

alchemyplatform / learn-solidity-presentations

🔍

📁

👤

<> Code

🕒 Issues

🔗 Pull requests 2

🎬 Actions

📁 Projects

🛡 Security

📧 Ir

learn-solidity-presentations / 3-reverting-transactions / presentation.md

⌵

⋮

 **Dan-Nolan** init

002c6ec · 2 years ago

🕒

80 lines (57 loc) · 1.31 KB

marp

true

Revert

- We talk to a contract with message calls
- A contract can REVERT a call, negating all state changes
- Each calling contract can choose to handle that success, or REVERT as well

learn-solidity-presentations / 3-reverting-transactions / presentation.md

↑ Top

PreviewCodeBlame

Raw📄⬇️🔍⌵⋮

EOA

✖

transaction failure

←

require(success);

Contract A

revert
no state changes

←

revert();

Contract B

onchain

🔗 Message Call Revert

- 🙅 No state changes occur

- 🙅 No value is transferred
- 🙅 No logs are emitted
- 📄 Gas is still spent

Require

Often you'll see `require` used like this:

```
contract X {  
  // shorthand!  
  address owner = msg.sender;  
  
  function ownerOnly() external {  
    // REVERT if not the owner  
    require(msg.sender == owner, "only owner!");  
    // do something owner-y  
  }  
}
```



Revert

Revert can be used with a string `revert("Unauthorized")` or, better yet:

```
contract X {  
  // @notice a non-privileged user attempted to access an admin-only  
  error Unauthorized();  
  
  function adminOnly() external {  
    if (!isAdmin(msg.sender)) {  
      revert Unauthorized();  
    }  
  }  
}
```



👉📄 Gas Efficient!

Assert

Use `assert` with things that should not happen:

```
contract X {  
    function withdraw() external {  
        uint balance = getBalance(msg.sender);  
        sendBalance(msg.sender);  
  
        // they should not still have a balance!  
        assert(getBalance(msg.sender) == 0);  
    }  
}
```

