveTypeSignatureBody, }

type OpenMoveTypeSignatureBody = "address" | "bool" | "u8" | "u16" | ... | "u256" | { vector: OpenMoveTypeSignatureBody } | { datatype { package: string, module: string, type: string, typeParameters: [OpenMoveTypeSignatureBody] } } | { typeParameter: number }

```
scalar OpenMoveTypeSignature
```

Member Of

OpenMoveType **object**

Edit this page

© 2025 SUI FOUNDATION | DOCUMENTATION DISTRIBUTED UNDER CC BY 4.0

OwnedOrImmutable

A Move object, either immutable, or owned mutable.

```
type OwnedOrImmutable {
  address: SuiAddress!
  version: UInt53!
  digest: String!
  object: Object
}
```

Fields

`OwnedOrImmutable.address`.`SuiAddress!` **non-null** **scalar**

ID of the object being read.

`OwnedOrImmutable.version`.`UInt53!` **non-null** **scalar**

Version of the object being read.

`OwnedOrImmutable.digest`.`String!` **non-null** **scalar**

32-byte hash that identifies the object's contents at this version, encoded as a Base58 string.

`OwnedOrImmutable.object`.`Object` **object**

The object at this version. May not be available due to pruning.

Implemented By

`TransactionInput` **union**

Edit this page

© 2025 SUI FOUNDATION | DOCUMENTATION DISTRIBUTED UNDER CC BY 4.0

Owner

An Owner is an entity that can own an object. Each Owner is identified by a SuiAddress which represents either an Address (corresponding to a public key of an account) or an Object, but never both (it is not known up-front whether a given Owner is an Address or an Object).

```
type Owner implements IOwner {
  address: SuiAddress!
  objects(
    first: Int
    after: String
    last: Int
    before: String
    filter: ObjectFilter
  ): MoveObjectConnection!
  balance(
    type: String
  ): Balance
  balances(
    first: Int
    after: String
    last: Int
    before: String
  ): BalanceConnection!
  coins(
    first: Int
    after: String
    last: Int
    before: String
    type: String
  ): CoinConnection!
  stakedSuis(
    first: Int
    after: String
    last: Int
    before: String
  ): StakedSuiConnection!
  defaultSuinsName(
    format: DomainFormat
  ): String
  suinsRegistrations(
    first: Int
    after: String
    last: Int
    before: String
  ): SuinsRegistrationConnection!
  asAddress: Address
  asObject: Object
```

```
  dynamicField(
    name: DynamicFieldName!
  ): DynamicField
  dynamicObjectField(
    name: DynamicFieldName!
  ): DynamicField
  dynamicFields(
    first: Int
    after: String
    last: Int
    before: String
  ): DynamicFieldConnection!
}
```

Fields

`Owner.address`.`SuiAddress!` non-null scalar

`Owner.objects`.`MoveObjectConnection!` non-null object

Objects owned by this object or address, optionally `filter`-ed.

`Owner.objects.first`.`Int` scalar

`Owner.objects.after`.`String` scalar

`Owner.objects.last`.`Int` scalar

`Owner.objects.before`.`String` scalar

`Owner.objects.filter`.`ObjectFilter` input

`Owner.balance`.`Balance` object

Total balance of all coins with marker type owned by this object or address. If type is not supplied, it defaults to `0x2::sui::SUI`.

`Owner.balance.type`.`String` scalar

`Owner.balances`.`BalanceConnection!` non-null object

The balances of all coin types owned by this object or address.

`Owner.balances.first`.`Int` scalar

`Owner.balances.after`.`String` scalar

`Owner.balances.last`.`Int` scalar

`Owner.balances.before`.`String` scalar

`Owner.coins`.`CoinConnection!` non-null object

The coin objects for this object or address.

`type` is a filter on the coin's type parameter, defaulting to `0x2::sui::SUI`.

`Owner.coins.first`.`Int` scalar

`Owner.coins.after`.`String` scalar

`Owner.coins.last`.`Int` scalar

`Owner.coins.before`.`String` scalar

`Owner.coins.type`.`String` scalar

`Owner.stakedSuis`.`StakedSuiConnection!` non-null object

The `0x3::staking_pool::StakedSui` objects owned by this object or address.

`Owner.stakedSuis.first`.`Int` scalar

`Owner.stakedSuis.after`.`String` scalar

`Owner.stakedSuis.last`.`Int` scalar

`Owner.stakedSuis.before`.`String` scalar

`Owner.defaultSuinsName`.`String` scalar

The domain explicitly configured as the default domain pointing to this object or address.

`Owner.defaultSuinsName.format`.`DomainFormat` enum

`Owner.suinsRegistrations`.`SuinsRegistrationConnection!` non-null object

The SuinsRegistration NFTs owned by this object or address. These grant the owner the capability to manage the associated domain.

`Owner.suinsRegistrations.first`.`Int` scalar

`Owner.suinsRegistrations.after`.`String` scalar

`Owner.suinsRegistrations.last`.`Int` scalar

`Owner.suinsRegistrations.before`.`String` scalar

`Owner.asAddress`.`Address` **object**

`Owner.asObject`.`Object` **object**

`Owner.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.

This field exists as a convenience when accessing a dynamic field on a wrapped object.

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

`Owner.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`).

This field exists as a convenience when accessing a dynamic field on a wrapped object.

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

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

The dynamic fields and dynamic object fields on an object.

This field exists as a convenience when accessing a dynamic field on a wrapped object.

`Owner.dynamicFields.first`.`Int` **scalar**

`Owner.dynamicFields.after`.`String` **scalar**

`Owner.dynamicFields.last`.`Int` **scalar**

`Owner.dynamicFields.before`.`String` **scalar**

Interfaces

`IOwner` **interface**

Interface implemented by GraphQL types representing entities that can own objects. Object owners are identified by an address which can represent either the public key of an account or another object. The same address can only refer to an account or an object, never both, but it is not possible to know which up-front.

Returned By

`owner` query

Member Of

`AddressOwner` object . `BalanceChange` object . `Parent` object . `Validator` object

Edit this page

© 2025 SUI FOUNDATION | DOCUMENTATION DISTRIBUTED UNDER CC BY 4.0

package

The package corresponding to the given address (at the optionally given version).

When no version is given, the package is loaded directly from the address given. Otherwise, the address is translated before loading to point to the package whose original ID matches the package at `address`, but whose version is `version`. For non-system packages, this might result in a different address than `address` because different versions of a package, introduced by upgrades, exist at distinct addresses.

Note that this interpretation of `version` is different from a historical object read (the interpretation of `version` for the `object` query).

```
package(
  address: SuiAddress!
  version: UInt53
): MovePackage
```

Arguments

`package.address`: `SuiAddress!` **non-null** **scalar**

`package.version`: `UInt53` **scalar**

Type

`MovePackage` **object**

A MovePackage is a kind of Move object that represents code that has been published on chain. It exposes information about its modules, type definitions, functions, and dependencies.

Edit this page

© 2025 SUI FOUNDATION | DOCUMENTATION DISTRIBUTED UNDER CC BY 4.0

packageByName

Fetch a package by its name (using dot move service)

```
packageByName(
  name: String!
): MovePackage
```

Arguments

`packageByName.name`.`String!` **non-null** **scalar**

Type

`MovePackage` **object**

A MovePackage is a kind of Move object that represents code that has been published on chain. It exposes information about its modules, type definitions, functions, and dependencies.

Edit this page

© 2025 SUI FOUNDATION | DOCUMENTATION DISTRIBUTED UNDER CC BY 4.0

packages

The Move packages that exist in the network, optionally filtered to be strictly before `beforeCheckpoint` and/or strictly after `afterCheckpoint`.

This query returns all versions of a given user package that appear between the specified checkpoints, but only records the latest versions of system packages.

```
packages(
  first: Int
  after: String
  last: Int
  before: String
  filter: MovePackageCheckpointFilter
): MovePackageConnection!
```

Arguments

`packages.first.Int` **scalar**

`packages.after.String` **scalar**

`packages.last.Int` **scalar**

`packages.before.String` **scalar**

`packages.filter.MovePackageCheckpointFilter` **input**

Type

`MovePackageConnection` **object**

Edit this page

© 2025 SUI FOUNDATION | DOCUMENTATION DISTRIBUTED UNDER CC BY 4.0

packageVersions

Fetch all versions of package at `address` (packages that share this package's original ID), optionally bounding the versions exclusively from below with `afterVersion`, or from above with `beforeVersion`.

```
packageVersions(
  first: Int
  after: String
  last: Int
  before: String
  address: SuiAddress!
  filter: MovePackageVersionFilter
): MovePackageConnection!
```

Arguments

`packageVersions.first.Int` **scalar**

`packageVersions.after.String` **scalar**

`packageVersions.last.Int` **scalar**

`packageVersions.before.String` **scalar**

`packageVersions.address.SuiAddress!` **non-null** **scalar**

`packageVersions.filter.MovePackageVersionFilter` **input**

Type

`MovePackageConnection` **object**

Edit this page

© 2025 SUI FOUNDATION | DOCUMENTATION DISTRIBUTED UNDER CC BY 4.0

PageInfo

Information about pagination in a connection

```
type PageInfo {
  hasPreviousPage: Boolean!
  hasNextPage: Boolean!
  startCursor: String
  endCursor: String
}
```

Fields

PageInfo.hasPreviousPage.Boolean! non-null scalar

When paginating backwards, are there more items?

PageInfo.hasNextPage.Boolean! non-null scalar

When paginating forwards, are there more items?

PageInfo.startCursor.String scalar

When paginating backwards, the cursor to continue.

PageInfo.endCursor.String scalar

When paginating forwards, the cursor to continue.

Member Of

ActiveJwkConnection object . AddressConnection object . BalanceChangeConnection object . BalanceConnection object . CheckpointConnection object . CoinConnection object . DependencyConnection object . DynamicFieldConnection object . EndOfEpochTransactionKindConnection object . EpochConnection object . EventConnection object . MoveDatatypeConnection object . MoveEnumConnection object . MoveFunctionConnection object . MoveModuleConnection object . MoveObjectConnection object . MovePackageConnection object . MoveStructConnection object . ObjectChangeConnection object . ObjectConnection object . ProgrammableTransactionConnection object . StakedSuiConnection object . SuinsRegistrationConnection object . TransactionBlockConnection object

- `TransactionInputConnection` **object**
- `UnchangedSharedObjectConnection` **object**
- `ValidatorConnection` **object**

Edit this page

© 2025 SUI FOUNDATION | DOCUMENTATION DISTRIBUTED UNDER CC BY 4.0

Parent

If the object's owner is a Parent, this object is part of a dynamic field (it is the value of the dynamic field, or the intermediate Field object itself), and it is owned by another object.

Although its owner is guaranteed to be an object, it is exposed as an Owner, as the parent object could be wrapped and therefore not directly accessible.

```
type Parent {
  parent: Owner
}
```

Fields

`Parent.parent`.`Owner` **object**

Implemented By

`ObjectOwner` **union**

Edit this page

© 2025 SUI FOUNDATION | DOCUMENTATION DISTRIBUTED UNDER CC BY 4.0

ProgrammableTransaction

A single transaction, or command, in the programmable transaction block.

```
union ProgrammableTransaction = MoveCallTransaction |
TransferObjectsTransaction | SplitCoinsTransaction | MergeCoinsTransaction |
PublishTransaction | UpgradeTransaction | MakeMoveVecTransaction
```

Possible types

`ProgrammableTransaction.MoveCallTransaction` **object**

A call to either an entry or a public Move function.

`ProgrammableTransaction.TransferObjectsTransaction` **object**

Transfers `inputs` to `address`. All inputs must have the `store` ability (allows public transfer) and must not be previously immutable or shared.

`ProgrammableTransaction.SplitCoinsTransaction` **object**

Splits off coins with denominations in `amounts` from `coin`, returning multiple results (as many as there are amounts.)

`ProgrammableTransaction.MergeCoinsTransaction` **object**

Merges `coins` into the first `coin` (produces no results).

`ProgrammableTransaction.PublishTransaction` **object**

Publishes a Move Package.

`ProgrammableTransaction.UpgradeTransaction` **object**

Upgrades a Move Package.

`ProgrammableTransaction.MakeMoveVecTransaction` **object**

Create a vector (possibly empty).

Member Of

`ProgrammableTransactionConnection` **object** . `ProgrammableTransactionEdge` **object**

Edit this page

© 2025 SUI FOUNDATION | DOCUMENTATION DISTRIBUTED UNDER CC BY 4.0

ProgrammableTransactionBlock

A user transaction that allows the interleaving of native commands (like transfer, split coins, merge coins, etc) and move calls, executed atomically.

```
type ProgrammableTransactionBlock {
  inputs(
    first: Int
    after: String
    last: Int
    before: String
  ): TransactionInputConnection!
  transactions(
    first: Int
    after: String
    last: Int
    before: String
  ): ProgrammableTransactionConnection!
}
```

Fields

ProgrammableTransactionBlock.inputs. TransactionInputConnection! non-null object

Input objects or primitive values.

ProgrammableTransactionBlock.inputs.first. Int scalar

ProgrammableTransactionBlock.inputs.after. String scalar

ProgrammableTransactionBlock.inputs.last. Int scalar

ProgrammableTransactionBlock.inputs.before. String scalar

ProgrammableTransactionBlock.transactions. ProgrammableTransactionConnection! non-null object

The transaction commands, executed sequentially.

ProgrammableTransactionBlock.transactions.first. Int scalar

`ProgrammableTransactionBlock.transactions.after`.`String` scalar

`ProgrammableTransactionBlock.transactions.last`.`Int` scalar

`ProgrammableTransactionBlock.transactions.before`.`String` scalar

Implemented By

`TransactionBlockKind` union

Edit this page

© 2025 SUI FOUNDATION | DOCUMENTATION DISTRIBUTED UNDER CC BY 4.0

ProgrammableTransactionConnection

No description

```
type ProgrammableTransactionConnection {
  pageInfo: PageInfo!
  edges: [ProgrammableTransactionEdge!]!
  nodes: [ProgrammableTransaction!]!
}
```

Fields

`ProgrammableTransactionConnection.pageInfo.PageInfo!` non-null object

Information to aid in pagination.

`ProgrammableTransactionConnection.edges.[ProgrammableTransactionEdge!]!` non-null object

A list of edges.

`ProgrammableTransactionConnection.nodes.[ProgrammableTransaction!]!` non-null union

A list of nodes.

Member Of

`ProgrammableTransactionBlock` object

Edit this page

© 2025 SUI FOUNDATION | DOCUMENTATION DISTRIBUTED UNDER CC BY 4.0

ProgrammableTransactionEdge

An edge in a connection.

```
type ProgrammableTransactionEdge {
  node: ProgrammableTransaction!
  cursor: String!
}
```

Fields

`ProgrammableTransactionEdge.node.ProgrammableTransaction!` **non-null** **union**

The item at the end of the edge

`ProgrammableTransactionEdge.cursor.String!` **non-null** **scalar**

A cursor for use in pagination

Member Of

`ProgrammableTransactionConnection` **object**

Edit this page

© 2025 SUI FOUNDATION | DOCUMENTATION DISTRIBUTED UNDER CC BY 4.0

protocolConfig

Fetch the protocol config by protocol version (defaults to the latest protocol version known to the GraphQL service).

```
protocolConfig(
  protocolVersion: UInt53
): ProtocolConfigs!
```

Arguments

`protocolConfig.protocolVersion.UInt53` **scalar**

Type

`ProtocolConfigs` **object**

Constants that control how the chain operates.

These can only change during protocol upgrades which happen on epoch boundaries.

Edit this page

© 2025 SUI FOUNDATION | DOCUMENTATION DISTRIBUTED UNDER CC BY 4.0

ProtocolConfigAttr

A single protocol configuration value.

```
type ProtocolConfigAttr {
  key: String!
  value: String
}
```

Fields

`ProtocolConfigAttr.key.String!` non-null scalar

`ProtocolConfigAttr.value.String` scalar

Member Of

`ProtocolConfigs` object

Edit this page

© 2025 SUI FOUNDATION | DOCUMENTATION DISTRIBUTED UNDER CC BY 4.0

ProtocolConfigFeatureFlag

Whether or not a single feature is enabled in the protocol config.

```
type ProtocolConfigFeatureFlag {
  key: String!
  value: Boolean!
}
```

Fields

`ProtocolConfigFeatureFlag.key`. `String!` non-null scalar

`ProtocolConfigFeatureFlag.value`. `Boolean!` non-null scalar

Member Of

`ProtocolConfigs` object

Edit this page

© 2025 SUI FOUNDATION | DOCUMENTATION DISTRIBUTED UNDER CC BY 4.0

ProtocolConfigs

Constants that control how the chain operates.

These can only change during protocol upgrades which happen on epoch boundaries.

```
type ProtocolConfigs {
  protocolVersion: UInt53!
  featureFlags: [ProtocolConfigFeatureFlag!]!
  configs: [ProtocolConfigAttr!]!
  config(
    key: String!
  ): ProtocolConfigAttr
  featureFlag(
    key: String!
  ): ProtocolConfigFeatureFlag
}
```

Fields

`ProtocolConfigs.protocolVersion.UInt53!` non-null scalar

The protocol is not required to change on every epoch boundary, so the protocol version tracks which change to the protocol these configs are from.

`ProtocolConfigs.featureFlags.[ProtocolConfigFeatureFlag!]!` non-null object

List all available feature flags and their values. Feature flags are a form of boolean configuration that are usually used to gate features while they are in development. Once a flag has been enabled, it is rare for it to be disabled.

`ProtocolConfigs.configs.[ProtocolConfigAttr!]!` non-null object

List all available configurations and their values. These configurations can take any value (but they will all be represented in string form), and do not include feature flags.

`ProtocolConfigs.config.ProtocolConfigAttr` object

Query for the value of the configuration with name `key`.

`ProtocolConfigs.config.key`.`String!` non-null scalar

`ProtocolConfigs.featureFlag`.`ProtocolConfigFeatureFlag` object

Query for the state of the feature flag with name `key`.

`ProtocolConfigs.featureFlag.key`.`String!` non-null scalar

Returned By

`protocolConfig` query

Member Of

`Epoch` object

Edit this page

© 2025 SUI FOUNDATION | DOCUMENTATION DISTRIBUTED UNDER CC BY 4.0

PublishTransaction

Publishes a Move Package.

```
type PublishTransaction {
  modules: [Base64!]!
  dependencies: [SuiAddress!]!
}
```

Fields

`PublishTransaction.modules.[Base64!]!` **non-null** **scalar**

Bytecode for the modules to be published, BCS serialized and Base64 encoded.

`PublishTransaction.dependencies.[SuiAddress!]!` **non-null** **scalar**

IDs of the transitive dependencies of the package to be published.

Implemented By

`ProgrammableTransaction` **union**

Edit this page

© 2025 SUI FOUNDATION | DOCUMENTATION DISTRIBUTED UNDER CC BY 4.0

Pure

BCS encoded primitive value (not an object or Move struct).

```
type Pure {
  bytes: Base64!
}
```

Fields

`Pure.bytes`.`Base64!` **non-null** **scalar**

BCS serialized and Base64 encoded primitive value.

Implemented By

`TransactionInput` **union**

Edit this page

© 2025 SUI FOUNDATION | DOCUMENTATION DISTRIBUTED UNDER CC BY 4.0

RandomnessStateCreateTransaction

No description

```
type RandomnessStateCreateTransaction {
  _: Boolean
}
```

Fields

`RandomnessStateCreateTransaction._` . `Boolean` **scalar**

A workaround to define an empty variant of a GraphQL union.

Implemented By

`EndOfEpochTransactionKind` **union**

Edit this page

© 2025 SUI FOUNDATION | DOCUMENTATION DISTRIBUTED UNDER CC BY 4.0

RandomnessStateUpdateTransaction

System transaction to update the source of on-chain randomness.

```
type RandomnessStateUpdateTransaction {
  epoch: Epoch
  randomnessRound: UInt53!
  randomBytes: Base64!
  randomnessObjInitialSharedVersion: UInt53!
}
```

Fields

`RandomnessStateUpdateTransaction.epoch.Epoch` **object**

Epoch of the randomness state update transaction.

`RandomnessStateUpdateTransaction.randomnessRound.UInt53!` **non-null** **scalar**

Randomness round of the update.

`RandomnessStateUpdateTransaction.randomBytes.Base64!` **non-null** **scalar**

Updated random bytes, encoded as Base64.

`RandomnessStateUpdateTransaction.randomnessObjInitialSharedVersion.UInt53!` **non-null** **scalar**

The initial version the randomness object was shared at.

Implemented By

`TransactionBlockKind` **union**

Edit this page

© 2025 SUI FOUNDATION | DOCUMENTATION DISTRIBUTED UNDER CC BY 4.0

Receiving

A Move object that can be received in this transaction.

```
type Receiving {
  address: SuiAddress!
  version: UInt53!
  digest: String!
  object: Object
}
```

Fields

`Receiving.address` . `SuiAddress!` non-null scalar

ID of the object being read.

`Receiving.version` . `UInt53!` non-null scalar

Version of the object being read.

`Receiving.digest` . `String!` non-null scalar

32-byte hash that identifies the object's contents at this version, encoded as a Base58 string.

`Receiving.object` . `Object` object

The object at this version. May not be available due to pruning.

Implemented By

`TransactionInput` union

Edit this page

© 2025 SUI FOUNDATION | DOCUMENTATION DISTRIBUTED UNDER CC BY 4.0

resolveSuinsAddress

Resolves a SuiNS `domain` name to an address, if it has been bound.

```
resolveSuinsAddress(
  domain: String!
): Address
```

Arguments

`resolveSuinsAddress.domain`.`String!` **non-null** **scalar**

Type

`Address` **object**

The 32-byte address that is an account address (corresponding to a public key).

Edit this page

© 2025 SUI FOUNDATION | DOCUMENTATION DISTRIBUTED UNDER CC BY 4.0

Result

The result of another transaction command.

```
type Result {
  cmd: Int!
  ix: Int
}
```

Fields

`Result.cmd`.`Int!` non-null scalar

The index of the previous command (0-indexed) that returned this result.

`Result.ix`.`Int` scalar

If the previous command returns multiple values, this is the index of the individual result among the multiple results from that command (also 0-indexed).

Implemented By

`TransactionArgument` union

Edit this page

© 2025 SUI FOUNDATION | DOCUMENTATION DISTRIBUTED UNDER CC BY 4.0

SafeMode

Information about whether epoch changes are using safe mode.

```
type SafeMode {
  enabled: Boolean
  gasSummary: GasCostSummary
}
```

Fields

`SafeMode.enabled`.`Boolean` **scalar**

Whether safe mode was used for the last epoch change. The system will retry a full epoch change on every epoch boundary and automatically reset this flag if so.

`SafeMode.gasSummary`.`GasCostSummary` **object**

Accumulated fees for computation and cost that have not been added to the various reward pools, because the full epoch change did not happen.

Member Of

`Epoch` **object**

Edit this page

© 2025 SUI FOUNDATION | DOCUMENTATION DISTRIBUTED UNDER CC BY 4.0

ServiceConfig

The enabled features and service limits configured by the server.

```
type ServiceConfig {
  isEnabled(
    feature: Feature!
  ): Boolean!
  enabledFeatures: [Feature!]!
  maxQueryDepth: Int!
  maxQueryNodes: Int!
  maxOutputNodes: Int!
  maxDbQueryCost: Int!
  defaultPageSize: Int!
  maxPageSize: Int!
  mutationTimeoutMs: Int!
  requestTimeoutMs: Int!
  maxTransactionPayloadSize: Int!
  maxQueryPayloadSize: Int!
  maxTypeArgumentDepth: Int!
  maxTypeArgumentWidth: Int!
  maxTypeNodes: Int!
  maxMoveValueDepth: Int!
  maxTransactionIds: Int!
  maxMultiGetObjectsKeys: Int!
  maxScanLimit: Int!
}
```

Fields

`ServiceConfig.isEnabled`.`Boolean!` non-null scalar

Check whether `feature` is enabled on this GraphQL service.

`ServiceConfig.isEnabled.feature`.`Feature!` non-null enum

`ServiceConfig.enabledFeatures`.`[Feature!]!` non-null enum

List of all features that are enabled on this GraphQL service.

`ServiceConfig.maxQueryDepth`.`Int!` non-null scalar

The maximum depth a GraphQL query can be to be accepted by this service.

`ServiceConfig.maxQueryNodes`. `Int!` **non-null** **scalar**

The maximum number of nodes (field names) the service will accept in a single query.

`ServiceConfig.maxOutputNodes`. `Int!` **non-null** **scalar**

The maximum number of output nodes in a GraphQL response.

Non-connection nodes have a count of 1, while connection nodes are counted as the specified 'first' or 'last' number of items, or the default_page_size as set by the server if those arguments are not set.

Counts accumulate multiplicatively down the query tree. For example, if a query starts with a connection of first: 10 and has a field to a connection with last: 20, the count at the second level would be 200 nodes. This is then summed to the count of 10 nodes at the first level, for a total of 210 nodes.

`ServiceConfig.maxDbQueryCost`. `Int!` **non-null** **scalar**

Maximum estimated cost of a database query used to serve a GraphQL request. This is measured in the same units that the database uses in EXPLAIN queries.

`ServiceConfig.defaultPageSize`. `Int!` **non-null** **scalar**

Default number of elements allowed on a single page of a connection.

`ServiceConfig.maxPageSize`. `Int!` **non-null** **scalar**

Maximum number of elements allowed on a single page of a connection.

`ServiceConfig.mutationTimeoutMs`. `Int!` **non-null** **scalar**

Maximum time in milliseconds spent waiting for a response from fullnode after issuing a a transaction to execute. Note that the transaction may still succeed even in the case of a timeout. Transactions are idempotent, so a transaction that times out should be resubmitted until the network returns a definite response (success or failure, not timeout).

`ServiceConfig.requestTimeoutMs`. `Int!` **non-null** **scalar**

Maximum time in milliseconds that will be spent to serve one query request.

`ServiceConfig.maxTransactionPayloadSize`. `Int!` **non-null** **scalar**

The maximum bytes allowed for the `txBytes` and `signatures` fields of the GraphQL mutation `executeTransactionBlock` node, or for the `txBytes` of a `dryRunTransactionBlock`.

It is the value of the maximum transaction bytes (including the signatures) allowed by the protocol, plus the Base64 overhead (roughly 1/3 of the original string).

`ServiceConfig.maxQueryPayloadSize`.`Int!` **non-null** **scalar**

The maximum bytes allowed for the JSON object in the request body of a GraphQL query, for the read part of the query. In case of mutations or dryRunTransactionBlocks the txBytes and signatures are not included in this limit.

`ServiceConfig.maxTypeArgumentDepth`.`Int!` **non-null** **scalar**

Maximum nesting allowed in type arguments in Move Types resolved by this service.

`ServiceConfig.maxTypeArgumentWidth`.`Int!` **non-null** **scalar**

Maximum number of type arguments passed into a generic instantiation of a Move Type resolved by this service.

`ServiceConfig.maxTypeNodes`.`Int!` **non-null** **scalar**

Maximum number of structs that need to be processed when calculating the layout of a single Move Type.

`ServiceConfig.maxMoveValueDepth`.`Int!` **non-null** **scalar**

Maximum nesting allowed in struct fields when calculating the layout of a single Move Type.

`ServiceConfig.maxTransactionIds`.`Int!` **non-null** **scalar**

Maximum number of transaction ids that can be passed to a `TransactionBlockFilter`.

`ServiceConfig.maxMultiGetObjectsKeys`.`Int!` **non-null** **scalar**

Maximum number of keys that can be passed to a `multiGetObjects` query.

`ServiceConfig.maxScanLimit`.`Int!` **non-null** **scalar**

Maximum number of candidates to scan when gathering a page of results.

Returned By

`serviceConfig` **query**

Edit this page

© 2025 SUI FOUNDATION | DOCUMENTATION DISTRIBUTED UNDER CC BY 4.0

Shared

A shared object is an object that is shared using the 0x2::transfer::share_object function. Unlike owned objects, once an object is shared, it stays mutable and is accessible by anyone.

```
type Shared {
  initialSharedVersion: UInt53!
}
```

Fields

`Shared.initialSharedVersion`: `UInt53!` **non-null** **scalar**

Implemented By

`ObjectOwner` **union**

Edit this page

© 2025 SUI FOUNDATION | DOCUMENTATION DISTRIBUTED UNDER CC BY 4.0

SharedInput

A Move object that's shared.

```
type SharedInput {
  address: SuiAddress!
  initialSharedVersion: UInt53!
  mutable: Boolean!
}
```

Fields

`SharedInput.address`.`SuiAddress!` **non-null** **scalar**

`SharedInput.initialSharedVersion`.`UInt53!` **non-null** **scalar**

The version that this this object was shared at.

`SharedInput.mutable`.`Boolean!` **non-null** **scalar**

Controls whether the transaction block can reference the shared object as a mutable reference or by value. This has implications for scheduling: Transactions that just read shared objects at a certain version (mutable = false) can be executed concurrently, while transactions that write shared objects (mutable = true) must be executed serially with respect to each other.

Implemented By

`TransactionInput` **union**

Edit this page

© 2025 SUI FOUNDATION | DOCUMENTATION DISTRIBUTED UNDER CC BY 4.0

SharedObjectCancelled

The transaction accpeted a shared object as input, but its execution was cancelled.

```
type SharedObjectCancelled {
  address: SuiAddress!
  version: UInt53!
}
```

Fields

`SharedObjectCancelled.address` . `SuiAddress!` **non-null** **scalar**

ID of the shared object.

`SharedObjectCancelled.version` . `UInt53!` **non-null** **scalar**

The assigned shared object version. It is a special version indicating transaction cancellation reason.

Implemented By

`UnchangedSharedObject` **union**

Edit this page

© 2025 SUI FOUNDATION | DOCUMENTATION DISTRIBUTED UNDER CC BY 4.0

SharedObjectDelete

The transaction accepted a shared object as input, but it was deleted before the transaction executed.

```
type SharedObjectDelete {
  address: SuiAddress!
  version: UInt53!
  mutable: Boolean!
}
```

Fields

`SharedObjectDelete.address.SuiAddress!` **non-null** **scalar**

ID of the shared object.

`SharedObjectDelete.version.UInt53!` **non-null** **scalar**

The version of the shared object that was assigned to this transaction during by consensus, during sequencing.

`SharedObjectDelete.mutable.Boolean!` **non-null** **scalar**

Whether this transaction intended to use this shared object mutably or not. See `SharedInput.mutable` for further details.

Implemented By

`UnchangedSharedObject` **union**

Edit this page

© 2025 SUI FOUNDATION | DOCUMENTATION DISTRIBUTED UNDER CC BY 4.0

SharedObjectRead

The transaction accepted a shared object as input, but only to read it.

```
type SharedObjectRead {
  address: SuiAddress!
  version: UInt53!
  digest: String!
  object: Object
}
```

Fields

SharedObjectRead.address: SuiAddress! non-null scalar

ID of the object being read.

SharedObjectRead.version: UInt53! non-null scalar

Version of the object being read.

SharedObjectRead.digest: String! non-null scalar

32-byte hash that identifies the object's contents at this version, encoded as a Base58 string.

SharedObjectRead.object: Object object

The object at this version. May not be available due to pruning.

Implemented By

UnchangedSharedObject union

Edit this page

© 2025 SUI FOUNDATION | DOCUMENTATION DISTRIBUTED UNDER CC BY 4.0

skip

No description

```
directive @skip(
  if: Boolean!
) on
  | FIELD
  | FRAGMENT_SPREAD
  | INLINE_FRAGMENT
```

Arguments

`skip.if`.`Boolean!` **non-null** **scalar**

Edit this page

© 2025 SUI FOUNDATION | DOCUMENTATION DISTRIBUTED UNDER CC BY 4.0

specifiedBy

Exposes a URL that specifies the behavior of this scalar.

```
directive @specifiedBy(
  url: String!
) on SCALAR
```

Arguments

`specifiedBy.url`.`String!` **non-null** **scalar**

The URL that specifies the behavior of this scalar.

Edit this page

© 2025 SUI FOUNDATION | DOCUMENTATION DISTRIBUTED UNDER CC BY 4.0

SplitCoinsTransaction

Splits off coins with denominations in `amounts` from `coin`, returning multiple results (as many as there are amounts.)

```
type SplitCoinsTransaction {
  coin: TransactionArgument!
  amounts: [TransactionArgument!]!
}
```

Fields

`SplitCoinsTransaction.coin.TransactionArgument!`

non-null **union**

The coin to split.

`SplitCoinsTransaction.amounts.[TransactionArgument!]!`

non-null **union**

The denominations to split off from the coin.

Implemented By

`ProgrammableTransaction` **union**

Edit this page

© 2025 SUI FOUNDATION | DOCUMENTATION DISTRIBUTED UNDER CC BY 4.0

StakedSui

Represents a `0x3::staking_pool::StakedSui` Move object on-chain.

```
type StakedSui implements IMoveObject, IObject, IOwner {
  address: SuiAddress!
  objects(
    first: Int
    after: String
    last: Int
    before: String
    filter: ObjectFilter
  ): MoveObjectConnection!
  balance(
    type: String
  ): Balance
  balances(
    first: Int
    after: String
    last: Int
    before: String
  ): BalanceConnection!
  coins(
    first: Int
    after: String
    last: Int
    before: String
    type: String
  ): CoinConnection!
  stakedSuis(
    first: Int
    after: String
    last: Int
    before: String
  ): StakedSuiConnection!
  defaultSuinsName(
    format: DomainFormat
  ): String
  suinsRegistrations(
    first: Int
    after: String
    last: Int
    before: String
  ): SuinsRegistrationConnection!
  version: UInt53!
  status: ObjectKind!
  digest: String
  owner: ObjectOwner
```

```
  previousTransactionBlock: TransactionBlock
  storageRebate: BigInt
  receivedTransactionBlocks(
    first: Int
    after: String
    last: Int
    before: String
    filter: TransactionBlockFilter
    scanLimit: Int
  ): TransactionBlockConnection!
  bcs: Base64
  contents: MoveValue
  hasPublicTransfer: Boolean!
  display: [DisplayEntry!]
  dynamicField(
    name: DynamicFieldName!
  ): DynamicField
  dynamicObjectField(
    name: DynamicFieldName!
  ): DynamicField
  dynamicFields(
    first: Int
    after: String
    last: Int
    before: String
  ): DynamicFieldConnection!
  stakeStatus: StakeStatus!
  activatedEpoch: Epoch
  requestedEpoch: Epoch
  poolId: SuiAddress
  principal: BigInt
  estimatedReward: BigInt
}
```

Fields

`StakedSui.address`.`SuiAddress!` **non-null** **scalar**

`StakedSui.objects`.`MoveObjectConnection!` **non-null** **object**

Objects owned by this object, optionally `filter`-ed.

`StakedSui.objects.first`.`Int` **scalar**

`StakedSui.objects.after`.`String` **scalar**

`StakedSui.objects.last`.`Int` **scalar**

`StakedSui.objects.before`.`String` **scalar**

`StakedSui.objects.filter`.`ObjectFilter` input

`StakedSui.balance`.`Balance` object

Total balance of all coins with marker type owned by this object. If type is not supplied, it defaults to `0x2::sui::SUI`.

`StakedSui.balance.type`.`String` scalar

`StakedSui.balances`.`BalanceConnection!` non-null object

The balances of all coin types owned by this object.

`StakedSui.balances.first`.`Int` scalar

`StakedSui.balances.after`.`String` scalar

`StakedSui.balances.last`.`Int` scalar

`StakedSui.balances.before`.`String` scalar

`StakedSui.coins`.`CoinConnection!` non-null object

The coin objects for this object.

`type` is a filter on the coin's type parameter, defaulting to `0x2::sui::SUI`.

`StakedSui.coins.first`.`Int` scalar

`StakedSui.coins.after`.`String` scalar

`StakedSui.coins.last`.`Int` scalar

`StakedSui.coins.before`.`String` scalar

`StakedSui.coins.type`.`String` scalar

`StakedSui.stakedSuis`.`StakedSuiConnection!` non-null object

The `0x3::staking_pool::StakedSui` objects owned by this object.

`StakedSui.stakedSuis.first`.`Int` scalar

`StakedSui.stakedSuis.after`.`String` scalar

`StakedSui.stakedSuis.last`.`Int` scalar

`StakedSui.stakedSuis.before`.`String` scalar

StakedSui.**defaultSuinsName**.String **scalar**

The domain explicitly configured as the default domain pointing to this object.

StakedSui.defaultSuinsName.**format**.DomainFormat **enum**

StakedSui.**suinsRegistrations**.SuinsRegistrationConnection! **non-null** **object**

The SuinsRegistration NFTs owned by this object. These grant the owner the capability to manage the associated domain.

StakedSui.suinsRegistrations.**first**.Int **scalar**

StakedSui.suinsRegistrations.**after**.String **scalar**

StakedSui.suinsRegistrations.**last**.Int **scalar**

StakedSui.suinsRegistrations.**before**.String **scalar**

StakedSui.**version**.UInt53! **non-null** **scalar**

StakedSui.**status**.ObjectKind! **non-null** **enum**

The current status of the object as read from the off-chain store. The possible states are: NOT_INDEXED, the object is loaded from serialized data, such as the contents of a genesis or system package upgrade transaction. LIVE, the version returned is the most recent for the object, and it is not deleted or wrapped at that version. HISTORICAL, the object was referenced at a specific version or checkpoint, so is fetched from historical tables and may not be the latest version of the object. WRAPPED_OR_DELETED, the object is deleted or wrapped and only partial information can be loaded."

StakedSui.**digest**.String **scalar**

32-byte hash that identifies the object's contents, encoded as a Base58 string.

StakedSui.**owner**.ObjectOwner **union**

The owner type of this object: Immutable, Shared, Parent, Address

StakedSui.**previousTransactionBlock**.TransactionBlock **object**

The transaction block that created this version of the object.

StakedSui.**storageRebate**.BigInt **scalar**

The amount of SUI we would rebate if this object gets deleted or mutated. This number is recalculated based on the present storage gas price.

`StakedSui.receivedTransactionBlocks`.`TransactionBlockConnection!` non-null object

The transaction blocks that sent objects to this object.

`scanLimit` restricts the number of candidate transactions scanned when gathering a page of results. It is required for queries that apply more than two complex filters (on function, kind, sender, recipient, input object, changed object, or ids), and can be at most `serviceConfig.maxScanLimit`.

When the scan limit is reached the page will be returned even if it has fewer than `first` results when paginating forward (`last` when paginating backwards). If there are more transactions to scan, `pageInfo.hasNextPage` (or `pageInfo.hasPreviousPage`) will be set to `true`, and `PageInfo.endCursor` (or `PageInfo.startCursor`) will be set to the last transaction that was scanned as opposed to the last (or first) transaction in the page.

Requesting the next (or previous) page after this cursor will resume the search, scanning the next `scanLimit` many transactions in the direction of pagination, and so on until all transactions in the scanning range have been visited.

By default, the scanning range includes all transactions known to GraphQL, but it can be restricted by the `after` and `before` cursors, and the `beforeCheckpoint`, `afterCheckpoint` and `atCheckpoint` filters.

`StakedSui.receivedTransactionBlocks.first`.`Int` scalar

`StakedSui.receivedTransactionBlocks.after`.`String` scalar

`StakedSui.receivedTransactionBlocks.last`.`Int` scalar

`StakedSui.receivedTransactionBlocks.before`.`String` scalar

`StakedSui.receivedTransactionBlocks.filter`.`TransactionBlockFilter` input

`StakedSui.receivedTransactionBlocks.scanLimit`.`Int` scalar

`StakedSui.bcs`.`Base64` scalar

The Base64-encoded BCS serialization of the object's content.

`StakedSui.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.

`StakedSui.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.

`StakedSui.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.

`StakedSui.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 Owner type.

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

`StakedSui.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.

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

`StakedSui.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.

`StakedSui.dynamicFields.first`.`Int` **scalar**

`StakedSui.dynamicFields.after`.`String` **scalar**

`StakedSui.dynamicFields.last`.`Int` **scalar**

`StakedSui.dynamicFields.before`.`String` **scalar**

`StakedSui.stakeStatus`.`StakeStatus!` **non-null** **enum**

A stake can be pending, active, or unstaked

`StakedSui.activatedEpoch`.`Epoch` **object**

The epoch at which this stake became active.

`StakedSui.requestedEpoch`.`Epoch` **object**

The epoch at which this object was requested to join a stake pool.

`StakedSui.poolId`.`SuiAddress` **scalar**

The object id of the validator staking pool this stake belongs to.

`StakedSui.principal`.`BigInt` **scalar**

The SUI that was initially staked.

`StakedSui.estimatedReward`.`BigInt` **scalar**

The estimated reward for this stake object, calculated as:

principal * (initial_stake_rate / current_stake_rate - 1.0)

Or 0, if this value is negative, where:

- `initial_stake_rate` is the stake rate at the epoch this stake was activated at.
- `current_stake_rate` is the stake rate in the current epoch.

This value is only available if the stake is active.

Interfaces

`IMoveObject` **interface**

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

`IObject` **interface**

Interface implemented by on-chain values that are addressable by an ID (also referred to as its address). This includes Move objects and packages.

`IOwner` **interface**

Interface implemented by GraphQL types representing entities that can own objects. Object owners are identified by an address which can represent either the public key of an account or another object. The same address can only refer to an account or an object, never both, but it is not possible to know which up-front.

Member Of

`MoveObject` **object** . `StakedSuiConnection` **object** . `StakedSuiEdge` **object**

Edit this page

© 2025 SUI FOUNDATION | DOCUMENTATION DISTRIBUTED UNDER CC BY 4.0

StakedSuiConnection

No description

```
type StakedSuiConnection {
  pageInfo: PageInfo!
  edges: [StakedSuiEdge!]!
  nodes: [StakedSui!]!
}
```

Fields

`StakedSuiConnection.pageInfo`.`PageInfo!` **non-null** **object**

Information to aid in pagination.

`StakedSuiConnection.edges`.`[StakedSuiEdge!]!` **non-null** **object**

A list of edges.

`StakedSuiConnection.nodes`.`[StakedSui!]!` **non-null** **object**

A list of nodes.

Member Of

`Address` **object** . `Coin` **object** . `CoinMetadata` **object** . `IOwner` **interface** . `MoveObject` **object** . `MovePackage` **object** . `Object` **object** . `Owner` **object** . `StakedSui` **object** . `SuinsRegistration` **object**

Edit this page

© 2025 SUI FOUNDATION | DOCUMENTATION DISTRIBUTED UNDER CC BY 4.0

StakedSuiEdge

An edge in a connection.

```
type StakedSuiEdge {
  node: StakedSui!
  cursor: String!
}
```

Fields

`StakedSuiEdge.node.StakedSui!` **non-null** **object**

The item at the end of the edge

`StakedSuiEdge.cursor.String!` **non-null** **scalar**

A cursor for use in pagination

Member Of

`StakedSuiConnection` **object**

Edit this page

© 2025 SUI FOUNDATION | DOCUMENTATION DISTRIBUTED UNDER CC BY 4.0

StakeStatus

The stake's possible status: active, pending, or unstaked.

```
enum StakeStatus {
  ACTIVE
  PENDING
  UNSTAKED
}
```

Values

`StakeStatus.ACTIVE`

The stake object is active in a staking pool and it is generating rewards.

`StakeStatus.PENDING`

The stake awaits to join a staking pool in the next epoch.

`StakeStatus.UNSTAKED`

The stake is no longer active in any staking pool.

Member Of

`StakedSui` **object**

Edit this page

© 2025 SUI FOUNDATION | DOCUMENTATION DISTRIBUTED UNDER CC BY 4.0

StakeSubsidy

Parameters that control the distribution of the stake subsidy.

```
type StakeSubsidy {
  balance: BigInt
  distributionCounter: Int
  currentDistributionAmount: BigInt
  periodLength: Int
  decreaseRate: Int
}
```

Fields

StakeSubsidy.balance.BigInt scalar

SUI set aside for stake subsidies -- reduces over time as stake subsidies are paid out over time.

StakeSubsidy.distributionCounter.Int scalar

Number of times stake subsidies have been distributed subsidies are distributed with other staking rewards, at the end of the epoch.

StakeSubsidy.currentDistributionAmount.BigInt scalar

Amount of stake subsidy deducted from the balance per distribution -- decays over time.

StakeSubsidy.periodLength.Int scalar

Maximum number of stake subsidy distributions that occur with the same distribution amount (before the amount is reduced).

StakeSubsidy.decreaseRate.Int scalar

Percentage of the current distribution amount to deduct at the end of the current subsidy period, expressed in basis points.

Member Of

Epoch object

Edit this page

© 2025 SUI FOUNDATION | DOCUMENTATION DISTRIBUTED UNDER CC BY 4.0

StorageFund

SUI set aside to account for objects stored on-chain.

```
type StorageFund {
  totalObjectStorageRebates: BigInt
  nonRefundableBalance: BigInt
}
```

Fields

`StorageFund.totalObjectStorageRebates.BigInt` **scalar**

Sum of storage rebates of live objects on chain.

`StorageFund.nonRefundableBalance.BigInt` **scalar**

The portion of the storage fund that will never be refunded through storage rebates.

The system maintains an invariant that the sum of all storage fees into the storage fund is equal to the sum of of all storage rebates out, the total storage rebates remaining, and the non-refundable balance.

Member Of

`Epoch` **object**

Edit this page

© 2025 SUI FOUNDATION | DOCUMENTATION DISTRIBUTED UNDER CC BY 4.0

String

The `String` scalar type represents textual data, represented as UTF-8 character sequences. The String type is most often used by GraphQL to represent free-form human-readable text.

```
scalar String
```

Returned By

`chainIdentifier` query

Member Of

`ActiveJwk` object . `ActiveJwkEdge` object . `Address` object . `AddressEdge` object . `BalanceChangeEdge` object . `BalanceEdge` object . `BridgeStateCreateTransaction` object . `Checkpoint` object . `CheckpointEdge` object . `CheckpointId` input . `checkpoints` query . `Coin` object . `CoinEdge` object . `coinMetadata` query . `CoinMetadata` object . `coins` query . `ConsensusCommitPrologueTransaction` object . `DependencyEdge` object . `deprecated` directive . `DisplayEntry` object . `DryRunResult` object . `dryRunTransactionBlock` query . `DynamicFieldEdge` object . `DynamicFieldName` input . `EndOfEpochTransactionKindEdge` object . `Epoch` object . `EpochEdge` object . `epochs` query . `EventEdge` object . `EventFilter` input . `events` query . `executeTransactionBlock` mutation . `ExecutionResult` object . `IMoveDatatype` interface . `IObject` interface . `IOwner` interface . `MoveCallTransaction` object . `MoveDatatype` object . `MoveDatatypeEdge` object . `MoveEnum` object . `MoveEnumEdge` object . `MoveEnumVariant` object . `MoveField` object . `MoveFunction` object . `MoveFunctionEdge` object . `MoveModule` object . `MoveModuleEdge` object . `MoveObject` object . `MoveObjectEdge` object . `MovePackage` object . `MovePackageEdge` object . `MoveStruct` object . `MoveStructEdge` object . `MoveType` object . `Object` object . `ObjectChangeEdge` object . `ObjectEdge` object . `ObjectFilter` input . `ObjectRef` input . `objects` query . `OpenMoveType` object . `OwnedOrImmutable` object . `Owner` object . `packageByName` query . `packages` query . `packageVersions` query . `PageInfo` object . `ProgrammableTransactionEdge` object . `ProtocolConfigAttr` object . `ProtocolConfigFeatureFlag` object . `Receiving` object . `resolveSuinsAddress` query . `SharedObjectRead` object . `specifiedBy` directive . `StakedSui` object . `StakedSuiEdge` object . `SuinsRegistration` object . `SuinsRegistrationEdge` object . `transactionBlock` query . `TransactionBlock` object . `TransactionBlockEdge` object . `TransactionBlockEffects` object . `TransactionBlockFilter` input . `transactionBlocks` query . `TransactionInputEdge` object . `type` query . `typeByName` query . `TypeOrigin`

object . UnchangedSharedObjectEdge object . Validator object . ValidatorCredentials object . ValidatorEdge object . ZkLoginVerifyResult object

Edit this page

© 2025 SUI FOUNDATION | DOCUMENTATION DISTRIBUTED UNDER CC BY 4.0

SuiAddress

String containing 32B hex-encoded address, with a leading "0x". Leading zeroes can be omitted on input but will always appear in outputs (SuiAddress in output is guaranteed to be 66 characters long).

```
scalar SuiAddress
```

Member Of

address query . Address object . Coin object . CoinMetadata object . EventFilter input . IOwner interface . latestPackage query . Linkage object . MoveCallTransaction object . MoveObject object . MovePackage object . object query . Object object . ObjectChange object . ObjectFilter input . ObjectKey input . ObjectRef input . OwnedOrImmutable object . owner query . Owner object . package query . packageVersions query . PublishTransaction object . Receiving object . SharedInput object . SharedObjectCancelled object . SharedObjectDelete object . SharedObjectRead object . StakedSui object . SuinsRegistration object . TransactionBlockFilter input . TransactionMetadata input . TypeOrigin object . UpgradeTransaction object . Validator object . ValidatorSet object . verifyZkloginSignature query

Edit this page

© 2025 SUI FOUNDATION | DOCUMENTATION DISTRIBUTED UNDER CC BY 4.0

SuinsRegistration

No description

```
type SuinsRegistration implements IMoveObject, IObject, IOwner {
  address: SuiAddress!
  objects(
    first: Int
    after: String
    last: Int
    before: String
    filter: ObjectFilter
  ): MoveObjectConnection!
  balance(
    type: String
  ): Balance
  balances(
    first: Int
    after: String
    last: Int
    before: String
  ): BalanceConnection!
  coins(
    first: Int
    after: String
    last: Int
    before: String
    type: String
  ): CoinConnection!
  stakedSuis(
    first: Int
    after: String
    last: Int
    before: String
  ): StakedSuiConnection!
  defaultSuinsName(
    format: DomainFormat
  ): String
  suinsRegistrations(
    first: Int
    after: String
    last: Int
    before: String
  ): SuinsRegistrationConnection!
  version: UInt53!
  status: ObjectKind!
  digest: String
  owner: ObjectOwner
```

```
  previousTransactionBlock: TransactionBlock
  storageRebate: BigInt
  receivedTransactionBlocks(
    first: Int
    after: String
    last: Int
    before: String
    filter: TransactionBlockFilter
    scanLimit: Int
  ): TransactionBlockConnection!
  bcs: Base64
  contents: MoveValue
  hasPublicTransfer: Boolean!
  display: [DisplayEntry!]
  dynamicField(
    name: DynamicFieldName!
  ): DynamicField
  dynamicObjectField(
    name: DynamicFieldName!
  ): DynamicField
  dynamicFields(
    first: Int
    after: String
    last: Int
    before: String
  ): DynamicFieldConnection!
  domain: String!
}
```

Fields

`SuinsRegistration.address`.`SuiAddress!` **non-null** **scalar**

`SuinsRegistration.objects`.`MoveObjectConnection!` **non-null** **object**

Objects owned by this object, optionally `filter`-ed.

`SuinsRegistration.objects.first`.`Int` **scalar**

`SuinsRegistration.objects.after`.`String` **scalar**

`SuinsRegistration.objects.last`.`Int` **scalar**

`SuinsRegistration.objects.before`.`String` **scalar**

`SuinsRegistration.objects.filter`.`ObjectFilter` **input**

`SuinsRegistration.balance`.`Balance` **object**

Total balance of all coins with marker type owned by this object. If type is not supplied, it defaults to `0x2::sui::SUI`.

`SuinsRegistration.balance.type`.`String` scalar

`SuinsRegistration.balances`.`BalanceConnection!` non-null object

The balances of all coin types owned by this object.

`SuinsRegistration.balances.first`.`Int` scalar

`SuinsRegistration.balances.after`.`String` scalar

`SuinsRegistration.balances.last`.`Int` scalar

`SuinsRegistration.balances.before`.`String` scalar

`SuinsRegistration.coins`.`CoinConnection!` non-null object

The coin objects for this object.

`type` is a filter on the coin's type parameter, defaulting to `0x2::sui::SUI`.

`SuinsRegistration.coins.first`.`Int` scalar

`SuinsRegistration.coins.after`.`String` scalar

`SuinsRegistration.coins.last`.`Int` scalar

`SuinsRegistration.coins.before`.`String` scalar

`SuinsRegistration.coins.type`.`String` scalar

`SuinsRegistration.stakedSuis`.`StakedSuiConnection!` non-null object

The `0x3::staking_pool::StakedSui` objects owned by this object.

`SuinsRegistration.stakedSuis.first`.`Int` scalar

`SuinsRegistration.stakedSuis.after`.`String` scalar

`SuinsRegistration.stakedSuis.last`.`Int` scalar

`SuinsRegistration.stakedSuis.before`.`String` scalar

`SuinsRegistration.defaultSuinsName`.`String` scalar

The domain explicitly configured as the default domain pointing to this object.

`SuinsRegistration.defaultSuinsName.format`.`DomainFormat` enum

`SuinsRegistration.suinsRegistrations`.`SuinsRegistrationConnection!` non-null object

The SuinsRegistration NFTs owned by this object. These grant the owner the capability to manage the associated domain.

`SuinsRegistration.suinsRegistrations.first`.`Int` scalar

`SuinsRegistration.suinsRegistrations.after`.`String` scalar

`SuinsRegistration.suinsRegistrations.last`.`Int` scalar

`SuinsRegistration.suinsRegistrations.before`.`String` scalar

`SuinsRegistration.version`.`UInt53!` non-null scalar

`SuinsRegistration.status`.`ObjectKind!` non-null enum

The current status of the object as read from the off-chain store. The possible states are: NOT_INDEXED, the object is loaded from serialized data, such as the contents of a genesis or system package upgrade transaction. LIVE, the version returned is the most recent for the object, and it is not deleted or wrapped at that version. HISTORICAL, the object was referenced at a specific version or checkpoint, so is fetched from historical tables and may not be the latest version of the object. WRAPPED_OR_DELETED, the object is deleted or wrapped and only partial information can be loaded."

`SuinsRegistration.digest`.`String` scalar

32-byte hash that identifies the object's contents, encoded as a Base58 string.

`SuinsRegistration.owner`.`ObjectOwner` union

The owner type of this object: Immutable, Shared, Parent, Address

`SuinsRegistration.previousTransactionBlock`.`TransactionBlock` object

The transaction block that created this version of the object.

`SuinsRegistration.storageRebate`.`BigInt` scalar

The amount of SUI we would rebate if this object gets deleted or mutated. This number is recalculated based on the present storage gas price.

SuinsRegistration.receivedTransactionBlocks.TransactionBlockConnection! non-null object

The transaction blocks that sent objects to this object.

`scanLimit` restricts the number of candidate transactions scanned when gathering a page of results. It is required for queries that apply more than two complex filters (on function, kind, sender, recipient, input object, changed object, or ids), and can be at most `serviceConfig.maxScanLimit`.

When the scan limit is reached the page will be returned even if it has fewer than `first` results when paginating forward (`last` when paginating backwards). If there are more transactions to scan, `pageInfo.hasNextPage` (or `pageInfo.hasPreviousPage`) will be set to `true`, and `PageInfo.endCursor` (or `PageInfo.startCursor`) will be set to the last transaction that was scanned as opposed to the last (or first) transaction in the page.

Requesting the next (or previous) page after this cursor will resume the search, scanning the next `scanLimit` many transactions in the direction of pagination, and so on until all transactions in the scanning range have been visited.

By default, the scanning range includes all transactions known to GraphQL, but it can be restricted by the `after` and `before` cursors, and the `beforeCheckpoint`, `afterCheckpoint` and `atCheckpoint` filters.

`SuinsRegistration.receivedTransactionBlocks.first`.`Int` scalar

`SuinsRegistration.receivedTransactionBlocks.after`.`String` scalar

`SuinsRegistration.receivedTransactionBlocks.last`.`Int` scalar

`SuinsRegistration.receivedTransactionBlocks.before`.`String` scalar

`SuinsRegistration.receivedTransactionBlocks.filter`.`TransactionBlockFilter` input

`SuinsRegistration.receivedTransactionBlocks.scanLimit`.`Int` scalar

SuinsRegistration.bcs.Base64 scalar

The Base64-encoded BCS serialization of the object's content.

SuinsRegistration.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.

SuinsRegistration.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.

`SuinsRegistration.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.

`SuinsRegistration.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 Owner type.

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

`SuinsRegistration.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.

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

`SuinsRegistration.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.

`SuinsRegistration.dynamicFields.first`.`Int` **scalar**

`SuinsRegistration.dynamicFields.after`.`String` **scalar**

`SuinsRegistration.dynamicFields.last`.`Int` **scalar**

`SuinsRegistration.dynamicFields.before`.`String` **scalar**

`SuinsRegistration.domain`.`String!` **non-null** **scalar**

Domain name of the SuinsRegistration object

Interfaces

`IMoveObject` **interface**

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

`IObject` **interface**

Interface implemented by on-chain values that are addressable by an ID (also referred to as its address). This includes Move objects and packages.

`IOwner` **interface**

Interface implemented by GraphQL types representing entities that can own objects. Object owners are identified by an address which can represent either the public key of an account or another object. The same address can only refer to an account or an object, never both, but it is not possible to know which up-front.

Member Of

`MoveObject` **object** • `SuinsRegistrationConnection` **object** • `SuinsRegistrationEdge` **object**

Edit this page

© 2025 SUI FOUNDATION | DOCUMENTATION DISTRIBUTED UNDER CC BY 4.0

SuinsRegistrationConnection

No description

```
type SuinsRegistrationConnection {
  pageInfo: PageInfo!
  edges: [SuinsRegistrationEdge!]!
  nodes: [SuinsRegistration!]!
}
```

Fields

SuinsRegistrationConnection.**pageInfo**.PageInfo! non-null object

Information to aid in pagination.

SuinsRegistrationConnection.**edges**.[SuinsRegistrationEdge!]! non-null object

A list of edges.

SuinsRegistrationConnection.**nodes**.[SuinsRegistration!]! non-null object

A list of nodes.

Member Of

Address object . Coin object . CoinMetadata object . IOwner interface . MoveObject object . MovePackage object . Object object . Owner object . StakedSui object . SuinsRegistration object

Edit this page

© 2025 SUI FOUNDATION | DOCUMENTATION DISTRIBUTED UNDER CC BY 4.0

SuinsRegistrationEdge

An edge in a connection.

```
type SuinsRegistrationEdge {
  node: SuinsRegistration!
  cursor: String!
}
```

Fields

`SuinsRegistrationEdge.node`.`SuinsRegistration!` **non-null** **object**

The item at the end of the edge

`SuinsRegistrationEdge.cursor`.`String!` **non-null** **scalar**

A cursor for use in pagination

Member Of

`SuinsRegistrationConnection` **object**

Edit this page

© 2025 SUI FOUNDATION | DOCUMENTATION DISTRIBUTED UNDER CC BY 4.0

SystemParameters

Details of the system that are decided during genesis.

```
type SystemParameters {
  durationMs: BigInt
  stakeSubsidyStartEpoch: UInt53
  minValidatorCount: Int
  maxValidatorCount: Int
  minValidatorJoiningStake: BigInt
  validatorLowStakeThreshold: BigInt
  validatorVeryLowStakeThreshold: BigInt
  validatorLowStakeGracePeriod: BigInt
}
```

Fields

`SystemParameters.durationMs.BigInt` **scalar**

Target duration of an epoch, in milliseconds.

`SystemParameters.stakeSubsidyStartEpoch.UInt53` **scalar**

The epoch at which stake subsidies start being paid out.

`SystemParameters.minValidatorCount.Int` **scalar**

The minimum number of active validators that the system supports.

`SystemParameters.maxValidatorCount.Int` **scalar**

The maximum number of active validators that the system supports.

`SystemParameters.minValidatorJoiningStake.BigInt` **scalar**

Minimum stake needed to become a new validator.

`SystemParameters.validatorLowStakeThreshold.BigInt` **scalar**

Validators with stake below this threshold will enter the grace period (see `validatorLowStakeGracePeriod`), after which they are removed from the active validator set.

`SystemParameters.validatorVeryLowStakeThreshold.BigInt` **scalar**

Validators with stake below this threshold will be removed from the active validator set at the next epoch boundary, without a grace period.

`SystemParameters.validatorLowStakeGracePeriod.BigInt` **scalar**

The number of epochs that a validator has to recover from having less than `validatorLowStakeThreshold` stake.

Member Of

`Epoch` **object**

Edit this page

© 2025 SUI FOUNDATION | DOCUMENTATION DISTRIBUTED UNDER CC BY 4.0

TransactionArgument

An argument to a programmable transaction command.

```
union TransactionArgument = GasCoin | Input | Result
```

Possible types

`TransactionArgument.GasCoin` **object**

Access to the gas inputs, after they have been smashed into one coin. The gas coin can only be used by reference, except for with `TransferObjectsTransaction` that can accept it by value.

`TransactionArgument.Input` **object**

One of the input objects or primitive values to the programmable transaction block.

`TransactionArgument.Result` **object**

The result of another transaction command.

Member Of

`DryRunMutation` **object** . `MakeMoveVecTransaction` **object** . `MergeCoinsTransaction` **object** . `MoveCallTransaction` **object** . `SplitCoinsTransaction` **object** . `TransferObjectsTransaction` **object** . `UpgradeTransaction` **object**

Edit this page

© 2025 SUI FOUNDATION | DOCUMENTATION DISTRIBUTED UNDER CC BY 4.0

TransactionBlock

No description

```
type TransactionBlock {
  digest: String
  sender: Address
  gasInput: GasInput
  kind: TransactionBlockKind
  signatures: [Base64!]
  effects: TransactionBlockEffects
  expiration: Epoch
  bcs: Base64
}
```

Fields

`TransactionBlock.digest`.`String` **scalar**

A 32-byte hash that uniquely identifies the transaction block contents, encoded in Base58. This serves as a unique id for the block on chain.

`TransactionBlock.sender`.`Address` **object**

The address corresponding to the public key that signed this transaction. System transactions do not have senders.

`TransactionBlock.gasInput`.`GasInput` **object**

The gas input field provides information on what objects were used as gas as well as the owner of the gas object(s) and information on the gas price and budget.

If the owner of the gas object(s) is not the same as the sender, the transaction block is a sponsored transaction block.

`TransactionBlock.kind`.`TransactionBlockKind` **union**

The type of this transaction as well as the commands and/or parameters comprising the transaction of this kind.

`TransactionBlock.signatures`.`[Base64!]` **list** **scalar**

A list of all signatures, Base64-encoded, from senders, and potentially the gas owner if this is a sponsored transaction.

`TransactionBlock.effects`.`TransactionBlockEffects` **object**

The effects field captures the results to the chain of executing this transaction.

`TransactionBlock.expiration`.`Epoch` **object**

This field is set by senders of a transaction block. It is an epoch reference that sets a deadline after which validators will no longer consider the transaction valid. By default, there is no deadline for when a transaction must execute.

`TransactionBlock.bcs`.`Base64` **scalar**

Serialized form of this transaction's `TransactionData`, BCS serialized and Base64 encoded.

Returned By

`transactionBlock` **query**

Member Of

`Coin` **object** . `CoinMetadata` **object** . `DependencyConnection` **object** . `DependencyEdge` **object** . `DryRunResult` **object** . `Event` **object** . `IObject` **interface** . `MoveObject` **object** . `MovePackage` **object** . `Object` **object** . `StakedSui` **object** . `SuinsRegistration` **object** . `TransactionBlockConnection` **object** . `TransactionBlockEdge` **object** . `TransactionBlockEffects` **object**

Edit this page

© 2025 SUI FOUNDATION | DOCUMENTATION DISTRIBUTED UNDER CC BY 4.0

TransactionBlockConnection

No description

```
type TransactionBlockConnection {
  pageInfo: PageInfo!
  edges: [TransactionBlockEdge!]!
  nodes: [TransactionBlock!]!
}
```

Fields

TransactionBlockConnection.pageInfo.PageInfo! non-null object

Information to aid in pagination.

TransactionBlockConnection.edges.[TransactionBlockEdge!]! non-null object

A list of edges.

TransactionBlockConnection.nodes.[TransactionBlock!]! non-null object

A list of nodes.

Returned By

transactionBlocks query

Member Of

Address object . Checkpoint object . Coin object . CoinMetadata object . Epoch object . IObject interface . MoveObject object . MovePackage object . Object object . StakedSui object . SuinsRegistration object

Edit this page

© 2025 SUI FOUNDATION | DOCUMENTATION DISTRIBUTED UNDER CC BY 4.0

TransactionBlockEdge

An edge in a connection.

```
type TransactionBlockEdge {
  node: TransactionBlock!
  cursor: String!
}
```

Fields

`TransactionBlockEdge.node`.`TransactionBlock!` non-null object

The item at the end of the edge

`TransactionBlockEdge.cursor`.`String!` non-null scalar

A cursor for use in pagination

Member Of

`TransactionBlockConnection` object

Edit this page

© 2025 SUI FOUNDATION | DOCUMENTATION DISTRIBUTED UNDER CC BY 4.0

TransactionBlockEffects

The effects representing the result of executing a transaction block.

```
type TransactionBlockEffects {
  transactionBlock: TransactionBlock
  status: ExecutionStatus
  lamportVersion: UInt53!
  errors: String
  dependencies(
    first: Int
    after: String
    last: Int
    before: String
  ): DependencyConnection!
  gasEffects: GasEffects
  unchangedSharedObjects(
    first: Int
    after: String
    last: Int
    before: String
  ): UnchangedSharedObjectConnection!
  objectChanges(
    first: Int
    after: String
    last: Int
    before: String
  ): ObjectChangeConnection!
  balanceChanges(
    first: Int
    after: String
    last: Int
    before: String
  ): BalanceChangeConnection!
  events(
    first: Int
    after: String
    last: Int
    before: String
  ): EventConnection!
  timestamp: DateTime
  epoch: Epoch
  checkpoint: Checkpoint
  bcs: Base64!
}
```

Fields

TransactionBlockEffects.transactionBlock.TransactionBlock object

The transaction that ran to produce these effects.

TransactionBlockEffects.status.ExecutionStatus enum

Whether the transaction executed successfully or not.

TransactionBlockEffects.lamportVersion.UInt53! non-null scalar

The latest version of all objects (apart from packages) that have been created or modified by this transaction, immediately following this transaction.

TransactionBlockEffects.errors.String scalar

The reason for a transaction failure, if it did fail. If the error is a Move abort, the error message will be resolved to a human-readable form if possible, otherwise it will fall back to displaying the abort code and location.

TransactionBlockEffects.dependencies.DependencyConnection! non-null object

Transactions whose outputs this transaction depends upon.

TransactionBlockEffects.dependencies.first.Int scalar

TransactionBlockEffects.dependencies.after.String scalar

TransactionBlockEffects.dependencies.last.Int scalar

TransactionBlockEffects.dependencies.before.String scalar

TransactionBlockEffects.gasEffects.GasEffects object

Effects to the gas object.

TransactionBlockEffects.unchangedSharedObjects.UnchangedSharedObjectConnection! non-null object

Shared objects that are referenced by but not changed by this transaction.

TransactionBlockEffects.unchangedSharedObjects.first.Int scalar

TransactionBlockEffects.unchangedSharedObjects.after.String scalar

`TransactionBlockEffects.unchangedSharedObjects.last`.`Int` scalar

`TransactionBlockEffects.unchangedSharedObjects.before`.`String` scalar

`TransactionBlockEffects.objectChanges`.`ObjectChangeConnection!` non-null object

The effect this transaction had on objects on-chain.

`TransactionBlockEffects.objectChanges.first`.`Int` scalar

`TransactionBlockEffects.objectChanges.after`.`String` scalar

`TransactionBlockEffects.objectChanges.last`.`Int` scalar

`TransactionBlockEffects.objectChanges.before`.`String` scalar

`TransactionBlockEffects.balanceChanges`.`BalanceChangeConnection!` non-null object

The effect this transaction had on the balances (sum of coin values per coin type) of addresses and objects.

`TransactionBlockEffects.balanceChanges.first`.`Int` scalar

`TransactionBlockEffects.balanceChanges.after`.`String` scalar

`TransactionBlockEffects.balanceChanges.last`.`Int` scalar

`TransactionBlockEffects.balanceChanges.before`.`String` scalar

`TransactionBlockEffects.events`.`EventConnection!` non-null object

Events emitted by this transaction block.

`TransactionBlockEffects.events.first`.`Int` scalar

`TransactionBlockEffects.events.after`.`String` scalar

`TransactionBlockEffects.events.last`.`Int` scalar

`TransactionBlockEffects.events.before`.`String` scalar

`TransactionBlockEffects.timestamp`.`DateTime` scalar

Timestamp corresponding to the checkpoint this transaction was finalized in.

`TransactionBlockEffects.epoch`.`Epoch` **object**

The epoch this transaction was finalized in.

`TransactionBlockEffects.checkpoint`.`Checkpoint` **object**

The checkpoint this transaction was finalized in.

`TransactionBlockEffects.bcs`.`Base64!` **non-null** **scalar**

Base64 encoded bcs serialization of the on-chain transaction effects.

Member Of

`ExecutionResult` **object** . `TransactionBlock` **object**

Edit this page

© 2025 SUI FOUNDATION | DOCUMENTATION DISTRIBUTED UNDER CC BY 4.0

TransactionBlockFilter

No description

```
input TransactionBlockFilter {
  function: String
  kind: TransactionBlockKindInput
  afterCheckpoint: UInt53
  atCheckpoint: UInt53
  beforeCheckpoint: UInt53
  affectedAddress: SuiAddress
  sentAddress: SuiAddress
  inputObject: SuiAddress
  changedObject: SuiAddress
  transactionIds: [String!]
}
```

Fields

`TransactionBlockFilter.function.String` **scalar**

Filter transactions by move function called. Calls can be filtered by the `package`, `package::module`, or the `package::module::name` of their function.

`TransactionBlockFilter.kind.TransactionBlockKindInput` **enum**

An input filter selecting for either system or programmable transactions.

`TransactionBlockFilter.afterCheckpoint.UInt53` **scalar**

Limit to transactions that occured strictly after the given checkpoint.

`TransactionBlockFilter.atCheckpoint.UInt53` **scalar**

Limit to transactions in the given checkpoint.

`TransactionBlockFilter.beforeCheckpoint.UInt53` **scalar**

Limit to transaction that occured strictly before the given checkpoint.

`TransactionBlockFilter.affectedAddress`.`SuiAddress` **scalar**

Limit to transactions that interacted with the given address. The address could be a sender, sponsor, or recipient of the transaction.

`TransactionBlockFilter.sentAddress`.`SuiAddress` **scalar**

Limit to transactions that were sent by the given address.

`TransactionBlockFilter.inputObject`.`SuiAddress` **scalar**

Limit to transactions that accepted the given object as an input. NOTE: this input filter has been deprecated in favor of `affectedObject` which offers an easier to under behavior.

This filter will be removed with 1.36.0 (2024-10-14), or at least one release after `affectedObject` is introduced, whichever is later.

`TransactionBlockFilter.changedObject`.`SuiAddress` **scalar**

Limit to transactions that output a versioon of this object. NOTE: this input filter has been deprecated in favor of `affectedObject` which offers an easier to understand behavor.

This filter will be removed with 1.36.0 (2024-10-14), or at least one release after `affectedObject` is introduced, whichever is later.

`TransactionBlockFilter.transactionIds`.`[String!]` **list** **scalar**

Select transactions by their digest.

Member Of

`transactionBlocks` **query**

Edit this page

© 2025 SUI FOUNDATION | DOCUMENTATION DISTRIBUTED UNDER CC BY 4.0

TransactionBlockKind

The kind of transaction block, either a programmable transaction or a system transaction.

```
union TransactionBlockKind = ConsensusCommitPrologueTransaction |
GenesisTransaction | ChangeEpochTransaction | ProgrammableTransactionBlock |
AuthenticatorStateUpdateTransaction | RandomnessStateUpdateTransaction |
EndOfEpochTransaction
```

Possible types

`TransactionBlockKind.ConsensusCommitPrologueTransaction` **object**

System transaction that runs at the beginning of a checkpoint, and is responsible for setting the current value of the clock, based on the timestamp from consensus.

`TransactionBlockKind.GenesisTransaction` **object**

System transaction that initializes the network and writes the initial set of objects on-chain.

`TransactionBlockKind.ChangeEpochTransaction` **object**

A system transaction that updates epoch information on-chain (increments the current epoch). Executed by the system once per epoch, without using gas. Epoch change transactions cannot be submitted by users, because validators will refuse to sign them.

This transaction kind is deprecated in favour of `EndOfEpochTransaction`.

`TransactionBlockKind.ProgrammableTransactionBlock` **object**

A user transaction that allows the interleaving of native commands (like transfer, split coins, merge coins, etc) and move calls, executed atomically.

`TransactionBlockKind.AuthenticatorStateUpdateTransaction` **object**

System transaction for updating the on-chain state used by zkLogin.

`TransactionBlockKind.RandomnessStateUpdateTransaction` object

System transaction to update the source of on-chain randomness.

`TransactionBlockKind.EndOfEpochTransaction` object

System transaction that supersedes `ChangeEpochTransaction` as the new way to run transactions at the end of an epoch. Behaves similarly to `ChangeEpochTransaction` but can accommodate other optional transactions to run at the end of the epoch.

Member Of

`TransactionBlock` object

Edit this page

© 2025 SUI FOUNDATION | DOCUMENTATION DISTRIBUTED UNDER CC BY 4.0

TransactionBlockKindInput

An input filter selecting for either system or programmable transactions.

```
enum TransactionBlockKindInput {
  SYSTEM_TX
  PROGRAMMABLE_TX
}
```

Values

`TransactionBlockKindInput.SYSTEM_TX`

A system transaction can be one of several types of transactions. See [unions/transaction-block-kind] for more details.

`TransactionBlockKindInput.PROGRAMMABLE_TX`

A user submitted transaction block.

Member Of

`TransactionBlockFilter` **input**

Edit this page

© 2025 SUI FOUNDATION | DOCUMENTATION DISTRIBUTED UNDER CC BY 4.0

transactionBlocks

The transaction blocks that exist in the network.

`scanLimit` restricts the number of candidate transactions scanned when gathering a page of results. It is required for queries that apply more than two complex filters (on function, kind, sender, recipient, input object, changed object, or ids), and can be at most `serviceConfig.maxScanLimit`.

When the scan limit is reached the page will be returned even if it has fewer than `first` results when paginating forward (`last` when paginating backwards). If there are more transactions to scan, `pageInfo.hasNextPage` (or `pageInfo.hasPreviousPage`) will be set to `true`, and `PageInfo.endCursor` (or `PageInfo.startCursor`) will be set to the last transaction that was scanned as opposed to the last (or first) transaction in the page.

Requesting the next (or previous) page after this cursor will resume the search, scanning the next `scanLimit` many transactions in the direction of pagination, and so on until all transactions in the scanning range have been visited.

By default, the scanning range includes all transactions known to GraphQL, but it can be restricted by the `after` and `before` cursors, and the `beforeCheckpoint`, `afterCheckpoint` and `atCheckpoint` filters.

```
transactionBlocks(
  first: Int
  after: String
  last: Int
  before: String
  filter: TransactionBlockFilter
  scanLimit: Int
): TransactionBlockConnection!
```

Arguments

`transactionBlocks.first`. `Int` **scalar**

`transactionBlocks.after`. `String` **scalar**

`transactionBlocks.last`. `Int` **scalar**

`transactionBlocks.before`. `String` **scalar**

`transactionBlocks.filter`.`TransactionBlockFilter` **input**

`transactionBlocks.scanLimit`.`Int` **scalar**

Type

`TransactionBlockConnection` **object**

Edit this page

© 2025 SUI FOUNDATION | DOCUMENTATION DISTRIBUTED UNDER CC BY 4.0

TransactionInput

No description

```
union TransactionInput = OwnedOrImmutable | SharedInput | Receiving | Pure
```

Possible types

`TransactionInput.OwnedOrImmutable` object

A Move object, either immutable, or owned mutable.

`TransactionInput.SharedInput` object

A Move object that's shared.

`TransactionInput.Receiving` object

A Move object that can be received in this transaction.

`TransactionInput.Pure` object

BCS encoded primitive value (not an object or Move struct).

Member Of

`TransactionInputConnection` object . `TransactionInputEdge` object

Edit this page

© 2025 SUI FOUNDATION | DOCUMENTATION DISTRIBUTED UNDER CC BY 4.0

TransactionInputConnection

No description

```
type TransactionInputConnection {
  pageInfo: PageInfo!
  edges: [TransactionInputEdge!]!
  nodes: [TransactionInput!]!
}
```

Fields

`TransactionInputConnection.pageInfo.PageInfo!` **non-null** **object**

Information to aid in pagination.

`TransactionInputConnection.edges.[TransactionInputEdge!]!` **non-null** **object**

A list of edges.

`TransactionInputConnection.nodes.[TransactionInput!]!` **non-null** **union**

A list of nodes.

Member Of

`ProgrammableTransactionBlock` **object**

Edit this page

© 2025 SUI FOUNDATION | DOCUMENTATION DISTRIBUTED UNDER CC BY 4.0

TransactionInputEdge

An edge in a connection.

```
type TransactionInputEdge {
  node: TransactionInput!
  cursor: String!
}
```

Fields

`TransactionInputEdge.node`.`TransactionInput!` **non-null** **union**

The item at the end of the edge

`TransactionInputEdge.cursor`.`String!` **non-null** **scalar**

A cursor for use in pagination

Member Of

`TransactionInputConnection` **object**

Edit this page

© 2025 SUI FOUNDATION | DOCUMENTATION DISTRIBUTED UNDER CC BY 4.0

TransactionMetadata

The optional extra data a user can provide to a transaction dry run. `sender` defaults to `0x0`. If gasObjects `is not present, or is an empty list, it is substituted with a mock Coin object,` gasPrice `defaults to the reference gas price,` gasBudget `defaults to the max gas budget and` gasSponsor` defaults to the sender.

```
input TransactionMetadata {
  sender: SuiAddress
  gasPrice: UInt53
  gasObjects: [ObjectRef!]
  gasBudget: UInt53
  gasSponsor: SuiAddress
}
```

Fields

`TransactionMetadata.sender`.`SuiAddress` **scalar**

`TransactionMetadata.gasPrice`.`UInt53` **scalar**

`TransactionMetadata.gasObjects`.`[ObjectRef!]` **list** **input**

`TransactionMetadata.gasBudget`.`UInt53` **scalar**

`TransactionMetadata.gasSponsor`.`SuiAddress` **scalar**

Member Of

`dryRunTransactionBlock` **query**

Edit this page

© 2025 SUI FOUNDATION | DOCUMENTATION DISTRIBUTED UNDER CC BY 4.0

TransferObjectsTransaction

Transfers `inputs` to `address`. All inputs must have the `store` ability (allows public transfer) and must not be previously immutable or shared.

```
type TransferObjectsTransaction {
  inputs: [TransactionArgument!]!
  address: TransactionArgument!
}
```

Fields

`TransferObjectsTransaction.inputs`.`[TransactionArgument!]!` non-null union

The objects to transfer.

`TransferObjectsTransaction.address`.`TransactionArgument!` non-null union

The address to transfer to.

Implemented By

`ProgrammableTransaction` union

Edit this page

© 2025 SUI FOUNDATION | DOCUMENTATION DISTRIBUTED UNDER CC BY 4.0

type

Fetch a structured representation of a concrete type, including its layout information. Fails if the type is malformed.

```
type(
  type: String!
): MoveType!
```

Arguments

`type.type`.`String!` **non-null** **scalar**

Type

`MoveType` **object**

Represents concrete types (no type parameters, no references).

Edit this page

© 2025 SUI FOUNDATION | DOCUMENTATION DISTRIBUTED UNDER CC BY 4.0

typeByName

Fetch a type that includes dot move service names in it.

```
typeByName(
  name: String!
): MoveType!
```

Arguments

`typeByName.name`.`String!` **non-null** **scalar**

Type

`MoveType` **object**

Represents concrete types (no type parameters, no references).

Edit this page

© 2025 SUI FOUNDATION | DOCUMENTATION DISTRIBUTED UNDER CC BY 4.0

TypeOrigin

Information about which previous versions of a package introduced its types.

```
type TypeOrigin {
  module: String!
  struct: String!
  definingId: SuiAddress!
}
```

Fields

TypeOrigin.module.String! non-null scalar

Module defining the type.

TypeOrigin.struct.String! non-null scalar

Name of the struct.

TypeOrigin.definingId.SuiAddress! non-null scalar

The storage ID of the package that first defined this type.

Member Of

MovePackage object

Edit this page

© 2025 SUI FOUNDATION | DOCUMENTATION DISTRIBUTED UNDER CC BY 4.0

UInt53

An unsigned integer that can hold values up to 2^53 - 1. This can be treated similarly to `Int`, but it is guaranteed to be non-negative, and it may be larger than 2^32 - 1.

```
scalar UInt53
```

Member Of

AuthenticatorStateExpireTransaction object · AuthenticatorStateUpdateTransaction object · Balance object · BridgeCommitteeInitTransaction object · ChangeEpochTransaction object · Checkpoint object · CheckpointId input · Coin object · CoinMetadata object · ConsensusCommitPrologueTransaction object · ConsensusV2 object · epoch query · Epoch object · IObject interface · Linkage object · MoveObject object · MovePackage object · MovePackageCheckpointFilter input · MovePackageVersionFilter input · object query · Object object · ObjectKey input · ObjectRef input · OwnedOrImmutable object · owner query · package query · protocolConfig query · ProtocolConfigs object · RandomnessStateUpdateTransaction object · Receiving object · Shared object · SharedInput object · SharedObjectCancelled object · SharedObjectDelete object · SharedObjectRead object · StakedSui object · SuinsRegistration object · SystemParameters object · TransactionBlockEffects object · TransactionBlockFilter input · TransactionMetadata input · Validator object

Edit this page

© 2025 SUI FOUNDATION | DOCUMENTATION DISTRIBUTED UNDER CC BY 4.0

UnchangedSharedObject

Details pertaining to shared objects that are referenced by but not changed by a transaction. This information is considered part of the effects, because although the transaction specifies the shared object as input, consensus must schedule it and pick the version that is actually used.

```
union UnchangedSharedObject = SharedObjectRead | SharedObjectDelete |
SharedObjectCancelled
```

Possible types

`UnchangedSharedObject.SharedObjectRead` **object**

The transaction accepted a shared object as input, but only to read it.

`UnchangedSharedObject.SharedObjectDelete` **object**

The transaction accepted a shared object as input, but it was deleted before the transaction executed.

`UnchangedSharedObject.SharedObjectCancelled` **object**

The transaction accpeted a shared object as input, but its execution was cancelled.

Member Of

`UnchangedSharedObjectConnection` **object** . `UnchangedSharedObjectEdge` **object**

Edit this page

© 2025 SUI FOUNDATION | DOCUMENTATION DISTRIBUTED UNDER CC BY 4.0

UnchangedSharedObjectConnection

No description

```
type UnchangedSharedObjectConnection {
  pageInfo: PageInfo!
  edges: [UnchangedSharedObjectEdge!]!
  nodes: [UnchangedSharedObject!]!
}
```

Fields

`UnchangedSharedObjectConnection.pageInfo.PageInfo!` non-null object

Information to aid in pagination.

`UnchangedSharedObjectConnection.edges.[UnchangedSharedObjectEdge!]!` non-null object

A list of edges.

`UnchangedSharedObjectConnection.nodes.[UnchangedSharedObject!]!` non-null union

A list of nodes.

Member Of

`TransactionBlockEffects` object

Edit this page

© 2025 SUI FOUNDATION | DOCUMENTATION DISTRIBUTED UNDER CC BY 4.0

UnchangedSharedObjectEdge

An edge in a connection.

```
type UnchangedSharedObjectEdge {
  node: UnchangedSharedObject!
  cursor: String!
}
```

Fields

`UnchangedSharedObjectEdge.node` . `UnchangedSharedObject!`

non-null **union**

The item at the end of the edge

`UnchangedSharedObjectEdge.cursor` . `String!` **non-null** **scalar**

A cursor for use in pagination

Member Of

`UnchangedSharedObjectConnection` **object**

Edit this page

© 2025 SUI FOUNDATION | DOCUMENTATION DISTRIBUTED UNDER CC BY 4.0

UpgradeTransaction

Upgrades a Move Package.

```
type UpgradeTransaction {
  modules: [Base64!]!
  dependencies: [SuiAddress!]!
  currentPackage: SuiAddress!
  upgradeTicket: TransactionArgument!
}
```

Fields

`UpgradeTransaction.modules.[Base64!]!` **non-null** **scalar**

Bytecode for the modules to be published, BCS serialized and Base64 encoded.

`UpgradeTransaction.dependencies.[SuiAddress!]!` **non-null** **scalar**

IDs of the transitive dependencies of the package to be published.

`UpgradeTransaction.currentPackage.SuiAddress!` **non-null** **scalar**

ID of the package being upgraded.

`UpgradeTransaction.upgradeTicket.TransactionArgument!` **non-null** **union**

The `UpgradeTicket` authorizing the upgrade.

Implemented By

`ProgrammableTransaction` **union**

Edit this page

© 2025 SUI FOUNDATION | DOCUMENTATION DISTRIBUTED UNDER CC BY 4.0

Validator

No description

```
type Validator {
  address: Address!
  credentials: ValidatorCredentials
  nextEpochCredentials: ValidatorCredentials
  name: String
  description: String
  imageUrl: String
  projectUrl: String
  operationCap: MoveObject
  stakingPool: MoveObject @deprecated
  stakingPoolId: SuiAddress!
  exchangeRates: MoveObject @deprecated
  exchangeRatesTable: Owner
  exchangeRatesSize: UInt53
  stakingPoolActivationEpoch: UInt53
  stakingPoolSuiBalance: BigInt
  rewardsPool: BigInt
  poolTokenBalance: BigInt
  pendingStake: BigInt
  pendingTotalSuiWithdraw: BigInt
  pendingPoolTokenWithdraw: BigInt
  votingPower: Int
  gasPrice: BigInt
  commissionRate: Int
  nextEpochStake: BigInt
  nextEpochGasPrice: BigInt
  nextEpochCommissionRate: Int
  atRisk: UInt53
  reportRecords(
    first: Int
    before: String
    last: Int
    after: String
  ): AddressConnection!
  apy: Int
}
```

Fields

`Validator.address.Address!` non-null object

The validator's address.

`Validator.credentials`.`ValidatorCredentials` **object**

Validator's set of credentials such as public keys, network addresses and others.

`Validator.nextEpochCredentials`.`ValidatorCredentials` **object**

Validator's set of credentials for the next epoch.

`Validator.name`.`String` **scalar**

Validator's name.

`Validator.description`.`String` **scalar**

Validator's description.

`Validator.imageUrl`.`String` **scalar**

Validator's url containing their custom image.

`Validator.projectUrl`.`String` **scalar**

Validator's homepage URL.

`Validator.operationCap`.`MoveObject` **object**

The validator's current valid `Cap` object. Validators can delegate the operation ability to another address. The address holding this `Cap` object can then update the reference gas price and tallying rule on behalf of the validator.

`Validator.stakingPool`.`MoveObject` **deprecated** **object**

> ⚠ **DEPRECATED**
>
> The staking pool is a wrapped object. Access its fields directly on the `Validator` type.

The validator's current staking pool object, used to track the amount of stake and to compound staking rewards.

`Validator.stakingPoolId`.`SuiAddress!` **non-null** **scalar**

The ID of this validator's `0x3::staking_pool::StakingPool`.

`Validator.exchangeRates`.`MoveObject` **deprecated** **object**

> **DEPRECATED**
>
> The exchange object is a wrapped object. Access its dynamic fields through the `exchangeRatesTable` query.

The validator's current exchange object. The exchange rate is used to determine the amount of SUI tokens that each past SUI staker can withdraw in the future.

`Validator.exchangeRatesTable.Owner` **object**

A wrapped object containing the validator's exchange rates. This is a table from epoch number to `PoolTokenExchangeRate` value. The exchange rate is used to determine the amount of SUI tokens that each past SUI staker can withdraw in the future.

`Validator.exchangeRatesSize.UInt53` **scalar**

Number of exchange rates in the table.

`Validator.stakingPoolActivationEpoch.UInt53` **scalar**

The epoch at which this pool became active.

`Validator.stakingPoolSuiBalance.BigInt` **scalar**

The total number of SUI tokens in this pool.

`Validator.rewardsPool.BigInt` **scalar**

The epoch stake rewards will be added here at the end of each epoch.

`Validator.poolTokenBalance.BigInt` **scalar**

Total number of pool tokens issued by the pool.

`Validator.pendingStake.BigInt` **scalar**

Pending stake amount for this epoch.

`Validator.pendingTotalSuiWithdraw.BigInt` **scalar**

Pending stake withdrawn during the current epoch, emptied at epoch boundaries.

`Validator.pendingPoolTokenWithdraw.BigInt` **scalar**

Pending pool token withdrawn during the current epoch, emptied at epoch boundaries.

`Validator.votingPower.Int` **scalar**

The voting power of this validator in basis points (e.g., 100 = 1% voting power).

`Validator.gasPrice`.`BigInt` **scalar**

The reference gas price for this epoch.

`Validator.commissionRate`.`Int` **scalar**

The fee charged by the validator for staking services.

`Validator.nextEpochStake`.`BigInt` **scalar**

The total number of SUI tokens in this pool plus the pending stake amount for this epoch.

`Validator.nextEpochGasPrice`.`BigInt` **scalar**

The validator's gas price quote for the next epoch.

`Validator.nextEpochCommissionRate`.`Int` **scalar**

The proposed next epoch fee for the validator's staking services.

`Validator.atRisk`.`UInt53` **scalar**

The number of epochs for which this validator has been below the low stake threshold.

`Validator.reportRecords`.`AddressConnection!` **non-null** **object**

The addresses of other validators this validator has reported.

`Validator.reportRecords.first`.`Int` **scalar**

`Validator.reportRecords.before`.`String` **scalar**

`Validator.reportRecords.last`.`Int` **scalar**

`Validator.reportRecords.after`.`String` **scalar**

`Validator.apy`.`Int` **scalar**

The APY of this validator in basis points. To get the APY in percentage, divide by 100.

Member Of

`ValidatorConnection` **object** . `ValidatorEdge` **object**

Edit this page

© 2025 SUI FOUNDATION | DOCUMENTATION DISTRIBUTED UNDER CC BY 4.0

ValidatorConnection

No description

```
type ValidatorConnection {
  pageInfo: PageInfo!
  edges: [ValidatorEdge!]!
  nodes: [Validator!]!
}
```

Fields

`ValidatorConnection.pageInfo.PageInfo!` **non-null** **object**

Information to aid in pagination.

`ValidatorConnection.edges.[ValidatorEdge!]!` **non-null** **object**

A list of edges.

`ValidatorConnection.nodes.[Validator!]!` **non-null** **object**

A list of nodes.

Member Of

`ValidatorSet` **object**

Edit this page

© 2025 SUI FOUNDATION | DOCUMENTATION DISTRIBUTED UNDER CC BY 4.0

ValidatorCredentials

The credentials related fields associated with a validator.

```
type ValidatorCredentials {
  protocolPubKey: Base64
  networkPubKey: Base64
  workerPubKey: Base64
  proofOfPossession: Base64
  netAddress: String
  p2PAddress: String
  primaryAddress: String
  workerAddress: String
}
```

Fields

`ValidatorCredentials.protocolPubKey.Base64` **scalar**

`ValidatorCredentials.networkPubKey.Base64` **scalar**

`ValidatorCredentials.workerPubKey.Base64` **scalar**

`ValidatorCredentials.proofOfPossession.Base64` **scalar**

`ValidatorCredentials.netAddress.String` **scalar**

`ValidatorCredentials.p2PAddress.String` **scalar**

`ValidatorCredentials.primaryAddress.String` **scalar**

`ValidatorCredentials.workerAddress.String` **scalar**

Member Of

`Validator` **object**

Edit this page

© 2025 SUI FOUNDATION | DOCUMENTATION DISTRIBUTED UNDER CC BY 4.0

ValidatorEdge

An edge in a connection.

```
type ValidatorEdge {
  node: Validator!
  cursor: String!
}
```

Fields

`ValidatorEdge.node`.`Validator!` **non-null** **object**

The item at the end of the edge

`ValidatorEdge.cursor`.`String!` **non-null** **scalar**

A cursor for use in pagination

Member Of

`ValidatorConnection` **object**

Edit this page

© 2025 SUI FOUNDATION | DOCUMENTATION DISTRIBUTED UNDER CC BY 4.0

ValidatorSet

Representation of `0x3::validator_set::ValidatorSet`.

```
type ValidatorSet {
  totalStake: BigInt
  pendingRemovals: [Int!]
  pendingActiveValidatorsId: SuiAddress
  pendingActiveValidatorsSize: Int
  stakingPoolMappingsId: SuiAddress
  stakingPoolMappingsSize: Int
  inactivePoolsId: SuiAddress
  inactivePoolsSize: Int
  validatorCandidatesId: SuiAddress
  validatorCandidatesSize: Int
  activeValidators(
    first: Int
    before: String
    last: Int
    after: String
  ): ValidatorConnection!
}
```

Fields

`ValidatorSet.totalStake.BigInt` scalar

Total amount of stake for all active validators at the beginning of the epoch.

`ValidatorSet.pendingRemovals.[Int!]` list scalar

Validators that are pending removal from the active validator set, expressed as indices in to `activeValidators`.

`ValidatorSet.pendingActiveValidatorsId.SuiAddress` scalar

Object ID of the wrapped object `TableVec` storing the pending active validators.

`ValidatorSet.pendingActiveValidatorsSize.Int` scalar

Size of the pending active validators table.

`ValidatorSet.stakingPoolMappingsId.SuiAddress` scalar

Object ID of the `Table` storing the mapping from staking pool ids to the addresses of the corresponding validators. This is needed because a validator's address can potentially change but the object ID of its pool will not.

`ValidatorSet.stakingPoolMappingsSize.Int` scalar

Size of the stake pool mappings `Table`.

`ValidatorSet.inactivePoolsId.SuiAddress` scalar

Object ID of the `Table` storing the inactive staking pools.

`ValidatorSet.inactivePoolsSize.Int` scalar

Size of the inactive pools `Table`.

`ValidatorSet.validatorCandidatesId.SuiAddress` scalar

Object ID of the `Table` storing the validator candidates.

`ValidatorSet.validatorCandidatesSize.Int` scalar

Size of the validator candidates `Table`.

`ValidatorSet.activeValidators.ValidatorConnection!` non-null object

The current set of active validators.

`ValidatorSet.activeValidators.first.Int` scalar

`ValidatorSet.activeValidators.before.String` scalar

`ValidatorSet.activeValidators.last.Int` scalar

`ValidatorSet.activeValidators.after.String` scalar

Member Of

`Epoch` object

Edit this page

© 2025 SUI FOUNDATION | DOCUMENTATION DISTRIBUTED UNDER CC BY 4.0

verifyZkloginSignature

Verify a zkLogin signature based on the provided transaction or personal message based on current epoch, chain id, and latest JWKs fetched on-chain. If the signature is valid, the function returns a `ZkLoginVerifyResult` with success as true and an empty list of errors. If the signature is invalid, the function returns a `ZkLoginVerifyResult` with success as false with a list of errors.

- `bytes` is either the personal message in raw bytes or transaction data bytes in BCS-encoded and then Base64-encoded.
- `signature` is a serialized zkLogin signature that is Base64-encoded.
- `intentScope` is an enum that specifies the intent scope to be used to parse bytes.
- `author` is the address of the signer of the transaction or personal msg.

```
verifyZkloginSignature(
  bytes: Base64!
  signature: Base64!
  intentScope: ZkLoginIntentScope!
  author: SuiAddress!
): ZkLoginVerifyResult!
```

Arguments

`verifyZkloginSignature.bytes.Base64!` **non-null** **scalar**

`verifyZkloginSignature.signature.Base64!` **non-null** **scalar**

`verifyZkloginSignature.intentScope.ZkLoginIntentScope!` **non-null** **enum**

`verifyZkloginSignature.author.SuiAddress!` **non-null** **scalar**

Type

`ZkLoginVerifyResult` **object**

The result of the zkLogin signature verification.

Edit this page

© 2025 SUI FOUNDATION | DOCUMENTATION DISTRIBUTED UNDER CC BY 4.0

ZkLoginIntentScope

An enum that specifies the intent scope to be used to parse the bytes for signature verification.

```
enum ZkLoginIntentScope {
  TRANSACTION_DATA
  PERSONAL_MESSAGE
}
```

Values

`ZkLoginIntentScope.TRANSACTION_DATA`

Indicates that the bytes are to be parsed as transaction data bytes.

`ZkLoginIntentScope.PERSONAL_MESSAGE`

Indicates that the bytes are to be parsed as a personal message.

Member Of

`verifyZkloginSignature` **query**

Edit this page

© 2025 SUI FOUNDATION | DOCUMENTATION DISTRIBUTED UNDER CC BY 4.0

ZkLoginVerifyResult

The result of the zkLogin signature verification.

```
type ZkLoginVerifyResult {
  success: Boolean!
  errors: [String!]!
}
```

Fields

`ZkLoginVerifyResult.success`.`Boolean!` **non-null** **scalar**

The boolean result of the verification. If true, errors should be empty.

`ZkLoginVerifyResult.errors`.`[String!]!` **non-null** **scalar**

The errors field captures any verification error

Returned By

`verifyZkloginSignature` **query**

Edit this page

© 2025 SUI FOUNDATION | DOCUMENTATION DISTRIBUTED UNDER CC BY 4.0