Implementation

SDK implementation guide.

1. Import dependencies in your project

```

Copy import { ArgentTMA, SessionAccountInterface } from '@argent/tma-wallet';

```

1. Initialize the ArgentTMA object

Initialize theArgentTMA object with your app configuration:

```

Copy const argentTMA = ArgentTMA.init({ environment: "sepolia", // "sepolia" | "mainnet" (Whitelisting required) appName: "My TG Mini Test Dapp", // Your Telegram app name appTelegramUrl: "https://t.me/my_telegram_bot/app_name", // Your Telegram app URL sessionParams: { allowedMethods: [ // List of contracts/methods allowed to be called by the session key { contract: "0x036133c88c1954413150db74c26243e2af77170a4032934b275708d84ec5452f", // contract address selector: "increment", //function selector } ], validityDays: 90 // session validity (in days) - default: 90 }, });

```

1. Request a connection

If the user is not connected, call therequestConnection() method to open the wallet and ask the user to approve the connection. At the same time, you can ask user for token approvals:

```

Copy const handleConnectButton = async () => { await argentTMA.requestConnection({ callbackData: 'custom_callback', approvalRequests: [ { tokenAddress: '0x049D36570D4e46f48e99674bd3fcc84644DdD6b96F7C741B1562B82f9e004dC7', amount: BigInt(1000000000000000000).toString(), spender: 'spender_address', } ], }); };

```

The wallet will redirect back to your app and the account will be available from theconnect() method.

1. Check connection status

You can check if the user is connected at any time using theisConnected() method:

```

Copy const isConnected = argentTMA.isConnected();

```

1. Connect to the wallet

Call theconnect() method when your app loads to check if the user is already connected. It is also used to fetch theaccount object. It is an extendedstarknet.js account object.

For instance, you could wrap this in auseEffect hook.

```

Copy useEffect(() => { // Call connect() as soon as the app is loaded argentTMA .connect() .then((res) => { if (!res) { // Not connected setIsConnected(false); return; }

const { account, callbackData } = res;

if (account.getSessionStatus() !== "VALID") { // Session has expired or scope (allowed methods) has changed // A new connection request should be triggered

// The account object is still available to get access to user's address // but transactions can't be executed const { account } = res;
```

setAccount(account); setIsConnected(false); return; }

// The session account is returned and can be used to submit transactions setAccount(account); setIsConnected(true); // Custom data passed to the requestConnection() method is available here console.log("callback data:", callbackData); }) .catch((err) => { console.error("Failed to connect", err); }); }, []);

```

1. Interact with Starknet using the account

You can interact with your contracts using starknet.js. For instance, you could do this:

```

Copy const { transaction_hash } = await account.execute(myCall, { version: 3, maxFee: 10 ** 15, feeDataAvailabilityMode: RPC.EDataAvailabilityMode.L1, resourceBounds: { l1_gas: { max_amount: num.toHex(maxQtyGasAuthorized), max_price_per_unit: num.toHex(maxPriceAuthorizeForOneGas), }, l2_gas: { max_amount: num.toHex(0), max_price_per_unit: num.toHex(0), }, }, });

```

1. Check account session status

```

Copy const sessionStatus = account.getSessionStatus(); // "VALID" | "EXPIRED" | "INVALID_SCOPE" | "INVALID_SIGNATURE"

```

1. Request approval

It is possible to ask to user to sign new approval transactions withrequestApprovals() .

```

Copy async function handleApproval() { try { const res = await argentTMA.requestApprovals([{ tokenAddress: '0x049D36570D4e46f48e99674bd3fcc84644DdD6b96F7C741B1562B82f9e004dC7', amount: BigInt(1000000000000000000).toString(), spender: 'spender_address', }],); } catch (error) { console.error('Approval failed:', error); } }

```

1. Clear session

CallingclearSession removes the session object from local storage. It is mostly used for debugging. The session would still be valid on-chain.

We could imagine doing something like this:

```

Copy const handleClearSessionButton = async () => { await argentTMA.clearSession(); setAccount(undefined); };

```

List of useful ressources:

- npm package
- Tamagotchi coding tutorial
- Building a Telegram Game on Starknet
-

Previous Setup guide Next Types and interfaces
```