Introduction to Chainlink Keepers

Automate your smart contracts using **Chainlink Keepers**, the decentralized and highly reliable smart contract automation service. Relying on Chainlink Keepers will help you get to market faster and save gas by offloading expensive on-chain automation logic to our decentralized Keepers Network.

## **Select a Trigger**

To start using Chainlink Keepers, you must determine which trigger mechanism to use. The trigger mechanism determines when your function should run. The following triggers are avilable:

* [Time-based trigger](https://docs.chain.link/docs/chainlink-keepers/introduction/#time-based-trigger): If your contract function needs to run repeatedly using a pre-specified time schedule, use a [time based trigger](https://docs.chain.link/docs/chainlink-keepers/introduction/#time-based-trigger).
* [Custom logic trigger](https://docs.chain.link/docs/chainlink-keepers/introduction/#custom-logic-trigger): If your contract requires custom logic to run, use a [custom logic trigger](https://docs.chain.link/docs/chainlink-keepers/introduction/#custom-logic-trigger). Examples of this include checking the balance on a contract, only executing limit orders when their levels are met, or changing the state of certain entities based on some on-chain conditions.

## **Example Contract**

Keepers-compatible contracts must meet the following requirements:

* Import KeepersCompatible.sol. You can refer to the [Chainlink Contracts](https://github.com/smartcontractkit/chainlink/tree/develop/contracts/src" \t "_blank) on GitHub to find the latest version.
* Use the KeepersCompatibleInterface from the library to ensure your checkUpkeep and performUpkeepfunction definitions match the definitions expected by the Keepers Network.
* Include a checkUpkeep function that contains the logic that will be executed off-chain to see if performUpkeep should be executed. checkUpkeep can use on-chain data and a specified checkData parameter to perform complex calculations off-chain and then send the result to performUpkeep as performData.
* Include a performUpkeep function that will be executed on-chain when checkUpkeep returns true. Because performUpkeep is external, users are advised to revalidate conditions and performData.

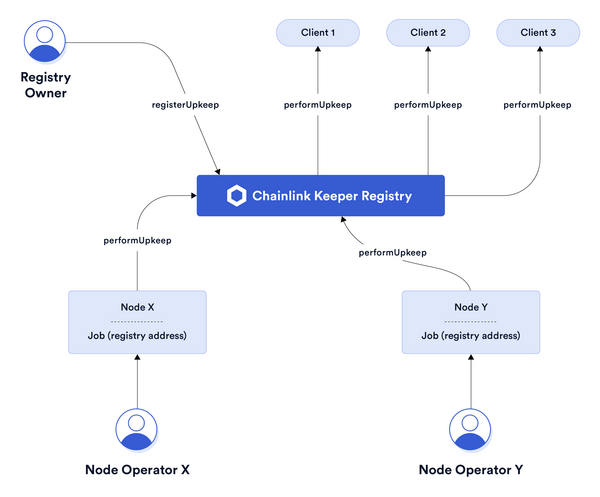
Chainlink Keepers Architecture

## **Overview**

Chainlink Keepers allow smart contracts to outsource regular maintenance tasks in a trust minimized and decentralized manner. The network aims to provide a protocol for incentivization of execution and governance of execution within the Keeper ecosystem.

There are three main actors in the ecosystem:

* **Upkeeps**: These are the maintenance tasks that smart contracts need external entities to service for them. These tasks are just functions on a smart contract and these contracts should be [Keepers-compatible](https://docs.chain.link/docs/chainlink-keepers/compatible-contracts/).
* **Keepers registry**: The contract through which anyone can [register](https://docs.chain.link/docs/chainlink-keepers/register-upkeep/), and manage, their **Upkeeps**.
* **Keepers**: Nodes in the Keepers Network that service registered and funded Upkeeps in the Keepers registry.



## **How it works**

Keepers take responsibility for Upkeeps in turns. Each turn is counted in blocks. See the [configuration](https://docs.chain.link/docs/chainlink-keepers/supported-networks/#configurations) section to find the current block count per turn for your network. The registered Upkeeps are broken into buckets based on the number of Keepers on the network. At the end of each turn, the buckets rotate from one Keeper to the next. Even if a Keeper goes down, we have built-in redundancy and your Upkeep will be performed by the next Keeper in line.

During every block the Keeper will check if the Upkeep is eligible using off-chain compute (a simulation), and then broadcast them on-chain when eligible.

Once a Keeper has performed an Upkeep, it cannot do so again until another Keeper on the network has subsequently performed the same Upkeep. This protects against a faulty or malicious Keeper from taking repeated action on a given Upkeep.

Refer - <https://docs.chain.link/docs/chainlink-keepers/> -for keeper docs