

# Week 4 - Solana Jargon, Programming model, Tokens [23 Aug 2024]

**Timing** - Every Friday 8:00pm to 10:00pm IST

**Class Slides** - [Programs, Accounts and the Token Program \(100xdevs.com\)](#)

**Class Video** - [Take your development skills from 0 to 100 and join the 100xdevs community](#)

**Assignment** - [Programs, Accounts and the Token Program \(100xdevs.com\)](#)  
- [Token 22 with metadata \(100xdevs.com\)](#)

**Syllabus** - [Notion – The all-in-one workspace for your notes, tasks, wikis, and databases. \(100xdevs.com\)](#)

## Goal of the Session

### 1. Introduction to Solana and Its Advantages Over Bitcoin and Ethereum

- **Understanding Solana's Purpose:** Explore why Solana was introduced and how it aims to improve upon the limitations of Bitcoin and Ethereum.
- **Comparative Analysis:**
  - **Bitcoin:** Primarily a digital currency, focusing on decentralization and security.
  - **Ethereum:** Enhances Bitcoin's functionality by introducing smart contracts.
  - **Solana:** Further expands capabilities by offering a high-performance blockchain, supporting thousands of transactions per second, with a unique architecture that optimizes for scalability.

### 2. Architecture of Solana

- **Accounts on Solana:** Explanation of the various types of accounts on Solana:
  - **Wallet Accounts:** Used to store and manage cryptocurrency.
  - **Data Storage Accounts:** Store various types of data on-chain, enabling more complex applications.
- **Scalability and Performance:** How Solana's architecture allows for millions of accounts and high transaction throughput.

### 3. Setting Up the Solana Development Environment

- **Installing Solana CLI:** Step-by-step instructions on how to install the Solana Command Line Interface (CLI) locally to interact with the blockchain and deploy smart contracts.

#### 4. Data Storage: Web2 vs. Solana

- **Web2 Data Storage:** Overview of how data is traditionally stored in centralized servers and databases.
- **Solana Data Storage:** Understanding how data is stored on the blockchain in Solana smart contracts, emphasizing the decentralized nature of storage.

#### 5. Building Smart Contracts on Solana

- **Data Persistence in Smart Contracts:** Discuss the necessity of data storage in smart contracts (e.g., user data, first names, last names) and how it's handled in Solana.
- **Creating Your Own Token:**
  - **JavaScript:** Instructions on how to create tokens using JavaScript.
  - **Rust/Python/Go:** Guidance on creating tokens using Rust, Python, or Go, highlighting the versatility and support for multiple programming languages on Solana.

#### 6. Solana's Token Program: An Overview

- **Token 2022 Program:** Explanation of the Token 2022 program, its features, and how it expands the functionality and utility of tokens on the Solana blockchain.

## Beyond Bitcoin

### The Limitations of Bitcoin

- Bitcoin primarily serves as a decentralized currency.
- It was not designed to support complex applications or diverse use cases.

### The Rise of Alternative Blockchains

- **Post-2012 Developments:** Various blockchains emerged, each tailored for specific purposes, such as lending protocols or decentralized exchanges.
- **Challenges Faced:**
  - **Fragmentation:** Each blockchain operated independently with its own set of miners and consensus mechanisms.
  - **Cold Start Problem:** New blockchains struggled to gain traction and security, unable to match Bitcoin's network size and robustness.

# Ethereum: A Revolutionary Solution

- Ethereum introduced the concept of smart contracts, allowing developers to build decentralized applications (dApps) directly on the Ethereum network.
- **Advantages of Ethereum:**
  - **No Cold Start Problem:** Developers could leverage Ethereum's existing decentralized network, avoiding the initial growth challenges faced by standalone blockchains.
  - **Flexibility and Innovation:** Enabled a wide range of applications, from finance to gaming, all on a single, secure blockchain.

## ▼ Example: Counter Contract in Solidity

```
// SPDX-License-Identifier: MIT
pragma solidity ^0.8.0;

contract Counter {
    uint public count;

    // Constructor to initialize count
    constructor() {
        count = 0;
    }

    // Function to increment the count
    function increment() public {
        count += 1;
    }

    // Function to decrement the count
    function decrement() public {
        require(count > 0, "Count cannot be negative");
        count -= 1;
    }

    // Function to get the current count
    function getCount() public view returns (uint) {
        return count;
    }
}
```

# Solana: A New Frontier

- **Smart Contracts vs. Programs:** What Ethereum calls `smart contracts`, Solana refers to as `programs`.

- Solana offers similar capabilities to Ethereum but with significantly faster transaction speeds and scalability, high throughput and low latency.
- The design and operation of Solana's programs differ from traditional blockchains, providing a distinct approach to building and running decentralized applications.

## Accounts in Solana

### What is an Account?

- In Solana, an account is a data structure that includes a public-private key pair (using the [ed25519](#) elliptic curve).
- **Types of Accounts:**
  - **Wallet Accounts:** These accounts represent user wallets that can hold **Lamports** (Solana's native currency).

#### ▼ Example: Wallet Account [Only Lamports]

[Account DNp2hBynGWFWomptmxISzhhTYjGxDsyu43RBrC6TzuMW | Solscan](#)

- **Data Accounts:** These accounts store data on the blockchain and can be used for various decentralized applications.

#### ▼ Example: Data Account [Data and Lamports]

[Transaction History | 4GQsAP5jYi5ysGF1GEEnWiV3zJHZLRcLWhLCSuim6aAkL | Solana](#)

- **Program Accounts:** These are special accounts that store executable code, allowing smart

contracts (known as "programs" in Solana) to run on the blockchain.

#### ▼ Example: Program Account [Data, Lamports and Executable code]

[Transaction History | TokenkegQfeZyiNwAJbNbGKPFXCWuBvf9Ss623VQ5DA | Solana](#)

Overview	
Address	<a href="#">TokenkegQfeZyiNwAJbNbGKPFXCWuBvf9Ss623VQ5DA</a>
Address Label	Token Program
Balance (SOL)	0.93408768
Allocated Data Size	134080 byte(s)
Assigned Program Id	<a href="#">BPF Loader 2</a>
Executable	Yes

## Rent on Solana

- **Purpose of Rent:** To prevent the blockchain from being clogged with inactive or unnecessary data, Solana charges rent for storing data.
  - **Rent Calculation:** Rent is based on the storage size and the duration the account remains on the blockchain.
  - **Rent Exemption:** If an account maintains a balance above a certain threshold, it becomes "rent-exempt," meaning it does not incur further rent charges. The rent paid is refundable when the account is closed.



[What is Rent on Solana and How to Calculate it | QuickNode](#)

## Efficient Storage Management

- **Incentivizing Minimal Storage:** The rent model encourages users to store only the necessary data, reducing blockchain bloat.
- **Removing Inactive Accounts:** Accounts that fail to pay the required rent are eventually removed from the ledger, ensuring efficient use of blockchain resources.

## Calculating Rent-Exempt Threshold

- **Methods to Calculate:**

1. **Solana CLI:** Command-line interface tools.
2. **Solana Web3.js Library:** JavaScript library for interacting with the Solana blockchain.

**3. Anchor's Space Constraint:** A framework-specific approach for calculating space requirements for accounts.

# Install Solana CLI

## MacOS & Linux

The below commands run for WSL as well.



[Install the Solana CLI | Solana Validator \(solanalabs.com\)](#)

## Windows

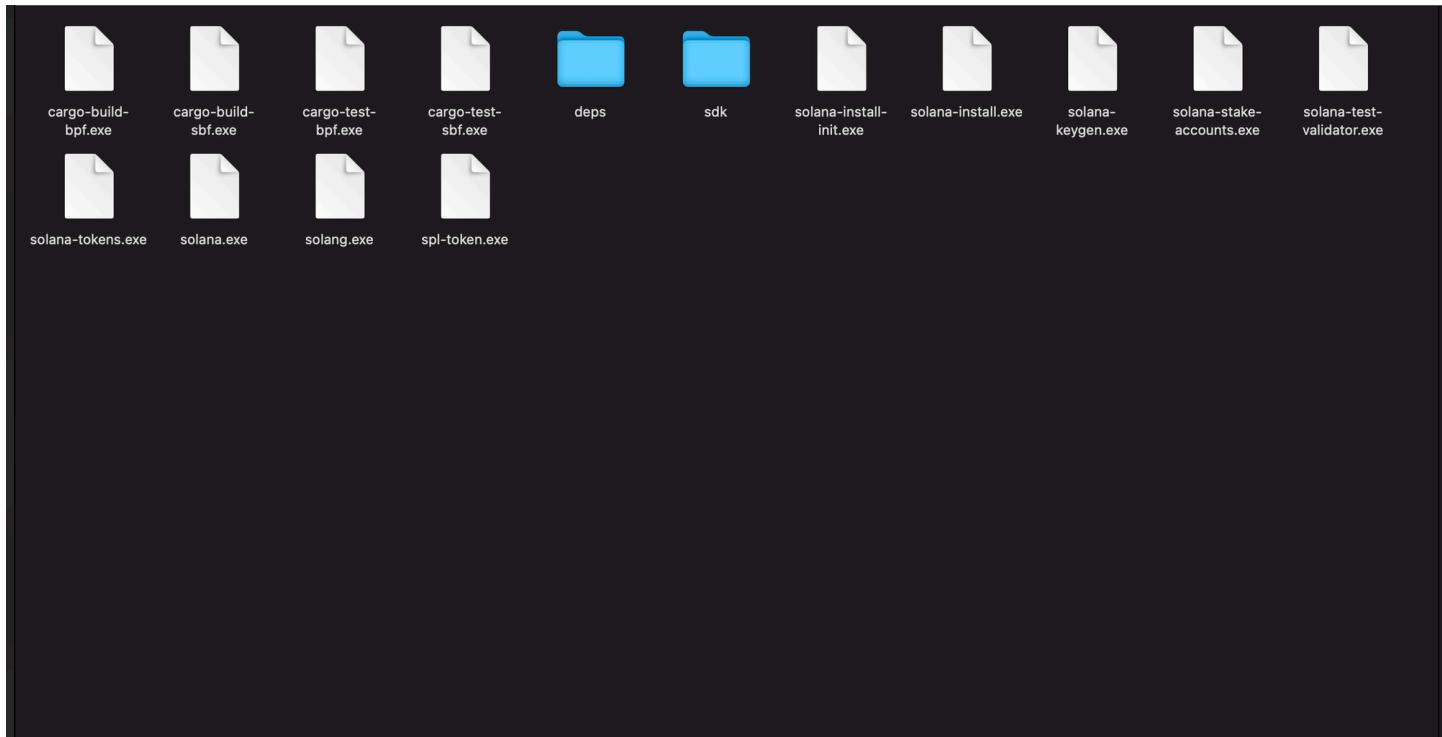
Download the highlighted folder from - <https://github.com/solana-labs/solana/releases>

▼Assets 15

<a href="#">sbf-sdk.tar.bz2</a>	18.2 KB	2 weeks ago
<a href="#">solana-install-init-aarch64-apple-darwin</a>	10.1 MB	2 weeks ago
<a href="#">solana-install-init-x86_64-apple-darwin</a>	10.7 MB	2 weeks ago
<a href="#">solana-install-init-x86_64-pc-windows-msvc.exe</a>	9.23 MB	2 weeks ago
<a href="#">solana-install-init-x86_64-unknown-linux-gnu</a>	22.1 MB	2 weeks ago
<a href="#">solana-release-aarch64-apple-darwin.tar.bz2</a>	147 MB	2 weeks ago
<a href="#">solana-release-aarch64-apple-darwin.yml</a>	96 Bytes	2 weeks ago
<a href="#">solana-release-x86_64-apple-darwin.tar.bz2</a>	155 MB	2 weeks ago
<a href="#">solana-release-x86_64-apple-darwin.yml</a>	95 Bytes	2 weeks ago
<a href="#">solana-release-x86_64-pc-windows-msvc.tar.bz2</a>	77.7 MB	2 weeks ago
<a href="#">solana-release-x86_64-pc-windows-msvc.yml</a>	98 Bytes	2 weeks ago
<a href="#">solana-release-x86_64-unknown-linux-gnu.tar.bz2</a>	329 MB	2 weeks ago
<a href="#">solana-release-x86_64-unknown-linux-gnu.yml</a>	100 Bytes	2 weeks ago
<a href="#">Source code (zip)</a>		2 weeks ago
<a href="#">Source code (tar.gz)</a>		2 weeks ago

(8) 9 people reacted

Unzip and you should see all the .exe files



## Confirm by running commands

### ▼ Get Solana version

```
solana --version
```

### ▼ Create your own keygen

```
solana-keygen new
```

### ▼ Get your Solana address

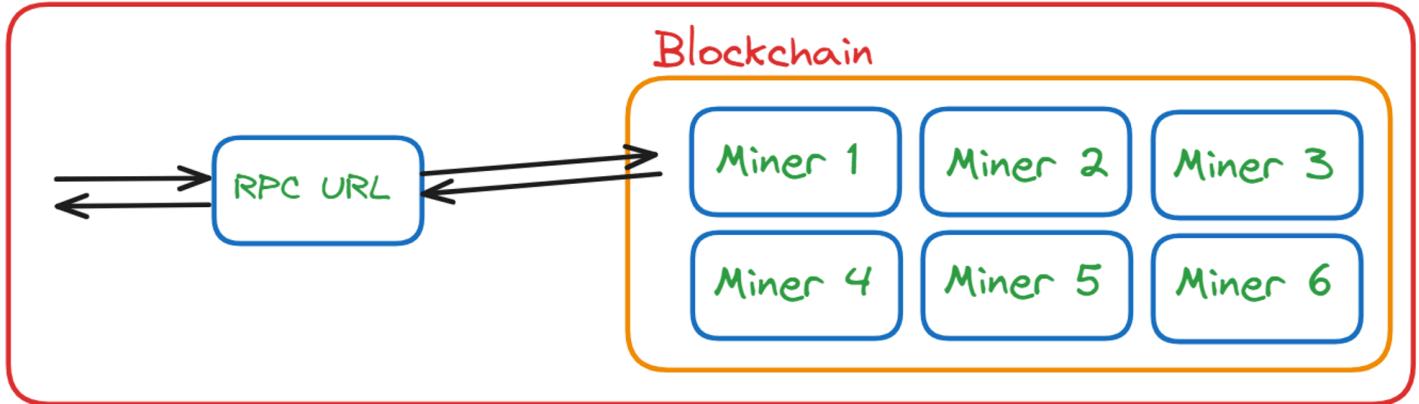
```
solana address
```

### ▼ Get your config file

```
solana config get
```

## RPC URL, Testnet, Devnet, and Mainnet

## Devnet/Testnet/Mainnet



## What is an RPC URL?

- RPC (Remote Procedure Call) URLs allow applications like browsers, wallets, and users to interact with the Solana blockchain.
- By sending requests to an RPC URL, you can query the blockchain and perform various operations.
- **How to Use RPC URLs:**
  - **Mainnet:** <https://api.mainnet-beta.solana.com>
  - **Devnet:** <https://api.devnet.solana.com>
  - **Testnet:** <https://api.testnet.solana.com>



[Clusters and Public RPC Endpoints | Solana](#)

## Changing RPC URLs

Use the following commands to set the appropriate RPC URL for your environment:

```
# Set RPC URL for Mainnet
solana config set --url https://api.mainnet-beta.solana.com

# Set RPC URL for Devnet
solana config set --url https://api.devnet.solana.com

# Set RPC URL for Testnet
solana config set --url https://api.testnet.solana.com
```

## Solana Networks

- **Mainnet:** The live Solana blockchain where real transactions occur. It's secure and used for deploying production applications.
- **Testnet:** A testing environment that mimics the mainnet but is used for testing applications before deploying them to the mainnet. It does not use real SOL.
- **Devnet:** A development environment similar to the testnet but specifically for developers to experiment and test applications. You can freely request SOL from the [Solana Devnet Faucet](#) for testing purposes.

## Using Devnet and Testnet

- **Airdrop SOL:** On Devnet, you can use the faucet to receive free SOL for testing.
- **Local Testing:** Use the Solana Test Validator to start a local validator for testing applications without relying on external networks.

`solana-test-validator`

## Solana Explorer

- **Explorer Tool:** You can view transactions, account balances, and more on the Solana blockchain using the [Solana Explorer](#), which supports Devnet, Testnet, Mainnet, and custom RPC URLs.

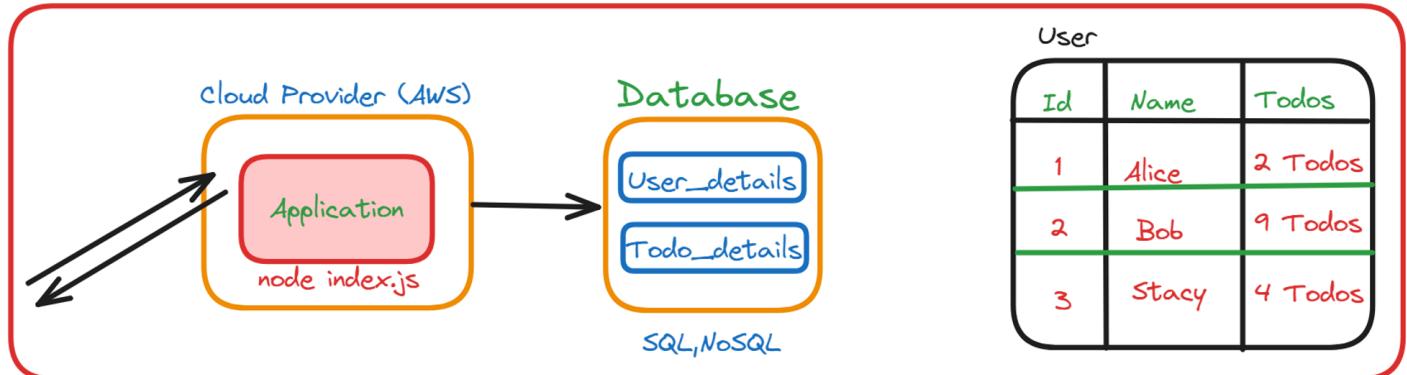
The screenshot shows the Solana Explorer (Beta) interface. At the top, there's a navigation bar with tabs for "Cluster Stats", "Supply", and "Inspector". To the right of the tabs is a button labeled "Choose a Cluster" with options: "Mainnet Beta" (selected), "Testnet", "Devnet", and "Custom RPC URL". Below the tabs is a search bar with placeholder text "Search for blocks, accounts, transactions, programs, and tokens". Underneath the search bar are two large boxes: "Circulating Supply" showing "466.3M / 583.4M" with "79.9% Is circulating" and "Active Stake" showing "380M / 583.4M" with "Delinquent stake: 0.0%". To the right of these boxes is a "Developer Settings" section with a toggle switch for "Enable custom url param" and a descriptive text: "Enable this setting to easily connect to a custom cluster via the \"customUrl\" url param." On the left side, there's a "Live Cluster Stats" sidebar with various metrics: Slot, Block height, Cluster time, Slot time (1min average), Slot time (1hr average), Epoch, and Epoch progress. The date "Aug 29, 2024 at 12:18:10 Coordinat" is also visible.

## Web2 Data Model vs. Solana Data Model

### Web2 Data Model

- Applications are typically deployed on cloud providers with backend code and data storage separated.
- **Databases:** Use SQL or NoSQL databases to store user data.
- **Data Management:** Adding or deleting a user involves simply adding or removing a row in the database.

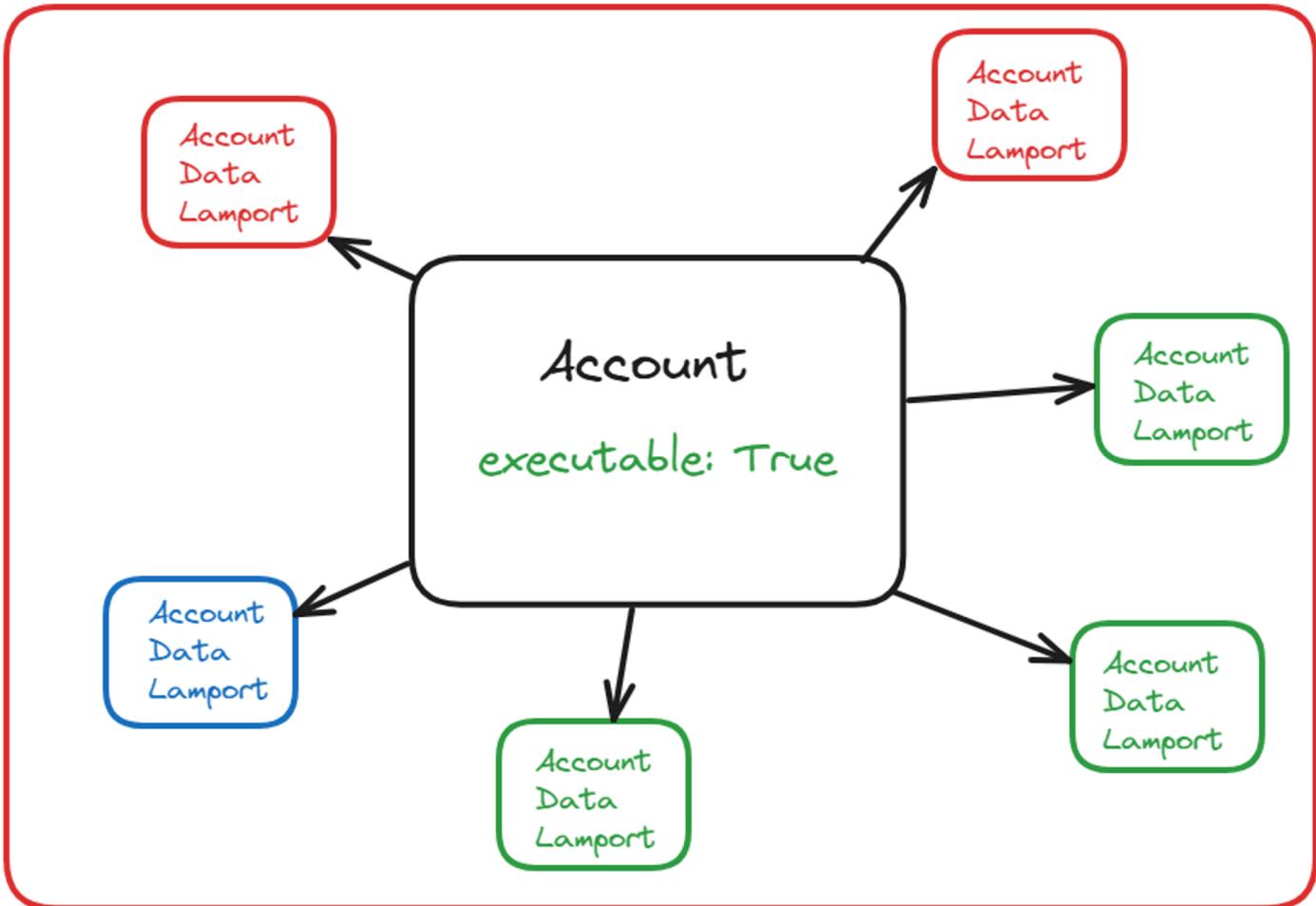
## Web 2 Data Model



## Solana Data Model

- **Smart Contracts:** In Web3, smart contracts are equivalent to backend applications in Web2. These are deployed on the blockchain.
- **Program Storage:** On Solana, smart contracts (called "programs") are stored in **executable accounts**.
- **Data Storage:** Unlike Ethereum, where data and smart contract code are stored together, Solana separates them:
  - **Data Accounts:** Store data independently from programs.
  - **Executable Accounts:** Store the smart contract code (programs).

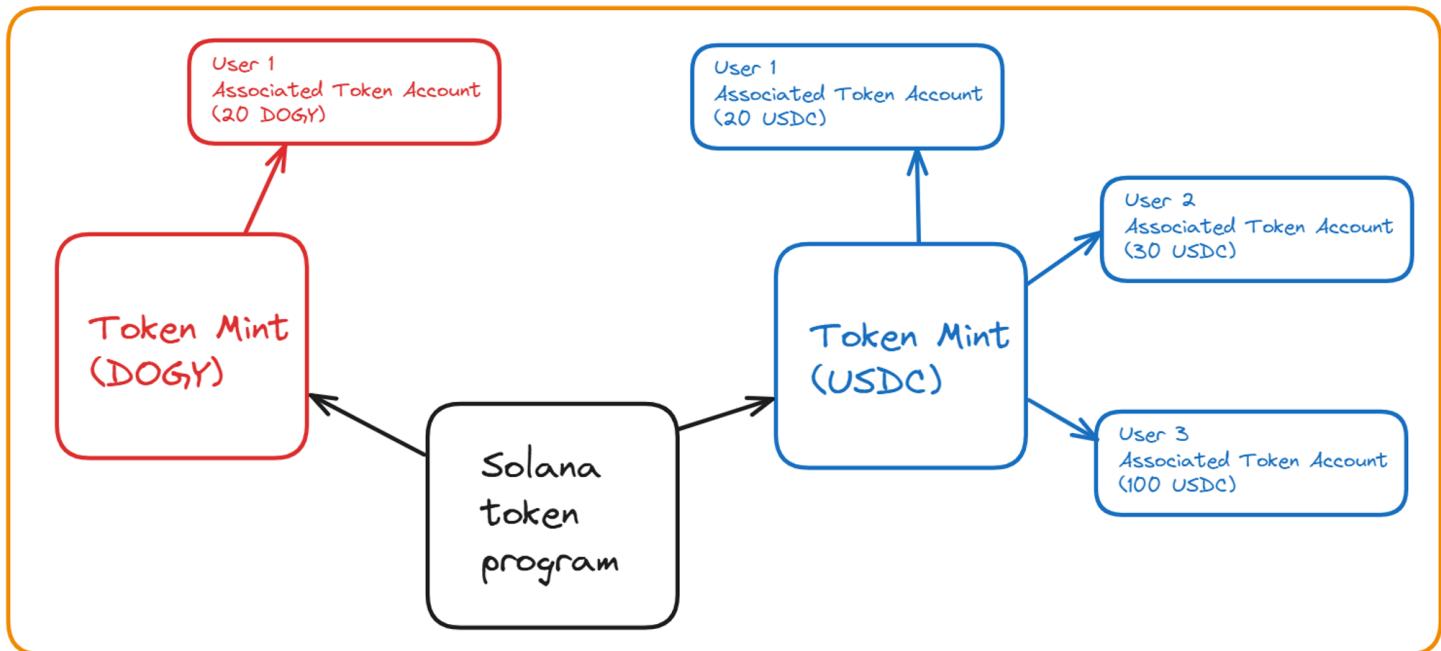
# Web 3 (Smart Contract) Solana Data Model



## Key Differences and Implications

- User Account Management:
  - In Solana, each user requires a separate account, which adds complexity compared to simply adding a row in a Web2 database.
  - **User-Paid Fees:** The responsibility for creating and funding these accounts, including paying for data rent, is delegated to the user.
  - **Decentralized Fees:** Users pay the fees for their individual accounts, not the program itself. If an account is closed, any remaining rent is refunded to the user.

## Token Program on Solana



## Creating Tokens: Ethereum vs. Solana

- Ethereum:
  - To create a token, you must deploy your own smart contract.
  - Ethereum provides a standard template (ERC-20), but each token requires its own contract to be deployed on the blockchain.
- Solana:
  - Solana simplifies this process with a **single, pre-deployed Token Program**.
  - Instead of deploying a new contract, you only need to create a token account under this program.
  - This significantly reduces complexity and deployment costs.

## Understanding Tokens

- Tokens represent one of the primary use cases of blockchain technology, acting as digital assets or currencies that can be transferred or traded on the blockchain.
  - Examples: USDC and USDT are popular stablecoins that have found significant market adoption. These tokens do not have their own blockchain; instead, they operate as smart contracts on existing blockchains like Ethereum and Solana.
- Why Use Existing Blockchains?
  - Tokens like USDC and USDT leverage the security and infrastructure of established blockchains such as Solana and Ethereum. This allows them to function without maintaining their own blockchain or miners.

- The **Token Program** is essentially a mapping from an account to a number, representing the balance of tokens held by an account.



Explore various tokens and their market data on [CoinMarketCap](#).

## Solana's Token Program

- **Centralized Token Program:**
  - Solana engineers recognized the importance of tokens and created a dedicated **Token Program**.
    - This program is pre-deployed on the Solana blockchain, simplifying token creation.
- **Mint Accounts:**
  - When creating a token on Solana, you establish a **mint account** under the Token Program.
  - A mint account functions like a bank for your token, overseeing its supply but without executing any code. It does not run transactions or logic on its own; instead, it is responsible for managing the minting and overall supply of tokens.
- **Associated token account:**
  - Associated Token Account is a token account whose address is deterministically derived using the owner's address and the mint account's address.
    - ▼ When you created an **associated token account**, you actually created a PDA.
      - <https://github.com/solana-labs/solana-program-library/blob/master/associated-token-account/program/src/lib.rs#L71>
      - JS - <https://github.com/solana-labs/solana-program-library/blob/ab830053c59c9c35bc3a727703aacf40c1215132/token/js/src/state/mint.ts#L171>



Program derived accounts (PDAs) - [Programming on Solana | Program Derived addresses and Cross Program Invocation - YouTube](#)



Solana program library: [solana-labs/solana-program-library](https://github.com/solana-labs/solana-program-library): A collection of Solana programs maintained by Solana Labs (github.com)

## Creating a Token using CLI

- ▼ Create a new CLI wallet, you can continue with your wallet as well

```
solana-keygen new
```

## ▼ Set the RPC URL

```
# Set RPC URL for Devnet
solana config set --url https://api.devnet.solana.com
```

## ▼ Airdrop yourself some SOL

```
solana airdrop 2
```

## ▼ Check your balance

```
solana balance
```

## ▼ Create token mint

```
spl-token create-token
```

## ▼ Will get an output something like this,

```
Creating token HDLXuxjbXsrE7Dqecjb3EkGh13u4X2J9Fp3ouo9x6nBs under program TokenkegQfeZyiNwAJbNbGKPFXCWuBvF9Ss623VQ5DA
Address: HDLXuxjbXsrE7Dqecjb3EkGh13u4X2J9Fp3ouo9x6nBs
Decimals: 9
Signature: 2DGKrxjBha4FP236J3K8EwDxiNs8EAK28yDhN7h3fU6A2RYNJM4J7r27p1qFT2njUqwcytJWR4A4BZDGV2FAYniHJ
```

**Decimals:** 1 coin can be broken down in  $10^9$  parts. If the decimal is 0, it creates a non-fungible token which cannot be broken in parts and sent.

## ▼ Verify token mint on chain

[Transaction History | HDLXuxjbXsrE7Dqecjb3EkGh13u4X2J9Fp3ouo9x6nBs | Solana](#)

Token Mint	
Address	HDLLXuxjbXsrE7Dqecjb3EkGh13u4X2J9Fp3ouo9x6nBs
Current Supply	0
Mint Authority	AvRTn81pWcYMPmYRt7LFNKcMmArpynayq8iPfk1qJ97s
Decimals	9

History   Transfers   Instructions

Transaction History   Refresh

## Postman Request

```
{
  "jsonrpc": "2.0",
  "id": 1,
  "method": "getAccountInfo",
  "params": ["HDLXuxjbXsrE7Dqecjb3EkGh13u4X2J9Fp3ouo9x6nBs"]
}
```

## Postman Response

```
{
  "jsonrpc": "2.0",
  "result": {
    "context": {
      "apiVersion": "2.0.5",
      "slot": 322383646
    },
    "value": {
      "data": "DK9N43Ug1H5x5LobZTrRuV3t3pHKC4jctCgnGHUZRZ37dyCwRzK8of8voCGRURpw9SQkqd\l",
      "executable": false,
      "lamports": 1461600,
      "owner": "TokenkegQfeZyiNwAJbNbGKPFXCWuBvf9Ss623VQ5DA",
      "rentEpoch": 18446744073709551615,
      "space": 82
    }
  },
  "id": 1
}
```

The screenshot shows the Postman application interface. At the top, there's a header bar with 'POST https://api.devnet.solana.com' and a status indicator 'No environment'. Below the header, the URL 'https://api.devnet.solana.com' is entered in the address bar. The main workspace has a 'Body' tab selected, showing a raw JSON request and response. The request body is identical to the one shown above:

```
1 {
2   "jsonrpc": "2.0",
3   "id": 1,
4   "method": "getAccountInfo",
5   "params": ["HDLXuxjbXsrE7Dqecjb3EkGh13u4X2J9Fp3ouo9x6nBs"]
6 }
```

The response body is also identical to the one shown above:

```
1 {
2   "jsonrpc": "2.0",
3   "result": {
4     "context": {
5       "apiVersion": "2.0.5",
6       "slot": 322383646
7     },
8     "value": {
9       "data": "DK9N43Ug1H5x5LobZTrRuV3t3pHKC4jctCgnGHUZRZ37dyCwRzK8of8voCGRURpw9SQkqd\l",
10      "executable": false,
11      "lamports": 1461600,
12      "owner": "TokenkegQfeZyiNwAJbNbGKPFXCWuBvf9Ss623VQ5DA",
13      "rentEpoch": 18446744073709551615,
14      "space": 82
15    }
16  },
17  "id": 1
18 }
```

At the bottom right of the response pane, there are status metrics: 'Status: 200 OK', 'Time: 354 ms', 'Size: 1.2 KB', and 'Save as example'.



Explore Solana program library: [solana-labs/solana-program-library](https://github.com/solana-labs/solana-program-library): A collection of Solana programs maintained by Solana Labs (github.com)

## ▼ Check the supply of the token

```
spl-token supply HDLXuxjbXsrE7Dqecjb3EkGh13u4X2J9Fp3ouo9x6nBs
```

## ▼ Create an **associated token account**

```
spl-token create-account HDLXuxjbXsrE7Dqecjb3EkGh13u4X2J9Fp3ouo9x6nBs
```

### Output:

Created an **associated token account** - CARa5swSHzqgxeLAL3WZMFFtB5PuRYeuzF4Pta4HfCxa for the **mint address** - HDLXuxjbXsrE7Dqecjb3EkGh13u4X2J9Fp3ouo9x6nBs

**Creating account** CARa5swSHzqgxeLAL3WZMFFtB5PuRYeuzF4Pta4HfCxa

**Signature:** 3DYQya24H5Ds4TVfKDe4pNSFS4Y3XZFF6GJdDbjMNK5vsvLYAyPwWwYNrYNKuXN36FrZC9AFBWugXTyk7ABq9PUM

Explore on Solana explorer - [Transaction History](#) |

CARa5swSHzqgxeLAL3WZMFFtB5PuRYeuzF4Pta4HfCxa | Solana

The screenshot shows the Solana Explorer (Beta) interface. At the top, there is a navigation bar with 'Cluster Stats', 'Supply', 'Inspector', and a 'Devnet' button. Below the navigation bar is a search bar with the placeholder 'Search for blocks, accounts, transactions, programs, and tokens'. The main content area has two sections: 'Token Account' and 'Transaction History'. The 'Token Account' section displays the following details for the newly created account:

Field	Value
Address	CARa5swSHzqgxeLAL3WZMFFtB5PuRYeuzF4Pta4HfCxa
Mint	HDLXuxjbXsrE7Dqecjb3EkGh13u4X2J9Fp3ouo9x6nBs
Owner	AvRTn81pWcYMPmYRt7LFNKcMmArpynayq8iPfk1qJ97s
Token balance	0

Below the 'Token Account' section is a navigation bar with tabs: 'History' (which is active), 'Tokens', and 'Domains'. The 'History' tab shows a table for 'Transaction History' with columns: TRANSACTION SIGNATURE, BLOCK, AGE, TIMESTAMP, and RESULT. There are no transactions listed.

## ▼ Mint some tokens to yourself

```
spl-token mint HDLXuxjbXsrE7Dqecjb3EkGh13u4X2J9Fp3ouo9x6nBs 100
```

By default, the minted token goes to associated token account -

**CARa5swSHzqgxeLAL3WZMFFtB5PuRYeuzF4Pta4HfCxa**

**Minting 100 tokens**

Token: HDLXuxjbXsrE7Dqecjb3EkGh13u4X2J9Fp3ouo9x6nBs

Recipient: CARa5swSHzqgxeLAL3WZMFFtB5PuRYeuzF4Pta4HfCxa

Signature: usMZZDdojQfcGYY78LTTxWiQ3kGFD1GMbgcRW8byqhzPtWdVBL1JAijnFBdAnX2UE4GRvoy2qDQa51Y3VcahLAh

Explore on - [Transaction History | CARa5swSHzqgxeLAL3WZMFFtB5PuRYeuzF4Pta4HfCxa | Solana](#)

The screenshot shows the Solana Explorer (Beta) interface. At the top, there is a navigation bar with tabs for Cluster Stats, Supply, Inspector, and a highlighted Devnet tab. Below the navigation bar is a search bar with placeholder text "Search for blocks, accounts, transactions, programs, and tokens".

**Token Account**

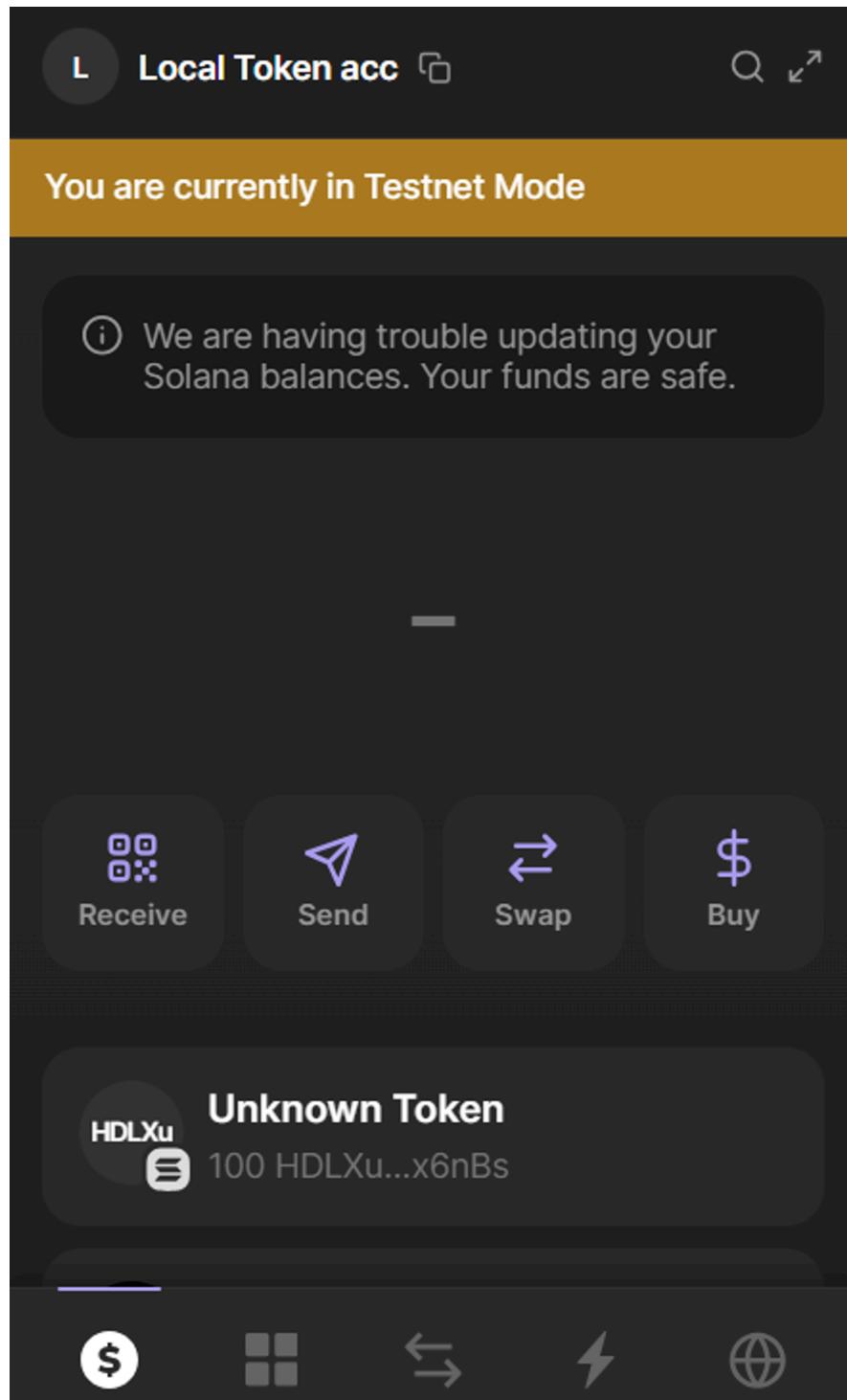
Address	CARa5swSHzqgxeLAL3WZMFFtB5PuRYeuzF4Pta4HfCxa
Mint	HDLXuxjbXsrE7Dqecjb3EkGh13u4X2J9Fp3ouo9x6nBs
Owner	AvRTn81pWcYNPmYRt7LFNKcMmArpynayq8iPfK1qJ97s
Token balance	100

Below the token account details, there are tabs for History, Tokens, and Domains. The History tab is selected.

**Transaction History**

TRANSACTION SIGNATURE	BLOCK	AOE	TIMESTAMP	RESULT
<a href="https://explorer.solana.com/?cluster=devnet">https://explorer.solana.com/?cluster=devnet</a>				

▼ Import the token in Phantom and see the balances



## Creating a Token using JavaScript

- ▼ Create a new CLI wallet, you can continue with your wallet as well

```
solana-keygen new
```

- ▼ Set the RPC URL

```
# Set RPC URL for Devnet
solana config set --url https://api.devnet.solana.com
```

- ▼ Create an empty JS file

```
npm init -y  
touch index.js
```

## ▼ Install dependencies

```
npm install @solana/web3.js @solana/spl-token
```

▼ Write a function to airdrop yourself some Solana

```
const {Connection, LAMPORTS_PER_SOL, clusterApiUrl, PublicKey} = require('@solana/web3.js');

const connection = new Connection(clusterApiUrl('devnet'));

async function airdrop(publicKey, amount) {
    const airdropSignature = await connection.requestAirdrop(new PublicKey(publicKey), amount);
    await connection.confirmTransaction({signature: airdropSignature})
}

airdrop("GokppTzVZi2LT1MSTWoEprM4YLDPy7wQ478Rm3r77yEw", LAMPORTS_PER_SOL).then(signature =>
    console.log('Airdrop signature:', signature);
);
```

▼ Check your balance

## solana balance

#### ▼ Create token mint

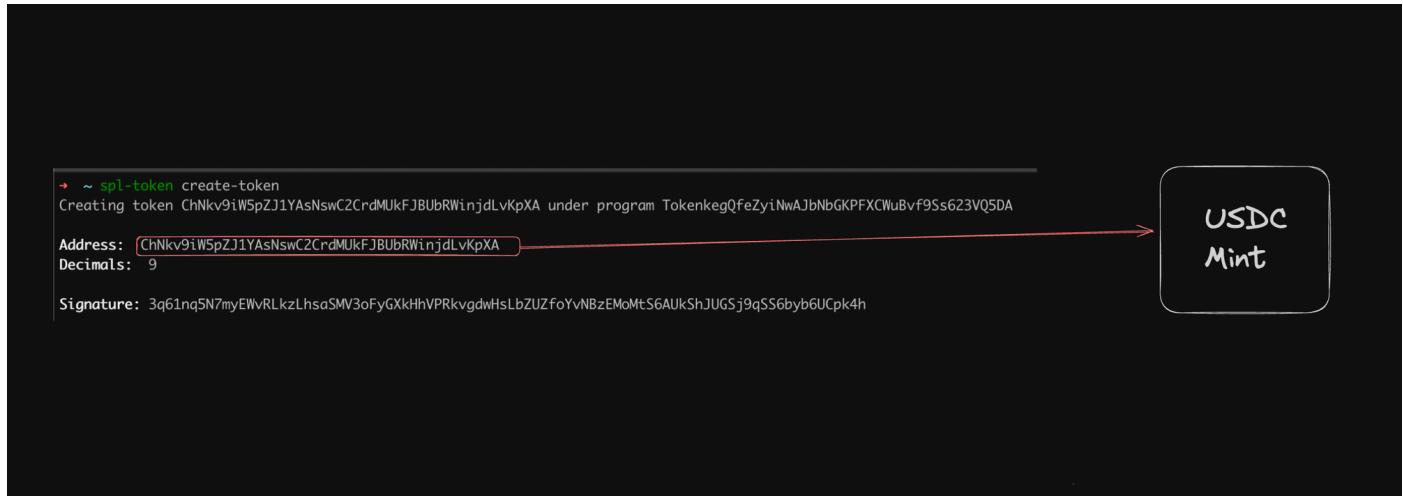
```

return mint;
}

async function main() {
    const mint = await createMintForToken(payer, mintAuthority.publicKey);
}

main();

```



## ▼ Verify token mint on chain

[Transaction History | ChNkv9iW5pZJ1YAsNswC2CrdMUKFJBUBRWinjdLvKpXA | Solana](#)

**Token Mint**

Address	ChNkv9iW5pZJ1YAsNswC2CrdMUKFJBUBRWinjdLvKpXA
Current Supply	100
Mint Authority	GokppTzVziLT1MSTwoErM4YLDPy7uQ478Rm3r77yEw
Decimals	9

**History**   **Transfers**   **Instructions**

**Transaction History**

TRANSACTION SIGNATURE	BLOCK	AGE	TIMESTAMP	RESULT
5THqzeKd6W7S1sAt6WkCeduJFufndqXhBBZz5EXS12kwZr7ERTYoYLTkdwh9...	321,945,884	2 days ago	Aug 24, 2024 at 05:22:36 UTC	Success
4nxH3SH9HCuxSuPvdd1uuNirk2M0dFrUqjn7HbM2v8mc6AEnztkAgtHjsTM...	321,043,420	6 days ago	Aug 24, 2024 at 08:13:08 UTC	Success
3akCxzfPmpS8mfqas0c1stmeBn4nzkRITx9998K5kDf7gBws7doalIs72Pwfj...	321,032,696	6 days ago	Aug 24, 2024 at 07:06:40 UTC	Success

## Postman Request

```
{
  "jsonrpc": "2.0",
  "id": 1,
  "method": "getAccountInfo",
  "params": [
    "ChNkv9iW5pZJ1YAsNswC2CrdMUKFJBUBRWinjdLvKpXA"
  ]
}
```

"params": [ "ChNkv9iW5pZJ1YAsNswC2CrdMUKFJBUbRWinjdLvKpXA" ]

## Postman Response

```
{
  "jsonrpc": "2.0",
  "result": {
    "context": {
      "apiVersion": "2.0.5",
      "slot": 322396843
    },
    "value": {
      "data": "DK9N6t1cUKGeS149Gnfgg5MqwXAQHFZk6MbQeL2bQjqqtKBrGBoNyZxCE8UxmREvxd4GQKQc",
      "executable": false,
      "lamports": 1461600,
      "owner": "TokenkegQfeZyiNwAJbNbGKPFXCWuBvf9Ss623VQ5DA",
      "rentEpoch": 18446744073709551615,
      "space": 82
    }
  },
  "id": 1
}
```

The screenshot shows the Postman interface with a successful API call to `https://api.devnet.solana.com`. The request method is `POST`, and the URL is `https://api.devnet.solana.com`. The `Body` tab is selected, containing raw JSON data:

```
1 {  
2     "jsonrpc": "2.0",  
3     "id": 1,  
4     "method": "getAccountInfo",  
5     "params": ["ChNkv9iw5pZJ1YAsNswC2CidMUKFJBUbRWinjdLvKpXA"]  
6 }
```

The response status is `200 OK`, time `269 ms`, size `1.2 KB`. The response body is displayed in pretty JSON format:

```
1 {  
2     "jsonrpc": "2.0",  
3     "result": {  
4         "context": {  
5             "apiVersion": "2.0.5",  
6             "slot": 322396843  
7         },  
8         "value": {  
9             "data": "DK9N6ticUKGeS149Gnfgg5MqwXAQHFZk6MbQeL2bQjqtKBzGBoNyZcE8UxmREvxd46QKQovaKoeYraUHRdokBx25c7bCwLdbPTqxkteJa4NPzI",  
10            "executable": false,  
11            "lamports": 1461600,  
12            "owner": "TokenkegQfeZyINwAJbNbGKPFXCuBvF9Ss623VQ5DA",  
13            "rentEpoch": 1846744673769551615,  
14            "space": 82  
15        },  
16    },  
17    "id": 1  
18 }
```

▼ Create an [associated token account](#), mint some tokens

```
const { createMint, getOrCreateAssociatedTokenAccount, mintTo } = require('@solana/spl-token')
const { Keypair, Connection, clusterApiUrl, TOKEN_PROGRAM_ID, PublicKey } = require('@solana/web3.js')

const payer = Keypair.fromSecretKey(Uint8Array.from([102, 144, 169, 42, 220, 87, 99, 85, 100, 128, 197]))
```

```

const mintAuthority = payer;

const connection = new Connection(clusterApiUrl('devnet'));

async function createMintForToken(payer, mintAuthority) {
    const mint = await createMint(
        connection,
        payer,
        mintAuthority,
        null,
        6,
        TOKEN_PROGRAM_ID
    );
    console.log('Mint created at', mint.toBase58());
    return mint;
}

async function mintNewTokens(mint, to, amount) {
    const tokenAccount = await getOrCreateAssociatedTokenAccount(
        connection,
        payer,
        mint,
        new PublicKey(to)
    );

    console.log('Token account created at', tokenAccount.address.toBase58());
    await mintTo(
        connection,
        payer,
        mint,
        tokenAccount.address,
        payer,
        amount
    )
    console.log('Minted', amount, 'tokens to', tokenAccount.address.toBase58());
}

async function main() {
    const mint = await createMintForToken(payer, mintAuthority.publicKey);
    await mintNewTokens(mint, mintAuthority.publicKey, 100);
}

main();

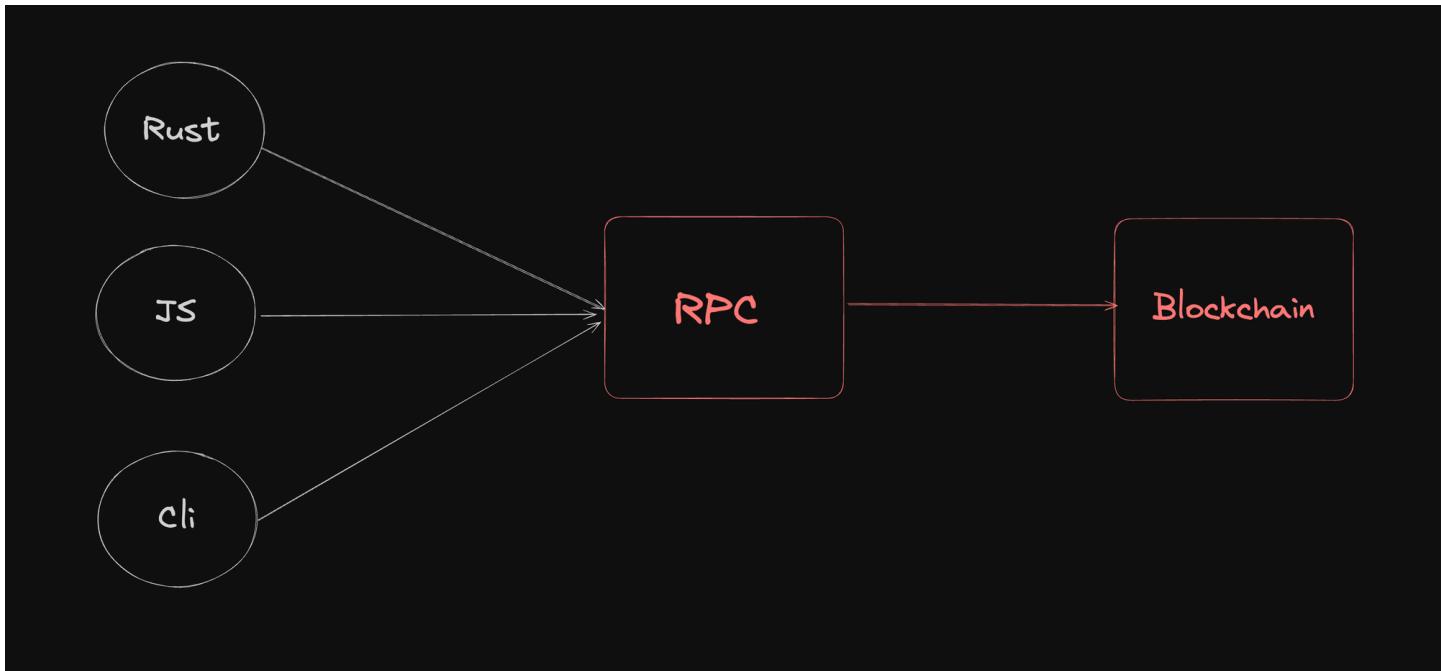
```

- Check your balance in the explorer
- Import the token in Phantom and see the balances

## Equivalent code in rust/python/go

Solana has libraries similar to [@solana/web3.js](#) in Rust, Python that would let you do the same thing.

In the end, they all are sending requests to an RPC server.



## Token 22 program

- A token program on the Solana blockchain, defining a common implementation for fungible and non-fungible tokens.
- The Token-2022 Program, also known as Token Extensions, is a superset of the functionality provided by the [Token Program](#).

<https://spl.solana.com/token-2022>

### ▼ Create token mint

```
spl-token create-token --program-id TokenzQdBNbLqP5VEhdkAS6EPFLC1PHnBqCXEpPxuEb
```

### ▼ Create an [associated token account](#)

```
spl-token create-account 8fTM5XYRaoTJU9PLUuyakF3EypQ4RXL5HxKtiw2z9pQQ
```

### ▼ Mint the tokens

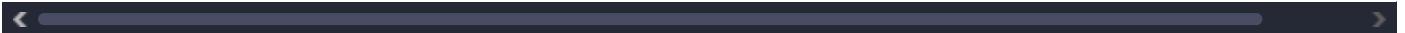
```
spl-token mint 8fTM5XYRaoTJU9PLUuyakF3EypQ4RXL5HxKtiw2z9pQQ 100
```

## Token 22 with metadata

## Metadata - <https://cdn.100xdevs.com/metadata.json>

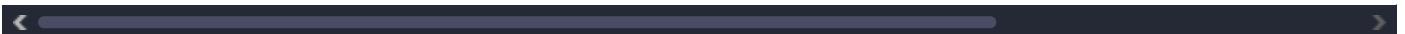
### ▼ Create a token with metadata enabled

```
spl-token --program-id TokenzQdBNbLqP5VEhdkAS6EPFLC1PHnBqCXEpPxuEb create-token --enable-met
```



### ▼ Create metadata

```
spl-token initialize-metadata pXFZ6Hg2s78m1iSRVsdzos9TmfkqkQdv5MmQrr77ZQK 100xx 100xxx https:
```



### ▼ Create associated token account

```
spl-token create-account pXFZ6Hg2s78m1iSRVsdzos9TmfkqkQdv5MmQrr77ZQK
```

### ▼ Mint

```
spl-token mint 1000
```

### ▼ Check out the token in your wallet

