**Mobile Programming**

**Assignment #4**

**Working with Canvas**

**Song noo ri**

**HW #4-A : Bbok Bbok (Bubble)**

**Activity Java code>> (MainActivity.java)**

**package** com.example.hw4\_a;

**import** android.app.Activity;

**import** android.os.Bundle;

**public** **class** MainActivity **extends** Activity {

@Override

**protected** **void** onCreate(Bundle savedInstanceState) {

**super**.onCreate(savedInstanceState);

setContentView(R.layout.*activity\_main*);

}

}

**View Java code>> (BbokBbok.java)**

**package** com.example.hw4\_a;

**import** android.content.Context;

**import** android.content.res.Resources;

**import** android.graphics.Bitmap;

**import** android.graphics.BitmapFactory;

**import** android.graphics.Canvas;

**import** android.util.AttributeSet;

**import** android.view.Display;

**import** android.view.MotionEvent;

**import** android.view.View;

**import** android.view.WindowManager;

**import** android.widget.Toast;

**public** **class** BbokBbok **extends** View {

**private** **int** c\_x=0, c\_y=0;

**private** **boolean**[][] check;

**private** Bitmap on, pop;

**private** **int** screenWidth, screenHeight;

**private** **boolean** finish;

**public** BbokBbok(Context c) {

**super**(c);

init();

Display display = ((WindowManager)c.getSystemService(Context.*WINDOW\_SERVICE*)).getDefaultDisplay();

screenWidth = display.getWidth();

screenHeight = display.getHeight();

}

**public** BbokBbok(Context c, AttributeSet a) {

**super**(c, a);

init();

Display display = ((WindowManager)c.getSystemService(Context.*WINDOW\_SERVICE*)).getDefaultDisplay();

screenWidth = display.getWidth();

screenHeight = display.getHeight();

}

**public** **void** init()

{

check = **new** **boolean**[6][10];

finish = **false**;

**for**(**int** i = 0; i<6; i++)

**for**(**int** j = 0; j<10; j++)

check[i][j] = **true**;

Resources res = getResources();

on = BitmapFactory.*decodeResource*(res, R.drawable.*on*);

pop = BitmapFactory.*decodeResource*(res, R.drawable.*pop*);

}

**protected** **void** onDraw(Canvas canvas)

{

**if**(finish)

Toast.*makeText*(getContext(), "Finish!!", Toast.*LENGTH\_LONG*).show();

**for**(**int** x=0; x<6; x++)

{

**for**(**int** y=0; y<10; y++)

{

**if**((c\_x < screenWidth/6 + x\*on.getWidth() && c\_x > x\*on.getWidth())

&& (c\_y < screenHeight/10 + y\*on.getHeight() && c\_y > y\*on.getHeight()))

{

check[x][y] = **false**;

canvas.drawBitmap(pop, x\*on.getWidth(), y\*on.getHeight(), **null**);

canvas.save();

}

**else**

{

**if**(check[x][y])

canvas.drawBitmap(on, x\*on.getWidth(), y\*on.getHeight(), **null**);

**else**

canvas.drawBitmap(pop, x\*on.getWidth(), y\*on.getHeight(), **null**);

}

}

}

}

**public** **boolean** onTouchEvent(MotionEvent event)

{

c\_x = (**int**)event.getX();

c\_y = (**int**)event.getY();

**if**(event.getAction() == MotionEvent.*ACTION\_UP*)

{

**if**(checkEnd())

finish = **true**;

invalidate();

}

**return** **true**;

}

**private** **boolean** checkEnd()

{

**for**(**int** i=0; i<6; i++)

{

**for**(**int** j=0; j<10; j++)

{

**if**(check[i][j])

**return** **false**;

}

}

**return** **true**;

}

}

**Layout file >> (activity\_main.xml)**

<FrameLayout xmlns:android=*"http://schemas.android.com/apk/res/android"*

xmlns:tools=*"http://schemas.android.com/tools"*

android:id=*"@+id/container"*

android:layout\_width=*"match\_parent"*

android:layout\_height=*"match\_parent"*

tools:context=*"com.example.hw4\_a.MainActivity"*

tools:ignore=*"MergeRootFrame"* >

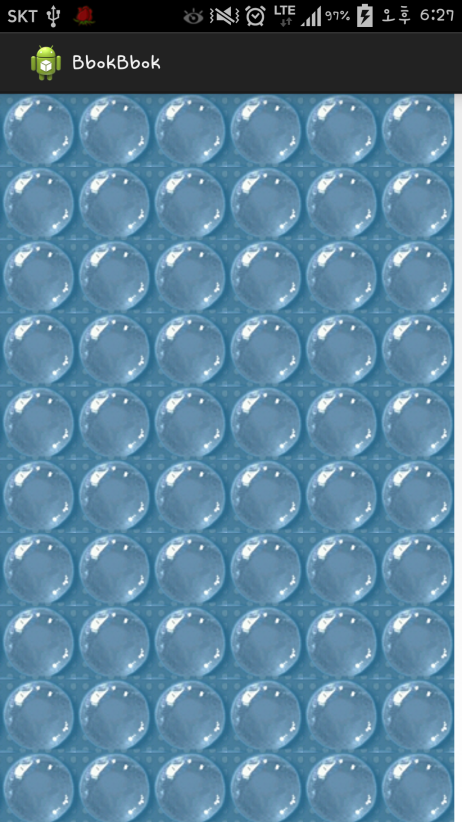
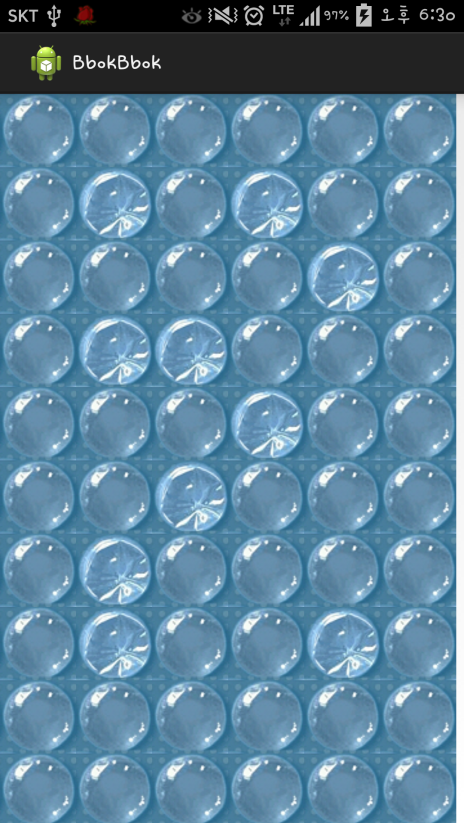
<com.example.hw4\_a.BbokBbok

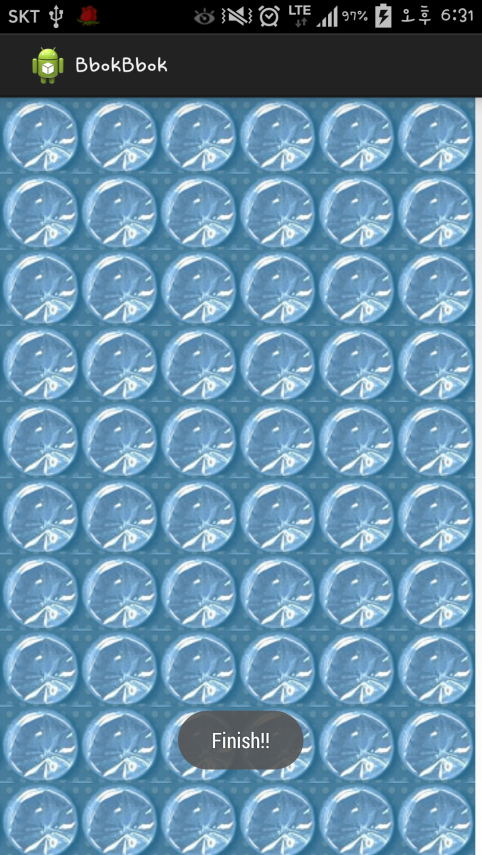
android:layout\_width=*"match\_parent"*

android:layout\_height=*"match\_parent"* />

</FrameLayout>

**Screenshots >>**

** **

****

**HW #4-B : Mazes**

**Activity Java code>> (MainActivity.java)**

**package** com.example.hw4\_b;

**import** android.app.Activity;

**import** android.content.Context;

**import** android.os.Bundle;

**public** **class** MainActivity **extends** Activity {

**public** **static** Context *c*;

@Override

**protected** **void** onCreate(Bundle savedInstanceState) {

**super**.onCreate(savedInstanceState);

setContentView(R.layout.*activity\_main*);

*c* = **this**;

}

}

**View Java code>> (Maze.java)**

**package** com.example.hw4\_b;

**import** android.app.Activity;

**import** android.app.AlertDialog;

**import** android.content.Context;

**import** android.content.DialogInterface;

**import** android.graphics.Bitmap;

**import** android.graphics.BitmapFactory;

**import** android.graphics.Canvas;

**import** android.graphics.Paint;

**import** android.graphics.Path;

**import** android.graphics.Point;

**import** android.graphics.Rect;

**import** android.util.AttributeSet;

**import** android.view.Display;

**import** android.view.MotionEvent;

**import** android.view.View;

**import** android.view.WindowManager;

**public** **class** Maze **extends** View {

**private** **boolean**[][] maze;

**private** **boolean**[][] image;

**private** **int** maxRow, maxCol;

**private** Bitmap wall, wall2, road, startBit, exit;

**private** **int** scrWidth, scrHeight;

**private** **double** mapWidth, mapHeight;

**private** Path mPath;

**private** Paint mPaint;

Point mPoint;

**private** **boolean** start;

**public** Maze(Context c) {

**super**(c);

init(c);

}

**public** Maze(Context c, AttributeSet a) {

**super**(c, a);

init(c);

}

// initialize the board

**public** **void** init(Context c) {

maze = **new** **boolean**[10][8];

image = **new** **boolean**[10][8];

Display display = ((WindowManager) c

.getSystemService(Context.*WINDOW\_SERVICE*)).getDefaultDisplay();

scrWidth = display.getWidth();

scrHeight = display.getHeight();

mPath = **new** Path();

mPaint = **new** Paint();

mPaint.setDither(**true**);

mPaint.setColor(0xFFFFFFFF);

mPaint.setStyle(Paint.Style.*STROKE*);

mPaint.setStrokeJoin(Paint.Join.*ROUND*);

mPaint.setStrokeCap(Paint.Cap.*ROUND*);

mPaint.setStrokeWidth(10);

mPoint = **new** Point();

start = **false**;

**for** (**int** i = 0; i < maze.length; i++)

**for** (**int** j = 0; j < maze[i].length; j++)

{

maze[i][j] = **false**;

image[i][j] =**false**;

}

maxRow = maze.length;

maxCol = maze[0].length;

maze[0][0] = **true**;

maze[maxRow - 1][maxCol - 1] = **true**;

mapWidth = scrWidth / (8.0);

mapHeight = scrHeight / (11.0) - 4.5;

road = BitmapFactory.*decodeResource*(getResources(), R.drawable.*road*);

wall = BitmapFactory.*decodeResource*(getResources(), R.drawable.*wall*);

wall2 = BitmapFactory.*decodeResource*(getResources(), R.drawable.*wall2*);

startBit = BitmapFactory.*decodeResource*(getResources(),

R.drawable.*start*);

exit = BitmapFactory.*decodeResource*(getResources(), R.drawable.*exit*);

makeRoad();

makeRandomWall();

}

**private** **void** makeRoad() {

**int** rand = 0, row = 0, col = 0;

// origin point is row = 0, col = 0;

**while** (**true**) {

rand = getRandom();

**if** (rand == 0) // right

col++;

**else** **if** (rand == 1) // down

row++;

**if** (col >= maxCol)

col--;

**if** (row >= maxRow)

row--;

maze[row][col] = **true**;

**if** (row == 9 && col == 7)

**break**;

}

makeWall();

}

**private** **void** makeWall() {

**int** rand = 0;

**for** (**int** i = 0; i < maxRow; i++) {

**for** (**int** j = 0; j < maxCol; j++) {

rand = getRandom();

**if** (rand == 1)

maze[i][j] = **true**;

}

}

}

**private** **int** getRandom() {

**int** rand = (**int**) (Math.*random*() \* 10);

**if** (rand <= 5)

rand = 0;

**else**

rand = 1;

**return** rand;

}

**private** **void** makeRandomWall()

{

**int** rand;

**for**(**int** i=0; i<maxRow; i++)

{

**for** (**int** j=0; j<maxCol; j++)

{

**if**(maze[i][j] == **false**)

{

rand = getRandom();

**if**(rand == 1)

image[i][j] = **true**;

}

}

}

}

**protected** **void** onDraw(Canvas canvas) {

Rect rect = **new** Rect();

**for** (**int** i = 0; i < maxRow; i++) {

**for** (**int** j = 0; j < maxCol; j++) {

rect.set((**int**) (j \* mapWidth), (**int**) (i \* mapHeight), (**int**) (j

\* mapWidth + mapWidth),

(**int**) (i \* mapHeight + mapHeight));

**if** (i == 0 && j == 0)

canvas.drawBitmap(startBit, **null**, rect, **null**);

**else** **if** (i == maxRow - 1 && j == maxCol - 1)

canvas.drawBitmap(exit, **null**, rect, **null**);

**else** {

**if** (maze[i][j])

canvas.drawBitmap(road, **null**, rect, **null**);

**else** {

**if** (image[i][j])

canvas.drawBitmap(wall, **null**, rect, **null**);

**else**

canvas.drawBitmap(wall2, **null**, rect, **null**);

}

}

}

}

canvas.drawPath(mPath, mPaint);

}

@Override

**public** **boolean** onTouchEvent(MotionEvent event) {

**float** eventX = event.getX();

**float** eventY = event.getY();

**if** (event.getAction() == MotionEvent.*ACTION\_DOWN*) {

mPath.reset(); // 이전까지 그린거 없어짐.

mPath.moveTo(eventX, eventY);

**if** ((0 < eventX && eventX < mapWidth)

&& (0 < eventY && eventY < mapHeight))

start = **true**;

} **else** **if** (event.getAction() == MotionEvent.*ACTION\_MOVE*) {

mPath.quadTo(eventX, eventY, (mPoint.x + eventX) / 2,

(mPoint.y + eventY) / 2);

**for** (**int** i = 0; i < maxRow; i++) {

**for** (**int** j = 0; j < maxCol; j++) {

**if** ((j \* mapWidth < eventX && eventX < j \* mapWidth

+ mapWidth)

&& (i \* mapHeight < eventY && eventY < i

\* mapHeight + mapHeight)) {

**if** (maze[i][j] == **false**) {

start = **false**;

**break**;

}

}

}

}

} **else** **if** (event.getAction() == MotionEvent.*ACTION\_UP*) {

**if** (start == **true**)

mPath.lineTo(eventX, eventY);

**else**

mPath.reset();

**if** (((maxCol - 1) \* mapWidth < eventX && eventX < (maxCol)

\* mapWidth)

&& ((maxRow - 1) \* mapHeight < eventY && eventY < (maxRow)

\* mapHeight)) {

**if** (start == **true**) {

AlertDialog dialBox = createDialogBox(**this**.getContext());

dialBox.show();

}

}

}

mPoint.x = (**int**) eventX;

mPoint.y = (**int**) eventY;

invalidate();

**return** **true**;

}

**private** AlertDialog createDialogBox(**final** Context c) {

AlertDialog myQuittingDialogBox = **new** AlertDialog.Builder(

**this**.getContext())

// set message, title, and icon

.setTitle("Finish!!")

.setMessage(

"You are genius!!\nYou solved the maze!!\nDo you want to play the game again?")

.setIcon(R.drawable.*kitty*)

// set three option buttons

.setPositiveButton("Yes",

**new** DialogInterface.OnClickListener() {

**public** **void** onClick(DialogInterface dialog,

**int** whichButton) {

// whatever should be done when answering "YES"

// goes here

init(c);

invalidate();

}

})// setPositiveButton

.setNeutralButton("Cancel",

**new** DialogInterface.OnClickListener() {

**public** **void** onClick(DialogInterface dialog,

**int** whichButton) {

// whatever should be done when answering

// "CANCEL" goes here

}// OnClick

})// setNeutralButton

.setNegativeButton("NO", **new** DialogInterface.OnClickListener() {

**public** **void** onClick(DialogInterface dialog, **int** whichButton) {

// whatever should be done when answering "NO" goes here

((Activity) MainActivity.*c*).finish();

}

})// setNegativeButton

.create();

**return** myQuittingDialogBox;

}

}

**Layout file>> (activity\_main.xml)**

<RelativeLayout xmlns:android=*"http://schemas.android.com/apk/res/android"*

xmlns:tools=*"http://schemas.android.com/tools"*

android:id=*"@+id/container"*

android:layout\_width=*"match\_parent"*

android:layout\_height=*"match\_parent"*

tools:context=*"com.example.hw4\_b.MainActivity"*

tools:ignore=*"MergeRootFrame"* >

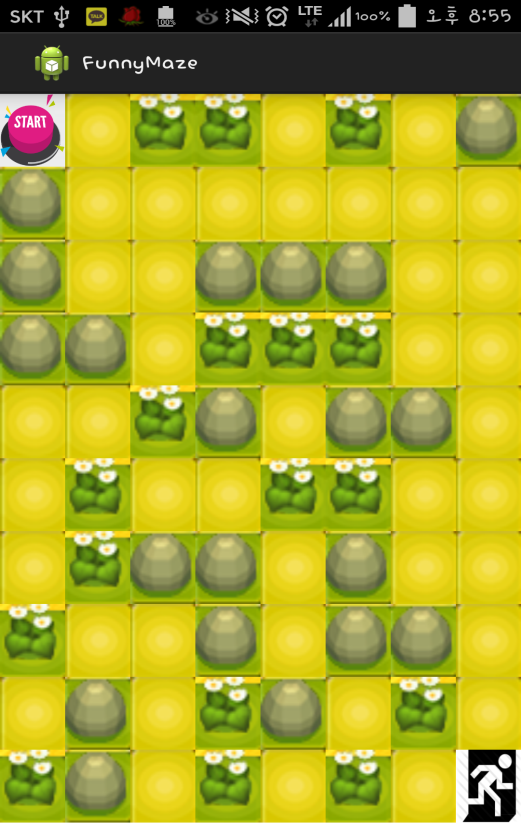
<com.example.hw4\_b.Maze

android:layout\_width=*"match\_parent"*

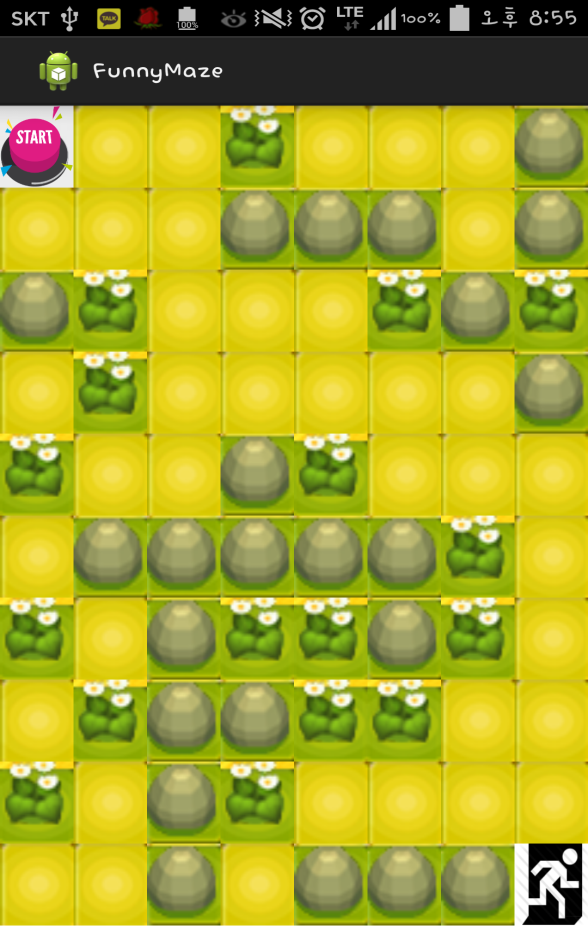
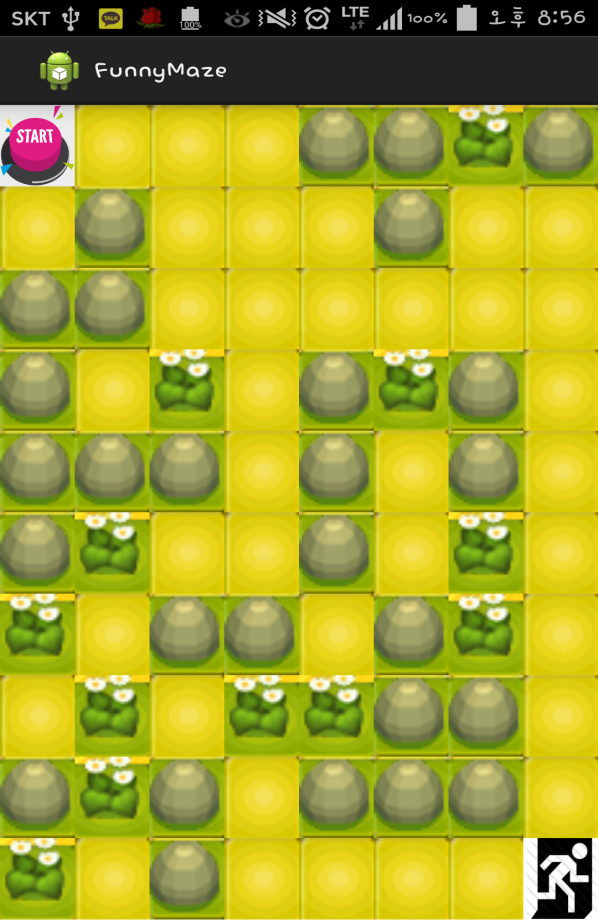
android:layout\_height=*"match\_parent"* />

</RelativeLayout>

**Screen shots>>**

** **

**(random maps)**

** **