Documentation of Staking Contract for CarbSwap

Upon deployment of the contract, call for initializeContract transition to get the contract started after all the following is set.

By default Contract is:

- Paused (Unpause using the transition unpauseContract)

Set the parameters of the contract:

- **Amount of token B to be given per epoch** (using the transition changeTokenBrewardsPerEpoch)
- **Set Minimum Stake**, (using the transition changeMinimumStake) by default 0
- **Setting the rewards interval in blocks** (Using the transition changeRewardBlocks)
- **Setting the unstaking period in blocks** (Using the transition changeUnstakeBlocks)
- **Setting the unstaking percentage of rewards reduction** (Using the transition changeUnstakePercentReward) by default 50% reduction in rewards
- **Setting the instant withdrawal penalty** (Using the transition changeUnstakePenalty) by default 10%.
- **Setting the burnt percentage from penalty**, (Using the transition changeInstantWithdrawalBurntPercent) By default 50%
- **Setting the burnt address to send the burnt tokens to**, by default is the owner of contract (Transition changeInstantWithdrawalBurnAddress)
- **Setting auto withdrawal fee** in the number of tokens A we would deduct from the staked amount to be sent to the sender who assisted the user in auto withdrawal. (Using the transition changeAutoWithdrawalFee)

Deployment of Contract

initial_owner : ByStr20, address of the owner of contract
tokenAContract : ByStr20, address of the staked token
tokenBContract : ByStr20, address of the reward token

Transitions within the Contract

procedure isOwner(), Checks to see if the _sender is the owner of contract.

procedure isPaused(), Checks to see if the contract is paused.

procedure minimumStake (amount: Uint128), Checks to see if staked amount is minimum staked.

procedure addRewards(staker: Pair ByStr20 Uint128), Internal function for distribution of rewards.

procedure penaltyToStakers(staker: Pair ByStr20 Uint128), Internal function for distribution of the penalty rewards to stakers.

transition initializeContract(), Initialize the contract starting blocks and all fields, this must be called and the reward timer will set the next reward block starting from time of this transition call.

transition rewardAll(), Anyone could call this function to distribute the reward of the contract, but block time must be met first.

transition giveBonus(address: ByStr20, bps: Uint128), Only the owner of the contract could call this function and reward a specific address bonus Token A into their pending fields. Owner of the contract must supply sufficient token A into the contract by transferring it directly to the contract address beforehand.

transition addStakedTokenA(amount: Uint128), Stakers must use this transition to add their stake into contract, Different from old staking contract where users could just transfer tokens to the contract.

transition withdrawPendingA(), Stakers will call this function to withdraw any pending rewards of Token A.

transition withdrawPendingB(), Stakers will call this function to withdraw any pending rewards of token B.

transition removeStakeIfEpochZero(), If staker just added stake and wish to remove this function be called to remove stake, only applicable if epoch = 0. If additional stake is added to a new stake it would not reset epoch.

transition removeStake(), Removes their full stake from the contract and enters the unstaking period, if the user calls this function their rewards will be reduced by the specified percentage.

transition withdrawStake(), Only can be called when the unstaking period is over to retrieve out Token A.

transition automaticWithdrawStake(recipient: ByStr20), Anyone could call this function to help unstake an address whom has passed the unstaking blocks, a fee will be given to the caller from the unstaking address and fee is specified within contract (autofee) number of tokens to be given as fee.

transition instantWithdrawal(), Function for instantWithdrawal to bypass the unstaking period, but a penalty will be imposed and the penalty will then be splitted across all stakers and a portion is burnt.

Owner functions

transition changeTokenBrewardsPerEpoch(amount: Uint128), Changes the number of Token B reward per Epoch

transition changeRewardBlocks(block: Uint128), Changes the duration/interval per epoch

transition changeUnstakeBlocks(block: Uint64), Changes the unstaking period in blocks.

transition changeUnstakePercentReward(percent: Uint128), Percentage reduction in rewards, (e.g. 60 is set, 100 carb reward to staker but due to reduction only 40 carb is given)

transition changeUnstakePenalty(percent: Uint128), Changes the penalty percentage in instant withdrawal function by default 10% of the users tokens.

transition changeInstantWithdrawalBurntPercent(percent: Uint128), Burn Penalty tokens percentage, by default 50%. (e.g. penalty is 100 carb, burnt percentage is 40%, 40 carbs be burnt and 60 is distributed across stakers)

transition changeInstantWithdrawalBurnAddress(addr: ByStr20), Sets the address to send the tokens to be burnt. Recommend using a new wallet that is solely meant for burning.

transition changeMinimumStake(amount: Uint128), Changes the minimum stake amount for token A.

transition changeAutoWithdrawalFee(fee: Uint128), Sets the autofee that is the number of tokens to be taken from the staker to be paid to the called of automaticWithdrawal function(e.g. carb is 100000000 = 1 carb, setting that would result in 1 carb being sent to the caller of automaticWithdrawal) to cover gas fees)

transition pauseContract(), Pauses the contract to prevent addition of stakes and rewards from being given.

transition unpauseContract(), Unpause the contract.

transition changeTokenAaddress(addr: ByStr20), Changes token A address

transition changeTokenBaddress(addr: ByStr20), Changes token B address

transition ownerWithdrawAddedTokenA (amount: Uint128), Withdraw any token A supplied by the contract owner

transition ownerWithdrawAddedTokenB (amount: Uint128), Withdraw any token B supplied by the contract owner

transition RequestOwnershipTransfer(new_owner : ByStr20), Changes owner of the contract

transition ConfirmOwnershipTransfer(), 2 step procedure for changing owner

transition RecipientAcceptTransfer(sender : ByStr20, recipient : ByStr20, amount : Uint128), Used for owner of contract to supply tokens for rewards.

transition RecipientAcceptTransferFrom(initiator : ByStr20,sender : ByStr20, recipient : ByStr20, amount : Uint128), Used by the addTokenA function for users to stake their tokens and verification is made if correct amount is staked.

CallBacks functions

transition TransferSuccessCallBack(
  sender : ByStr20,
  recipient : ByStr20,
  amount : Uint128
)

transition TransferFromSuccessCallBack (
  initiator : ByStr20,
  sender : ByStr20,
  recipient : ByStr20,
  amount : Uint128
)