Send transactions

You can send a transaction in MetaMask using the eth_sendTransaction RPC method.

For example, the following JavaScript gets the user's accounts and sends a transaction when they select each button:

index.js const ethereumButton =

document . querySelector (".enableEthereumButton") ; const sendEthButton =

document . querySelector (".sendEthButton") ;

let accounts =

[ ] ;

// Send Ethereum to an address. sendEthButton . addEventListener ("click" ,

( )

=>

{ provider // Or window.ethereum if you don't support EIP-6963. . request ({ method :

"eth_sendTransaction" , // The following sends an EIP-1559 transaction. Legacy transactions are also supported. params :

[{ // The user's active address. from : accounts [0] , // Required except during contract publications. to :

< recipient address

, // Only required to send ether to the recipient from the initiating external account. value :

< value in wei to send

, // Customizable by the user during MetaMask confirmation. gasLimit :

'0x5028' , // Customizable by the user during MetaMask confirmation. maxPriorityFeePerGas :

'0x3b9aca00' , // Customizable by the user during MetaMask confirmation. maxFeePerGas :

'0x2540be400' , } ,] , }) . then ((txHash)

=>

console . log (txHash)) . catch ((error)

=>

console . error (error)) ; }) ;

ethereumButton . addEventListener ("click" ,

( )

=>

{ getAccount ( ) ; }) ;

async

function

getAccount ( )

{ accounts =

await provider // Or window.ethereum if you don't support EIP-6963. . request ({

method :

"eth_requestAccounts"

}) ; } The following HTML displays the buttons:

index.html < button

class

" enableEthereumButton btn "

Enable Ethereum </ button

< button

class

" sendEthButton btn "

Send ETH </ button

Transaction parameters

Nonce

note MetaMask ignores this field. In Ethereum, every transaction has a nonce, so each transaction can only be processed by the blockchain once. To be a valid transaction, either:

- The nonce must be0
- .
- A transaction with a nonce of the previous number, from the same account, must have been processed.

This means that transactions are always processed in order for a given account.

Nonces are easy to mess up, especially when a user is interacting with multiple applications with pending transactions using the same account, potentially across multiple devices. Because of this, MetaMask doesn't allow dapp developers to customize nonces. Instead, MetaMaskassists the user in managing their transaction queue themselves.

Gas price

Gas price is an optional parameter, and best used on private blockchains.

In Ethereum, every transaction specifies a price for the gas it consumes. To maximize their profit, block producers pick pending transactions with higher gas prices first when creating the next block. This means that a high gas price usually causes your transaction to be processed faster, at the cost of greater transaction fees.

Some networks, such as Layer 2 networks, might have a constant gas price or no gas price. So while you can ignore this parameter on MetaMask's default networks, you might include it when your dapp knows more about the target network than MetaMask does. On the default networks, MetaMask allows users to choose between slow, medium, and fast options for their gas price.

Read abouthow to use advanced gas controls.

Gas limit

Gas limit is an optional parameter, since MetaMask automatically calculates a reasonable gas price.

To

Theto parameter is a hex-encoded Ethereum address. It's required for transactions with a recipient (all transactions except for contract creation).

Contract creation occurs when there is noto value but there is adata value.

Value

Thevalue parameter is a hex-encoded value of the network's native currency to send. On Mainnet, this isether , which is denominated in wei.

These numbers are often far higher precision than native JavaScript numbers, and can cause unpredictable behavior if not anticipated. We recommend usingBN.js when manipulating values intended for Ethereum.

Data

Thedata parameter is required for smart contract creation.

This field is also used for specifying contract methods and their parameters. See theSolidity ABI spec for more information on how the data is encoded.

Chain ID

note MetaMask ignores this field. The chain ID is derived from the user's current selected network. Useeth_chainId to get the user's chain ID. If you need the network version, usenet_version .

In the future, MetaMask might allow connecting to multiple networks at the same time, at which point this parameter will become important, so it might be useful to be in the habit of including it now.

Edit this page