Installation

This section covers the steps to set up your local environment for Solana development.

Install Dependencies#

- Windows users must first install WSL (Windows subsystem for Linux) and then
- install the dependencies specified in the Linux section below.
- Linux users should first install the dependencies specified in the Linux
- section below.
- Mac users should start with the Rust installation instructions below.

Windows Subsystem for Linux (WSL)

Linux

Install Rust#

Solana programs are written in the Rust programming language .

The recommended installation method for Rust is rustup .

Run the following command to install Rust:

curl --proto '=https' --tlsv1.2 -sSf https://sh.rustup.rs | sh -s -- -y You should see the following message after the installation completes:

Successful Rust Install Message

Run the following command to reload your PATH environment variable to include Cargo's bin directory:

. " HOME /.cargo/env" To verify that the installation was successful, check the Rust version:

rustc --version You should see output similar to the following:

rustc 1.80.1 (3f5fd8dd4 2024-08-06)

Install the Solana CLI#

The Solana CLI provides all the tools required to build and deploy Solana programs.

Install the Solana CLI tool suite using the official install command:

sh -c "(curl -sSfL https://release.anza.xyz/stable/install)" You can replacestable with the release tag matching the software version of your desired release (i.e.v2.0.3), or use one of the three symbolic channel names:stable ,beta , oredge .

If it is your first time installing the Solana CLI, you may see the following message prompting you to add a PATH environment variable:

Close and reopen your terminal to apply the PATH changes or run the following in your existing shell:

export PATH="/Users/test/.local/share/solana/install/active_release/bin:PATH" Linux Mac If you are using a Linux or WSL terminal, you can add the PATH environment variable to your shell configuration file by running the command logged from the installation or by restarting your terminal.

export PATH = " HOME /.local/share/solana/install/active_release/bin: PATH " To verify that the installation was successful, check the Solana CLI version:

solana --version You should see output similar to the following:

solana-cli 1.18.22 (src:9efdd74b; feat:4215500110, client:Agave) You can view all available versions on the Agave Github repo .

Info Agave is the validator client from Anza , formerly known as Solana Labs validator client. To later update the Solana CLI to the latest version, you can use the following command:

agave-install update

Install Anchor CLI#

Anchor is a framework for developing Solana programs. The Anchor framework leverages Rust macros to simplify the process of writing Solana programs.

There are two ways to install the Anchor CLI and tooling:

1. Using Anchor Version Manager (AVM) - is therecommended installation
2. method since it simplifies updating Anchor versions in the future
3. Without AVM - this requires more a manual process to update Anchor versions
4. later

AVM Without AVM The Anchor version manager (AVM) allows you to install and manage different Anchor versions on your system, including more easily updating Anchor versions in the future.

Install AVM with the following command:

cargo install --git https://github.com/coral-xyz/anchor avm --force Test to ensure AVM was installed and is accessible:

avm --version Install the latest version of Anchor CLI using AVM:

avm install latest avm use latest Or install a specific version of the Anchor CLI by declaring which version you want to install:

avm install 0.30.1 avm use 0.30.1 Info Don't forget to run theavm use command to declare which Anchor CLI version should be used on your system.

- If you installed thelatest
- version, runavm use latest
- .
- If you installed the version0.30.1
- , runavm use 0.30.1
- . You may see the following warning during installation. However, it does not affect the installation process.

warning: unexpected cfg condition name: nightly

To verify that the installation was successful, check the Anchor CLI version:

anchor --version You should see output similar to the following:

anchor-cli 0.30.1 When installing the Anchor CLI on Linux or WSL, you may encounter this error:

error: could not exec the linker cc = note: Permission denied (os error 13) If you see this error message, follow these steps:

1. Install the dependencies listed in the Linux section at the top of this page.
2. Retry installing the Anchor CLI.

Node.js and Yarn#

Node.js and Yarn are required to run the default Anchor project test file (TypeScript) created with theanchor init command. (Rust test template is also available usinganchor init --test-template rust)

Node Installation

Yarn Installation

When runninganchor build , if you encountererror: not a directory similar following:

error: not a directory: '.../solana-release/bin/sdk/sbf/dependencies/platform-tools/rust/lib' Try these solutions:

1. Force install using the following command:

cargo build-sbf --force-tools-install 1. If the above doesn't work, clear the Solana cache:

rm -rf ~/.cache/solana/ * After applying either solution, attempt to runanchor build again.

If you are on Linux or WSL and encounter the following errors when runninganchor test after creating a new Anchor project, it's may be due to missing Node.js or Yarn:

Permission denied (os error 13) No such file or directory (os error 2)

Solana CLI Basics#

This section will walk through some common Solana CLI commands to get you started.

Solana Config#

To see your current config:

solana config get You should see output similar to the following:

Config File: /Users/test/.config/solana/cli/config.yml RPC URL: https://api.mainnet-beta.solana.com WebSocket URL: wss://api.mainnet-beta.solana.com/ (computed) Keypair Path: /Users/test/.config/solana/id.json Commitment: confirmed The RPC URL and Websocket URL specific the Solana cluster the CLI will make requests to. By default this will be mainnet-beta.

You can update the Solana CLI cluster using the following commands:

solana config set --url mainnet-beta solana config set --url devnet solana config set --url localhost solana config set --url testnet You can also use the following short options:

solana config set -um # For mainnet-beta solana config set -ud # For devnet solana config set -ul # For localhost solana config set -ut # For testnet The Keypair Path specifies the location of the default wallet used by the Solana CLI (to pay transaction fees and deploy programs). The default path is~/.config/solana/id.json . The next step walks through how to generate a keypair at the default location.

Create Wallet#

To interact with the Solana network using the Solana CLI, you need a Solana wallet funded with SOL.

To generate a keypair at the default Keypair Path, run the following command:

solana-keygen new You should see output similar to the following:

Generating a new keypair

For added security, enter a BIP39 passphrase

NOTE! This passphrase improves security of the recovery seed phrae NOT the keypair file itself, which is stored as insecure plain text

BIP39 Passphrase (empty for none):

Wrote new keypair to /Users/test/.config/solana/id.json

pubkey: 8dBTPrjnkXyuQK3KDt9wrZBfizEZijmmUQXVHpFbVwGT

Save this seed phrase and your BIP39 passphrase to recover your new keypair: cream bleak tortoise ocean nasty game gift forget fancy salon mimic amazing === Info If you already have a file system wallet saved at the default location, this command willNOT override it unless you explicitly force override using the--force flag. Once a keypair is generated, you can get the address (public key) of the keypair with the following command:

solana address

Airdrop SOL#

Once you've set up your local wallet, request an airdrop of SOL to fund your wallet. You need SOL to pay for transaction fees and to deploy programs.

Set your cluster to the devnet:

solana config set -ud Then request an airdrop of devnet SOL:

solana airdrop 2 To check your wallet's SOL balance, run the following command:

solana balance Info Thesolana airdrop command is currently limited to 5 SOL per request on devnet. Errors are likely due to rate limits.

Alternatively, you can get devnet SOL using the Solana Web Faucet .

Run Local Validator#

The Solana CLI comes with the test validator built-in. Running a local validator will allow you to deploy and test your programs locally.

In a separate terminal, run the following command to start a local validator:

solana-test-validator Info In WSL you may need to first navigate to a folder where you have default write access:

cd ~ mkdir validator cd validator solana-test-validator Make sure to update the Solana CLI config to localhost before commands.

solana config set -ul

Previous «Cross Program Invocation Next Intro to Development»