MICROOH 麦可网

Android-从程序员到架构师之路

出品人：Sundy
讲师：高焕堂（台湾）

http://www.microoh.com

B04_b

SurfaceView的UI多线程(b)

By 高煥堂

2、使用SurfaceView画2D图

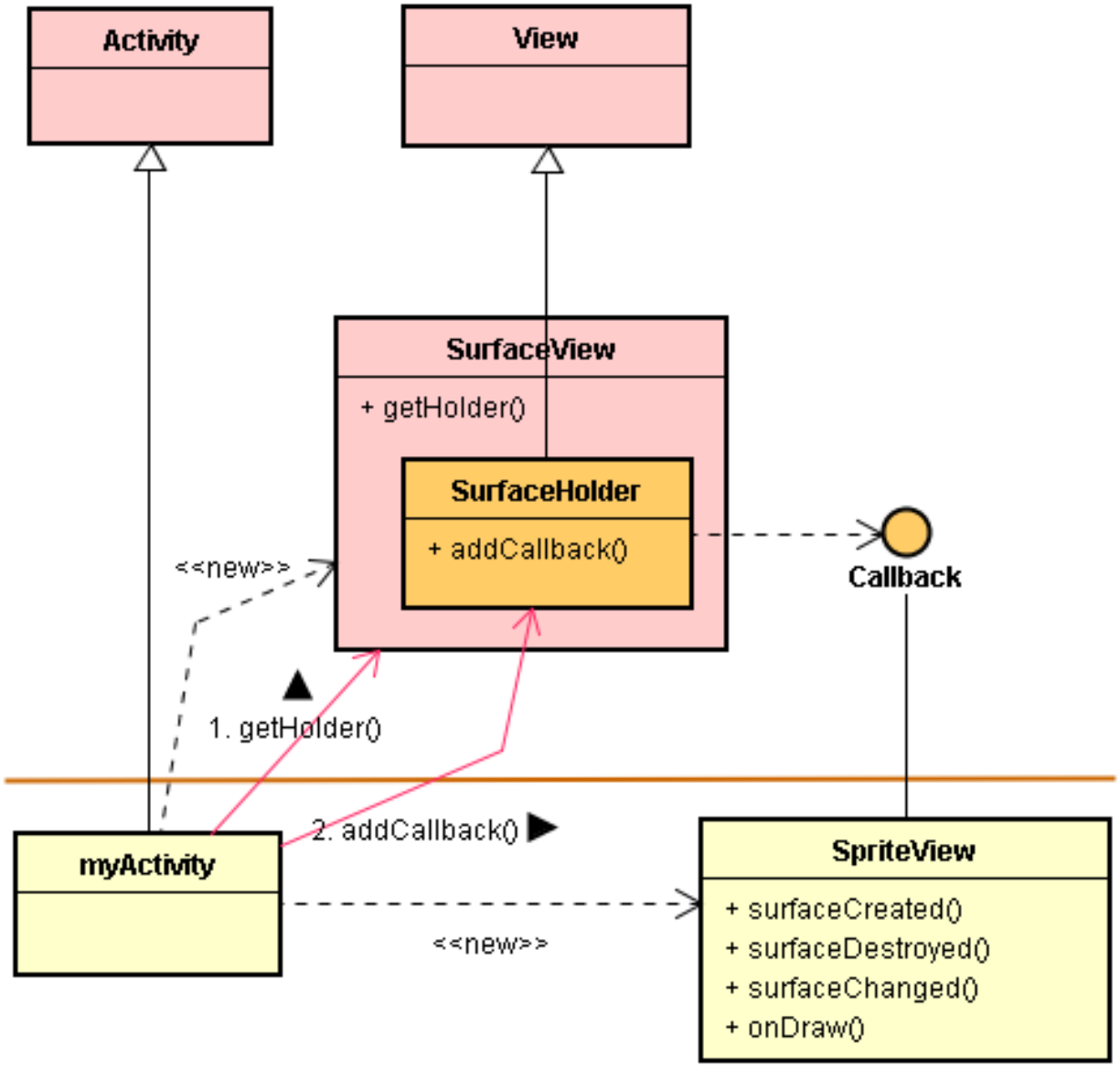
范例一

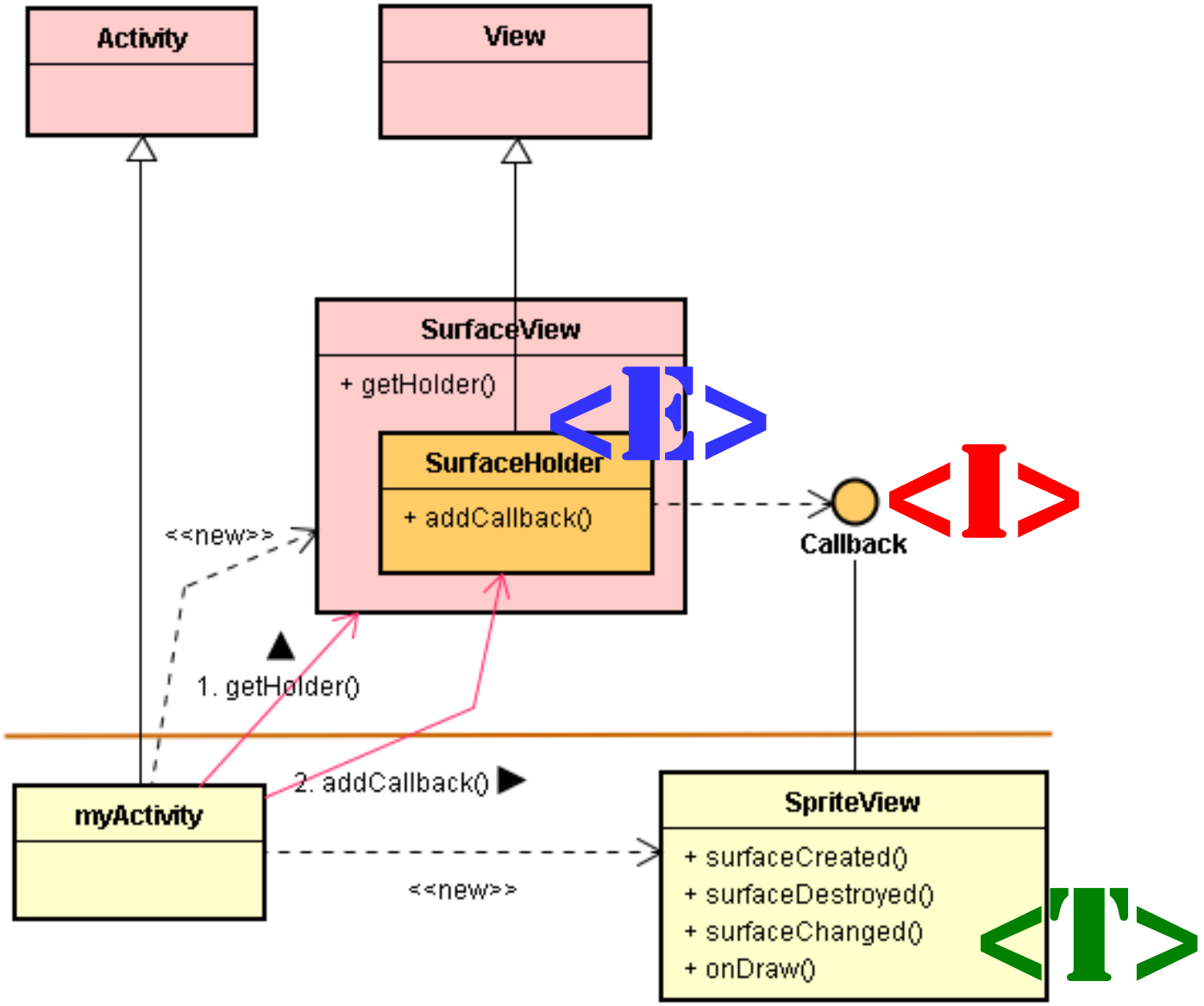
- 以SurfaceView绘出Bitmap图像
- 设计SpriteView类别来实作SurfaceHolder.Callback接口

- 首先来看个简单的程序，显示出一个Bitmap图像。这个图像就构成Sprite动画的基本图形。这个图像如下：

此图像是取自网页的范例：
http://obviam.net/index.php/sprite-animation-with-android/

此范例执行时呈现的画面：


Activity
View
SurfaceView
+ getHolder()
SurfaceHolder
+ addCallback()
Callback
<<new>>
1. getHolder()
2. addCallback()
myActivity
<<new>>
SpriteView
+ surfaceCreated()
+ surfaceDestroyed()
+ surfaceChanged()
+ onDraw()

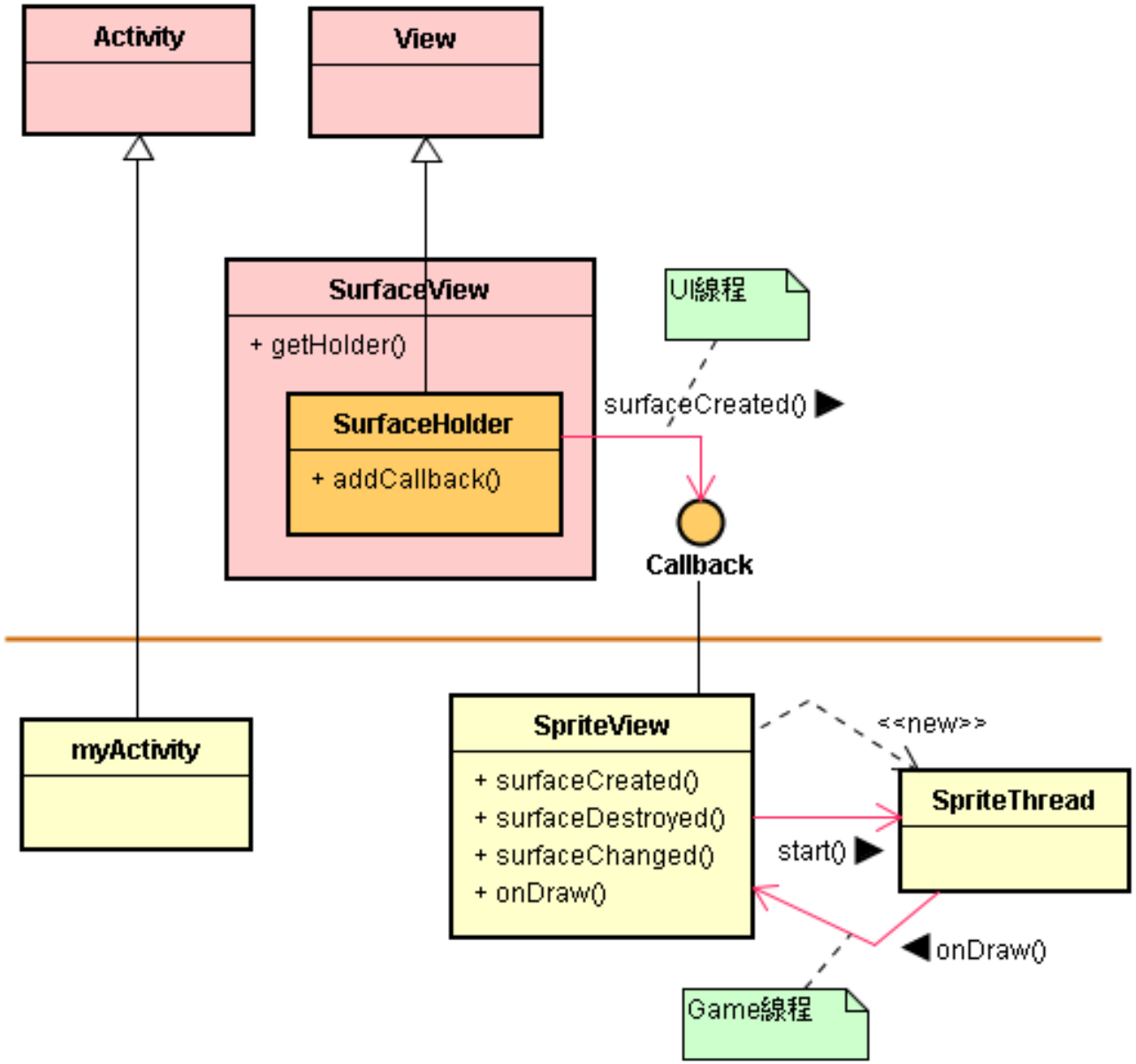

Activity
View
SurfaceView
+ getHolder()
SurfaceHolder
+ addCallback()
<<new>>
Callback
1. getHolder()
2. addCallback()
myActivity
<<new>>
SpriteView
+ surfaceCreated()
+ surfaceDestroyed()
+ surfaceChanged()
+ onDraw()

```
// SpriteView.java
// .........
public class SpriteView implements SurfaceHolder.Callback{
    private SpriteThread sThread;
    private Paint paint;
    private Bitmap bm;
    public SpriteView(Bitmap bmp) {  bm = bmp;  }
    @Override public void surfaceCreated(SurfaceHolder holder) {
            sThread = new SpriteThread(holder, this);
            sThread.start();
       }
    @Override public void surfaceDestroyed(SurfaceHolder holder) {}
    @Override
    public void surfaceChanged(SurfaceHolder holder, int format,
                        int width, int height) {}
    protected void onDraw(Canvas canvas) {
            paint= new Paint();
            canvas.drawColor(Color.WHITE);
            canvas.drawBitmap(bm, 10, 10, paint);
    }
```

```
public class SpriteThread extends Thread{
    private SpriteView mView;
    private SurfaceHolder mHolder;
    private Canvas c;
    SpriteThread(SurfaceHolder h, SpriteView v){
                mHolder = h      mView = v;
     }
    public void run(){
        try{
            c = mHolder.lockCanvas(null);
            synchronized (mHolder){
                mView.onDraw(c);
              }
        }finally{
              if(c!=null)
                mHolder.unlockCanvasAndPost(c);
               }
}}
```

设计GameLoop类别

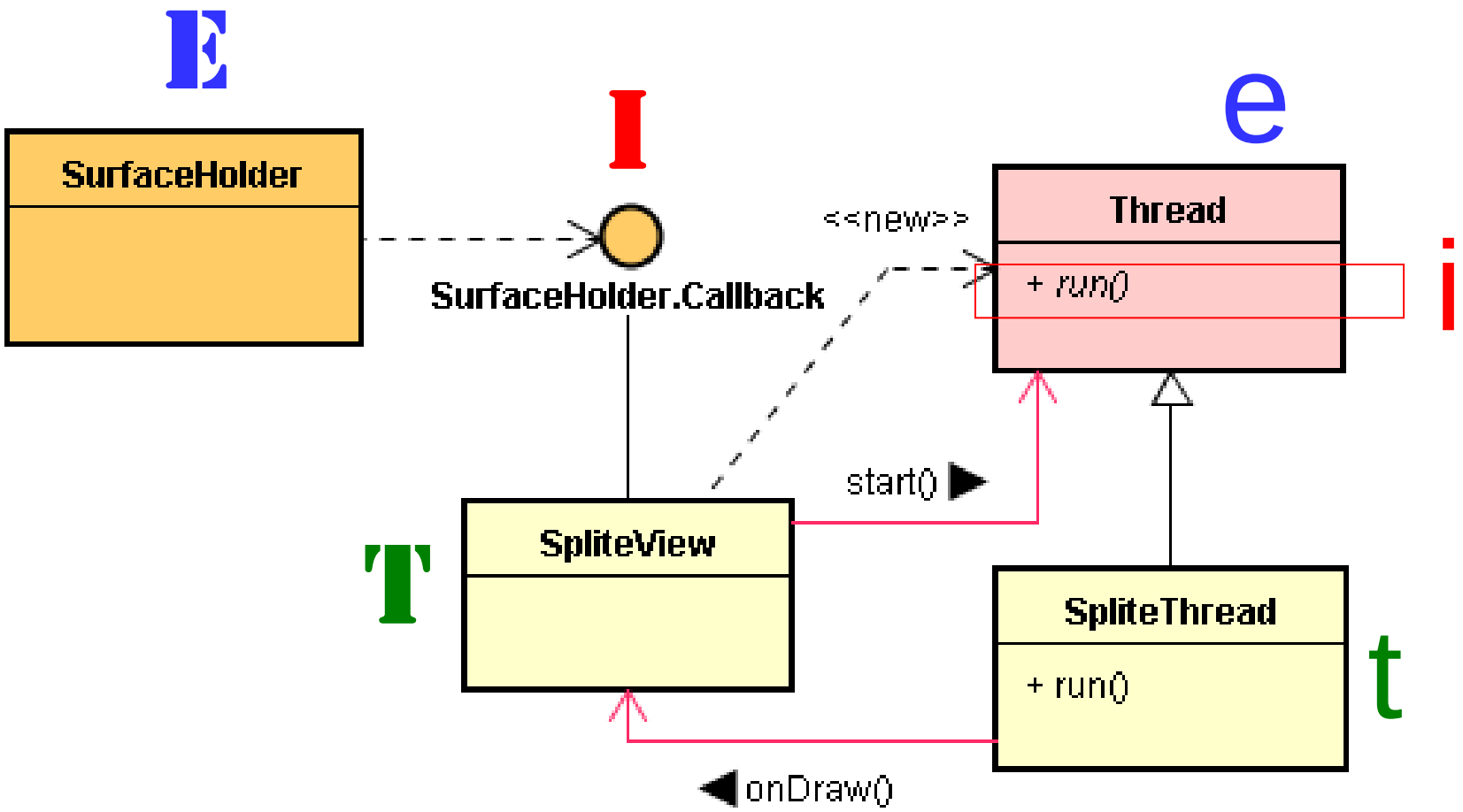
(把小线程移出来)


Activity
View
SurfaceView
+ getHolder()
SurfaceHolder
+ addCallback()
UI線程
surfaceCreated()
Callback
myActivity
SpriteView
+ surfaceCreated()
+ surfaceDestroyed()
+ surfaceChanged()
+ onDraw()
<<new>>
SpriteThread
start()
onDraw()
Game線程

```
// SpriteView.java
// ……..
public class SpriteView implements SurfaceHolder.Callback{
        private SpriteThread sThread;
        private Paint paint;
         private Bitmap bm;

        public SpriteView(Bitmap bmp) {  bm = bmp;  }
        @Override
        public void surfaceCreated(SurfaceHolder holder) {
                sThread = new SpriteThread(holder, this);
                sThread.start();
        }
        @Override
        public void surfaceDestroyed(SurfaceHolder holder)
                { }
```

```
@Override
public void surfaceChanged(SurfaceHolder holder, int format, int width,
                           int height) { }

protected void onDraw(Canvas canvas) {
                paint= new Paint();
                canvas.drawColor(Color.WHITE);
                canvas.drawBitmap(bm, 10, 10, paint);
    }
}
```

```
// SpriteThread.java
//.........
public class SpriteThread extends Thread {
    private SpriteView mView;
    private SurfaceHolder mHolder;
    private Canvas c;
    SpriteThread(SurfaceHolder h, SpriteView v){
            mHolder = h;     mView = v;
        }
    public void run(){
            try{
              c = mHolder.lockCanvas(null);
              synchronized (mHolder){  mView.onDraw(c);  }
            }finally{
              if(c!=null){ mHolder.unlockCanvasAndPost(c); }
            }
 }}
```


E
I
e
SurfaceHolder
SurfaceHolder.Callback
<<new>>
Thread
+ run()
i
start()
T
SpliteView
SpliteThread
+ run()
t
onDraw()

```
// myActivity.java
// ........
public class myActivity extends Activity
                              implements OnClickListener {
        private SurfaceView sv = null;
        private Button ibtn;
        private Bitmap bm;
        private SpriteView spView;

   @Override  protected void onCreate(Bundle icicle) {
      super.onCreate(icicle);
      LinearLayout layout = new LinearLayout(this);
      layout.setOrientation(LinearLayout.VERTICAL);
      sv = new SurfaceView(this);
      bm = BitmapFactory.decodeResource(
                  getResources(), R.drawable.walk_elaine);
      spView = new SpriteView(bm);
      sv.getHolder().addCallback(spView);
```

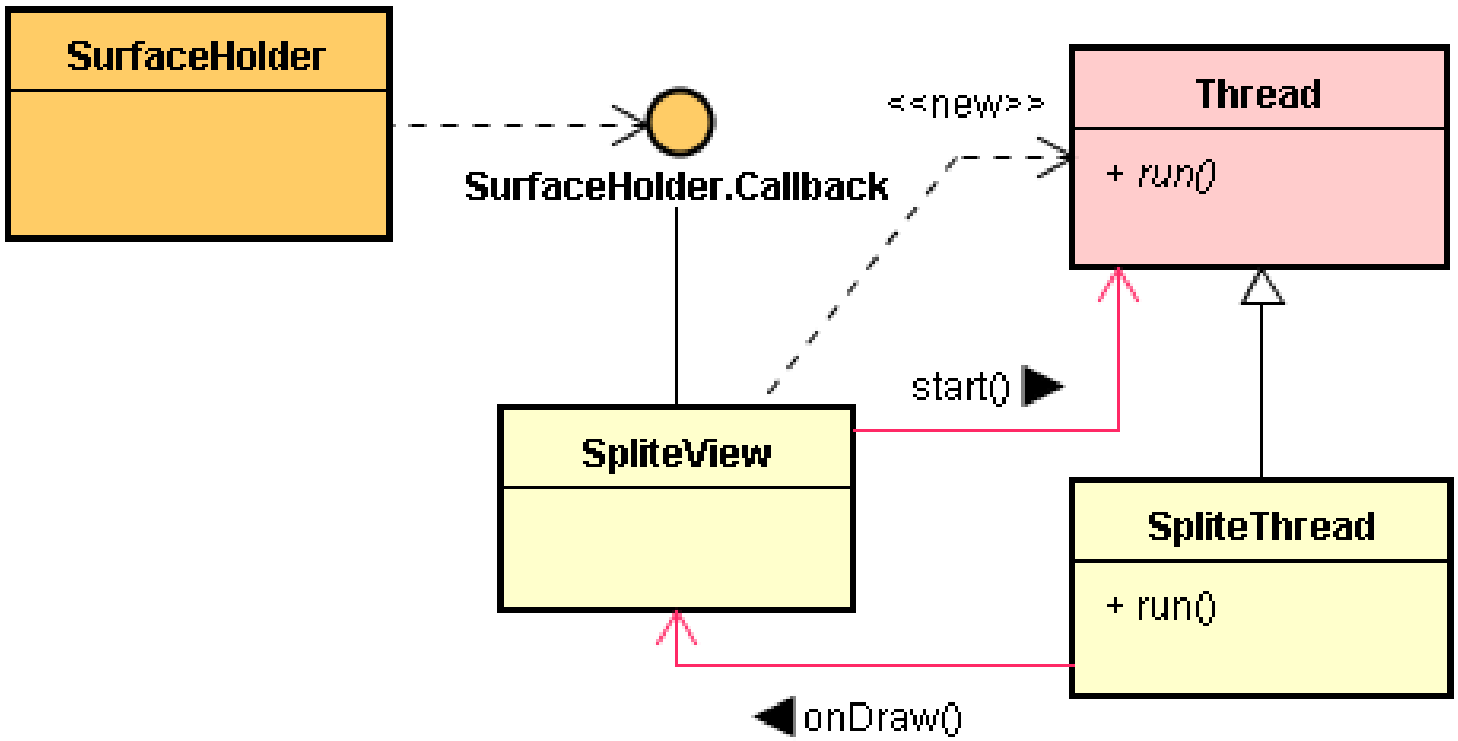
```
        LinearLayout.LayoutParams param =
                new LinearLayout.LayoutParams(200, 200);
        param.topMargin = 10;  param.leftMargin = 10;
        layout.addView(sv, param);
        //-----------------------------------------
        ibtn = new Button(this); ibtn.setOnClickListener(this);
        ibtn.setText("Exit");
        ibtn.setBackgroundResource(R.drawable.gray);
        LinearLayout.LayoutParams param1 =
            new LinearLayout.LayoutParams(200, 65);
        param1.topMargin = 10;  param1.leftMargin = 10;
        layout.addView(ibtn, param1);
        setContentView(layout);
        }
    public void onClick(View v) {  finish();            }
}
```

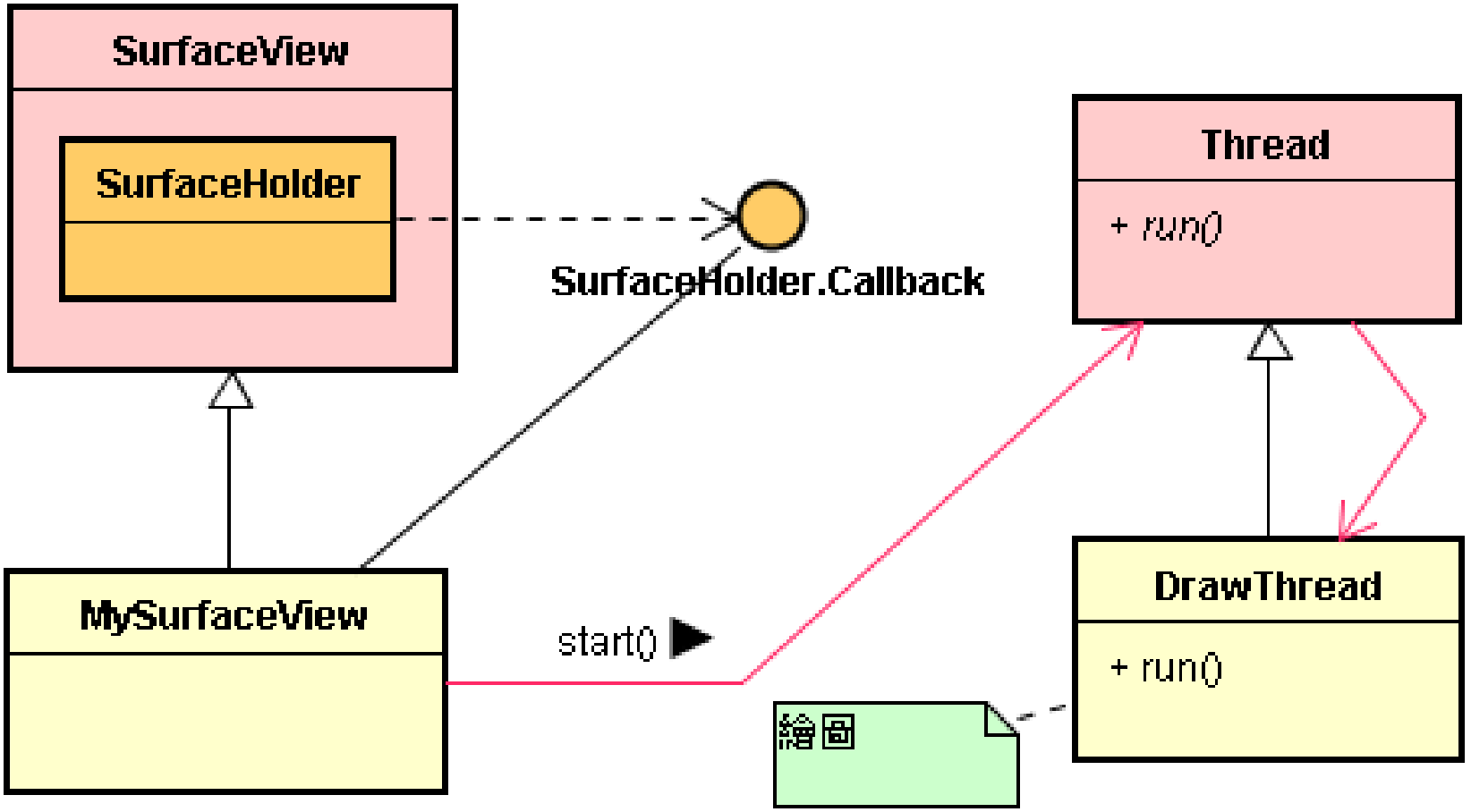
請問：那一條線程執行
SpliteView裡的onDraw()函數呢?

范例二

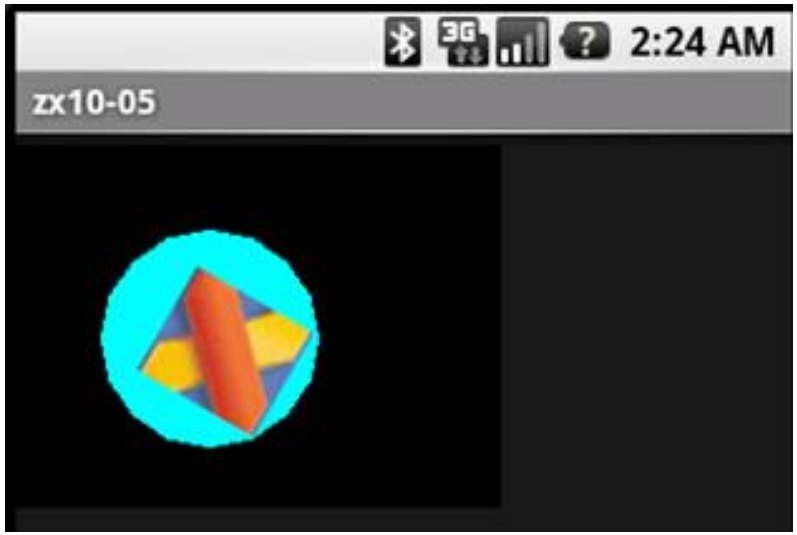
- 让图像在SurfaceView里旋转

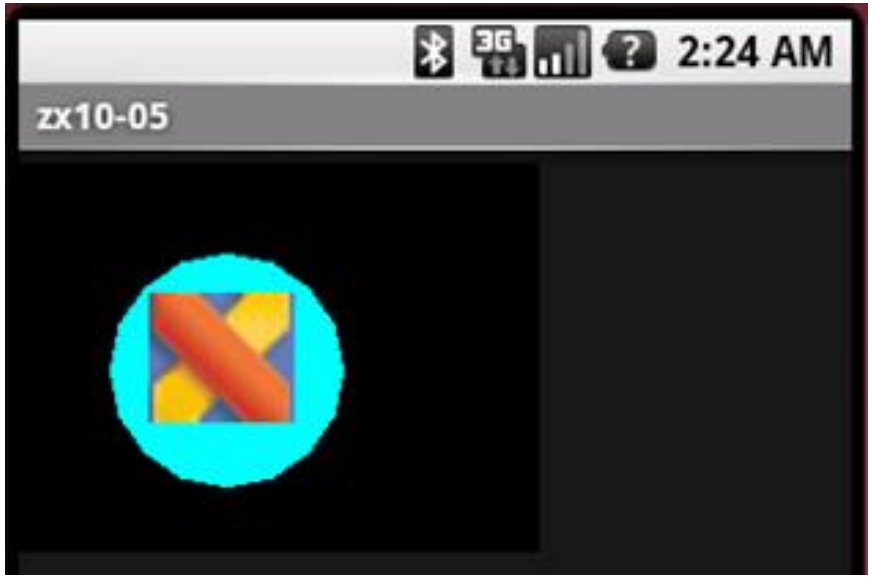
复习：


SurfaceHolder
SurfaceHolder.Callback
<<new>>
Thread
+ run()
start()
SpliteView
SpliteThread
+ run()
onDraw()


SurfaceView
SurfaceHolder
SurfaceHolder.Callback
Thread
+ run()
MySurfaceView
start()
DrawThread
+ run()

- 在MySurfaceView里定义一个DrawThread类，它诞生一个单独的线程来执行画图的任务。
- 当主线程侦测到绘图画面(Surface)被开启时，就会诞生DrawThread对象，启动新线程去画图。
- 一直到主要线程侦测到绘图画面被关闭时，就停此正在绘图的线程。

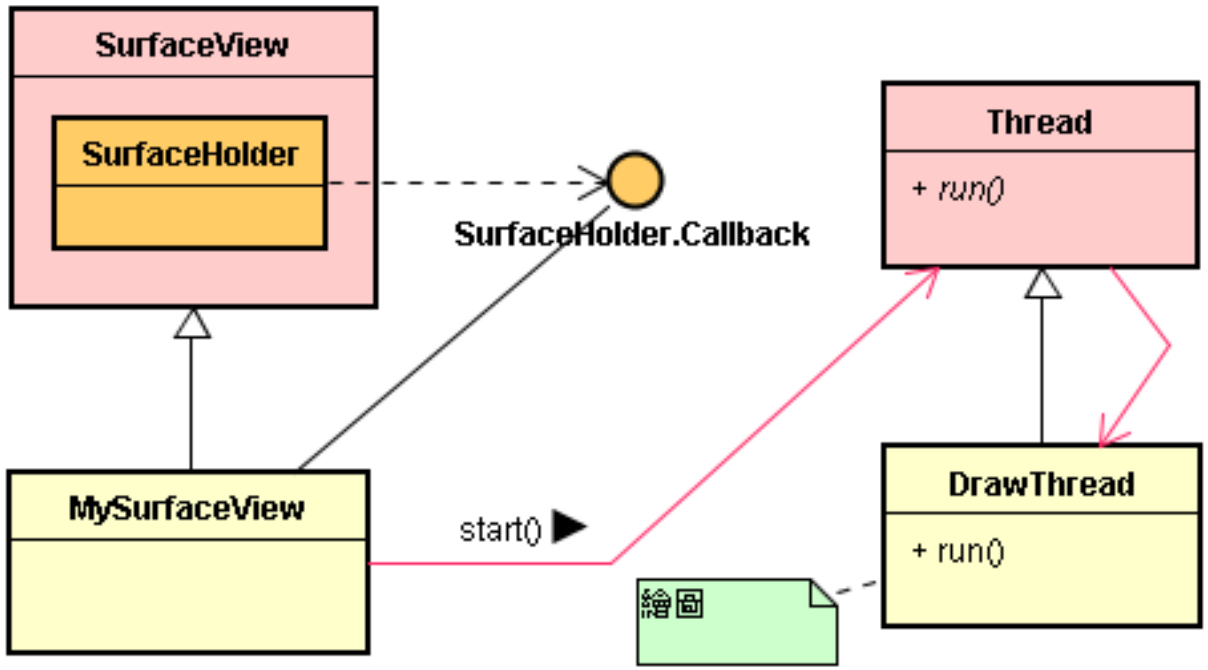

2:24 AM
zx10-05


2:24 AM
zx10-05

```
class MySurfaceView extends SurfaceView
                            implements SurfaceHolder.Callback {
    private SurfaceHolder mHolder;
    private DrawThread mThread;

    MySurfaceView(Context context) {
        super(context);
        getHolder().addCallback(this);
      }

     public void surfaceCreated(SurfaceHolder holder) {
        mHolder = holder;
        mThread = new DrawThread();
        mThread.start();   }
```

```
 public void surfaceDestroyed(SurfaceHolder holder) {
          mThread.finish();
          mThread = null;
        }
public void surfaceChanged(SurfaceHolder holder, int format,
                        int w, int h) { }
class DrawThread extends Thread {
        int degree = 36;
        boolean mFinished = false;

        DrawThread() {  super();  }
        @Override public void run() {
             Bitmap bmp
                  = BitmapFactory.decodeResource(getResources(),
                                        R.drawable.x_xxx);
            Matrix matrix;
            degree = 0;
```

```
        while(!mFinished){
            Paint paint = new Paint();  paint.setColor(Color.CYAN);
            Canvas cavans = mHolder.lockCanvas();
            cavans.drawCircle(80, 80, 45, paint);
            //------ rotate ----------------------------
            matrix = new Matrix();    matrix.postScale(1.5f, 1.5f);
            matrix.postRotate(degree);
            Bitmap newBmp
                    = Bitmap.createBitmap(bmp, 0, 0, bmp.getWidth(),
                                    bmp.getHeight(), matrix, true);
            cavans.drawBitmap(newBmp, 50, 50, paint);
            mHolder.unlockCanvasAndPost(cavans);
            degree += 15;
            try { Thread.sleep(100);
            } catch (Exception e) {}
    }}
    void finish(){ mFinished = true;  }
}}
```


SurfaceView
SurfaceHolder
SurfaceHolder.Callback
Thread
+ run()
MySurfaceView
start()
DrawThread
+ run()

```
// ac01.java-
//……..
public class ac01 extends Activity {
    @Override protected void onCreate(Bundle icicle) {
        super.onCreate(icicle);
        MySurfaceView mv = new MySurfaceView(this);
                LinearLayout layout = new LinearLayout(this);
        layout.setOrientation(LinearLayout.VERTICAL);
        LinearLayout.LayoutParams param =
            new LinearLayout.LayoutParams(200, 150);
        param.topMargin = 5;
        layout.addView(mv, param);
        setContentView(layout);
  }}
```

Thanks...

高煥堂