Intro:

In this tutorial we are going to learn how to manage collective data.

Clip 1:

Until now you learned only how to call doStoreState to broadcast the user's move. In this chapter you will use two new operations called do**All**SetTurn and do**All**EndMatch. As the name implies, **all** players must collectively call these operations with identical arguments. The reason is obvious: we can’t have two players believing they won, or two players believing it’s their turn.

Functions that must be unanimously called by all players are marked with a do**All** prefix. Such calls can only be made during the processing of these 3 callbacks: gotMatchStarted, gotStateChanged and gotMatchEnded.

Clip 2:

The function doAllSetTurn gets a user id that has the current turn and the time for the user to make his move. If the player doesn’t make his move in given time, he automatically loses; in our class we call this function from the next Turn function.  
The nextTurn function sets the turn, based on the allPlayerIds array, that was received in the gotStateChanged callback, thus making sure all players will set the turn to the same player.

Clip 3:

The next commands we will learn is doAllEndMatch, which gets an array of PlayerMatchOver elements, every PlayerMatchOver class must be created by its static create function.

Each PlayerMatchOver instance will contain a user id of a player, the player's percentage from the pot, and the player's score in the game.

In this example, you can see that the game ends as soon as one of the wining conditions is achieved, because the game over function goes over all the game players and ends the game for them.

Clip 4:

Now let’s visit the emulator and see the changes we’ve made (pause)

Notice that now each player can only play on his turn, and that the game actually ends when a victory or a tie is achieved.