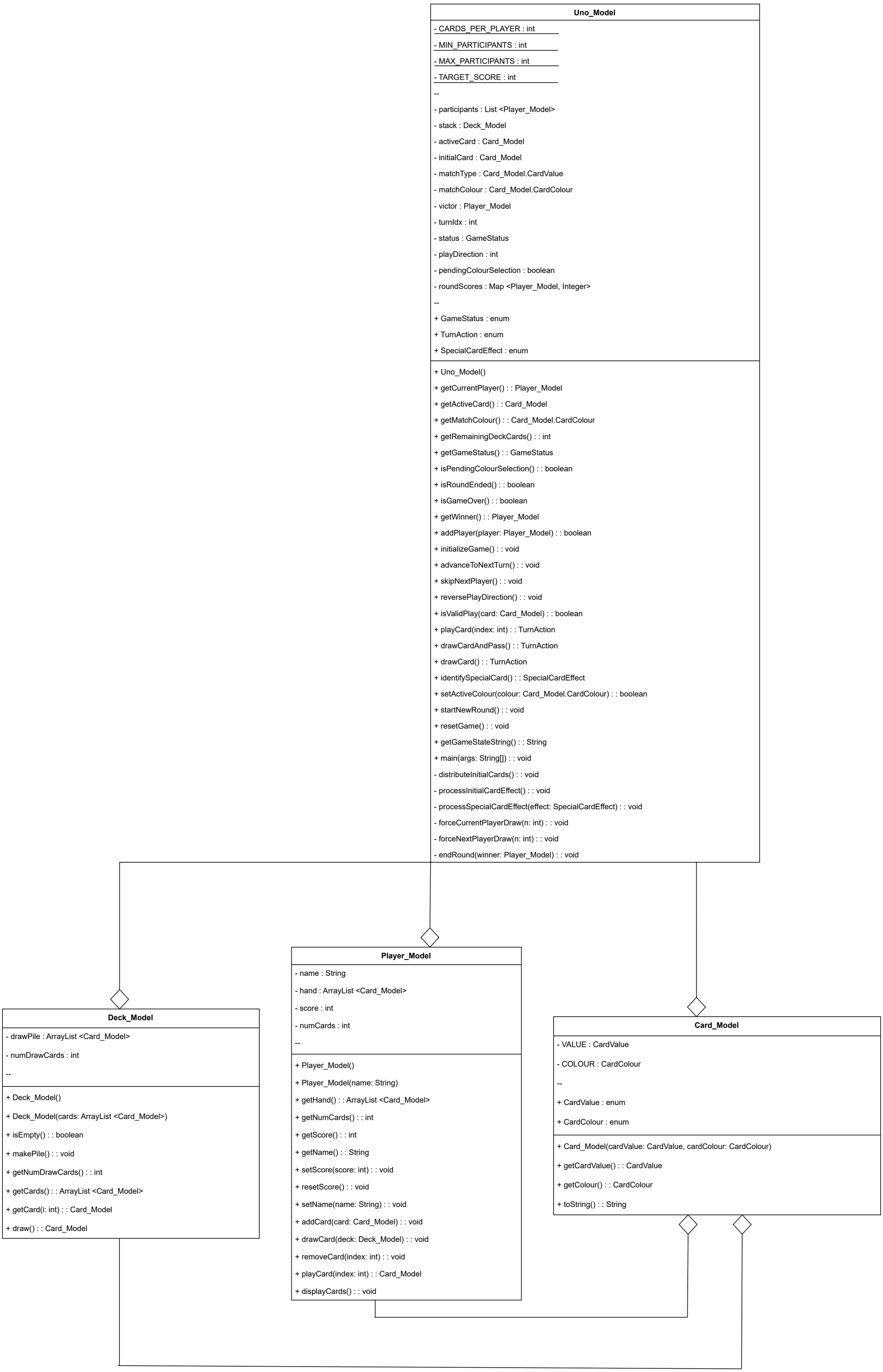
Uno_Model
- CARDS_PER_PLAYER : int
- MIN_PARTICIPANTS : int
- MAX_PARTICIPANTS : int
- TARGET_SCORE : int
--
- participants : List <Player_Model>
- stack : Deck_Model
- activeCard : Card_Model
- initialCard : Card_Model
- matchType : Card_Model.CardValue
- matchColour : Card_Model.CardColour
- victor : Player_Model
- turnIdx : int
- status : GameStatus
- playDirection : int
- pendingColourSelection : boolean
- roundScores : Map <Player_Model, Integer>
--
+ GameStatus : enum
+ TurnAction : enum
+ SpecialCardEffect : enum
+ Uno_Model()
+ getCurrentPlayer() : : Player_Model
+ getActiveCard() : : Card_Model
+ getMatchColour() : : Card_Model.CardColour
+ getRemainingDeckCards() : : int
+ getGameStatus() : : GameStatus
+ isPendingColourSelection() : : boolean
+ isRoundEnded() : : boolean
+ isGameOver() : : boolean
+ getWinner() : : Player_Model
+ addPlayer(player: Player_Model) : : boolean
+ initializeGame() : : void
+ advanceToNextTurn() : : void
+ skipNextPlayer() : : void
+ reversePlayDirection() : : void
+ isValidPlay(card: Card_Model) : : boolean
+ playCard(index: int) : : TurnAction
+ drawCardAndPass() : : TurnAction
+ drawCard() : : TurnAction
+ identifySpecialCard() : : SpecialCardEffect
+ setActiveColour(colour: Card_Model.CardColour) : : boolean
+ startNewRound() : : void
+ resetGame() : : void
+ getGameStateString() : : String
+ main(args: String[]) : : void
- distributeInitialCards() : : void
- processInitialCardEffect() : : void
- processSpecialCardEffect(effect: SpecialCardEffect) : : void
- forceCurrentPlayerDraw(n: int) : : void
- forceNextPlayerDraw(n: int) : : void
- endRound(winner: Player_Model) : : void
Deck_Model
- drawPile : ArrayList <Card_Model>
- numDrawCards : int
--
+ Deck_Model()
+ Deck_Model(cards: ArrayList <Card_Model>)
+ isEmpty() : : boolean
+ makePile() : : void
+ getNumDrawCards() : : int
+ getCards() : : ArrayList <Card_Model>
+ getCard(i: int) : : Card_Model
+ draw() : : Card_Model
Player_Model
- name : String
- hand : ArrayList <Card_Model>
- score : int
- numCards : int
--
+ Player_Model()
+ Player_Model(name: String)
+ getHand() : : ArrayList <Card_Model>
+ getNumCards() : : int
+ getScore() : : int
+ getName() : : String
+ setScore(score: int) : : void
+ resetScore() : : void
+ setName(name: String) : : void
+ addCard(card: Card_Model) : : void
+ drawCard(deck: Deck_Model) : : void
+ removeCard(index: int) : : void
+ playCard(index: int) : : Card_Model
+ displayCards() : : void
Card_Model
- VALUE : CardValue
- COLOUR : CardColour
--
+ CardValue : enum
+ CardColour : enum
+ Card_Model(cardValue: CardValue, cardColour: CardColour)
+ getCardValue() : : CardValue
+ getColour() : : CardColour
+ toString() : : String