FOREWORD

之前使用的text软件由于涉及敏感问题，已卸载，本文中不再出现

1、简单下坠模型

这是一个最简单的刚体下落的demo

这个demo在官网有，为了方便我师弟学习

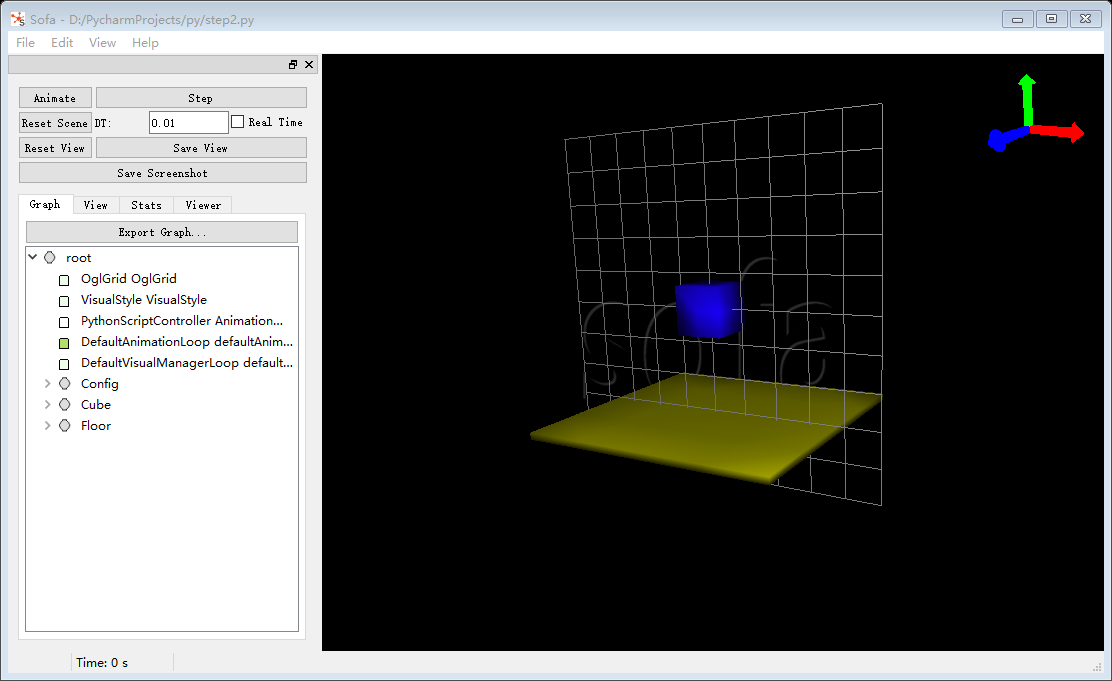
我准备一个一个demo的做中文教程，也方便想学习sofa&softrobots的同学

毕竟，看中文比看英文舒服嘛

会进行持续更新更多的demo以及我自己做的project

看不起的高玩朋友不要骂我直接退出可好

首先，先来看一下这个demo



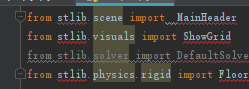
从graph中，很显然，只包含了一个root节点，以及2个root下的子节点，config不算哦，这个是配置文件。

要重新来创建这个demo，第一步需要看看能用到哪些现成的组件

在stlib的插件中，开发组已经给出了很多

刚体的库在stlib.physics.rigid下，这里提供了cube floor以及sphere，3个最简单的刚体，在这个demo中，只是创建了一个floor文件，并没有创建cube，为啥呢，因为要学动力学建模啊，要告诉你怎么导入自己的mesh。这个后面再来说

Python语言惯例，先import

So：(edited in Pycharm,. This software has already been activated through education approach)

注意大小写和拼写，不能有一点点的错！这个很重要，用过py和linux的都知道吧

需要用到的库我们加载完了，接下来，就该配置环境了

第一步，创建一个rootNode，在官方的教程中间，有这么一句” at least one function named **createScene** taking a **single parameter**, the **root** of the scene hierarchy”这是什么意思呢，就是说，你在每一个pyscn文件（也可以是py文件）中，都必须包含一个createScene（rootNode）来定义一个环境。

所以才有了这么一句： def createScene(rootNode)，（这里最好不要用别的名字，我试过，虽然demo可以跑，但是warming一大堆的。而且也不能def两个createScene）

这份文件是在win下写的，改.py我还是推荐用notebook++，pycharm，记事本什么的写这个太神经病了。

定义完了以后，要让网格线可见，就需要，ShowGrid(rootNode)这样在rootNode下所有的网格线就都可见了。ShowGrid就是之前import的在visual下的函数

而在场景中，我们需要定义重力的，源码中说的很明白，要把MainHeader指定到rootNode上并定义重力。

所以接下来 MainHeader(rootNode, gravity=[0.0, -981.0, 0.0])

Gravity是vec3f来定义的，表示了x，y，z轴的速度，单位cm/s^2

这步做完，场景基本上就配置完了，最简单的那种

接下来就是要给小方块来建模了，没有使用rigid库中的cube，所以我们线来定义一个基于rootNode的cube节点

cube=rootNode.createChild(“Cube”) #大写的Cube是这个节点的名称

节点创建完毕，需要定义简单的动力学参数，教程上给出的

totalMass 总质量 volume 体积 inertiaMatrix 转动惯量的矩阵形式计算方法参考刚体动力学

totalMass=1.0

volume=1.0  
 inertiaMatrix=[1.0,0.0,0.0,0.0,1.0,0.0,0.0,0.0,1.0]

注意格式，缩进，这里不是矩阵

然后我们开始描述这个cube节点

cube.createObject('MechanicalObject',name="DOF",template="Rigid3",translation=[0.0,0.0,0.0],

rotation=[0.0,0.0,0.0])

调用sofa本身的库，这个库的名字是MechanicalObject，位置是~.\include\SofaBaseMechanics我之后会把这个库下的代码都看一遍的，现在先写了。

这一项给Object命名，只要是英文，都行

这一项，你需要告诉sofa你是使用2d还是3d，源码中的template有Rigid2d,Vec2d, Rigid3d,ExtVec3d, Rigid3d,Vec3d.这里的rigid3就是调用了Rigid3d这一个模板

这一项，描述这个物体的初始位置，vec3f写的，定义了沿着x,y,z3轴平移的距离cm

这一项，描述了物体沿着x,y,z3个轴旋转的初始状态。

cube.createObject('UniformMass',name="vertexMass",vertexMass=[totalMass,volume, inertiaMatrix[:]])

这一项调用的UniformMass库，用来干啥，用来把你之前定义的质量，体积，转动惯量写进软件

cube.createObject('EulerImplicit', name='odesolver')

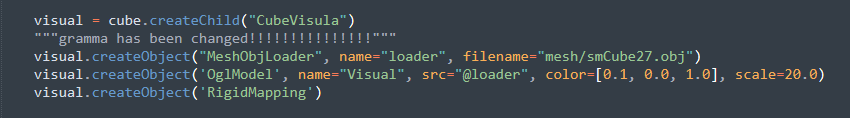
cube.createObject('CGLinearSolver', name='Solver')

这两个，加入解算器，在stlib的doc里，添加了defaultsolver之后，你还是需要添加解算器的，2个求解器，一个欧拉法，一个共轭梯度发，一个是时间积分的用来求每个步长的控制方程的求姐，共轭梯度也是一个求解器，用来求解耦合在系统中的方程，更新位置信息什么的求解器。

全部定义完后这个cube的子节点就OK了

然后，是配置可视化

直接说代码含义吧



第一句，创建一个在cube下，名字叫CubeVisual的子节点

第二句，make attention网页版的教程上关于网格，颜色等定义的语法已经被弃用，根据提示的用法为创见2个object分别是MeshObjLoader以及OglModel，第一个object是用来加载mesh文件的，obj文件最好是放在share里面

第三句，说真的，src这个我没看懂，我猜是继承，改天好好看看源码就知道了

最后，创建一个floor

之前加载过，所以直接用Floor()就行了

加载完后，return rootNode “””这句总不会知道了吧”””

2、下坠+碰撞

碰撞，也就是说需要在cube节点下添加一个collision节点来描述他的碰撞

使用python的好处就是，创建碰撞的描述真的很方便

首先就是先在cube下建立一个子节点，首先命名一下，名字随意，任意英文或者数字就行

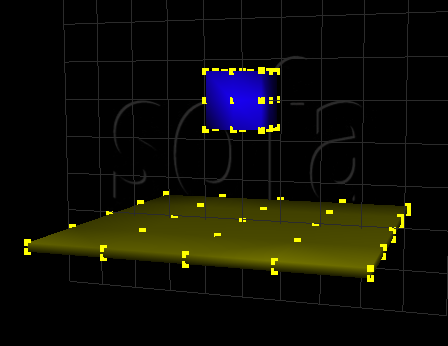
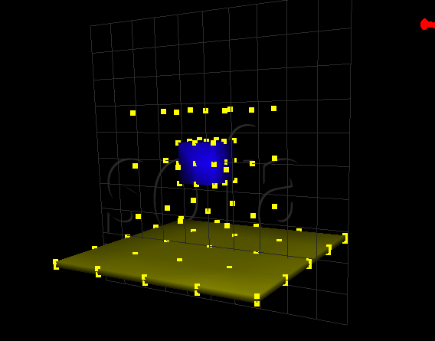
Collision=cube.createChild(“cubecollision”)

这样这个子节点就创立好了，但是现在里面什么参数都没有，所以需要createObject

Collision.createObject(‘MeshObjLoader’,

name=”loader”, filename=”mesh/smCube27.obj”.triangulate=”true”, scale=20.0)

这里需要注意一下哈scale上下要统一，这里之前的scale值是20，如果这里变成50会怎么样呢？会这样，这里过度放大了他的三角形网格，所以，计算的时候会出现，网格碰撞了，但是物体没有接触，这是因为计算过程是通过网格节点来进行计算的。源码的描述是定义一个特定的mesh文件

接着：

Collision.createObject(‘MeshTopology’, src=”@loader”)

这里就很明显了，数据结构继承之前的loader命名的MeshObjLoader

Collision.createObject(‘MechanicalObject’)

创建动力学动力学，继承之前的

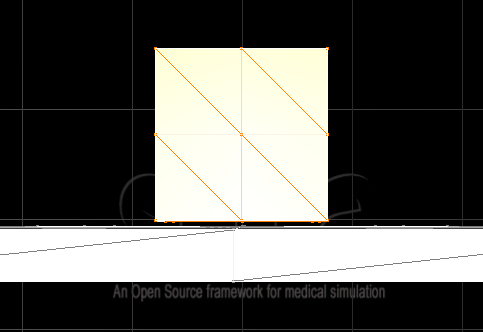
Collision.createObject(‘TTriangleModel’) 这个应该是针对网格的，表面是三角形网格撞击也产生在表面

Collision.createObject(‘TLinerModel’) 表面的线

Collision.createObject(‘TPointModel’) 表面的点

Collision.createObject(‘RigidMapping’) 源码的解释是用来计算每一个step的坐标的，也就是说，不定义，你之前的碰撞就没用了

到这一步，所有的碰撞就定义完成了



看一下，这样看起来，误差还挺大的呢

多物体，下落+碰撞