Technical

`delegate(address delegatee)` (public)

Delegate votes from `msg.sender` to `delegatee`

`delegateBySig(address delegatee, uint256 nonce, uint256 expiry, uint8 v, bytes32 r, bytes32 s)` (public)

Delegates votes from signatory to `delegatee`

`getCurrentVotes(address account) → uint256` (external)

Gets the current votes balance for `account`

`getPriorVotes(address account, uint256 blockNumber) → uint256` (public)

Determine the prior number of votes for an account as of a block number

Block number must be a finalized block or else this function will revert to prevent misinformation.

`_delegate(address delegator, address delegatee)` (internal)

`_moveDelegates(address srcRep, address dstRep, uint256 amount)` (internal)

`_writeCheckpoint(address delegatee, uint32 nCheckpoints, uint256 oldVotes, uint256 newVotes)` (internal)

`safe32(uint256 n, string errorMessage) → uint32` (internal)

`addCreditETH(address job)` (external)

Add ETH credit to a job to be paid out for work

addCredit(address credit, address job, uint256 amount) (external)

Add credit to a job to be paid out for work

approveLiquidity(address liquidity) (external)

Approve a liquidity pair for being accepted in future

revokeLiquidity(address liquidity) (external)

Revoke a liquidity pair from being accepted in future

pairs() → address[] (external)

Displays all accepted liquidity pairs

addLiquidityToJob(address liquidity, address job, uint256 amount) (external)

Allows liquidity providers to submit jobs

`applyCreditToJob(address provider, address liquidity, address job)` (external)

Applies the credit provided in addLiquidityToJob to the job

`unbondLiquidityFromJob(address liquidity, address job, uint256 amount)` (external)

Unbond liquidity for a pending keeper job

`removeLiquidityFromJob(address liquidity, address job)` (external)

Allows liquidity providers to remove liquidity

`mint(uint256 amount)` (external)

Allows governance to mint new tokens to treasury

`burn(uint256 amount)` (external)

burn owned tokens

`_mint(address dst, uint256 amount)` (internal)

`_burn(address dst, uint256 amount)` (internal)

`workReceipt(address keeper, uint256 amount)` (external)

Implemented by jobs to show that a keeper performend work

`receipt(address credit, address keeper, uint256 amount)` (external)

Implemented by jobs to show that a keeper performend work

`receiptETH(address keeper, uint256 amount)` (external)

Implemented by jobs to show that a keeper performend work

`_bond(address bonding, address _from, uint256 _amount)` (internal)

`_unbond(address bonding, address _from, uint256 _amount)` (internal)

`addJob(address job)` (external)

Allows governance to add new job systems

`getJobs() → address[]` (external)

Full listing of all jobs ever added

`removeJob(address job)` (external)

Allows governance to remove a job from the systems

`setKeep3rHelper(contract Keep3rHelper _kprh)` (external)

Allows governance to change the Keep3rHelper for max spend

`setGovernance(address _governance)` (external)

Allows governance to change governance (for future upgradability)

acceptGovernance() (external)

Allows pendingGovernance to accept their role as governance (protection pattern)

isKeeper(address keeper) → bool (external)

confirms if the current keeper is registered, can be used for general (non critical) functions

isMinKeeper(address keeper, uint256 minBond, uint256 earned, uint256 age) → bool (external)

confirms if the current keeper is registered and has a minimum bond, should be used for protected functions

isBondedKeeper(address keeper, address bond, uint256 minBond, uint256 earned, uint256 age) → bool (external)

confirms if the current keeper is registered and has a minimum bond, should be used for protected functions

bond(address bonding, uint256 amount) (external)

begin the bonding process for a new keeper

`getKeepers() → address[] (external)`

get full list of keepers in the system

`activate(address bonding) (external)`

allows a keeper to activate/register themselves after bonding

`unbond(address bonding, uint256 amount) (external)`

begin the unbonding process to stop being a keeper

`withdraw(address bonding) (external)`

withdraw funds after unbonding has finished

`down(address keeper) (external)`

slash a keeper for downtime

`dispute(address keeper) → uint256 (external)`

allows governance to create a dispute for a given keeper

`slash(address bonded, address keeper, uint256 amount)`

(public)

allows governance to slash a keeper based on a dispute

`revoke(address keeper)` (external)

blacklists a keeper from participating in the network

`resolve(address keeper)` (external)

allows governance to resolve a dispute on a keeper

`allowance(address account, address spender) → uint256`

(external)

Get the number of tokens `spender` is approved to spend on behalf of `account`

`approve(address spender, uint256 amount) → bool`

(public)

Approve `spender` to transfer up to `amount` from `src`

This will overwrite the approval amount for `spender` and is subject to issues noted here

`permit(address owner, address spender, uint256 amount, uint256 deadline, uint8 v, bytes32 r, bytes32 s)`

(external)

Triggers an approval from owner to spends

`balanceOf(address account) → uint256` (external)

Get the number of tokens held by the `account`

`transfer(address dst, uint256 amount) → bool`

(public)

Transfer `amount` tokens from `msg.sender` to `dst`

`transferFrom(address src, address dst, uint256 amount) → bool`

(external)

Transfer `amount` tokens from `src` to `dst`

`_transferTokens(address src, address dst, uint256 amount)`

(internal)

_getChainId() → uint256 (internal)

DelegateChanged(address delegator, address fromDelegate, address toDelegate)

An event thats emitted when an account changes its delegate

DelegateVotesChanged(address delegate, uint256 previousBalance, uint256 newBalance)

An event thats emitted when a delegate account's vote balance changes

Transfer(address from, address to, uint256 amount)

The standard EIP-20 transfer event

Approval(address owner, address spender, uint256 amount)

The standard EIP-20 approval event

SubmitJob(address job, address provider, uint256 block, uint256 credit)

Submit a job

ApplyCredit(address job, address provider, uint256 block, uint256 credit)

Apply credit to a job

RemoveJob(address job, address provider, uint256 block, uint256 credit)

Remove credit for a job

UnbondJob(address job, address provider, uint256 block, uint256 credit)

Unbond credit for a job

JobAdded(address job, uint256 block, address governance)

Added a Job

JobRemoved(address job, uint256 block, address governance)

Removed a job

KeeperWorked(address credit, address job, address keeper, uint256 block)

Worked a job

KeeperBonding(address keeper, uint256 block, uint256 active, uint256 bond)

Keeper bonding

KeeperBonded(address keeper, uint256 block, uint256 activated, uint256 bond)

Keeper bonded

KeeperUnbonding(address keeper, uint256 block, uint256 deactive, uint256 bond)

Keeper unbonding

KeeperUnbound(address keeper, uint256 block, uint256 deactivated, uint256 bond)

Keeper unbound

KeeperSlashed(address keeper, address slasher, uint256 block, uint256 slash)

Keeper slashed

KeeperDispute(address keeper, uint256 block)

Keeper disputed

KeeperResolved(address keeper, uint256 block)

Keeper resolved

AddCredit(address credit, address job, address creditor, uint256 block, uint256 amount)