

NEW Chainlink Data Streams have officially launched on mainnet. Sign up for early access.

 [Back](#)

## Using Data Feeds on EVM Chains

The code for reading Data Feeds is the same across all EVM-compatible blockchains and Data Feed types. You choose different types of feeds for different uses, but the request and response format are the same. To read a feed, specify the following variables:

- **RPC endpoint URL:** This determines which network that your smart contracts will run on. You can use a [node provider service](#) or point to your own [client](#). If you are using a Web3 wallet, it is already configured with the RPC endpoints for several networks and the [Remix IDE](#) will automatically detect them for you.
- **LINK token contract address:** The address for the LINK token contract is different for each network. You can find the full list of addresses for all supported networks on the [LINK Token Contracts](#) page.
- **Feed contract address:** This determines which data feed your smart contract will read. Contract addresses are different for each network. You can find the available contract addresses on the following pages:
  - [Price Feed Addresses](#)
  - [Proof of Reserve Feed Addresses](#)
  - [NFT Floor Price Feed Addresses](#)

The examples in this document indicate these variables, but you can modify the examples to work on different networks and read different feeds.

This guide shows example code that reads data feeds using the following languages:

- Onchain consumer contracts:
  - [Solidity](#)
  - [Vyper](#)
- Offchain reads using Web3 packages:



- [Golang](#) with [go-ethereum](#)

## Reading data feeds onchain

These code examples demonstrate how to deploy a consumer contract onchain that reads a data feed and stores the value.

### Solidity

To consume price data, your smart contract should reference [AggregatorV3Interface](#), which defines the external functions implemented by Data Feeds.

```
// SPDX-License-Identifier: MIT
pragma solidity ^0.8.7;

import {AggregatorV3Interface} from "@chainlink/contracts/src/v0.8/shared/interfaces/AggregatorV

/**
 * THIS IS AN EXAMPLE CONTRACT THAT USES HARDCODED
 * VALUES FOR CLARITY.
 * THIS IS AN EXAMPLE CONTRACT THAT USES UN-AUDITED CODE.
 * DO NOT USE THIS CODE IN PRODUCTION.
 */

/**
 * If you are reading data feeds on L2 networks, you must
 * check the latest answer from the L2 Sequencer Uptime
 * Feed to ensure that the data is accurate in the event
 * of an L2 sequencer outage. See the
 * https://docs.chain.link/data-feeds/l2-sequencer-feeds
 * page for details.
 */

contract DataConsumerV3 {
    AggregatorV3Interface internal dataFeed;

    /**
     * Network: Sepolia
     * Aggregator: BTC/USD
     * Address: 0x1b44F3514812d835EB1BDB0acB33d3fA3351Ee43
     */
    constructor() {
```





```
}

/**
 * Returns the latest answer.
 */
function getChainlinkDataFeedLatestAnswer() public view returns (int) {
    // prettier-ignore
    (
        /* uint80 roundID */,
        int answer,
        /*uint startedAt*/,
        /*uint timeStamp*/,
        /*uint80 answeredInRound*/
    ) = dataFeed.latestRoundData();
    return answer;
}
```

[Open in Remix](#)[What is Remix?](#)

The `latestRoundData` function returns five values representing information about the latest price data. See the [Data Feeds API Reference](#) for more details.

## Vyper

To consume price data, your smart contract should import `AggregatorV3Interface` which defines the external functions implemented by Data Feeds. You can find it [here](#). You can find a `PriceConsumer` example [here](#). Read the ***apeworx-starter-kit*** [README](#) to learn how to run the example.

## Reading data feeds offchain

These code examples demonstrate how to read data feeds directly off chain using Web3 packages for each language.

## Javascript



web3.js

ethers.js



```
/**
 * 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 Web3 = require("web3") // for nodejs only
const web3 = new Web3("https://rpc.ankr.com/eth_sepolia")
const aggregatorV3InterfaceABI = [
  {
    inputs: [],
    name: "decimals",
    outputs: [{ internalType: "uint8", name: "", type: "uint8" }],
    stateMutability: "view",
    type: "function",
  },
  {
    inputs: [],
    name: "description",
    outputs: [{ internalType: "string", name: "", type: "string" }],
    stateMutability: "view",
    type: "function",
  },
  {
    inputs: [{ internalType: "uint80", name: "_roundId", type: "uint80" }],
    name: "getRoundData",
    outputs: [
      { internalType: "uint80", name: "roundId", type: "uint80" },
      { internalType: "int256", name: "answer", type: "int256" },
      { internalType: "uint256", name: "startedAt", type: "uint256" },
      { internalType: "uint256", name: "updatedAt", type: "uint256" },
      { internalType: "uint80", name: "answeredInRound", type: "uint80" },
    ],
    stateMutability: "view",
    type: "function",
  },
  {
    inputs: [],
    name: "latestRoundData",
    outputs: [
      { internalType: "uint80", name: "roundId", type: "uint80" },
      { internalType: "int256", name: "answer", type: "int256" },
      { internalType: "uint256", name: "startedAt", type: "uint256" },
      { internalType: "uint256", name: "updatedAt", type: "uint256" },
      { internalType: "uint80", name: "answeredInRound", type: "uint80" },
    ],
  },
]
```



```

    },
    {
      inputs: [],
      name: "version",
      outputs: [{ internalType: "uint256", name: "", type: "uint256" }],
      stateMutability: "view",
      type: "function",
    },
  ],
  const addr = "0x1b44F3514812d835EB1BDB0acB33d3fA3351Ee43"
  const priceFeed = new web3.eth.Contract(aggregatorV3InterfaceABI, addr)
  priceFeed.methods
    .latestRoundData()
    .call()
    .then((roundData) => {
      // Do something with roundData
      console.log("Latest Round Data", roundData)
    })
  })

```

Latest Price:

Latest Price

## Python

This example uses [Web3.py](#) to retrieve feed data from the [BTC / USD feed](#) on the Sepolia testnet.

```

# 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.

```



```

from web3 import Web3

# Change this to use your own RPC URL
web3 = Web3(Web3.HTTPProvider('https://rpc.ankr.com/eth_sepolia'))
# AggregatorV3Interface ABI
abi = '[{"inputs":[],"name":"decimals","outputs":[{"internalType":"uint8","name":"","type":"uint8"}],"stateMutability":"view","type":"function"}]'
# Price Feed address
addr = '0x1b44F3514812d835EB1BDB0acB33d3fA3351Ee43'

# Set up contract instance
contract = web3.eth.contract(address=addr, abi=abi)
# Make call to latestRoundData()

```

[Run this Python example](#)

## Golang

You can find an example with all the source files [here](#). This example uses [go-ethereum](#) to retrieve feed data from the [BTC / USD feed](#) on the Sepolia testnet. To learn how to run the example, see the [README](#).



### NEW FEED REGISTRY

You can use the [Feed Registry](#) to reference data feed assets by name or currency identifier instead of by pair/proxy address.

## Getting a different price denomination

Chainlink Data Feeds can be used in combination to derive denominated price pairs in other currencies.

If you require a denomination other than what is provided, you can use two data feeds to derive the pair that you need. For example, if you needed a BTC / EUR price, you could take the BTC / USD feed and the EUR / USD feed and derive BTC / EUR using division.

$$\frac{BTC/USD}{EUR/USD} = BTC/EUR$$



### IMPORTANT

If your contracts require Solidity versions that are  $\geq 0.6.0$   $< 0.8.0$ , use [OpenZeppelin's SafeMath version 3.4](#).

```
// SPDX-License-Identifier: MIT  
pragma solidity ^0.8.7;
```



```
import {AggregatorV3Interface} from "@chainlink/contracts/src/v0.8/shared/interfaces/AggregatorV  
  
/**  
 * Network: Sepolia
```



```

* Quote Address: 0x1a81afB8146aeFfCFc5E50e8479e826E7D55b910
* Decimals: 8
*/

/**
 * THIS IS AN EXAMPLE CONTRACT THAT USES HARDCODED VALUES FOR CLARITY.
 * THIS IS AN EXAMPLE CONTRACT THAT USES UN-AUDITED CODE.
 * DO NOT USE THIS CODE IN PRODUCTION.
 */

contract PriceConverter {
    function getDerivedPrice(
        address _base,
        address _quote,
        uint8 _decimals
    ) public view returns (int256) {
        require(
            _decimals > uint8(0) && _decimals <= uint8(18),
            "Invalid _decimals"
        );
        int256 decimals = int256(10 ** uint256(_decimals));
        (, int256 basePrice, , , ) = AggregatorV3Interface(_base)
            .latestRoundData();
        uint8 baseDecimals = AggregatorV3Interface(_base).decimals();
        basePrice = scalePrice(basePrice, baseDecimals, _decimals);

        (, int256 quotePrice, , , ) = AggregatorV3Interface(_quote)
            .latestRoundData();
        uint8 quoteDecimals = AggregatorV3Interface(_quote).decimals();
        quotePrice = scalePrice(quotePrice, quoteDecimals, _decimals);

        return (basePrice * decimals) / quotePrice;
    }

    function scalePrice(
        int256 _price,
        uint8 _priceDecimals,
        uint8 _decimals
    ) internal pure returns (int256) {
        if (_priceDecimals < _decimals) {
            return _price * int256(10 ** uint256(_decimals - _priceDecimals));
        } else if (_priceDecimals > _decimals) {
            return _price / int256(10 ** uint256(_priceDecimals - _decimals));
        }
        return _price;
    }
}

```

[Open in Remix](#)[What is Remix?](#)

## More aggregator functions

Getting the latest price is not the only data that aggregators can retrieve. You can also retrieve historical price data. To learn more, see the [Historical Price Data](#) page.

To understand different use cases for Chainlink Price Feeds, refer to [Other Tutorials](#).

## What's next

[> Historical Price Data](#)[> Data Feeds API Reference](#)

## Stay updated on the latest Chainlink news

[Subscribe now](#)

### Developers

Developer resources





VRF

Automation

Functions

CCIP

## Solutions

Overview

DeFi

Chainlink VRF

## Community

Chainlink Hackathon

Community overview

Grant program

Events

Become an advocate

Code of conduct

## Chainlink

Ecosystem

Press

Blog

Team

Brand assets

FAQs



Security

Support

Press inquiries

## Social

 Twitter

 YouTube

 Discord

 Telegram

 WeChat

 Reddit

[Privacy Policy](#)

[Terms of Use](#)