WEEK 3 - DEMO CODE

TiledLayer Demo

```
public class TiledLayerDemo extends MIDlet {

    private Display mDisplay;

    protected void destroyApp(boolean unconditional) throws MIDletStateChangeException {
    }

    protected void pauseApp() {
    }

    protected void startApp() throws MIDletStateChangeException {
        mDisplay = Display.getDisplay(this);
        GameCanvasTiledLayerDemo cv = new GameCanvasTiledLayerDemo(false);
        mDisplay.setCurrent(cv);

    }
}

class GameCanvasTiledLayerDemo extends GameCanvas {

    private Image img;
    private TiledLayer backgroundLayer;
    private LayerManager lManager;
    private int cells[] = {
        1, 0, 2,
        2, 1, 0,
        0, 2, 2,
        0, 0, 1,
        2, 2, 2,
        1, 1, 1
    };

    public GameCanvasTiledLayerDemo(boolean suppressKeyEvents) {
        super(suppressKeyEvents);
        lManager = new LayerManager();


        try {
            img = Image.createImage("/background.png");
        } catch (IOException ex) {
            ex.printStackTrace();
        }
```

```
        backgroundLayer = new TiledLayer(3, 6, img, 32, 32);

        for (int i = 0; i < cells.length; i++) {
            int col = i % 3;
            int row = (i - col) / 3;
            backgroundLayer.setCell(col, row, cells[i]);

        }

        lManager.append(backgroundLayer);

        lManager.paint(getGraphics(), 0, 0);
        flushGraphics();
    }
}
```

Animated Sprite Demo

```
public class SpriteDemo extends MIDlet {

    private Display mDisplay;

    protected void destroyApp(boolean unconditional) throws MIDletStateChangeException {
    }

    protected void pauseApp() {
    }

    protected void startApp() throws MIDletStateChangeException {
        mDisplay = Display.getDisplay(this);
        GameCanvasSpriteDemo cv = new GameCanvasSpriteDemo(false);
        new Thread(cv).start();
        mDisplay.setCurrent(cv);

    }
}

class GameCanvasSpriteDemo extends GameCanvas implements Runnable {

    private Image img;
    private Sprite mSprite;
    private LayerManager lManager;

    public GameCanvasSpriteDemo(boolean suppressKeyEvents) {
        super(suppressKeyEvents);
        lManager = new LayerManager();
```

```
        try {
            img = Image.createImage("/spriteImg.png");
        } catch (IOException ex) {
            ex.printStackTrace();
        }

        mSprite = new Sprite(img, 50, 67);
        lManager.append(mSprite);

    }

    public void run() {
        while (true) {

            lManager.paint(getGraphics(), 0, getHeight() - 100);
            flushGraphics();
            mSprite.nextFrame();

            try {
                Thread.sleep(200);
            } catch (InterruptedException ex) {
                ex.printStackTrace();
            }
        }
    }

    public void animateSprite() {
        if (mSprite.getFrame() < 5) {
        }
    }
}
```

Simple animation with different layers and collision

```
** @author George Nguyen
public class BasicGameCanvas extends GameCanvas implements Runnable {

    private Graphics offScreenGraphics;
    private Image background, rockSpriteImg, smithSpriteImg, explodeImg;
    private LayerManager layerManager;
    private TiledLayer grassBackground;
    private Sprite rSprite, sSprite, eSprite;
    private int x = 3, y = 5;
    boolean isCollide = false;

    public BasicGameCanvas(boolean bln) {
        super(bln);
        setFullScreenMode(true);
```

```
}

public void start() throws IOException {
    layerManager = new LayerManager();

    createBackGround();
    createRockSprite();
    createSmithSprite();
    createExplosionSprite();
    Thread t = new Thread(this);
    t.start();
}

public void updateCanvas() {
    Graphics g = getOffScreenGraphics();
    g.setColor(255, 255, 255);
    g.fillRect(0, 0, getWidth(), getHeight());
    g.setColor(0, 0, 0);
    // Get the â€˜off screenâ€™ Graphics object
    //Graphics g = getOffScreenGraphics();
    // Draw the elements
    layerManager.paint(g, 0, 0);

    // Flush to the display
    flushGraphics();
}

private Graphics getOffScreenGraphics() {
    // Re-use the off screen graphics object
    if (offScreenGraphics == null) {
        offScreenGraphics = getGraphics();
    }
    return offScreenGraphics;
}

public void run() {
    while (true) {
        updateCanvas();
        System.out.println("getX:" + rSprite.getX());
        if (rSprite.getX() > getWidth() || rSprite.collidesWith(grassBackground, true)) {
            sSprite.setVisible(true);
            rSprite.setPosition(0, 10);
        }
        System.out.println("x:" + rSprite.getX() + ",y:" + rSprite.getY());
        rSprite.move(x, y);
        createExplosionAnimation();

        try {
            Thread.sleep(100);
```

```
            } catch (InterruptedException ex) {
                ex.printStackTrace();
            }
            if (rSprite.collidesWith(sSprite, true)) {
                isCollide = true;
            }
        }
    }

    //Create Background
    public void createBackGround() throws IOException {
        background = Image.createImage("Background.png");

        grassBackground = new TiledLayer(8, 1, background, 32, 32);

        for (int i = 0; i < 8; i++) {

            grassBackground.setCell(i, 0, i % 8 + 1);
        }
        grassBackground.setPosition(0, getHeight() - grassBackground.getHeight());

        layerManager.append(grassBackground);
    }

    //Create Sprite
    public void createRockSprite() throws IOException {
        rockSpriteImg = Image.createImage("rock.png");
        rSprite = new Sprite(rockSpriteImg, 12, 12);
        rSprite.setPosition(10, 10);
        layerManager.append(rSprite);

    }

    public void createSmithSprite() throws IOException {

        smithSpriteImg = Image.createImage("mrsmith.png");
        sSprite = new Sprite(smithSpriteImg, 30, 30);
        sSprite.setPosition(55, 110);
        layerManager.append(sSprite);
    }

    public void createExplosionSprite() throws IOException {

        explodeImg = Image.createImage("explosion.png");
        eSprite = new Sprite(explodeImg, 75, 75);
        eSprite.setPosition(55, 110);
        eSprite.setVisible(false);
        layerManager.append(eSprite);
    }
```

```
    public void createExplosionAnimation() {
        if (isCollide) {
            if (sSprite.isVisible()) {
                sSprite.setVisible(false);
                eSprite.setPosition(sSprite.getX() - sSprite.getWidth() / 2, sSprite.getY() - sSprite.getHeight() / 2);
                eSprite.setVisible(true);
                eSprite.setFrame(0);
            } else {
                if (eSprite.getFrame() < 11) {
                    eSprite.nextFrame();
                } else {
                    eSprite.nextFrame();
                    eSprite.setVisible(false);
                    isCollide = false;
                    //sSprite.setTransform(Sprite.TRANS_NONE);
                }
            }
        }
    }
}
```

GameDemo MIDlet

** @author George Nguyen

```
public class GameDemo extends MIDlet {

    private BasicGameCanvas cv;
    private Display mDisplay;

    public GameDemo() throws IOException {
        cv = new BasicGameCanvas(false);
    }

    public void startApp() {
        mDisplay = Display.getDisplay(this);
        mDisplay.setCurrent(cv);
        try {
            cv.start();
        } catch (IOException ex) {
            ex.printStackTrace();
        }
    }

    public void pauseApp() {    }
    public void destroyApp(boolean unconditional) {    }
}
```