Transaction types

You can interact with theEthereum JSON-RPC API using different transaction types (specified by thetype parameter).

The following methods use a unique format depending on the transaction type:

- eth_call
- eth_estimateGas
- eth_getTransactionByBlockHashAndIndex
- eth_getTransactionByBlockNumberAndIndex
- eth_getTransactionByHash
- eth_getTransactionReceipt

Legacy transactions

Transactions with type0x0 are legacy transactions that use the transaction format existing before typed transactions were introduced inEIP-2718 . They contain the parametersnonce ,gasPrice ,gasLimit ,to ,value ,data ,v ,r , ands . Legacy transactions don't useaccess lists or incorporateEIP-1559 fee market changes .

Access list transactions

Transactions with type0x1 are transactions introduced inEIP-2930 . They contain, along with thelegacy parameters , anaccessList parameter, which specifies an array of addresses and storage keys that the transaction plans to access (anaccess list). Access list transactions must specify an access list, and they don't incorporateEIP-1559 fee market changes .

Also, access list transactions contain theyParity parameter. The returned values for this parameter can either be0x0 or0x1 . This is the parity (0 for even, 1 for odd) of the y-value of asecp256k1 signature.

Use theeth_createAccessList API to simulate a transaction which returns the addresses and storage keys that may be used to send the real transaction, and the approximate gas cost.

info View theInfura article that describes howeth_createAccessList can help optimize gas costs, reduce out-of-gas errors, and verify clients for infrastructure access.

EIP-1559 transactions

Transactions with type0x2 are transactions introduced inEIP-1559 , included in Ethereum'sLondon fork . EIP-1559 addresses the network congestion and overpricing of transaction fees caused by the historical fee market, in which users send transactions specifying a gas price bid using thegasPrice parameter, and miners choose transactions with the highest bids.

EIP-1559 transactions don't specifygasPrice , and instead use an in-protocol, dynamically changingbase fee per gas. At each block, the base fee per gas is adjusted to address network congestion as measured by a gas target.

EIP-1559 transactions contain theaccessList andyParity parameters andlegacy parameters (except forgasPrice).

They also contain amaxPriorityFeePerGas parameter, which specifies the maximum fee the sender is willing to pay per gas above the base fee (the maximumpriority fee per gas), and amaxFeePerGas parameter, which specifies the maximum total fee (base fee + priority fee) the sender is willing to pay per gas.

An EIP-1559 transaction always pays the base fee of the block it's included in, and it pays a priority fee as priced bymaxPriorityFeePerGas or, if the base fee per gas +maxPriorityFeePerGas exceedsmaxFeePerGas , it pays a priority fee as priced bymaxFeePerGas minus the base fee per gas. The base fee is burned, and the priority fee is paid to the miner that included the transaction. A transaction's priority fee per gas incentivizes miners to include the transaction over other transactions with lower priority fees per gas.

Read theConsenSys EIP-1559 primer for more information on how EIP-1559 changes Ethereum.