Collective data management

In this chapter we will learn:

1) Setting a player’s turn

2) How to end a game, declare a winner, give each player a score and a certain percentage of the pot.

Tutorial introduction

Till now we learned how to handle subjective data, such as what move did a player choose to do, this information was sent using the doStoreState function.  
Every function that handles subjective data will start with do… marking it as an action, unlike the got... which describes callbacks.

In this tutorial we are going to learn about functions that are associated with collective data, this functions will start with a doAll… marking them as actions everyone must agree on, such as whose turn it is, or who won, there can be no disputes in this matters because those are collective facts.

doAll… functions will only be called as a trigger of gotStateChanged, gotMatchStarted, or gotMatchEnded, the later won’t be discussed in these tutorial. The reason we only call doAll… functions as a trigger of the above callbacks is because a doAll… function must be triggered by something all the users got, and not a Timer or a mouse click.

Setting a player’s turn

In the third tutorial we learned how to send data about our move to the entire set of users, in this tutorial we will learn how to set the next players turn.

1. Setting a turn, like the rest of the doAll… functions will only be called as a result of the gotStateChanged, gotMatchStarted or gotMatchOver.
2. All players must call doAllSetTurn with the same values to make sure the decision is unanimous and that no one is cheating.
3. To set a player’s turn all players must call doAllSetTurn(userId:int, milliSecondsInTurn:int)

userId – the userId of the player whose turn it is

milliSecondsInTurn – time to give the player to make his move in milliseconds, before declaring a profit, set -1 for default value.

For more information about doAllSetTurn visit: <http://code.google.com/p/multiplayer-api/wiki/doAllSetTurn>

Ending the game

In our game every time the user calls the makeTurn function, the games logic checks if one of the user ‘s has won, or if the game is tied, in case one of this situations is present the game’s logic calls a GameOverEvent describing if one of the users has won, or if the game was tied.

To declare the game’s result we should end the match:

1. Ending a match, like the rest of the doAll… functions will only be called as a result of the gotStateChanged, gotMatchStarted or gotMatchOver.
2. All players must call doAllEndMatch with the same values to make sure the decision is unanimous and that no one is cheating.
3. To end the game for all or some players, all players must call doAllEndMatch(finishedPlayers:Array)

finishedPlayers – is an Array of PayerMatchOver elements, each representing a player in the game, which we want the game to end for.

To create a PayerMatchOver entry we call the PayerMatchOver.create(playerId:int, score:int, potPercentage:int)  
playerId – the user id of the player this entry refers to.  
score – the score the user got in the game.  
potPercentage – the percentage of the pot the player will get.

In our example the game ends for everyone when one of the players wins, but in your game you can make the game continue, ending the game only for the wining/losing player.

\*for more information about doAllEndMatch visit:

<http://code.google.com/p/multiplayer-api/wiki/doAllEndMatch>

http://code.google.com/p/multiplayer-api/wiki/PlayerMatchOver