Using Data Feeds Offchain (Solana)

Chainlink Data Feeds are the quickest way to access market prices for real-world assets. This guide demonstrates how to read Chainlink Data Feeds on the Solana Devnet using offchain examples in theChainlink Solana Starter Kit . To learn how to use Data Feeds in your onchain Solana programs, see theUsing Data Feeds onchain guide.

To get the full list of Chainlink Data Feeds on Solana, see theSolana Feeds page.

Select quality data feeds

Be aware of the quality of the data that you useLearn more about making responsible data quality decisions.

The Chainlink Data Feeds Store Program

The program that contains the logic required for the storing and retrieval of Chainlink Data Feeds data on both Devnet and Mainnet iscjg3oHmg9uuPsP8D6g29NWvhySJkdYdAo9D25PRbKXJ . This is the program ID that you use to read price data from offchain. You can find the source code for this program in thesmartcontractkit/chainlink-solana on GitHub.

You canadd data feeds to an existing offchain projectoruse the Solana Starter Kit .

Adding Data Feeds to an existing offchain project

You can read Chainlink Data Feeds offchain in your existing project by using theChainlink Solana NPM library .

Reading feed data

Although you can directly query the data feed accounts, you should not rely on the memory layout always being the same as it currently is. Based on this, the recommendation is to always use the consumer library.

Install the necessary components and include the example code in your project. Optionally, you can run the example code by itself to learn how it works before you integrate it with your project.

1. Install the latest Mainnet version ofthe Solana CLI and export the path to the CLI:

sh-c"$(curl-sSfLhttps://release.solana.com/v1.13.6/install)"&&exportPATH="~/.local/share/solana/install/active_release/bin:$PATH"Runsolana--versionto make sure the Solana CLI is installed correctly.

solana--version 2. InstallNode.js 14 or higher . Runnode --versionto verify which version you have installed:

node--version 3. Change to your project directory or create a new directory.

mkdiroff-chain-project&&cdoff-chain-project 4. Optionallyinstall Yarn to use as a package manager and initialize yarn if your project does not already have apackage.jsonfile:

npminstall-gyarn&&yarninit 5. Add theAnchor library to your project:

npmyarnnpmi @project-serum/anchoryarnadd@project-serum/anchor 6. Add theChainlink Solana NPM library to your project:

npmyarnnpmi-g@chainlink/solana-sdkyarnadd@chainlink/solana-sdk 7. Create a temporary Solana wallet to use for this example. Alternatively, if you have an existing wallet that you want to use, locate the path to yourkeypair file and use it as the keypair for the rest of this guide.

solana-keygen new--outfile./id.json 8. Set theAnchor environment variables . Anchor uses these to determine which wallet to use and how to get a connection to a Solana cluster. Because this example does not generate or sign any transactions, no lamports are required. The wallet is required only by the Anchor library. For a list of available networks and endpoints, see theSolana Cluster RPC Endpoints documentation.

exportANCHOR_PROVIDER_URL=https://api.devnet.solana.com&&exportANCHOR_WALLET=./id.json 9. Copy the sample code into your project. This example queries price data offchain. By default, the script reads the SOL/USD feed, but you can change theCHAINLINK_FEED_ADDRESSvariable to point to thefeed account addresses that you want to query. You can take the components of these code samples and integrate them with your existing project. Because these examples read data feeds without making any onchain changes, no lamports are required to run them.

/ * THIS IS EXAMPLE CODE THAT USES HARDCODED VALUES FOR CLARITY. * THIS IS EXAMPLE CODE THAT USES UN-AUDITED CODE. * DO NOT USE THIS CODE IN PRODUCTION. */const anchor=require("@project-serum/anchor")const chainlink=require("@chainlink/solana-sdk")const provider=anchor.AnchorProvider.env()asyncfunctionmain(){anchor.setProvider(provider)const CHAINLINK_FEED_ADDRESS="99B2bTijsU6f1GCT73HmdR7HCFFjGMBcPZY6jZ96ynrR"const CHAINLINK_PROGRAM_ID=newanchor.web3.PublicKey("cjg3oHmg9uuPsP8D6g29NWvhySJkdYdAo9D25PRbKXJ")const feedAddress=newanchor.web3.PublicKey(CHAINLINK_FEED_ADDRESS)//SOL-USD Devnet Feed//load the data feed accountletdataFeed=await chainlink.OCR2Feed.load(CHAINLINK_PROGRAM_ID,provider)letlistener=null//listen for

events agains the price feed, and grab the latest rounds price datalistener=dataFeed.onRound(feedAddress,(event)=> {console.log(event.answer.toNumber())})//block execution and keep waiting for events to be emitted with price dataawaitnewPromise(function(){})}main().then(()=>process.exit(),(err)=>{console.error(err)process.exit(-1)})/ * THIS IS EXAMPLE CODE THAT USES HARDCODED VALUES FOR CLARITY. * THIS IS EXAMPLE CODE THAT USES UN-AUDITED CODE. * DO NOT USE THIS CODE IN PRODUCTION. */importasanchorfrom"@project-serum/anchor"import{OCR2Feed}from"@chainlink/solana-sdk"asyncfunctionmain(){const provider=anchor.AnchorProvider.env()anchor.setProvider(provider)const CHAINLINK_FEED_ADDRESS="99B2bTijsU6f1GCT73HmdR7HCFFjGMBcPZY6jZ96ynrR"const CHAINLINK_PROGRAM_ID=newanchor.web3.PublicKey("cjg3oHmg9uuPsP8D6g29NWvhySJkdYdAo9D25PRbKXJ")const feedAddress=newanchor.web3.PublicKey(CHAINLINK_FEED_ADDRESS)//SOL-USD Devnet Feed//load the data feed accountletdataFeed=await OCR2Feed.load(CHAINLINK_PROGRAM_ID,provider)letlistener:null|number=null//listen for events agains the price feed, and grab the latest rounds price datalistener=dataFeed.onRound(feedAddress,(event)=> {console.log(event.answer.toNumber())})//block execution and keep waiting for events to be emitted with price dataawaitnewPromise(function(){})}main().then(()=>process.exit(),(err)=>{console.error(err)process.exit(-1)})You can run these examples using the following commands:

JavascriptTypescriptnodejavascript-example.jsyarnaddts-node typescript&&yarnts-node typescript-example.ts

To learn more about Solana and Anchor, see theSolana Documentation and theAnchor Documentation .

Using the Solana Starter Kit

This example reads price data from an offchain client using theSolana Starter Kit .

Install the required tools

Before you begin, set up your environment for development on Solana:

1. InstallGit if it is not already configured on your system.
2. Install the latest Mainnet version ofthe Solana CLI and export the path to the CLI:

sh-c"$(curl-sSfLhttps://release.solana.com/v1.13.6/install)"&&exportPATH="~/.local/share/solana/install/active_release/bin:$PATH"Runsolana --versionto make sure the Solana CLI is installed correctly.

solana--version 3. InstallNode.js 14 or higher . Runnode --versionto verify which version you have installed:

node--version 4. Install Anchor . On some operating systems, you might need to build and install Anchor locally. See theAnchor documentation for instructions. 5. InstallYarn to simplify package management and run code samples in the Starter Kit.

npminstall-gyarn

Run the example program

After you install the required tools, clone the example code from thesolana-starter-kit repository.

1. In a terminal, clone thesolana-starter-kit repository and change to thesolana-starter-kitdirectory:

gitclone https://github.com/smartcontractkit/solana-starter-kit&&cd./solana-starter-kitYou can see the complete code for the example onGitHub . 2. In the./solana-starter-kitdirectory, install Node.js dependencies defined in thepackage.jsonfile:

yarninstall 3. Create a temporary Solana wallet file to use for this example. Because your application runs offchain and does not run any functions or alter data onchain, the wallet does not require any SOL tokens to function.

solana-keygen new--outfile./id.json 4. Set theAnchor environment variables . Anchor uses these to determine which wallet to use and Solana cluster to use. Take note that because we are not generating or signing any transactions, the wallet isn't used, it's just required by the Anchor library. For a list of available networks and endpoints, see theSolana Cluster RPC Endpoints documentation.

exportANCHOR_PROVIDER_URL=https://api.devnet.solana.com&&exportANCHOR_WALLET=./id.json 5. Run the example:

JavascriptTypescriptnoderead-data.jsyarnrun read-dataThe example code retrieves and prints the current price feed data until you close the application:

4027000000 4026439929 4026476542 4023000000

To learn more about Solana and Anchor, see theSolana Documentation and theAnchor Documentation .