**Introduction to Chainlink Automation**

**Chainlink Automation**

Amazing work! Our project starts looking good!

Looking through it we can see that there's an obvious problem. For the winner to be picked we need someone to call pickWinner. Manually calling this day after day is not optimal, and we, as engineers, need to come up with a better solution! Let's discuss Chainlink Automation!

**Chainlink Automation** is a decentralized service designed to automate key functions and DevOps tasks within smart contracts in a highly reliable, trust-minimized, and cost-efficient manner. It allows smart contracts to automatically execute transactions based on predefined conditions or schedules.

This lesson will be centered on creating a time-based automation using Chainlink's UI. The relevant section in the documentation starts [here](https://docs.chain.link/chainlink-automation/guides/compatible-contracts) and [here](https://docs.chain.link/chainlink-automation/guides/job-scheduler).

In this video, Richard provides a walkthrough on Chainlink’s Keepers, starting with how to connect a wallet from the Chainlink Keepers UI, registering a new upkeep, and implementing time-based trigger mechanisms.

Let's open the contract available [here](https://docs.chain.link/chainlink-automation/guides/compatible-contracts#example-automation-compatible-contract-using-custom-logic-trigger) in Remix by pressing the Open in Remix button.

Following Richard's tutorial let's delete the is AutomationCompatibleInterface inheritance, both the interval and lastTimeStamp variables, adjust the constructor and delete both available functions. Create a new function called count which increments the counter state variable. It should look like this:

// SPDX-License-Identifier: MIT

pragma solidity 0.8.19;

contract Counter {

uint256 public counter;

constructor() {

counter = 0;

}

function count() external {

counter = counter + 1;

}

}

Wow! This is all we need!

Let's deploy this contract on Sepolia. If you are brave enough follow Richard and deploy it to Fuji Avalanche.

Amazing! Check if the counter is 0 by clicking on it! Also check if the count function works by clicking it, signing the transaction and then clicking counter again.

Open up the [Chainlink Automation link](https://automation.chain.link/) and press the blue button saying Register new Upkeep. Connect your wallet. Now we are asked to select a trigger for the automation. Please select Time-based. At the next step, we are asked to provide a Target contract address and copy-paste the address of the contract we just deployed on Sepolia.

Given that we didn't verify the contract we need to provide an ABI. Return to the Remix tab and on the menu on the left select the SOLIDITY COMPILER (It has the Solidity language logo). Ensure the proper contract is selected. Click on ABI, this will copy the ABI in your clipboard. Paste it inside the field Chainlink asks for it and press Next.

At this point, you will have access to a dropdown list containing all the callable functions. Select count ... the only function our contract has. Again press Next.

We need to specify our time schedule, i.e. the amount of time Chainlink Automation needs to wait between function calls. This takes the form of a Cron expression.

Chainlink provides a small but super intuitive tutorial that helps you to craft your Cron expression:

What is a CRON expression?

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.

Cron schedules are interpreted according to this format:

┌───────── 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

- can be used for range e.g. "0 8-16 \* \* \*"

/ can be used for interval e.g. "0 \*/2 \* \* \*"

, can be used for list e.g. "0 17 \* \* 0,2,4"

Special limitations:

no special characters: ? L W #

lists can have a max length of 26

no words

You can find out more about how to properly craft these by playing around with [crontab.guru](https://crontab.guru/) or using your favorite AI chatbot. Even better, you could ask the AI chatbot to craft it for you!

I've configured mine to work every two minutes: \*/2 \* \* \* \*.

After you provide the Cron expression press Next.

Now we got to the Upkeep details section. Give it a name. The Admin Address should be defaulted to the address you used to deploy the contract. You should have some test LINK there. If you don't have any pick some up from [here](https://faucets.chain.link/sepolia). You have the option of specifying a Gas limit. Specify a Starting balance, I've used 10 LINK. You don't need to provide a project name or email address.

Click on Register Upkeep and sign the transactions that pop up.

I had to sign 3 transactions, after that let's click on View Upkeep.

In the History section, you can see the exact date and tx hash of the automated call. Make sure you fund the upkeep to always be above the Minimum Balance. You can fund your upkeep using the blue Actions button. Use the same button to edit your upkeep, change the frequency, or the gas limit, pause the upkeep or cancel it.

From time to time go back to Remix and check the counter value. You'll see it incremented with a number corresponding to the number of calls visible in the History we talked about earlier.

Ok, this was fun, let's pause/cancel the upkeep to save some of that sweet testnet LINK.

Amazing work!