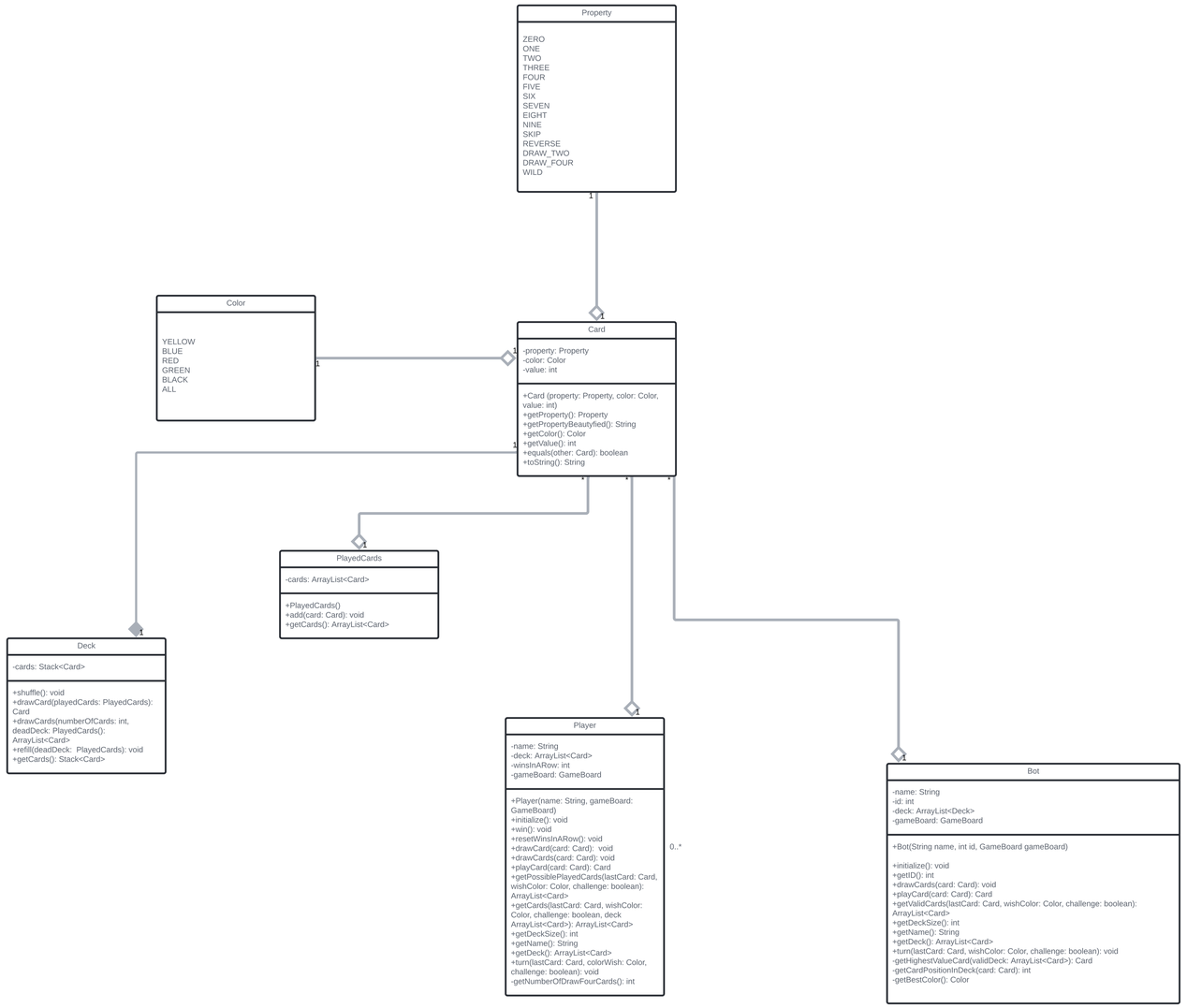
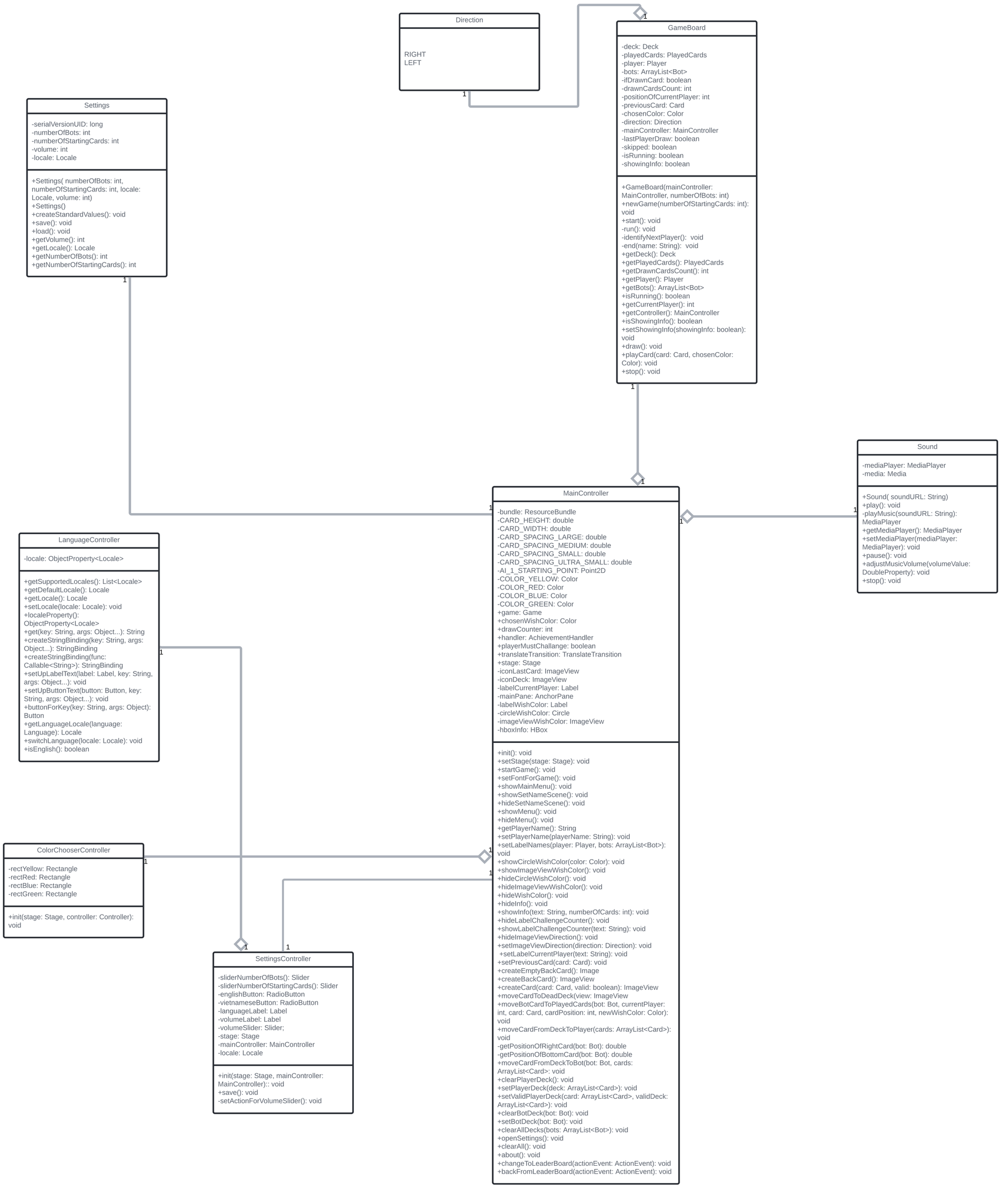
MODEL



Property
ZERO
ONE
TWO
THREE
FOUR
FIVE
SIX
SEVEN
EIGHT
NINE
SKIP
REVERSE
DRAW_TWO
DRAW_FOUR
WILD
Color
YELLOW
BLUE
RED
GREEN
BLACK
ALL
Card
-property: Property
-color: Color
-value: int
+Card (property: Property, color: Color, value: int)
+getProperty(): Property
+getPropertyBeautyfied(): String
+getColor(): Color
+getValue(): int
+equals(other: Card): boolean
+toString(): String
PlayedCards
-cards: ArrayList<Card>
+PlayedCards()
+add(card: Card): void
+getCards(): ArrayList<Card>
Deck
-cards: Stack<Card>
+shuffle(): void
+drawCard(playedCards: PlayedCards): Card
+drawCards(numberOfCards: int, deadDeck: PlayedCards(): ArrayList<Card>
+refill(deadDeck: PlayedCards): void
+getCards(): Stack<Card>
Player
-name: String
-deck: ArrayList<Card>
-winsInARow: int
-gameBoard: GameBoard
+Player(name: String, gameBoard: GameBoard)
+initialize(): void
+win(): void
+resetWinsInARow(): void
+drawCard(card: Card): void
+drawCards(card: Card): void
+playCard(card: Card): Card
+getPossiblePlayedCards(lastCard: Card, wishColor: Color, challenge: boolean): ArrayList<Card>
+getCards(lastCard: Card, wishColor: Color, challenge: boolean, deck ArrayList<Card>): ArrayList<Card>
+getDeckSize(): int
+getName(): String
+getDeck(): ArrayList<Card>
+turn(lastCard: Card, colorWish: Color, challenge: boolean): void
-getNumberOfDrawFourCards(): int
Bot
-name: String
-id: int
-deck: ArrayList<Deck>
-gameBoard: GameBoard
+Bot(String name, int id, GameBoard gameBoard)
+initialize(): void
+getID(): int
+drawCards(card: Card): void
+playCard(card: Card): Card
+getValidCards(lastCard: Card, wishColor: Color, challenge: boolean): ArrayList<Card>
+getDeckSize(): int
+getName(): String
+getDeck(): ArrayList<Card>
+turn(lastCard: Card, wishColor: Color, challenge: boolean): void
-getHighestValueCard(validDeck: ArrayList<Card>): Card
-getCardPositionInDeck(card: Card): int
-getBestColor(): Color
1
1
1
1
1
1
1
1
1
1
1
1
0..*

Controller


Direction
RIGHT
LEFT
1
1
GameBoard
-deck: Deck
-playedCards: PlayedCards
-player: Player
-bots: ArrayList<Bot>
-ifDrawnCard: boolean
-drawnCardsCount: int
-positionOfCurrentPlayer: int
-previousCard: Card
-chosenColor: Color
-direction: Direction
-mainController: MainController
-lastPlayerDraw: boolean
-skipped: boolean
-isRunning: boolean
-showingInfo: boolean
+GameBoard(mainController: MainController, numberOfBots: int)
+newGame(numberOfStartingCards: int): void
+start(): void
-run(): void
-identifyNextPlayer(): void
-end(name: String): void
+getDeck(): Deck
+getPlayedCards(): PlayedCards
+getDrawnCardsCount(): int
+getPlayer(): Player
+getBots(): ArrayList<Bot>
+isRunning(): boolean
+getCurrentPlayer(): int
+getController(): MainController
+isShowingInfo(): boolean
+setShowingInfo(showingInfo: boolean): void
+draw(): void
+playCard(card: Card, chosenColor: Color): void
+stop(): void
Settings
-serialVersionUID: long
-numberOfBots: int
-numberOfStartingCards: int
-volume: int
-locale: Locale
+Settings(numberOfBots: int, numberOfStartingCards: int, locale: Locale, volume: int)
+Settings()
+createStandardValues(): void
+save(): void
+load(): void
+getVolume(): int
+getLocale(): Locale
+getNumberOfBots(): int
+getNumberOfStartingCards(): int
1
1
Sound
-mediaPlayer: MediaPlayer
-media: Media
+Sound(soundURL: String)
+play(): void
-playMusic(soundURL: String): MediaPlayer
+getMediaPlayer(): MediaPlayer
+setMediaPlayer(mediaPlayer: MediaPlayer): void
+pause(): void
+adjustMusicVolume(volumeValue: DoubleProperty): void
+stop(): void
1
1
MainController
-bundle: ResourceBundle
-CARD_HEIGHT: double
-CARD_WIDTH: double
-CARD_SPACING_LARGE: double
-CARD_SPACING_MEDIUM: double
-CARD_SPACING_SMALL: double
-CARD_SPACING_ULTRA_SMALL: double
-AI_1_STARTING_POINT: Point2D
-COLOR_YELLOW: Color
-COLOR_RED: Color
-COLOR_BLUE: Color
-COLOR_GREEN: Color
+game: Game
+chosenWishColor: Color
+drawCounter: int
+handler: AchievementHandler
+playerMustChallange: boolean
+translateTransition: TranslateTransition
+stage: Stage
-iconLastCard: ImageView
-iconDeck: ImageView
-labelCurrentPlayer: Label
-mainPane: AnchorPane
-labelWishColor: Label
-circleWishColor: Circle
-imageViewWishColor: ImageView
-hboxInfo: HBox
+init(): void
+setStage(stage: Stage): void
+startGame(): void
+setFontForGame(): void
+showMainMenu(): void
+showSetNameScene(): void
+hideSetNameScene(): void
+showMenu(): void
+hideMenu(): void
+getPlayerName(): String
+setPlayerName(playerName: String): void
+setLabelNames(player: Player, bots: ArrayList<Bot>): void
+showCircleWishColor(color: Color): void
+showImageViewWishColor(): void
+hideCircleWishColor(): void
+hideImageViewWishColor(): void
+hideWishColor(): void
+hideInfo(): void
+showInfo(text: String, numberOfCards: int): void
+hideLabelChallengeCounter(): void
+showLabelChallengeCounter(text: String): void
+hideImageViewDirection(): void
+setImageViewDirection(direction: Direction): void
+setLabelCurrentPlayer(text: String): void
+setPreviousCard(card: Card): void
+createEmptyBackCard(): Image
+createBackCard(): ImageView
+createCard(card: Card, valid: boolean): ImageView
+moveCardToDeadDeck(view: ImageView
+moveBotCardToPlayedCards(bot: Bot, currentPlayer: int, card: Card, cardPosition: int, newWishColor: Color): void
+moveCardFromDeckToPlayer(cards: ArrayList<Card>): void
-getPositionOfRightCard(bot: Bot): double
-getPositionOfBottomCard(bot: Bot): double
+moveCardFromDeckToBot(bot: Bot, cards: ArrayList<Card>: void
+clearPlayerDeck(): void
+setPlayerDeck(deck: ArrayList<Card>): void
+setValidPlayerDeck(card: ArrayList<Card>, validDeck: ArrayList<Card>): void
+clearBotDeck(bot: Bot): void
+setBotDeck(bot: Bot): void
+clearAllDecks(bots: ArrayList<Bot>): void
+openSettings(): void
+clearAll(): void
+about(): void
+changeToLeaderBoard(actionEvent: ActionEvent): void
+backFromLeaderBoard(actionEvent: ActionEvent): void
1
1
1
1
LanguageController
-locale: ObjectProperty<Locale>
+getSupportedLocales(): List<Locale>
+getDefaultLocale(): Locale
+getLocale(): Locale
+setLocale(locale: Locale): void
+localeProperty(): ObjectProperty<Locale>
+get(key: String, args: Object...): String
+createStringBinding(key: String, args: Object...): StringBinding
+createStringBinding(func: Callable<String>): StringBinding
+setUpLabelText(label: Label, key: String, args: Object...): void
+setUpButtonText(button: Button, key: String, args: Object...): void
+buttonForKey(key: String, args: Object): Button
+getLanguageLocale(language: Language): Locale
+switchLanguage(locale: Locale): void
+isEnglish(): boolean
1
1
ColorChooserController
-rectYellow: Rectangle
-rectRed: Rectangle
-rectBlue: Rectangle
-rectGreen: Rectangle
+init(stage: Stage, controller: Controller): void
1
1
SettingsController
-sliderNumberOfBots(): Slider
-sliderNumberOfStartingCards(): Slider
-englishButton: RadioButton
-vietnameseButton: RadioButton
-languageLabel: Label
-volumeLabel: Label
-volumeSlider: Slider;
-stage: Stage
-mainController: MainController
-locale: Locale
+init(stage: Stage, mainController: MainController):: void
+save(): void
-setActionForVolumeSlider(): void
1
1