**依赖关系：(2015.12.12)**

**层次结构：(2015.12.12)**

* **NoiseGUI:**

1. (2015.10.20)NoiseGUI包括一个NoiseGUIBasicContainerInfo的基类，NoiseGUIMananger，NoiseGUIText,NoiseGUIScrollBar
2. (2015.10.20)NoiseGUI用了**基于事件**的机制，由NoiseGUIManager来DispatchMessage。需要**传递一个InputEngine的指针**拿来用，判断鼠标状态等。还需要传入渲染窗口的HWND来获取各种参数。
3. (2015.12.12)为了保证内置控件的正常显示，所有GraphicObject都在控件信息更新之后进行更新。

* **NoiseGUIBasicContainerInfo:**

1. (2015.10.20)NoiseGUIBasicContainerInfo基类概括了大部分控件会用到的参数，例如Texture,Position,Size（但是不知道为什么在exe中派生类的基类函数对Intelli Sense不可见导致要全部函数都要对应的定义一个接口才不画红色波浪线，虽然编译是正常的，好鬼烦）。
2. (2015.10.20)NoiseGUIManager里面内置了GraphicObject，AddToRenderList实际上也是把这个graphicObj加到renderList。

* **NoiseGUIManager:**

1. (2015.10.23)NoiseGUIManager，用于创建、销毁控件等，还有用于接受鼠标等输入信息来判断控件状态并**发送消息**。（用子控件储存的函数指针）。在Update函数里面处理了所有控件的事件生成和发送处理。
2. (2015.11.12)改成每个子对象都内置一个GraphicObj，从而让用户更好地选择调整的可见性，渲染顺序等属性
3. (2015.11.26)不再提供AddChildObjectToRenderList函数，用户自行决定渲染顺序，而且这样不会重复渲染childObject（有些控件可能会内嵌其他控件例如ScrollBar里面有button）
4. (2015.12.8)因为窗口大小可能会有变化，所以会有一个scaleFactor，用于适应这种缩放，把屏幕空间的鼠标坐标转换到BackBufferSpace的坐标，这样GUI的鼠标判断才不会有偏移。
5. (2015.12.12)新增对窗体是否带边框进行判断，然后用GetSystemMetrics来获取边框和标题框的长度。

* **NoiseGUIButton:**

1. (2015.10.23)NoiseGUIButton：如果exe有需要进行事件的处理，还需要传入**回调函数**指针，有点像windows消息机制，消息处理的回调函数的输入参数是NOISE\_GUI\_EVENTS，用户在exe里可以用switch来处理对应事件。(2015.11.23)消息的获取可以是主动式(GetEventList)或者是被动式(Callback Function)。**callback function由manager来call**，控件只负责储存函数指针。（以前用的VB甚至一个事件对应一个消息处理函数，现在设计是一个控件一个函数）。所以在exe中，在对应的消息处理函数中需要用switch或if等来判断传入的NoiseGUIEvents的值来做相应的处理。
2. (2015.10.30)NoiseGUIButton： 现在Button可以通过SetDragableX（或者Y）来决定是否可以鼠标拖拽（按下左键时）。现在为了简化起见拖拽的时候按键中心会跟着鼠标(2015.11.26)会记录下第一次按下左键时button中心和鼠标的坐标的偏移，这样拖拽就不会让button突然跳跃（用鼠标相对位移叠加的话会因为有延迟导致效果不太好）。

* **NoiseGUIScrollBar:**

1. (2015.11.26)内置一个GUIButton作为拖拽的滚动块，一个GraphicObj作为滚动槽的图片容器。ScrollBar可以设置为横/竖 向摆放，只需要修改滚动块的DragableX/Y属性即可。
2. (2015.11.26)在GUIManager的Update函数里面会根据摆放方向和滚动块位置更新ScrollBar的Value。其中Value可以通过滚动条的rangeMin、rangeMax、滑块位置等通过求线性插值的t取得。先后顺序：先更新位置（把滑块button绑定到滚动槽上），再更新Value。最左/上是最小值，最右/下是最大值

* **NoiseGUITextBoxs:**

1. (2015.12.12)内置一个TextDynamic和背景GraphicObj。GUIManager需要设置fontMgr。
2. (2015.12.12)TextBox为了防止按个按键都给它输入文本，特地的设置的焦点(focus)这一个属性，鼠标在textbox内点击就能获得焦点。只有在获取焦点的时候才可以输入文本。(当然还需要enabled和editable)。
3. (2015.12.12)DirectInput的那些scan code 暂时还要用自己写的函数(惨痛- -)映射到ascii char。
4. (2015.12.12)因为通常在输入文字的时候，按下一个按键的时间是渲染多帧的时间，为了防止按下一个key就键入很多个相同的字符，GUIManager特地内置了一个Timer来处理Textbox这种问题，记录**按下的时间**和按键**是否和上次按下的键一样**来判断是否增加字符。