

★ Get unlimited access to the best of Medium for less than \$1/week. [Become a member](#) X

Smart Contract Triggering with Chainlink Automation



Joe · Follow

Published in Web3 Writers Guild

5 min read · Nov 14, 2023

Listen

Share

More



You might have properly written a smart contract before and you have been told that smart contracts functions must be triggered either by an EOA (Externally owned account) or another smart contract.

And then for every time, you want to make a change to your contract storage, you have to trigger a function using your EOA or writing another smart contract. If you are coming from the web2 space or you have a slight idea of how the backend system works in web2 space. You will know about cron jobs.

Cron jobs are a powerful tool for automating tasks in Node.js. They allow you to run code at specific intervals, such as every minute, hour, day, week, or month. This can be used to do things like send emails, run backups, or update data.

What if you can implement something like that in your smart contracts, and then you don't have to be the one doing the job of calling functions yourself.

Welcome to Chainlink Automation. Let me take you through a summary of how it works

Chainlink Automation are more like triggers that trigger functions executions in smart contracts and it is categorized into three triggers and links to Chainlink to read more about them, and there are as follows;

Time-based trigger: Use a time-based trigger to execute your function according to a time schedule. This feature is also called the Job Scheduler and it is a throwback to the Ethereum Alarm Clock. Time-based trigger contracts don't need to be compatible with the **AutomationCompatibleInterface** contract.

Custom logic trigger: Use a custom logic trigger to provide custom solidity logic that Automation Nodes evaluate (off-chain) to determine when to execute your function on-chain. Your contract must meet the requirements to be compatible with the **AutomationCompatibleInterface** contract. Custom logic examples include checking the balance on a contract, only executing limit orders when their levels are met, any one of our coded examples, and many more.

Log trigger: Use log data as both trigger and input. Your contract must meet the requirements to be compatible with the **AutomationCompatibleInterface** contract.

For this example, I will be showing you an implementation of chainlink Automation using Time-based trigger. I will take you through a simple contract written in solidity.

We will create a contract called automatedPay that allows

```
// SPDX-License-Identifier: UNLICENSED

pragma solidity 0.8.18;

contract AutomatedPay {
    address payee;
    uint unlocked;
    uint lockedTime;
    address owner;

    constructor(address _payee, uint _unlocked) {
        payee = _payee;
        unlocked = _unlocked;
        lockedTime = block.timestamp;
        owner = msg.sender;
    }

    error TimeNotReached();

    function deposit() external payable {
        require(msg.value > 0, "ZeroAmount");
    }

    function withdraw() external {
        require(msg.sender == owner, "Unauthorized");
        if (block.timestamp <= (lockedTime + unlocked)) {
            revert TimeNotReached();
        }
        (bool sent, ) = payee.call{value: address(this).balance}("");
        require(sent, "Failed to send Ether");
    }

    receive() external payable { }

}
```

The smart contract above allows withdrawal of funds after the locked time has been reached. The unlocked time is set at constructor level, the account to pay is also set at constructor level.

Deploy the contract above and save the address, you will need it. Deposit any amount of ethers into the contract and then any account to receive the ethers when the time elapses.

The withdrawal function can only be called by the account that deployed the contract. Now in the case that we ain't closed accessing our wallet, what then happens, we will then have to wait till the we do.

But with chainlink automation, we can automate the contract to pay out the funds when the time elapses.

Let's get to that part.

First of all, we will need to set a Time-Based Upkeep to carry out the trigger,

If you are deploying to sepolia, then you will need some sepolia teest net. If its Polygon, then you will need some matic tetsnet.

I deployed my contract to Sepolia, so I will be preceeding with sepolia.

Proceed to [Chainlink Automation to register a new upkeep](#).

Home / Register new Upkeep

Automate your smart contract with Chainlink's hyper-reliable Automation network.

Trigger

Time-based

Target contract address

Enter contract address

What is the contract address?

This is the address of the contract you deployed that contains the function that you want Automation nodes to execute. You will need the Application Binary Interface (ABI) of the deployed contract if you did not verify the contract on chain. The ABI is needed to know how to interact with your contract.

Next

Open in app ↗

Stay updated on the latest from Chainlink

Then select the type of trigger mechanism you want to implement for your contract. For this implementation, am using Time-based automation.

The screenshot shows the 'Trigger' section of the 'Register new Upkeep' form. It asks 'Select the trigger mechanism for automation' and lists three options: 'Time-based' (selected), 'Custom logic', and 'Log trigger'. Below each option is a brief description and a 'Next' button.

Automation triggers

The trigger is used by the Automation network to determine when to run your upkeep. If you use Custom logic or log trigger, you may have additional logic in your contract that will determine "eligibility".

Use time-based if you just want to execute an existing external function using a time schedule.

Use custom logic to trigger using your logic stored in an AutomationCompatibleInterface interface contract.

Use log trigger to trigger following matching on-chain log events and your ILogAutomation interface contract. If you don't have access, please select log trigger and apply to get access.

Learn more about an Automation-compatible contracts.

Need help or have questions? [Talk to an expert](#) or visit the [Automation webpage](#) to learn more

Moving on, you will add the deployed contract address, as well as the ABI of the compiled contract.

The screenshot shows the 'Target contract address' field containing the hex value '0x8129448cF96850182574ae73CB81B688CD1a5542'. Below it, an error message says 'Couldn't fetch ABI' and instructs the user to 'Please paste ABI manually below.' A large blue box highlights the 'ABI' input area, which contains a JSON object:

```

    {
      "type": "constructor"
    },
    {
      "inputs": [],
      "name": "TimeNotReached",
      "type": "error"
    },
    {
      "inputs": []
    }
  
```

ABI

The Application Binary Interface (ABI) is needed to know how to interact with your contract. It can be found in the software you used to compile your contract before you deployed it.

Then you will select the function you want chainlink to automatically trigger when the time you will set reaches.

The screenshot shows the 'Register new Upkeep' page on automation.chain.link. The 'Trigger' section is set to 'Time-based'. The 'Target contract address' is 0x8129...5542. In the 'Contract call' section, the 'Target function' dropdown shows 'deposit' selected, with 'withdraw' also listed. To the right, a box titled 'Select the function and inputs to execute' contains instructions: 'Use the target function drop-down to select the function you want Automation nodes to execute. Only functions that initiate state change can be selected. Where relevant, provide the function inputs to be used during execution.'

Finally, you will set the time that you want the function to be triggered. Since we are locking the fund till a certain time. You have to add a time that will be after the unlocked time you set when deploying the contract.

From the image below, you can see that I set a time (30minutes after), then our function will be triggered.

The screenshot shows the 'Specify your time schedule' section. A blue arrow points to the 'Cron expression' input field, which contains */30 * * * *. Below it, a note says 'Use one of the templates below to schedule your jobs or provide your own cron expression'. Buttons for scheduling include 'Every 15 minutes', 'Every hour', 'First of every month', '30 minutes past every two hours on every weekday', and 'Monday, Wednesday, Friday at 8:00 and 16:00'. To the right, a box titled 'What is a CRON expression?' explains the format: 'The CRON expression is a shorthand way to create a time schedule. Use the provided example buttons to experiment with different schedules and then create your own.' It also states 'Cron schedules are interpreted according to this format:' and provides a detailed diagram of the cron expression structure:

```

minute (0 - 59)
hour (0 - 23)
day of the month (1 - 31)
month (1 - 12)
day of the week (0 - 6) (Sunday to Saturday)
* * * * *
All times are in UTC

```

After following every step, you should see an overview like the image below:

The screenshot shows the Chainlink Automation interface on a web browser. At the top, it displays the URL `automation.chain.link/sepolia/4373543103554255925947189336449196053717090328259277013184...`. Below the header, there are tabs for "Chainlink" and "Automation". The main area is titled "Details" and contains two sections: "Registration" and "Upkeep".

- Registration:**
 - Owner address: 0x1b6e...8e92
 - Date: November 13, 2023 at 23:46 UTC
 - Transaction Hash: 0x4473...6ac4
 - Forwarder address: 0x30f7...A8F5
- Upkeep:**
 - ID: 437354...6451
 - Upkeep address: 0x6503...953D
 - Gas limit: 500,000

On the right side, there is a "View Function" panel which includes a "Contract call" section and a "Function inputs" section.

After 30 minutes, you will see that the withdraw function was triggered. The contract I wrote and deployed had the withdrawn function triggered by chainlink.

The screenshot shows the same Chainlink Automation interface, but now the "History" tab is selected. It displays a table of events:

Date UTC	Transaction hash	Activity type	Trigger	Amount LINK	Balance LINK
November 14, 2023 at 00:00	0x97e3...4c6f	Perform Upkeep	Condition	-0.010095006310090578	6.99
November 13, 2023 at 23:46	0x4473...6ac4	Fund Upkeep		7	7

A black arrow points from the text "Next projected times" in the sidebar to the "Activity type" column of the table, specifically pointing to the "Perform Upkeep" entry.

Need help or have questions? [Talk to an expert](#) or visit the [Automation webpage](#) to learn more.

With the illustration above, you can do more with chainlink automation. Different concepts can be implemented using this feature provided by chainlink. You can at anytime, stop the time trigger action, pause and resume the trigger actions, etc.

Applications like staking contracts, lottery games, voting contracts, etc can all be built and have its functions triggered by time, an event or a customized logic.

You can explore other functionalities as well, also try implementing them too.

[Chain Link](#)[Smart Contracts Tutorial](#)[Chainlink Automation](#)[Smart Contract Blockchain](#)[Solidity Tutorial](#)[Follow](#)

Written by Joe

82 Followers · Writer for Web3 Writers Guild

Smart Contract. Pisces. Music. Books over People. above all, Gigantura Indica.

More from Joe and Web3 Writers Guild

 Joe

Signature Verification in Smart Contracts

I always like to start explaining things from a real world perspective. Let's say you get a signed message from a person.

3 min read · Oct 9, 2023

 85 +

...

 Natachi Nnamaka in Web3 Writers Guild

Learning From The Biggest DeFi Hacks of January 2024

Introduction

9 min read · Feb 6, 2024

👏 64



+

...



Esther Oche in Web3 Writers Guild

Step-by-Step process of configuring and deploying your own Arbitrum Orbit chain.

Introduction

7 min read · Dec 8, 2023

👏 117



+

...



stock | #563588194

 Joe

Using ethers.js to get an ERC20 token details

In this article, I will be showing you how to set up ethers, and how to write a script that shows an ERC20 token symbol, decimal, supply...

4 min read · Aug 15, 2023

 6  +

...

[See all from Joe](#)[See all from Web3 Writers Guild](#)

Recommended from Medium

Integrating Uniswap V4 into Your Smart Contract: Step-by-Step Guide



BuildBear Team in BuildBear Labs

Integrating Uniswap V4 into Your Smart Contracts: A Step-by-Step Guide

Uniswap V4 represents a significant leap forward for the Uniswap protocol, enhancing the capabilities of decentralized exchanges (DEXes) and...

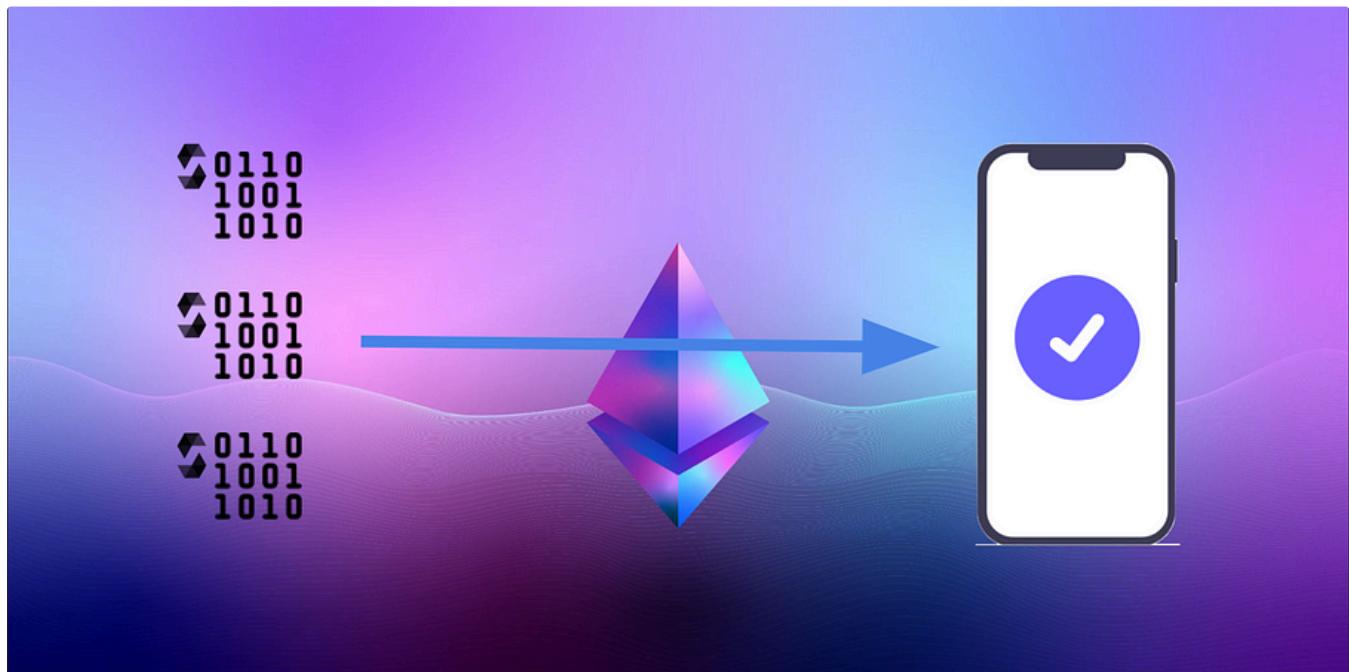
4 min read · Mar 1, 2024



8



...



Leonardo Simone Digiorgio in CoinsBench

Optimizing Smart Contract Interactions: A Guide to Multi Call integration in Web3 Front-End DApp

Each transaction incurs a gas fee. Executing calls individually is costly. Multi Call reduces expenses, offering an efficient solution.

5 min read · Oct 16, 2023



111

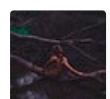


1



...

Lists



Staff Picks

609 stories · 872 saves



Stories to Help You Level-Up at Work

19 stories · 543 saves



Self-Improvement 101

20 stories · 1540 saves



Productivity 101

20 stories · 1430 saves

The advertisement features a dark background with glowing, colorful 3D geometric shapes (red, green, blue) forming a stylized 'E' and 'R'. The text 'Create Your Own ERC404 Token Without Coding in Five Minutes' is displayed in a large, white, sans-serif font. In the bottom left corner, there is a small logo consisting of a stylized 'G' inside a hexagon, followed by the word 'GraphLinq'.

GraphLinq

Create Your Own ERC404 Token Without Coding in Five Minutes

In the dynamic world of cryptocurrency, a new trend is shaking things up: the ERC404 token standard.

4 min read · Feb 17, 2024



...



Hanif Olayiwola

A Developer's Intro to Wormhole and Product Ideas

This a detailed deep dive into developing on Wormhole with 5 potential product ideas that can be built on it

41 min read · Mar 17, 2024



...



 Sena aslibay

10 Most Common Ethereum Smart Contract Vulnerabilities

1.Missing Access Control

9 min read · Oct 11, 2023

 54



...

MEV Bots Switch from Searcher to Solver



 CoW Protocol

MEV Bots: Switch From Searcher to Solver

Searchers make Ethereum worse for everyone, but with just a few tweaks they could contribute to the ecosystem instead

4 min read · Mar 8, 2024



...

See more recommendations