**最后更新时间：2012-10-22**

# **UI库**

这是一份UI库的文档，下面包含了所有UI库中类库的公共接口定义与说明。UI库中的每一个类（除了scroller）都是合法节点，可以按照基础节点的方式去使用（至少包含了DrawingBase的所有接口）。UI库中为了保证节点合法，用了大量的继承，每个类说明下都有这个类的基类说明，这也是可操作接口的一部分。最后就是建议大家尽量用UI库而不要直接用core层，因为UI库提供了多一层封装也就意味着可以写更少的代码，另一方面也可以屏蔽底层，当引擎核心改动之后，UI库可以把这些细节屏蔽掉，使游戏代码适应性更强。

## Node

类说明：

Node继承自DrawingEmpty ，是一个空节点，包含所有DrawingEmpty，DrawingBase中的一切属性。二次开发过程中，若遇到需要继承自空节点，或需要使用空节点，应使用此类，而不是DrawingEmpty。

## Image

类说明：

Image提供了创建图片的快捷方式，对用core层的res和drawing创建可见图片做了简单封装。

Image类继承自DrawingImage，所以包含所有DrawingImage的成员变量与成员方法，这一切在Image中都可以直接使用。

接口说明：

|  |  |  |
| --- | --- | --- |
| 函数名 | 函数原型及参数说明 | 接口说明 |
| Image.ctor | function(self,file,fmt,filter)  file—图片名（可包含一resources/iamges为根目录的相对路径）  fmt—加载图片格式，可为kRGBA8888,  kRGBA4444,kRGBA5551,kRGB565  filter—纹理贴图过滤方式，可为kFilterNearest,kFilterLinear | 此为构造函数，new对象的时候会自动调用，构造一个Image对象。 |
| Image.setFile | function(self, file)  file – 同构造函数 | 调用此接口，可改变显示的图片，此接口效率极低，与重新构造图片效率几乎等价，不要频繁调用。 |
| Image.dtor | function(self) | 析构函数，释放所有信息 |

## Images

类说明：

Images封装了一个图片序列，传入一个图片路径数组可以创建一个Images，这样就可以通过setImageIndex切换图片。

Images继承自Image，所以可以调用Image的任何接口。

接口说明：

|  |  |  |
| --- | --- | --- |
| 函数名 | 函数原型及参数说明 | 接口说明 |
| Images.ctor | function(self,filenameArray,fmt,filter) filenameArray – 一个包含图片的路径的纯数组，不可包含空洞。  其他参数与Image相同 | 此为构造函数，new对象的时候会自动调用，构造一个Images对象，并载入所有贴图。 |
| Images.setImageIndex | function(self, index)  index – 图片序列的索引，比如有三张图片，则索引为0-2 | 此函数重载了DrawingImage.setImageIndex，  做了一些安全检查。 |
| Images.dtor | function(self) | 析构函数，释放所有信息 |

## Button

类说明：

Button 提供了一个简单的按钮封装，背景是一张png图片，点击的时候会变灰。可以设置为不可点击，不可点击是也会变灰。

Button类继承自Image，所以实质上Button只是一个重载了事件处理函数的Image而已。

目前只会收到 TouchupInside消息。将来可能会随着需求加入其他消息类型。如果需要自定义消息类型，可继承Button，重写onClick方法。

接口说明：

|  |  |  |
| --- | --- | --- |
| 函数名 | 函数原型及参数说明 | 接口说明 |
| Button.ctor | function(self,file,fmt,filter)  同Image.ctor | 此为构造函数，new对象的时候会自动调用，构造一个Button对象。 |
| Button.setEnable | function(self,enable)  enable – true/false | 设置button是否可点击，若enable为false，则button不可点击，同时变为灰色。 |
| Button.setOnClick | function(self, obj,func)  obj --可为任意参数  func – 必须为一函数 | 设置一个回调函数，当button点击事件出发时回调设置的func，同时将obj作为第一参数传入。  注意：button默认只处理TouchupInside这一种事件。如需处理其他事件，继承这一函数重写Button.onClick方法即可。 |
| Button.dtor | function(self) | 析构函数，释放所有信息 |

## Button2

类说明：

Button2 提供了另一种按钮，背景是一张png图片，点击的时候会变为另一张图片。可以设置为不可点击，不可点击时也会变为第二张图片。

Button类继承自Images，大部分行为和Button完全一致。

接口说明：

|  |  |  |
| --- | --- | --- |
| 函数名 | 函数原型及参数说明 | 接口说明 |
| Button2.ctor | function(self, file, file\_disable, fmt, filter)  与button相比只是多了file\_disable，这就是上面说的第二张图片。 | 此为构造函数，new对象的时候会自动调用，构造一个Button2对象。 |
| Button2.setEnable | 与button.setEnable相同 | 。 |
| Button2.setOnClick | 与button.setOnClick相同 |  |
| Button2.dtor | function(self) | 析构函数，释放所有信息 |

## CheckBoxGroup

类说明：

CheckBoxGroup 提供了复选框组的封装，与下面的CheckBox结合使用，下面的CheckBox表示复选框组中的每一个复选项，这个类提供了对所有这些选项的统一管理。

CheckBoxGroup 继承自Node是一个空节点，可以作为任何节点的子节点加入。

接口说明：

|  |  |  |
| --- | --- | --- |
| 函数名 | 函数原型及参数说明 | 接口说明 |
| CheckBoxGroup.ctor | function(self) | 无需任何参数，构造一个复选框组。 |
| CheckBoxGroup.addCheckBox | function(self,box)  box – CheckBox对象 | 在复选框组中加入一个复选项 |
| CheckBoxGroup.getIndex | function(self,box)  参数意义同上 | 通过CheckBox对象，得到这一复选项在这一组中的索引，未找到返回空 |
| CheckBoxGroup.getCheckBox | function(self,index)  index – 复选框组中的索引 | 通过索引找到对象的复选项，未找到返回空 |
| CheckBoxGroup.removeCheckBox | function(self,box)  box意义同上 | 从复选框组中移除指定复选框，若找到则移除后返回该复选项，未找到返回空。  注意：此移除并不会从内存中删除该复选项，所以如需释放，需手动delete返回值。 |
| CheckBoxGroup.removeCheckBoxByIndex | function(self,index)  index意义同上 | 说明同上 |
| CheckBoxGroup.getResult | function(self) | 得到当前有哪些选项被选中，返回一个数组包含被选中项的索引 |
| CheckBoxGroup.setOnChange | function(self,obj,func)  回调参数的意义全部都相同，可见Image | 当有一个复选项的状态改变时调用这一函数，回调函数的第二参数为复选项，第三参数为当前状态 |
| CheckBoxGroup.setEnable | function(self,enable)  enable意义同Image | 设置当前复选框是否可点击 |
| CheckBoxGroup.dtor | function(self) | 析构函数，首先释放所有这组中的复选框，然后释放自己 |

## CheckBox

类说明：

参见CheckBoxGroup

CheckBox继承自Images。

接口说明：

|  |  |  |
| --- | --- | --- |
| 函数名 | 函数原型及参数说明 | 接口说明 |
| CheckBox.changeDefaultImages | function(image)  image 应该是这样的"ui/checkBox%d.png"，其中%d可以用1,2,替换为未选中状态和选中状态的图片。传入空表示用默认的。 | 静态接口，调用之后将改变所有checkbox的图片样式。 |
| CheckBox.ctor | function(self,fileNameArray)  fileNameArray – 包含图片路径名的数组，此数组应包含两张图片的路径，第一张为未选中状态图片，第二张为选中状态图片。此参数可为空，使用默认图片。 |  |
| CheckBox.setState | function(self,checked)  checked – bool型 | 手动设置这一复选项的状态，传入true设为选中，false设为未选中 |
| CheckBox.getState | function(self) | 得到这一复选项当前状态 |
| CheckBox.dtor | function(self) | 析构函数释放所有信息。 |

## RadioButtonGroup & RadioButton

类说明：

此两个类与上面CheckBoxGroup 和CheckBox扮演的角色完全相同，接口也完全一致，继承链也相同。只是这一组处理的是单选按钮。

接口说明：

参见CheckBoxGroup 和CheckBox。所有接口名带CheckBox的全部替换为Button即为此两个类中的接口名。

## Text

类说明：

Text 单行文本，通过本类创建的文本强制为单行，指定宽高不足时会自动扩展到文本显示所需宽高。

Text 继承自DrawingImage所以包含DrawingImage的所有属性。

接口说明：

|  |  |  |
| --- | --- | --- |
| 函数名 | 函数原型及参数说明 | 接口说明 |
| Text.ctor | function(self, str, width, height, align,fontName,fontSize,r, g, b)  str—要创建的字符串  width,height – 宽高  align 对其方式，在宽高足够的时候生效  fontName – 字体名  fontSize – 字体大小  r,g,b 字符显示颜色 | 构造函数，创建一个字符串 |
| Text.setText | function(self, str,width,height)  参数同上 | 重新设置字符串和宽高 |
| Text.getText | function(self) | 得到当前字符串 |
| Text.dtor | function(self) | 析构函数释放所有信息 |

## TextView

类说明：

TextView 多行文本，通过本类创建的文本可为单行或多行，若指定宽高不足会在宽度不变的情况下，纵向扩展。如果高度不为0，且高度不足，而出现滚动条，会在指定高度范围内显示可滚动文本。

Text 继承自Node，包含节点所有属性。

接口说明：

接口同Text。

## EditText

类说明：

显示属性和Text相同，多了编辑属性。

EditText 继承自Text。

接口说明：

|  |  |  |
| --- | --- | --- |
| 函数名 | 函数原型及参数说明 | 接口说明 |
| EditText.ctor | function(self, str, width, height, align,fontName,fontSize,r, g, b)  同Text | 构造函数，创建一个字符串 |
| EditText.setEnable | function(self,enable)  enable -- bool | 设置是否可编辑 |
| EditText.setMaxLength | function(self,maxLength) | 设置文本最大长度 |
| EditText.dtor | function(self) | 析构函数，释放所有信息 |

## EditTextView

类说明：

显示属性和TextView相同，多了编辑属性。

EditTextView 继承自TextView。

接口说明：

同EditText。

## ScorllBar

类说明：

此类封装了一个滚动条，只需在构造时将最大高度和试图高度传入，在滚动时将当前位置传入，就可以显示在正确位置。

ScrollBar 继承自DrawingImage。

接口说明：

|  |  |  |
| --- | --- | --- |
| 函数名 | 函数原型及参数说明 | 接口说明 |
| ScrollBar.changeDefaultImages | function(image)  image 是一个图片路径，传入空表示使用默认。 | 静态接口，调用之后将改变所有ScrollBar的图片样式。 |
| ScrollBar.ctor | function(self,x,y,width,viewHeight,scrollHeight)  x,y -– 初始位置，也就是未滚动时位置  width – 滚动条宽度  viewHeight – 视图高度  scrollHeight – 最大高度 | 构造函数，创建滚动条 |
| ScrollBar.setViewHeight | function(self,viewHeight)  viewHeight – 同构造函数 | 此接口在滚动中设置可能会使滚动条出现跳动 |
| ScrollBar.setScrollHeight | function(self,scrollHeight)  scrollHeight – 同构造函数 | 此接口在滚动中设置可能会使滚动条出现跳动 |
| ScrollBar.setBgImage | function(self,image)  image – 图片名 | 设置滚动条显示图片，不调用则默认为灰色图片 |
| ScrollBar.setInvisibleTime | function(self,timer)  timer – 时间，单位ms | 设置滚动条渐变消失时间 |
| ScrollBar.setScrollPos | function(self,scrollY)  scroll--当前滚动位置 | 若当前位置超出应该滚动位置，则滚动条会缩小。 |
| ScrollBar.setVisible | function(self,isVisible)  isVisible – bool | 设置是否可见，若为false，则回淡出 |
| ScrollBar.setVisibleImmediately | function(self,isVisible)  isVisible – bool | 设置是否可见，无淡入淡出效果，立马生效 |
| ScrollBar.dtor | function(self) | 析构函数，释放所有信息 |

## Scorller

类说明：

此类封装了一个常规滚动，将当前接受到的事件类型、位置传入此类对象中，它会跟踪这一事件，传回当前滚动事件类型、此次滚动的变换、所有滚动累计变化。而在这过程中，它会自动计算滑动、弹回等。

Scroller不是一个显示节点，只是一个控制结构。

注意：

1. 此类并不自己注册事件，必须由宿主在事件处理函数中，调用其接口Scroller.onEventTouch = function(self,finger\_action,x,y,drawing\_id\_first,drawing\_id\_current) 来将事件传入。
2. 若使用此类处理滚动，则不要手动改变位置，若必须改动，则调用setOffset接口以便将累计值告知此类。

接口说明：

|  |  |  |
| --- | --- | --- |
| 函数名 | 函数原型及参数说明 | 接口说明 |
| Scroller.ctor | function(self,height,maxHeight)  height – 视图高度  maxHeight– 最大高度 | 构造函数，创建一个滚动记录者 |
| Scroller.setOffset | function(self,offset)  offset – 数值型 | 设置当前滚动累计，谨慎调用，建议只用于重新初始化。要不然结果未定义。 |
| Scroller.setHeight | function(self,height)  height–数值型 | 设置视图高度，警告同上。 |
| Scroller.setMaxHeight | function(self,maxHeight)  maxHeight–数值型 | 设置最大高度，警告同上。 |
| Scroller.setFlippingFrames | function(self,frames)  frames– 数值型，表示帧数 | 设置滑动效果的影响帧数，建议在一次事件完全结束之后调用，否则结果未定义。默认为20。 |
| Scroller.setFlippingSpeedFactor | function(self,factor)  factor– 数值型 | 设置滑动速度因子。警告同上。默认为0.01。 |
| Scroller.setFlippingOverFactor | function(self,factor)  factor– 数值型 | 设定滑动停止点。此参数为比例，与试图高度相乘极为停止点。警告同上。默认为0.3。 |
| Scroller.setReboundFrames | function(self,frames)  frames --数值型，表示帧数 | 设置回弹影响帧数。警告同上。默认为20。 |
| Scroller.setScrollCallback | function(self,obj,func)  参数同一切回到方法参数 | 回调时参数是这样的：  func(obj,滚动状态,此次滚动变化 ,累计变化);  其中第二个参数可选值为：  kScrollerStatusStart=0;  kScrollerStatusMoving=1;  kScrollerStatusStop=2; |
| Scroller.dtor | function(self) | 析构函数。停止事件传回，释放所有控制结构 |

## ViewPager

类说明：

ViewPager 封装了一个可以左右切换页面的试图。

ViewPager 继承自 Node 有，DrawingBase的一切属性。

接口说明：

|  |  |  |
| --- | --- | --- |
| 函数名 | 函数原型及参数说明 | 接口说明 |
| ViewPager.ctor | function(self,adapter,x,y,w,h)  adapter – Adapter 对象  x,y,w,h – 位置和大小 | 构造函数，创建一个ViewPager |
| ViewPager.setReboundFrames | 同scroller |  |
| ViewPager.setFlipOverFactor | 同scroller |  |
| ViewPager.setAdapter | function(self,adapter)  参数同构造函数 | 重新设置适配器，所有视图被清空重构，从第一页开始显示 |
| ViewPager.getCurIndex | function(self) | 得到当前显示的是第几页 |
| ViewPager.setPage | function(self,index)  index – 数值型 | 手动设置当前显示第几页 |
| ViewPager.setEventPageChanged | function(self,obj,func)  参数与一切回调函数相同 | 每当当前显示也发生变化时回调，回调时是这样的：  func(obj,当前显示页); |
| ViewPage.dtor | function(self) | 析构函数，释放所有子视图及自己 |

## ListView

类说明：

ListView 封装了一个可以上下移动的试图，里面可以包含大量子试图。所有子试图都是动态创建， 离开显示区域则消失。

ListView继承自 Node 有，基础节点的一切属性。

注意：

1. ListView包含的所有子视图的大小应该是完全相同的。

接口说明：

|  |  |  |
| --- | --- | --- |
| 函数名 | 函数原型及参数说明 | 接口说明 |
| ListView.ctor | function(self,adapter,x,y,w,h)  与ViewPager相同 | 构造函数，创建一个ListView |
| ListView.getScroller | function(self) | ListView内部使用scroller来接管移动，通过此接口可以得到scroller来做控制。 |
| ListView.getScrollBar | function(self) | ListView内部在需要的时候创建滚动条，用了ScrollBar，通过此接口得到内部对象，进行操作。若未创建，返回nil。 |
| ListView.setAdapter | function(self,adapter)  参数同构造函数 | 与ViewPager相同 |
| ListView.dtor | function(self) | 析构函数，释放所有子视图，然后释放自己 |

## ScorllView

类说明：

ScrollView封装了一个可以上下移动的试图，里面可以包含大量子试图。所有子试图都是在外部创建好才加进来的， 子视图的大小不必相同。

ScrollView继承自 Node 有，基础节点的一切属性。

接口说明：

|  |  |  |
| --- | --- | --- |
| 函数名 | 函数原型及参数说明 | 接口说明 |
| ScrollView.ctor | function(self ,x,y,w,h)  与ViewPager相同 | 构造函数，创建一个ScrollView |
| ScrollView.getScroller | function(self) | ScrollView内部使用scroller来接管移动，通过此接口可以得到scroller来做控制。 |
| ScrollView.getScrollBar | function(self) | ScrollView内部在需要的时候创建滚动条，用了ScrollBar，通过此接口得到内部对象，进行操作。若未创建，返回nil。 |
| ScrollView.removeChild | function(self,child) | 此函数要特别注意，这一函数只能用在删除ScrollView之前调用，清理其子视图。  也就是说一旦调用此函数ScrollView本身就进入非法状态，必须删除，否则结果未定义。 |
| ScrollView.dtor | function(self) | 析构函数，释放所有 |

## ScorllViewEx

类说明：

ScrollViewEx与ScrollView基本相同，唯一不同的地方在于，ScrollView在添加子节点时无需也不能指定子节点的位置，所有子节点的位置都是在ScrollView内部根据原有子节点依次排列所占空间自动控制的。但ScrollViewEx中子节点位置需要手动指定，ScrollViewEx中会根据这个位置和大小去计算滚动。

需要注意的地方是，要添加进ScrollView Ex的子节点的大小和位置必须先行设置，再通过addChild加入ScrollViewEx中。

接口说明：

与ScrollView相同。

## Adapter & CacheAdapter

类说明：

这两个辅助类，供ListView和ViewPager使用。CacheAdapter 做了缓冲，当在移动过程中释放页面的时候，实际并不释放只是隐藏。

接口说明：

|  |  |  |
| --- | --- | --- |
| 函数名 | 函数原型及参数说明 | 接口说明 |
| Adapter.ctor  CacheAdapter.ctor | function(self,view,data)  view – 必须是一个节点  data – table，其中的没一项都作为view的构造函数参数 | 构造函数，创建一个适配器，供ViewPager和ListView使用 |