Lab 5: Touch events and sensor basics

Design and Development of Mobile Applications
E. De Coninck, S. Leroux, P. Simoens
2015-2016

In this lab session you will create a Pong application controlled by touch events and motion events detected via various sensors. The Pong game logic and GUI is provided and can be downloaded from Minerva. It is encouraged to experiment with the different sensor types to acquire the necessary knowledge on their performance.

1 The Pong application

Start from the code provided on Minerva. Download the source code and **import** this project into Android Studio. In the source code you will find TODO's for each task of this exercise. Make each task in sequence and read them carefully. Figure 1 shows the application we will create. The game logic and the GUI are provided in the start code.

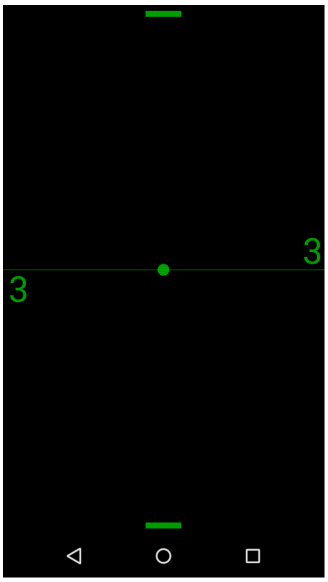

Figure 1: `SurfaceView` of the Pong application.

The `PongActivity` class is the launcher `Activity` of this application and of which the layout is provided in the (currently empty) `activity_pong.xml` file. The `GameView` class is the `SurfaceView` on which the game is drawn. A `SurfaceView` is a view component that provides a dedicated drawing surface. Just like any other view component, it can be embedded inside the view hierarchy.

- Add the `GameView` to the `activity_pong.xml` layout file. Make this new view fill the parent view group and keep the screen always on. When running your application you should see the same layout as

Figure 1.

The `GameState` class is the state of the Pong game. The methods you need to use are listed below:

- `play()`: play a new round or start a new game when a player has lost all his lives. The ball blinks if the round has not started.
- `movePaddleTo(int x)`: moves the bottom paddle to position x. The paddle moves with a certain speed and always moves in the direction of position x.
- `movePaddle(int dx)`: moves the bottom paddle dx pixels to the right if the value is positive and to the left if the value is negative. The same rule applies for the speed of the paddle.

2 Game control with touch events

At first we will implement the game controls with touch events. On touching the screen the game needs to start and the paddle moves to the position of the touch event. Let the paddle follow the motion of the touch event.

- Override OnTouchEvent in the `GameView`. This method will be called automatically when a user touches or makes a gesture on the view.

3 Game control with gyroscope and virtual sensors

- To control the bottom paddle with the gyroscope we need to tell the application we are interested in this sensor and configure the listener. Get a reference to the `SensorManger` in the `PongActivity`'s `onCreate` method and write all sensors to Logcat to see your device's capabilities.
- Get a reference to the `Sensor` you want to test and override the `onResume` and `onPause` methods in `PongActivity` to register and unregister the sensor listener with the `SensorManager`. The sensor listener should be implemented in the `GameView` class by implementing the `SensorEventListener` interface. The `SensorEventListener` interface has two methods but we will only use `onSensorChanged()`.
- Implement the game control with the GYROSCOPE. Hint: you can use the `android.graphics.Matrix` class for matrix multiplication.
- Change the orientation to landscape by editing the manifest file and check the functionality of the application.

Question: Why does the functionality change when the orientation is changed? How can we fix this?

- Implement the game control with TYPE_GAME_ROTATION_VECTOR and TYPE_ROTATION_VECTOR.