RAGE
Split-Screen (Viewports)

```
// imports go here, including  import ray.rage.rendersystem.*;

public class MyGame extends VariableFrameRateGame
{ . . .
  private Camera3Pcontroller orbitController1, orbitController2;
  private Action moveFwdActD, moveFwdActE;    // avatar actions
  . . .
  //  set up the window is the same as before
  protected void setupWindow(RenderSystem rs, GraphicsEnvironment ge)
  { rs.createRenderWindow(new DisplayMode(1000, 700, 24, 60), false);
  }
-------------------------------------------------------------------
  //  now we add setting up viewports in the window
  protected void setupWindowViewports(RenderWindow rw)
  { rw.addKeyListener(this);

    Viewport topViewport = rw.getViewport(0);
    topViewport.setDimensions(.51f, .01f, .99f, .49f);          // B,L,W,H
    topViewport.setClearColor(new Color(1.0f, .7f, .7f));

    Viewport botViewport = rw.createViewport(.01f, .01f, .99f, .49f);
    botViewport.setClearColor(new Color(.5f, 1.0f, .5f));
  }
-------------------------------------------------------------------
  //  we need a camera for each viewport
  protected void setupCameras(SceneManager sm, RenderWindow rw)
  { SceneNode rootNode = sm.getRootSceneNode();

    Camera camera = sm.createCamera("MainCamera",
                                    Projection.PERSPECTIVE);
    rw.getViewport(0).setCamera(camera);
    SceneNode cameraN =
        rootNode.createChildSceneNode("MainCameraNode");
    cameraN.attachObject(camera);
    camera.setMode('n');
    camera.getFrustum().setFarClipDistance(1000.0f);

    Camera camera2 = sm.createCamera("MainCamera2",
                                    Projection.PERSPECTIVE);
    rw.getViewport(1).setCamera(camera2);
    SceneNode cameraN2 =
      rootNode.createChildSceneNode("MainCamera2Node");
    cameraN2.attachObject(camera2);
    camera2.setMode('n');
    camera2.getFrustum().setFarClipDistance(1000.0f);
  }
-------------------------------------------------------------------
  protected void setupScene(Engine eng, SceneManager sm)
                                                  throws IOException
  { im = new GenericInputManager();

    // dolphin avatar for player in the top window
    Entity dolphinE = sm.createEntity("dolphin", "dolphinHighPoly.obj");
    dolphinE.setPrimitive(Primitive.TRIANGLES);
    SceneNode dolphinN =
        sm.getRootSceneNode().createChildSceneNode("dolphinNode");
    dolphinN.attachObject(dolphinE);

    // earth avatar for player in the bottom window
    Entity earthE = sm.createEntity("earth", "earth.obj");
    earthE.setPrimitive(Primitive.TRIANGLES);
    SceneNode earthN =
        sm.getRootSceneNode().createChildSceneNode("earthNode");
    earthN.attachObject(earthE);
    earthN.setLocalPosition(-1.0f, 0.0f, 0.0f);
    earthN.setLocalScale(0.2f, 0.2f, 0.2f);
    . . .
    // make manual objects - line axes
    // set up lights as before
    . . .
    setupOrbitCameras(eng, sm);
    setupInputs(sm);
    dolphinN.yaw(Degreef.createFrom(45.0f));
  }
-------------------------------------------------------------------
  protected void setupOrbitCameras(Engine eng, SceneManager sm)
  { SceneNode dolphinN = sm.getSceneNode("dolphinNode");
    SceneNode cameraN = sm.getSceneNode("MainCameraNode");
    Camera camera = sm.getCamera("MainCamera");
    String gpName = im.getFirstGamepadName();
    orbitController1 =
      new Camera3Pcontroller(camera, cameraN, dolphinN, gpName, im);

    SceneNode earthN = sm.getSceneNode("earthNode");
    SceneNode cameraN2 = sm.getSceneNode("MainCamera2Node");
    Camera camera2 = sm.getCamera("MainCamera2");
    String msName = im.getMouseName();
    orbitController2 =
      new Camera3Pcontroller(camera2, cameraN2, earthN, msName, im);
  }
-------------------------------------------------------------------
  protected void setupInputs(SceneManager sm)
  { String kbName = im.getKeyboardName();
    String gpName = im.getFirstGamepadName();
    String msName = im.getMouseName(); System.out.println(msName);
    SceneNode dolphinN =
          getEngine().getSceneManager().getSceneNode("dolphinNode");
    SceneNode earthN =
          getEngine().getSceneManager().getSceneNode("earthNode");

    // movements of the avatars
    // move forward (dolphin)
    moveFwdActD = new MoveForwardAction(dolphinN);
    im.associateAction(gpName,
      net.java.games.input.Component.Identifier.Button._3,
      moveFwdActD, InputManager.INPUT_ACTION_TYPE.REPEAT_WHILE_DOWN);

    // move forward (earth)
    moveFwdActE = new MoveForwardAction(earthN);
    im.associateAction(kbName,
      net.java.games.input.Component.Identifier.Key.D,
      moveFwdActE, InputManager.INPUT_ACTION_TYPE.REPEAT_WHILE_DOWN);
  }
-------------------------------------------------------------------
  protected void update(Engine engine)
  { // build and set both HUDs
    rs = (GL4RenderSystem) engine.getRenderSystem();
    elapsTime += engine.getElapsedTimeMillis();
    elapsTimeSec = Math.round(elapsTime/1000.0f);
    elapsTimeStr = Integer.toString(elapsTimeSec);
    dispStr = "Earth Time = " + elapsTimeStr;
    rs.setHUD(dispStr, 15, 15);
    dispStr = "Dolphin Time = " + elapsTimeStr;
    rs.setHUD2(dispStr, 15, 345);

    // tell the input manager to process the inputs
    im.update(elapsTime);
    orbitController1.updateCameraPosition();
    orbitController2.updateCameraPosition();
  } }
```