Jeffrey Scholz Jan 11 3 min read

How Chainlink Price Feeds Work

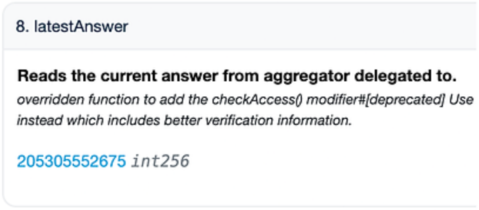
Updated: Feb 29

Chainlink price oracles are smart contracts with public view functions that return the price of a particular asset denominated in USD.

Off-chain nodes collect the prices from various sources like exchanges and write the price data to the smart contract.

Here is the smart contract for getting the price of ETH / USD:
https://etherscan.io/address/0x5f4eC3Df9cbd43714FE2740f5E3616155c5b8419/advanced#readContract

When we call the function latestAnswer() we get the price of Ether. When we query decimals() we get the number of decimals to interpret the answer with.

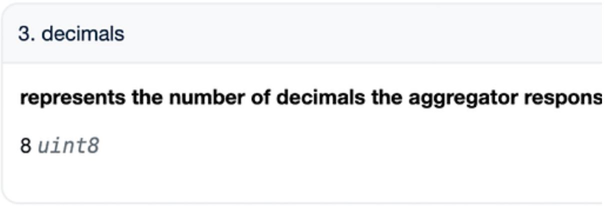


8. latestAnswer

Reads the current answer from aggregator delegated to.

overridden function to add the checkAccess() modifier#[deprecated] Use instead which includes better verification information.

205305552675 *int256*



3. decimals

represents the number of decimals the aggregator respons

8 *uint8*

Therefore, the current price of Ether is $2053.05552675 according to the oracle (true at the time of writing).

If you just want an idea of how Chainlink oracles work, you can stop here — that's all a price oracle is!

What follows is important implementation details if you plan on using them in a project.

We will use ETH / USD as a running example, but Chainlink supports many more asset prices.

Using latestAnswer() is not recommended — use latestRoundData() instead

This function latestAnswer() does not tell us the last time the price updated. If price updates are delayed, the smart contract might make decisions based on outdated prices.

In the green box below, we see the same price we got from latestAnswer() and in the blue box we see when it was last updated as a unix timestamp.

Smart contracts may want to set a threshold such that they use an alternative oracle or suspend critical decisions until the updatedAt field is sufficiently recent.

Price Aggregation

It would be unsafe to rely on a single node or data source to obtain prices, so Chainlink price feeds have several whitelisted nodes that supply prices.

The suppliers of the ETH / USD price feed are screenshotted below.

The range of prices they supply at a given time can be seen in the small variation in reported price. Note that the "cents" portion of the price varies from 26 cents (top row) to 73 cents (bottom row).

transmit()

The off-chain prices enter the smart contract ecosystem via the transmit function. The function takes a list of (sorted) prices and a list of signatures from the nodes. The price reported on the oracle is the median of the prices. Below we show the relevant line of code from Etherscan.

Smart contract Architecture

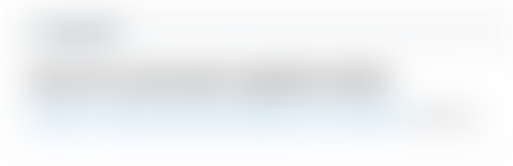
The reader may have noticed that the latestRoundData() function is not in the same contract as transmit(). There are three smart contracts at play:

1. The price feed contract
2. The aggregator contract

Price update transaction

During a price update, the signatures and prices of the nodes are batched together and sent to transmit() in the aggregator contract. The aggregator contract then calls the validate function in the validator contract. Subject to the rules there, the price update might be rejected. The tenderly trace of such a transaction is screenshotted below. The purple call codes show the cross contract calls.

Viewing the price



Improving the gas efficiency of reading price oracles

Because viewing the price involves a cross-contract call, it is recommended to save 200 gas by “pre-warming” the aggregator call using an accessing list transaction.

Recent Posts

See All

Price update frequency

It isn't practical to keep sending state-updating transactions to the blockchain every minute. Therefore, Chainlink updates the price

one h

nk ET

Understanding the Function Selector in Solidity

17 0 1 ♡

How ERC721 Enumerable Works

176 0 2 ♡

Subscribe to our newsletter Subscribe Now

We do not sell your information to anyone. Period.

Home About Curriculum Pricing Testimonials Tutorials Test Yours APPLY NOW

Web3 Blockchain Bootcamp

Tutorials

Instructor Bios

Testimonials

Learn Solidity

Pricing

About RareSkills.

Follow us on

Curriculum

Hire our Developers

Test Yourself

Admission Process and Policy

Contact Us

Privacy Policy