FRC758 - Time Framed Token

Overview
An new implementable token standard for time sliced tokens on the Fusion blockchain. Ownership of FRC758 tokens may be divided by time, allowing for a trustless lending mechanism. Owners may assign a use case of their wealth for a clearly defined period of time, after which the usage rights are returned to them.

How it Works
The ownership of each token is defined by time. Different people can own different time frames of the same token. For example, owner A can transfer 10 tokens to owner B, starting Jan 1 and ending Dec 31. During that time, owner B transfers 5 tokens to owner C and can set the start and end times to anytime within owner B's allotted time frame. This can go on and on, but at Dec 31, all 10 tokens once again become the property of owner A and owner B and owner C lose access.

Approvals
Token holders can approve other addresses (operators) to spend their balance of time sliced tokens. This permits operators to transfer tokens as if they were the owner. When an owner approves an operator, they can now spend any and all of owner's tokens. Owner can revoke operator's rights at any time.

Use Cases
Implementations of FRC758 include tokenised BTC and USDT. Holders may swap their BTC and USDT for time framed BTC and USDT. This opens a window of opportunities to them. They can make use of their wealth in a way that guarantees it will be returned to them when expected.

Contract: Ownable
Allows functions to be owner only. Sets “owner” to be deployer by default.

transferOwnership: - Allows owner to change owner to new address, provided it is not zero address.
Emits OwnershipTransferred event.

Contract: Controllable
A list of addresses with “controller” priviliges.
Inherits from Ownable.

addController: - Allows owner to add an array of new controllers, provided they are not zero address.

removeController: - Allows owner to remove an array of controllers, provided they are not zero address.

Abstract Contracts
TimeSlicedTokenBasic
TimeSlicedTokenMetadata
TimeSlicedTokenReceiver

Events
OwnershipTransferred: - address indexed previousOwner, address indexed newOwner

Transfer: - address indexed _from, address index _to, uint256 amount, uint256 tokenStart, uint256 tokenEnd, uint256 newTokenStart, uint256 newTokenEnd

ApprovalForAll: - address indexed _owner, address indexed _operator, bool _approved

Contract: BasicTimeSlicedToken
Basic functionality for time sliced token.
Inherits from TimeSlicedTokenBasic, TimeSlicedTokenMetaData, Controllable.

_**TIMESLICEDTOKEN_RECEIVED**: - (bytes4) Hash of a function call which verifies that a transaction has been safely transferred, see "onTimeSlicedTokenReceived".

MAX_UINT: - (uint256) Maximum uint256: 2^256 - 1.

MAX_BLOCKNUMBER: - (uint256) Maximum block number before timestamps are used instead: 10^9 - 1.

SlicedToken: - (struct) Contains token amount, start block number or start timestamp, and end block number or end timestamp.

balances: - (mapping) The balance of an address with a counter. Balance is defined by a "SlicedToken" struct.

ownedSlicedTokensCount: - (mapping) The count of sliced token balances an address has.

operatorApprovals: - (mapping) Whether an address is approved to transfer the owners tokens.

_**totalSupply**: - (uint256) The total supply of tokens, changes on mint and burn.

BasicTimeSlicedToken: Internal Functions
_**checkRights**: - Requires the input statement to return true.

_**validateAddress**: - Requires the input address to be non-zero address.

_**validateAmount**: - Requires the input amount to be greater than zero.

_**validateTokenStartAndEnd**: - Require that token end time is after token start time. Require that the entire specified time frame takes place either before max block number, or after max block number.
Require that either: token end time is after current block number, or: token end time is after current block timestamp.

_**mint**: - Mints specified amount of tokens to specified address, and updates their balance. Recipient cannot be zero address, amount cannot be zero, minted token start times and end times must be zero and max time (block number or timestamp) respectively.
Emits Transfer event.

_**burn**: - Burns specified amount of tokens from specified address, and updates their balance. Sender cannot be zero address, amount cannot be zero, burned token start times and end times must be zero and max time (block number or timestamp) respectively.
Emits Transfer event.

_**mergeTheSame**: - Loops through the sliced token balances of owner. If there are any duplicates, where two different sliced token balances have the same start and end times, the amounts are merged into one balance. Calls "_removeEmpty".

_**removeEmpty**: - Loops through the sliced token balance of owner. If there are any occurences where amount equals zero, or where the start time is erroneously after end time, that sliced token gets removed. All other sliced token balances are kept.

removeTokenFrom: - Validates token start and end times and calls _removeToken.

_**removeToken**: - Finds sliced tokens in owner's balance corresponding to specified token start time and token end time. This sliced token balance must at least be "amount". If amount specified is less than amount in sliced token balance, the sliced token balance gets split in two with the amount getting removed. If the entire amount is to be transferred, then the specified time slice gets removed from owner's sliced token balance. The sliced token balance gets updated based on which time frame was removed. Returns "_mergeTheSame " for owner if this succeeds, otherwise returns false.

addTokenTo: - Validates token start and end times and calls _addToken, then calls "_mergeTheSame" for owner.

_**addToken**: - Adds the specified amount of tokens to recipient's sliced token balance. If amount is equal to an existing amount in sliced token balance and if the new token start or end times meet exactly with the existing token start or end times, the two sliced tokens get merged. Otherwise, if the new token start and end times meet at the exact same times as an existing sliced token start and end times, the new amount simply gets added to the existing amount for that sliced token balance. If no sliced token balances were merged, then a new sliced token balance is created for the recipient.

_**isContract**: - Returns true if the specified address is a contract.

checkAndCallSafeTransfer: - Returns true if either "_to" address is not a contract, or if call to "onTimeSlicedTokenReceived" matches "_TIMESLICEDTOKEN_RECEIVED".

BasicTimeSlicedToken: User Functions

balanceOf: - Returns three arrays: One representing the amount in each sliced token, one representing the start times of each sliced token, and one representing the end times of each sliced token.

setApprovalForAll: - Allows the owner of a sliced token to approve another specified address to spend all sliced tokens.
Emits ApprovalForAll event.

isApprovedForAll: - Returns true if specified operator address is approved to spend owner's sliced tokens.

isApprovedOrOwner: - Returns true if inputted spender is owner or if inputted spender is approved to spend owner's sliced tokens.

transferFrom: - Allows transfer from one address to a different address, by first removing specified slice from owner's sliced token balances, then adding this slice to recipient's sliced token balance. Owner must not be zero address. Recipient must not be zero address. Amount must be greater than zero. Sender must be approved to spend owner's sliced tokens.
Emits Transfer event.

safeTransferFrom: - Calls "transferFrom" and requires that "checkAndCallSafeTransfer" succeeds. The "data" argument is optional.

totalSupply: - Returns total supply of sliced tokens.

Contract: TimeSlicedToken

TimeSlicedToken: constructor: - Sets name, symbol, and decimal values for token.

name: - Returns the token name.

symbol: - Returns the token symbol.

decimals: - Returns the decimals for this token.

Contract: MyTestToken

MyTestToken: constructor: - Calls airdrop to msg.sender for tokens with max block number as end time, and calls airdrop to msg.sender for tokens with max timestamp as end time.

airdrop: - onlyController callable function that mints specified amount into an address with specified tokenStart and tokenEnd.

onTimeSlicedTokenReceived: - Callable function to verify if a token was safely transferred. The check is done by comparing the hash of the function call to the constant value, "_TIMESLICEDTOKEN_RECEIVED".