Farseer Physics Engine Change Log

Changes For Version 0.1.0.0

FarseerXNAGame

- Refactored RectangleEntity, PolygonEntity, and PointEntity to inherit from a common base class: PolygonEntityBase.
- Added CircleEntity to go along with the other entity types.
- Made all private class level variables use the naming convention: _name
- Added a “RigidBodyDiagnosticEntityView” object to the Entities namespace. This object creates a diagnostic view of a RigidBody. Currently, it shows the vertices of the RigidBody and the collision points. (See FarseerDemo8)

FarseerXNAPhysics

- Added CollisionEnabled boolean property to RigidBody.
- Added CollisionResponseEnabled boolean to RigidBody. This will allow collision info to be generated without the bodies interacting physically.
- Modified PolygonRigidBody, RectangleRigidBody, and CircleRigidBody. Hopefully, the process of creating a rigid body and setting the way its collision grid is constructed is a little cleaner. I added a ‘CollisionPrecisionType’. If this value is set to “Relative”, the collision precision value will be multiplied by the shortest side of the rigid body’s geometry and this value will be used for the grid cell size. If the collision precision type is set to “Absolute”, the collision precision parameter will be used directly for the grid cell size. Currently, the Rigid Body Entities in the FarseerXNAGame project all use the “Relative” setting, but this change will allow for more flexibility if desired.
- Removed the PhysicsSimulatorBase class. It really wasn’t being used.
- Made all PRIVATE class level variables use the naming convention: _name
- Added RemoveJoint and RemoveSpring methods to the PhysicsSimulator class.
- Added Reset method to the PhysicsSimulator class. This method simply clears out all the list objects: RigidBody, Joint, Spring.
- Added CollisionEvent to RigidBody. The CollisionEventArgs object that is returned when the event fires will return a list of contact points.
- Added AngularJoint. This acts to keep 2 bodies at a specified relative angle. It is usually used in conjunction with a RevoluteJoint. Very similar to AngularSpring, but stiffer.
- Added Collide(Vector2 point) method to RigidBody. This adds the ability to check collision of a point with a rigid body. The method returns a bool.
- Added a “CollisonPoint” object. This object contains a boolean and a pointer to a RigidBody. It is used for doing collision tests between a point and the physics world. (see next bullet item.)
- Added CollidePoint(Vector2 point) to the PhysicsSimulator class. This method will return a CollisionPoint object. If the CollisionPoint’s “IsCollision” property is true, then you can use the CollisionPoint.RigidBody property to determine what body was collided with.
- The above two bullet points give the ability to do mouse dragging of rigid bodies. I’ve implemented a version of this in FarseerDemo8.