

Outside Execution

Enabling meta-transactions on Starknet

Outside execution allows external contracts the ability to execute transactions from outside an account contract, thereby creating opportunities for meta-transactions, transaction scheduling (limit orders) etc.

While we wait for native paymasters on Starknet, there are certain use cases that requires giving external contracts access to execute transactions through an account contract. This is currently not possible due to the structure of the `execute` entrypoint in existing account contracts.

In a bid to prevent re-entrancy attacks, the `execute` entrypoint prevents calls from external contracts, which means in order to enable meta-transactions we need to introduce a new entrypoint that allows calls from external contracts, `execute_from_outside`.

...

```
Copy fn execute_from_outside( refself:ContractState, outside_execution:OutsideExecution, signature:Array)->Array>
```

...

Studying the interface above, we notice the `execute_from_outside` entrypoint requires two

parameters:

- The `OutsideExecution`
- struct
- A valid signature
-

In the next section, let's take a look at how you can execute "outside transactions" as an application builder.

Executing from Outside as a dApp developer

To get started executing outside transactions:

1. Build your
2. `OutsideExecution`
3. Struct
- 4.

An `OutsideExecution` struct contains the details of the transaction to be executed from outside.

...

```
Copy struct OutsideExecution{ caller:ContractAddress, nonce:felt252, execute_after:u64, execute_before:u64, calls:Span}
```

...

For an outside execution transaction to be valid, it needs to contain a `caller` (the address allowed to call the `execute_from_outside`), a `nonce` (a unique value to prevent signature reuse), an `execute_after` value (specifying the time after which the transaction can succeed), an `execute_before` value (specifying the time before which the transaction should succeed), and finally an array of calls to be executed.

2. Sign it using EIP-712 typed data hashing

The signature signs over the EIP712 message encoding of `outside_execution`. Dapps are encouraged to request signatures following the EIP712 standard for a clearer UX.

Refer to the standard [here](#).

1. Call the
2. `execute_from_outside`
3. method on the account contract
- 4.

To do this we advise that you first verify that the account contract being interacted with has support for outside execution. To do this, simply call the `supports_interface` method on the account contract:

...

```
Copy letaccount=IErc165Dispatcher{ contract_address:account_address };
letis_supported=account.supports_interface(ERC165_OUTSIDE_EXECUTION_INTERFACE_ID);
```

...

The interface ID to be queried for is:0x68cfd18b92d1907b8ba3cc324900277f5a3622099431ea85dd8089255e4181 .

If it returns true, then you can go ahead to call theexecute_from_outside method:

...

```
Copy letaccount=IOutsideExecutionDispatcher{ contract_address:account_address };
```

...

It's worth noting that at the time of this writing,Outside Execution is still an early access feature on Argent, thus if you need to provide support for it in your dApp, do reach out to us atdapps@argent.xyz .

[Previous Session Keys - \[early access feature\]](#)[Next Paymasters](#)

Last updated8 days ago