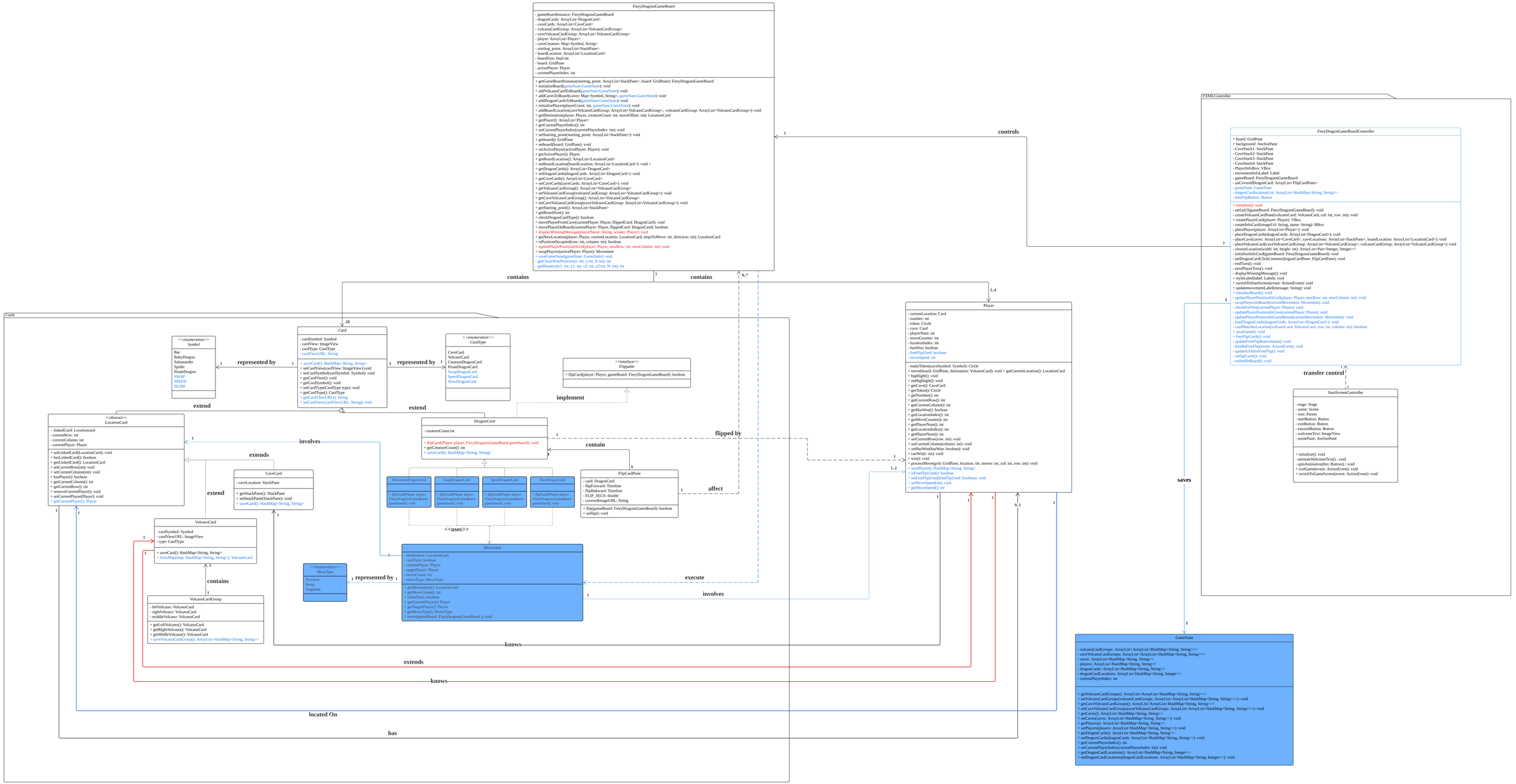
FieryDragonsGameBoard
+ gameBoardInstance: FieryDragonsGameBoard
- dragonCards: ArrayList<DragonCard>
- caveCards: ArrayList<CaveCard>
- volcanoCardGroup: ArrayList<VolcanoCardGroup>
- caveVolcanoCardGroup: ArrayList<VolcanoCardGroup>
- player: ArrayList<Player>
- caveCreature: Map<Symbol, String>
- starting_point: ArrayList<StackPane>
- boardLocation: ArrayList<LocationCard>
- boardSize: final int
- board: GridPane
- activePlayer: Player
- currentPlayerIndex: int
+ getGameBoardInstance(starting_point: ArrayList<StackPane>, board: GridPane): FieryDragonsGameBoard
+ initializeBoard(gameState:GameState): void
+ addVolcanoCardToBoard(gameState:GameState): void
+ addCavesToBoard(caves: Map<Symbol, String>, gameState:GameState): void
+ addDragonCardsToBoard(gameState:GameState): void
+ initializePlayer(playerCount: int, gameState:GameState): void
+ addBoardLocation(caveVolcanoCardGroup: ArrayList<VolcanoCardGroup>, volcanoCardGroup: ArrayList<VolcanoCardGroup>): void
+ getDestination(player: Player, creatureCount: int, moveOffset: int): LocationCard
+ getPlayer(): ArrayList<Player>
+ getCurrentPlayerIndex(): int
+ setCurrentPlayerIndex(currentPlayerIndex: int): void
+ setStarting_point(starting_point: ArrayList<StackPane>): void
+ getboard(): GridPane
+ setboard(board: GridPane): void
+ setActivePlayer(activePlayer: Player): void
+ getActivePlayer(): Player
+ getBoardLocation(): ArrayList<LocationCard>
+ setBoardLocation(boardLocation: ArrayList<LocationCard>): void +
+ getDragonCards(): ArrayList<DragonCard>
+ setDragonCards(dragonCards: ArrayList<DragonCard>): void
+ getCaveCards(): ArrayList<CaveCard>
+ setCaveCards(caveCards: ArrayList<CaveCard>): void
+ getVolcanoCardGroup(): ArrayList<VolcanoCardGroup>
+ setVolcanoCardGroup(volcanoCardGroup: ArrayList<VolcanoCardGroup>): void
+ getCaveVolcanoCardGroup(): ArrayList<VolcanoCardGroup>
+ setCaveVolcanoCardGroup(caveVolcanoCardGroup: ArrayList<VolcanoCardGroup>): void
+ getStarting_point(): ArrayList<StackPane>
+ getBoardSize(): int
+ checkDragonCardType(): boolean
+ movePlayerFromCave(currentPlayer: Player, flippedCard: DragonCard): void
+ movePlayerOnBoard(currentPlayer: Player, flippedCard: DragonCard): boolean
+ displayWinningMessage(playerName: String, winner: Player): void
+ getNewLocation(player: Player, currentLocation: LocationCard, stepsToMove: int, direction: int): LocationCard
+ isPositionOccupied(row: int, column: int): boolean
+ updatePlayerPositionInGrid(player: Player, newRow: int, newColumn: int): void
+ swapPlayers(activePlayer: Player): Movement
+ saveGameState(gameState: GameState): void
- getClockWisePosition(x: int, y:int, N :int): int
- getDistance(x1: int, y1: int, x2: int, y2:int, N: int): int
controls
contains
contains
Cards
<<enumeration>>
Symbol
Bat
BabyDragon
Salamander
Spider
PirateDragon
SWAP
SPEED
SLOW
represented by
Card
- cardSymbol: Symbol
- cardView: ImageView
- cardType: CardType
- cardViewURL: String
+ saveCard(): HashMap<String, String>
+ setCardView(cardView: ImageView):void
+ setCardSymbol(cardSymbol: Symbol): void
+ getCardView(): void
+ getCardSymbol(): void
+ setCardTypeCardType type): void
+ getCardType(): CardType
+ getCardViewURL(): String
+ setCardView(cardViewURL: String): void
represented by
<<enumeration>>
CardType
CaveCard
VolcanoCard
CreatureDragonCard
PirateDragonCard
SwapDragonCard
SpeedDragonCard
SlowDragonCard
<<interface>>
Flippable
+ flipCard(player: Player, gameBoard: FieryDragonGameBoard): boolean
implement
extend
extend
<<abstract>>
LocationCard
- linkedCard: Locationcard
- currentRow: int
- currentColumn: int
- currentPlayer: Player
+ setLinkedCard(LocationCard): void
+ hasLinkedCard(): boolean
+ getLinkedCard(): LocationCard
+ setCurrentRow(int): void
+ setCurrentColumn(int): void
+ hasPlayer(): boolean
+ getCurrentColumn(): int
+ getCurrentRow(): int
+ removeCurrentPlayer(): void
+ setCurrentPlayer(Player): void
+ getCurrentPlayer(): Player
involves
extends
extend
DragonCard
- creatureCount:int
+ flipCard(Player player, FieryDragonsGameBoard gameboard): void
+ getCreatureCount(): int
+ saveCard(): HashMap<String, String>
contain
flipped by
CaveCard
- caveLocation: StackPane
+ getStackPane(): StackPane
+ setStackPane(StackPane): void
+ saveCard(): HashMap<String, String>
MovementDragonCard
+ flipCard(Player player, FieryDragonsGameBoard gameboard): void
SwapDragonCard
+ flipCard(Player player, FieryDragonsGameBoard gameboard): void
SpeedDragonCard
+ flipCard(Player player, FieryDragonsGameBoard gameboard): void
SlowDragonCard
+ flipCard(Player player, FieryDragonsGameBoard gameboard): void
FlipCardPane
- card: DragonCard
- flipForward: Timeline
- flipBakward: Timeline
- FLIP_SECS: double
- coveredImageURL: String
+ flip(gameBoard: FieryDragonsGameBoard): boolean
+ unflip(): void
affect
<<uses>>
VolcanoCard
- cardSymbol: Symbol
- cardViewURL: ImageView
- type: CardType
+ saveCard(): HashMap<String, String>
+ fromMap(map: HashMap<String, String>): VolcanoCard
contains
VolcanoCardGroup
- leftVolcano: VolcanoCard
- rightVolcano: VolcanoCard
- middleVolcano: VolcanoCard
+ getLeftVolcano(): VolcanoCard
+ getRightVolcano(): VolcanoCard
+ getMidlleVolcano(): VolcanoCard
+ saveVolcanoCardGroup(): ArrayList<HashMap<String, String>>
<<enumeration>>
MoveType
Traverse
Swap
Augment
represented by
Movement
- destination: LocationCard
- endTurn: boolean
- currentPlayer: Player
- targetPlayer: Player
- moveCount: int
- moveType: MoveType
+ getDestination(): LocationCard
+ getMoveCount(): int
+ isEndTurn: boolean
+ getCurrentPlayer(): Player
+ getTargetPlayer(): Player
+ getMoveType[]: MoveType
+ move(gameBoard: FieryDragonsGameBoard): void
execute
involves
knows
extends
knows
located On
has
Player
- currentLocation: Card
- number: int
- token: Circle
- cave: Card
- playerNum: int
- moveCounter: int
- locationIndex: int
- hasWon: boolean
- freeFlipUsed: boolean
- moveSpeed: int
- makeToken(caveSymbol: Symbol): Circle
+ moveBoard: GridPane, destination: VolcanoCard): void + getCurrentLocation(): LocationCard
+ highlight(): void
+ unHighlight(): void
+ getCave(): CaveCard
+ getToken(): Circle
+ getNumber(): int
+ getCurrentRow(): int
+ getCurrentColumn(): int
+ getHasWon(): boolean
+ getLocationIndex(): int
+ getMoveCounter(): int
+ getPlayerNum(): int
+ getLocationIndex(): int
+ getPlayerNum(): int
+ setCurrentRow(row: int): void
+ setCurrentColumn(column: int): void
+ setHasWon(hasWon: boolean): void
+ canWin(): int: void
+ win(): void
+ processMove(grid: GridPane, location: int. moves: int, col: int, row: int): void
+ savePlayer(): HashMap<String, String>
+ isFreeFlipUsed(): boolean
+ setFreeFlipUsed(freeFlipUsed: boolean): void
+ setMoveSpeed(int): void
+ getMoveSpeed(): int
saves
FXMLController
FieryDragonGameBoardController
+ board: GridPane
+ background: AnchorPane
- CaveStack1: StackPane
- CaveStack2: StackPane
- CaveStack3: StackPane
- CaveStack4: StackPane
- PlayerInfoBox: VBox
- movementInfoLabel: Label
- gameBoard: FieryDragonsGameBoard
- unCoveredDragonCard: ArrayList<FlipCardPane>
- gameState: GameState
- dragonCardlocationList: ArrayList<HashMap<String, String>>
- freeFlipButton: Button
+ initialize(): void
- setUpUI(gameBoard: FieryDragonsGameBoard): void
- createVolcanoCardPane(volcanoCard: VolcanoCard, col: int, row: int): void
+ createPlayerCard(player: Player): VBox
- createInfoCard(imageUrl: String, name: String): HBox
- placePlayer(player: ArrayList<Player>): void
- placeDragonCards(dragonCards: ArrayList<DragonCard>): void
- placeCave(caves: ArrayList<CaveCard> , caveLocations: ArrayList<StackPane>, boardLocation: ArrayList<LocationCard>): void
- placeVolcanoCard(caveVolcanoCardGroup: ArrayList<VolcanoCardGroup>, volcanoCardGroup: ArrayList<VolcanoCardGroup>): void
- chooseLocation(width: int, height: int): ArrayList<Pair<Integer, Integer>>
- initializeInfoCard(gameBoard: FieryDragonsGameBoard): void
- setDragonCardClickListener(dragonCardPane: FlipCardPane): void
- endTurn(): void
- nextPlayerTurn(): void
- displayWinningMessage(): void
+ styleLabel(label: Label): void
+ switchToStartScreen(event: ActionEvent): void
+ updatemovementLabel(message: String): void
+ initializeBoard(): void
- updatePlayerPositionInGrid(player: Player, newRow: int, newColumn: int): void
- swapPlayersInBoard(currentMovement: Movement): void
- checkForWin(currentPlayer: Player): void
- updatePlayerPositionInCave(currentPlayer: Player): void
- updatePlayerPositionInGameBoard(currentMovement: Movement): void
- loadDragonCards(dragonCards: ArrayList<DragonCard>): void
- cardMatchesLocation(volcanoCard: VolcanoCard, row: int, column: int): boolean
+ saveGame(): void
+ freeFlipCards(): void
- updateFreeFlipButtonState(): void
- handleFreeFlipEvent(event: ActionEvent): void
- updateUIAfterFreeFlip(): void
- unflipCards(): void
- reshuffleBoard(): void
transfer control
StartScreenController
- stage: Stage
- scene: Scene
- root: Parent
- startButton: Button
- exitButton: Button
- tutorialButton: Button
- welcomeText: ImageView
- scenePane: AnchorPane
+ initialize(): void
- animateWelcomeText() : void
- spinAnimation(btn: Button) : void
+ exitGame(event: ActionEvent): void
+ switchToGameScreen(event: ActionEvent): void
GameState
- volcanoCardGroups: ArrayList<ArrayList<HashMap<String, String>>>
- caveVolcanoCardGroups: ArrayList<ArrayList<HashMap<String, String>>>
- caves: ArrayList<HashMap<String, String>>
- players: ArrayList<HashMap<String, String>>
- dragonCards: ArrayList<HashMap<String, String>>
- dragonCardLocations: ArrayList<HashMap<String, Integer>>
- currentPlayerIndex: int
+ getVolcanoCardGroups(): ArrayList<ArrayList<HashMap<String, String>>>
+ setVolcanoCardGroups(volcanoCardGroups: ArrayList<ArrayList<HashMap<String, String>>>): void
+ getCaveVolcanoCardGroups(): ArrayList<ArrayList<HashMap<String, String>>>
+ setCaveVolcanoCardGroups(caveVolcanoCardGroups: ArrayList<ArrayList<HashMap<String, String>>>): void
+ getCaves(): ArrayList<HashMap<String, String>>
+ setCaves(caves: ArrayList<HashMap<String, String>>): void
+ getPlayers(): ArrayList<HashMap<String, String>>
+ setPlayers(players: ArrayList<HashMap<String, String>>): void
+ getDragonCards(): ArrayList<HashMap<String, String>>
+ setDragonCards(dragonCards: ArrayList<HashMap<String, String>>): void
+ getCurrentPlayerIndex(): int
+ setCurrentPlayerIndex(currentPlayerIndex: int): void
+ getDragonCardLocations(): ArrayList<HashMap<String, Integer>>
+ setDragonCardLocations(dragonCardLocations: ArrayList<HashMap<String, Integer>>): void