Signing and Sending a Transaction

Suggest Edits

Sending a transaction on the Solana network requires the creation of a Transaction object, followed by the transaction being signed and submitted by the user's Magic Eden wallet.

Solana Web3js

For more information on interacting with Solana on the client side, check out the docs forSolana Web3js

For more general Solana development information, check out the courses and resources offered onSoldev

Creating a Transaction

Before you can sign or send a transaction, you must create one! Below is a basic example of creating a tx that has a user send funds right back to themselves

JavaScript import { Transaction, SystemProgram, Connection, PublicKey, } from "@solana/web3.js";

/* * *Creates an example transaction to transfer 100 lamports to the same account. * @param {String} publicKey a wallet public key * @param {Connection} connection an RPC connection * @returns {Transaction} a transaction* / export const createTransaction = async (publicKey: PublicKey, connection: Connection): Promise => { const latestBlockhash = (await connection.getLatestBlockhash()).blockhash; const transaction = new Transaction().add(SystemProgram.transfer({ fromPubkey: publicKey, toPubkey: publicKey, lamports: 1000, })); transaction.feePayer = publicKey; transaction.recentBlockhash = latestBlockhash;

return transaction; };

Signing and Sending a Transaction

Once the transaction is created, you can choose then prompt the user to sign and send it. These operations can also be performed separately, but this example will illustrate them happening in tandem.

JavaScript import { Transaction } from "@solana/web3.js";

import { MagicEdenProvider } from "../types/types";

/* * *Signs and sends the given created transaction using the MagicEdenProvider object. * @param {MagicEdenProvider} provider The MagicEdenProvider object * @param {Transaction} transaction The transaction to sign and send * @returns {Promise} A promise that resolves to the signature of the transaction* / export const signAndSendTransaction = async (provider: MagicEdenProvider, transaction: Transaction): Promise => { try { const { signature } = await provider.signAndSendTransaction(transaction); return signature; } catch (error) { console.warn(error); throw new Error("An unexpected error occured while signing the transaction."); } }; All of this functionality and more can be found at ourSolana Demo , so feel free to check it out if you want to learn more about handling wallet functionality when connecting directly to the Magic Eden wallet. Updatedabout 1 month ago * Table of Contents * * Creating a Transaction * * Signing and Sending a Transaction