Venture Moloch Contract

1. States
   1. periodDuration

Time unit to limit blocks

* 1. votingPeriodLength

Length of voting period for proposal, default to 35 periods

* 1. gracePeriodLength

Length of admin revert time after voting period, default to 35 periods

* 1. abortWindow

Length of time to abort proposal

* 1. delutionBound

bound for ragequit when process proposal, stop processing if mass ragequit

* 1. summoningTime

Contract created time.

* 1. contributonToken

base token for guild bank accounting.

* 1. tributeToken

token reference for subscription or alternative contribution for members // not using for total, but using for individuals

* 1. guildBank

guild bank contract reference

* 1. totalShares

total shares across all members

* 1. totalSharesRequested

unused

* 1. totalContributed

total contributed amount

* 1. totalWithdrawals

total Withdraw amount

* 1. totalValuePerShare

total contribute amount / total share

* 1. members

all members as map

* 1. memberAddressByDelegateKey

address by delegate key

* 1. ProposalQueue

Array for proposal

* 1. numProposal

count of proposal

* 1. memberAccts

all members as array

* 1. equityHoldingAddress

holding address by illegal order

1. Struct
   1. Member
      * delegateKey

key for submitting proposals and voting, default to member address

* + - shares

count of shares assigned to this member

* + - exists

true if this member has been created

* + - tributeAmount

amount contributed to guild bank

* + - highestIndexYesVote

highest proposal index number which the member voted YES, used for rageQuit

* + - allowedDividends

authorized amount to withdraw

* + - valuePerShare

tributeAmount / shares

* 1. Proposal
     + proposer

the member who submitted the proposal

* + - applicant

applicant who wishes to become a member, will be used for withdraws

* + - tributeAmount

amount of tokens offered as tribute

* + - tributeToken

the tribute token reference for subscription

* + - fundsRequests

the funds requested for applicant

* + - details

proposal details – could be IPFS hash, plaintext or JSON

* + - startingPeriod

the period in which voting can start for this proposal

* + - yesVotes

total number of YES votes for this proposal

* + - noVotes

total number of NO votes for this proposal

* + - processed

true if the proposal has been processed

* + - didPass

true if the proposal passed

* + - fundsTransferred

true if the funds transferred to applicant

* + - aborted

true if applicant calls ‘abort’ before end of voting period

* + - maxTotalSharesAtYesVote

max number of share when yes vote to this proposal

* + - votesByMember

the votes on this proposal by each member

1. Events
   1. SubmitProposal()
   2. SubmitVote()
   3. ProcessProposal()
   4. Ragequit()
   5. Abort()
   6. UpdateDelegateKey()
   7. MemberAdded()
   8. DividendDeclared()
   9. ProposalFunded()
   10. Deposit()
2. Functions
   1. constructor()

Initialize contract

* + - initialize contribution token
    - initialize guild bank
    - initialize some state params

periodDuration, votingPeriodLength, gracePeriodLength, abortWindow, dilutionBound, summoningTime

* 1. submitProposal(params)

Submit proposal

* + - Collect token from applicant to this contract address
    - Make a Proposal Instance with input params and push it to ProposalQueue
    - Emit SumitProposal()
  1. submitVoteOnProposal()

Vote to a Proposal

* + - Require current time if after startingPeriod for the proposal
    - Require the Vote is Yes/No, and the Proposal is not aborted
    - Register Vote to the Proposal with member address
    - If Vote is YES, update highestIndexYesVote of the member and maxTotalSharesAtYesVote of the Proposal
    - Emit SumitVote()
  1. processProposal()

Process Proposal

* + - Require current time is after voting + grace time for proposal
    - Require the Proposal is not processed yet
    - Require the previous Proposal is processed
    - Check the dilutionBound is true
    - If not, return all tribute token to the applicant
    - If yes, Emit ProcessProposal()
  1. fundApprovedProposal()

Fund a project that has passed and been processed.

* + - Require the Proposal has been passed
    - Increase total withdrawals
    - Add the tributeToken to equityHoldingAddress
    - Withdraw funds from guildBank to applicant of the Proposal
    - Send tributed Tokens to guildBank
    - Emit ProposalFunded()
  1. rageQuit()

quit immediately

* + - Require the highest index proposal with YES voted is processed
    - Increase total withdrawals
    - Decrease total value per share
    - Remove member from memberAccts
    - Withdraw token of the member from guildBank to the member
    - Emit Ragequit()
  1. abortProposal()

An applicant can cancel their proposal within grace period

* + - Require sender is same with applicant
    - Require current time is before abortWindow period
    - Withdraw tributeToken to applicant
    - Emit Abort()
  1. updateDelegateKey()

Proxy for voting rights of member

* + - Require the delegate key is not exists on members
    - Update delegate key on member
  1. adminWithdrawAsset()

Summoner withdraws and administers tribute token

* + - Require asset token is same as contribution token
    - Withdraw tokens from guildBank to receiver
  1. addMember()

Add member without submit proposal process

* + - Require same member address is not exists
    - Add member to members
    - Set shares with 100000
    - Add totalContributed
    - Update totalValuePerShare
    - Send tokens from new Member to guildBank
    - Add Member to memberAccounts
    - Emit MemberAdded()
  1. getMemberEconomicWeight()

Calculate Economic weight for member

Return valuePerShare / totoalValuePerShare

* 1. removeMember()

Remove member

* + - Require shares > 0
    - Update totalShares
    - Update totalWithdrawals
    - Withdraw tokens from guildBank to currentMember
    - Emit RageQuit()
  1. declareDividend()

Update dividend amount

* + - Calculate total available
    - Update allowedDividends for all members
    - Emit DividendDeclared()
  1. withdrawAuthorizedDividend()

withdraw allowed dividends

* + - Update totalWithdrawal s
    - Withdraw dividends tokens from buildBank to sender
    - Emit DividendWithdrawn()
  1. deposit()

deposit money to guildBank

* 1. max()

math function for max

* 1. getCurrentPeriod()

Get current duration after create contract

* 1. getMember()

return all member

* 1. getMemberDetails()

return member detail

* 1. getProposalDetails()

return proposal detail

* 1. getProposalQueueLength()

return proposal count

* 1. canRageQuit()

return if last Yes voted proposal is processed

* 1. hasVotingPeroidExpired()

return if voting Period is expired

* 1. getProposalVote()

return vote of member for proposal

* 1. getEnquityBalance

return balance of guildBank for token

* 1. getEnquityTokenAddresses()

return holding addresses

* 1. getTotalAvailable()

return total balance of guildBank

* 1. whenProposalVotingPeriodEnd()

return Ending Period for a Proposal