# Physics 2, Project 2

By David Zuniga Navarro

## How to build:

1. Project is made and compiled in VS2019.
2. Open the solution file, and make sure configuration is set to x64, Release.
3. Release is recommended for a better experience.

## How to use:

* WASD - Move the camera
* Q, E - Move the camera
* Mouse - While left clicking moves the camera
* Arrows - Move the selected sphere
* M - Make sphere move up in Y axis
* F3 - Change selected sphere

## Items:

**Compile and be Self Contained**:

It is a single project (GDP2019-20) to which these projects were added to:

* PhysicsLibrary
* PhysicsInterfaces
* PhysicsWrapper

**Decoupling:**

Inside the GDP2019-20 folder there is a folder called “PhysicsConfig” where the actual includes to the engine are made, the header is then included to the main cpp file “theMain.cpp”.

**External File:**

The jsons that describe the scene are loaded from a folder inside the main project called “configFiles” inside there are 2 important files called gameObjects.json and lights.json, which as their names imply are the ones that control which objects are rendered to the scene the gameObjects one being the most relevant in this case.

**Reasonable Rendering:**

Easily appreciated once the simulation is running. Models loaded can be found in /assets/models/\* they are PLY models.

**A good camera:**

A new camera was implemented for this functionality it can be found in the files “cFollowCamera.h & .cpp” inside the “cFlyCamera” folder.

**Good Integration:**

This can be found directly in the “PhysicsLibrary” project in the “Integrator” .h & .cpp files, where you can find the implementation of RK4 based on the link provided:

<https://gafferongames.com/post/integration_basics/>

**Planes:**

Loaded in the gameObjects.json files like this:

{

    "accel": [0.0, 0.0, 0.0],

    "alphaTransparency": 1.0,

    "debugColour": [1.0, 1.0, 1.0, 1.0],

    "diffuseColour": [1.0, 1.0, 1.0, 1.0],

    "friendlyName": "thePlane",

    "inverseMass": 0.0,

    "isVisible": true,

    "isWireframe": false,

    "mass": 1,

    "meshName": "planeMesh",

    "meshURL": "assets/models/plane\_XYZ\_n\_uv.ply",

    "objectColourRGBA": [0.0, 0.0, 0.0, 1.0],

    "planeNormal": [0,1,0],

    "physicsShapeType": 3,

    "positionXYZ": [0.0, 0.0, 0.0],

    "rotationXYZ": [0.0, -0.0, 0.0],

    "scale": 1.0,

    "specularColour": [1.0, 1.0, 1.0, 1.0],

    "texture": ["oak.bmp", "", "", ""],

    "textureRatio": [1.0, 0.0, 0.0, 0.0],

    "velocity": [0.0, 0.0, 0.0]

  },

Being *Position*, *planeNormal* the relevant parameters for each physical plane.

**Spheres:**

Loaded in the gameObjects.json files like this:

{

    "accel": [0.0, 0.0, 0.0],

    "alphaTransparency": 1.0,

    "debugColour": [1.0, 1.0, 1.0, 1.0],

    "diffuseColour": [1.0, 1.0, 1.0, 1.0],

    "friendlyName": "sphereRed",

    "inverseMass": 0.0,

    "isVisible": true,

    "isWireframe": false,

    "mass": 1,

    "meshName": "sphereMesh",

    "meshURL": "assets/models/Sphere\_Radius\_1\_XYZ\_n\_uv.ply",

    "objectColourRGBA": [0.0, 0.0, 0.0, 1.0],

    "physicsShapeType": 1,

    "positionXYZ": [0.0, 80.0, 0.0],

    "rotationXYZ": [0.0, -0.0, 0.0],

    "scale": 2.0,

    "specularColour": [1.0, 1.0, 1.0, 1.0],

    "texture": ["White.bmp", "Red.bmp", "", ""],

    "textureRatio": [1.0, 0.0, 0.0, 0.0],

    "velocity": [0.0, 0.0, 0.0]

  },

Being *mass*, *scale*, *positionXYZ* & *velocity* the relevant parameters.

(Accel and InverseMass are legacy parameters from physics1)

**Interactions:**

Easily appreciated once the simulation is running.

**Control:**

Spheres can be controlled via the arrow keys to move on the x-z axis (aligned to the plane)

Also, with the M key the ball can be “lifted” on to the air.

All applied forces are relative to the camera in the x-z plane and parallel to the "floor" plane.