CoW Swap Widget

Integrate the power ofCoW Swap into your product! With the widget, you can create an incredible trading interface. Specify the required pair of currencies, customize the look and much more!

Create your own widget using the configuratorhttps://widget.cow.fi .

Install

```
yarn add @cowprotocol/widget-lib npm install @cowprotocol/widget-lib
```

Quick start

```
import

{ cowSwapWidget , CowSwapWidgetParams }

from

'@cowprotocol/widget-lib'

// HTML element where the widget will be rendered const widgetContainer = document . getElementById ( 'cowswap-widget'
)

const params : CowSwapWidgetParams =

{ appCode :

'NAME-OF-YOU-APP' ,

// Add here the name of your app. e.g. "Pig Swap" width :

600 , height :

640 , sell :

{ asset :

'DAI' } , buy :

{ asset :

'USDC' , amount :

'0.1' } }

cowSwapWidget ( widgetContainer , params )
```

App key

You must specify theappCode parameter when initializing the widget. This parameter is used to identify the source of orders. The key must be a UTF8 string of up to 50 chars. It will be a part of orders meta-data, see more in theCoW Protocol Docs .

Partner fee

You may participate in the Partner Fee program to collect fee on trades executed by your users through the Widget by adding the following parameter to your Widget:

```
import

{ cowSwapWidget , CowSwapWidgetParams }

from

'@cowprotocol/widget-lib'

const widgetContainer = document . getElementById ( 'cowswap-widget' )

const params : CowSwapWidgetParams =
```

{ partnerFee :

{ bps :

50 ,

// 0.5% recipient :

'0x0000000000000000000000000000000000000000'

// Fee destination address } }

cowSwapWidget (widgetContainer , params) The fee in basis points (BPS). One basis point is equivalent to 0.01% (1/100th of a percent).

note The fee cannot exceed 1% (100 bps). The recipient is the address to which the fee will be sent.

Once you have set up the partner fee, you will see the fee in the CoW Swap UI:

Terms and conditions apply.

Seehere for detailed info about fee calculation and examples.

Wallet provider

You can pass the wallet provider from your application to seamlessly use the widget as part of your application. Also, you can not specify the provider, in this case the widget will work in standalone mode with the ability to connect any wallet supported in CoW Swap.

A provider must comply withEIP-1193 and implement the interface:

interface

EthereumProvider

{ on (event :

string , args :

unknown) :

void

request < T

    (params : JsonRpcRequest) :

Promise < T

enable ( ) :

Promise < void

    }

interface

JsonRpcRequest

{ id :

number method :

string params :

unknown [ ] } An example of connecting a widget to Rabby Wallet or Metamask:

import

{ cowSwapWidget , CowSwapWidgetParams }

from

'@cowprotocol/widget-lib'

cowSwapWidget (document . getElementById ('cowswap-widget') , { provider : window . ethereum // <------- })

Configuration

CowSwapWidgetParams

All params are optional Parameter Type Default Description width string 400px The width of the widget in css values (px, vh, etc.). height string 600px The height of the widget in css values (px, vh, etc.). appCode string

---

The unique identifier of the widget consumer. Please fill the for to let us know a little about you:https://cowprotocol.typeform.com/to/rONXaxHV . provider EthereumProvider

---

The Ethereum provider to be used for interacting with a wallet. To connect, for example, to Rabby Wallet or Metamask, just setwindow.ethereum . You also might like to usehttps://web3modal.com . chainId number 1 The blockchain ID on which the trade will take place. Currently supported: 1 (Mainnet), 11155111 (Sepolia), 100 (Gnosis chain). tradeType TradeType 'swap' The type of trade. Can beswap orlimit oradvanced . env CowSwapWidgetEnv 'prod' The environment of the widget (local ,prod ,dev ,pr). SeeCOWSWAP_URLS const value for urls. tradeAssets TradeAssets Same as in swap.cow.fi An object containing information about the selling and buying assets. Example:{ asset: 'WBTC', amount: 12 } or{ asset: '0xa0b86991c6218b36c1d19d4a2e9eb0ce3606eb48' } . theme CowSwapTheme 'light' The theme of the widget ('dark' for dark theme or'light' for light theme). It is also possible to set your own colors (CowSwapWidgetPalette). SeeCustom theme section for more details. logoUrl string

---

Allows to set a custom logo for the widget. disableToastMessages boolean false CoW Swap displays a pop-up notification when certain events occur, for example: an order has been filled. You may want to handle these events yourself and disable the display of notifications in the widget, to do this you need to enable this option. SeeEvents handling section for more details. standaloneMode boolean false When this option is enabled, the widget will use its own Ethereum provider and the user will be able to connect a wallet from within the widget. hideLogo boolean false Option to hide the logo in the widget. hideNetworkSelector boolean false Disables an opportunity to change the network from the widget UI. enabledTradeTypes Array All are enabled CoW Swap provides three trading widgets:swap ,limit andadvanced orders. Using this option you can narrow down the list of available trading widgets. partnerFee PartnerFee

---

You can enable a fee for all trades in the widget. SeePartner fee section for more details.

Custom theme

By setting custom colors you can very flexibly control the appearance of the widget. To do this, you need to pass an object of typeCowSwapWidgetPalette to thetheme parameter.

You actually can specify only 2 or 3 basic colors, the rest of the palette is efficiently derived (also with the help of thecolor2k library to get 'AA' rated text/background color contrasts).

Example:

import

{ cowSwapWidget , CowSwapWidgetParams , CowSwapWidgetPalette }

from

'@cowprotocol/widget-lib'

const widgetContainer = document . getElementById ('cowswap-widget')

// Set custom colors const theme : CowSwapWidgetPalette =

{ baseTheme :

'light' , primary :

'#00ff85' , background :

'#f7f7f7' , paper :

'#1a4435' , text :

'#ffffff' , warning :

'#ffb700' , alert :

'#b8ffb2' , success :

'#19ff64' } const params : CowSwapWidgetParams =

{ appCode :

'NAME-OF-YOU-APP' ,

// Add here the name of your app. e.g. "Pig Swap" theme }

cowSwapWidget (widgetContainer , params) Try it yourselfhttps://widget.cow.fi .

Events handling

The widget provides a bus of events that occur in the widget. Using it, you can receive notifications, for example, when an order has been executed or when an approval transaction has been mined.

Separately, it is worth noting theON_TOAST_MESSAGE event; it occurs every time any notification is displayed in CoW Swap. To avoid double display of notifications, enable thedisableToastMessages option in the widget configurations.

A list of all possible events can be found in the source code on[Github](#) .

Events usage example

import

{ cowSwapWidget , CowSwapWidgetParams , CowEventListeners , CowEvents }

from

'@cowprotocol/widget-lib'

const params : CowSwapWidgetParams =

{ appCode :

'YOUR_APP_ID' }

const listeners : CowEventListeners =

[{ event : CowEvents . ON_TOAST_MESSAGE , handler :

(event)

=>

console . log ('[ON_TOAST_MESSAGE]' , event . message) } , { event : CowEvents . ON_EXPIRED_ORDER , handler :

(event)

=>

console . log ('Order was expired:' , event . order) } ,]

const

{ updateListeners }

=

cowSwapWidget (container ,

{ params , listeners })

// If you want to change listeners at some point, you can do it like: updateListeners ([{ event : CowEvents . ON_CANCELLED_ORDER , handler :

(event)

=>

console . log ('Order was cancelled:' , event . order) }])

Widget updating

You can change all possible widget options on the fly:

import

{ cowSwapWidget , CowSwapWidgetParams }

from

'@cowprotocol/widget-lib'

const container = document . getElementById ('cowswap-widget')

const params : CowSwapWidgetParams =

{ appCode :

'NAME-OF-YOU-APP' ,

// Add here the name of your app. e.g. "Pig Swap" logoUrl :

'YOUR_LOGO_URL' }

const updateWidget =

cowSwapWidget (container , params)

// Update the widget updateWidget ({ ... params , theme :

'dark' ,

// <- Change theme to dark hideNetworkSelector :

true

// <- Hide the network selector })

Widget URL

Most of the widget parameters are controlled via the URL, which means that you can create the URL yourself and embed the iframe. An example of URL:

https://swap.cow.fi/#/100/swap/WXDAI/GNO?sellAmount=200&theme=dark