CS 480 SOLAR SYSTEM

John Montesa and Kaden Nesch | December 19, 2022

Sphere

+ hasTex:bool
+matAmbient[4]:float
+matDiff[4]:float
+matSpec[4]:float
+matShininess[4]:float
- pivotLocation:vec3
- model:mat4
- Vertices:vector<Vertex>
- Indices:vector<unsigned int>
- VB:GLuint
- IB:GLuint
- m_texture:Texture*
- vao:GLuint
- angle:float
- numVertices:int
- numIndices:int
- indices:vector<int>
- vertices:vector<vec3>
- texCoords:vector<vec2>
- normals:vector<vec3>
- pvalues:vector<float>
- tvalues:vector<float>
- nvalues:vector<float>

+ Sphere()
+ Sphere(prec:int)
+ Render(positionAttribLoc:GLint, colorAttribLoc:GLint):void
+ Render(positionAttribLoc:GLint, colorAttribLoc:GLint, tcAttribLoc:GLint, hasTex:GLint):void
+sphereLoad(Filename: const char*, Type: int):void
+ GetModel():mat4
+ Update(matModel:mat4):void
+ getNumVertices():int
+ getNumIndices():int
+ getIndices():vector<int>
+ getVertices():vector<vec3>
+ getTexCoords():vector<vec2>
+ getNormals():vector<vev3>
+ getTextureID():GLuint
- setupVertices():void
- setupBuffers():void
- setupModelMatrix(pivotLoc:vec3, angle:float, scale:float):void
- init(prec:int):void
- toRadians(degrees:float):float

Object

- pivotLocation:vec3
- model:mat4
- Vertices:vector<Vertex>
- Indices:vector<unsigned int>
- VB:GLuint
- IB:GLuint
- vao:GLuint
- angle:float

+ Object()
+ Object(pivot:vec3)
+ ~Object()
+ Update(model:mat4):void
+ Render(posAttrib:GLint, colAttrib:GLint):void
+ GetModel():mat4
+ InitBuffers():bool
+ setupVertices():void

«struct»
Vertex

+ vertex:vec3
+ normal:vec3
+ texcoord:vec2

+ Vertex(v:vec3, n:vec3, tc:vec2): vertex(v), normal(n) , texcoord(tc)

Shader

- m_shaderProg:GLuint
- m_shaderObjList:vector<GLuint>

+ Shader()
+ ~Shader()
+ Initialize():bool
+ Enable():void
+ AddShader(ShaderType:GLenum):bool
+ Finalize:bool
+ GetUniformLocation(pUniformName:const char*):GLint
+ GetAttribLocation(pAttribName:const char*):GLint

Mesh

+ hasTex:bool
+matAmbient[4]:float
+matDiff[4]:float
+matSpec[4]:float
+matShininess[4]:float
- pivotLocation:vec3
- model:mat4
- Vertices:vector<Vertex>
- Indices:vector<unsigned int>
- VB:GLuint
- IB:GLuint
- m_texture:Texture*
- vao:GLuint
- angle:float

+ Mesh()
+ Mesh(pivot:vec3, fname:const char*)
+loadMesh(fileName: const char*, Type: int) void
+ ~Mesh()
+ Update(model:mat4):void
+ Render(posAttrib:GLint, colAttrib:GLint):void
+ Render(positionAttribLoc:GLint, colorAttribLoc:GLint, tcAttribLoc:GLint, hasTex:GLint):void
+ GetModel():mat4
+ InitBuffers():bool
+ loadModelFromFile(path:const char*):bool
+ getTextureID:GLuint

Graphics

+ switchMode:bool
+ planetView[15]:mat4
+globalAmbient[4]:float
+lightAmbient[4]:float
+lightDiffuse[4]:float
+lightSpecular[4]:float
+lightPositionViewSpace[4]: float

- modelStack:stack<mat4>
- m_camera:Camera*
- m_shader:Shader*
- m_projectionMatrix:GLint
- m_viewMatrix:GLint
- m_modelMatrix:GLint
- m_positionAttrib:GLint
- m_colorAttrib:GLint
- m_tcAttrib:GLint
- m_hasTexture:GLint
- m_skysphere:Sphere*
- m_sun:Sphere*
- m_mercury:Sphere*
- m_venus:Sphere*
- m_earth:Sphere*
- m_mars:Sphere*
- m_jupiter:Sphere*
- m_saturn:Sphere*
- m_satring:Mesh*
- m_moons[20]:Sphere*
- m_asteroid[20]:Mesh*
- m_ast[20]:Mesh*
- m_as[20]:Mesh*
- m_a[20]:Mesh*
- m_comet[20]:Mesh*
- m_uranus:Sphere*
- m_neptune:Sphere*

+ Graphics()
+ ~Graphics
+ Initialize(width:int, height:int):bool
+ HierarchicalUpdate2(dt:double):void
+ Render():void
+ getCamera():Camera*
- ErrorString(error:GLenum):string
- collectShPrLocs():bool
- ComputeTransforms (dt:double, speed:vector<float> , dist:vector<float>, rotSpeed:vector<float>, rotVector:vec3, scale:vector<float>, tmat:mat4&, rmat:mat4&, smat:mat4&);

Camera

+ cameraPos:vec3
+ cameraFront:vec3
+ cameraUp:vec3
- m_speed:vec3
- projection:mat4
- view:mat4
- newmodel:mat4
- newv:mat4

+ Camera()
+ ~Camera()
+ Initialize(w:int, h:int):bool
+ GetProjection():mat4
+ GetView():mat4
+ Update(cameraPos:vec3, cameraFront:vec3, cameraUp:vec3):void
+ void Update2(cameraFront:vec3):void
+ void UpdateProjection(projectionUP:mat4):void
+ planetView(view:mat4):void

Window

- gWindow:GLFWwindow*

+ Window(name:const char*, width:int*, height:int*)
+ ~Window()
+ Initialize():bool
+ Swap():void
+ getWindow():GLFWwindow*

Texture

- m_TextureID:GLuint

+ Texture()
+ Texture(fileName:const char*, Type: int)
+ loadTexture(texFile:const char*, Type: int):bool
+ getTextureID(Type: int):GLuint
- initializeTexture():bool

Engine

- m_WINDOW_NAME:const char*
- m_WINDOW_WIDTH:int
- m_WINDOW_HEIGHT:int
- m_FULLSCREEN:bool
- m_running:bool
- m_camera:Camera*

+ Engine(name:const char*, width:int, height:int)
+ ~Engine()
+ Initialize():bool
+ Run():void
+ ProcessInput():void
+ getDT:unsigned int
+ GetCurrentTimeMillis():long long
+ Display(GLFWwindow*, double):void

This arrow means it #includes it in the file