Robin Karmakar

Application 3: Gestures

Janne Lindquivst

Building this application was far simpler than the other applications that came before. Its simplicity arose from the tools that were available to the developer. Also it mentions what methods will be needed in the assignment description. For instance it says to override the onDraw method of the extended View class. These hints went a long way in helping develop the code. The main activity or rather for my application MultiTouchActivity which extends the Activity class simply calls setContentView() passing in the an instance of another class called MultiTouchView to generate the screen to play on. The MultiTouchView, which extends the View class, makes sure the app acts in real time. This means that if there is pressure on the screen then a text displaying the dot’s ID, x and y locations are displayed, and once there is no more pressure the dot is removed. Same concept applies for multiple locations of applied pressures creating multiple dots. The application implements all 100 points.

For the first 60 points, the app can display the touch position by a colored circle. This is accomplished through the drawCircle method in the overridden onDraw method. The 70 points are accomplished by the drawText method in the same onDraw method. The next 80 points is accomplished by implement the case of MotionEven.ACTION\_MOVE in the overridden onTouchEvent class. In this case one would update the location of the dot. The last 20 points was a little more difficult to accomplish, because it required some thinking to give birth to multiple dots and orchestrate their movements. Luckily Java has a PointF class that can account for two floating-point attributes, perfect for a location of a dot on 2D plain. We store this in a SparseArray too so that it is efficient when updates are occurring on the canvas. MotionEvent.ACTION\_POINTER\_DOWN allows for creating at most 5 dots and MotionEvent.ACTION\_POINTER\_UP removes the dot. One thing to note is that while there are multiple dots on the screen and one of the fingers is released to remove a dot, another dot obtains the color of the updated dot and the text accordingly gets updated with the updated dot. So information remains synchronized.