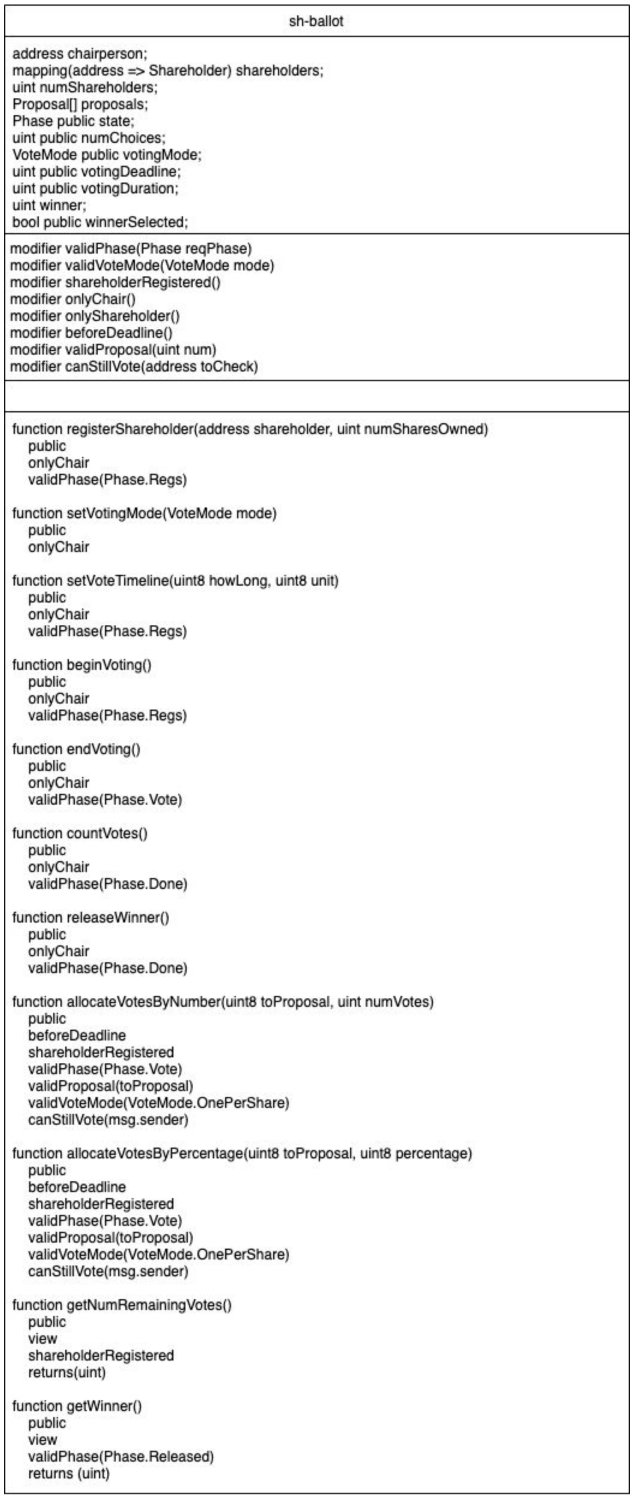
Smart Contract Diagram



```
sh-ballot

address chairperson;
mapping(address => Shareholder) shareholders;
uint numShareholders;
Proposal[] proposals;
Phase public state;
uint public numChoices;
VoteMode public votingMode;
uint public votingDeadline;
uint public votingDuration;
uint winner;
bool public winnerSelected;

modifier validPhase(Phase reqPhase)
modifier validVoteMode(VoteMode mode)
modifier shareholderRegistered()
modifier onlyChair()
modifier onlyShareholder()
modifier beforeDeadline()
modifier validProposal(uint num)
modifier canStillVote(address toCheck)

function registerShareholder(address shareholder, uint numSharesOwned)
    public
    onlyChair
    validPhase(Phase.Regs)

function setVotingMode(VoteMode mode)
    public
    onlyChair

function setVoteTimeline(uint8 howLong, uint8 unit)
    public
    onlyChair
    validPhase(Phase.Regs)

function beginVoting()
    public
    onlyChair
    validPhase(Phase.Regs)

function endVoting()
    public
    onlyChair
    validPhase(Phase.Vote)

function countVotes()
    public
    onlyChair
    validPhase(Phase.Done)

function releaseWinner()
    public
    onlyChair
    validPhase(Phase.Done)

function allocateVotesByNumber(uint8 toProposal, uint numVotes)
    public
    beforeDeadline
    shareholderRegistered
    validPhase(Phase.Vote)
    validProposal(toProposal)
    validVoteMode(VoteMode.OnePerShare)
    canStillVote(msg.sender)

function allocateVotesByPercentage(uint8 toProposal, uint8 percentage)
    public
    beforeDeadline
    shareholderRegistered
    validPhase(Phase.Vote)
    validProposal(toProposal)
    validVoteMode(VoteMode.OnePerShare)
    canStillVote(msg.sender)

function getNumRemainingVotes()
    public
    view
    shareholderRegistered
    returns(uint)

function getWinner()
    public
    view
    validPhase(Phase.Released)
    returns (uint)
```