Action Request

The main difference between tokens and coins is that tokens do not allow transfers, conversions, or spends by default. There is an authorization mechanism, however, that allows these actions. This mechanism is called an ActionRequest . You can choose to allow or disallow any of the actions independently (see the Request confirmation section).

Tokens have four protected actions that create an ActionRequest :

ActionRequest is defined in the sui::token module and contains the following fields:

Rules can use these fields to determine whether the action should be allowed or not. Rules are custom modules that implement restriction logic. See Rules for more details.

An example of a function creating an ActionRequest :

There are three ways to confirm an ActionRequest using a:

You can't use TokenPolicyCap to confirm spend requests.

Use the TreasuryCap to confirm any action request for the token. It's useful for administrator actions (like mint and transfer), as well as for simple applications that don't require a token policy and wrap the TreasuryCap into the main object.

The signature for the token::confirm_with_treasury_cap function is:

An example of a transaction implemented in TypeScript with sui.js, confirming an action request with a TreasuryCap . Here the admin account owns the TreasuryCap , which is used to mint and confirm the transfer request for the token:

TokenPolicy is a way of enabling certain actions network-wide. After sharing, the TokenPolicy is available to everyone. Hence, wallets or other clients can use it to confirm allowed operations.

The signature for the token::confirm_request function is:

If it's a spend request, use the confirm_request_mut function instead.

An example of a client transfer request confirmation in JavaScript:

Use TokenPolicyCap to confirm action requests. A convenient approach when the TreasuryCap is wrapped in another object, and TokenPolicy does not allow certain action or has rules that make the default way of confirming impossible.

You can't use TokenPolicyCap to confirm spend requests.

An example of a client transfer request confirmation in JavaScript:

ActionRequest s can collect approvals - witness stamps from applications or rules. They carry the confirmation that a certain module or a rule has approved the action. This mechanic allows gating actions behind certain requirements.

The signature for the token::add_approval function is:

Approvals are mostly used for rules, but they can carry confirmations from any module.

Anyone can create a new ActionRequest using the token::new_request function. You can use it to create custom actions and rules, not necessarily related to the token itself.

Because you can create an ActionRequest freely for any type T , you can't use them as a proof of the action. Their purpose is authorization , not proof.

The signature for the token::new_request function is:

Protected actions

Tokens have four protected actions that create an ActionRequest :

ActionRequest is defined in the sui::token module and contains the following fields:

Rules can use these fields to determine whether the action should be allowed or not. Rules are custom modules that implement restriction logic. See Rules for more details.

An example of a function creating an ActionRequest :

There are three ways to confirm an ActionRequest using a:

You can't use TokenPolicyCap to confirm spend requests.

Use the TreasuryCap to confirm any action request for the token. It's useful for administrator actions (like mint and transfer), as well as for simple applications that don't require a token policy and wrap the TreasuryCap into the main object.

The signature for the token::confirm_with_treasury_cap function is:

An example of a transaction implemented in TypeScript with sui.js, confirming an action request with a TreasuryCap . Here the admin account owns the TreasuryCap , which is used to mint and confirm the transfer request for the token:

TokenPolicy is a way of enabling certain actions network-wide. After sharing, the TokenPolicy is available to everyone. Hence, wallets or other clients can use it to confirm allowed operations.

The signature for the token::confirm_request function is:

If it's a spend request, use the confirm_request_mut function instead.

An example of a client transfer request confirmation in JavaScript:

Use TokenPolicyCap to confirm action requests. A convenient approach when the TreasuryCap is wrapped in another object, and TokenPolicy does not allow certain action or has rules that make the default way of confirming impossible.

You can't use TokenPolicyCap to confirm spend requests.

An example of a client transfer request confirmation in JavaScript:

ActionRequest s can collect approvals - witness stamps from applications or rules. They carry the confirmation that a certain module or a rule has approved the action. This mechanic allows gating actions behind certain requirements.

The signature for the token::add_approval function is:

Approvals are mostly used for rules, but they can carry confirmations from any module.

Anyone can create a new ActionRequest using the token::new_request function. You can use it to create custom actions and rules, not necessarily related to the token itself.

Because you can create an ActionRequest freely for any type T , you can't use them as a proof of the action. Their purpose is authorization , not proof.

The signature for the token::new_request function is:

ActionRequest structure

ActionRequest is defined in the sui::token module and contains the following fields:

Rules can use these fields to determine whether the action should be allowed or not. Rules are custom modules that implement restriction logic. See Rules for more details.

An example of a function creating an ActionRequest :

There are three ways to confirm an ActionRequest using a:

You can't use TokenPolicyCap to confirm spend requests.

Use the TreasuryCap to confirm any action request for the token. It's useful for administrator actions (like mint and transfer), as well as for simple applications that don't require a token policy and wrap the TreasuryCap into the main object.

The signature for the token::confirm_with_treasury_cap function is:

An example of a transaction implemented in TypeScript with sui.js, confirming an action request with a TreasuryCap . Here the admin account owns the TreasuryCap , which is used to mint and confirm the transfer request for the token:

TokenPolicy is a way of enabling certain actions network-wide. After sharing, the TokenPolicy is available to everyone. Hence, wallets or other clients can use it to confirm allowed operations.

The signature for the token::confirm_request function is:

If it's a spend request, use the confirm_request_mut function instead.

An example of a client transfer request confirmation in JavaScript:

Use TokenPolicyCap to confirm action requests. A convenient approach when the TreasuryCap is wrapped in another object, and TokenPolicy does not allow certain action or has rules that make the default way of confirming impossible.

You can't use TokenPolicyCap to confirm spend requests.

An example of a client transfer request confirmation in JavaScript:

ActionRequest s can collect approvals - witness stamps from applications or rules. They carry the confirmation that a certain module or a rule has approved the action. This mechanic allows gating actions behind certain requirements.

The signature for the token::add_approval function is:

Approvals are mostly used for rules, but they can carry confirmations from any module.

Anyone can create a new ActionRequest using the token::new_request function. You can use it to create custom actions and rules, not necessarily related to the token itself.

Because you can create an ActionRequest freely for any type T , you can't use them as a proof of the action. Their purpose is authorization , not proof.

The signature for the token::new_request function is:

Request confirmation

There are three ways to confirm an ActionRequest using a:

You can't use TokenPolicyCap to confirm spend requests.

Use the TreasuryCap to confirm any action request for the token. It's useful for administrator actions (like mint and transfer), as well as for simple applications that don't require a token policy and wrap the TreasuryCap into the main object.

The signature for the token::confirm_with_treasury_cap function is:

An example of a transaction implemented in TypeScript with sui.js, confirming an action request with a TreasuryCap . Here the admin account owns the TreasuryCap , which is used to mint and confirm the transfer request for the token:

TokenPolicy is a way of enabling certain actions network-wide. After sharing, the TokenPolicy is available to everyone. Hence, wallets or other clients can use it to confirm allowed operations.

The signature for the token::confirm_request function is:

If it's a spend request, use the confirm_request_mut function instead.

An example of a client transfer request confirmation in JavaScript:

Use TokenPolicyCap to confirm action requests. A convenient approach when the TreasuryCap is wrapped in another object, and TokenPolicy does not allow certain action or has rules that make the default way of confirming impossible.

You can't use TokenPolicyCap to confirm spend requests.

An example of a client transfer request confirmation in JavaScript:

ActionRequest s can collect approvals - witness stamps from applications or rules. They carry the confirmation that a certain module

or a rule has approved the action. This mechanic allows gating actions behind certain requirements.

The signature for the token::add_approval function is:

Approvals are mostly used for rules, but they can carry confirmations from any module.

Anyone can create a new ActionRequest using the token::new_request function. You can use it to create custom actions and rules, not necessarily related to the token itself.

Because you can create an ActionRequest freely for any type T , you can't use them as a proof of the action. Their purpose is authorization , not proof.

The signature for the token::new_request function is:

Approving actions

ActionRequest s can collect approvals - witness stamps from applications or rules. They carry the confirmation that a certain module or a rule has approved the action. This mechanic allows gating actions behind certain requirements.

The signature for the token::add_approval function is:

Approvals are mostly used for rules, but they can carry confirmations from any module.

Anyone can create a new ActionRequest using the token::new_request function. You can use it to create custom actions and rules, not necessarily related to the token itself.

Because you can create an ActionRequest freely for any type T , you can't use them as a proof of the action. Their purpose is authorization , not proof.

The signature for the token::new_request function is:

Creating a custom request

Anyone can create a new ActionRequest using the token::new_request function. You can use it to create custom actions and rules, not necessarily related to the token itself.

Because you can create an ActionRequest freely for any type T , you can't use them as a proof of the action. Their purpose is authorization , not proof.

The signature for the token::new_request function is: