The goal of this assignment is to create a simple game using an OrientationSensor and to store persistent information on the devices using a TinyDB.

1. [10] Create a new App Inventor project called Assignment3. Set the Screen’s ScreenOrientation to Portrait, which will lock the screen’s orientation so that it doesn’t change while tilting the device. Add a Canvas named GameCanvas to your project with dimensions 300x300 and a light gray background color. Also add an OrientationSensor, a Clock (named TimeClock with a 1000-millisecond interval), and a TinyDB to the project.

2. [10] Add twenty ImageSprites to your Canvas to act as obstacles. You can use square.jpg (linked on the course announcements page) or an image of your choosing, but be aware that images with rectangular content will provide the best collision results (we’ll discuss in class).

3. [10] Add a large (e.g., radius 50) Ball named GoalBall to one of the corners of the Canvas and a small (e.g., radius 5) Ball named PlayerBall to another corner of the Canvas. Color the Balls to your liking. It is important that PlayerBall is added to the Canvas after the GoalBall and the ImageSprite obstacles so that it is drawn on top of these components.

4. [10] Add a HorizontalArrangements containing four labels: a “Score: “ name label, a score value label (initially 1000), a “Hi: “ name label and a hi value label (initially 0). Also add a Button named ResetButton with text “Reset”. Your project should now look something like:
5. [10] In the Blocks Editor, add a variable named “enabled” which is initially set to true. This variable will be used to keep track of whether or not the game is currently active. Add an OrientationSensor.OrientationChanged block which first checks to see if the enabled variable is true. If so, set the heading of PlayerBall to the angle of the OrientationSensor and the speed of the PlayerBall to the magnitude of the OrientationSensor multiplied by 100. If the enabled variable was set to false, simply set the speed of PlayerBall to 0. Also add a PlayerBall.EdgeReached block which causes the Ball to bounce off of the appropriate edge (i.e., using the PlayerBall.Bounce() procedure). You should now be able to move the PlayerBall about the Canvas by manipulating the orientation of the phone, but collisions with the obstacles or GoalBall will not yet be handled.

6. [10] Add a procedure named “decreaseScore“ which takes a single argument named “amount”. The procedure should reduce the score displayed in the score value label by the specified amount. Then, if the new score is < 0, the player has lost the game; set score value label to 0 and the enabled variable set to false.

7. [10] Add a TimerClock.Timer block which, if the enabled variable is true, calls the decreaseScore procedure with an argument of 5 (i.e., decreaseScore(5)). Also add a ResetButton.Click block which sets the score value text to 1000, the PlayerBall’s speed to 0, the PlayerBall X and Y to their defaults, and the enabled variable to true.

8. [10] Add a Screen1.Initialize block which, if the “hi” entry in the TinyDB has length > 0, sets the hi value label to the “hi” entry in the TinyDB.

9. [20] Add a PlayerBall.CollidedWith ball which first checks to see if the collision was with the GoalBall.

   - If the collision was with the GoalBall, the player has won the game; set the enabled variable to false. Then check to see if the text in the score value label is greater than the text in the hi value label. If so, set the text of the hi value label to the text of the score value label and store the text of the score value label in the TinyDB as “hi”.
   - If the collision was not with the GoalBall, call decreaseScore(10) and set the heading of the PlayerBall to the current heading of the PlayerBall minus 180 (we’ll discuss in class).

10. [+20] For extra credit, add a dynamic obstacle. The obstacle should be a Ball or ImageSprite which changes its speed and direction randomly at a regular interval. The dynamic obstacle should handle collisions with the static obstacles with a behavior similar to that of the PlayerBall and have a score penalty similar to that of the static obstacles. Tune the behavior of the dynamic obstacle to make the game more challenging but still fun. Ideally, this obstacle should be added prior to adding the PlayerBall so that the PlayerBall is drawn on top of the
obstacle, but be aware that deleting your PlayerBall will remove all of the associated blocks WITH NO UNDO!

Download your Assignment3 project directory to your computer (i.e. My Projects -> More Actions -> Download Source) and then upload to Blackboard (i.e., Course Documents -> Assignment 3).