

## Controller

GameController serves 3 purposes:

1. Scripting the Game, by calling the appropriate actions one after the other.
2. Receiving instructions from the Views and making those changes in the Model.
3. Notifying the Views whenever a change to the Model is made.

Since a GameController contains the whole Model inside of it, a GameController corresponds to a whole match.

Name for the Game/Match:  
the Player who creates the  
Game gives it a name (or uses  
the default one), and other  
Players see that name in the  
"Game List Screen".  
This is used for the "Multiple  
Games" FA

Model

View

<<enumeration>>  
GameStatus

WAITING\_FOR\_PLAYERS  
NOT\_IN\_GAME  
CONNECTING  
WAITING\_FOR\_SERVER  
READY  
READY\_TO\_CHOOSE\_TOKEN  
READY\_TO\_CHOOSE\_SECRET\_OBJECTIVE  
READY\_TO\_CHOOSE\_STARTER\_CARD  
READY\_TO\_DRAW\_STARTING\_HAND  
READY\_TO\_PLAY  
PLAYING  
MY\_TURN  
NOT\_MY\_TURN  
LAST\_TURN  
COUNTING\_POINTS  
END\_GAME

1

### GameController

[final] - game: Game

[final] - views:  
CopyOnWriteArrayList<RemoteViewController>

- name: String

[final] - unusedViews:  
CopyOnWriteArrayList<RemoteViewController>

[final] - viewOwners:  
HashMap<RemoteViewController, Integer>

- isStarted: boolean

- numberOfDisconnectedPlayers: int

- currentStatus: GameStatus

[final] - listeners: ArrayList<Listener>

+ setCurrentStatus(GameStatus): void

+ startGame(): void

+ addPlayer(Player): void

+ kickPlayer(Player): bool

+ nextTurn(): void

+ playCard(int, int, int, int): void

+ flipCard(int, int): void

+ drawStartingHand(): void

+ drawCard(Player, PlayingDeck): void

+ grabCard(Player, PlayingDeck, int): void

+ drawSecretObjectives(): void

+ beginStarterCardChoosing(): void

+ beginTokenChoosing(): void

+ addView(RemoteViewController): void

- checkIfGameCanContinue(): void

- setupViews(): void

+ rejoinGame(int): void

- nextStatus(): void

+ disconnectPlayer(int): void

- currentStatus

Checks if all the Players are on the same  
status as the GameController: if so,  
GameController changes its status  
following a script defined in nextStatus(),  
triggering the next action for each Player