pragma solidity ^0.5.5;

import "./SegesCoin.sol";

import "https://github.com/OpenZeppelin/openzeppelin-contracts/blob/release-v2.5.0/contracts/crowdsale/Crowdsale.sol";

import "https://github.com/OpenZeppelin/openzeppelin-contracts/blob/release-v2.5.0/contracts/crowdsale/emission/MintedCrowdsale.sol";

import "https://github.com/OpenZeppelin/openzeppelin-contracts/blob/release-v2.5.0/contracts/crowdsale/validation/CappedCrowdsale.sol";

import "https://github.com/OpenZeppelin/openzeppelin-contracts/blob/release-v2.5.0/contracts/crowdsale/validation/TimedCrowdsale.sol";

import "https://github.com/OpenZeppelin/openzeppelin-contracts/blob/release-v2.5.0/contracts/crowdsale/distribution/RefundablePostDeliveryCrowdsale.sol";

contract SegesCoinSale is Crowdsale, MintedCrowdsale, CappedCrowdsale, TimedCrowdsale, RefundablePostDeliveryCrowdsale {

constructor(uint rate, address payable wallet, SegesCoin token, uint open\_time, uint close\_time, uint goal, uint cap)

Crowdsale (rate, wallet, token)

TimedCrowdsale (open\_time, close\_time)

CappedCrowdsale (cap)

RefundableCrowdsale (goal)

public

{

// constructor can stay empty

}

}

contract SegesCoinSaleDeployer {

address public token\_sale\_address;

address public token\_address;

constructor(

string memory name,

string memory symbol,

address payable wallet,

uint goal

)

public

{

SegesCoin token = new SegesCoin (name, symbol, 0);

token\_address = address(token);

SegesCoinSale seges\_sale = new SegesCoinSale (1, wallet, token, now, now+104 weeks, goal, 300000000000000000000000);

token\_sale\_address = address(seges\_sale);

token.addMinter(token\_sale\_address);

token.renounceMinter();

}

}