StakBank

READ METHOD

1. function checkDetailStakingRequest(address user, uint idStake)
public view
returns (uint timestamp, uint stakedAmount, uint ethReward, uint usdtReward, bool isUnstaked)

return detail of staking request of each staker.

Params
- user: address of staker
- idStake: id of staking request (count from 1)

Reverted conditions
None

2. function countdownToNextDistribution() public view returns (uint)
return number of seconds to trigger reward distribution.

Params
None

Reverted conditions
None

3. decimal() returns (uint)
return number of decimal

4. function estimateNextDistribution() public view returns (uint)
return estimate timestamp of next distribution

Params
None

Reverted conditions
None

5. function feeCalculator(uint amount) public view returns (uint)
return platform fee with corressponding JST
formula: feePlatform = amount * feePerDecimal / (10 ** decimal)

Params
- amount: staking JST

Reverted conditions
None

6. feePerDecimal() returns (uint)
return feePerDecimal

7. lasDis() returns (uint)
return timestamp of last distribution

8. minAmountToStake returns (uitn)
return minimum amount of JST to stake

9. function numEthToReward() public view returns (uint)
return number of Eth to reward

Params
None

Reverted conditions
None

10. function numUsdtToReward() public view returns (uint)
return number of USDT to reward

Params
None

Reverted conditions
None

11. numberDistribution() returns (uint)
return number of reward distribution

12.function numberOfStakeHolder() public view returns (uint)
return number of staker in pool

Params
None

Reverted conditions
None

13. owner() returns (address)
return admin's address of StakBank

14. paused() returns (bool)
return true if StakBank is paused

15. periodTime() returns (uint)
return period to trigger distribution

16. function stakingOf(address user) public view returns (uint)
return amount of JST a staker staked

17. token() returns (address)
return address of Jig Stack's contract

19. usdt() returns (address)
return address of Tether’s contract

WRITE METHOD

1. function closeStakBank(uint number) public onlyOwner whenNotPaused
send reward and USDT back to number staker (too large “number” may cause gas over)
if number of staker is 0, then send all money from pool to admin

Reverted conditions
when number > numberOfStakeHolder of number   0

2.function pause() onlyOwner whenNotPaused public
make StakBank paused

3.function rewardDistribution() public onlyOwner whenNotPaused
admin trigger distribution

Reverted conditions
- number of JST in pool is 0
- countdownToNextDistribution() > 0

4. function setDecimal(uint _decimal) external onlyOwner whenNotPaused
(formula is above)

5.function setFeePerDecimal(uint _feePerDecimal) external onlyOwner whenNotPaused
(formular is above)

6. function setMinAmountToStake(uint _minAmountToStake) external onlyOwner whenNotPaused

7. function setPeriodTime(uint _periodTime) external onlyOwner whenNotPaused
Reverted conditions
- _preriodTime > 0

8. function stake(uint stakedAmount) public payable whenNotPaused
user stake

Reverted conditions
- Admin cannot stake
- stakedAmount >= minAmountToStake()
- Limit of pool: 10 billions JST
- eth send to pool >= feeCalculator(stakedAmount)

9. function transferOwnership(address _newOwner) public onlyOwner
change owner

10. function unpause() onlyOwner whenPaused public
unpause StakBank

11. function unstakeAll() public whenNotPaused
user unstakeAll
pool sends reward to user
pool sends JST back to user

Reverted conditions
(if user has too many staking request, this may cause over gas => FE use ***unstakeWithId***)
- Not a staker

12. function unstakeWithId(uint idStake) public whenNotPaused
user unstake certain request

Reverted conditions
- that idStake unstaked
- not a staker

13. function withdrawReward() public whenNotPaused
users claim reward. Send eth and usdt to their address.

Reverted conditions
- not a staker