Start a Standard Exit

Exits allow a user to withdraw funds from the OMG Network back onto the root chain. There are two types of exits: standard exits and in-flight exits. This section will cover standard exits.

A standard exit can be initiated by a user for an output generated by a transaction that was included in a valid block.

Implementation

1. Install `omg-js`, `web3`

To access network features from your application, use our official libraries:

Node | Browser | React Native

Requires Node >= 8.11.3 < 13.0.0

Copy

```
npm install @omisego/omg-js web3
```

JavaScript (ESNext)

2. Import dependencies

Staring an exit on the OMG Network involves using 2 `omg-js` objects. Here's an example of how to instantiate them:

Copy

```
import Web3 from "web3";
import { ChildChain, RootChain, OmgUtil } from "@omisego/omg-js";

const web3 = new Web3(new Web3.providers.HttpProvider(web3_provider_url));
const rootChain = new RootChain({ web3, plasmaContractAddress });
const childChain = new ChildChain({ watcherUrl });

// Alice address
const erc20Exit = {
  token: "0xd92e713d051c37ebb2561803a3b5fbabc4962431",
  address: "0x8CB0DE6206f459812525F2BA043b14155C2230C0",
  privateKey: "CD55F2A7C476306B27315C7986BC50BD81DB4130D4B5CFD49E3EAF9ED1EDE4F7"
}

// Bob address
const ethExit = {
  token: OmgUtil.transaction.ETH_CURRENCY,
  address: "0xA9cc140410c2bfEB60A7260B3692dcF29665c254",
  privateKey: "E4F82A4822A2E6A28A6E8CE44490190B15000E58C7CBF62B4729A3FDC9515FD2"
}
```

- `web3_provider_url` - the URL to a full Ethereum RPC node (local or from infrastructure provider, e.g. Infura).
- `plasmaContractAddress` - `CONTRACT_ADDRESS_PLASMA_FRAMEWORK` for defined environment.
- `watcherUrl` - the Watcher Info URL for defined environment (personal or from OMG Network).

3. Start a standard exit

Exits are processed in queues that contain tokens you can use to start an exit. Before you start an exit, the network has to verify that the exit queue for the token used in the UTXO exists.

You can exit only 1 UTXO at a time. Consider merging multiple UTXOs into 1 UTXO if you don't want to exit all of your funds at once with a single UTXO. By default, the priority is given to UTXOs that were created the earliest.

Every exit requires an extra fee called an exit bond as an incentive mechanism for users of the OMG Network to exit honestly and challenge dishonest exits. The bond is currently fixed at an amount estimated to cover the gas cost of submitting a challenge. You can check the current gas cost at ETH Gas Station.

> The standard exit process is the same for both ETH and ERC20 UTXOs. The tutorial shows how to work with ERC20 tokens. For working with ETH, change `erc20Exit` into `ethExit`.

```
async function startStandardExit() {
  // check if the exit queue exists for a given token
  const queueForTokenExists = await rootChain.hasToken(erc20Exit.token);
  if (!queueForTokenExists) {
    // add the exit queue for this token if it doesn't exist
    const addToken = await rootChain.addToken({
      token: erc20Exit.token,
      txOptions: {
        from: erc20Exit.address,
        privateKey: erc20Exit.privateKey,
        gas: gaslLimit
      },
    });
  }

  // get utxo information
  const aliceUtxos = await childChain.getUtxos(erc20Exit.address);
  const aliceUtxoToExit = aliceUtxos.find(
    (i) => i.currency === erc20Exit.token
  );
  const exitData = await childChain.getExitData(aliceUtxoToExit);

  // start a standard exit
  const receipt = await rootChain.startStandardExit({
    utxoPos: exitData.utxo_pos,
    outputTx: exitData.txbytes,
    inclusionProof: exitData.proof,
    txOptions: {
      from: erc20Exit.address,
      privateKey: erc20Exit.privateKey,
      gas: gaslLimit
    },
  });
  console.log("Alice started a standard exit: " + receipt.transactionHash);

  const exitId = await rootChain.getStandardExitId({
    txBytes: exitData.txbytes,
    utxoPos: exitData.utxo_pos,
    isDeposit: aliceUtxoToExit.blknum % 1000 !== 0,
  });
  console.log("Exit id: " + exitId);

  const { msUntilFinalization } = await rootChain.getExitTime({
    exitRequestBlockNumber: receipt.blockNumber,
    submissionBlockNumber: aliceUtxoToExit.blknum,
  });
  console.log("Exit time: " + msUntilFinalization + " ms");
  return receipt;
}
```

> - `gasLimit` - gas limit for your transaction. Please check the current data on Gas Station or similar resources.

A standard exit creates an exit receipt. The receipt has the following structure:

```
{
  "blockHash": "0x11ba9b64986b6a7d5763735d445f77e3b854d41e7041b3c2943e1f9f39c114e4",
  "blockNumber": 7804882,
  "contractAddress": null,
  "cumulativeGasUsed": 1613315,
  "from": "0x8b63bb2b829813ece5c2f378d47b2862be271c6c",
  "gasUsed": 365705,
  "logs": [
    {
      "address": "0x96d5D8bc539694e5fa1ec0dAb0e6327CA9E680F9",
      "topics": [
        "0xe15a4f223f922b625f5fdd941101a23fa0c097e522233d47a7cbea2167e92701",
        "0x00000000000000000000000000617d9cf823fa147946c05adad21e6e417b29e5"
      ],
      "data": "0x0017aa6a1d400002dd231b0000617d9cf823fa147946c05adad21e6e417b29e5",
      "blockNumber": 7804882,
      "transactionHash": "0x63025e84861677312a04e601477916bf948053ce7ca0ed25be41aae4b57aab68",
      "transactionIndex": 11,
      "blockHash": "0x11ba9b64986b6a7d5763735d445f77e3b854d41e7041b3c2943e1f9f39c114e4",
      "logIndex": 2,
      "removed": false,
      "id": "log_e2805243"
    },
    {
      "address": "0x08C569c5774110EB84A80B292e6d6f039E18915A",
      "topics": [
        "0xdd6f755cba05d0a420007aef6afc05e4889ab424505e2e440ecd1c434ba7082e",
        "0x0000000000000000000000008b63bb2b829813ece5c2f378d47b2862be271c6c"
      ],
      "data": "0x00000000000000000000000000617d9cf823fa147946c05adad21e6e417b29e5",
      "blockNumber": 7804882,
      "transactionHash": "0x63025e84861677312a04e601477916bf948053ce7ca0ed25be41aae4b57aab68",
      "transactionIndex": 11,
      "blockHash": "0x11ba9b64986b6a7d5763735d445f77e3b854d41e7041b3c2943e1f9f39c114e4",
      "logIndex": 3,
      "removed": false,
      "id": "log_38e077a9"
    }
  ],
  "logsBloom": "0x00000000000000000020000000000000000000008000000000040000040080000020010010000000000000000000000000
  "status": true,
  "to": "0x08c569c5774110eb84a80b292e6d6f039e18915a",
  "transactionHash": "0x63025e84861677312a04e601477916bf948053ce7ca0ed25be41aae4b57aab68",
  "transactionIndex": 11
}
```

After a standard exit starts, you'll have to wait a certain amount of time before you can process that exit and receive your funds back on Ethereum Network. This time is called a challenge period and equals to two `minExitPeriod` (2 weeks) on the mainnet and 1 day on the Ropsten testnet (see `EXIT_PERIOD` in Environments for the latest info). A challenge period is an incentivized security measure that allows other users to challenge anybody's exits for validity and honesty.

Lifecycle

1. A user calls the `hasToken` function on the `PlasmaFramework` contract to check if there's an exit queue for the token in question. If no exit queue is registered, a user needs to register it using the `addToken` function. The corresponding `PlasmaFramework` contract functions used in this step are `hasExitQueue` and `addExitQueue`.
2. A user calls the `getExitData` function on the Watcher to get the necessary exit data to start a standard exit. A transaction is termed exitable if it is correctly formed and properly signed by the owner(s) of the transaction input(s).
3. A user calls the `startStandardExit` function on the `PaymentExitGame` contract and commits an exit bond to the exit.
4. After a challenge period a user can process this exit.

> You can only exit one UTXO at a time. It is therefore recommended to merge your UTXOs if you would like to exit multiple ones.

Demo Project

This section provides a demo project that contains a detailed implementation of the tutorial. If you consider integrating with the OMG Network, you can use this sample to significantly reduce the time of development. It also provides step-by-step instructions and sufficient code guidance that is not covered on this page.

JavaScript

For running a full `omg-js` code sample for the tutorial, please use the following steps:

1. Clone omg-js-samples repository:

```
git clone https://github.com/omgnetwork/omg-js-samples.git
```

2. Create `.env` file and provide the required configuration values.

3. Run these commands:

```
npm install
npm run start
```

4. Open your browser at http://localhost:3000.

5. Select `Start a Standard ETH Exit` or `Start a Standard ERC20 Exit` on the left side, observe the logs on the right.

> Code samples for all tutorials use the same repository — `omg-js-samples`, thus you have to set up the project and install dependencies only one time.