Postman API Suite

Suggest Edits

We are happy to share our Web3 Services Postman collections with you! If you are not familiar withPostman , it is one of the easiest ways to send API requests and try out Circle’s APIs. Each of our Postman collections contains templated requests that you can use to start testing Circle’s W3S APIs.

How It's Organized

We’ve created a Web3 Services Postman Workspace that includes one collection per product i.e. Programmable Wallets, Smart Contract Platform (coming soon!). Within each of these collections are folders that are organized in a way that matches ourAPI References .

How To Use It

Start by selecting Run in Postman below. From there, you can then choose to fork the collection to your workspace, view the collection in the public workspace, or import the collection into Postman.

1. Fork:
2. Creates a copy of the collection while maintaining a link to the parent.
3. View:
4. Allows you to quickly try out the API without having to import anything into your Postman suite.
5. Import:
6. Creates a copy of the collection but does not maintain a link to Circle’s copy.

Collection Run it! Programmable WalletsSmart Contract Platform

Additional Configuration

Authorization

For authorization, you can paste yourAPI key in the “Authorization” tab of the collection, as shown in the image below. If you do not have an API Key, you can acquire one by signing up for a Circle Developer accounthere . If you’d like to store your API key as a variable for more advanced testing, you can learn more in Postman’svariables guide.

Entity Secret

If you’re using Developer-Controlled wallets, you’ll need to generate a unique 32-byte entity secret. The guide on how to do so can be foundhere . Once you’ve registered your entity secret in the dashboard of your Circle Developer account, you can add yourhex-encoded entity secret (not the encrypted entity secret ciphertext) as the variable value shown below.

For security purposes, Circle enforces uniqueness for your entity secret for each request you make. The helper scripts included in the Postman collection will re-encrypt the entity secret ciphertext you provided from the step above automatically before each call!

To enable this setup, run the “Get public key for entity” request one time. This will set your entity public key as a variable that will be used to encrypt your entity secret.

Variables & Helper Scripts

baseURL

Variable Name baseUrl Variable Value https://api.circle.com/v1/w3s Originating API N/A API(s) using variable All Description Circle's API URL for our Web3 Services platform.

hex-encoded-entity-secret

Variable Name hex-encoded-entity-secret Variable Value your hex-encoded entity secret Originating API N/A API(s) using variable All Developer-Controlled Wallets calls Description Unique 32-byte entity secret generated and encoded by the developer.

entity-public-key

Variable Name entity-public-key Variable Value data.publicKey Originating API GET /config/entity/publicKey API(s) using variable N/A Description The RSA public key associated with your Circle Developer account. Used to encrypt the encoded entity secret.

X-user-token

Variable Name X-user-token Variable Value data.userToken Originating API POST /users/token API(s) using variable All User-Controlled Wallets calls Description The temporary session token generated each time a user is active. Must be regenerated for each session.

challengeId

Variable Name challengeId Variable Value data.challengeId Originating API POST /user/initialize POST /user/pin PUT /user/pin POST /user/pin/restore API(s) using variable GET /user/challenges/:id Description The unique UUID that corresponds to the various PIN flows. Passed in the User-Controlled Wallets SDK to complete each flow.

{blockchain}.{TOKEN SYMBOL}

Variable Name {blockchain}.{TOKEN SYMBOL} Variable Value data.tokenBalances[n].token.id Originating API GET /wallets/:id/balances API(s) using variable All Monitored Tokens/Token Lookup calls Description The name of this variable is the blockchain of the token in lowercase followed by the token symbol in all caps. The variable value is the unique UUID associated with the token.

userId

Variable Name userId Variable Value User ID Originating API POST /users API(s) using variable POST /users/token GET /users/:id Description The unique UUID associated with each user. Unlike the user token, this value is static and does not expire or rotate.

entitySecretCiphertext

Variable Name entitySecretCiphertext Variable Value Entity Secret Ciphertext Originating API N/A - Generated by pre-request script API(s) using variable All Developer-Controlled Wallets POST calls Description The encoded entity secret linked to your Circle Developer Account, encrypted via your entity public key. The entity secret is re-encrypted before each POST call, as the same ciphertext cannot be used twice.

walletSetId

Variable Name walletSetId Variable Value data.walletSet.id Originating API POST /developer/walletSets API(s) using variable GET /walletSets/:id PUT /developer/walletSets/:id POST /developer/wallets Description The unique UUID associated with each wallet set.

walletId

Variable Name walletId Variable Value Wallet ID Originating API POST /developer/wallets GET /wallets/:id API(s) using variable GET /wallets/:id GET /wallets/:id/balances GET /wallets/:id/nfts PUT /wallets/:id POST /transactions/transfer/estimateFee POST /transactions/contractExecution/estimateFee POST /user/transactions/transfer POST /user/transactions/contractExecution POST /developer/transactions/transfer POST /developer/transactions/contractExecution Description The unique UUID associated with each wallet.

importedContractId

Variable Name importedContractId Variable Value Contract ID Originating API POST /contracts/import API(s) using variable POST /contracts/:id/read GET /contracts/:id PATCH /contracts/:id Description The unique, Circle-generated UUID associated with each imported smart contract.

deployedContractId

Variable Name deployedContractId Variable Value Contract ID Originating API POST /contracts/deploy API(s) using variable POST /contracts/:id/read GET /contracts/:id PATCH /contracts/:id Description The unique, Circle-generated UUID associated with each deployed smart contract. Updated4 months ago