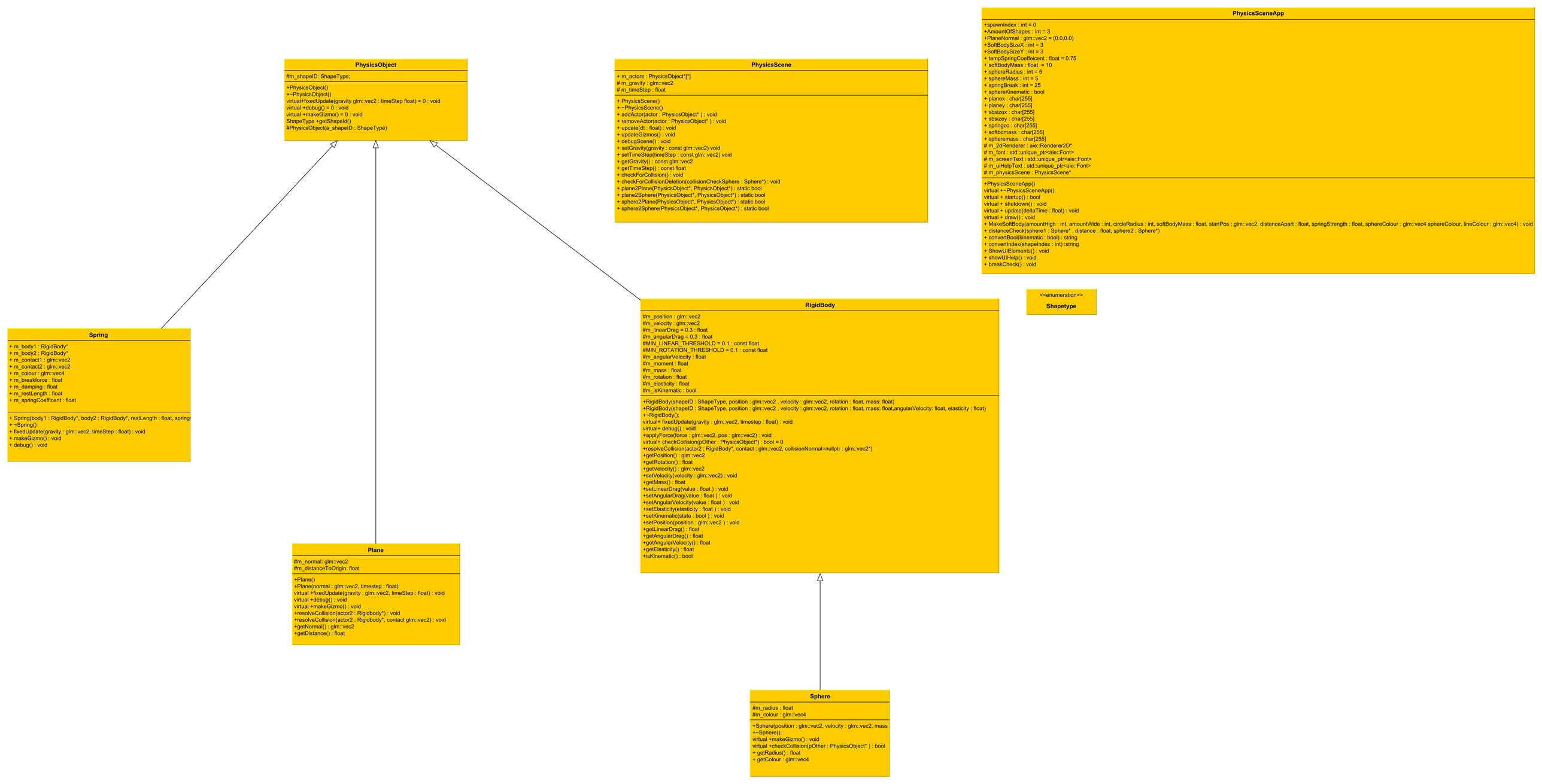
PhysicsObject
#m_shapeID: ShapeType;
+PhysicsObject()
+~PhysicsObject()
virtual+fixedUpdate(gravity glm::vec2 : timeStep float) = 0 : void
virtual +debug() = 0 : void
virtual +makeGizmo() = 0 : void
ShapeType +getShapeId()
#PhysicsObject(a_shapeID : ShapeType)
PhysicsScene
+ m_actors : PhysicsObject*[*]
# m_gravity : glm::vec2
# m_timeStep : float
+ PhysicsScene()
+ ~PhysicsScene()
+ addActor(actor : PhysicsObject*) : void
+ removeActor(actor : PhysicsObject*) : void
+ update(dt : float) : void
+ updateGizmos() : void
+ debugScene() : void
+ setGravity(gravity : const glm::vec2) void
+ setTimeStep(timeStep : const glm::vec2) void
+ getGravity() : const glm::vec2
+ getTimeStep() : const float
+ checkForCollision() : void
+ checkForCollisionDeletion(collisionCheckSphere : Sphere*) : void
+ plane2Plane(PhysicsObject*, PhysicsObject*) : static bool
+ plane2Sphere(PhysicsObject*, PhysicsObject*) : static bool
+ sphere2Plane(PhysicsObject*, PhysicsObject*) : static bool
+ sphere2Sphere(PhysicsObject*, PhysicsObject*) : static bool
PhysicsSceneApp
+spawnIndex : int = 0
+AmountOfShapes : int = 3
+PlaneNormal : glm::vec2 = (0.0,0.0)
+SoftBodySizeX : int = 3
+SoftBodySizeY : int = 3
+ tempSpringCoefficent : float = 0.75
+ softBodyMass : float = 10
+ sphereRadius : int = 5
+ sphereMass : int = 5
+ springBreak : int = 25
+ sphereKinematic : bool
+ planex : char[255]
+ planey : char[255]
+ sbsizex : char[255]
+ sbsizey : char[255]
+ springco : char[255]
+ softbdmass : char[255]
+ spheremass : char[255]
# m_2dRenderer : aie::Renderer2D*
# m_font : std::unique_ptr<aie::Font>
# m_screenText : std::unique_ptr<aie::Font>
# m_uiHelpText : std::unique_ptr<aie::Font>
# m_physicsScene : PhysicsScene*
+PhysicsSceneApp()
virtual +~PhysicsSceneApp()
virtual + startup() : bool
virtual + shutdown() : void
virtual + update(deltaTime : float) : void
virtual + draw() : void
+ MakeSoftBody(amountHigh : int, amountWide : int, circleRadius : int, softBodyMass : float, startPos : glm::vec2, distanceApart : float, springStrength : float, sphereColour : glm::vec4 sphereColour, lineColour : glm::vec4) : void
+ distanceCheck(sphere1 : Sphere* , distance : float, sphere2 : Sphere*)
+ convertBool(kinematic : bool) : string
+ convertIndex(shapeIndex : int) :string
+ ShowUIElements() : void
+ showUIHelp() : void
+ breakCheck() : void
Spring
+ m_body1 : RigidBody*
+ m_body2 : RigidBody*
+ m_contact1 : glm::vec2
+ m_contact2 : glm::vec2
+ m_colour : glm::vec4
+ m_breakforce : float
+ m_damping : float
+ m_restLength : float
+ m_springCoefficent : float
+ Spring(body1 : RigidBody*, body2 : RigidBody*, restLength : float, spring
+ ~Spring()
+ fixedUpdate(gravity : glm::vec2, timeStep : float) : void
+ makeGizmo() : void
+ debug() : void
RigidBody
#m_position : glm::vec2
#m_velocity : glm::vec2
#m_linearDrag = 0.3 : float
#m_angularDrag = 0.3 : float
#MIN_LINEAR_THRESHOLD = 0.1 : const float
#MIN_ROTATION_THRESHOLD = 0.1 : const float
#m_angularVelocity : float
#m_moment : float
#m_mass : float
#m_rotation : float
#m_elasticity : float
#m_isKinematic : bool
+RigidBody(shapeID : ShapeType, position : glm::vec2 , velocity : glm::vec2, rotation : float, mass: float)
+RigidBody(shapeID : ShapeType, position : glm::vec2 , velocity : glm::vec2, rotation : float, mass: float,angularVelocity: float, elasticity : float)
+~RigidBody();
virtual+ fixedUpdate(gravity : glm::vec2, timestep : float) : void
virtual+ debug() : void
+applyForce(force : glm::vec2, pos : glm::vec2) : void
virtual+ checkCollision(pOther : PhysicsObject*) : bool = 0
+resolveCollision(actor2 : RigidBody*, contact : glm::vec2, collisionNormal=nullptr : glm::vec2*)
+getPosition() : glm::vec2
+getRotation() : float
+getVelocity() : glm::vec2
+setVelocity(velocity : glm::vec2) : void
+getMass() : float
+setLinearDrag(value : float) : void
+setAngularDrag(value : float) : void
+setAngularVelocity(value : float) : void
+setElasticity(elasticity : float) : void
+setKinematic(state : bool) : void
+setPosition(position : glm::vec2) : void
+getLinearDrag() : float
+getAngularDrag() : float
+getAngularVelocity() : float
+getElasticity() : float
+isKinematic() : bool
<<enumeration>>
Shapetype
Plane
#m_normal: glm::vec2
#m_distanceToOrigin: float
+Plane()
+Plane(normal : glm::vec2, timestep : float)
virtual +fixedUpdate(gravity : glm::vec2, timeStep : float) : void
virtual +debug() : void
virtual +makeGizmo() : void
+resolveCollision(actor2 : Rigidbody*) : void
+resolveCollision(actor2 : Rigidbody*, contact glm::vec2) : void
+getNormal() : glm::vec2
+getDistance() : float
Sphere
#m_radius : float
#m_colour : glm::vec4
+Sphere(position : glm::vec2, velocity : glm::vec2, mass
+~Sphere();
virtual +makeGizmo() : void
virtual +checkCollision(pOther : PhysicsObject*) : bool
+ getRadius() : float
+ getColour : glm::vec4