Defining function inputs

If the function you are automating accepts arguments you have two alternative ways that you can define the arguments.

Pre-define inputs (No resolver)

Pre-defining the function arguments would mean that every time Gelato calls the function, it would be using the same argument.

Dynamic inputs via Resolver

By using a resolver, you can dynamically encode the arguments of the function you are automating.

Here is an example:

This is the function we are automating.increaseCount increases a counter on the smart contract byamount which is the argument.

```

Copy functionincreaseCount(uint256amount)external{ require( ((block.timestamp-lastExecuted)>300), "Counter: increaseCount: Time not elapsed" );

count+=amount; lastExecuted=block.timestamp; }

```

This resolver returns the data to the function callincreaseCount .

```

Copy functionchecker() external view override returns(boolcanExec,bytesmemoryexecData) { uint256lastExecuted=ICounter(COUNTER).lastExecuted();

canExec=(block.timestamp-lastExecuted)>300;

uint256countToIncrease=ICounter(COUNTER).count*2

execPayload=abi.encodeCall( ICounter.increaseCount, (countToIncrease) ); }

```

The increment on each execution is different every time ascountToIncrease is different after every execution.

Previous What tasks can be automated?Next Custom logic triggers Last updated3 months ago On this page *Pre-define inputs (No resolver) * Dynamic inputs via Resolver