Sui Kiosk

Kiosk is a decentralized system for commerce applications on Sui. It consists of Kiosk objects - shared objects owned by individual parties that store assets and allow listing them for sale as well as utilize custom trading functionality - for example, an auction. While being highly decentralized, Sui Kiosk provides a set of strong guarantees:

Practically, Kiosk is a part of the Sui framework, and it is native to the system and available to everyone out of the box.

See the Kiosk SDK documentation for examples of working with Kiosk using TypeScript.

Anyone can create a Sui Kiosk. Ownership of a kiosk is determined by the owner of the KioskOwnerCap , a special object that grants full access to a single kiosk. As the owner, you can sell any asset with a type (T) that has a shared TransferPolicy available, or you can use a kiosk to store assets even without a shared policy. You can't sell or transfer any assets from your kiosk that do not have an associated transfer policy available.

To sell an item, if there is an existing transfer policy for the type (T), you just add your assets to your kiosk and then list them. You specify an offer amount when you list an item. Anyone can then purchase the item for the amount of SUI specified in the listing. The associated transfer policy determines what the buyer can do with the purchased asset.

A kiosk owner can:

A buyer is a party that purchases (or - more generally - receives) items from kiosks, anyone on the network can be a buyer (and, for example, a kiosk owner at the same time).

Benefits:

Responsibilities:

Guarantees:

As a marketplace operator, you can implement Sui Kiosk to watch for offers made in a collection of kiosks and display them on a marketplace site. You can also implement a custom system using Kiosk extensions (created by the community or third-parties). For example, marketplaces can use a TransferPolicyCap to implement application-specific transfer rules.

As a creator, Sui Kiosk supports strong enforcement for transfer policies and associated rules to protect assets and enforce asset ownership. Sui Kiosk gives creators more control over their creations, and puts creators and owners in control of how their works can be used.

Creator is a party that creates and controls the TransferPolicy for a single type. For example, the authors of SuiFrens are the Creators of the SuiFren type and act as creators in the Kiosk ecosystem. Creators set the policy, but they might also be the first sellers of their assets through a kiosk.

Creators can:

All of the above is effective immediately and globally.

Creators cannot:

Sui Kiosk provides a set of guarantees that Sui enforces through smart contracts. These guarantees include:

In practice, these guarantees mean that:

Sui Kiosk is a shared object that can store heterogeneous values, such as different sets of asset collectibles. When you add an asset to your kiosk, it has one of the following states:

When someone purchases an asset from a kiosk, the asset leaves the kiosk and ownership transfers to the buyer's address.

To use a Sui Kiosk, you must create one and have the KioskOwnerCap that matches the Kiosk object. You can create a new kiosk using a single transaction by calling the kiosk::default function. The function creates and shares a Kiosk , and transfers the KioskOwnerCap to your address.

Beginning with the Sui v1.24.1 release , the --gas-budget option is no longer required for CLI commands.

For more advanced use cases, when you want to choose the storage model or perform an action right away, you can use the

programmable transaction block (PTB) friendly function kiosk::new . Kiosk is designed to be shared. If you choose a different storage model, such as owned, your kiosk might not function as intended or not be accessible to other users. You can make sure your kiosk works by testing it on Sui Testnet.

Sui CLI does not support PTBs and transaction chaining yet. You can use the kiosk::default function instead.

As a kiosk owner, you can place any assets into your Sui Kiosk. You can take any item from your kiosk that is not currently listed for sale.

There's no limitations on which assets you can place in your kiosk. However, you can't necessarily list and trade all of the items you place in your kiosk. The TransferPolicy associated with the type for the item determines whether you can trade it. To learn more, see the Purchase items from a kiosk section.

To place an item to the kiosk, the owner needs to call the sui::kiosk::place function on the Kiosk object and pass the KioskOwnerCap and the Item as arguments.

ITEM_TYPE in the following examples represents the full type of the item.

To take an item from a kiosk you must be the kiosk owner. As the owner, call the sui::kiosk::take function on the Kiosk object, and pass the KioskOwnerCap and ID of the item as arguments.

ITEM_TYPE in the following examples represents the full type of the item.

The kiosk::take function is built to be PTB friendly and returns the asset. The Sui CLI does not yet support transaction chaining.

Some policies require that assets never get removed from a kiosk, such as for strong royalty enforcement. To support this, Sui Kiosk provides a locking mechanism. Locking is similar to placing except that you can't take a locked asset out of the kiosk.

To lock an asset in a kiosk, call the sui::kiosk::lock function. To ensure that you can later unlock the asset you must associate a TransferPolicy with the asset.

After you lock an asset, you must use list or list_with_purchase_cap functions to list it.

When you use the lock function, similar to using the place function, you specify the KioskOwnerCap and the Item as arguments. But to lock the item, you must also show the TransferPolicy .

in the following examples represents the full type of the asset.

Sui Kiosk provides basic trading functionality. As a kiosk owner, you can list assets for sale, and buyers can discover and purchase them. Sui Kiosk supports listing items by default with three primary functions:

Anyone on the network can purchase an item listed from a Sui Kiosk. To learn more about the purchase flow, see the Purchase section . To learn more about asset states and what can be done with a listed item, see the Asset States section.

As a kiosk owner, you can use the kiosk::list function to list any asset you added to your kiosk. Include the item to sell and the list price as arguments. All listings on Sui are in SUI tokens. When you list an item, Sui emits a kiosk::ItemListed event that contains the kiosk ID, item ID, type of the item, and the list price.

As a kiosk owner you can use the kiosk::delist to delist any currently listed asset. Specify the item to delist as an argument.

When you delist an item, Sui returns to the kiosk owner the gas fees charged to list the item.

When you delist an item, Sui emits a kiosk::ItemDelisted event that contains the kiosk ID, item ID, and the type of the item.

Anyone that has an address on the Sui network can purchase an item listed from a Sui Kiosk. To purchase an item, you can use the kiosk::purchase function. Specify the item to purchase and pay the list price set by the kiosk owner.

You can discover the items listed on the network with the kiosk::ItemListed event.

When you use the kiosk::purchase function, it returns the purchased asset and the TransferRequest for the type associated with the asset. To complete the purchase, you must meet the terms defined in the TransferPolicy applied to the asset.

As a kiosk owner, you can access an asset placed or locked in a kiosk without taking the asset from the kiosk. You can always borrow the asset immutably. Whether you can mutably borrow an asset depends on the state of the asset. For example, you can't borrow a listed asset because you can't modify it while listed. The functions available include:

You can always borrow an asset from a kiosk immutably. You can use the kiosk::borrow function to borrow an asset, however, it is not possible to use references within a programmable transaction block. To access the asset you must use a published module (function).

You can mutably borrow an asset from a kiosk if it is not listed. You can use the kiosk::borrow_mut function to mutably borrow an asset. However, it is not possible to use references within a PTB, so to access the mutably borrowed asset you must use a published module (function).

You can use the PTB-friendly kiosk::borrow_val function. It allows you to take an asset and place it back in the same transaction. To make sure the asset is placed back into the kiosk, the function "obliges" the caller with a “Hot Potato”.

When someone purchases an item, Sui stores the proceeds from the sale in the kiosk. As the kiosk owner, you can withdraw the proceeds at any time by calling the kiosk::withdraw function. The function is simple, but because it is PTB friendly it is not currently supported in the Sui CLI.

This action is not currently supported in the CLI environment.

Sui Kiosk owners

Anyone can create a Sui Kiosk. Ownership of a kiosk is determined by the owner of the KioskOwnerCap , a special object that grants full access to a single kiosk. As the owner, you can sell any asset with a type (T) that has a shared TransferPolicy available, or you can use a kiosk to store assets even without a shared policy. You can’t sell or transfer any assets from your kiosk that do not have an associated transfer policy available.

To sell an item, if there is an existing transfer policy for the type (T), you just add your assets to your kiosk and then list them. You specify an offer amount when you list an item. Anyone can then purchase the item for the amount of SUI specified in the listing. The associated transfer policy determines what the buyer can do with the purchased asset.

A kiosk owner can:

A buyer is a party that purchases (or - more generally - receives) items from kiosks, anyone on the network can be a buyer (and, for example, a kiosk owner at the same time).

Benefits:

Responsibilities:

Guarantees:

As a marketplace operator, you can implement Sui Kiosk to watch for offers made in a collection of kiosks and display them on a marketplace site. You can also implement a custom system using Kiosk extensions (created by the community or third-parties). For example, marketplaces can use a TransferPolicyCap to implement application-specific transfer rules.

As a creator, Sui Kiosk supports strong enforcement for transfer policies and associated rules to protect assets and enforce asset ownership. Sui Kiosk gives creators more control over their creations, and puts creators and owners in control of how their works can be used.

Creator is a party that creates and controls the TransferPolicy for a single type. For example, the authors of SuiFrens are the Creators of the SuiFren type and act as creators in the Kiosk ecosystem. Creators set the policy, but they might also be the first sellers of their assets through a kiosk.

Creators can:

All of the above is effective immediately and globally.

Creators cannot:

Sui Kiosk provides a set of guarantees that Sui enforces through smart contracts. These guarantees include:

In practice, these guarantees mean that:

Sui Kiosk is a shared object that can store heterogeneous values, such as different sets of asset collectibles. When you add an asset to your kiosk, it has one of the following states:

When someone purchases an asset from a kiosk, the asset leaves the kiosk and ownership transfers to the buyer's address.

To use a Sui Kiosk, you must create one and have the KioskOwnerCap that matches the Kiosk object. You can create a new kiosk using a single transaction by calling the kiosk::default function. The function creates and shares a Kiosk , and transfers the KioskOwnerCap to your address.

Beginning with the Sui v1.24.1 release , the --gas-budget option is no longer required for CLI commands.

For more advanced use cases, when you want to choose the storage model or perform an action right away, you can use the programmable transaction block (PTB) friendly function kiosk::new . Kiosk is designed to be shared. If you choose a different storage model, such as owned, your kiosk might not function as intended or not be accessible to other users. You can make sure your kiosk works by testing it on Sui Testnet.

Sui CLI does not support PTBs and transaction chaining yet. You can use the kiosk::default function instead.

As a kiosk owner, you can place any assets into your Sui Kiosk. You can take any item from your kiosk that is not currently listed for sale.

There's no limitations on which assets you can place in your kiosk. However, you can't necessarily list and trade all of the items you place in your kiosk. The TransferPolicy associated with the type for the item determines whether you can trade it. To learn more, see the Purchase items from a kiosk section.

To place an item to the kiosk, the owner needs to call the sui::kiosk::place function on the Kiosk object and pass the KioskOwnerCap and the Item as arguments.

ITEM_TYPE in the following examples represents the full type of the item.

To take an item from a kiosk you must be the kiosk owner. As the owner, call the sui::kiosk::take function on the Kiosk object, and pass the KioskOwnerCap and ID of the item as arguments.

ITEM_TYPE in the following examples represents the full type of the item.

The kiosk::take function is built to be PTB friendly and returns the asset. The Sui CLI does not yet support transaction chaining.

Some policies require that assets never get removed from a kiosk, such as for strong royalty enforcement. To support this, Sui Kiosk provides a locking mechanism. Locking is similar to placing except that you can't take a locked asset out of the kiosk.

To lock an asset in a kiosk, call the sui::kiosk::lock function. To ensure that you can later unlock the asset you must associate a TransferPolicy with the asset.

After you lock an asset, you must use list or list_with_purchase_cap functions to list it.

When you use the lock function, similar to using the place function, you specify the KioskOwnerCap and the Item as arguments. But to lock the item, you must also show the TransferPolicy .

in the following examples represents the full type of the asset.

Sui Kiosk provides basic trading functionality. As a kiosk owner, you can list assets for sale, and buyers can discover and purchase them. Sui Kiosk supports listing items by default with three primary functions:

Anyone on the network can purchase an item listed from a Sui Kiosk. To learn more about the purchase flow, see the Purchase section . To learn more about asset states and what can be done with a listed item, see the Asset States section.

As a kiosk owner, you can use the kiosk::list function to list any asset you added to your kiosk. Include the item to sell and the list price as arguments. All listings on Sui are in SUI tokens. When you list an item, Sui emits a kiosk::ItemListed event that contains the kiosk ID, item ID, type of the item, and the list price.

As a kiosk owner you can use the kiosk::delist to delist any currently listed asset. Specify the item to delist as an argument.

When you delist an item, Sui returns to the kiosk owner the gas fees charged to list the item.

When you delist an item, Sui emits a kiosk::ItemDelisted event that contains the kiosk ID, item ID, and the type of the item.

Anyone that has an address on the Sui network can purchase an item listed from a Sui Kiosk. To purchase an item, you can use the kiosk::purchase function. Specify the item to purchase and pay the list price set by the kiosk owner.

You can discover the items listed on the network with the kiosk::ItemListed event.

When you use the kiosk::purchase function, it returns the purchased asset and the TransferRequest for the type associated with the asset. To complete the purchase, you must meet the terms defined in the TransferPolicy applied to the asset.

As a kiosk owner, you can access an asset placed or locked in a kiosk without taking the asset from the kiosk. You can always borrow the asset immutably. Whether you can mutably borrow an asset depends on the state of the asset. For example, you can't borrow a listed asset because you can't modify it while listed. The functions available include:

You can always borrow an asset from a kiosk immutably. You can use the kiosk::borrow function to borrow an asset, however, it is not possible to use references within a programmable transaction block. To access the asset you must use a published module (function).

You can mutably borrow an asset from a kiosk if it is not listed. You can use the kiosk::borrow_mut function to mutably borrow an asset. However, it is not possible to use references within a PTB, so to access the mutably borrowed asset you must use a published module (function).

You can use the PTB-friendly kiosk::borrow_val function. It allows you to take an asset and place it back in the same transaction. To make sure the asset is placed back into the kiosk, the function "obliges" the caller with a "Hot Potato".

When someone purchases an item, Sui stores the proceeds from the sale in the kiosk. As the kiosk owner, you can withdraw the proceeds at any time by calling the kiosk::withdraw function. The function is simple, but because it is PTB friendly it is not currently supported in the Sui CLI.

This action is not currently supported in the CLI environment.

Sui Kiosk for buyers

A buyer is a party that purchases (or - more generally - receives) items from kiosks, anyone on the network can be a buyer (and, for example, a kiosk owner at the same time).

Benefits:

Responsibilities:

Guarantees:

As a marketplace operator, you can implement Sui Kiosk to watch for offers made in a collection of kiosks and display them on a marketplace site. You can also implement a custom system using Kiosk extensions (created by the community or third-parties). For example, marketplaces can use a TransferPolicyCap to implement application-specific transfer rules.

As a creator, Sui Kiosk supports strong enforcement for transfer policies and associated rules to protect assets and enforce asset ownership. Sui Kiosk gives creators more control over their creations, and puts creators and owners in control of how their works can be used.

Creator is a party that creates and controls the TransferPolicy for a single type. For example, the authors of SuiFrens are the Creators of the SuiFren type and act as creators in the Kiosk ecosystem. Creators set the policy, but they might also be the first sellers of their assets through a kiosk.

Creators can:

All of the above is effective immediately and globally.

Creators cannot:

Sui Kiosk provides a set of guarantees that Sui enforces through smart contracts. These guarantees include:

In practice, these guarantees mean that:

Sui Kiosk is a shared object that can store heterogeneous values, such as different sets of asset collectibles. When you add an asset to your kiosk, it has one of the following states:

When someone purchases an asset from a kiosk, the asset leaves the kiosk and ownership transfers to the buyer's address.

To use a Sui Kiosk, you must create one and have the KioskOwnerCap that matches the Kiosk object. You can create a new kiosk

using a single transaction by calling the kiosk::default function. The function creates and shares a Kiosk , and transfers the KioskOwnerCap to your address.

Beginning with the Sui v1.24.1 release , the --gas-budget option is no longer required for CLI commands.

For more advanced use cases, when you want to choose the storage model or perform an action right away, you can use the programmable transaction block (PTB) friendly function kiosk::new . Kiosk is designed to be shared. If you choose a different storage model, such as owned, your kiosk might not function as intended or not be accessible to other users. You can make sure your kiosk works by testing it on Sui Testnet.

Sui CLI does not support PTBs and transaction chaining yet. You can use the kiosk::default function instead.

As a kiosk owner, you can place any assets into your Sui Kiosk. You can take any item from your kiosk that is not currently listed for sale.

There's no limitations on which assets you can place in your kiosk. However, you can't necessarily list and trade all of the items you place in your kiosk. The TransferPolicy associated with the type for the item determines whether you can trade it. To learn more, see the Purchase items from a kiosk section.

To place an item to the kiosk, the owner needs to call the sui::kiosk::place function on the Kiosk object and pass the KioskOwnerCap and the Item as arguments.

ITEM_TYPE in the following examples represents the full type of the item.

To take an item from a kiosk you must be the kiosk owner. As the owner, call the sui::kiosk::take function on the Kiosk object, and pass the KioskOwnerCap and ID of the item as arguments.

ITEM_TYPE in the following examples represents the full type of the item.

The kiosk::take function is built to be PTB friendly and returns the asset. The Sui CLI does not yet support transaction chaining.

Some policies require that assets never get removed from a kiosk, such as for strong royalty enforcement. To support this, Sui Kiosk provides a locking mechanism. Locking is similar to placing except that you can't take a locked asset out of the kiosk.

To lock an asset in a kiosk, call the sui::kiosk::lock function. To ensure that you can later unlock the asset you must associate a TransferPolicy with the asset.

After you lock an asset, you must use list or list_with_purchase_cap functions to list it.

When you use the lock function, similar to using the place function, you specify the KioskOwnerCap and the Item as arguments. But to lock the item, you must also show the TransferPolicy .

in the following examples represents the full type of the asset.

Sui Kiosk provides basic trading functionality. As a kiosk owner, you can list assets for sale, and buyers can discover and purchase them. Sui Kiosk supports listing items by default with three primary functions:

Anyone on the network can purchase an item listed from a Sui Kiosk. To learn more about the purchase flow, see the Purchase section . To learn more about asset states and what can be done with a listed item, see the Asset States section.

As a kiosk owner, you can use the kiosk::list function to list any asset you added to your kiosk. Include the item to sell and the list price as arguments. All listings on Sui are in SUI tokens. When you list an item, Sui emits a kiosk::ItemListed event that contains the kiosk ID, item ID, type of the item, and the list price.

As a kiosk owner you can use the kiosk::delist to delist any currently listed asset. Specify the item to delist as an argument.

When you delist an item, Sui returns to the kiosk owner the gas fees charged to list the item.

When you delist an item, Sui emits a kiosk::ItemDelisted event that contains the kiosk ID, item ID, and the type of the item.

Anyone that has an address on the Sui network can purchase an item listed from a Sui Kiosk. To purchase an item, you can use the kiosk::purchase function. Specify the item to purchase and pay the list price set by the kiosk owner.

You can discover the items listed on the network with the kiosk::ItemListed event.

When you use the kiosk::purchase function, it returns the purchased asset and the TransferRequest for the type associated with the

asset. To complete the purchase, you must meet the terms defined in the TransferPolicy applied to the asset.

As a kiosk owner, you can access an asset placed or locked in a kiosk without taking the asset from the kiosk. You can always borrow the asset immutably. Whether you can mutably borrow an asset depends on the state of the asset. For example, you can't borrow a listed asset because you can't modify it while listed. The functions available include:

You can always borrow an asset from a kiosk immutably. You can use the kiosk::borrow function to borrow an asset, however, it is not possible to use references within a programmable transaction block. To access the asset you must use a published module (function).

You can mutably borrow an asset from a kiosk if it is not listed. You can use the kiosk::borrow_mut function to mutably borrow an asset. However, it is not possible to use references within a PTB, so to access the mutably borrowed asset you must use a published module (function).

You can use the PTB-friendly kiosk::borrow_val function. It allows you to take an asset and place it back in the same transaction. To make sure the asset is placed back into the kiosk, the function "obliges" the caller with a "Hot Potato".

When someone purchases an item, Sui stores the proceeds from the sale in the kiosk. As the kiosk owner, you can withdraw the proceeds at any time by calling the kiosk::withdraw function. The function is simple, but because it is PTB friendly it is not currently supported in the Sui CLI.

This action is not currently supported in the CLI environment.

Sui Kiosk for marketplaces

As a marketplace operator, you can implement Sui Kiosk to watch for offers made in a collection of kiosks and display them on a marketplace site. You can also implement a custom system using Kiosk extensions (created by the community or third-parties). For example, marketplaces can use a TransferPolicyCap to implement application-specific transfer rules.

As a creator, Sui Kiosk supports strong enforcement for transfer policies and associated rules to protect assets and enforce asset ownership. Sui Kiosk gives creators more control over their creations, and puts creators and owners in control of how their works can be used.

Creator is a party that creates and controls the TransferPolicy for a single type. For example, the authors of SuiFrens are the Creators of the SuiFren type and act as creators in the Kiosk ecosystem. Creators set the policy, but they might also be the first sellers of their assets through a kiosk.

Creators can:

All of the above is effective immediately and globally.

Creators cannot:

Sui Kiosk provides a set of guarantees that Sui enforces through smart contracts. These guarantees include:

In practice, these guarantees mean that:

Sui Kiosk is a shared object that can store heterogeneous values, such as different sets of asset collectibles. When you add an asset to your kiosk, it has one of the following states:

When someone purchases an asset from a kiosk, the asset leaves the kiosk and ownership transfers to the buyer's address.

To use a Sui Kiosk, you must create one and have the KioskOwnerCap that matches the Kiosk object. You can create a new kiosk using a single transaction by calling the kiosk::default function. The function creates and shares a Kiosk , and transfers the KioskOwnerCap to your address.

Beginning with the Sui v1.24.1 release , the --gas-budget option is no longer required for CLI commands.

For more advanced use cases, when you want to choose the storage model or perform an action right away, you can use the programmable transaction block (PTB) friendly function kiosk::new . Kiosk is designed to be shared. If you choose a different storage model, such as owned, your kiosk might not function as intended or not be accessible to other users. You can make sure your kiosk works by testing it on Sui Testnet.

Sui CLI does not support PTBs and transaction chaining yet. You can use the kiosk::default function instead.

As a kiosk owner, you can place any assets into your Sui Kiosk. You can take any item from your kiosk that is not currently listed for sale.

There's no limitations on which assets you can place in your kiosk. However, you can't necessarily list and trade all of the items you place in your kiosk. The TransferPolicy associated with the type for the item determines whether you can trade it. To learn more, see the Purchase items from a kiosk section.

To place an item to the kiosk, the owner needs to call the sui::kiosk::place function on the Kiosk object and pass the KioskOwnerCap and the Item as arguments.

ITEM_TYPE in the following examples represents the full type of the item.

To take an item from a kiosk you must be the kiosk owner. As the owner, call the sui::kiosk::take function on the Kiosk object, and pass the KioskOwnerCap and ID of the item as arguments.

ITEM_TYPE in the following examples represents the full type of the item.

The kiosk::take function is built to be PTB friendly and returns the asset. The Sui CLI does not yet support transaction chaining.

Some policies require that assets never get removed from a kiosk, such as for strong royalty enforcement. To support this, Sui Kiosk provides a locking mechanism. Locking is similar to placing except that you can't take a locked asset out of the kiosk.

To lock an asset in a kiosk, call the sui::kiosk::lock function. To ensure that you can later unlock the asset you must associate a TransferPolicy with the asset.

After you lock an asset, you must use list or list_with_purchase_cap functions to list it.

When you use the lock function, similar to using the place function, you specify the KioskOwnerCap and the Item as arguments. But to lock the item, you must also show the TransferPolicy .

in the following examples represents the full type of the asset.

Sui Kiosk provides basic trading functionality. As a kiosk owner, you can list assets for sale, and buyers can discover and purchase them. Sui Kiosk supports listing items by default with three primary functions:

Anyone on the network can purchase an item listed from a Sui Kiosk. To learn more about the purchase flow, see the Purchase section . To learn more about asset states and what can be done with a listed item, see the Asset States section.

As a kiosk owner, you can use the kiosk::list function to list any asset you added to your kiosk. Include the item to sell and the list price as arguments. All listings on Sui are in SUI tokens. When you list an item, Sui emits a kiosk::ItemListed event that contains the kiosk ID, item ID, type of the item, and the list price.

As a kiosk owner you can use the kiosk::delist to delist any currently listed asset. Specify the item to delist as an argument.

When you delist an item, Sui returns to the kiosk owner the gas fees charged to list the item.

When you delist an item, Sui emits a kiosk::ItemDelisted event that contains the kiosk ID, item ID, and the type of the item.

Anyone that has an address on the Sui network can purchase an item listed from a Sui Kiosk. To purchase an item, you can use the kiosk::purchase function. Specify the item to purchase and pay the list price set by the kiosk owner.

You can discover the items listed on the network with the kiosk::ItemListed event.

When you use the kiosk::purchase function, it returns the purchased asset and the TransferRequest for the type associated with the asset. To complete the purchase, you must meet the terms defined in the TransferPolicy applied to the asset.

As a kiosk owner, you can access an asset placed or locked in a kiosk without taking the asset from the kiosk. You can always borrow the asset immutably. Whether you can mutably borrow an asset depends on the state of the asset. For example, you can't borrow a listed asset because you can't modify it while listed. The functions available include:

You can always borrow an asset from a kiosk immutably. You can use the kiosk::borrow function to borrow an asset, however, it is not possible to use references within a programmable transaction block. To access the asset you must use a published module (function).

You can mutably borrow an asset from a kiosk if it is not listed. You can use the kiosk::borrow_mut function to mutably borrow an

asset. However, it is not possible to use references within a PTB, so to access the mutably borrowed asset you must use a published module (function).

You can use the PTB-friendly kiosk::borrow_val function. It allows you to take an asset and place it back in the same transaction. To make sure the asset is placed back into the kiosk, the function "obliges" the caller with a “Hot Potato”.

When someone purchases an item, Sui stores the proceeds from the sale in the kiosk. As the kiosk owner, you can withdraw the proceeds at any time by calling the kiosk::withdraw function. The function is simple, but because it is PTB friendly it is not currently supported in the Sui CLI.

This action is not currently supported in the CLI environment.

Sui Kiosk for creators

As a creator, Sui Kiosk supports strong enforcement for transfer policies and associated rules to protect assets and enforce asset ownership. Sui Kiosk gives creators more control over their creations, and puts creators and owners in control of how their works can be used.

Creator is a party that creates and controls the TransferPolicy for a single type. For example, the authors of SuiFrens are the Creators of the SuiFren type and act as creators in the Kiosk ecosystem. Creators set the policy, but they might also be the first sellers of their assets through a kiosk.

Creators can:

All of the above is effective immediately and globally.

Creators cannot:

Sui Kiosk provides a set of guarantees that Sui enforces through smart contracts. These guarantees include:

In practice, these guarantees mean that:

Sui Kiosk is a shared object that can store heterogeneous values, such as different sets of asset collectibles. When you add an asset to your kiosk, it has one of the following states:

When someone purchases an asset from a kiosk, the asset leaves the kiosk and ownership transfers to the buyer’s address.

To use a Sui Kiosk, you must create one and have the KioskOwnerCap that matches the Kiosk object. You can create a new kiosk using a single transaction by calling the kiosk::default function. The function creates and shares a Kiosk , and transfers the KioskOwnerCap to your address.

Beginning with the Sui v1.24.1 release , the --gas-budget option is no longer required for CLI commands.

For more advanced use cases, when you want to choose the storage model or perform an action right away, you can use the programmable transaction block (PTB) friendly function kiosk::new . Kiosk is designed to be shared. If you choose a different storage model, such as owned, your kiosk might not function as intended or not be accessible to other users. You can make sure your kiosk works by testing it on Sui Testnet.

Sui CLI does not support PTBs and transaction chaining yet. You can use the kiosk::default function instead.

As a kiosk owner, you can place any assets into your Sui Kiosk. You can take any item from your kiosk that is not currently listed for sale.

There's no limitations on which assets you can place in your kiosk. However, you can’t necessarily list and trade all of the items you place in your kiosk. The TransferPolicy associated with the type for the item determines whether you can trade it. To learn more, see the Purchase items from a kiosk section.

To place an item to the kiosk, the owner needs to call the sui::kiosk::place function on the Kiosk object and pass the KioskOwnerCap and the Item as arguments.

ITEM_TYPE in the following examples represents the full type of the item.

To take an item from a kiosk you must be the kiosk owner. As the owner, call the sui::kiosk::take function on the Kiosk object, and pass the KioskOwnerCap and ID of the item as arguments.

ITEM_TYPE in the following examples represents the full type of the item.

The kiosk::take function is built to be PTB friendly and returns the asset. The Sui CLI does not yet support transaction chaining.

Some policies require that assets never get removed from a kiosk, such as for strong royalty enforcement. To support this, Sui Kiosk provides a locking mechanism. Locking is similar to placing except that you can't take a locked asset out of the kiosk.

To lock an asset in a kiosk, call the sui::kiosk::lock function. To ensure that you can later unlock the asset you must associate a TransferPolicy with the asset.

After you lock an asset, you must use list or list_with_purchase_cap functions to list it.

When you use the lock function, similar to using the place function, you specify the KioskOwnerCap and the Item as arguments. But to lock the item, you must also show the TransferPolicy .

in the following examples represents the full type of the asset.

Sui Kiosk provides basic trading functionality. As a kiosk owner, you can list assets for sale, and buyers can discover and purchase them. Sui Kiosk supports listing items by default with three primary functions:

Anyone on the network can purchase an item listed from a Sui Kiosk. To learn more about the purchase flow, see the Purchase section . To learn more about asset states and what can be done with a listed item, see the Asset States section.

As a kiosk owner, you can use the kiosk::list function to list any asset you added to your kiosk. Include the item to sell and the list price as arguments. All listings on Sui are in SUI tokens. When you list an item, Sui emits a kiosk::ItemListed event that contains the kiosk ID, item ID, type of the item, and the list price.

As a kiosk owner you can use the kiosk::delist to delist any currently listed asset. Specify the item to delist as an argument.

When you delist an item, Sui returns to the kiosk owner the gas fees charged to list the item.

When you delist an item, Sui emits a kiosk::ItemDelisted event that contains the kiosk ID, item ID, and the type of the item.

Anyone that has an address on the Sui network can purchase an item listed from a Sui Kiosk. To purchase an item, you can use the kiosk::purchase function. Specify the item to purchase and pay the list price set by the kiosk owner.

You can discover the items listed on the network with the kiosk::ItemListed event.

When you use the kiosk::purchase function, it returns the purchased asset and the TransferRequest for the type associated with the asset. To complete the purchase, you must meet the terms defined in the TransferPolicy applied to the asset.

As a kiosk owner, you can access an asset placed or locked in a kiosk without taking the asset from the kiosk. You can always borrow the asset immutably. Whether you can mutably borrow an asset depends on the state of the asset. For example, you can’t borrow a listed asset because you can’t modify it while listed. The functions available include:

You can always borrow an asset from a kiosk immutably. You can use the kiosk::borrow function to borrow an asset, however, it is not possible to use references within a programmable transaction block. To access the asset you must use a published module (function).

You can mutably borrow an asset from a kiosk if it is not listed. You can use the kiosk::borrow_mut function to mutably borrow an asset. However, it is not possible to use references within a PTB, so to access the mutably borrowed asset you must use a published module (function).

You can use the PTB-friendly kiosk::borrow_val function. It allows you to take an asset and place it back in the same transaction. To make sure the asset is placed back into the kiosk, the function "obliges" the caller with a “Hot Potato”.

When someone purchases an item, Sui stores the proceeds from the sale in the kiosk. As the kiosk owner, you can withdraw the proceeds at any time by calling the kiosk::withdraw function. The function is simple, but because it is PTB friendly it is not currently supported in the Sui CLI.

This action is not currently supported in the CLI environment.

Sui Kiosk guarantees

Sui Kiosk provides a set of guarantees that Sui enforces through smart contracts. These guarantees include:

In practice, these guarantees mean that:

Sui Kiosk is a shared object that can store heterogeneous values, such as different sets of asset collectibles. When you add an asset to your kiosk, it has one of the following states:

When someone purchases an asset from a kiosk, the asset leaves the kiosk and ownership transfers to the buyer's address.

To use a Sui Kiosk, you must create one and have the KioskOwnerCap that matches the Kiosk object. You can create a new kiosk using a single transaction by calling the kiosk::default function. The function creates and shares a Kiosk , and transfers the KioskOwnerCap to your address.

Beginning with the Sui v1.24.1 release , the --gas-budget option is no longer required for CLI commands.

For more advanced use cases, when you want to choose the storage model or perform an action right away, you can use the programmable transaction block (PTB) friendly function kiosk::new . Kiosk is designed to be shared. If you choose a different storage model, such as owned, your kiosk might not function as intended or not be accessible to other users. You can make sure your kiosk works by testing it on Sui Testnet.

Sui CLI does not support PTBs and transaction chaining yet. You can use the kiosk::default function instead.

As a kiosk owner, you can place any assets into your Sui Kiosk. You can take any item from your kiosk that is not currently listed for sale.

There's no limitations on which assets you can place in your kiosk. However, you can't necessarily list and trade all of the items you place in your kiosk. The TransferPolicy associated with the type for the item determines whether you can trade it. To learn more, see the Purchase items from a kiosk section.

To place an item to the kiosk, the owner needs to call the sui::kiosk::place function on the Kiosk object and pass the KioskOwnerCap and the Item as arguments.

ITEM_TYPE in the following examples represents the full type of the item.

To take an item from a kiosk you must be the kiosk owner. As the owner, call the sui::kiosk::take function on the Kiosk object, and pass the KioskOwnerCap and ID of the item as arguments.

ITEM_TYPE in the following examples represents the full type of the item.

The kiosk::take function is built to be PTB friendly and returns the asset. The Sui CLI does not yet support transaction chaining.

Some policies require that assets never get removed from a kiosk, such as for strong royalty enforcement. To support this, Sui Kiosk provides a locking mechanism. Locking is similar to placing except that you can't take a locked asset out of the kiosk.

To lock an asset in a kiosk, call the sui::kiosk::lock function. To ensure that you can later unlock the asset you must associate a TransferPolicy with the asset.

After you lock an asset, you must use list or list_with_purchase_cap functions to list it.

When you use the lock function, similar to using the place function, you specify the KioskOwnerCap and the Item as arguments. But to lock the item, you must also show the TransferPolicy .

in the following examples represents the full type of the asset.

Sui Kiosk provides basic trading functionality. As a kiosk owner, you can list assets for sale, and buyers can discover and purchase them. Sui Kiosk supports listing items by default with three primary functions:

Anyone on the network can purchase an item listed from a Sui Kiosk. To learn more about the purchase flow, see the Purchase section . To learn more about asset states and what can be done with a listed item, see the Asset States section.

As a kiosk owner, you can use the kiosk::list function to list any asset you added to your kiosk. Include the item to sell and the list price as arguments. All listings on Sui are in SUI tokens. When you list an item, Sui emits a kiosk::ItemListed event that contains the kiosk ID, item ID, type of the item, and the list price.

As a kiosk owner you can use the kiosk::delist to delist any currently listed asset. Specify the item to delist as an argument.

When you delist an item, Sui returns to the kiosk owner the gas fees charged to list the item.

When you delist an item, Sui emits a kiosk::ItemDelisted event that contains the kiosk ID, item ID, and the type of the item.

Anyone that has an address on the Sui network can purchase an item listed from a Sui Kiosk. To purchase an item, you can use the kiosk::purchase function. Specify the item to purchase and pay the list price set by the kiosk owner.

You can discover the items listed on the network with the kiosk::ItemListed event.

When you use the kiosk::purchase function, it returns the purchased asset and the TransferRequest for the type associated with the asset. To complete the purchase, you must meet the terms defined in the TransferPolicy applied to the asset.

As a kiosk owner, you can access an asset placed or locked in a kiosk without taking the asset from the kiosk. You can always borrow the asset immutably. Whether you can mutably borrow an asset depends on the state of the asset. For example, you can't borrow a listed asset because you can't modify it while listed. The functions available include:

You can always borrow an asset from a kiosk immutably. You can use the kiosk::borrow function to borrow an asset, however, it is not possible to use references within a programmable transaction block. To access the asset you must use a published module (function).

You can mutably borrow an asset from a kiosk if it is not listed. You can use the kiosk::borrow_mut function to mutably borrow an asset. However, it is not possible to use references within a PTB, so to access the mutably borrowed asset you must use a published module (function).

You can use the PTB-friendly kiosk::borrow_val function. It allows you to take an asset and place it back in the same transaction. To make sure the asset is placed back into the kiosk, the function "obliges" the caller with a "Hot Potato".

When someone purchases an item, Sui stores the proceeds from the sale in the kiosk. As the kiosk owner, you can withdraw the proceeds at any time by calling the kiosk::withdraw function. The function is simple, but because it is PTB friendly it is not currently supported in the Sui CLI.

This action is not currently supported in the CLI environment.

Open a Sui Kiosk

To use a Sui Kiosk, you must create one and have the KioskOwnerCap that matches the Kiosk object. You can create a new kiosk using a single transaction by calling the kiosk::default function. The function creates and shares a Kiosk , and transfers the KioskOwnerCap to your address.

Beginning with the Sui v1.24.1 release , the --gas-budget option is no longer required for CLI commands.

For more advanced use cases, when you want to choose the storage model or perform an action right away, you can use the programmable transaction block (PTB) friendly function kiosk::new . Kiosk is designed to be shared. If you choose a different storage model, such as owned, your kiosk might not function as intended or not be accessible to other users. You can make sure your kiosk works by testing it on Sui Testnet.

Sui CLI does not support PTBs and transaction chaining yet. You can use the kiosk::default function instead.

As a kiosk owner, you can place any assets into your Sui Kiosk. You can take any item from your kiosk that is not currently listed for sale.

There's no limitations on which assets you can place in your kiosk. However, you can't necessarily list and trade all of the items you place in your kiosk. The TransferPolicy associated with the type for the item determines whether you can trade it. To learn more, see the Purchase items from a kiosk section.

To place an item to the kiosk, the owner needs to call the sui::kiosk::place function on the Kiosk object and pass the KioskOwnerCap and the Item as arguments.

ITEM_TYPE in the following examples represents the full type of the item.

To take an item from a kiosk you must be the kiosk owner. As the owner, call the sui::kiosk::take function on the Kiosk object, and pass the KioskOwnerCap and ID of the item as arguments.

ITEM_TYPE in the following examples represents the full type of the item.

The kiosk::take function is built to be PTB friendly and returns the asset. The Sui CLI does not yet support transaction chaining.

Some policies require that assets never get removed from a kiosk, such as for strong royalty enforcement. To support this, Sui Kiosk provides a locking mechanism. Locking is similar to placing except that you can't take a locked asset out of the kiosk.

To lock an asset in a kiosk, call the sui::kiosk::lock function. To ensure that you can later unlock the asset you must associate a TransferPolicy with the asset.

After you lock an asset, you must use list or list_with_purchase_cap functions to list it.

When you use the lock function, similar to using the place function, you specify the KioskOwnerCap and the Item as arguments. But to lock the item, you must also show the TransferPolicy .

in the following examples represents the full type of the asset.

Sui Kiosk provides basic trading functionality. As a kiosk owner, you can list assets for sale, and buyers can discover and purchase them. Sui Kiosk supports listing items by default with three primary functions:

Anyone on the network can purchase an item listed from a Sui Kiosk. To learn more about the purchase flow, see the Purchase section . To learn more about asset states and what can be done with a listed item, see the Asset States section.

As a kiosk owner, you can use the kiosk::list function to list any asset you added to your kiosk. Include the item to sell and the list price as arguments. All listings on Sui are in SUI tokens. When you list an item, Sui emits a kiosk::ItemListed event that contains the kiosk ID, item ID, type of the item, and the list price.

As a kiosk owner you can use the kiosk::delist to delist any currently listed asset. Specify the item to delist as an argument.

When you delist an item, Sui returns to the kiosk owner the gas fees charged to list the item.

When you delist an item, Sui emits a kiosk::ItemDelisted event that contains the kiosk ID, item ID, and the type of the item.

Anyone that has an address on the Sui network can purchase an item listed from a Sui Kiosk. To purchase an item, you can use the kiosk::purchase function. Specify the item to purchase and pay the list price set by the kiosk owner.

You can discover the items listed on the network with the kiosk::ItemListed event.

When you use the kiosk::purchase function, it returns the purchased asset and the TransferRequest for the type associated with the asset. To complete the purchase, you must meet the terms defined in the TransferPolicy applied to the asset.

As a kiosk owner, you can access an asset placed or locked in a kiosk without taking the asset from the kiosk. You can always borrow the asset immutably. Whether you can mutably borrow an asset depends on the state of the asset. For example, you can’t borrow a listed asset because you can’t modify it while listed. The functions available include:

You can always borrow an asset from a kiosk immutably. You can use the kiosk::borrow function to borrow an asset, however, it is not possible to use references within a programmable transaction block. To access the asset you must use a published module (function).

You can mutably borrow an asset from a kiosk if it is not listed. You can use the kiosk::borrow_mut function to mutably borrow an asset. However, it is not possible to use references within a PTB, so to access the mutably borrowed asset you must use a published module (function).

You can use the PTB-friendly kiosk::borrow_val function. It allows you to take an asset and place it back in the same transaction. To make sure the asset is placed back into the kiosk, the function "obliges" the caller with a “Hot Potato”.

When someone purchases an item, Sui stores the proceeds from the sale in the kiosk. As the kiosk owner, you can withdraw the proceeds at any time by calling the kiosk::withdraw function. The function is simple, but because it is PTB friendly it is not currently supported in the Sui CLI.

This action is not currently supported in the CLI environment.

Place items in and take items from your kiosk

As a kiosk owner, you can place any assets into your Sui Kiosk. You can take any item from your kiosk that is not currently listed for sale.

There's no limitations on which assets you can place in your kiosk. However, you can’t necessarily list and trade all of the items you

place in your kiosk. The TransferPolicy associated with the type for the item determines whether you can trade it. To learn more, see the Purchase items from a kiosk section.

To place an item to the kiosk, the owner needs to call the sui::kiosk::place function on the Kiosk object and pass the KioskOwnerCap and the Item as arguments.

ITEM_TYPE in the following examples represents the full type of the item.

To take an item from a kiosk you must be the kiosk owner. As the owner, call the sui::kiosk::take function on the Kiosk object, and pass the KioskOwnerCap and ID of the item as arguments.

ITEM_TYPE in the following examples represents the full type of the item.

The kiosk::take function is built to be PTB friendly and returns the asset. The Sui CLI does not yet support transaction chaining.

Some policies require that assets never get removed from a kiosk, such as for strong royalty enforcement. To support this, Sui Kiosk provides a locking mechanism. Locking is similar to placing except that you can't take a locked asset out of the kiosk.

To lock an asset in a kiosk, call the sui::kiosk::lock function. To ensure that you can later unlock the asset you must associate a TransferPolicy with the asset.

After you lock an asset, you must use list or list_with_purchase_cap functions to list it.

When you use the lock function, similar to using the place function, you specify the KioskOwnerCap and the Item as arguments. But to lock the item, you must also show the TransferPolicy .

in the following examples represents the full type of the asset.

Sui Kiosk provides basic trading functionality. As a kiosk owner, you can list assets for sale, and buyers can discover and purchase them. Sui Kiosk supports listing items by default with three primary functions:

Anyone on the network can purchase an item listed from a Sui Kiosk. To learn more about the purchase flow, see the Purchase section . To learn more about asset states and what can be done with a listed item, see the Asset States section.

As a kiosk owner, you can use the kiosk::list function to list any asset you added to your kiosk. Include the item to sell and the list price as arguments. All listings on Sui are in SUI tokens. When you list an item, Sui emits a kiosk::ItemListed event that contains the kiosk ID, item ID, type of the item, and the list price.

As a kiosk owner you can use the kiosk::delist to delist any currently listed asset. Specify the item to delist as an argument.

When you delist an item, Sui returns to the kiosk owner the gas fees charged to list the item.

When you delist an item, Sui emits a kiosk::ItemDelisted event that contains the kiosk ID, item ID, and the type of the item.

Anyone that has an address on the Sui network can purchase an item listed from a Sui Kiosk. To purchase an item, you can use the kiosk::purchase function. Specify the item to purchase and pay the list price set by the kiosk owner.

You can discover the items listed on the network with the kiosk::ItemListed event.

When you use the kiosk::purchase function, it returns the purchased asset and the TransferRequest for the type associated with the asset. To complete the purchase, you must meet the terms defined in the TransferPolicy applied to the asset.

As a kiosk owner, you can access an asset placed or locked in a kiosk without taking the asset from the kiosk. You can always borrow the asset immutably. Whether you can mutably borrow an asset depends on the state of the asset. For example, you can’t borrow a listed asset because you can’t modify it while listed. The functions available include:

You can always borrow an asset from a kiosk immutably. You can use the kiosk::borrow function to borrow an asset, however, it is not possible to use references within a programmable transaction block. To access the asset you must use a published module (function).

You can mutably borrow an asset from a kiosk if it is not listed. You can use the kiosk::borrow_mut function to mutably borrow an asset. However, it is not possible to use references within a PTB, so to access the mutably borrowed asset you must use a published module (function).

You can use the PTB-friendly kiosk::borrow_val function. It allows you to take an asset and place it back in the same transaction.

To make sure the asset is placed back into the kiosk, the function "obliges" the caller with a “Hot Potato”.

When someone purchases an item, Sui stores the proceeds from the sale in the kiosk. As the kiosk owner, you can withdraw the proceeds at any time by calling the kiosk::withdraw function. The function is simple, but because it is PTB friendly it is not currently supported in the Sui CLI.

This action is not currently supported in the CLI environment.

Take items from a kiosk

To take an item from a kiosk you must be the kiosk owner. As the owner, call the sui::kiosk::take function on the Kiosk object, and pass the KioskOwnerCap and ID of the item as arguments.

ITEM_TYPE in the following examples represents the full type of the item.

The kiosk::take function is built to be PTB friendly and returns the asset. The Sui CLI does not yet support transaction chaining.

Some policies require that assets never get removed from a kiosk, such as for strong royalty enforcement. To support this, Sui Kiosk provides a locking mechanism. Locking is similar to placing except that you can't take a locked asset out of the kiosk.

To lock an asset in a kiosk, call the sui::kiosk::lock function. To ensure that you can later unlock the asset you must associate a TransferPolicy with the asset.

After you lock an asset, you must use list or list_with_purchase_cap functions to list it.

When you use the lock function, similar to using the place function, you specify the KioskOwnerCap and the Item as arguments. But to lock the item, you must also show the TransferPolicy .

in the following examples represents the full type of the asset.

Sui Kiosk provides basic trading functionality. As a kiosk owner, you can list assets for sale, and buyers can discover and purchase them. Sui Kiosk supports listing items by default with three primary functions:

Anyone on the network can purchase an item listed from a Sui Kiosk. To learn more about the purchase flow, see the Purchase section . To learn more about asset states and what can be done with a listed item, see the Asset States section.

As a kiosk owner, you can use the kiosk::list function to list any asset you added to your kiosk. Include the item to sell and the list price as arguments. All listings on Sui are in SUI tokens. When you list an item, Sui emits a kiosk::ItemListed event that contains the kiosk ID, item ID, type of the item, and the list price.

As a kiosk owner you can use the kiosk::delist to delist any currently listed asset. Specify the item to delist as an argument.

When you delist an item, Sui returns to the kiosk owner the gas fees charged to list the item.

When you delist an item, Sui emits a kiosk::ItemDelisted event that contains the kiosk ID, item ID, and the type of the item.

Anyone that has an address on the Sui network can purchase an item listed from a Sui Kiosk. To purchase an item, you can use the kiosk::purchase function. Specify the item to purchase and pay the list price set by the kiosk owner.

You can discover the items listed on the network with the kiosk::ItemListed event.

When you use the kiosk::purchase function, it returns the purchased asset and the TransferRequest for the type associated with the asset. To complete the purchase, you must meet the terms defined in the TransferPolicy applied to the asset.

As a kiosk owner, you can access an asset placed or locked in a kiosk without taking the asset from the kiosk. You can always borrow the asset immutably. Whether you can mutably borrow an asset depends on the state of the asset. For example, you can’t borrow a listed asset because you can’t modify it while listed. The functions available include:

You can always borrow an asset from a kiosk immutably. You can use the kiosk::borrow function to borrow an asset, however, it is not possible to use references within a programmable transaction block. To access the asset you must use a published module (function).

You can mutably borrow an asset from a kiosk if it is not listed. You can use the kiosk::borrow_mut function to mutably borrow an asset. However, it is not possible to use references within a PTB, so to access the mutably borrowed asset you must use a published module (function).

You can use the PTB-friendly kiosk::borrow_val function. It allows you to take an asset and place it back in the same transaction. To make sure the asset is placed back into the kiosk, the function "obliges" the caller with a “Hot Potato”.

When someone purchases an item, Sui stores the proceeds from the sale in the kiosk. As the kiosk owner, you can withdraw the proceeds at any time by calling the kiosk::withdraw function. The function is simple, but because it is PTB friendly it is not currently supported in the Sui CLI.

This action is not currently supported in the CLI environment.

Lock items in a kiosk

Some policies require that assets never get removed from a kiosk, such as for strong royalty enforcement. To support this, Sui Kiosk provides a locking mechanism. Locking is similar to placing except that you can't take a locked asset out of the kiosk.

To lock an asset in a kiosk, call the sui::kiosk::lock function. To ensure that you can later unlock the asset you must associate a TransferPolicy with the asset.

After you lock an asset, you must use list or list_with_purchase_cap functions to list it.

When you use the lock function, similar to using the place function, you specify the KioskOwnerCap and the Item as arguments. But to lock the item, you must also show the TransferPolicy .

in the following examples represents the full type of the asset.

Sui Kiosk provides basic trading functionality. As a kiosk owner, you can list assets for sale, and buyers can discover and purchase them. Sui Kiosk supports listing items by default with three primary functions:

Anyone on the network can purchase an item listed from a Sui Kiosk. To learn more about the purchase flow, see the Purchase section . To learn more about asset states and what can be done with a listed item, see the Asset States section.

As a kiosk owner, you can use the kiosk::list function to list any asset you added to your kiosk. Include the item to sell and the list price as arguments. All listings on Sui are in SUI tokens. When you list an item, Sui emits a kiosk::ItemListed event that contains the kiosk ID, item ID, type of the item, and the list price.

As a kiosk owner you can use the kiosk::delist to delist any currently listed asset. Specify the item to delist as an argument.

When you delist an item, Sui returns to the kiosk owner the gas fees charged to list the item.

When you delist an item, Sui emits a kiosk::ItemDelisted event that contains the kiosk ID, item ID, and the type of the item.

Anyone that has an address on the Sui network can purchase an item listed from a Sui Kiosk. To purchase an item, you can use the kiosk::purchase function. Specify the item to purchase and pay the list price set by the kiosk owner.

You can discover the items listed on the network with the kiosk::ItemListed event.

When you use the kiosk::purchase function, it returns the purchased asset and the TransferRequest for the type associated with the asset. To complete the purchase, you must meet the terms defined in the TransferPolicy applied to the asset.

As a kiosk owner, you can access an asset placed or locked in a kiosk without taking the asset from the kiosk. You can always borrow the asset immutably. Whether you can mutably borrow an asset depends on the state of the asset. For example, you can’t borrow a listed asset because you can’t modify it while listed. The functions available include:

You can always borrow an asset from a kiosk immutably. You can use the kiosk::borrow function to borrow an asset, however, it is not possible to use references within a programmable transaction block. To access the asset you must use a published module (function).

You can mutably borrow an asset from a kiosk if it is not listed. You can use the kiosk::borrow_mut function to mutably borrow an asset. However, it is not possible to use references within a PTB, so to access the mutably borrowed asset you must use a published module (function).

You can use the PTB-friendly kiosk::borrow_val function. It allows you to take an asset and place it back in the same transaction. To make sure the asset is placed back into the kiosk, the function "obliges" the caller with a “Hot Potato”.

When someone purchases an item, Sui stores the proceeds from the sale in the kiosk. As the kiosk owner, you can withdraw the

proceeds at any time by calling the kiosk::withdraw function. The function is simple, but because it is PTB friendly it is not currently supported in the Sui CLI.

This action is not currently supported in the CLI environment.

List and delist items from a kiosk

Sui Kiosk provides basic trading functionality. As a kiosk owner, you can list assets for sale, and buyers can discover and purchase them. Sui Kiosk supports listing items by default with three primary functions:

Anyone on the network can purchase an item listed from a Sui Kiosk. To learn more about the purchase flow, see the Purchase section . To learn more about asset states and what can be done with a listed item, see the Asset States section.

As a kiosk owner, you can use the kiosk::list function to list any asset you added to your kiosk. Include the item to sell and the list price as arguments. All listings on Sui are in SUI tokens. When you list an item, Sui emits a kiosk::ItemListed event that contains the kiosk ID, item ID, type of the item, and the list price.

As a kiosk owner you can use the kiosk::delist to delist any currently listed asset. Specify the item to delist as an argument.

When you delist an item, Sui returns to the kiosk owner the gas fees charged to list the item.

When you delist an item, Sui emits a kiosk::ItemDelisted event that contains the kiosk ID, item ID, and the type of the item.

Anyone that has an address on the Sui network can purchase an item listed from a Sui Kiosk. To purchase an item, you can use the kiosk::purchase function. Specify the item to purchase and pay the list price set by the kiosk owner.

You can discover the items listed on the network with the kiosk::ItemListed event.

When you use the kiosk::purchase function, it returns the purchased asset and the TransferRequest for the type associated with the asset. To complete the purchase, you must meet the terms defined in the TransferPolicy applied to the asset.

As a kiosk owner, you can access an asset placed or locked in a kiosk without taking the asset from the kiosk. You can always borrow the asset immutably. Whether you can mutably borrow an asset depends on the state of the asset. For example, you can’t borrow a listed asset because you can’t modify it while listed. The functions available include:

You can always borrow an asset from a kiosk immutably. You can use the kiosk::borrow function to borrow an asset, however, it is not possible to use references within a programmable transaction block. To access the asset you must use a published module (function).

You can mutably borrow an asset from a kiosk if it is not listed. You can use the kiosk::borrow_mut function to mutably borrow an asset. However, it is not possible to use references within a PTB, so to access the mutably borrowed asset you must use a published module (function).

You can use the PTB-friendly kiosk::borrow_val function. It allows you to take an asset and place it back in the same transaction. To make sure the asset is placed back into the kiosk, the function "obliges" the caller with a “Hot Potato”.

When someone purchases an item, Sui stores the proceeds from the sale in the kiosk. As the kiosk owner, you can withdraw the proceeds at any time by calling the kiosk::withdraw function. The function is simple, but because it is PTB friendly it is not currently supported in the Sui CLI.

This action is not currently supported in the CLI environment.

Purchase an item from a kiosk

Anyone that has an address on the Sui network can purchase an item listed from a Sui Kiosk. To purchase an item, you can use the kiosk::purchase function. Specify the item to purchase and pay the list price set by the kiosk owner.

You can discover the items listed on the network with the kiosk::ItemListed event.

When you use the kiosk::purchase function, it returns the purchased asset and the TransferRequest for the type associated with the asset. To complete the purchase, you must meet the terms defined in the TransferPolicy applied to the asset.

As a kiosk owner, you can access an asset placed or locked in a kiosk without taking the asset from the kiosk. You can always borrow the asset immutably. Whether you can mutably borrow an asset depends on the state of the asset. For example, you can’t

borrow a listed asset because you can’t modify it while listed. The functions available include:

You can always borrow an asset from a kiosk immutably. You can use the kiosk::borrow function to borrow an asset, however, it is not possible to use references within a programmable transaction block. To access the asset you must use a published module (function).

You can mutably borrow an asset from a kiosk if it is not listed. You can use the kiosk::borrow_mut function to mutably borrow an asset. However, it is not possible to use references within a PTB, so to access the mutably borrowed asset you must use a published module (function).

You can use the PTB-friendly kiosk::borrow_val function. It allows you to take an asset and place it back in the same transaction. To make sure the asset is placed back into the kiosk, the function "obliges" the caller with a “Hot Potato”.

When someone purchases an item, Sui stores the proceeds from the sale in the kiosk. As the kiosk owner, you can withdraw the proceeds at any time by calling the kiosk::withdraw function. The function is simple, but because it is PTB friendly it is not currently supported in the Sui CLI.

This action is not currently supported in the CLI environment.

Borrow an item from a kiosk

As a kiosk owner, you can access an asset placed or locked in a kiosk without taking the asset from the kiosk. You can always borrow the asset immutably. Whether you can mutably borrow an asset depends on the state of the asset. For example, you can’t borrow a listed asset because you can’t modify it while listed. The functions available include:

You can always borrow an asset from a kiosk immutably. You can use the kiosk::borrow function to borrow an asset, however, it is not possible to use references within a programmable transaction block. To access the asset you must use a published module (function).

You can mutably borrow an asset from a kiosk if it is not listed. You can use the kiosk::borrow_mut function to mutably borrow an asset. However, it is not possible to use references within a PTB, so to access the mutably borrowed asset you must use a published module (function).

You can use the PTB-friendly kiosk::borrow_val function. It allows you to take an asset and place it back in the same transaction. To make sure the asset is placed back into the kiosk, the function "obliges" the caller with a “Hot Potato”.

When someone purchases an item, Sui stores the proceeds from the sale in the kiosk. As the kiosk owner, you can withdraw the proceeds at any time by calling the kiosk::withdraw function. The function is simple, but because it is PTB friendly it is not currently supported in the Sui CLI.

This action is not currently supported in the CLI environment.

Withdraw proceeds from a completed sale

When someone purchases an item, Sui stores the proceeds from the sale in the kiosk. As the kiosk owner, you can withdraw the proceeds at any time by calling the kiosk::withdraw function. The function is simple, but because it is PTB friendly it is not currently supported in the Sui CLI.

This action is not currently supported in the CLI environment.