Game Programming: Exercise 8: Bird

Learning objectives	**Learning objectives** • Integrate Box2D physics into a game • Use a component based game architecture • Use a physics world using different units than rendered world. • Use force and impulses in Box2D • Use collision callback in Box2D **Handing in**: Create a zip-version of source-files, header-files and resources (CMakeFileLists.txt, .json and png). We will build your project using CMake, so make sure it works before hand-in (Note: If you make sure to keep all files in the same directory it should work without you need to change the CMakeFileLists.txt). Do not submit SimpleRenderEngineProject files. Changes to SimpleRenderEngineProject are not allowed. This has to be solved and handed in **individually**. **Controls:** Space: Add upwards force to the bird 'd': Show physics debug information See an example solution here: http://www.itu.dk/~mnob/bird/bird.html
8-1	**Controlling the bird** • The bird should have a constant linear velocity with an appropriate value, to make the bird move to the right at a constant speed. • When space-key is pressed, an upwards impulse with an appropriate value must be added to the bird.
8-2	**Create wall colliders** • Add a PhysicsComponent to the walls • The walls should not be able to move or rotate in world space. Only the bird moves in the game (and the camera follows) • When the bird collides with a wall then set the game state to GameState::GameOver.
8-3	**Creating bird pickups** • Spawn coin pickups in the level (by modifying BirdGame::init()). • The coins must not collide with the walls • When the bird collides with a coin then it must disappear • The coin collision must not affect the velocity of the bird in any way.
8-4	**Implement missing features (optional)**