Tracking your Relay Request

Learn how to check the status of your relay request When submitting your Gelato Relay requests, you'll receive ataskId in response. ThistaskId allows you to track the status of your request in two primary ways:

1. Websocket Subscriptions
2. : This is the recommended and most efficient method. By subscribing via websocket, the Gelato backend will automatically push updates for all your tasks to your Relay SDK client. To start receiving these updates, you must register a callback function which will be triggered every time one of your tasks gets updated.
3. Polling for Updates
4. : Alternatively, you can periodically query the Gelato task status API for updates. If you're using the Gelato Relay SDK, thegetTaskStatus
5. method makes this easy.
6.

For both methods, if you aren't using the Gelato Relay SDK package, you can still interact directly with the websocket or REST APIs, as detailed in the documentation linkedhere .

Websocket Subscriptions

Using Gelato Relay SDK

Support for Websocket Subscriptions was introduced in Gelato Relay SDK version5.5.0 , make sure to update your package. You can subscribe to websocket updates by registering a callback handler function like this:

```

Copy import{ GelatoRelay,TransactionStatusResponse }from"@gelatonetwork/relay-sdk";

constgelatoRelay=newGelatoRelay();

gelatoRelay.onTaskStatusUpdate((taskStatus:TransactionStatusResponse)=>{ console.log("Task status update",taskStatus); });

constrequest:SponsoredCallRequest={ chainId, target, data, };

constresponse=awaitgelatoRelay.sponsoredCall( request, sponsorApiKey );

```

Using websocket API

You can interact with the websocket API directly by connecting to this endpoint:

```

Copy wss://api.gelato.digital/tasks/ws/status

```

Once connected, you can subscribe to updates usingtaskIds of your submitted tasks by sending messages like this:

```

Copy { action:"subscribe", taskId:"0x..." }

```

To unsubscribe from updates:

```

Copy { action:"unsubscribe", taskId:"0x..." }

```

Polling for Updates

Using Gelato Relay SDK

To query the latest task status you can use the following method:

```

Copy import{ GelatoRelay }from"@gelatonetwork/relay-sdk";

constgelatoRelay=newGelatoRelay();

constrequest:SponsoredCallRequest={ chainId, target, data, };

constresponse=awaitgelatoRelay.sponsoredCall( request, sponsorApiKey );

consttaskStatus=awaitgelatoRelay.getTaskStatus( response.taskId );
```

Querying from Gelato API

```

Copy https://api.gelato.digital/tasks/status/:taskId
```

For example, if yourtaskId returned from your Relay response is:

```

Copy { taskId:0x93a3defc618ff97c32a37bdd567b15c50748a5c3e8e858bca67f0c967b74a7fe }
```

then the URL to go to is:

```

Copy https://api.gelato.digital/tasks/status/0x93a3defc618ff97c32a37bdd567b15c50748a5c3e8e858bca67f0c967b74a7fe
```

For thistaskId , here is the returned task information:

```

Copy { "task":{ "chainId":5, "taskId":"0x93a3defc618ff97c32a37bdd567b15c50748a5c3e8e858bca67f0c967b74a7fe",
"taskState":"ExecSuccess", "creationDate":"2022-10-10T10:15:03.932Z", "executionDate":"2022-10-10T10:15:28.718Z",
"transactionHash":"0x9d260d1bbe075be0cda52a3271df062748f3182ede91b3aae5cd115f7b26552b",
"blockNumber":7744557 } }
```

Task Status Response

The task status response object has the following format:

```

Copy typeTransactionStatusResponse={ chainId:number; taskId:string; taskState:TaskState; creationDate:string;
lastCheckDate?:string; lastCheckMessage?:string; transactionHash?:string; blockNumber?:number; executionDate?:string;
gasUsed?:string; effectiveGasPrice?:string; };

enumTaskState{ CheckPending="CheckPending", ExecPending="ExecPending",
WaitingForConfirmation="WaitingForConfirmation", ExecSuccess="ExecSuccess", ExecReverted="ExecReverted",
Cancelled="Cancelled", }
```

Task states

For thetaskState key, these are the possible values:

- CheckPending
- : the relay request has been received by Gelato Relay (pending simulation).

- ExecPending
- : the relay task is executable and is awaiting inclusion into the blockchain.
- WaitingForConfirmation
- : the task was included into the blockchain but is still awaiting the required amount of blocks confirmations.
- ExecSuccess
- : the task has been successfully executed.
- Cancelled
- : the task has been cancelled due to failed simulations or other errors. The error message will be shown in thelastCheckMessage
- key.
- ExecReverted
- : the task transaction has been reverted.
-

What if my task is cancelled?

If your task is cancelled, you can find your error message under thelastCheckMessage key, for example:

```

Copy { "task":{ "chainId":56, "taskId":"0x5f0200652404f9f113a757b4208984f7f4ca25754ddd5c49ca28330e72160c12", "taskState":"Cancelled", "creationDate":"2023-03-03T14:01:14.327Z", "lastCheckDate":"2023-03-03T14:01:44.128Z", "lastCheckMessage":"Execution error: GelatoRelay.sponsoredCall:root already sent" } }

```

The error message in this case refers to the target contract reverting with the message "root already sent" when being called by Gelato Relay'ssponsoredCall function. If you get something similar and you are stuck on troubleshooting, pleaseget in touch with us via Discord and ask in therelay channel! We will be sure to figure out what's going on.

Previous Optional Parameters Next Supported Networks Last updated4 months ago On this page *Websocket Subscriptions * Using Gelato Relay SDK * Using websocket API * Polling for Updates * Using Gelato Relay SDK * Querying from Gelato API * Task Status Response * Task states * What if my task is cancelled?