## repeat the assignment you are implementing;

Create a solar system. With some randomization in it.

Have the camera be controllable.

Create a Michellen man.

Animate the Michellen man via a bone class structure.

## explain your approach;

For the Solar system I started with an orbital class. This class has a value ready for the object that this object will orbit around.(Execution is not perfect)

For the camera I am simply executing some mathematics.

For the Michellen man I create a bone class to propagate all the transformation matrixes through in a proper way.(This was not perfect)

For animation I only added in blinking as the rest of the code was still not working the way I wanted it too. I hope to improve this for the remedial opertunity.

## describe your code;

Added a number of new member variables too the sphere class:

public Vector3 scalar;

public Vector3 position;

public Vector3 rotationAxisSpeed;//x,y,z

public Vector3 rotationRadiansSpeed;//x,y,z

public Vector3 rotationAxis;//x,y,z

public Vector3 rotationRadians;//x,y,z

Axis values are intended for rotation around yourself whereas Radians were rotations around the position. Scalar is straight forward.

Then I utilize these values for matrix transformations.

Orbital class looks similar but has the added value of:

protected Sphere orbiting;

Which also adds in the transformation matrix of the orbiting’s transform.Translation.

The bone class looks mostly similar too the orbiting class but instead has three new member variables:

Bone parent;

List<Sphere> children = new List<Sphere>();

Vector3 anchor;

If the bone has a parent it will move the parents Transform into itself. However I never got the order here right, then ended up with several messes that did not work, so I went back to a state that kind of worked. The draw method ends by calling each child’s draw, this is to ensure that the order of draws is parent first.

For the blink animation I change the colour of certain vertices in the eye bone’s,

## show (relevant) code snippets;

// Step 1: Study the way the Sphere class is used in Initialize()

// Step 2: Scale the sun uniformly (= the same factor in x, y and z directions) by a factor 2

spheres.Add(sun = new Sphere(Matrix.Identity, Color.Yellow, 30, new Vector3(2f), new Vector3(0f, 0f, 0f), new Vector3(0f, 0f, 0f), Vector3.Zero));

// Step 3: Create an earth Sphere, with radius, distance and color as given in the assignment

spheres.Add(earth = new Orbital(Matrix.Identity, Color.Navy, 30, new Vector3(1f), new Vector3(16f, 0f, 0f), Vector3.Zero, new Vector3(0f, 1f, 0f), sun));

// Step 4: Create 4 other planets: mars, jupiter, saturnus, uranus (radius, distance and color as given)

// Step 5: Randomize the orbital rotation (in the Y plane) relative to the sun for each planet

spheres.Add(new Orbital(Matrix.Identity, Color.Red, 30, new Vector3(.6f), new Vector3(21f, 0f, 0f), Vector3.Zero, new Vector3(0f, (float)(.15 + r.NextDouble() \* .35), 0f), sun));

spheres.Add(new Orbital(Matrix.Identity, Color.Orange, 30, new Vector3(1.7f), new Vector3(27f, 0f, 0f), Vector3.Zero, new Vector3(0f, (float)(.15 + r.NextDouble() \* .35), 0f), sun));

spheres.Add(new Orbital(Matrix.Identity, Color.Khaki, 30, new Vector3(1.6f), new Vector3(36f, 0f, 0f), Vector3.Zero, new Vector3(0f, (float)(.15 + r.NextDouble() \* .35), 0f), sun));

spheres.Add(new Orbital(Matrix.Identity, Color.Cyan, 30, new Vector3(1.5f), new Vector3(43f, 0f, 0f), Vector3.Zero, new Vector3(0f, (float)(.15 + r.NextDouble() \* .35), 0f), sun));

// Step 7: Create the moon (radius, distance and color as given)

spheres.Add(moon = new Orbital(Matrix.Identity, Color.LightGray, 30, new Vector3(0.5f), new Vector3(2f, 0f, 0f), Vector3.Zero, new Vector3(0f, -1.5f, 0f), earth));

// Bonus: Create a bone transform class for spheres, with a parent transform (anchor position for the first bone), orientation and length/scale,

// and let the creation and animation of the spheres be handled by that class.

// Step 11: Create the Michelin man

// Create the body, scales and positions below

// body1, scale: (2.9f, 1.3f, 2.5f), position: (0f, 18.7f, 0f)

bones.Add(skeletonCenter = new Bone(Matrix.Identity, Color.Red, 30, new Vector3(2.9f, 1.3f, 2.5f), new Vector3(0f, 18.7f, 0f), Vector3.Zero, Vector3.Zero));

//bones[0].rotationAxisSpeed = new Vector3(1f, 1f, 0f);

// body2, scale: (3.1f, 1.5f, 2.7f), position: (0f, 20f, 0f)

bones.Add(new Bone(Matrix.Identity, Color.Blue, 30, new Vector3(3.1f, 1.5f, 2.7f), new Vector3(0f, 0f, 0f), Vector3.Zero, Vector3.Zero, bones[bones.Count -1], new Vector3(0f, -1.3f, 0f)));

// body3, scale: (3.0f, 1.5f, 2.6f), position: (0f, 21.5f, 0f)

bones.Add(new Bone(Matrix.Identity, Color.Green, 30, new Vector3(3.0f, 1.5f, 2.6f), new Vector3(0f, 0f, 0f), Vector3.Zero, Vector3.Zero, bones[bones.Count - 1], new Vector3(0f, -1.5f, 0f)));

// body4, scale: (2.7f, 1.3f, 2.4f), position: (0f, 22.8f, 0f)

bones.Add(new Bone(Matrix.Identity, Color.Orange, 30, new Vector3(2.7f, 1.3f, 2.4f), new Vector3(0f, 0f, 0f), Vector3.Zero, Vector3.Zero, bones[bones.Count - 1], new Vector3(0f, -1.3f, 0f)));

// Create the Michelin man Left arm

// Create the upper left arm

// scale: (1.6f, 1.0f, 1.0f), anchor(!) position: (2.3f, 22.8f, 0f)

// rotate upper left arm by -0.3f along the Z axis

bones.Add(new Bone(Matrix.Identity, Color.Purple, 30, new Vector3(1.6f, 1.0f, 1.0f), new Vector3(0, 0f, 0f), Vector3.Zero, Vector3.Zero, skeletonCenter, new Vector3(2.3f, 0f, 0f)));

bones[bones.Count - 1].rotationAxis += new Vector3(0f, 0f, -.3f);

// Create the left elbow

// scale: (1.0f, 0.9f, 0.9f), center position: 1f along the frame of the left upper arm

bones.Add(new Bone(Matrix.Identity, Color.Red, 30, new Vector3(1.0f, 0.9f, 0.9f), new Vector3(0f, 0f, 0f), Vector3.Zero, Vector3.Zero, bones[bones.Count - 1], new Vector3(1f, 0f, 0f)));

// Create the lower left arm

// scale: (1.4f, 0.9f, 0.9f), anchor(!) position: 1f along the frame of the left upper arm (same as elbow)

// rotate lower left arm by 1.6f along the Z axis, relative to the orientation of the upper left arm

bones.Add(new Bone(Matrix.Identity, Color.White, 30, new Vector3(1.4f, 0.9f, 0.9f), new Vector3(0f, 0f, 0f), Vector3.Zero, Vector3.Zero, bones[bones.Count - 2], new Vector3(1.4f, 0.9f, 0.9f)));

bones[bones.Count - 1].rotationAxis += new Vector3(0f, 0f, -1.6f);

// Create the left hand

// scale: (1.4f, 0.9f, 0.9f), anchor(!) position: 0.6f along the frame of the lower left arm

// rotate left hand by 0.1f along the Z axis, relative to the orientation of the lower left arm

bones.Add(new Bone(Matrix.Identity, Color.RosyBrown, 30, new Vector3(1.4f, 0.9f, 0.9f), new Vector3(0f, 0f, 0f), Vector3.Zero, Vector3.Zero, bones[bones.Count - 1], new Vector3(0.6f, 0f, 0f)));

bones[bones.Count - 1].rotationAxis += new Vector3(0f, 0f, .1f);

// Bonus: Create the Michelin man Right arm (mirror the left one's positions and rotations)

bones.Add(new Bone(Matrix.Identity, Color.Purple, 30, new Vector3(1.6f, 1.0f, 1.0f), new Vector3(0, 0f, 0f), Vector3.Zero, Vector3.Zero, skeletonCenter, new Vector3(-2.3f, 0f, 0f)));

bones[bones.Count - 1].rotationAxis += new Vector3(0f, 0f, .3f);

bones.Add(new Bone(Matrix.Identity, Color.Red, 30, new Vector3(1.0f, 0.9f, 0.9f), new Vector3(0f, 0f, 0f), Vector3.Zero, Vector3.Zero, bones[bones.Count - 1], new Vector3(-1f, 0f, 0f)));

bones.Add(new Bone(Matrix.Identity, Color.White, 30, new Vector3(1.4f, 0.9f, 0.9f), new Vector3(0f, 0f, 0f), Vector3.Zero, Vector3.Zero, bones[bones.Count - 2], new Vector3(-1.4f, 0.9f, 0.9f)));

bones[bones.Count - 1].rotationAxis += new Vector3(0f, 0f, 1.6f);

bones.Add(new Bone(Matrix.Identity, Color.RosyBrown, 30, new Vector3(1.4f, 0.9f, 0.9f), new Vector3(0f, 0f, 0f), Vector3.Zero, Vector3.Zero, bones[bones.Count - 1], new Vector3(-0.6f, 0f, 0f)));

bones[bones.Count - 1].rotationAxis += new Vector3(0f, 0f, .1f);

// Create the Michelin man Left leg

// Create the upper left leg

// scale: (2.0f, 1.7f, 1.7f), anchor(!) position: (1.4f, 17.5f, 0f)

// rotate upper left leg by -0.7f along the Y axis, -1.5f along the Y axis, -0.2f along the X axis.

bones.Add(new Bone(Matrix.Identity, Color.Green, 30, new Vector3(2.0f, 1.7f, 1.7f), new Vector3(0f, 0f, 0f), Vector3.Zero, Vector3.Zero, bones[3], -new Vector3(1.4f, 2f, 0f)));

bones[bones.Count - 1].rotationAxis += new Vector3(-.7f, -1.5f, -.2f);

// Create the left knee

// scale: (1.3f, 1.3f, 1.3f), center position: 1f along the frame of the left upper leg

bones.Add(new Bone(Matrix.Identity, Color.White, 30, new Vector3(1.3f, 1.3f, 1.3f), new Vector3(0f, 0f, 0f), Vector3.Zero, Vector3.Zero, bones[bones.Count - 1], -new Vector3(0f, 1f, 0f)));

// Create the lower left leg

// scale: (2.0f, 1.5f, 1.5f), anchor(!) position: 1f along the frame of the left upper leg (same as knee)

// rotate lower left leg by 1.4f along the X axis, relative to the orientation of the upper left leg

bones.Add(new Bone(Matrix.Identity, Color.White, 30, new Vector3(2.0f, 1.5f, 1.5f), new Vector3(0f, 0f, 0f), Vector3.Zero, Vector3.Zero, bones[bones.Count - 1], -new Vector3(0f, 1f, 0f)));

bones[bones.Count - 1].rotationAxis += new Vector3(0f, 0f, -.7f);

// Create the left foot

// scale: (1.8f, 1.0f, 0.7f), anchor(!) position: 0.6f along the frame of the lower left leg

// rotate left foot by -1.4f along the X axis, relative to the orientation of the lower left leg

bones.Add(new Bone(Matrix.Identity, Color.White, 30, new Vector3(1.8f, 1.0f, 0.7f), new Vector3(0f, 0f, 0f), Vector3.Zero, Vector3.Zero, bones[bones.Count - 1], -new Vector3(0f, 1f, 0f)));

bones[bones.Count - 1].rotationAxis += new Vector3(-1.4f, 0f, 0f);

// Bonus: Create the Michelin man Right leg (mirror the left one's positions and rotations)

bones.Add(new Bone(Matrix.Identity, Color.White, 30, new Vector3(2.0f, 1.7f, 1.7f), new Vector3(0f, 0f, 0f), Vector3.Zero, Vector3.Zero, bones[3], -new Vector3(-1.4f, 2f, 0f)));

bones[bones.Count - 1].rotationAxis += new Vector3(1.3f, -1.5f, -.2f);

bones.Add(new Bone(Matrix.Identity, Color.White, 30, new Vector3(1.3f, 1.3f, 1.3f), new Vector3(0f, 0f, 0f), Vector3.Zero, Vector3.Zero, bones[bones.Count - 1], -new Vector3(0f, 1f, 0f)));

bones.Add(new Bone(Matrix.Identity, Color.White, 30, new Vector3(2.0f, 1.5f, 1.5f), new Vector3(0f, 0f, 0f), Vector3.Zero, Vector3.Zero, bones[bones.Count - 1], -new Vector3(0f, 1f, 0f)));

bones[bones.Count - 1].rotationAxis += new Vector3(0f, 0f, .7f);

bones.Add(new Bone(Matrix.Identity, Color.White, 30, new Vector3(1.8f, 1.0f, 0.7f), new Vector3(0f, 0f, 0f), Vector3.Zero, Vector3.Zero, bones[bones.Count - 1], -new Vector3(0f, 1f, 0f)));

bones[bones.Count - 1].rotationAxis += new Vector3(.6f, 0f, 0f);

// Create the Michelin man Neck and head

// neck, scale: (1.4f, 1.2f, 1.3f), position: (0f, 24f, 0f)

bones.Add(new Bone(Matrix.Identity, Color.White, 30, new Vector3(1.4f, 1.2f, 1.3f), new Vector3(0f, 0f, 0f), Vector3.Zero, Vector3.Zero, skeletonCenter, new Vector3(0, 2f, 0f)));

// head1, scale: (2.0f, 1.2f, 1.7f), position: (0f, 25f, 0f)

bones.Add(new Bone(Matrix.Identity, Color.White, 30, new Vector3(2.0f, 1.2f, 1.7f), new Vector3(0f, 0f, 0f), Vector3.Zero, Vector3.Zero, bones[bones.Count - 1],new Vector3(0, 1f, 0f)));

// head2, scale: (1.4f, 1.4f, 1.4f), position: (0f, 25.8f, 0f)

bones.Add(new Bone(Matrix.Identity, Color.White, 30, new Vector3(1.4f, 1.4f, 1.4f), new Vector3(0f, 0f, 0f), Vector3.Zero, Vector3.Zero, bones[bones.Count - 1],new Vector3(0, .8f, 0f)));

// Bonus: Give the Michelin man eyes

bones.Add(new Bone(Matrix.Identity, Color.Black, 30, new Vector3(.4f, .4f, .4f), new Vector3(0f, 0f, 0f), Vector3.Zero, Vector3.Zero, bones[bones.Count - 1], new Vector3(.5f, .4f, -1f)));

//bones[bones.Count - 1].rotationRadiansSpeed += new Vector3(0f, 0f, .3f);

bones.Add(new Bone(Matrix.Identity, Color.Black, 30, new Vector3(.4f, .4f, .4f), new Vector3(0f, 0f, 0f), Vector3.Zero, Vector3.Zero, bones[bones.Count - 2], new Vector3(-.5f, .4f, -1f)));

//bones[bones.Count - 1].rotationAxisSpeed += new Vector3(0f, 0f, -.3f);

Setup of ALL the spheres.

float deltatime = (float)gameTime.ElapsedGameTime.TotalSeconds;

if (Keyboard.GetState().IsKeyDown(Keys.Right))

{

// Step 10: Make the camera position rotate around the origin depending on gameTime.ElapsedGameTime.TotalSeconds

camRad -= deltatime;

cameraPosition.X = (float)Math.Cos(camRad) \* camdist;

cameraPosition.Z = (float)Math.Sin(camRad) \* camdist;

SetupCamera();

}

else if (Keyboard.GetState().IsKeyDown(Keys.Left))

{

// Step 10: Make the camera position rotate around the origin depending on gameTime.ElapsedGameTime.TotalSeconds

camRad += deltatime;

cameraPosition.X = (float)Math.Cos(camRad) \* camdist;

cameraPosition.Z = (float)Math.Sin(camRad) \* camdist;

SetupCamera();

}

else if (Keyboard.GetState().IsKeyDown(Keys.Up))

{

// Step 10: Make the camera position rotate around the origin depending on gameTime.ElapsedGameTime.TotalSeconds

camdist += deltatime \* 100;

cameraPosition.X = (float)Math.Cos(camRad) \* camdist;

cameraPosition.Z = (float)Math.Sin(camRad) \* camdist;

SetupCamera();

}

else if (Keyboard.GetState().IsKeyDown(Keys.Down))

{

// Step 10: Make the camera position rotate around the origin depending on gameTime.ElapsedGameTime.TotalSeconds

camdist -= deltatime \* 100;

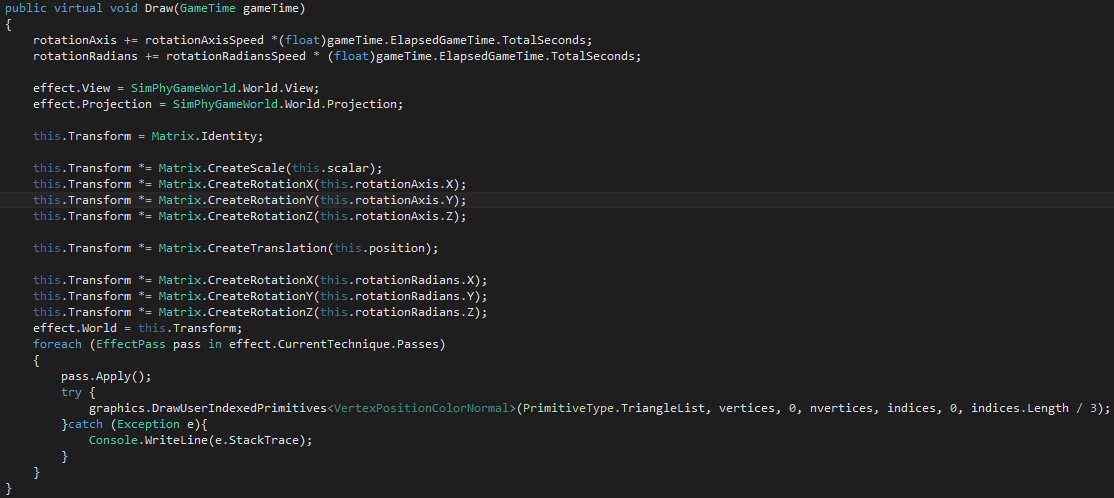
cameraPosition.X = (float)Math.Cos(camRad) \* camdist;

cameraPosition.Z = (float)Math.Sin(camRad) \* camdist;

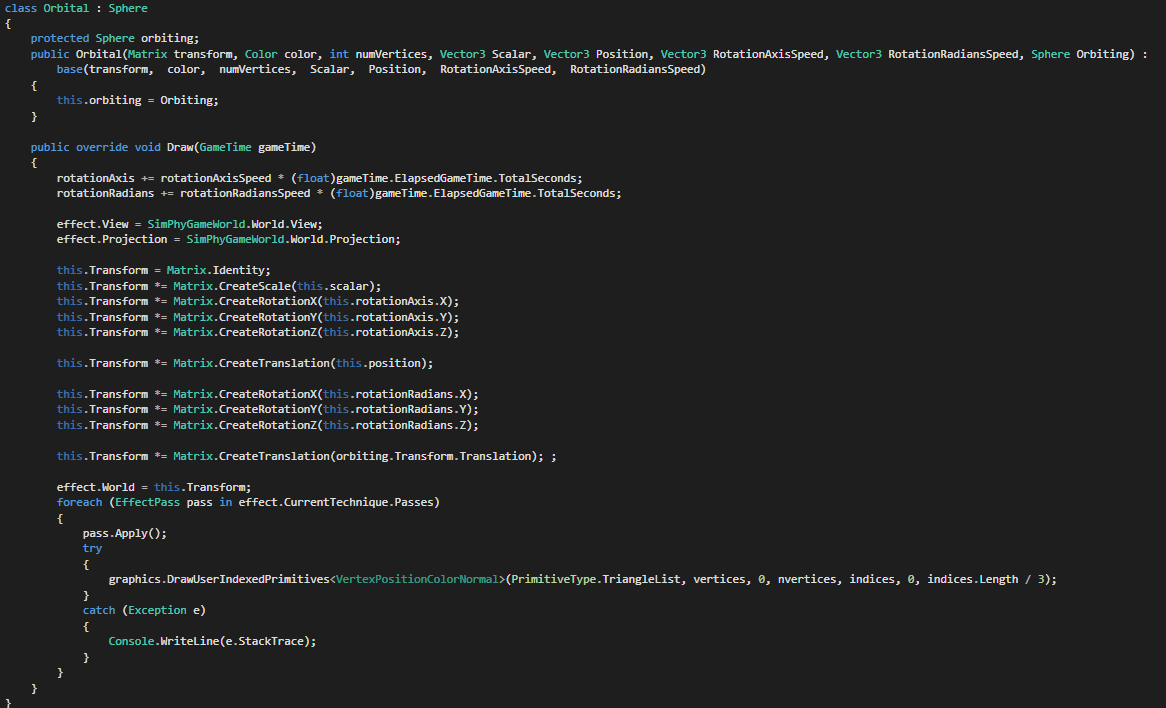
SetupCamera();

}

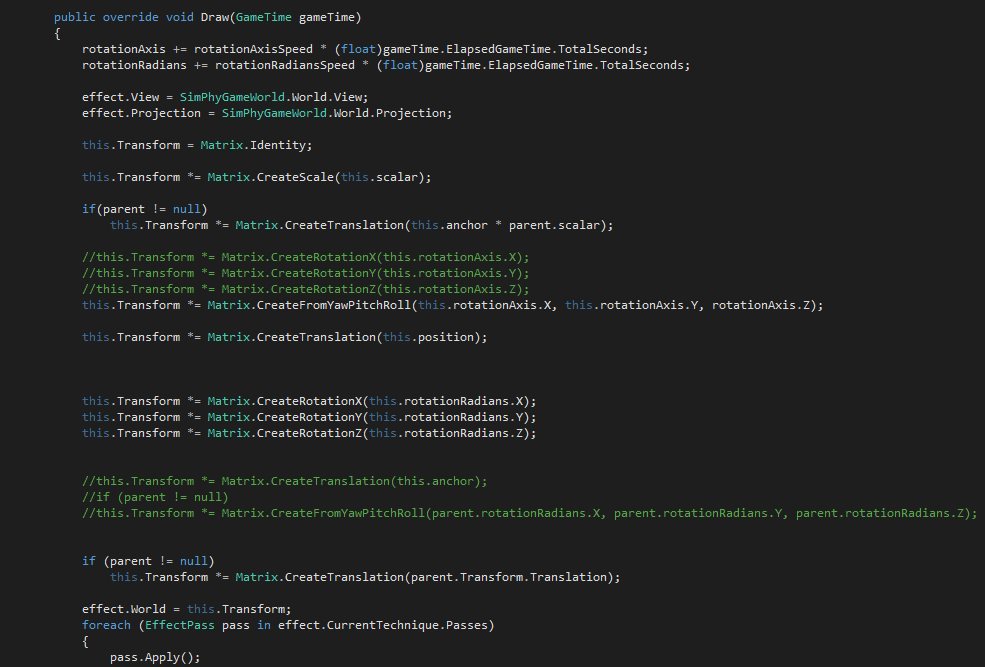
Camera movement



Sphere class draw



Entire orbital class(Bone class is very similar so only have the draw shown)



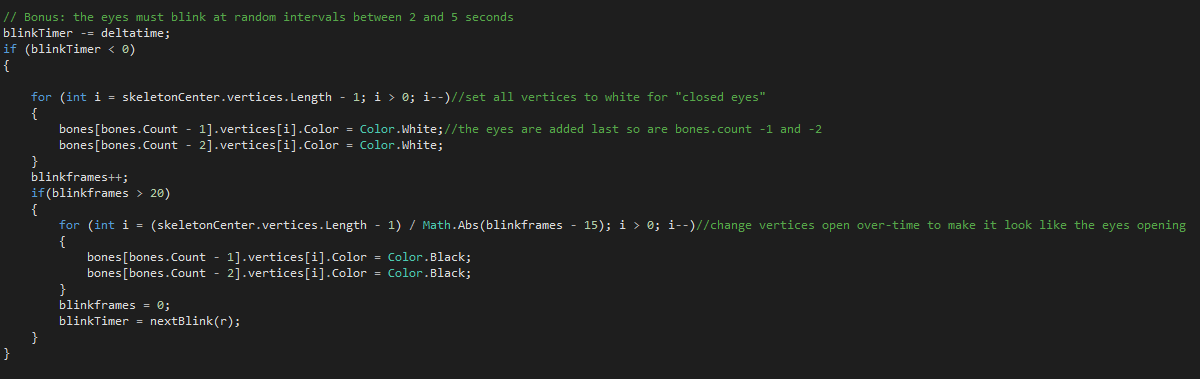
foreach(Bone child in children)

{

child.Draw(gameTime);

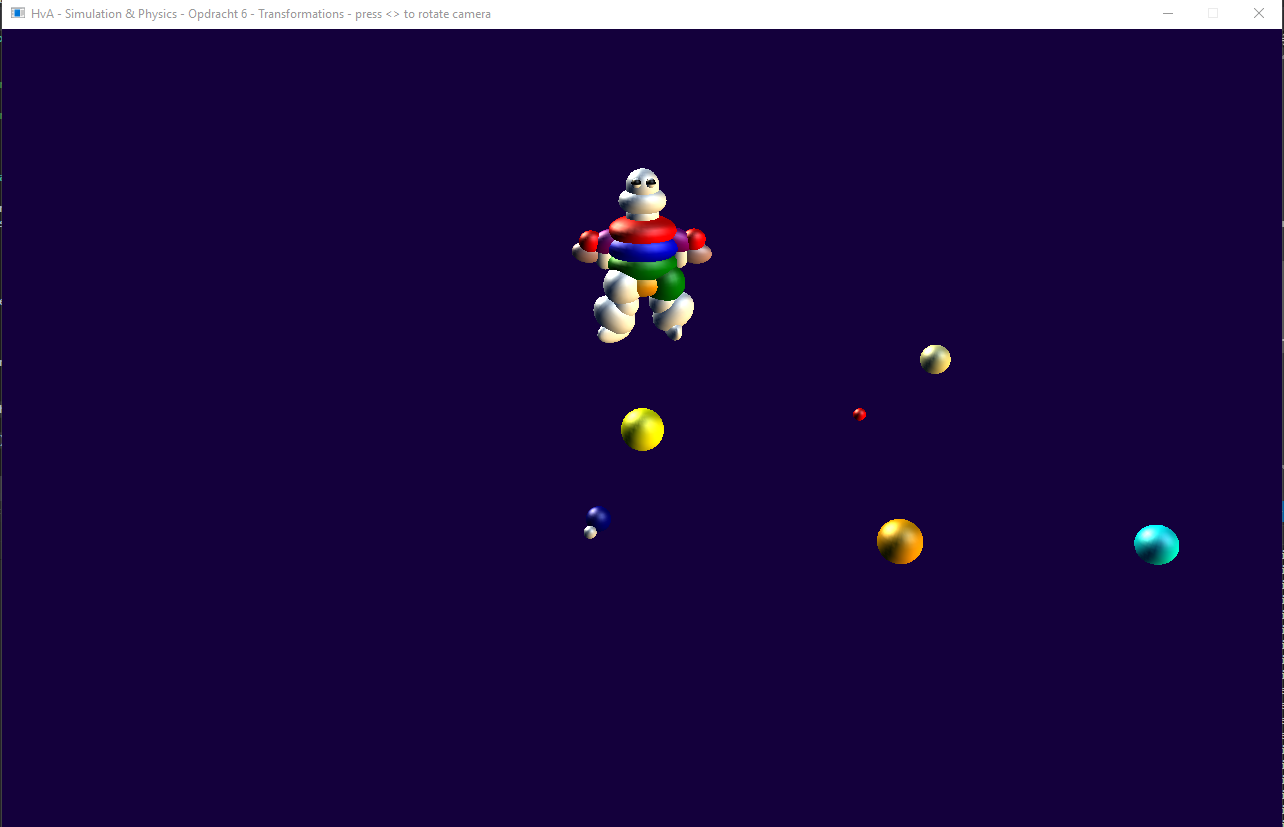
}

Bone class draw stuff



Blinking code.

## include a screenshot of your program

Again I have made video’s for the class as well These are found here: [https://www.youtube.com/watch?v=YYU0BukxEn8&index=8&list=PLARkMALdMekM6EMkY0gcQSKvADVAx9zK5](https://www.youtube.com/watch?v=YYU0BukxEn8&index=8&list=PLARkMALdMekM6EMkY0gcQSKvADVAx9zK5&t=6803s)