Connect to a Snap

Dapps can connect to Snaps designed to communicate with dapps. Dapps can use these Snaps to take advantage of new features enabled by Snaps. This is possible because Snaps can expose a custom JSON-RPC API .

Detect wallet

To connect to a Snap, dapps must first detect MetaMask in the user's browser. See the Wallet documentation on how to connect to MetaMask .

Detect MetaMask Flask

If you want to connect to Snap in MetaMask Flask , you first need to know whether the user has Flask installed.

The following example uses the @metamask/detect-provider package to get the provider object from MetaMask first:

index.js import

detectEthereumProvider

from

"@metamask/detect-provider" ;

// This resolves to the value of window.ethereum or null. const provider =

await

detectEthereumProvider ( ) ;

// web3_clientVersion returns the installed MetaMask version as a string. const isFlask =

(await provider ?. request ({

method :

"web3_clientVersion"

})) ?. includes ("flask") ;

if

(provider && isFlask)

{ console . log ("MetaMask Flask successfully detected!") ; // Now you can use Snaps! }

else

{ console . error ("Please install MetaMask Flask!" , error) ; }

Detect multiple wallets

See the following resources for detecting multiple wallets (via EIP-6963) in the user's browser:

- How to connect to MetaMask
- Connect to Snap via EIP-6963 example

Connect to a Snap

Connect to a Snap by calling the wallet_requestSnaps method from your dapp. If a user doesn't have the Snap installed in their MetaMask wallet, MetaMask prompts the user to install the Snap.

The following are different possible outcomes from calling wallet_requestSnaps .

User rejects the installation request

If the user rejects the installation request, the call to wallet_requestSnaps throws the following error:

{

"code" :

4001 ,

"message" :

"User rejected the request."

}

User approves the installation request

If the user approves the installation request, the call towallet_requestSnaps returns an object with the following shape:

{ "SNAP_ID" :

{ "blocked" :

false , "enabled" :

true , "id" :

"SNAP_ID" , "initialPermissions" :

{ // The permissions in the Snap's manifest file. } , "version" :

"SNAP_VERSION" } }

Snap is already installed

If the Snap is already installed, the call towallet_requestSnaps returns the same object as for a new installation of the Snap, but the user won't see a confirmation pop-up asking them to install the Snap.

important Snaps are installed into the MetaMask instance of each user. If a Snap stores data, that data is specific to that user's MetaMask instance. However, that data can be shared with multiple dapps. Do not assume that data stored by a Snap is unique to your dapp.

Determine whether a Snap is installed

Determine whether a Snap is installed by calling thewallet_getSnaps method from your dapp. This method returns a list of only those Snaps that are connected to your current dapp.

The response is in the form of an object keyed by the ID of the Snap. Each value is a nested object with additional information, such as the version of the Snap that is installed.

note wallet_getSnaps only returns the Snaps that are connected to your dapp. The user may have other Snaps installed that your dapp is not aware of. The following example verifies whether a Snap with IDnpm:super-snap is installed:

index.ts const snaps =

await ethereum . request ({ method :

"wallet_getSnaps" }) ;

const isMySnapInstalled = Object . keys (snaps) . includes ("npm:super-snap") ; If you need to work with a specific version of a Snap, you can instead iterate overObject.values(snaps) , and use theid andversion properties inside each object to determine whether the Snap is installed with the required version.

note A user cannot install multiple versions of a Snap into a single MetaMask instance. You should avoid requiring a specific version of a Snap unless you absolutely need to. In most cases, you should request the latest version of the Snap and architect your dapp to be able to work with that version.

Reconnect to a Snap

At any time, a user can open their MetaMask Snaps settings menu and see all the dapps connected to a Snap. From that menu they can revoke a dapp connection. If your dapp loses the connection to a Snap, you can reconnect by callingwallet_requestSnaps . Since the Snap is already installed, this returns a success response without MetaMask showing a pop-up. However, if the user has disabled the Snap, the response hasenabled set tofalse for yourSNAP_ID :

{ "SNAP_ID" :

{ "blocked" :

false , "enabled" :

false , "id" :

"SNAP_ID" , "initialPermissions" :

{ // The permissions in the Snap's manifest file. } , "version" :

"SNAP_VERSION" } }

Edit this page