*1. Bridge Deposit Contract*

# Overview

This is the 3rd of 5 assignments in the Bridge Project.

In this assignment, you will be writing the deposit contract for the source side of the chain.

# Bridge Deposit Contract

Your token bridge will accept ERC20 deposits on the source chain.

In this assignment, you will write the deposit contract which will custody the tokens provided by users.

## Registering tokens

The bridge operator can register new tokens by calling the registerToken() function. The deposit contract needs to keep track of the list of registered tokens, and will only accept deposits of tokens that have been registered by the bridge operator (i.e., the address that deployed the deposit contract). The registered tokens represent the list of token addresses that can be bridged.

## Depositing

Users can deposit ERC20 tokens from the deposit contract. If the token being deposited is on the list of registered tokens, the deposit contract needs to emit a Deposit event so that the bridge operator can see that it's time to mint a BridgeToken on the destination side.

Note that if a user simply "transfers" a token to the deposit contract (i.e., by calling the "transfer" function on the ERC20 contract), the token will **not** be sent over the bridge. Users

*1. Bridge Deposit Contract*

receiver. Instead, the deposit contract needs to use the approve + transferFrom pattern.

# Withdrawing

When a user initiates a withdrawal (on the destination side), the bridge operator will burn bridge-wrapped tokens on the destination side, then call "withdraw()" on the deposit contract to release the “real” assets.

The withdraw function needs to send the send the tokens to the target recipient, but the transaction should be rejected if the function caller does not have the correct Role.

# Assignment

Using the skeleton Source.sol, complete the three functions, with the following functionality.

1. deposit()
    1. Check if the token being deposited has been “registered”
    2. Use the ERC20 “transferFrom” function to pull the tokens into the deposit contract
    3. Emit a “Deposit” event so that the bridge operator knows to make the necessary actions on the destination side
2. withdraw()
    1. Check that the function is being called by the contract owner
    2. Push the tokens to the recipient using the ERC20 “transfer” function
    3. Emit a “Withdraw” event
3. registerToken()

## *1. Bridge Deposit Contract*

2. Check that the token has not already been registered

3. Add the token address to the list of registered tokens

4. Emit a Registration event

Next ▶