

# Advanced Graphics

# Graphics2D Quick Review

```
// Entire Screen Display
```

```
@Override
```

```
public void onCreate (Bundle savedInstanceState)
{
    super.onCreate (savedInstanceState);
    requestWindowFeature (Window.FEATURE_NO_TITLE);
    this.getWindow ().setFlags
        (WindowManager.LayoutParams.FLAG_FULLSCREEN,
         WindowManager.LayoutParams.FLAG_FULLSCREEN);

    WindowManager window = getWindowManager ();
    mDisplay = window.getDefaultDisplay ();

    setContentView (new DrawSurface (this));
}
```

# Graphics with View

```
class Screen extends View
{
    public Screen (Context context)
    {
        super (context);
    }

    @Override
    public void onDraw(Canvas canvas)
    {
        Bitmap sprite =
            BitmapFactory.decodeResource (getResources (),
            R.drawable.ball_blue);

        canvas.drawColor (Color.BLACK);
        canvas.drawBitmap (sprite, mDisplay.getWidth () / 2,
            mDisplay.getHeight () / 2, null);
    }
}
```

# Animating Balls and Paddle

13.Code\SimplePaddleGame

# Graphics with a SurfaceView

- SurfaceView
  - Provides a dedicated drawing surface embedded inside a view hierarchy
  - Places the surface at correct location on screen
  - Allows surface formatting such as size change to occur
  - Allows surface access via the SurfaceHolder interface retrieved using `getHolder ()`

# Why SurfaceView?

- Provides a surface for a secondary thread to render to the screen
- IMPORTANT Thread issues
  - All SurfaceView and callback methods are called from the thread running the SurfaceView's window
  - Thread synchronization is necessary
  - Drawing thread can only touch Surface while the Surface exists

# SurfaceHolder.Callback Interface

- **surfaceCreated (SurfaceHolder holder) - called when surface is first created and used to start up the rendering code**
- **surfaceDestroyed (SurfaceHolder holder) - called right before the surface is destroyed**
- **surfaceChanged (SurfaceHolder, int format, int width, int height) - called immediately after any structural changes are made**

# Using a Drawing Thread

- 13.Code\AdvancedGraphics



# Adding Touch Events

```
setFocusable (true);
```

```
@Override public boolean  
    onTouchEvent (MotionEvent event)  
    {  
        x = (int) event.getX();  
        y = (int) event.getY();  
        return true;  
    }
```

# Problem

Modify AdvancedGraphics to create as many balls moving randomly on the screen as possible. Meaning the only real limitation is memory.