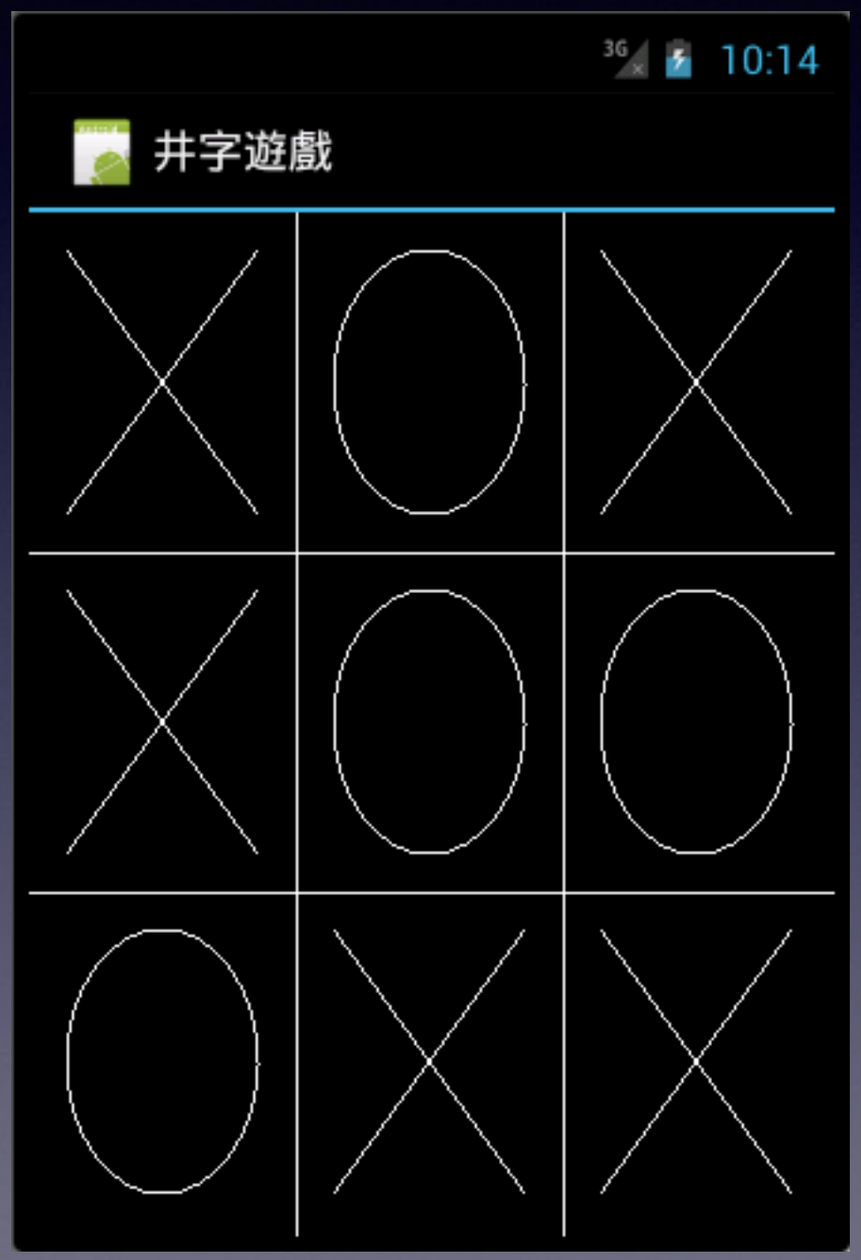
Android程式設計

井字遊戲

王昱景 Brian Wang
brian.wang.frontline@gmail.com

井字遊戲



- 井字遊戲是 2D 繪圖的應用
- 使用 2D 繪圖方法來繪出井字形的遊戲板和在指定儲存格繪出 O 和 X 的圖形

TicTacToeDemo

- 開啟和執行 Android 專案
- 建立 Activity 活動執行井字遊戲
- 建立井字形遊戲板的 MainBoard 類別
- 建立邏輯儲存格的 CellBoard 類別

1.開啟和執行 Android 專案

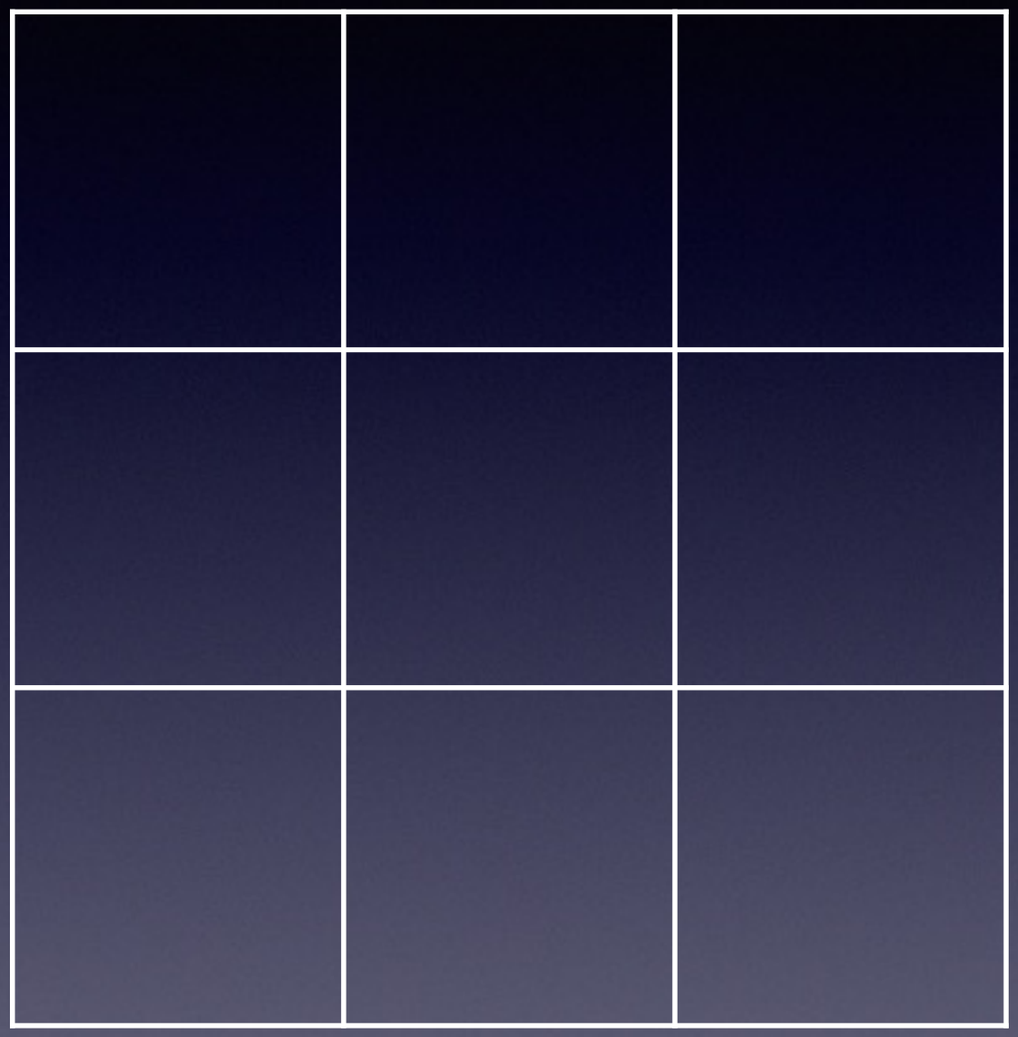
- 請啟動 Eclipse IDE
- 建立 Android 專案
  - Project Name: TicTacToeDemo
  - Build Target: Android 4.0.3
  - Package Name: tw.edu.vnu

2.建立 Activity 活動執行井字遊戲

```
package tw.edu.vnu;

import android.app.Activity;
import android.os.Bundle;

public class TieTacToeDemoActivity extends Activity {

    /** Called when the activity is first created. */
    @Override
    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(new MainBoard(this));
    }

}
```

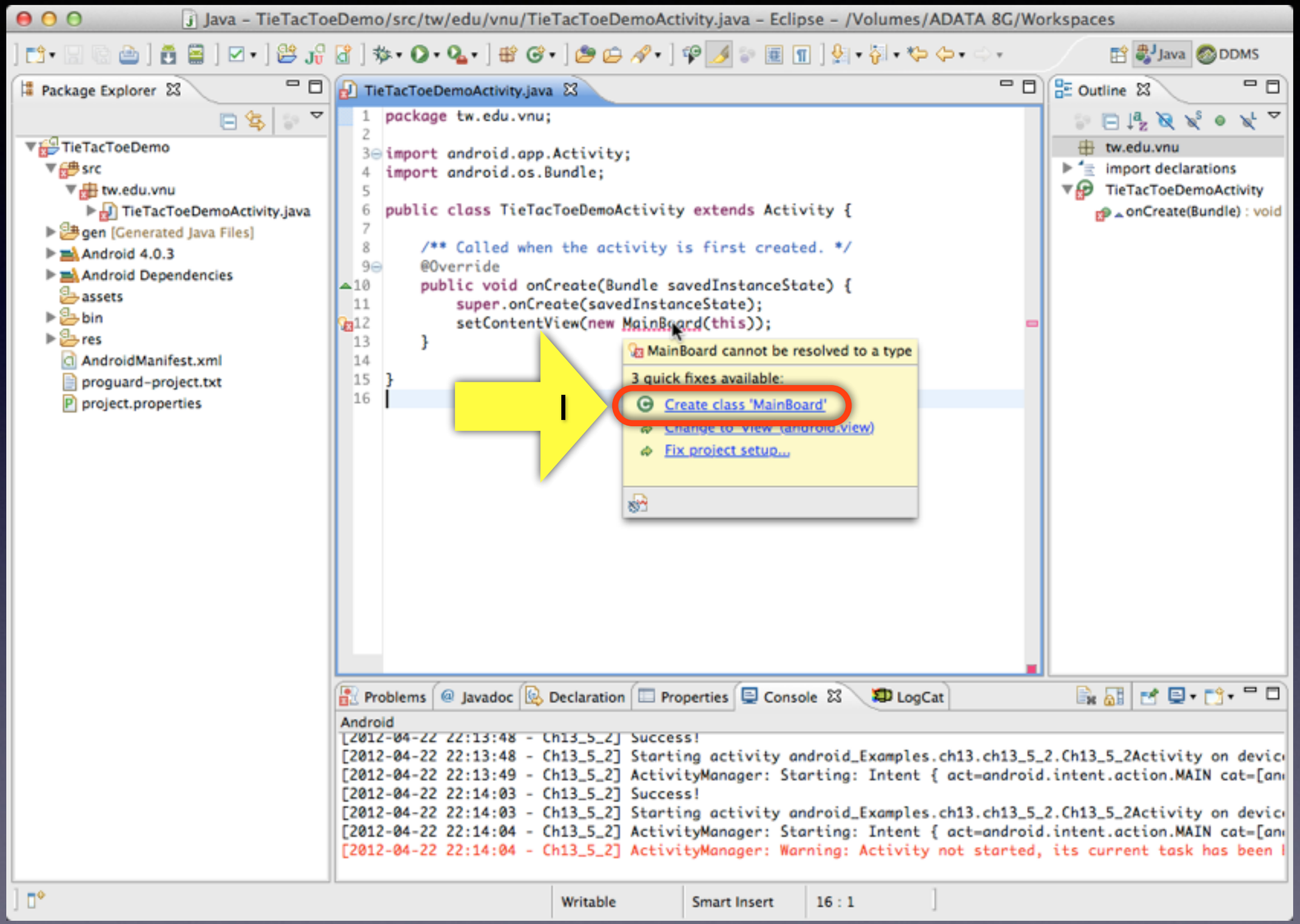
3.建立井字形遊戲板的 MainBoard 類別

- 建立 MainBoard 類別
- 宣告成員變數
- 建構子
- onDraw() 方法
- onMeasure() 方法
- calculateLinePlacements() 方法
- onTouchEvent() 事件處理方法
- drawBoard() 方法



MainBoard 類別是在 View 物件上繪出井字形的遊戲板。

長方形框是行動裝置的螢幕，由上而下；由左至右繪出 4 條線來建立井字形，將這 4 條線繪在 Bitmap 物件。


Java – TieTacToeDemo/src/tw/edu/vnu/TieTacToeDemoActivity.java – Eclipse – /Volumes/ADATA 8G/Workspaces
Java
DDMS
Package Explorer
TieTacToeDemo
src
tw.edu.vnu
TieTacToeDemoActivity.java
gen [Generated Java Files]
Android 4.0.3
Android Dependencies
assets
bin
res
AndroidManifest.xml
proguard-project.txt
project.properties
TieTacToeDemoActivity.java
1 package tw.edu.vnu;
2
3 import android.app.Activity;
4 import android.os.Bundle;
5
6 public class TieTacToeDemoActivity extends Activity {
7
8 /** Called when the activity is first created. */
9 @Override
10 public void onCreate(Bundle savedInstanceState) {
11 super.onCreate(savedInstanceState);
12 setContentView(new MainBoard(this));
13 }
14
15 }
16
MainBoard cannot be resolved to a type
3 quick fixes available:
Create class 'MainBoard'
Change to 'View' (android.view)
Fix project setup...
Outline
tw.edu.vnu
import declarations
TieTacToeDemoActivity
onCreate(Bundle) : void
Problems
Javadoc
Declaration
Properties
Console
LogCat
Android
[2012-04-22 22:13:48 - Ch13_5_2] Success!
[2012-04-22 22:13:48 - Ch13_5_2] Starting activity android_Examples.ch13.ch13_5_2.Ch13_5_2Activity on devic
[2012-04-22 22:13:49 - Ch13_5_2] ActivityManager: Starting: Intent { act=android.intent.action.MAIN cat=[an
[2012-04-22 22:14:03 - Ch13_5_2] Success!
[2012-04-22 22:14:03 - Ch13_5_2] Starting activity android_Examples.ch13.ch13_5_2.Ch13_5_2Activity on devic
[2012-04-22 22:14:04 - Ch13_5_2] ActivityManager: Starting: Intent { act=android.intent.action.MAIN cat=[an
[2012-04-22 22:14:04 - Ch13_5_2] ActivityManager: Warning: Activity not started, its current task has been
Writable
Smart Insert
16 : 1

New

Java Class

Create a new Java class.

Source folder: TieTacToeDemo/src — Browse...

Package: tw.edu.vnu — Browse...

Enclosing type: tw.edu.vnu.TieTacToeDemoActivity — Browse...

Name: MainBoard

Modifiers: public default private protected

abstract final static

Superclass: android.view.View — Browse...

Interfaces: Add... Remove

Which method stubs would you like to create?

public static void main(String[] args)

Constructors from superclass

Inherited abstract methods

Do you want to add comments? (Configure templates and default value here)

Generate comments

2

Finish

Java – TieTacToeDemo/src/tw/edu/vnu/MainBoard.java – Eclipse – /Volumes/ADATA 8G/Workspaces

Java DDMS

Package Explorer

- TieTacToeDemo
  - src
    - tw.edu.vnu
      - MainBoard.java
      - TieTacToeDemoActivity.java
  - gen [Generated Java Files]
  - Android 4.0.3
  - Android Dependencies
  - assets
  - bin
  - res
  - AndroidManifest.xml
  - proguard-project.txt
  - project.properties

TieTacToeDemoActivity.java | MainBoard.java

```
1  package tw.edu.vnu;
2
3  import android.view.View;
4
5  public class MainBoard extends View {
6
7  }
```

3

Implicit super constructor View() is undefined for default constructor. Must define an explicit constructor

3 quick fixes available:

- Add constructor 'MainBoard(Context)'
- Add constructor 'MainBoard(Context,AttributeSet)'
- Add constructor 'MainBoard(Context,AttributeSet,int)'

Press 'F2' for focus

Outline

- tw.edu.vnu
- import declarations
- MainBoard

Problems | Javadoc | Declaration | Properties | Console | LogCat

Android

```
[2012-04-22 22:13:48 - Ch13_5_2] Success!
[2012-04-22 22:13:48 - Ch13_5_2] Starting activity android_Examples.ch13.ch13_5_2.Ch13_5_2Activity on devic
[2012-04-22 22:13:49 - Ch13_5_2] ActivityManager: Starting: Intent { act=android.intent.action.MAIN cat=[an
[2012-04-22 22:14:03 - Ch13_5_2] Success!
[2012-04-22 22:14:03 - Ch13_5_2] Starting activity android_Examples.ch13.ch13_5_2.Ch13_5_2Activity on devic
[2012-04-22 22:14:04 - Ch13_5_2] ActivityManager: Starting: Intent { act=android.intent.action.MAIN cat=[an
[2012-04-22 22:14:04 - Ch13_5_2] ActivityManager: Warning: Activity not started, its current task has been
```

Writable | Smart Insert | 1 : 1

Java – TieTacToeDemo/src/tw/edu/vnu/MainBoard.java – Eclipse – /Volumes/ADATA 8G/Workspaces

Java DDMS

Package Explorer

- TieTacToeDemo
  - src
    - tw.edu.vnu
      - MainBoard.java
      - TieTacToeDemoActivity.java
  - gen [Generated Java Files]
  - Android 4.0.3
  - Android Dependencies
  - assets
  - bin
  - res
  - AndroidManifest.xml
  - proguard-project.txt
  - project.properties

TieTacToeDemoActivity.java | MainBoard.java

```
1  package tw.edu.vnu;
2
3  import android.content.Context;
4  import android.view.View;
5
6  public class MainBoard extends View {
7
8      public MainBoard(Context context) {
9          super(context);
10         // TODO Auto-generated constructor stub
11     }
12
13 }
14
```

Outline

- tw.edu.vnu
- import declarations
- MainBoard
  - MainBoard(Context)

Problems | Javadoc | Declaration | Properties | Console | LogCat

Android

```
[2012-04-22 22:13:48 - Ch13_5_2] Success!
[2012-04-22 22:13:48 - Ch13_5_2] Starting activity android_Examples.ch13.ch13_5_2.Ch13_5_2Activity on devic
[2012-04-22 22:13:49 - Ch13_5_2] ActivityManager: Starting: Intent { act=android.intent.action.MAIN cat=[an
[2012-04-22 22:14:03 - Ch13_5_2] Success!
[2012-04-22 22:14:03 - Ch13_5_2] Starting activity android_Examples.ch13.ch13_5_2.Ch13_5_2Activity on devic
[2012-04-22 22:14:04 - Ch13_5_2] ActivityManager: Starting: Intent { act=android.intent.action.MAIN cat=[an
[2012-04-22 22:14:04 - Ch13_5_2] ActivityManager: Warning: Activity not started, its current task has been
```

Writable | Smart Insert | 8 : 29

宣告成員變數

```
package tw.edu.vnu;

import android.content.Context;
import android.graphics.Bitmap;
import android.graphics.Canvas;
import android.graphics.Paint;
import android.graphics.Point;
import android.view.View;

public class MainBoard extends View {

    private int h, w;
    private Bitmap bitmap;
    private Canvas buffer;
    private Paint paint;
    private boolean isDrawX;
    private Point[] hLine01, hLine02, vLine01, vLine02;
    private CellBoard cellBoard;

    public MainBoard(Context context) {
        super(context);
    }

}
```

建構子

```
// 建構子
public MainBoard(Context context) {
    super(context);

    paint = new Paint();
    paint.setColor(Color.WHITE);
    paint.setStyle(Paint.Style.STROKE);
    isDrawX = true;
}
```

onDraw() 方法

```
@Override
protected void onDraw(Canvas canvas) {
    // 繪出Bitmap物件
    canvas.drawBitmap(bitmap, 0, 0, paint);
```

onMeasure() 方法

```
@Override
protected void onMeasure(int widthMeasureSpec, int heightMeasureSpec) {
    // 取得螢幕寬高尺寸
    h = View.MeasureSpec.getSize(heightMeasureSpec);
    w = View.MeasureSpec.getSize(widthMeasureSpec);

    setMeasuredDimension(w, h); // 儲存View物件的寬和高

    // 建立Bitmap物件
    bitmap = Bitmap.createBitmap(w, h, Bitmap.Config.ARGB_8888);
    buffer = new Canvas(bitmap); // 使用Bitmap建立Canvas物件
    // 計算點的座標
    calculateLinePlacements();
    drawBoard(); // 繪出井字
}
```

calculateLinePlacements() 方法

```
private void calculateLinePlacements() {
    int cellH = h / 3;
    int cellW = w / 3;

    hLine01 = new Point[] { new Point(0, cellH), new Point(w, cellH) };
    hLine02 = new Point[] { new Point(0, 2 * cellH), new Point(w, 2 * cellH) };

    vLine01 = new Point[] { new Point(cellW, 0), new Point(cellW, h) };
    vLine02 = new Point[] { new Point(2 * cellW, 0), new Point(2 * cellW, h) };

    // 建立CellBoard物件
    cellBoard = new CellBoard(cellW, cellH);
}
```

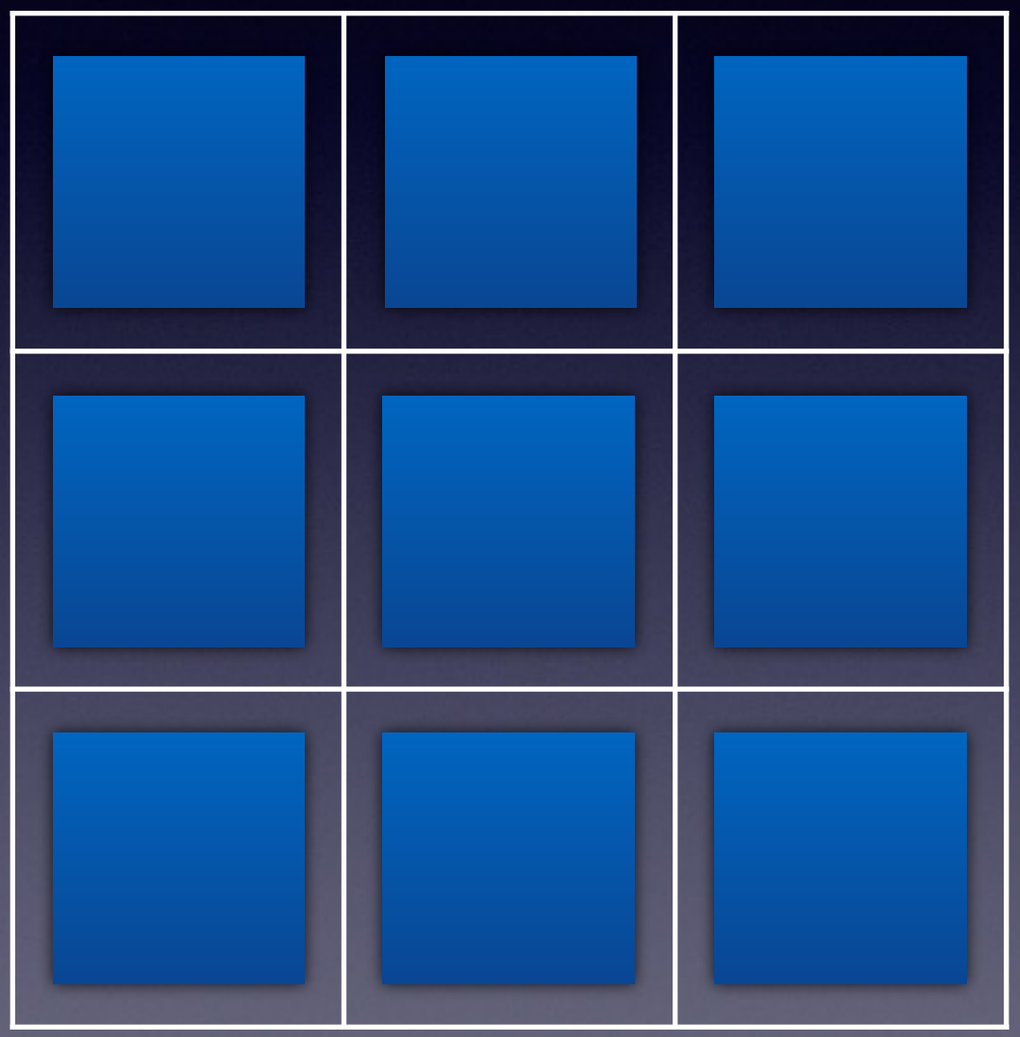
onTouchEvent() 事件處理方法

```
@Override
public boolean onTouchEvent(MotionEvent event) {
    if (event.getAction() == MotionEvent.ACTION_DOWN) {
        // 取得繪圖長方形的RectF
        RectF position = cellBoard.getCellToFill(event.getX(), event.getY());

        if (position != null) {
            if (isDrawX) {
                // 畫X
                buffer.drawLine(position.left, position.top, position.right, position.bottom, paint);
                buffer.drawLine(position.right, position.top, position.left, position.bottom, paint);
            } else {
                // 畫X
                buffer.drawOval(position, paint);
            }

            isDrawX = !isDrawX;
            invalidate();
        }
    }
```

drawBoard() 方法

```
// 繪出遊戲的井字
private void drawBoard() {
    buffer.drawLine(hLine01[0].x, hLine01[0].y, hLine01[1].x, hLine01[1].y, paint);
    buffer.drawLine(hLine02[0].x, hLine02[0].y, hLine02[1].x, hLine02[1].y, paint);
    buffer.drawLine(vLine01[0].x, vLine01[0].y, vLine01[1].x, vLine01[1].y, paint);
    buffer.drawLine(vLine02[0].x, vLine02[0].y, vLine02[1].x, vLine02[1].y, paint);

    invalidate();
}
```

4.建立邏輯儲存格的 CellBoard 類別



CellBoard 類別是在井字形的遊戲板上建立 9 個邏輯儲存格的 Cell 物件。

9 個長方形是邏輯儲存格，比井字形的儲存格小一些，就是繪出 X 或 O 圖形的長方形，也就是 Cell 物件。

宣告成員變數

```
package tw.edu.vnu;

import android.graphics.RectF;

public class CellBoard {

   private int w;
   private int h;
   protected Cell[] position;

}
```

建構子

```
// 建構子
public CellBoard(int cellWidth, int cellHeight) {
   w = cellWidth;
   h = cellHeight;
   initialBoardCells();
}
```

Cell 內層類別

```
// 宣告儲存格類別
private class Cell extends RectF {
    private boolean filled;

    // 建構子
    public Cell(float left, float top, float right, float bottom) {
        super(left, top, right, bottom);
        filled = false;
    }

    // 指定已填入圖形
    public void setFilled(boolean filled) {
        this.filled = filled;
    }

    // 檢查是否已填入圖形
    public boolean isFilled() {
        return filled;
    }
}
```

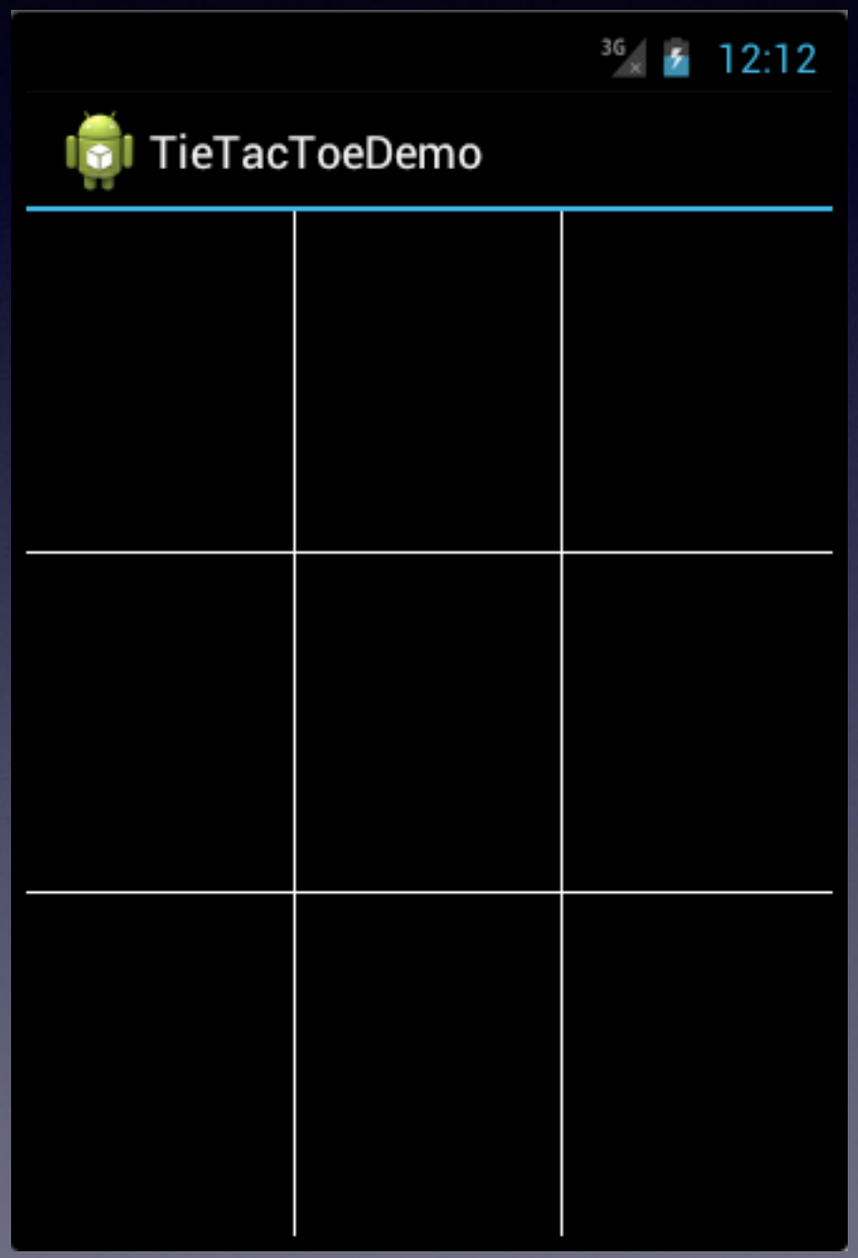
initialBoardCells() 方法

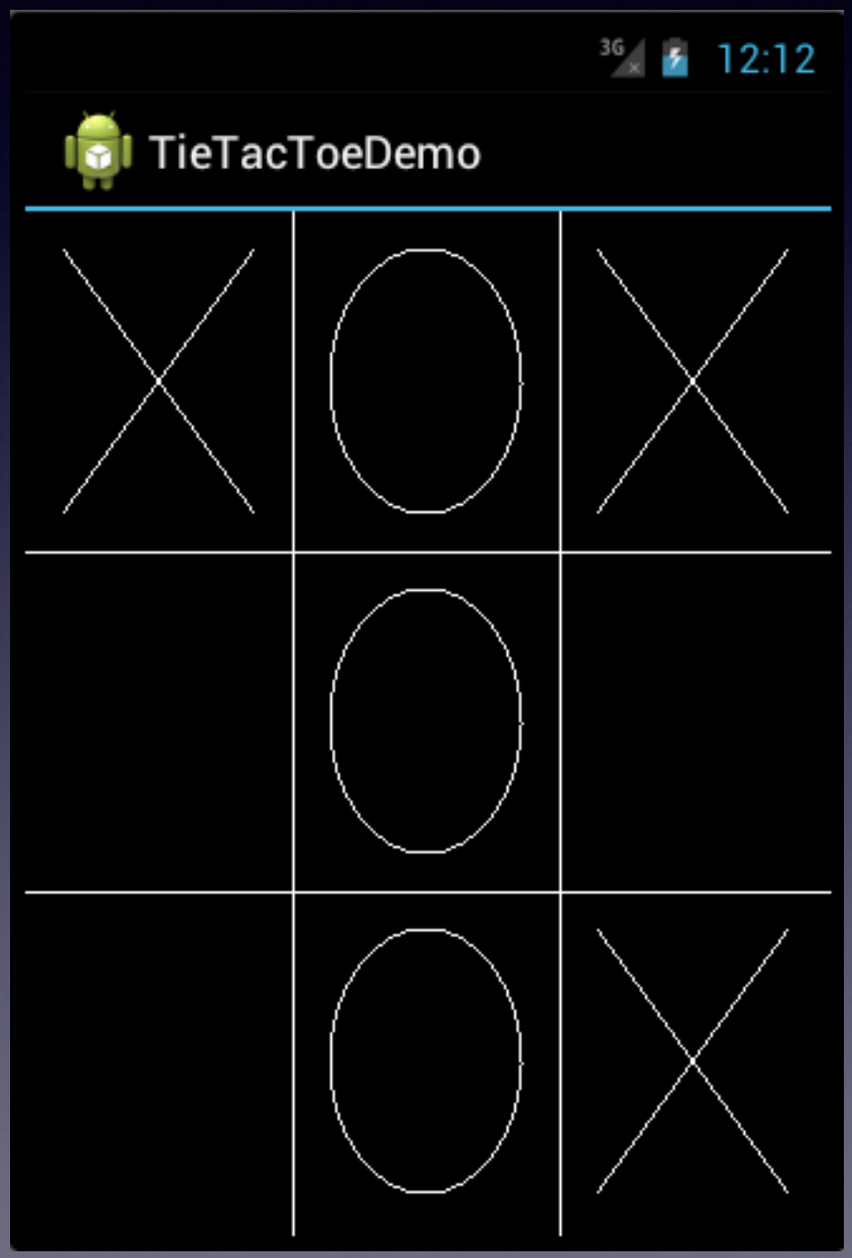
```
// 計算儲存格陣列各儲存格的座標
private void initialBoardCells() {
    int offset = 15; // 位移量
    position = new Cell[9];

    // 第一欄
    position[0] = new Cell(0 + offset, 0 + offset, w - offset, h - offset);
    position[1] = new Cell(w + offset, 0 + offset, 2 * w - offset, h - offset);
    position[2] = new Cell(2 * w + offset, 0 + offset, 3 * w - offset, h - offset);
    // 第二欄
    position[3] = new Cell(0 + offset, h + offset, w - offset, 2 * h - offset);
    position[4] = new Cell(w + offset, h + offset, 2 * w - offset, 2 * h - offset);
    position[5] = new Cell(2 * w + offset, h + offset, 3 * w - offset, 2 * h - offset);
    // 第三欄
    position[6] = new Cell(0 + offset, 2 * h + offset, w - offset, 3 * h - offset);
    position[7] = new Cell(w + offset, 2 * h + offset, 2 * w - offset, 3 * h - offset);
    position[8] = new Cell(2 * w + offset, 2 * h + offset, 3 * w - offset, 3 * h - offset);
}
```

getCellToFill() 方法

```
// 取得填入的Cell物件
public RectF getCellToFill(float x, float y) {
    for (Cell bp : position) {

        if (bp.contains(x, y) && !bp.isFilled()) {
            RectF retCell = new RectF(bp);
            bp.setFilled(true);
            return retCell;
        }
    }

    return null;
}
```

5.執行 Android 模擬器佈署程式





MainBoard 完整程式碼

```
package tw.edu.vnu;

import android.content.Context;
import android.graphics.Bitmap;
import android.graphics.Canvas;
import android.graphics.Color;
import android.graphics.Paint;
import android.graphics.Point;
import android.graphics.RectF;
import android.view.MotionEvent;
import android.view.View;

public class MainBoard extends View {

    private int h, w;
    private Bitmap bitmap;
    private Canvas buffer;
    private Paint paint;
    private boolean isDrawX;
    private Point[] hLine01, hLine02, vLine01, vLine02;
    private CellBoard cellBoard;

    // 建構子
    public MainBoard(Context context) {
        super(context);

        paint = new Paint();
        paint.setColor(Color.WHITE);
        paint.setStyle(Paint.Style.STROKE);
        isDrawX = true;
    }
```

```java
@Override
protected void onDraw(Canvas canvas) {
    // 繪出Bitmap物件
    canvas.drawBitmap(bitmap, 0, 0, paint);
}

@Override
protected void onMeasure(int widthMeasureSpec, int heightMeasureSpec) {
    // 取得螢幕寬高尺寸
    h = View.MeasureSpec.getSize(heightMeasureSpec);
    w = View.MeasureSpec.getSize(widthMeasureSpec);

    setMeasuredDimension(w, h); // 儲存View物件的寬和高

    // 建立Bitmap物件
    bitmap = Bitmap.createBitmap(w, h, Bitmap.Config.ARGB_8888);
    buffer = new Canvas(bitmap); // 使用Bitmap建立Canvas物件
    // 計算點的座標
    calculateLinePlacements();
    drawBoard(); // 繪出井字
}

private void calculateLinePlacements() {
    int cellH = h / 3;
    int cellW = w / 3;

    hLine01 = new Point[] { new Point(0, cellH), new Point(w, cellH) };
    hLine02 = new Point[] { new Point(0, 2 * cellH), new Point(w, 2 * cellH) };

    vLine01 = new Point[] { new Point(cellW, 0), new Point(cellW, h) };
    vLine02 = new Point[] { new Point(2 * cellW, 0), new Point(2 * cellW, h) };

    // 建立CellBoard物件
    cellBoard = new CellBoard(cellW, cellH);
}
```

```java
@Override
public boolean onTouchEvent(MotionEvent event) {
    if (event.getAction() == MotionEvent.ACTION_DOWN) {
        // 取得繪圖長方形的RectF
        RectF position = cellBoard.getCellToFill(event.getX(), event.getY());

        if (position != null) {
            if (isDrawX) {
                // 畫X
                buffer.drawLine(position.left, position.top, position.right, position.bottom, paint);
                buffer.drawLine(position.right, position.top, position.left, position.bottom, paint);
            } else {
                // 畫X
                buffer.drawOval(position, paint);
            }

            isDrawX = !isDrawX;
            invalidate();
        }
    }

    return true;
}

// 繪出遊戲的井字
private void drawBoard() {
    buffer.drawLine(hLine01[0].x, hLine01[0].y, hLine01[1].x, hLine01[1].y, paint);
    buffer.drawLine(hLine02[0].x, hLine02[0].y, hLine02[1].x, hLine02[1].y, paint);
    buffer.drawLine(vLine01[0].x, vLine01[0].y, vLine01[1].x, vLine01[1].y, paint);
    buffer.drawLine(vLine02[0].x, vLine02[0].y, vLine02[1].x, vLine02[1].y, paint);

    invalidate();
}

}
```

CellBoard 完整程式碼

```
package tw.edu.vnu;

import android.graphics.RectF;

public class CellBoard {

    private int w;
    private int h;
    protected Cell[] position;

    // 建構子
    public CellBoard(int cellWidth, int cellHeight) {
        w = cellWidth;
        h = cellHeight;
        initialBoardCells();
    }
```

```
// 宣告儲存格類別
private class Cell extends RectF {
    private boolean filled;

    // 建構子
    public Cell(float left, float top, float right, float bottom) {
        super(left, top, right, bottom);
        filled = false;
    }

    // 指定已填入圖形
    public void setFilled(boolean filled) {
        this.filled = filled;
    }

    // 檢查是否已填入圖形
    public boolean isFilled() {
        return filled;
    }
}
```

```
// 計算儲存格陣列各儲存格的座標
private void initialBoardCells() {
     int offset = 15; // 位移量
     position = new Cell[9];

    // 第一欄
     position[0] = new Cell(0 + offset, 0 + offset, w - offset, h - offset);
     position[1] = new Cell(w + offset, 0 + offset, 2 * w - offset, h - offset);
     position[2] = new Cell(2 * w + offset, 0 + offset, 3 * w - offset, h - offset);
     // 第二欄
     position[3] = new Cell(0 + offset, h + offset, w - offset, 2 * h - offset);
     position[4] = new Cell(w + offset, h + offset, 2 * w - offset, 2 * h - offset);
     position[5] = new Cell(2 * w + offset, h + offset, 3 * w - offset, 2 * h - offset);
     // 第三欄
     position[6] = new Cell(0 + offset, 2 * h + offset, w - offset, 3 * h - offset);
     position[7] = new Cell(w + offset, 2 * h + offset, 2 * w - offset, 3 * h - offset);
     position[8] = new Cell(2 * w + offset, 2 * h + offset, 3 * w - offset, 3 * h - offset);
}
    }

    // 取得填入的Cell物件
    public RectF getCellToFill(float x, float y) {
         for (Cell bp : position) {

              if (bp.contains(x, y) && !bp.isFilled()) {
                   RectF retCell = new RectF(bp);
                   bp.setFilled(true);
                   return retCell;
              }
         }

         return null;
    }

}
```