Callbacks Callbacks can be used to manage the outcome of your transaction submission. This advanced feature enables your functions to adapt based on the execution status, whether successful or not, thus providing a robust way to handle different scenarios that may occur during task execution. Let’s explore the two types of callbacks available:

Callback Function Example:

```

Copy import{ Web3Function, Web3FunctionContext, Web3FunctionFailContext, Web3FunctionSuccessContext, }from"@gelatonetwork/web3-functions-sdk"; import{ Contract }from"@ethersproject/contracts"; importkyfrom"ky";// Using ky for HTTP requests

constORACLE_ABI=[ "function lastUpdated() external view returns(uint256)", "function updatePrice(uint256)", ];

// Callback for successful execution Web3Function.onSuccess(async(context:Web3FunctionSuccessContext)=>{ const{transactionHash}=context; //onSuccess Logic goes here });

// Callback for handling failures Web3Function.onFail(async(context:Web3FunctionFailContext)=>{ const{reason,transactionHash,callData}=context; //onFail Logic goes here });

// Main function logic Web3Function.onRun(async(context:Web3FunctionContext)=>{ const{userArgs,multiChainProvider}=context;

// Core logic goes here to prepare callData

return{ canExec:false, message:"Nothing to execute yet" }; });

```

Types of Callbacks

onSuccess Callback

This callback gets invoked after a successful on-chain execution. It’s especially useful for tracking successful transactions or for further processing after a task completes.

```

Copy Web3Function.onSuccess(async(context:Web3FunctionSuccessContext)=>{ const{transactionHash}=context; console.log("onSuccess: txHash: ",transactionHash); });

```

The Web3FunctionSuccessContext offers access to the transactionHash, allowing you to reference and track the successful transaction within your application.

onFail Callback

Triggered when an on-chain execution encounters issues such as

- InsufficientFunds
- : When the account executing the function does not have enough balance to cover the transaction fees.
- SimulationFailed
- : If the execution simulation (a pre-run of the transaction) fails, indicating that the actual transaction might also fail.
- ExecutionReverted
- : When the actual transaction is executed on the blockchain but is reverted due to a condition in the smart contract code or because it runs out of gas.
-

This callback is crucial for handling errors and implementing fallback logic.

```

Copy Web3Function.onFail(async(context:Web3FunctionFailContext)=>{ const{reason}=context;

if(reason==="ExecutionReverted") { console.log(onFail:{reason}txHash:{context.transactionHash}); }elseif(reason==="SimulationFailed") { console.log( onFail:{reason}callData:{JSON.stringify(context.callData)} ); }else{ console.log(onFail:{reason}); } });
```

```

In the context of the onFail callback:

- reason
- : This is a string indicating why the failure occurred.
- transactionHash
- : Provided when the reason for failure isExecutionReverted
- , this is the unique identifier of the reverted transaction.
- callData
- : Available when the reason isSimulationFailed
- , this is the data that was used during the function run, which can be useful for debugging the failure.
-

Testing Your Callbacks

You can test your callbacks locally using specific flags during the test execution. This helps in ensuring that your callbacks function as intended before deployment.

For onFail Callback :

```

Copy yarntestsrc/web3-functions/callbacks/index.ts--logs--onFail

```

```

Copy Web3Function building...

Web3Function Build result: ✓ Schema: web3-functions\oracle-callback\schema.json ✓ Built file: C:\Users\aniru\OneDrive\Desktop\Gelato_internship\w3f-example\web3-functions-hardhat-template.tmp\index.js ✓ File size: 0.64mb ✓ Build time: 150.48ms

Web3Function user args validation: ✓ currency: ethereum ✓ oracle: 0x71B9B0F6C999CBbB0FeF9c92B80D54e4973214da

Web3Function running logs:

> userArgs: undefined onFail: SimulationFailed callData: [{"to":"0x0000000000000000000000000000000000000000","data":"0x00000000"}]

Web3Function onFail result: ✓ Success

Web3Function Runtime stats: ✓ Duration: 0.20s ✓ Memory: 0.00mb ✓ Storage: 0.04kb ✓ Network: 0 req [DL: 0.00kb / UL: 0.00kb] ✓ Rpc calls: 0 Done in 1.33s.

```

ForonSuccesss Callback

```

Copy yarntestsrc/web3-functions/callbacks/index.ts--logs--onSuccess

```

```

Copy Web3Function Build result: ✓ Schema: web3-functions\oracle-callback\schema.json ✓ Built file: C:\Users\aniru\OneDrive\Desktop\Gelato_internship\w3f-example\web3-functions-hardhat-template.tmp\index.js ✓ File size: 0.64mb ✓ Build time: 143.00ms

Web3Function user args validation: ✓ currency: ethereum ✓ oracle: 0x71B9B0F6C999CBbB0FeF9c92B80D54e4973214da

Web3Function running logs:

> userArgs: undefined onSuccess: txHash: undefined

Web3Function onSuccess result: ✓ Success

Web3Function Runtime stats: ✓ Duration: 0.19s ✓ Memory: 0.00mb ✓ Storage: 0.04kb ✓ Network: 0 req [DL: 0.00kb / UL:

0.00kb] ✓ Rpc calls: 0 Done in 1.47s.

```

Previous Private Typescript Functions Next Test, Deploy & Run Typescript functions Last updated3 months ago On this page * 
```