游戏中的表，协议，等数据使用工程MogoSolution，生成UnityEngine.dll，实现热更新，UI、战斗、AI等模块没法实现热更新.

NGUI：

camera，UICamera：UI要在摄像头camera视野内才能正常显示，可以有多个摄像头，则可以显示多个范围内的UI，摄像头设置深度depth，可以改变不同视野范围内UI重叠时的层次；

UIView：设置摄像头的视野范围；

UIDraggableCamera：与camera组件绑定在同一个gambObejct上，通过拖动camera改变摄像头视野范围，实现滑动列表的拖拽效果；

UIGrid：乘放item的格子；

UIDragCamera：可以通过拖动摄像机实现拖拽功能的格子；

影响UI层次的不止有depth值，还有位置中的Z值；

UGUI：

Canvas：所有ui必须布置在画布Canvas下。Canvas可以设置渲染模式：Overlay，UI会显示在屏幕最上层，与摄像机无关；camera，UI随着摄像机的位置和视野改变大小；world space，UI类似场景中的物体一样被绘制

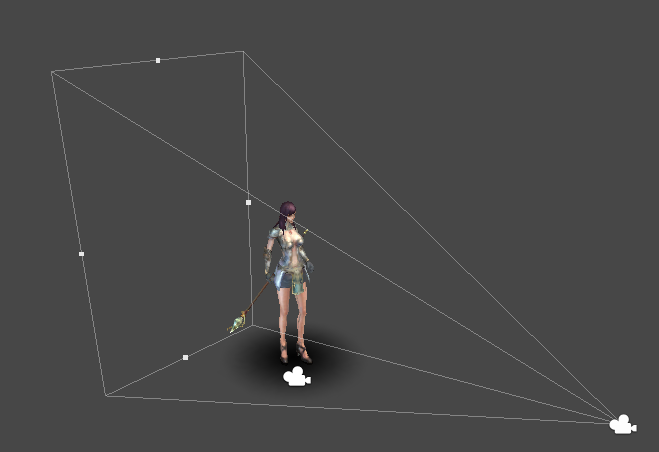
pivot，中心点；Anchors，描点( 被固定的点)

网络：

2016.06.28

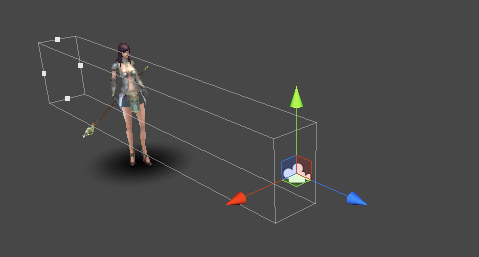
camera投影方式Projection分为两种Perspective透视投影，和Orthographic正交投影；

Perspective透视投影



属性Field of View视野范围，定义摄像机窗口的大小；透视投影多用于场景和人物模型显示；

Orthographic正交投影



属性size定义正交窗口层次大小；正交投影用于UI的显示；

Clipping Planes剪切面，定义摄像机的近平面和远平面；camer下的UI必须在剪切面中，才能被显示出来，条件UI的z坐标就是调节UI离摄像机的远近，z坐标越小离摄像机越近，z坐标的层次优先于Depth深度值

2016.06.29

UIViewPoint组件：视野投影，该组件的GameObject必备Camera组件mCam；功能是把自身摄像机mCam拍摄到的视野放到原摄像机sourceCamera的topLeft与bottomRight区域内；摄像机mCam的照射区域由topLeft与bottomRight决定；siez由UIViewPoint的FullSize与topLeft、bottomRight共同决定（FullSize是整个UIViewPoint的视野）；

拖拽UI：要实现拖拽UI功能，是通过一个可拖拽的摄像机UIDraggableCamera实现的

2016.07.04

UI设计模式：

每个UI分两个类，一个是XXXView类，负责设置UI的显示，一个是XXXManager类，负责更新数据，UI操作；XXXManager调用XXXView的函数进行UI刷新，XXXView想要触发事件通过EventDispatcher消息处理机制触发相应的UI消息（优点：数据与逻辑分开，实现了MVC框架，缺点：单个UI逻辑较简单，用多个类实现起来太过繁琐，有点过度设计，数据处理量较小，基本上服务端已经将数据处理了，相当于服务端已经把数据与UI显示分开了，单独一个XXXManager类没必要）