Quickstart: Crypto Deposits

Quickly start accepting USDC crypto payments on your site using our code examples.Suggest Edits

USDC is a “stablecoin,” a cryptocurrency that provides a stable level of purchasing power. As such, it is a popular currency for online transactions, and your customers will want to use it as a payment medium. This guide will walk you through the basics of accepting USDC on your site. Here we use the Ethereum blockchain as an example, but the same steps apply to other currencies and chains as well.

Steps:

1. Set up payment intent to pay with Crypto
2. Acquire blockchain address customer will pay to
3. Customer pays
4. Receive payment

1. Pay with Crypto

Once the customer reaches checkout and confirms they would like to pay with USDC on Ethereum, your system will send Circle a request to create a payment intent . Within this payment intent request you will specify the amount, currency, settlement currency, and chain.

Create a payment intent:

cURL JavaScript curl --location --request POST 'https://api-sandbox.circle.com/v1/paymentIntents' \ --header 'X-Requested-Id: {GUID}' \ --header 'Authorization: Bearer {YOUR_API_KEY}' \ --header 'Content-Type: application/json' \ --data-raw '{ "idempotencyKey": "17607606-e383-4874-87c3-7e46a5dc03dd", "amount": { "amount": "1.00", "currency": "USD" }, "settlementCurrency": "USD", "paymentMethods": [{ "type": "blockchain", "chain": "ETH" }] }' /* * *See installation instructions at * https://developers.circle.com/developer/docs/circle-sdk* / import { Circle, CircleEnvironments, PaymentIntentCreationRequest } from "@circle-fin/circle-sdk"; import crypto from 'crypto';

const circle = new Circle('', CircleEnvironments.sandbox // API base url);

async function createCryptoPayment() { const reqBody: PaymentIntentCreationRequest = { amount: { amount: "1.00", currency: "USD" }, settlementCurrency: "USD", paymentMethods: [{ type: "blockchain", chain: "ETH" }], idempotencyKey: crypto.randomUUID() }; const resp = await circle.cryptoPaymentIntents.createPaymentIntent(reqBody); console.log(resp.data); } createCryptoPayment(); Response { "data": { "id": "6e4d4047-db14-4c09-b238-1215aee50d03", "amount": { "amount": "1.00", "currency": "USD" }, "amountPaid": { "amount": "0.00", "currency": "USD" }, "amountRefunded": { "amount": "0.00", "currency": "USD" }, "settlementCurrency": "USD", "paymentMethods": [{ "type": "blockchain", "chain": "ETH" }], "paymentIds": [], "timeline": [{ "status": "created", "time": "2022-07-21T20:13:35.579331Z" }], "createDate": "2022-07-21T20:13:35.578678Z", "updateDate": "2022-07-21T20:19:24.859052Z", } } Webhook notification { "clientId": "f1397191-56e6-42fd-be86-0a7b9bd91522", "notificationType": "paymentIntents", "version": 1, "customAttributes": { "clientId": "f1397191-56e6-42fd-be86-0a7b9bd91522" }, "paymentIntent": { "id": "6e4d4047-db14-4c09-b238-1215aee50d03", "amount": { "amount": "1.00", "currency": "USD" }, "amountPaid": { "amount": "0.00", "currency": "USD" }, "amountRefunded": { "amount": "0.00", "currency": "USD" }, "settlementCurrency": "USD", "paymentMethods": [{ "type": "blockchain", "chain": "ETH" }], "paymentIds": [], "refundIds": [], "timeline": [{ "status": "created", "time": "2022-07-21T20:13:35.579331Z" }], "createDate": "2022-07-21T20:13:35.578678Z", "updateDate": "2022-07-21T20:13:35.578678Z" } } If you wish, you can create “long-lived” payment intent at a sub-wallet level (e.g., one deposit address and sub-wallet per sender). Start by creating a deposit wallet using theCreate a Wallet endpoint, then, send a request to create a continuous payment intent. Within this payment intent request specify the type: "continuous" , currency , settlementCurrency , chain , and merchantWalletId (optional) to receive funds into the sub-wallet instead of a master wallet.

2. Acquire blockchain address customer will pay to

For security reasons, we do not synchronously return the deposit blockchain address. To retrieve the blockchain deposit address, you have two options:

1. Subscribe to webhook notifications
2. Poll Circle APIs

Option 1: Webhook Notification

To receive webhook notifications, visit thenotification quickstart . and follow the steps shown. After you subscribe to notifications, you will receive updates for the payment intent whenever the resource has been updated. In this case, when the paymentMethods.address has been set, a notification will be sent with a new timeline object with a status of pending .

Payment intent webhook notification { "clientId": "f1397191-56e6-42fd-be86-0a7b9bd91522", "notificationType": "paymentIntents", "version": 1, "customAttributes": { "clientId": "f1397191-56e6-42fd-be86-0a7b9bd91522" }, "paymentIntent": { "id": "6e4d4047-db14-4c09-b238-1215aee50d03", "amount": { "amount": "1.00", "currency": "USD" }, "amountPaid": { "amount": "0.00", "currency": "USD" }, "amountRefunded": { "amount": "0.00", "currency": "USD" }, "settlementCurrency": "USD", "paymentMethods": [{ "type": "blockchain", "chain": "ETH", "address": "0x97de855690955e0da79ce5c1b6804847e7070c7f" }], "fees": [{ "type": "blockchainLeaseFee", "amount": "0.00", "currency": "USD" }], "paymentIds": [], "refundIds": [], "timeline": [{ "status": "pending", "time": "2022-07-21T20:13:38.188286Z" }, { "status": "created", "time": "2022-07-21T20:13:35.579331Z" }], "createDate": "2022-07-21T20:13:35.578678Z", "updateDate": "2022-07-21T20:13:38.186831Z", "expiresOn": "2022-07-21T21:13:38.087275Z" } }

Option 2: Poll Payment Intent Endpoint

If you prefer to pollget a payment intent , you can send get request until you receive paymentMethods.address .

Retrieve payment intent:

cURL JavaScript curl --location --request GET 'https://api-sandbox.circle.com/v1/paymentIntents/{id}' \ --header 'X-Requested-Id: {GUID}' \ --header 'Authorization: Bearer {YOUR_API_KEY}' /* * *See installation instructions at * https://developers.circle.com/developer/docs/circle-sdk* / import { Circle, CircleEnvironments } from "@circle-fin/circle-sdk";

const circle = new Circle('', CircleEnvironments.sandbox // API base url);

// Create promise that gets resolved in time milliseconds function delay(time: number) { return new Promise((resolve) => setTimeout(resolve, time)); }

async function getPaymentIntent(paymentInentId: string) { const resp = await circle.cryptoPaymentIntents.getPaymentIntent(paymentInentId);

return resp.data;

}

async function pollPaymentIntent() { const paymentInentId = 'payment-intent-id'; const pollInterval = 500; // Interval (in ms) by which to poll

```
let resp = undefined;
while (true) {
   resp = await getPaymentIntent(paymentInentId);
   let depositAddress = resp.data?.paymentMethods[0].address;

   if (depositAddress) break;
   await delay(pollInterval);
}
console.log(resp);
```

} pollPaymentIntent(); Response { "data": { "id": "6e4d4047-db14-4c09-b238-1215aee50d03", "amount": { "amount": "1.00", "currency": "USD" }, "amountPaid": { "amount": "0.00", "currency": "USD" }, "amountRefunded": { "amount": "0.00", "currency": "USD" }, "settlementCurrency": "USD", "paymentMethods": [{ "type": "blockchain", "chain": "ETH", "address": "0x97de855690955e0da79ce5c1b6804847e7070c7f" }], "fees": [{ "type": "blockchainLeaseFee", "amount": "0.00", "currency": "USD" }], "paymentIds": [], "refundIds": [], "timeline": [{ "status": "pending", "time": "2022-07-21T20:13:38.188286Z" }, { "status": "created", "time": "2022-07-21T20:13:35.579331Z" }], "createDate": "2022-07-21T20:13:35.578678Z", "updateDate": "2022-07-21T20:13:38.186831Z", "expiresOn": "2022-07-21T21:13:38.087275Z" } }

3. Enable customer payment

Once you receive the deposit address via paymentMethods.address , you'll provide it to the customer along with the amount and currency to be paid. You can present the address two ways:

1. as plain text the customer can cut and paste; or
2. as a QR code the customer can scan via an app.

The customer then sends payment from their wallet (custodial or non-custodial).

You can specify the timeframe in which the customer must send payment by adjusting the exipiresOn setting.

4. Receive payment from Circle

Once Circle obtains payment on-chain, we create a Payment resource linked to the Payment Intent created earlier and update the status of that Payment Intent. Your firm will then receive payment via the method specified. See the State Management Timeline for more details.

Option 1: Webhook Notifications

Payment intent webhook notification { "clientId": "f1397191-56e6-42fd-be86-0a7b9bd91522", "notificationType": "paymentIntents", "version": 1, "customAttributes": { "clientId": "f1397191-56e6-42fd-be86-0a7b9bd91522" }, "paymentIntent": { "id": "6e4d4047-db14-4c09-b238-1215aee50d03", "amount": { "amount": "1.00", "currency": "USD" }, "amountPaid": { "amount": "1.00", "currency": "USD" }, "amountRefunded": { "amount": "0.00", "currency": "USD" }, "settlementCurrency": "USD", "paymentMethods": [{ "type": "blockchain", "chain": "ETH", "address": "0x97de855690955e0da79ce5c1b6804847e7070c7f" }], "fees": [{ "type": "blockchainLeaseFee", "amount": "0.00", "currency": "USD" }, { "type": "totalPaymentFees", "amount": "0.01", "currency": "USD" }], "paymentIds": ["66c56b6a-fc79-338b-8b94-aacc4f0f18de"], "refundIds": [], "timeline": [{ "status": "complete", "context": "paid", "time": "2022-07-21T20:19:24.861094Z" }, { "status": "pending", "time": "2022-07-21T20:13:38.188286Z" }, { "status": "created", "time": "2022-07-21T20:13:35.579331Z" }], "createDate": "2022-07-21T20:13:35.578678Z", "updateDate": "2022-07-21T20:19:24.859052Z", "expiresOn": "2022-07-21T21:13:38.087275Z" } } Payment webhook notification { "clientId": "f1397191-56e6-42fd-be86-0a7b9bd91522", "notificationType": "payments", "version": 1, "customAttributes": { "clientId": "f1397191-56e6-42fd-be86-0a7b9bd91522" }, "payment": { "id": "66c56b6a-fc79-338b-8b94-aacc4f0f18de", "type": "payment", "status": "paid", "amount": { "amount": "1.00", "currency": "USD" }, "fees": { "amount": "0.01", "currency": "USD" }, "createDate": "2022-07-21T20:16:35.092Z", "updateDate": "2022-07-21T20:19:24.719Z", "merchantId": "f1397191-56e6-42fd-be86-0a7b9bd91522", "merchantWalletId": "1000999922", "paymentIntentId": "6e4d4047-db14-4c09-b238-1215aee50d03", "settlementAmount": { "amount": "1.00", "currency": "USD" }, "fromAddresses": { "chain": "ETH", "addresses" : ["0x0d4344cff68f72a5b9abded37ca5862941a62050"] }, "depositAddress": { "chain": "ETH", "address": "0x97de855690955e0da79ce5c1b6804847e7070c7f" }, "transactionHash": "0x7351585460bd657f320b9afa02a52c26d89272d0d10cc29913eb8b28e64fd906" } }

Option 2: Retrieve Payment Intent and Payment

Retrieve a payment intent:

cURL JavaScript curl --location --request GET 'https://api-sandbox.circle.com/v1/paymentIntents/{id}' \ --header 'X-Requested-Id: {GUID}' \ --header 'Authorization: Bearer {YOUR_API_KEY}' /* *See installation instructions at * https://developers.circle.com/developer/docs/circle-sdk* / import { Circle, CircleEnvironments } from "@circle-fin/circle-sdk";

const circle = new Circle('', CircleEnvironments.sandbox // API base url);

async function getPaymentIntent() { const paymentIntentId = ''; const resp = await circle.cryptoPaymentIntents.getPaymentIntent(paymentIntentId);

console.log(resp.data);

} getPaymentIntent(); Response { "data": { "id": "6e4d4047-db14-4c09-b238-1215aee50d03", "amount": { "amount": "1.00", "currency": "USD" }, "amountPaid": { "amount": "1.00", "currency": "USD" }, "amountRefunded": { "amount": "0.00", "currency": "USD" }, "settlementCurrency": "USD", "paymentMethods": [{ "type": "blockchain", "chain": "ETH", "address": "0x97de855690955e0da79ce5c1b6804847e7070c7f" }], "fees": [{ "type": "blockchainLeaseFee", "amount": "0.00", "currency": "USD" }, { "type": "totalPaymentFees", "amount": "0.01", "currency": "USD" }], "paymentIds": ["66c56b6a-fc79-338b-8b94-aacc4f0f18de"], "refundIds": [], "timeline": [{ "status": "complete", "context": "paid", "time": "2022-07-21T20:19:24.861094Z" }, { "status": "pending", "time": "2022-07-21T20:13:38.188286Z" }, { "status": "created", "time": "2022-07-21T20:13:35.579331Z" }], "createDate": "2022-07-21T20:13:35.578678Z", "updateDate": "2022-07-21T20:19:24.859052Z", "expiresOn": "2022-07-21T21:13:38.087275Z" } } Retrieve a payment:

cURL JavaScript curl --location --request GET 'https://api-sandbox.circle.com/v1/payments/{id}' \ --header 'X-Requested-Id: {GUID}' \ --header 'Authorization: Bearer {YOUR_API_KEY}' /* *See installation instructions at * https://developers.circle.com/developer/docs/circle-sdk* / import { Circle, CircleEnvironments } from "@circle-fin/circle-sdk";

const circle = new Circle('', CircleEnvironments.sandbox // API base url);

async function getPayment() { const paymentId = '' const resp = await circle.payments.getPayment(paymentId);

console.log(resp.data);

} getPayment(); Response { "data": { "id": "66c56b6a-fc79-338b-8b94-aacc4f0f18de", "type": "payment", "status": "paid", "amount": { "amount": "1.00", "currency": "USD" }, "fees": { "amount": "0.01", "currency": "USD" }, "createDate": "2022-07-21T20:16:35.092852Z", "updateDate": "2022-07-21T20:19:24.719313Z", "merchantId": "f1397191-56e6-42fd-be86-0a7b9bd91522", "merchantWalletId": "1000999922", "paymentIntentId": "6e4d4047-db14-4c09-b238-1215aee50d03", "settlementAmount": { "amount": "1.00", "currency": "USD" }, "fromAddresses": { "chain": "ETH", "addresses" : ["0x0d4344cff68f72a5b9abded37ca5862941a62050"] }, "depositAddress": { "chain": "ETH", "address": "0x97de855690955e0da79ce5c1b6804847e7070c7f" }, "transactionHash": "0x7351585460bd657f320b9afa02a52c26d89272d0d10cc29913eb8b28e64fd906" } }

For blockchains that require a 'memo' or 'address tag' (XLM, HBAR, etc.), the optional addressTag field will be present in the depositAddress object.

Gas Abstracted Payment: Network Fee Flexibility

In a regular payment, the customer pays for the gas fees (e.g., blockchain transaction fees). In a pull-based payment, the gas fees can be paid by you or by the customer. In the latter case, customers can pay via USDC on an EVM-compatible chain such as Ethereum rather than in a native token. For example, while making a USDC payment on Ethereum, the customer can pay for the gas in USDC rather than ETH. By doing so, the customer does not need native tokens and can solely rely on USDC for both gas and the actual payment. Gas fees and the preferences can be configured by working with Circle's customer success team.

In the case where a customer is paying for gas, Circle makes a real-time gas fee estimate and adds that in the total amount of payment that the customer authorizes. There are four steps in the flow:

1. Set up payment intent and acquire blockchain address for deposits
2. Receive a typed message for customers to sign and authorize the payment
3. Circle will receive the signature and broadcast the transaction for payment
4. Receive payment

Refer to the Pay with Crypto section to set up payment intent and retrieve a blockchain address. Once that has been completed, follow the steps below for gas decoupled payments.

1. Get a Typed Message for Signing

Call the GET: /payment/presign endpoint to fetch the typed message. The two required parameters are paymentIntentId , which can be retrieved from the GET: /paymentIntent response or webhook notification; and endUserAddress which is the consumer wallet address.

cURL curl --location 'https://api-sandbox.circle.com/v1/payments/presign?paymentIntentId=6e4d4047-db14-4c09-b238-1215aee50d03&endUserAddress=0x8381470ED67C3802402dbbFa0058E8871F017A6F' \ --header 'X-Requested-Id: {GUID}' \ --header 'Authorization: Bearer {YOUR_API_KEY}' Response { "data": { "typedData": { "domain": { "name": "USD Coin", "verifyingContract": "0x07865c6e87b9f70255377e024ace6630c1eaa37f", "version": "2", "chainId": 5 }, "message": { "from": "0x8381470ED67C3802402dbbFa0058E8871F017A6F", "to": "0xdB055877e6c13b6A6B25aBcAA29B393777dD0a73", "value": "5003456", "validAfter": 1675317371, "validBefore": 1675107967, "nonce": "0x4f716a578b96592599c3ca7b2a46453df87511d4809f608cec982d6e83632604" }, "totalAmount": { "amount": "3.14", "currency": "USD" }, "feeChargeModel": "endUser", "networkFeeQuote": { "quoteId": "5d0bb4e7-bd83-44dd-b8f5-53ddd5f35a3a", "amount": { "amount": "3.14", "currency": "USD" }, "expiresAt": "2023-02-02T06:01:21.152737680Z" }, "types": { "EIP712Domain": [{ "name": "name", "type": "string" }, { "name": "version", "type": "string" }, { "name": "chainId", "type": "uint256" }, { "name": "verifyingContract", "type": "address" }], "TransferWithAuthorization": [{ "name": "from", "type": "address" }, { "name": "to", "type": "address" }, { "name": "validAfter", "type": "uint256" }, { "name": "validBefore", "type": "uint256" }, { "name": "nonce", "type": "byte32" }] }, "primaryType": "TransferWithAuthorization" } } }

2. Customers Sign the Transaction From Their Wallet

You will need to trigger the customer's wallet using the response from /presign .

For example, if you use thesignTypedData function from wagmi , you will need to fill out the domain, types and values fields.

Once the customer signs into the wallet, you will receive a string of raw signatures.

3. Create a Crypto Payment with Customers' Signature

With a customer's raw signature, Circle can create a crypto payment with the POST /payment/crypto endpoint. After the payment has been created, Circle will be able to track the payment status following the instructionshere .

cURL curl --location 'https://api-sandbox.circle.com/v1/payments/crypto' \ --header 'X-Requested-Id: {GUID}' \ --header 'Authorization: Bearer {YOUR_API_KEY}' \ --header 'Content-Type: application/json' \ --data '{ "idempotencyKey": "ba943ff1-ca16-49b2-ba55-1057e70ca5c7", "paymentIntentId": "4e9fa5b1-3964-4f02-a7ba-811cc5d94be1", "protocolMetadata": { "type": "TransferWithAuthorization", "metaTxNonce": "0xfdb476566b75311fdd14444e6a77630c36e653a3e255adcaa7c34f3babc1de76", "signatureValidAfter": "1675104393", "signatureValidBefore": "1675107967", "rawSignature": "0xcff7ef7a24b88d83fa3d6e81b41c9cef19cc0119c085a6ef98cb1b6bc9436a9f18dcb2d46b9cb4d31a7031466b450bbe1e6c0230c5503c7a68e04055b4be0cbc1b" }, "amount": { "amount": "3.14", "currency": "USD" }, "source": { "address": "0xdB055877e6c13b6A6B25aBcAA29B393777dD0a73", "type": "blockchain" }, "destination": { "address": "0xda1ab716f7f7b3cb036a7fd74e5ca852126834c1", "chain": "ETH" }, "quoteId": "c6ac001e-9812-4bc1-8dc3-1549b5adaa23" }' Response { "data": { "id":"27827c45-6a3e-4012-8eaa-f6d712f70c2e", "type":"payment", "status":"pending", "amount":{ "amount":"10.00", "currency":"USD" } "createDate":"2023-02-14T01:21:07.116369Z", "updateDate":"2023-02-14T01:21:07.116371Z", "merchantId":"a49f9b1d-75e0-44a9-b8d2-4293b3f11ebd", "merchantWalletId":"1000563095", "paymentIntentId":"dfcf96ab-dd3f-417e-b009-ef899ee4e8da", "fromAddresses":{ "chain":"ETH", "addresses":["0xbdE3d06b700A5f622767c85e79e42014C5376B6B"] } "depositAddress":{ "chain":"ETH", "address":"0x5f1090b5db3e00e13e94b93055b3d1c4254e1720" } } }

4. Payment Received

Refer to thepayments received section for more details. Updated2 months ago *Table of Contents * * 1. Pay with Crypto * * 2. Acquire blockchain address customer will pay to * * * Option 1: Webhook Notification * * * Option 2: Poll Payment Intent Endpoint * * 3. Enable customer payment * * 4. Receive payment from Circle * * * Option 1: Webhook Notifications * * * Option 2: Retrieve Payment Intent and Payment * * Gas Abstracted Payment: Network Fee Flexibility * * * 1. Get a Typed Message for Signing * * * 2. Customers Sign the Transaction From Their Wallet * * * 3. Create a Crypto Payment with Customers' Signature * * * 4. Payment Received