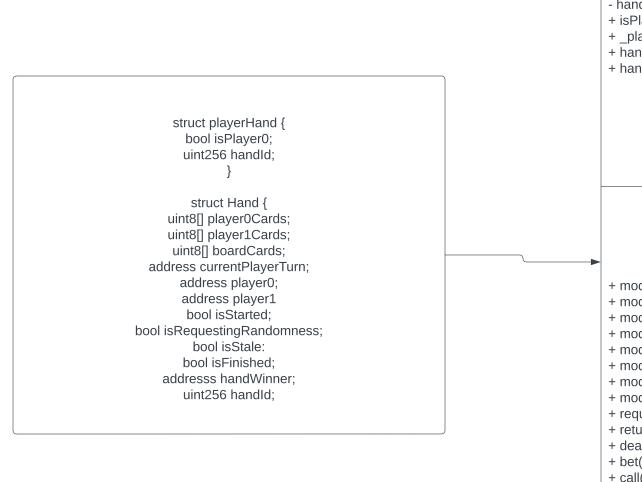
```
POK3R VRF
requestQueue:randomRequests[]
- vrfCoordinator:address = 0x7a1BaC17Ccc5b313516C5E16fb24f7659aA5ebed (polygon mumbai)
- keyHash:bytes32 = 0x4b09e658ed251bcafeebbc69400383d49f344ace09b9576fe248bb02c003fe9f
- callbackGasLimit:uint32 = 2 500 000
- requestConfirmations:uint16 = 3
- uint32:numWords = 10
- isPendingVRFFulfill:bool = false
- requestCounter:Counters.Counter = 0
- subscriptionId:uint64
- requestIndexToRandomRequest:mapping(uint256 => randomRequest)
- randomRequest:struct
- requestFulfilled:event(uint256 requestIndex, uint256 handId, uint8 randomCard, randomRequestType
_randomRequestType)
requestPlayer0Deal(uint256 handId)
requestPlayer1Deal(uint256 handId)
- requestFlop(uint256 handId)
requestTurn(uint256 handId)
requestRiver(uint256 handId)
- sendRandomRequest(uint256 handId, randomRequestType _randomRequestType)
requestRandomWords()
- fulfillRandomWords(uint256 requestId, uint256[] randomWords)
- fulfillRandomRequest(uint256 requestIndex, uint8 randomNumber)
```



+ uint256:minBet = 0.1 Ether + uint256:maxBet = 10 Ether - handCounter:Counters.counter = 0 + isPlaying:mapping(address => bool) = false + _playerHand:mapping(address => playerHand) = { false, 0 } + handOwners:mapping(uint256 => address[]) = [0x00...., 0x00....] + handIdToHand:mapping(uint256 => Hand) + modifier onlyHandOwner(uint256 _handId) + modifier onlyPlayableHand(uint256 handId) + modifier onlyPlayer0(uint256 _handId) + modifier onlyPlayer1(uint256 handId) + modifier onlyNotRequestingRandomness(uint256 _handId) + modifier onlyBetWithinLimits() + modifier onlyPlayOneHand() + modifier onlyAlreadyPlaying() + requestGameWithPlayer(address otherPlayer) + returnBetFromNotStartedGame() + dealHoleCards() + bet(uint256 amount) + call() + raise(uint256 amount) + fold() + resolveHand() + retrieveWinningsFromPlayerTimeout + returnBetFromNotFulfilledRandomRequest

POK3R Game Logic