V1.1: Do more programmatically, extra webhooks, plus loads more wallet support!

The upgrade guide for this release containing only implementation related changes can be foundhere . This release is not a literal major one as far as versioning is concerned, but it's packed with helpful updates!

Phantom Mobile Redirect UX

Ever wished you could connect directly to Phantom from your mobile browser, instead of opening in the in-app browser? Now you can with a new propmobileExperience . Just set it toredirect and enjoy the new Phantom deeplinking UX on mobile.

ERC 1155 support

Our NFT Gating feature helps you gate by NFT ownership. We've now added support for ERC 1155 tokens, so you can now gate by NFT ownership for both ERC 721 and ERC 1155 token types!

New webhooks

We've added a bunch of new webhook events to help you keep track of what's going on in your app. You can find the full list of webhookshere . Here's what's new:

- user.passkeyRecovery.started
- user.passkeyRecovery.completed
- user.social.linked
- user.social.unlinked
- wallet.transferred
- visit.created

userId in all webhook payloads

You wanted an easier way to always have the userId of the user that triggered the event - now we also include the userId of the trigger in all webhook event objects!

Extra wallet support

- Argent Web and Mobile now supported
- StarkNet support for Sepolia added
- Enabled Ledger for Glow, Solflare, and Backpack
- Enabled hardware wallet for Solana wallets
- Magic integration now supports Flow wallets

Wallet account export

Wallet account private key export is now available for Embedded HD walllets. This is a great way to allow your users to export their wallet account's private key for use elsewhere. Now users can select between exporting either their entire wallet's seed phrase, or a single wallet's private key.

One-Time Codes & Session keys for Embedded Wallets

You can now configure embedded wallets to sign for transactions using one-time email verification codes. You can also define a session length that allows users to transact without a secondary prompt for the defined time period.

More information abou this featurehere .

Solana support for Embedded Wallets

Dynamic now offers embedded wallets on EVM compatible networks and Solana. If you enable both, they will both be created at once and whichever you have marked as “primary” will be shown as the primary address in their profile upon sign in.

Solana embedded wallets support signing off-chain messages, sending on-chain transactions and more.

More information about that featurehere .

Extra text in the signup/login UI

Programmable Views allow you much more control over what signup/login methods you display at any time. Additionally, you can now create views section for extra text in the signup/login UI:

views : [{ type : 'login' , sections : [{ type : 'text' , label : 'Intro Text' , alignment : 'center' } , ...] , } ,] ,

Hooks, methods & widgets

- showDynamicUserProfile
- Currently the setShowDynamicUserProfile method exists on useDynamicContext which helps you trigger the user profile modal, however you had no way of knowing if the modal was open or not. Now you can use the showDynamicUserProfile boolean to get this information.

export const Example = ( ) => { const { showDynamicUserProfile} = useDynamicContext ( ) ; React . useEffect (( ) => { if (showDynamicUserProfile) { / *On widget opens*/ } else { /*On widget closes*/ } } , [showDynamicUserProfile]) ; return null ; } ; * useConnectWithEmailOtp * Now you can handle headless email signup with the useConnectWithEmailOtp hook! It exposesconnectWithEmail * andverifyOneTimePassword * functions. It supports both Dynamic native login and our Magic integration. * setShowLinkNewWalletModal * This new method comes as part ofuseDynamicContext * and allows you to trigger the link new wallet method programmatically.

const LinkNewWalletButton = ( ) => { const { setShowLinkNewWalletModal} = useDynamicContext ( ) ; return (< button onClick = { ( ) => setShowLinkNewWalletModal (true) }

Link a new wallet </ button

) ; } ; *setAuthMode * Previously, you could set the auth mode via theinitialAuthenticationMode * prop onDynamicContextProvider * but this would be set in stone once the app loads. Now you can also set it programmatically via thesetAuthMode * method onuseDynamicContext * .

const Example = ( ) => { const { setAuthMode} = useDynamicContext ( ) ; return (< button onClick = { ( ) => setAuthMode ("connect-only") }

Connect Only </ button

) ; } ; * NewmultiWalletPromptsWidget

This allows you to use the multi wallet prompts without needing the full user profile which in which these prompts are normally bundled.

New callbacks

- onAuthFlowCancel
- This callback is triggered when the user cancels the auth flow before successful completion. It will get called alongsideonAuthFlowClose
- .

Was this page helpful?

Yes No TypeScript Compilation Errors V1: Pre-generated Wallets, Account Abstraction & More twitter linkedin slack Powered