Setting up the environment to build you’re multiplayer game

In this chapter we will:

1) Learn how to build the basic file structure for your game

2) Go over the Ticktacktoe Logic

Explaining the basic file structure for the game

1. Make sure that the come2play\_as3 directory is in the same path in all your tutorial chapters; you should not change the files in this directory.
2. Now go into the Chapter 1 directory you will notice an \*.fla file, this file will be used only as the graphics source; it will not change during the chapters of the tutorial.
3. You will also see a directory named ticktactoeTuturial which will contain our tutorial game files (but represents every multiplayer game you will ever want to make)
4. Notice that the ticktactoeTuturial folder contains the following files: TickTacToeTuturialMain.as, TickTacToeTuturialLogic.as, TickTacToeTuturialGraphic.as and TickTacToeMove.as.

TickTacToeTuturialMain extends ClientGameAPI, the ClientGameAPI will give your game all its server client connectivity, and therefore all calls and callbacks will be made only in TickTacToeTuturialMain.

TickTacToeTuturialLogic

This class contains all the game’s logic, and an instance of the game’s graphics.

No graphic action should be made in this class, and no logical calculation should be made outside of this class, working this way will make debugging your multiplayer game much easier because you will be able to divide each error into one of three categories: logic based errors, graphic based errors and server input based errors.

Notice that user actions don’t have a direct effect on the game’s logic, every user action is first sent to the main class and only then the action is sent back into the game’s logic, in the next tutorial we will look into why we do it in further detail.

Our game’s logic class contains 2 public functions for now

1. startNewGame – which gets the number of players, and starts a new game.
2. makeTurn – which gets a TickTacToeMove instance representing a game move and commits it to the board