有两个脚本可以用来绑定BIM数据到模型上，InitNavisFileInfoByModel和NavisModelRoot。

InitNavisFileInfoByModel用来全局处理，NavisModelRoot用来局部处理。

为了避免查找对应模型时去查找无关的模型，推荐用NavisModelRoot来将一个模型根节点和一个vue文件根节点关联起来。

这个关联的过程是将NavisFileInfo.xml和场景中的模型关联，将xml中的每个模型节点找到对应的模型对象(GameObject)。NavisFileInfo中包括了从Navis中保存的坐标信息，通过该坐标信息去匹配三维模型的坐标，找到距离最近的那个模型。

ListOfNavs->ListOfModels，MinDistance

按钮功能：

LoadModels：读取模型信息，根据ModelName，ModelName为空的话默认和游戏对象同名。

1. 从当前游戏对象的子物体中找出可以匹配的子游戏物体列表，这里会排除部分无关的物体（如LOD1，空的分组物体，无关的物体名称）
2. 从navisFile中找到ModelName对应的根模型，并获取全部子物体，并根据坐标是否为(0,0,0)和是否为Drawable放到4个不同的列表中。1.Drawable&Zero 2.Drawable&NoZero 3.NoDrawable&Zero 4.NoDrawable&NoZero。
3. 找出vue模型信息(ModelItemInfo)对应的游戏物体并绑定上BIMModelInfo脚本。不同的列表采用不同的方式处理。
   1. Drawable&Zero：根据名称或者UID找到相同的名称，或者名称就算为UID的游戏对象。
   2. Drawable&NoZero：根据坐标信息找到相同的坐标游戏对象，主要的处理列表。
   3. NoDrawable&Zero：不需要关联
   4. NoDrawable&NoZero：根据坐标信息找到相同的坐标游戏对象。
4. UID具有唯一性，在Navis中，因此，先通过UID查找是否存在同名的游戏对象，找到后再判断距离是否合适。
   1. 确认文件中的UID是唯一的。
   2. 确认游戏对象中的UID是唯一的。
   3. 通过UID关联游戏对象。
5. Navisworks导出的fbx模型中并没有一个标识可以标致节点的唯一性，Navis也没有提供重命名的功能，最终采用根据模型坐标来关联信息的方式。从Navisworks导出的xml中保存了模型的坐标信息(Navisworks的API提供了获取三维坐标信息的接口)，在Unity中于模型的坐标进行比较，坐标相同（或者很接近的情况下），再考虑一下命名，将模型信息和属性信息关联起来。

这里有两个列表，

列表1 ModelList：Navisworks导出的xml的文件中的节点列表，有坐标信息。

列表2 TransformList：Unity中的GameObject模型对象列表，有坐标信息。

算法的目的是从目录树定位到模型上，目录树是ModelList，模型是TransformList。

ModelInfo获取坐标信息，和TransformList中的模型坐标进行比较，坐标相同的则作为相同的物体关联起来。

可能出现的情况，

1. 不同的模型有相同的坐标。
2. 模型更新过程中坐标发生里变化。