

Week 2 of 6

2 Wallets & on-chain basics

Task

Get devnet SOL and USDC, send USDC to a partner, read the transaction on Orb.

Blog post

How I interacted with my first Solana dApp.

Today

1

Keys

2

Accounts

3

Tokens

4

Transactions



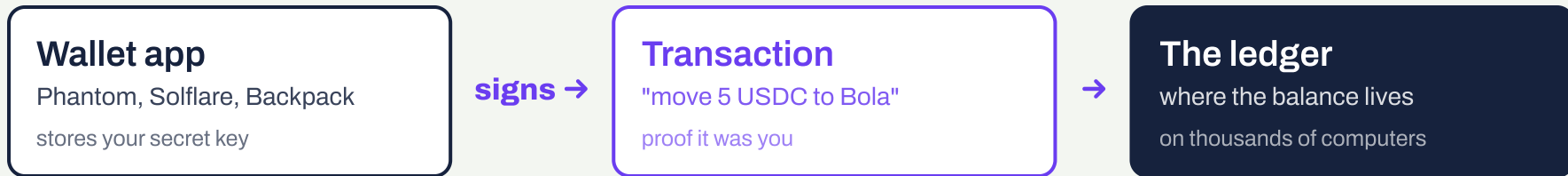
Send devnet
USDC



Keys

Your wallet doesn't hold coins.

A wallet is a keyring, not a purse



Lose the phone, keep the phrase: **your money is fine.**

Lose the phrase: **nobody can help you.**

Twelve words, every address

Recovery phrase

example only, never real

1. orbit 2. canvas 3. pepper
4. tribe 5. ladder 6. sunny
7. velvet 8. mango 9. harbor
10. oxygen 11. quiet 12. zebra

→
one-way maths

Seed

5eb00bbd...a3c7f1

→
path

Derivation path

m/44'/501'/0'/0'

501 is Solana's coin number

Keypair

secret key

public key = address

7xKp...Qm2a

Same 12 words, path m/44'/501'/1'/0' → a second address. One phrase can restore every account.

Sign something

A real keypair, made in this browser. Sign, then edit the message.

Generate a keypair

Public key = your address

share freely

not yet

Secret key

never share

Message

Pay Bola 5 USDC

Sign with secret key

Signature

not yet

Who holds the key?

Custodial

An exchange holds it for you.

Easy recovery. They can freeze you. If they fail, you queue with everyone else (FTX).

Non-custodial

You hold it. Phantom, Solflare, Backpack.

Full control. No password reset. Hardware wallets keep the key off your laptop.

The one rule

Nobody legitimate will ever ask for your recovery phrase. Not support, not a mentor, not a "wallet validator" site.

'Phantom Support' DMs you: there's a sync bug, enter your recovery phrase at their link to fix it. What do you do?

A Enter it, support needs it to fix things

B Send only the first 6 words

C Ignore and block. It's a theft attempt

D Move funds first, then enter it



Accounts

Everything on Solana is one.

One shape for everything

Your wallet

A token mint

Your token account

A program

address a public key, like 7xKp...Qm2a

lamports	2,500,000,000 (2.5 SOL)	SOL balance, in billionths
data	empty	bytes the owner program reads
owner	System Program	the only program allowed to change it
executable	false	is this account code?

Just SOL. No data. Owned by the System Program, which only lets your key move it.

Rent is a deposit, not a fee

Create an account

Lock a small SOL deposit for the space it uses.

token account: 0.00203928 SOL

Use it

The deposit sits there. Nothing is charged over time.

Close it

The deposit comes back to you.

+0.00203928 SOL

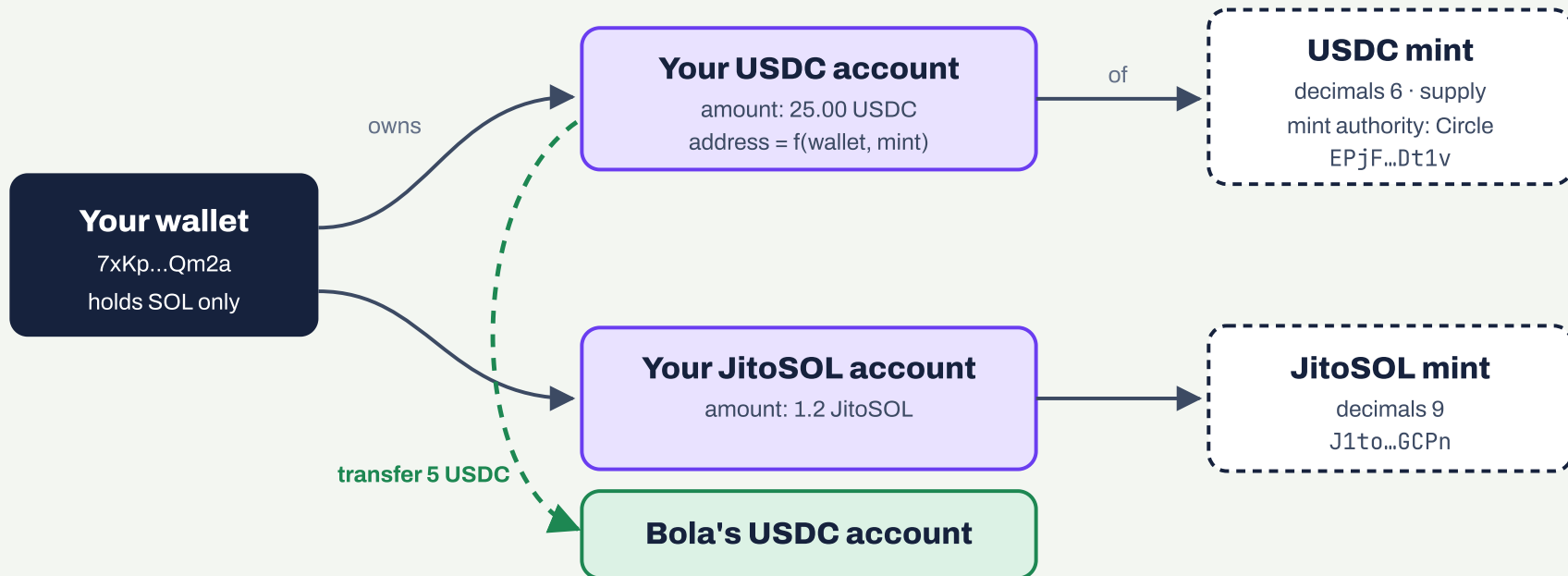
That's why your **first** USDC send to someone costs more than the second.



Tokens

One program, many mints.

Wallet → token account → mint



Which one is USDC?



USDC

USD Coin

BRjpCHtyQLNCo8gqRUr8jtdAj5AjPYQaoqbvc
ZiHok1k



USDC

USD Coin

4zMMC9srt5Ri5X14GAgXhaHi13GnPAEERYPJg
ZJDncDU



USDC

USD Coin

USDCvG8kT3pQf2X5mHnB7rJ1sLw9eYcZ4aKdM
6tPq2x

Same name, same logo. Which one is Circle's devnet USDC?

Two token programs

SPL Token

The original. Mint, transfer, burn, freeze.

USDC, most older tokens.

Token-2022

Same basics, plus optional **extensions**.

PYUSD, USDG, many newer tokens.

Transfer fee

Transfer hook

Permanent delegate

Non-transferable

Confidential transfer

Metadata

Both programs have a **mint authority** and a **freeze authority**. Check who holds them before you buy anything.

Where does your USDC balance actually live?

A Inside your wallet account

B Inside the USDC mint account

C In your own USDC token account

D In the Token Program's code

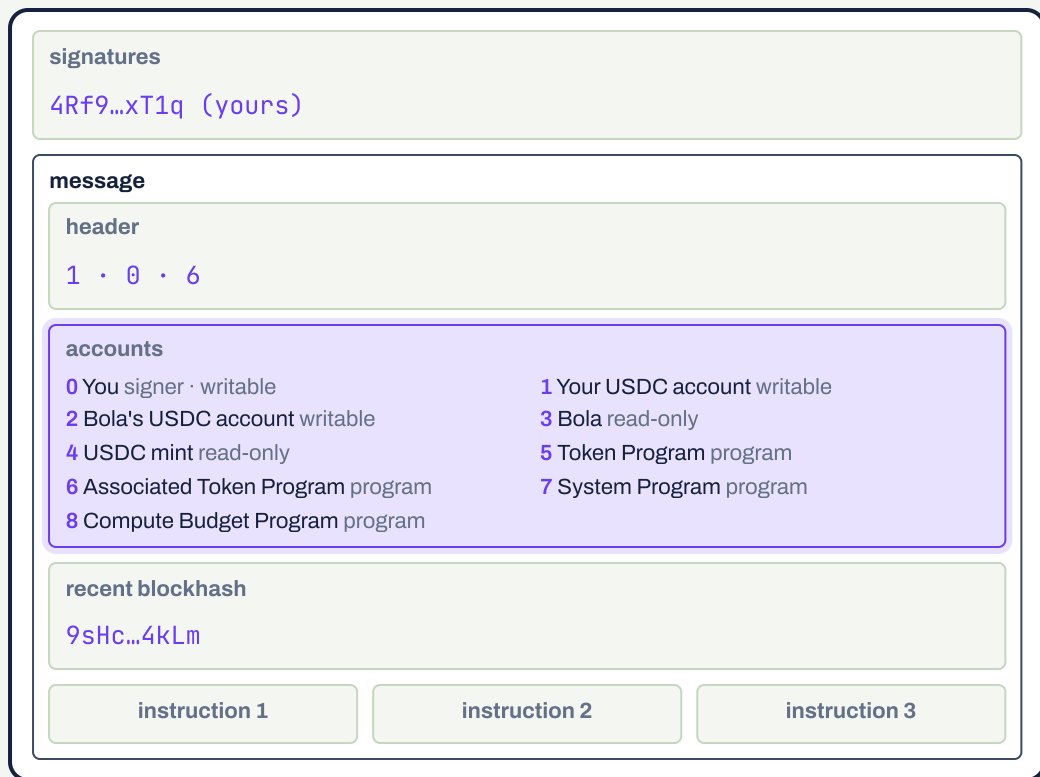


Transactions

What you actually sign.

Inside a transaction

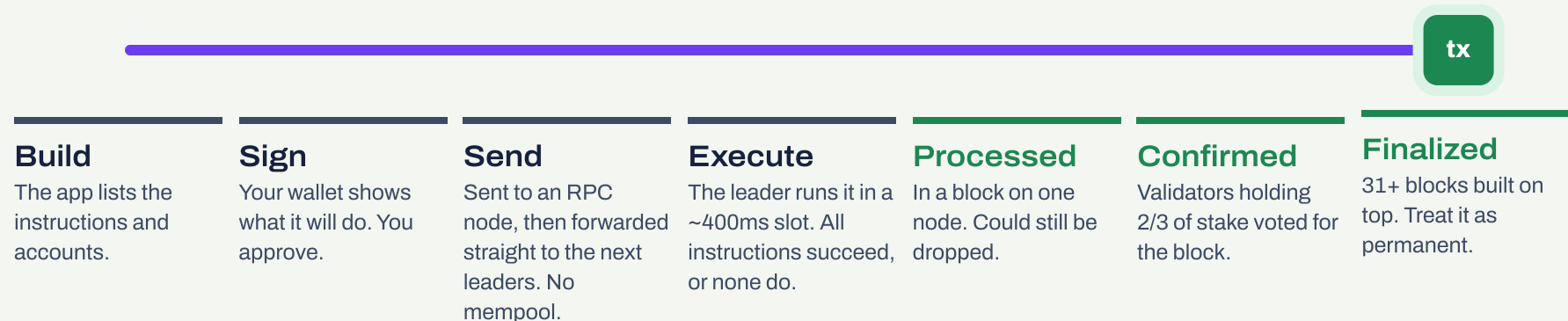
Sending 5 USDC to Bola, who has never held USDC. Hover each part.



Account list

Every account the transaction will touch, listed up front. This is what lets Solana run transactions in parallel.

The life of a transaction



tx

Build

The app lists the instructions and accounts.

Sign

Your wallet shows what it will do. You approve.

Send

Sent to an RPC node, then forwarded straight to the next leaders. No mempool.

Execute

The leader runs it in a ~400ms slot. All instructions succeed, or none do.

Processed

In a block on one node. Could still be dropped.

Confirmed

Validators holding 2/3 of stake voted for the block.

Finalized

31+ blocks built on top. Treat it as permanent.

What it costs

Signatures 1



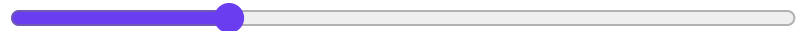
Compute limit 200,000 CU



Priority price 10,000 μ -lamports/CU



SOL price (example) \$150



☒ Creates the recipient's token account

Base fee | 0.000005 SOL

Priority fee | 0.000002 SOL

Rent deposit  0.00203928 SOL

Fee (gone)

\$0.00105

Deposit (refundable)

\$0.306

Your swap fails because the price moved too much. What happens to the fee?

A Refunded automatically

B Charged anyway

C Charged only if you retry

D Paid by the DEX

Reading it on an explorer

orb.helius.dev/tx/4Rf9...xT1q?cluster=devnet

Signature

4Rf9kQ2...mD8xT1q

Result

Success

Slot 412,880,113 · 14 Oct 2026 10:02

Fee

0.000005 SOL

1 signature, no priority fee

SOL balance changes

You

-0.002044 SOL

Bola's USDC account (new)

+0.002039 SOL

Token balance changes

You · USDC

-5.00

Bola · USDC

+5.00

Instructions

#1 Associated Token Program · create (idempotent)

#2 Token Program · transferChecked 5.000000 USDC

Everything an analyst needs is on one page. Write down what you can and can't interpret.

Sample data for teaching.



This week's task

Your first devnet transaction.

The task

Users

1. Install **Phantom** (or Solflare, Backpack). Turn on **Testnet Mode** and pick **Solana Devnet**. Check the banner.
2. Get devnet SOL: **faucet.solana.com**
3. Get devnet USDC: **faucet.circle.com**
4. Send USDC to a partner.
5. Open it on **Orb** with `?cluster=devnet`. Find: the **fee**, the **token instruction**, the **token account it created**.

Builders

Everything on the left, plus:

1. Create your own token with the `spl-token` CLI.
2. Mint 1,000 to yourself.
3. **Revoke the mint authority.**
4. Look at the mint account again. What changed?

Builders: your own token

```
1  solana config set --url devnet
2  solana-keygen new -o ~/devnet.json && solana airdrop 2
3
4  spl-token create-token
5  # → prints your new MINT address
6
7  spl-token create-account <MINT>
8  # → creates your token account (ATA) for it
9
10 spl-token mint <MINT> 1000
11 spl-token supply <MINT>
12
13 spl-token authorize <MINT> mint --disable
14 spl-token display <MINT> # mint authority: (not set)
```

Blog post

"How I interacted with my first Solana dApp."

Include **one transaction link** and an **annotated explorer screenshot**.



Teacher prep

For the mentor. Skip in class.

You, before class

- Make a **fresh wallet** and run the whole task on it. Note which faucet rate-limits you.
- **Pre-fund a teacher wallet** with devnet SOL and USDC. Faucets break when a whole room shares one Wi-Fi.
- Prepare one whiteboard diagram: **wallet → token account (ATA) → mint**. Weeks 3 and 4 build on it.
- Read Solana's intro to transactions in the official docs.

Faucet limit

Circle's faucet gives 20 USDC every 2 hours per address (confirmed in Circle's FAQ, 14 Oct 2026). Plan the transfer task around that.

Words from week 2

Keypair

A secret key and the public key made from it.

Recovery phrase

12 or 24 words that rebuild every key.

Account

lamports, data, owner, executable. Everything is one.

Rent

A refundable deposit for an account's storage.

Token account (ATA)

Holds one wallet's balance of one token.

Blockhash

Recent block ID that makes a transaction expire.

Address

A public key written in base58.

Signature

Proof that a key holder approved this exact message.

Lamport

One billionth of a SOL.

Mint

The account that defines a token.

Instruction

One call to one program inside a transaction.

Devnet

Solana's free test network for apps.