Event-Driven 應用開發與活用 UML 狀態機

1. 狀態機與 Android 的天作之合

1.1 Android 是狀態機動力的來源

當 Android 應用程式執行時，Android 會不斷發出訊息(表示事件發生)給我們所設計的狀態機，持續推動狀態機的運轉。Android 成爲狀態機運轉的動力來源。如圖 1-15 所示。

1.2 畫面布局(Layout)與狀態的聯想

如何從 Android 畫面的變化中找出狀態及其轉移呢？這是大家最常提出的問題。其最容易的答案是：從 Android 的 Layout 聯想到狀態。一開始，不需要太完美的切入點，只要一個 Layout 對應到一個狀態就行了。

1.3 狀態機直接與 Android 親密互動

請留意，有人常誤解這裡使用狀態機的目的。一般而言，使用狀態機(或 UML 的狀態圖)的目的有二：

1. 做爲系統分析與設計的工具：以發掘 User 的需求爲主，先規劃狀態機，然後才思考如何在 Android 畫面上實現該狀態機之設計。
2. 做爲創造高可靠度的工具：以嚴格控制系統爲主，先完成畫面 Layout 的設計(方法途徑不拘)，然後設計狀態機來精確控制 User 與系統的互動行爲。

於此，我們是針對上述的第 2 項目的而使用狀態機。

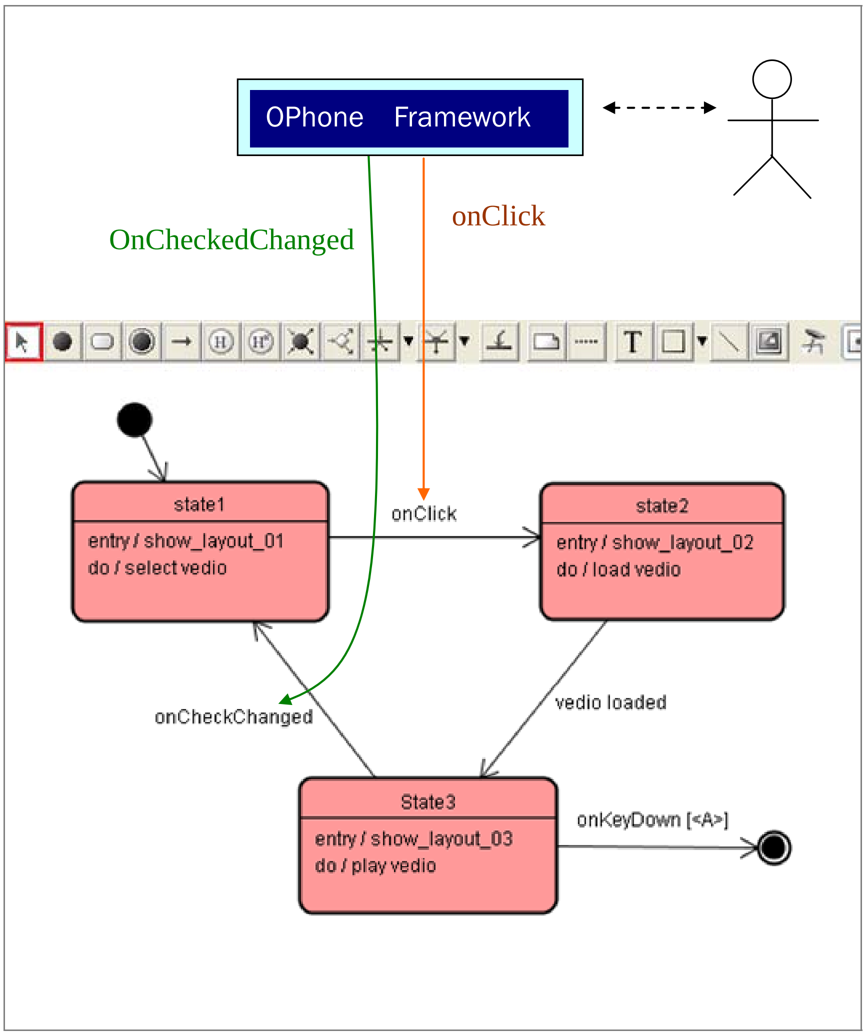


圖 1、Android 推動狀態機的運轉

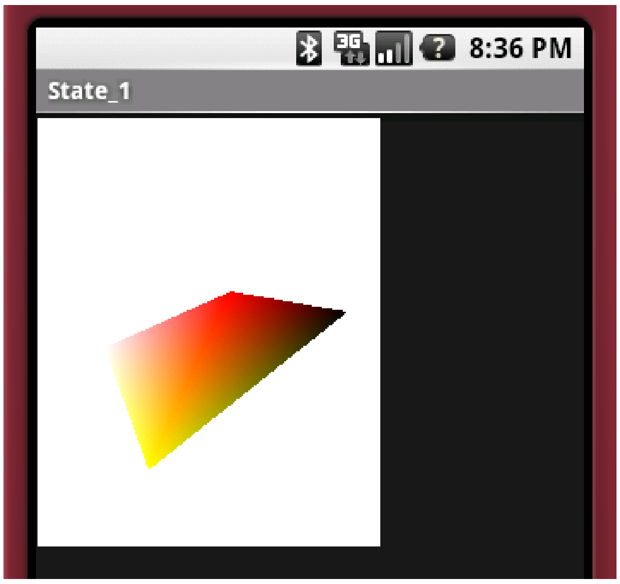
當 User 按下<A>鍵，Linux 作業系統偵測到此事件發生了，會傳達訊息給 Android 應用框架(Framwork)，Android 就發出 OnKeyDown 訊息(或稱爲事件)給我們設計的狀態機，就推動狀態機的運轉了。同樣地，當 User 畫面上的 CheckBox 時，Android 就發出 OnCheckedChanged 訊息(或稱爲事件)給狀態機，觸發狀態機的轉移了。

1.4 以實例解說 Android 與狀態機的互動

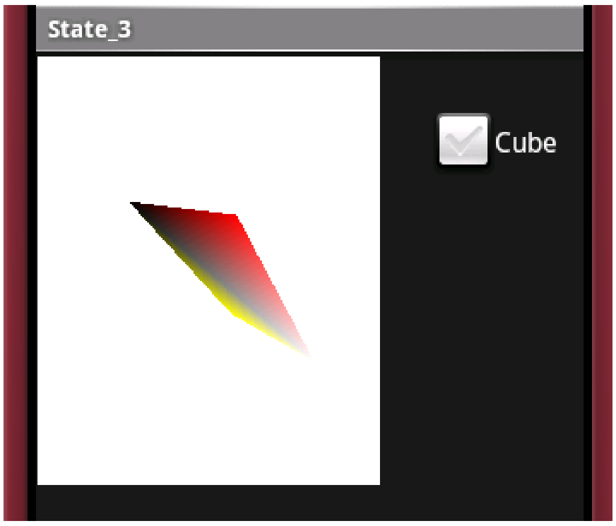
此範例含有一個 3D 繪圖視窗，以及一個 CheckBox 按鈕。預設情形是：CheckBox 未打勾，而且 3D 視窗只繪出一個錐型體。當 User 在 CheckBox 上打勾時，3D 視窗除了繪出一個錐型體之外，還繪出一個正立方體。

Layout 規劃與呈現

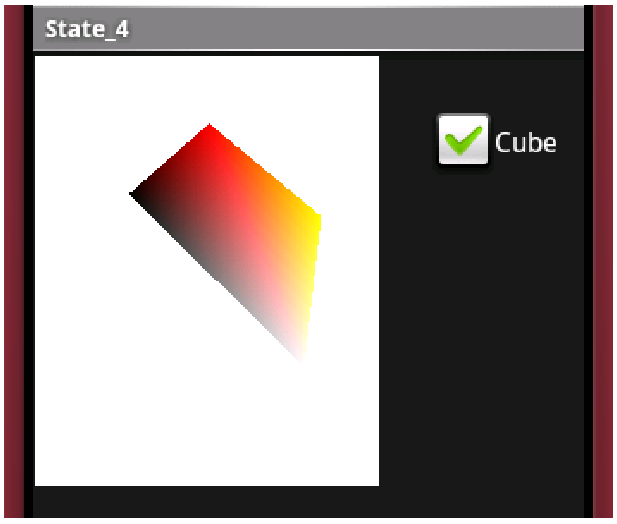
此範例含有兩個畫面佈局：layout_01 和 layout_02。其 layout_01 如下：



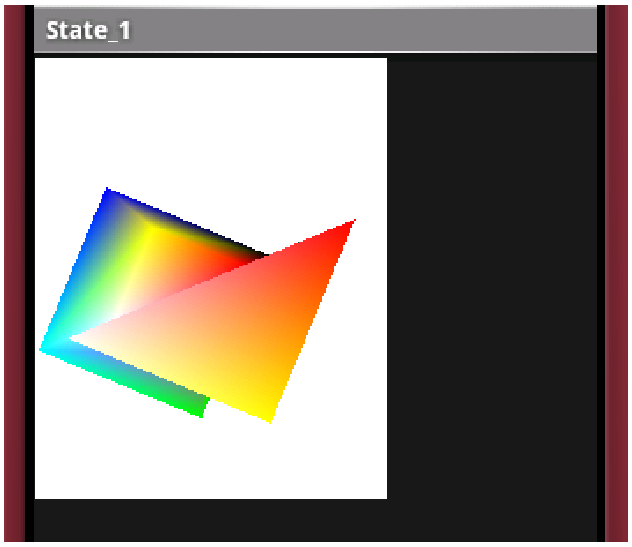
按下<A>鍵之後，就變換到 layout_02。如下：



將 CheckBox 打勾，仍然是 layout_02，如下：



然後，按下<A>鍵，又變換回到 layout_01 畫面。如下圖：



再按下<A>鍵之後，又變換到 layout_02。如此，兩個 Layout 來回變換。

設計狀態機

上述的 UI 畫面設計，含有兩個 Layout，基於上述聯想模式，我們的狀態機只需要兩個上層狀態就可以了。但是狀態 2 內需要另一個並行狀態來表示 CheckBox 所觸發的事件。如下圖 2。

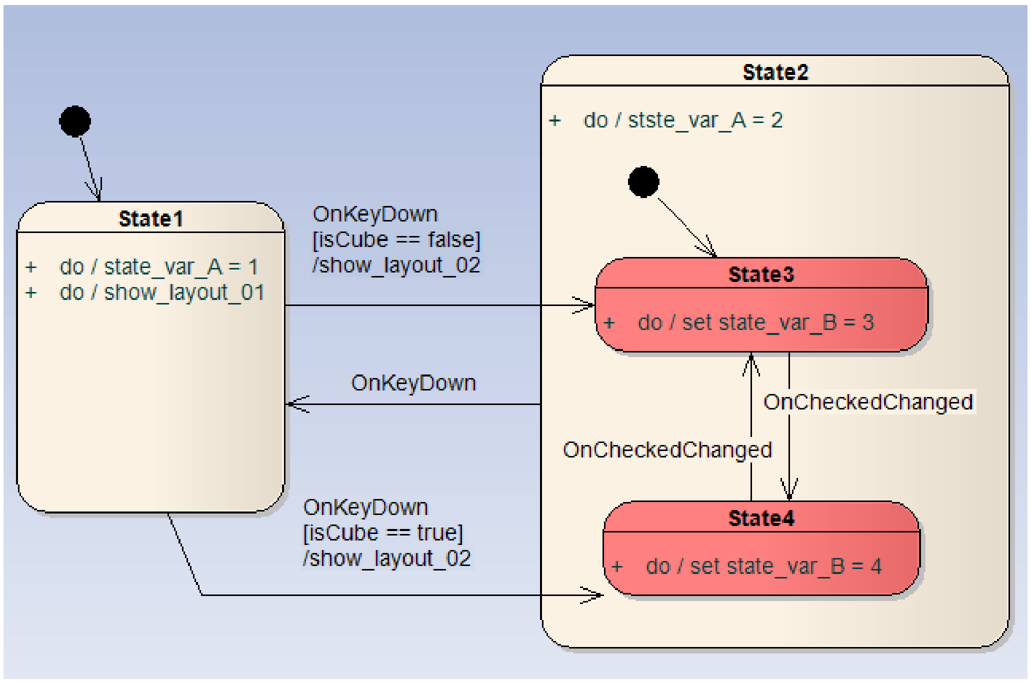


圖 2、 依據 Layout 畫面而設計的狀態機

此時，如果 User 按下<A>鍵，Linux 作業系統偵測到此事件發生了，會傳達訊息給 Android 應用框架(Framwork)，Android 就發出 OnKeyDown 訊息(或稱爲事件)給我們設計的狀態機，就推動狀態機的運轉了。同樣地，當 User 畫面上的 CheckBox 時，Android 就發出 OnCheckedChanged 訊息(或稱爲事件)給狀態機，觸發狀態機的轉移了。

撰寫 Android 程式碼

茲以 Android 程式來實現圖 2 的設計，其原始程式碼如下：

// --------   ac01.java 程式碼   ---------

```
package com.misoo.ppxx;
import android.app.Activity;
import android.os.Bundle;
import android.view.KeyEvent;
import android.widget.CheckBox;
import android.widget.CompoundButton;
import android.widget.LinearLayout;
import android.widget.CompoundButton.OnCheckedChangeListener;

public class ac01 extends Activity implements OnCheckedChangeListener {
        private final int WC = LinearLayout.LayoutParams.WRAP_CONTENT;
        private int state_var_A, state_var_B;
        public boolean isCube;

        @Override protected void onCreate(Bundle icicle) {
            super.onCreate(icicle);
            setContentView(R.layout.main);
            state_var_A = 0;    state_var_B = 0;     isCube = false;
            goto_state_1();
       }
      void goto_state_1() {
           setTitle("State_1");
           state_var_A = 1;
           LinearLayout layout_01 = new LinearLayout(this);
            layout_01.setOrientation(LinearLayout.VERTICAL);
            LinearLayout.LayoutParams param =
                 new LinearLayout.LayoutParams(200, 250);
           param.leftMargin = 1;    param.topMargin = 3;
           if(state_var_B == 0 || state_var_B == 3)     isCube = false;
           else    isCube = true;
           MySurfaceView sv = new MySurfaceView(this);
           sv.setCube(isCube);
           layout_01.addView(sv,param);
           setContentView(layout_01);
          }
        void goto_state_2() {
          setTitle("State_2");    state_var_A = 2;
           goto_state_3();    // default state
          }
        void show_layout_02() {
            LinearLayout.LayoutParams para;
            LinearLayout layout_02 = new LinearLayout(this);
            layout_02.setOrientation(LinearLayout.HORIZONTAL);
            para = new LinearLayout.LayoutParams(200, 250);
```

```
                para.leftMargin = 1;
                para.topMargin = 3;
                MySurfaceView sv = new MySurfaceView(this);
                sv.setCube(isCube);
                layout_02.addView(sv,para);

                CheckBox cx = new CheckBox(this);
                cx.setText("Cube");
                cx.setChecked(isCube);
                cx.setOnCheckedChangeListener(this);
                para = new LinearLayout.LayoutParams(WC, WC);
                para.leftMargin = 30;
                para.topMargin = 30;
                layout_02.addView(cx, para);
                setContentView(layout_02);
              }
          void goto_state_3()   { state_var_A = 2;   state_var_B = 3;   setTitle("State_3"); }
          void goto_state_4()   {    state_var_A = 2; state_var_B = 4;   setTitle("State_4"); }
          @Override public boolean onKeyDown(int keyCode, KeyEvent msg) {
                switch(keyCode) {
                    case KeyEvent.KEYCODE_A:
                        if(state_var_A == 1)
                              if(isCube) {    this.show_layout_02(); goto_state_4();    }
                              else {    this.show_layout_02(); goto_state_3();    }
                        else if(state_var_A == 2)    goto_state_1();
                    break;
                  }
                  return true;
               }
       public void onCheckedChanged(CompoundButton buttonView, boolean isChecked) {
              if(state_var_A == 2){
                     if(state_var_B == 3)    goto_state_4();
                     else if(state_var_B == 4)    goto_state_3();
       }}}
```

以下是 3D 繪圖的程式碼，其詳細的 3D 繪圖原理和技術，請參閱本書的「第 10 章第 1 節：如何繪製 3D 圖形」，有詳細的說明。

// --------   MySurfaceView.java 程式碼   ---------

```
package com.misoo.ppxx;
import android.app.Activity;
import android.content.Context;
import android.util.AttributeSet;
import android.view.SurfaceHolder;
import android.view.SurfaceView;

import javax.microedition.khronos.egl.EGL10;
import javax.microedition.khronos.egl.EGL11;
import javax.microedition.khronos.egl.EGLConfig;
import javax.microedition.khronos.egl.EGLContext;
import javax.microedition.khronos.egl.EGLDisplay;
import javax.microedition.khronos.egl.EGLSurface;
import javax.microedition.khronos.opengles.GL10;

class MySurfaceView extends SurfaceView implements SurfaceHolder.Callback {
               SurfaceHolder mHolder;
               private GLThread mGLThread;
               private MyCube mCube_cb, mCube_pr;
               private float mAngle;
               private boolean isCube;
```

```
        MySurfaceView(Context context) {    super(context);    init();    }
        public void setCube(boolean cb) {    isCube = cb;    }
        public MySurfaceView(Context context, AttributeSet attrs) {
            super(context, attrs);    init();    }
        private void init() {
            mHolder = getHolder();    mHolder.addCallback(this);
            mHolder.setType(SurfaceHolder.SURFACE_TYPE_GPU);
        }
        public void surfaceCreated(SurfaceHolder holder) {
            mGLThread = new GLThread();
            mGLThread.start();
        }
        public void surfaceDestroyed(SurfaceHolder holder) {
            mGLThread.requestExitAndWait();
            mGLThread = null;
        }
        public void surfaceChanged(SurfaceHolder holder, int format, int w, int h) {
            mGLThread.onWindowResize(w, h);
        }
// ----------------------------------------------------------------------
class GLThread extends Thread {
    private boolean mDone;
    private boolean mSizeChanged = true;
    private int mWidth, mHeight;

    GLThread() {
        super();
        mDone = false;    mWidth = 0;    mHeight = 0;
        byte indices_cb[] = {
                0, 7, 3,     7, 0, 4,
                4, 0, 5,     5, 0, 1,
                5, 1, 2,     5, 2, 6,
                6, 2, 7,     7, 2, 3,
                2, 1, 3,     3, 1, 0,
                7, 4, 5,     6, 7, 5,
               };

         byte indices_pr[] = {
                6, 0, 1,   5, 1, 0,
                1, 5, 6,   0, 6, 5,
            };

         mCube_cb = new MyCube(indices_cb);
         mCube_pr = new MyCube(indices_pr);
      }
    @Override    public void run() {
        EGL10 egl = (EGL10)EGLContext.getEGL();
        EGLDisplay dpy = egl.eglGetDisplay(EGL10.EGL_DEFAULT_DISPLAY);
        int[] version = new int[2];
        egl.eglInitialize(dpy, version);
        int[] configSpec = {
                EGL10.EGL_RED_SIZE,        8,
                EGL10.EGL_GREEN_SIZE,      8,
                EGL10.EGL_BLUE_SIZE,       8,
                EGL10.EGL_DEPTH_SIZE,     16,
                EGL10.EGL_NONE
            };
         EGLConfig[] configs = new EGLConfig[1];
         int[] num_config = new int[1];
```

```
            egl.eglChooseConfig(dpy, configSpec, configs, 1, num_config);
            EGLConfig config = configs[0];
            EGLContext glc = egl.eglCreateContext(dpy, config,
                    EGL10.EGL_NO_CONTEXT, null);
            EGLSurface surface = null;
            GL10 gl = null;
            while (!mDone) {
                int w, h;
                boolean changed;
                synchronized(this) {
                    changed = mSizeChanged;
                    w = mWidth;
                    h = mHeight;
                    mSizeChanged = false;
                 }
                if (changed) {
                    surface = egl.eglCreateWindowSurface(dpy, config, mHolder, null);
                    egl.eglMakeCurrent(dpy, surface, surface, glc);
                    gl = (GL10)glc.getGL();
                    gl.glDisable(GL10.GL_DITHER);
                    gl.glHint(GL10.GL_PERSPECTIVE_CORRECTION_HINT,
                                        GL10.GL_FASTEST);
                    gl.glClearColor(1,1,1,1);
                    gl.glEnable(GL10.GL_CULL_FACE);
                    gl.glShadeModel(GL10.GL_SMOOTH);
                    gl.glEnable(GL10.GL_DEPTH_TEST);
                    gl.glViewport(0, 0, w, h);
                    float ratio = (float)w / h;
                    gl.glMatrixMode(GL10.GL_PROJECTION);
                    gl.glLoadIdentity();
                    gl.glFrustumf(-ratio, ratio, -1, 1, 1, 10);
                }
                drawFrame(gl);
                egl.eglSwapBuffers(dpy, surface);
                if (egl.eglGetError() == EGL11.EGL_CONTEXT_LOST) {
                                Context c = getContext();
                                if (c instanceof Activity) {
                                    ((Activity)c).finish();
                        }}}
                        egl.eglMakeCurrent(dpy, EGL10.EGL_NO_SURFACE,
                                EGL10.EGL_NO_SURFACE,
                                EGL10.EGL_NO_CONTEXT);
                        egl.eglDestroySurface(dpy, surface);
                        egl.eglDestroyContext(dpy, glc);
                        egl.eglTerminate(dpy);
    }
  private void drawFrame(GL10 gl) {
    gl.glClear(GL10.GL_COLOR_BUFFER_BIT | L10.GL_DEPTH_BUFFER_BIT);
    gl.glMatrixMode(GL10.GL_MODELVIEW);
    gl.glLoadIdentity();
    gl.glTranslatef(0, 0, -3.0f);
    gl.glRotatef(mAngle,          0, 1, 0);
    gl.glRotatef(mAngle*0.25f,    1, 0, 0);

    gl.glEnableClientState(GL10.GL_VERTEX_ARRAY);
    gl.glEnableClientState(GL10.GL_COLOR_ARRAY);
    mCube_pr.draw(gl);

    gl.glRotatef(mAngle*2.0f, 0, 1, 1);
    gl.glTranslatef(0.5f, 0.5f, 0.5f);
```

```
            if(isCube) mCube_cb.draw(gl);
            mAngle += 1.2f;
        }
        public void onWindowResize(int w, int h) {
                synchronized(this) {
                        mWidth = w;    mHeight = h;
                        mSizeChanged = true;
                }}
        public void requestExitAndWait() {
                mDone = true;
                try {    join();    }
                 catch (InterruptedException ex) { }
       }}}
```

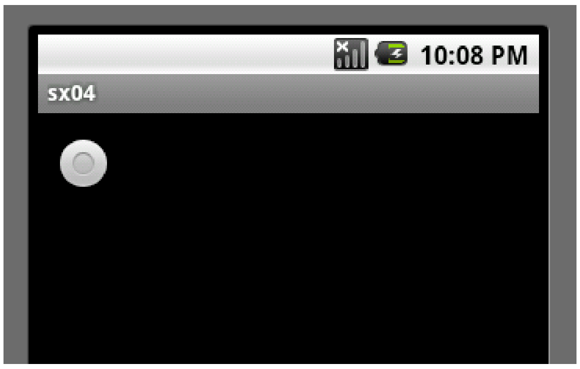
// --------   MyCube.java 程式碼   ---------

```
package com.misoo.ppxx;
import java.nio.ByteBuffer;
import java.nio.ByteOrder;
import java.nio.IntBuffer;
import javax.microedition.khronos.opengles.GL10;

class MyCube {
        private int mTriangles;
        public MyCube(byte [] indices) {
                int one = 0x10000;
                int vertices[] = {
                            -one, -one, -one,
                              one, -one, -one,
                              one,    one, -one,
                            -one,    one, -one,
                            -one, -one,    one,
                              one, -one,    one,
                              one,    one,    one,
                            -one,    one,    one,
                        };

                int colors[] = {
                                 0,        0,        0,    one,
                              one,        0,        0,    one,
                              one,        0,    one,    one,
                                 0,    one,        0,    one,
                                 0,        0,    one,    one,
                              one,    one,        0,    one,
                              one,    one,    one,    one,
                                 0,    one,    one,    one,
                        };

        ByteBuffer vbb = ByteBuffer.allocateDirect(vertices.length*4);
        vbb.order(ByteOrder.nativeOrder());
        mVertexBuffer = vbb.asIntBuffer();
        mVertexBuffer.put(vertices);
        mVertexBuffer.position(0);

        ByteBuffer cbb = ByteBuffer.allocateDirect(colors.length*4);
        cbb.order(ByteOrder.nativeOrder());
        mColorBuffer = cbb.asIntBuffer();
        mColorBuffer.put(colors);
        mColorBuffer.position(0);
        mIndexBuffer = ByteBuffer.allocateDirect(indices.length);
        mIndexBuffer.put(indices);
        mIndexBuffer.position(0);
        mTriangles = indices.length;
        }
    @SuppressWarnings("static-access")
         public void draw(GL10 gl) {
               gl.glFrontFace(GL10.GL_CW);
               gl.glVertexPointer(3, gl.GL_FIXED, 0, mVertexBuffer);
               gl.glColorPointer(4, gl.GL_FIXED, 0, mColorBuffer);
               gl.glDrawElements(gl.GL_TRIANGLES, mTriangles, gl.GL_UNSIGNED_BYTE,
                                              mIndexBuffer);
        }
        private IntBuffer    mVertexBuffer;
        private IntBuffer    mColorBuffer;
        private ByteBuffer    mIndexBuffer;
}
```

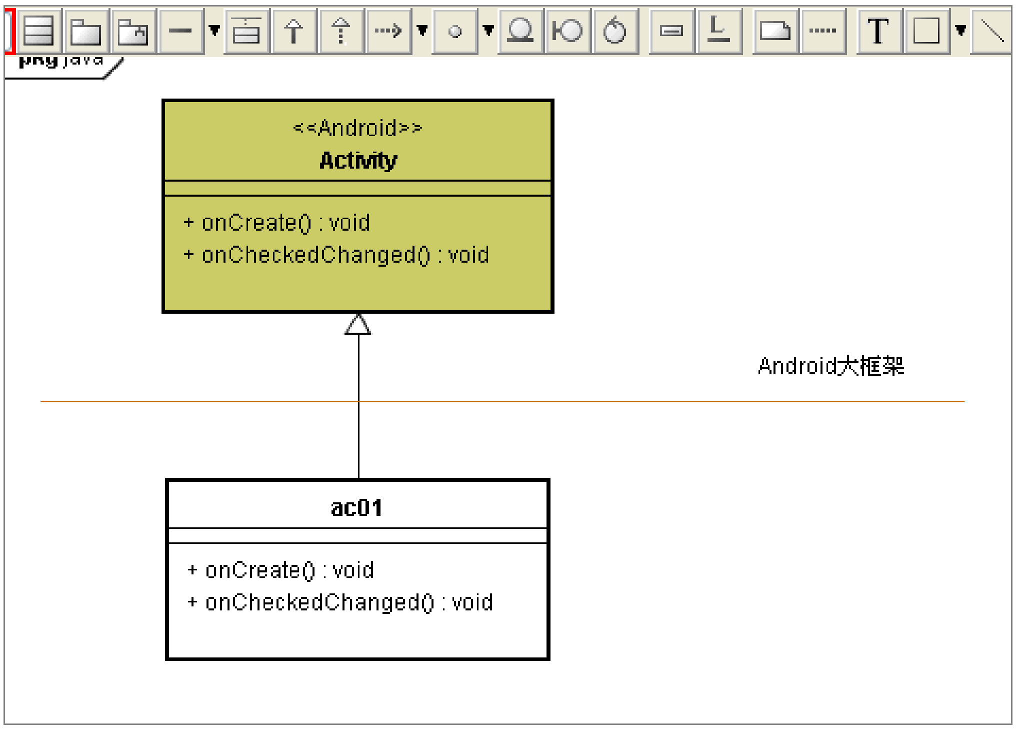
從這個範例中，你可以看到，加上了狀態機，只改變 Activity 類別的結構，對於 3D 繪圖的 MySurfaceView 類別和 MyCube 類別並無任何更動。如此的簡單動作就能將系統與 User 的互動行爲之可靠度提升到極高境界，何樂而不爲呢! ◆

1.5 練習：事件驅動觀點

// 畫面設計



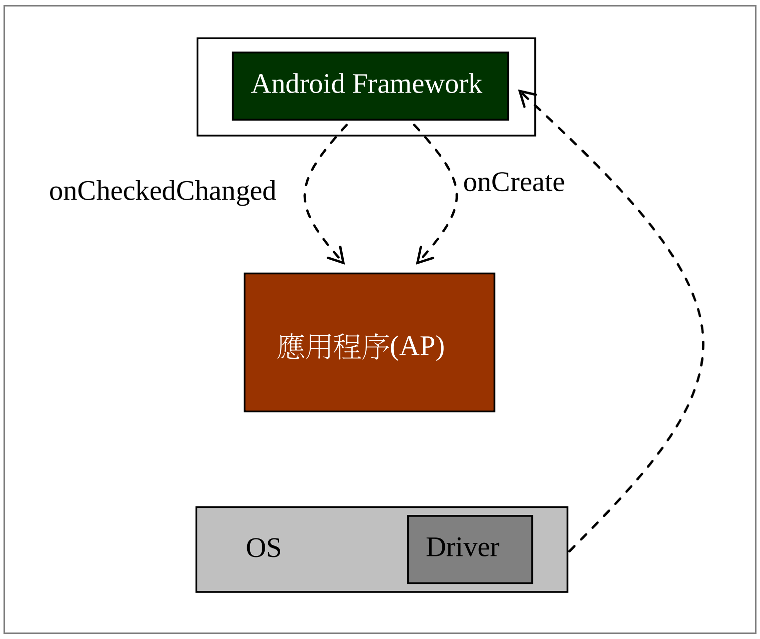
// 類圖設計



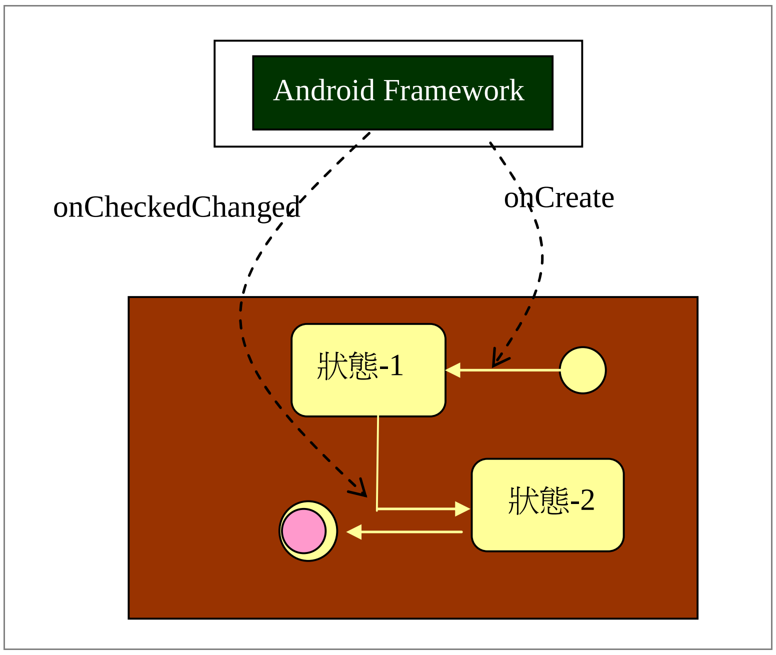
//實現代碼

```java
package com.misoo.ppxx;
import android.app.Activity;
import android.os.Bundle;
import android.widget.CompoundButton;
import android.widget.LinearLayout;
import android.widget.RadioButton;
import android.widget.CompoundButton.OnCheckedChangeListener;

public class ac01 extends Activity implements OnCheckedChangeListener {
	private final int WC = LinearLayout.LayoutParams.WRAP_CONTENT;

	@Override
	public void onCreate(Bundle savedInstanceState) {
		super.onCreate(savedInstanceState);

	LinearLayout layout_01 = new LinearLayout(this);
	layout_01.setOrientation(LinearLayout.VERTICAL);
	RadioButton	ra_btn = new RadioButton(this);
	LinearLayout.LayoutParams param =
		new LinearLayout.LayoutParams(WC, WC);
	param.leftMargin = 10;
	param.topMargin = 10;
	ra_btn.setOnCheckedChangeListener(this);
	layout_01.addView(ra_btn,param);
	setContentView(layout_01);
	}

	@Override
	public void onCheckedChanged(CompoundButton arg0, boolean arg1) {
		finish();
	}
}
```

//事件驅動觀點



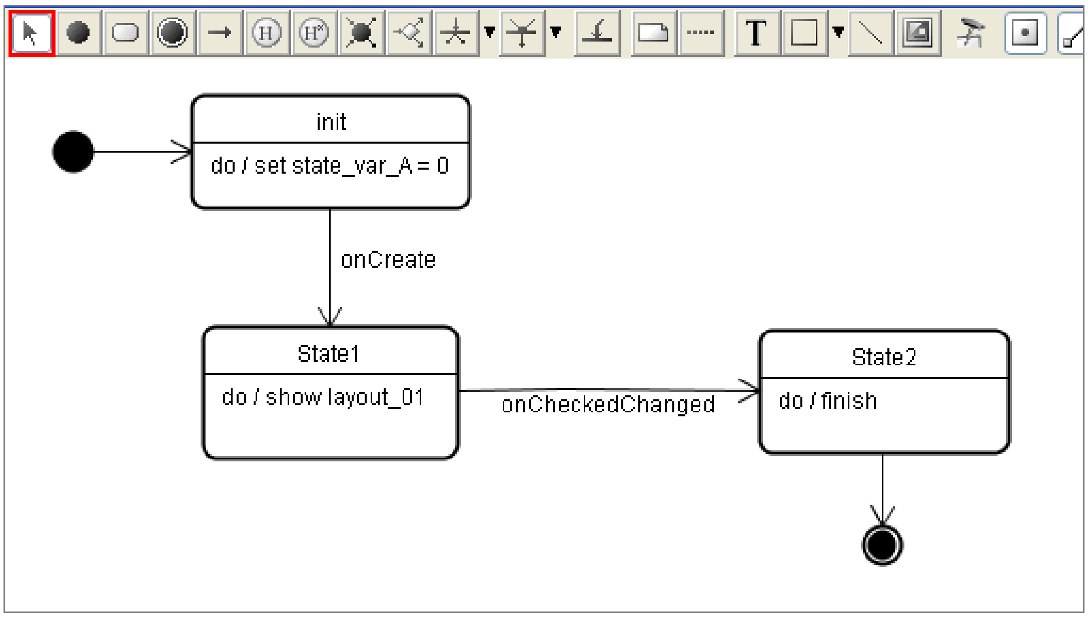
//事件觀點看 AP(使用狀態圖)



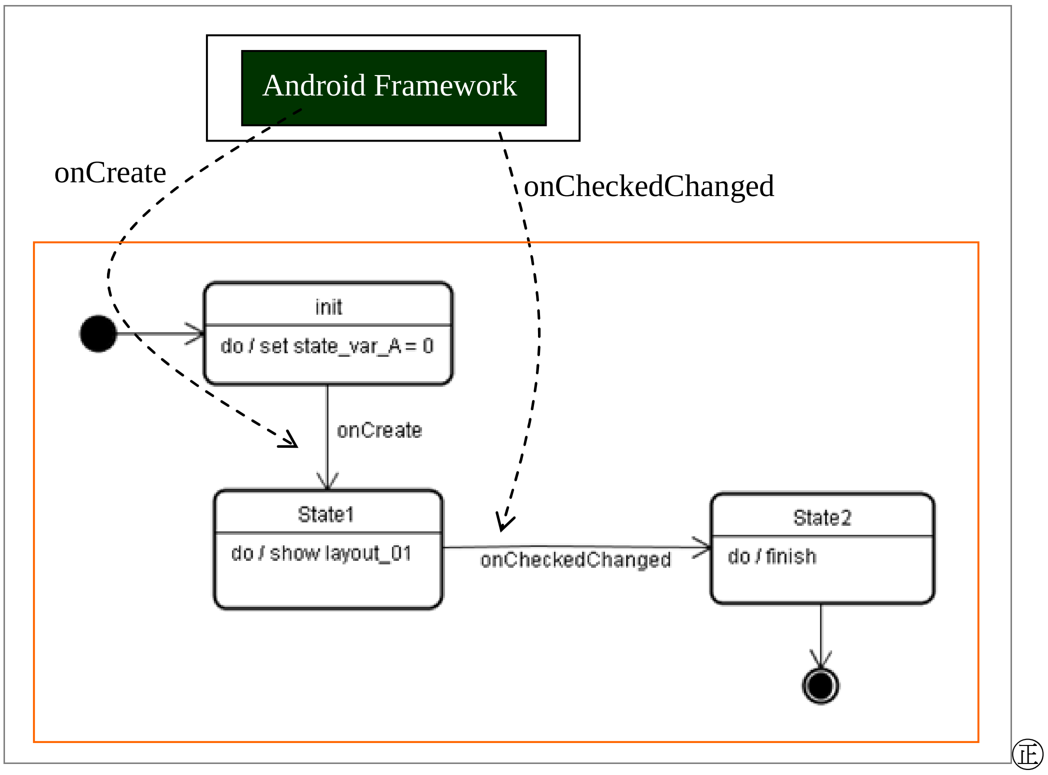
//調整代碼結構

```
package com.misoo.ppxx;
import android.app.Activity;
import android.os.Bundle;
import android.widget.CompoundButton;
import android.widget.LinearLayout;
import android.widget.RadioButton;
import android.widget.CompoundButton.OnCheckedChangeListener;

public class ac01 extends Activity implements OnCheckedChangeListener {
      private final int WC = LinearLayout.LayoutParams.WRAP_CONTENT;
      private int state_var_A = 0;

      @Override
      public void onCreate(Bundle savedInstanceState) {
            super.onCreate(savedInstanceState);
            goto_state_01();
      }

      private void goto_state_01(){
              state_var_A = 1;
                    LinearLayout layout = prepare_Layout_01();
                    setContentView(layout);
      }

      private LinearLayout prepare_Layout_01(){
            LinearLayout layout = new LinearLayout(this);
            layout.setOrientation(LinearLayout.VERTICAL);
            RadioButton    ra_btn = new RadioButton(this);
            LinearLayout.LayoutParams param =
                  new LinearLayout.LayoutParams(WC, WC);
            param.leftMargin = 10;
            param.topMargin = 10;
            ra_btn.setOnCheckedChangeListener(this);
            layout.addView(ra_btn,param);
            return layout;
        }

      private void    goto_state_02(){
            state_var_A = 2;
            finish();
        }

      @Override
      public void onCheckedChanged(CompoundButton arg0, boolean arg1) {
            if(state_var_A == 1)
                    goto_state_02();
      }
}
```

// 以 UML 狀態基表示


init
do / set state_var_A = 0
onCreate
State1
do / show layout_01
onCheckedChanged
State2
do / finish

// 框架的驅動


Android Framework
onCreate
onCheckedChanged
init
do / set state_var_A = 0
onCreate
State1
do / show layout_01
onCheckedChanged
State2
do / finish

~~ END ~~