# Fall 2016: ELG5124/CSI5151
# Assignment 4

Due: Tuesday, November 22nd, 2016, 11:00pm in Virtual Campus
University of Ottawa - Université d'Ottawa

Jochen Lang

## 1 Overview

This assignment will give you a chance to familiarize yourself with rigid body behaviour using the *Bullet* physics engine through the *Jmonkey* interface. This will mean you can concentrate on using physics and will have to do little in terms of implementing an rigid body solver.

### 1.1 Getting Started

This assignment will again be using *JMonkey 3.0* (`http://www.jmonkeyengine.org/`). The interface to *Bullet* to *JMonkey* works by running the physics engine in parallel to the rendering engine and synchronizing between the two.

1. **Initialization:** A `RigidBodyControl` object is added to the geometry node. During construction of the `RigidBodyControl` object a mass and a collision shape can be set. The `RigidBodyControl` is then added to the physics space in the `BulletAppState`.

2. **Time Stepping:** The `BulletAppState` has two synchronization methods: On the one hand there is `render` and `postRender` to connect to the render manager, on the other hand there is `prePhysicsTick` and `physicsTick` to connect to the physics space.

3. **Interacting with Physics:** The results of physics can be fed back into the rendering and input can be applied to the physics through physics listeners. A physics collision listener will receive a `PhysicsCollisionEvent` and the methods `prePhysicsTick` and `PhysicsTick` are called in each physics step when a `PhysicsTickListener` is registered. This is used in control objects, e.g., `RigidBodControl`.

You can learn more about physics in the *JmonkeyEngine* by running the *Hello Physics* beginner's tutorial and by reading the advanced documentation on *Physics: Gravity, Collisions, Forces*. In `jme3test/bullet/` are numerous test programs to explore the interface further.

### 1.2 Box Object [2]

Create a box that limits where the physics objects go and is hence a static object, i.e., objects colliding with it will have forces applied to them but the box does not move. Set it up such that we always look into the box from the front.

### 1.3 Sphere Objects [3]

Create sphere objects entering the box from behind the camera. Choose the start location arbitrarily anywhere on the invisible front plane of the box. Choose the velocity in arbitrary direction and magnitude into the box. A new sphere should be created when hitting the space bar.

### 1.4 Tiger [3]

Use the tiger mesh as a character under user control. The tiger move should be controlled by applying force to it. The arrow keys should apply a force parallel to the ground plane of the box. Note that this should apply a force to the centre of mass of the tiger. While the tiger should start on (or really just above) the ground plane, collision with the spheres may cause it to bounce.

### 1.5 Exit Door [2]

Use a door in the rear plane connected with a hinge joint to the rest of the rear plane. Set it up such that a sphere if hitting the door with a large enough momentum opens the door and exits the box.

## 2 Submission

You will need to submit your solution (only the source directory and your assets directory along with the `*.xml` project files but no other files) to BB learn by the deadline. No late submissions are allowed, you can submit multiple times but only your last submission is marked.