IMCUI API 文档 0.0.1

- **IMCUI 基于 UNITY 5.3 UGUI 源码扩展而来，对于 UGUI 原生并且没有修改过的 API 不再解释（请自己参考 unity 官方 api 文档）**
- **IMCUI API 文档不会对所有 private 类型的变量&方法&属性&枚举&对象等进行注解**
- **IMCUI 并非对 UGUI 源码的所有类(Class)名全部更改为以 IMC 开头！仅限所有的控件类以及个别类**
- **IMCUI 将 UGUI 中的所有类全部合并到命名空间(namespace) IMCUI.UI 下**

目录

public enum ControlType

public enum ControlType
控件类型枚举，会在所有控件的 Awake 函数执行时赋值
用途：可以判断该对象的控件类型

public enum ContainerType

public enum ContainerType
容器类型枚举，会在所有控件的 Awake 函数执行时赋值
用途：可以判断该对象的容器类型

IMCUIBehaviour (基类)

public abstract class IMCUIBehaviour : MonoBehaviour

OnDestroy()

protected virtual void OnDestroy()
即将销毁函数
在控件销毁之前调用，并且判断父级是否为容器类型，将自身从父级的容器 controls 数组中将其移除

m_controlType

protected ControlType m_controlType;
控件类型枚举变量
用途：继承此类可以设置该控件的控件类型

controlType

public ControlType controlType
控件类型枚举属性【此属性只有 get 方法】
用途：方便外部调用该属性进行判断/归类

m_containerType

protected ContainerType m_containerType;
容器类型枚举变量
用途：继承此类可以设置该控件的容器类型

containerType

```
public ContainerType containerType
```

容器类型枚举属性【此属性只有 get 方法】
用途：方便外部调用该属性进行判断/归类

parent

```
public IMCUIBehaviour parent
```

父级属性【此属性与 transform.parent 并非一个，此属性的类型为 IMCUIBehaviour 类型】
用途：容器类 （IMCUIContainer）用于设置非容器类控件设置容器类空间为父级

blocker

```
public IMCBlocker blocker
```

阻拦器属性
用途：指定该控件的对应的阻拦器控件

initialize

```
protected bool initialize = false;
```

是否初始化变量？
用途：控制控件禁止多次初始化方法调用

rectTransform

```
public RectTransform rectTransform
```

矩形变换属性【此属性只有 get 方法】
用途：可以对 UI 控件进行缩放/位移/设置锚点/设置 pivot

customID

```
public string customID;
```

自定义 ID 变量
用途：可以设置指定的 customID，用于判断当前的控件是否为目标控件

intanceID

```
public int intanceID
```

系 unity 自带 API，只不过在该类中添加，方便使用【该属性只有 get 方法】

用途：instanceID 是每一个 Object 独有的，用这个 instanceID 可以准确的找到指定的 Object

active

```
public bool active
```

激活属性
用途：用此属性可以设置/获取控件的激活状态

raycast

```
public virtual bool raycast
```

射线属性
用途：用此属性可以设置控件射线的开关状态【对于多层级 UI，会遍历子物体】

graphics

```
protected Graphic[] graphics
```

图形数组属性【此属性只有 get 方法】
用途：？？？

alpha

```
public virtual float alpha
```

α 透明度属性【该属性为虚属性】
用途：重写此属性自定义属性效果

interact

```
public virtual bool interact
```

交互属性【该属性为虚属性】
用途：重写此属性自定义属性效果

Initialize()

```
public virtual void Initialize()
```

初始化函数，在由 IMCCanvas 控件 Awake 函数触发自身以下所有 UI 的初始化函数【该函数为虚函数】
用途：重写此函数自定义初始化效果【一般控件都已经重写此函数】

UnInit(int time = 0)

```
public virtual void UnInit(int time = 0)
```

卸载函数，销毁控件，参数 time 变量默认为 0 即立即销毁，若大于 0 则延迟销毁【该函数为虚函数】
用途：重写次函数自定义效果

GetContainerObject()

```
public virtual IMCUIContainer GetContainerObject()
```

获得容器对象函数【该函数为虚函数】
用途：？？？

T GetBaseType<T>()

```
public virtual T GetBaseType<T>()
```

返回组件的类型函数【该函数为虚函数】
用途：？？？

GetSiblintIndex()

```
public virtual int GetSiblintIndex()
```

获取组件当前 Hierarchy 面板中的层级函数【该函数为虚函数】
用途：用于获取层级判断层级关系

SetSiblingIndex(int index)

```
public virtual void SetSiblingIndex(int index)
```

设置组件当前 Hierarchy 面板中的层级函数【该函数为虚函数】
用途：用于设置层级

SetAsFirstSibling()

```
public virtual void SetAsFirstSibling()
```

将组件移动至当前 Hierarchy 面板层级最上方【该函数为虚函数】
用途:用于设置层级

SetAsLastSibling()

```
public virtual void SetAsLastSibling()
```

将组件移动至当前 Hierarchy 面板层级最下方【该函数为虚函数】
用途：用于设置层级

Unparent()

```
public IMCUIBehaviour Unparent()
```

解除父子关系函数，并返回组件
用途:解除父子关系

TweenCallBack

```
public UnityAction TweenCallBack;
```

渐变回调？委托
用途：执行 MoveTo RotationTo ScaleTo AlphaTo 函数时回调

MoveTo(Vector3 position, float time, UnityAction callBack = null)

```
public void MoveTo(Vector3 position, float time, UnityAction callBack = null)
```

位移动画函数，参数 position 为目标地点，time 为动画时间，callBack 为执行完回调【起始位置是以当前位置为准】
用途：位移操作

RotationTo(Vector3 angle, float time, UnityAction callBack = null)

```
public void RotationTo(Vector3 angle, float time, UnityAction callBack = null)
```

旋转动画函数，参数 angle 为目标角度，time 为动画时间，callBack 为执行完回调【起始角度是以当前角度为准】
用途：旋转操作

ScaleTo(Vector3 size, float time, UnityAction callBack = null)

```
public void ScaleTo(Vector3 size, float time, UnityAction callBack = null)
```

缩放动画函数，参数 size 为目标缩放值，time 为动画时间，callBack 为执行完回调【起始 scale 值是以当前大小(size)值为准】
用途：缩放操作

AlphaTo(float targetAlpha, float time, UnityAction callBack = null)

```
public virtual void AlphaTo(float targetAlpha, float time, UnityAction callBack = null)
```

阿尔法动画函数（透明度动画函数），参数 targetAlpha 为目标阿尔法值【0~1 之间】，time 为动画时间，callBack 为执行完回调【起始阿尔法值为当前阿尔法值为准】【该函数为虚函数】
用途：渐显渐隐操作

anchoredPosition3D

```
public Vector3 anchoredPosition3D
```

锚点 3D 位置属性
用途：设置自身在 UI 画布（Canvas）上的 3D 位置

anchoredPosition

public Vector2 anchoredPosition
锚点 2D 位置属性
用途：设置自身在 UI 画布（Canvas）上的 2D 位置

anchorMax

public Vector2 anchorMax
锚点最大值属性
用途：更改锚点

anchorMin

public Vector2 anchorMin
锚点最小值属性
用途：更改锚点

offsetMax

public Vector2 offsetMax
最大偏移量属性
用途：用于计算 UI 相对右上角锚点的偏移

offsetMin

public Vector2 offsetMin
最小偏移量属性
用途：用于计算 UI 相对做小脚锚点的偏移

pivot

public Vector2 pivot
中心点属性
用途：用于设置/获取中心点（小蓝圈）的位置

rect

public Rect rect
矩阵属性
用途：可以设置/获取当前 UI 的矩阵信息（x,y,width,height）

sizeDelta

public Vector2 sizeDelta
大小增量属性
用途：可以设置/获取当前 UI 的大小

GetLocalCorners(Vector3[] fourCornersArray)

public void GetLocalCorners(Vector3[] fourCornersArray)
获取折角的本地坐标函数【获取矩阵的四个点位置】
用途：？？

GetWorldCorners(Vector3[] fourCornersArray)

public void GetWorldCorners(Vector3[] fourCornersArray)
获取折角的世界坐标函数【获取矩阵的四个点位置】
用途：？？

SetInsetAndSizeFromParentEdge(RectTransform.Edge edge, float inset, float size)

SetInsetAndSizeFromParentEdge(RectTransform.Edge edge, float inset, float size)
设置从父变换边缘插入和尺寸函数【edge 对齐方式，inset 距离边缘距离，size 宽度】
用途：用途设置 UI 的位置

SetSizeWithCurrentAnchors(RectTransform.Axis axis, float size)

public void SetSizeWithCurrentAnchors(RectTransform.Axis axis, float size)
设置当前锚点大小函数【axis 指定轴，size 宽度】
用途:用于设置 UI 的大小

UIMirror(bool isMirror)

public void UIMirror(bool isMirror)
镜像 UI 函数【isMirror 为 flase 为镜像 ui】【影响子集】
用途：用于设置 UI 镜像翻转显示

IMCText

Awake()

```
protected override void Awake()
```

重写 Awake 函数，更改自身控件类型枚举和容器类型枚举变量【该函数为虚函数】
用途：重写该函数设置自定义功能

alpha

```
public override float alpha
```

透明度属性【取值范围 0~1】【该属性为虚属性】
用途：更改组件的透明度

interact

```
public override bool interact
```

交互开关属性【该属性为虚属性】
用途：？？

raycast

```
public override bool raycast
```

射线开关属性【该属性为虚属性】
用途：判断该控件是否需要射线检测

cachedTextGenerator

```
public TextGenerator cachedTextGenerator
```

缓存文本生成器属性
用途：可以获取很多该控件有用的属性，比如文字符号数量【换行也算一个字符】、顶点数量、控件大小、行数、字号等等

IMCImage

SetNativeSize()

public override void SetNativeSize()
设置本地大小函数【该函数为虚函数】
用途：可以根据 sprite 大小将控件也缩放至对应大小

Awake()

protected override void Awake()
重写 Awake 函数，更改自身控件类型枚举和容器类型枚举变量【该函数为虚函数】
用途：重写该函数设置自定义功能

alpha

public override float alpha
透明度属性【取值范围 0~1】【该属性为虚属性】
用途：更改组件的透明度

interact

public override bool interact
交互开关属性【该属性为虚属性】
用途：？？

raycast

public override bool raycast
射线开关属性【该属性为虚属性】
用途：判断该控件是否需要射线检测

SaveImageToPath(string path)

public bool SaveImageToPath(string path)
保存 Image 的 path 图片到指定路径函数【依赖于 FileSystem 脚本】
用途：可以保存 Image 的 Sprite 图片到指定路径下

IMCRawImage

Awake()

```
protected override void Awake()
```

重写 Awake 函数，更改自身控件类型枚举和容器类型枚举变量【该函数为虚函数】
用途：重写该函数设置自定义功能

alpha

```
public override float alpha
```

透明度属性【取值范围 0~1】【该属性为虚属性】
用途：更改组件的透明度

interact

```
public override bool interact
```

交互开关属性【该属性为虚属性】
用途：？？

raycast

```
public override bool raycast
```

射线开关属性【该属性为虚属性】
用途：判断该控件是否需要射线检测

IMCButton

Text

```
public IMCText Text
```

对 button 子物体 text 控件的索引属性【该属性为只读属性】
用途：获取 text 控件，对其控件进行操作

Awake()

```
protected override void Awake()
```

重写 Awake 函数，更改自身控件类型枚举和容器类型枚举变量【该函数为虚函数】
用途：重写该函数设置自定义功能

alpha

```
public override float alpha
```

透明度属性【取值范围 0~1】【该属性为虚属性】
用途：更改组件的透明度

interact

```
public override bool interact
```

交互开关属性【该属性为虚属性】
用途：？？

raycast

```
public override bool raycast
```

射线开关属性【该属性为虚属性】
用途：判断该控件是否需要射线检测

Initialize()

```
public override void Initialize()
```

重写初始化函数，使其只能进行一次初始化操作【该函数为虚函数】
用途：需要初始化的代码在重写该函数后添加

AddListener(UnityAction action)

```
public void AddListener(UnityAction action)
```

添加侦听委托函数，参数 action 为无参委托
用途：动态添加委托

RemoveListener(UnityAction action)

```
public void RemoveListener(UnityAction action)
```

删除侦听函数，参数 action 为需要删除的委托
用途：动态删除委托

RemoveAllListener()

```
public void RemoveAllListener()
```

删除全部侦听函数
用途：动态清除所有的委托

StartWithinStipulatedTimeCall(float time, UnityAction action)

```
public void StartWithinStipulatedTimeCall(float time, UnityAction action)
```

按住抬起在规定时间内回调函数，参数 time 为时间，action 为需要绑定的无参回调【执行完一次后自动清空回调】
用途：用于特定情况下调用

StopWithinStipulatedTimeCall()

```
public void StopWithinStipulatedTimeCall()
```

关闭 StartWithinStipulatedTimeCall 函数所执行的操作
用途：用于停止 StartWithinStipulatedTimeCall 的绑定

StartOutsideStipulatedTimeCall(float time, UnityAction<GameObject> action)

```
public void StartOutsideStipulatedTimeCall(float time, UnityAction<GameObject> action)
```

按住抬起在规定时间后回调函数，参数 time 为时间，action 为有参回调【回调参数为按钮自身的 gameobject】【执行完一次后自动清空回调】
用途：用于特定情况下调用

StopOutsideStipulatedTimeCall()

```
public void StopOutsideStipulatedTimeCall()
```

关闭 StartOutsideStipulatedTimeCall 函数所执行的操作
用途：用于停止 StartOutsideStipulatedTimeCall 的绑定

AddOnPointerLongListener(UnityAction<GameObject> action)

```
public void AddOnPointerLongListener(UnityAction<GameObject> action)
```

添加长按委托侦听事件函数，参数 action 为需要添加的委托
用途：动态添加长按事件委托

RemoveOnPointerLongListener(UnityAction<GameObject> action)

```
public void RemoveOnPointerLongListener(UnityAction<GameObject> action)
```

删除长按委托侦听事件函数，参数 action 为需要删除的委托
用途：动态删除长按事件委托

StartDelayCall(float time, UnityAction action)

```
public void StartDelayCall(float time, UnityAction action)
```

延迟执行函数，按下后延迟 time 秒后回调，参数 time 为时间，action 为无参回调【执行完一次后自动清空回调】
用途：用途特定情况下调用

StopDelayCall()

```
public void StopDelayCall()
```

关闭 StartDelayCall 函数所执行的操作
用途：用于停止 StartDelayCall 的绑定

onPointerLongPress

```
public UnityAction<GameObject> onPointerLongPress;
```

StartOutsideStipulatedTimeCall 函数所绑定委托
用途：更改 onPointerLongPress 回调委托

longpressTime

```
public float longpressTime;
```

长按时间变量
用途：更改长按时间

LongPressEnable

```
public bool LongPressEnable = false;
```

长按事件开关变量
用途：更改长按事件是否触发开关

isZoomAnimation

```
public bool isZoomAnimation
```

是否缩放动画属性
用途：更改按钮是否需要缩放动画

isPlayAudio

```
public bool isPlayAudio
```

是否播放音频属性
用途：更改该属性确定当前按钮是否有按钮点击音效

clip

```
public AudioClip clip
```

按钮点击音效属性
用途：添加该音效则按钮触发当前音效否则触发 IMCCanvas 组件下的“统一”按钮音效

Create(string buttonTitle = "Button", Sprite backgroundPicture = null)

```
public void Create(string buttonTitle = "Button", Sprite backgroundPicture = null)
```

创建函数，参数 buttonTitle 为 button.text.text 的内容，backgroundPicture 为 button.image.sprite 的图片
用途：动态更改按钮

Create(UnityAction call, string buttonTitle = "Button")

```
public void Create(UnityAction call, string buttonTitle = "Button")
```

创建函数，参数 call 为 button.ocClick 绑定的委托【此委托是添加到委托列表中的，并非直接赋值】,buttonTitle 为 button.text.text 的内容
用途：动态更改按钮

Create(UnityAction call, string buttonTitle = "Button", Sprite backgroundPicture = null)

```
public void Create(UnityAction call, string buttonTitle = "Button", Sprite
```

backgroundPicture = null)
创建函数，参数 call 为 button.ocClick 绑定的委托【此委托是添加到委托列表中的，并非直接赋值】，buttonTitle 为 button.text.text 的内容， backgroundPicture 为 button.image.sprite 的图片
用途：动态更改按钮

OnPointerEnter(PointerEventData eventData)

public override void OnPointerEnter(PointerEventData eventData)
鼠标介入函数， 参数 eventData 为每次触摸事件都会创建其中一个包含所有相关信息的事件【此函数为虚函数】
用途：？？？

OnPointerExit(PointerEventData eventData)

鼠标离开函数， 参数 eventData 为每次触摸事件都会创建其中一个包含所有相关信息的事件【此函数为虚函数】
用途：？？？

IMCToggle

Awake()

```
protected override void Awake()
```

重写 Awake 函数，更改自身控件类型枚举和容器类型枚举变量【该函数为虚函数】
用途：重写该函数设置自定义功能

alpha

```
public override float alpha
```

透明度属性【取值范围 0~1】【该属性为虚属性】
用途：更改组件的透明度

interact

```
public override bool interact
```

交互开关属性【该属性为虚属性】
用途：？？

raycast

```
public override bool raycast
```

射线开关属性【该属性为虚属性】
用途：判断该控件是否需要射线检测

Initialize()

```
public override void Initialize()
```

重写初始化函数，使其只能进行一次初始化操作【该函数为虚函数】
用途：需要初始化的代码在重写该函数后添加

label

```
public IMCText label
```

标签属性，获取该控件下的标签控件【该属性为只读属性】
用途：更改标签内容

AddListener(UnityAction<bool> action)

```
public void AddListener(UnityAction<bool> action)
```

添加侦听委托函数，参数 action 为需要添加的侦听委托
用途：动态添加委托函数

RemoveListener(UnityAction<bool> action)

```
public void RemoveListener(UnityAction<bool> action)
```

删除侦听委托函数，参数 action 为需要删除的侦听委托
用途：动态删除委托函数

RemoveAllListener()

```
public void RemoveAllListener()
```

删除全部侦听委托函数
用途：动态删除全部委托函数

isPlayAudio

```
public bool isPlayAudio
```

是否播放属性
用途：更改属性确定该控件是否有点击音效

clip

```
public AudioClip clip
```

音频属性
用途：添加该音效则按钮触发当前音效否则触发 IMCCanvas 组件下的“统一”按钮音效

IMCSlider

value

```
public virtual float value
```

数值属性，【更改 set 方法，赋值时不会触发回调。更改为鼠标点击后触发回调】【该属性为虚属性】
用途：动态更改滑动条状态

Awake()

```
protected override void Awake()
```

重写 Awake 函数，更改自身控件类型枚举和容器类型枚举变量【该函数为虚函数】
用途：重写该函数设置自定义功能

alpha

```
public override float alpha
```

透明度属性【取值范围 0~1】【该属性为虚属性】
用途：更改组件的透明度

interact

```
public override bool interact
```

交互开关属性【该属性为虚属性】
用途：？？

raycast

```
public override bool raycast
```

射线开关属性【该属性为虚属性】
用途：判断该控件是否需要射线检测

Initialize()

```
public override void Initialize()
```

重写初始化函数，使其只能进行一次初始化操作【该函数为虚函数】
用途：需要初始化的代码在重写该函数后添加

IMCScrollbar

Awake()

```
protected override void Awake()
```

重写 Awake 函数，更改自身控件类型枚举和容器类型枚举变量【该函数为虚函数】
用途：重写该函数设置自定义功能

alpha

```
public override float alpha
```

透明度属性【取值范围 0~1】【该属性为虚属性】
用途：更改组件的透明度

interact

```
public override bool interact
```

交互开关属性【该属性为虚属性】
用途：？？

raycast

```
public override bool raycast
```

射线开关属性【该属性为虚属性】
用途：判断该控件是否需要射线检测

Initialize()

```
public override void Initialize()
```

重写初始化函数，使其只能进行一次初始化操作【该函数为虚函数】
用途：需要初始化的代码在重写该函数后添加

IMCDropdown

OptionData

public class OptionData
选项数据类【该类为内部类】
用途：用于设置 dropdown 下每个 Item 的数据类

OptionData.itemToggle

public IMCToggle itemToggle
Item toggle（item 自身）控件属性
用途：设置 item 上的 toggle 控件

OptionData.itemLabel

public IMCText itemLabel
Item Label 控件属性
用途：设置 item 控件下 ItemLabel 控件

OptionData.itemBackground

public IMCImage itemBackground
Item Background 控件属性
用途：设置 item 控件下 Item Background 控件

OptionData.itemCheckmark

public IMCImage itemCheckmark
Item Checkmark 控件属性
用途：设置 item 控件下 Item Checkmark 控件

ShowItemAction

public UnityAction ShowItemAction = null;
Item 显示委托，显示下拉列表时触发的委托回调
用途：显示下拉列表时需要触发事件绑定此委托

HideItemAction

public UnityAction<int> HideItemAction = null;

Item 隐藏委托，隐藏下拉列表时触发的委托回调
用途：隐藏下拉列表时需要触发事件绑定此委托

onValueChanged

```
public DropdownEvent onValueChanged
```

值发生变化事件，当 value 发生变化时触发的事件回调【当 value 没有发生变化的时候不触发事件】
用途：value 变换需要触发事件绑定此事件

onHideChanged

```
public DropdownEvent onHideChanged
```

值发生变化事件，当 value 被赋值时触发的事件回调【无论手动触发还是代码触发】
用途：value 被赋值时触发事件绑定此事件

onSelect

```
public DropdownEvent onSelect
```

选择事件，当下拉列表显示时，修改 value 触发事件回调
用途：显示下拉列表时，value 被赋值时需要触发事件绑定此事件

isShow

```
public bool isShow
```

是否显示属性，是否显示下拉列表
用途：用于判断当前下拉列表是否显示

alpha

```
public override float alpha
```

透明度属性【取值范围 0~1】【该属性为虚属性】
用途：更改组件的透明度

interact

```
public override bool interact
```

交互开关属性【该属性为虚属性】
用途：？？

raycast

```
public override bool raycast
```

射线开关属性【该属性为虚属性】

用途：判断该控件是否需要射线检测

IMCInputField

Awake()

```
protected override void Awake()
```

重写 Awake 函数，更改自身控件类型枚举和容器类型枚举变量【该函数为虚函数】
用途：重写该函数设置自定义功能

alpha

```
public override float alpha
```

透明度属性【取值范围 0~1】【该属性为虚属性】
用途：更改组件的透明度

interact

```
public override bool interact
```

交互开关属性【该属性为虚属性】
用途：？？

raycast

```
public override bool raycast
```

射线开关属性【该属性为虚属性】
用途：判断该控件是否需要射线检测

Initialize()

```
public override void Initialize()
```

重写初始化函数，使其只能进行一次初始化操作【该函数为虚函数】
用途：需要初始化的代码在重写该函数后添加

IMCCanvas

Name

```
public string Name = "";
```
名字变量
用途：可以把它看做是一个标签，可以通过这个名字查找对应的 canvas

buttonClickClip

```
public AudioClip buttonClickClip;
```
按钮点击音效
用途：可以使需要触发点击音效的按钮播放这个音效

forms

```
public List<IMCForm> forms = new List<IMCForm>();
```
IMCForm 集合
用途：可以通过该集合查找 IMCForm

PlayAudio(AudioClip clip)

```
public void PlayAudio(AudioClip clip)
```
播放音频函数【外界调用】
用途：播放按钮点击音效

RegisterForm(IMCForm form)

```
public void RegisterForm(IMCForm form)
```
添加 form 函数，参数 form 为需要添加的对应 form
用途：可以动态向 forms 集合中添加 form

UnRegisterForm(IMCForm form)

```
public void UnRegisterForm(IMCForm form)
```
移除 form 函数，参数 form 为需要移除的对应 form
用途：可以动态从 forms 集合中移除 form

FindFormByName(string name)

```
public IMCForm FindFormByName(string name)
```

通过 form 的 name 属性获取 form 函数，参数 name 为目标 form 的 name 属性【如果没有符合条件的元素，返回 null】
用途：可以查找 form

FindFormsByName(string name)

```
public List<IMCForm> FindFormsByName(string name)
```

通过 form 的 name 属性获取 form 集合函数，参数 name 为目标 form 的 name 属性【如果没有符合条件的元素，返回长度为 0 的 IMCForm 类型的集合】
用途：可以查找多个同名 form

FindFormByByte<T>() where T:IMCForm

```
public T FindFormByByte<T>() where T:IMCForm
```

通过泛型查找 forms 集合中的 form 函数【限制泛型 T 类型为 IMCForm 的子类】
用途：可以查找 form

FindFormsByByte<T>() where T : IMCForm

```
public List<T> FindFormsByByte<T>() where T : IMCForm
```

通过 form 的类型获取 form 集合函数，泛型 T 为目标 form 的类型【如果没有符合条件的元素，返回长度为 0 的 IMCForm 类型的集合】
用途：可以查找多个同类型 form

RaycastSwitch(bool isSwitch)

```
public void RaycastSwitch(bool isSwitch)
```

射线开关函数，参数 isSwitch 为开关变量
用途：可以设置当前 IMCCanvas 下的所有 ui 能否被射线检测到

IMCMessageBox

MessageBoxButtonAttribute

public struct MessageBoxButtonAttribute
消息框按钮属性结构体
用途：？？？

background

public IMCImage background;
背景图对象
用途：对该控件的背景图进行操作

title

public IMCText title;
标题对象
用途：对该控件的标题进行操作

content

public IMCText content;
内容对象
用途：对该控件的文字内容进行操作

buttonParent

public GameObject buttonParent;
按钮父级（载体）对象
用途：对三个按钮统一进行管理

closeButton

public IMCButton closeButton;
关闭按钮对象
用途：可以获取该对象对其进行绑定回调，隐藏显示等操作

closeObject

被关闭对象

用途：关闭 MessageBox 时可以销毁（Destroy）一个目标对象

btns

```
public List<IMCButton> btns = new List<IMCButton>();
```

buttonParent 下的 Button 集合【在 Awake 时获取】
用途：可以对集合内元素进行自定义操作

Awake()

```
protected override void Awake()
```

重写 Awake 函数，更改自身控件类型枚举和容器类型枚举变量【该函数为虚函数】
用途：重写该函数设置自定义功能

alpha

```
public override float alpha
```

透明度属性【取值范围 0~1】【影响子物体】【该属性为虚属性】
用途：更改自身的透明度

interact

```
public override bool interact
```

交互开关属性【影响子物体】【该属性为虚属性】
用途：？？

raycast

```
public override bool raycast
```

射线开关属性【影响子物体】【该属性为虚属性】
用途：判断该控件是否需要射线检测

Deploy(UnityAction<int> choiceBack, string messageBoxTitle, string messageBoxContent, Sprite messageBoxBackground, MessageBoxButtonAttribute[] messageBoxAttribute)

```
public void Deploy(UnityAction<int> choiceBack, string messageBoxTitle, string messageBoxContent, Sprite messageBoxBackground, MessageBoxButtonAttribute[] messageBoxAttribute)
```

部署 MessageBox 函数，参数 choiceBack 为有参回调，messageBoxTitle 为控件标题，messageBoxContent 为控件文字内容，messageBoxBackground 为控件背景图，messageBoxAttribute

为对每个 btns 数组里的按钮单独属性赋值
用途：设置 MessageBox

Deploy(UnityAction<int> choiceBack)

public void Deploy(UnityAction<int> choiceBack)
部署 MessageBox 函数，参数 choiceBack 为有参回调【返回的参数是 btns 数组中对应的 button 索引号】
用途：设置 MessageBox

Deploy(UnityAction<int> choiceBack, string messageBoxTitle, string messageBoxContent)

public void Deploy(UnityAction<int> choiceBack, string messageBoxTitle, string messageBoxContent)
部署 MessageBox 函数，参数 choiceBack 为有参回调【返回的参数是 btns 数组中对应的 button 索引号】，messageBoxTitle 为控件标题，messageBoxContent 为控件文字内容
用途：设置 MessageBox

Deploy(UnityAction<int> choiceBack, string messageBoxTitle, string messageBoxContent, Sprite messageBoxBackground)

public void Deploy(UnityAction<int> choiceBack, string messageBoxTitle, string messageBoxContent, Sprite messageBoxBackground)
部署 MessageBox 函数，参数 choiceBack 为有参回调【返回的参数是 btns 数组中对应的 button 索引号】，messageBoxTitle 为控件标题，messageBoxContent 为控件文字内容，messageBoxBackground 为控件背景图
用途：部署 MessageBox

SetButton(int buttonIndex, string buttonTitle = "", Sprite buttonSprite = null, UnityAction buttonAction = null)

public void SetButton(int buttonIndex, string buttonTitle = "", Sprite buttonSprite = null, UnityAction buttonAction = null)
设置按钮函数，参数 buttonIndex 为按钮所在 btns 数组中的索引查找对应的按钮，buttonTitle 为按钮的 text 控件内容，buttonSprite 为按钮的 image 控件的 sprite 图片，buttonAction 为按钮所需要添加绑定的委托【针对于 btns 数组中的函数，不包含 MessageBox 的 closeBtn 控件】
用途：设置 btns 数组中的按钮

SetButton(int buttonIndex, MessageBoxButtonAttribute buttonAttribute)

public void SetButton(int buttonIndex, MessageBoxButtonAttribute buttonAttribute)

设置按钮函数，参数 buttonIndex 为按钮所在 btns 数组中的索引查找对应的按钮，参数 buttonAttribute 为消息框按钮属性结构体【针对于 btns 数组中的函数，不包含 MessageBox 的 closeBtn 控件】
用途：设置 btns 数组中的按钮

SetButton(IMCButton buttonObject, string buttonTitle = "", Sprite buttonSprite = null, UnityAction buttonAction = null)

```
public void SetButton(IMCButton buttonObject, string buttonTitle = "", Sprite buttonSprite = null, UnityAction buttonAction = null)
```

设置按钮函数,参数 buttonObject 为目标按钮,buttonTitle 为按钮的 text 控件内容,buttonSprite 为按钮的 image 控件的 sprite 图片，buttonAction 为按钮所需要添加绑定的委托【针对于 btns 数组中的函数，不包含 MessageBox 的 closeBtn 控件】

RemovButtonElement(int buttonIndex)

```
public void RemovButtonElement(int buttonIndex)
```

移除按钮元素函数，参数 buttonIndex 为 btns 数组中的索引
用途：销毁 btns 中的指定元素

RemoveButtonElement(IMCButton buttonObject)

```
public void RemoveButtonElement(IMCButton buttonObject)
```

移除按钮元素函数，参数 buttonObject 为 btns 数组中的按钮对象
用途：销毁 btns 中的置指定元素

IMCCascadeDropdown

public class IMCCascadeDropdown : IMCUIBehaviour
级联·下拉控件

dropDowns

public List<IMCDropdown> dropDowns
下拉控件集合
用途:可以自定义集合内的元素

Awake()

protected override void Awake()
重写 Awake 函数，更改自身控件类型枚举和容器类型枚举变量【该函数为虚函数】
用途：重写该函数设置自定义功能

alpha

public override float alpha
透明度属性【取值范围 0~1】【影响 dropDowns 集合中的元素】【该属性为虚属性】
用途：更改自身的透明度

interact

public override bool interact
交互开关属性【影响 dropDowns 集合中的元素】【该属性为虚属性】
用途：？？

raycast

public override bool raycast
射线开关属性【影响 dropDowns 集合中的元素】【该属性为虚属性】
用途：判断该控件是否需要射线检测

GetDropDownsIndex(IMCDropdown dropDown)

public int GetDropDownsIndex(IMCDropdown dropDown)
获取下拉控件索引函数，参数 dropdown 为 dropDowns 集合中的对应下拉控件【如果没有对应索引，则返回-1】
用途：通过参数 dropDown 返回对应 dropDowns 数组中的索引

currentDropdown

```
public IMCDropdown currentDropdown
```

当前下拉控件属性

用途：获取当前正在工作的下拉控件

IMCSwitchButton

public class IMCSwitchButton : IMCSlider
【switchButton 为 IOS 的官方命名】

Awake()

protected override void Awake()
重写 Awake 函数，更改自身控件类型枚举和容器类型枚举变量【该函数为虚函数】
用途：重写该函数设置自定义功能

alpha

public override float alpha
透明度属性【取值范围 0~1】【影响子物体】【该属性为虚属性】
用途：更改自身的透明度

interact

public override bool interact
交互开关属性【影响子物体】【该属性为虚属性】
用途：？？

raycast

public override bool raycast
射线开关属性【影响子物体】【该属性为虚属性】
用途：判断该控件是否需要射线检测

content

public IMCText content
内容控件属性
用途：自定义该控件显示内容

onClick

public switchButtonEvent onClick = new switchButtonEvent();
点击委托【该委托为有参委托，回调参数为 switchButton 当前的 isOn】
用途：用于动态绑定委托

moveSpeed

```
public float moveSpeed
```

移动速度属性
用途：用于动态设置 Handle 的移动速度

isOn

```
public bool isOn
```

开关属性
用途：用于动态设置 switchButton 控件的开关

onContent

```
public string onContent
```

打开时显示文字属性
用途：用于设置 switchButton 控件 isOn 为 true 时显示的文字内容

offContent

```
public string offContent
```

关闭时显示文字属性
用途：用于设置 switchButton 控件 isOn 为 flase 时显示的文字内容

isContent

```
public bool isContent
```

是否打开文字显示属性
用途：用于设置 switchButton 控件的文字显示功能是否打开

isColor

```
public bool isColor
```

是否打开颜色显示属性【默认颜色为白色】
用途：用于设置 switchButton 控件的颜色显示功能是否打开

onColor

```
public Color onColor
```

打开时显示的颜色
用途：用于设置 switchButton 控件 isOn 为 true 时显示的颜色

offColor

```
public Color offColor
```

关闭时显示的颜色
用途：用于设置 switchButton 控件 isOn 为 flase 时显示的颜色

canvasGroup

```
public  CanvasGroup canvasGroup
```

画布组控件属性【该属性为只读属性】
用途：设置画布组控件自定义效果

IsOnToggle(bool ison)

```
public void IsOnToggle(bool ison)
```

isOn 开关函数,参数 isOn 为开关函数【因为继承 slider 的原因，如果需要代码调用开关的话，要用此函数】
用途：动态设置 isOn 的值

IMCOrderShowButtons

public class IMCOrderShowButtons : IMCUIBehaviour
【翻页按钮组】

targets

public List<IMCUIBehaviour> targets;
目标集合
用途：用于动态操作目标集合

leftBtn

public IMCButton leftBtn;
左按钮
用途：向左翻页

rightBtn

public IMCButton rightBtn;
右按钮
用途：向右翻页

Awake()

protected override void Awake()
重写 Awake 函数，更改自身控件类型枚举和容器类型枚举变量【该函数为虚函数】
用途：重写该函数设置自定义功能

alpha

public override float alpha
透明度属性【取值范围 0~1】【影响子物体】【该属性为虚属性】
用途：更改自身的透明度

interact

public override bool interact
交互开关属性【影响子物体】【该属性为虚属性】
用途：？？

raycast

```
public override bool raycast
```

射线开关属性【影响子物体】【该属性为虚属性】
用途：判断该控件是否需要射线检测

LeftBtnClick()

```
public void LeftBtnClick()
```

左按钮事件函数
用途：向左翻页

RightBtnClick()

```
public void RightBtnClick()
```

右按钮事件函数
用途：向右翻页

ResetTargets(int index = 0)

```
public void ResetTargets(int index = 0)
```

刷新目标函数，参数 index 为刷新后显示的 target 目标索引
用途：用于重置显示状态

IMCHyperlinksLabel

```
public class IMCHyperlinksLabel :IMCText ,IPointerDownHandler,IPointerClickHandler
```

onHrefClick

```
public HrefClickEvent onHrefClick
```

超链接点击事件【默认绑定打开 OpenURL 函数】
用途：用于绑定超链接被点击时触发的委托

hyperlinksLaberFontSize

```
public int hyperlinksLaberFontSize
```

超链接文字大小属性
用途：更改超链接字体大小

hyperlinksLaberColor

```
public Color hyperlinksLaberColor
```

超链接字体颜色属性
用途：更改超链接文字颜色

SetHyperlinksContent(string url, string content)

```
public void SetHyperlinksContent(string url, string content)
```

设置超链接内容函数，参数 url 为 url 地址，content 为代替 url 显示的内容
用途：用于设置超链接控件内容

SetHyperlinksContent(string url, string content, Color contentColor, int contentFontSize)

```
public void SetHyperlinksContent(string url, string content, Color contentColor, int contentFontSize)
```

设置超链接内容函数,参数 url 为 url 地址,content 为代替 url 显示的内容,contentColor 为 content 字体颜色，contentFontSize 为 content 字体大小
用途：用于设置超链接控件内容

SetHyperlinksContent(string url, string content, string contentColor, int contentFontSize)

```
public void SetHyperlinksContent(string url, string content, string contentColor, int
```

contentFontSize)
设置超链接内容函数,参数 url 为 url 地址,content 为代替 url 显示的内容,contentColor 为 content 字体颜色， contentFontSize 为 content 字体大小
用途：用于设置超链接控件内容

IMCProgressBar

```
public class IMCProgressBar : IMCUIBehaviour
```

Awake()

```
protected override void Awake()
```

重写 Awake 函数，更改自身控件类型枚举和容器类型枚举变量【该函数为虚函数】
用途：重写该函数设置自定义功能

animationStyle

```
public AnimationSyle animationStyle = AnimationSyle.None;
```

动画类型枚举
用途：更改动画的风格

value

```
public float value
```

数值属性
用途：对于个别动画状态更改数值可以显示进度条百分比

sprite

```
public Sprite sprite
```

精灵图属性【此属性为只读属性】
用途：可以更改进度条图片

background

```
public Sprite background
```

背景图属性【此属性为只读属性】
用途：可以更改背景图

tempView

```
public IMCImage tempView
```

临时视图 image 控件属性，是 sprite 属性的载体【此属性为只读属性】
用途：更改进度条等一系列自定义操作

backgroundImage

public IMCImage backgroundImage
背景图 image 控件属性【此属性为只读属性】
用途：更改背景

rotateSpeed

public float rotateSpeed = -100f;
旋转速度变量，旋转动画状态的旋转速度
用途：更改旋转速度，以及旋转方向

isTempViewCustomSize

public bool isTempViewCustomSize = false;
TempView 自定义大小开关变量
用途：此布尔变量为 true 时，tempView 控件 size 会设置为 tempViewSize

tempViewSize

public Vector2 tempViewSize
TempView 大小属性【该属性在 isTempViewCustomSize 为 true 时才被触发】
用途：更改 TempView 控件大小

isTempViewPosition

public bool isTempViewPosition = false;
TempView 自定义位置开关变量
用途：此布尔变量为 true 时，tempView 控件 position 会设置为 tempViewPosition

tempViewPosition

public Vector3 tempViewPosition
TempView 位置属性【该属性在 isTempViewPosition 为 true 时才被触发】
用途：更改 TempView 控件位置

Begin(Sprite tempViewSprite = null, Sprite backgroundSprite = null)

public void Begin(Sprite tempViewSprite = null, Sprite backgroundSprite = null)
开始函数，参数 tempViewSprite 为进度条图片，backgroundSprite 为背景图片

用途：启动进度条控件动画

Begin(AnimationStyle style, Sprite tempViewSprite = null, Sprite backgroundSprite = null)

```
public void Begin(AnimationStyle style, Sprite tempViewSprite = null, Sprite backgroundSprite = null)
```

开始函数，参数 style 为动画类型枚举，参数 tempViewSprite 为进度条图片，backgroundSprite 为背景图片
用途:启动进度条控件动画

Stop(bool isDestroy=false)

```
public void Stop(bool isDestroy=false)
```

关闭进度条，参数 isDestroy 为是否销毁自身组件
用途：关闭进度条动画控件显示

IMCUIContainer

```
public class IMCUIContainer : IMCUIBehaviour
```
【容器类】

canvasGroup

```
public CanvasGroup canvasGroup
```
画布组控件属性【此属性为只读属性】
用途：更改控件及子物体射线开关，交互开关以及 alpha 数值

image

```
public IMCImage image
```
图像控件属性【此属性为只读属性】
用途：自定义自身图片的显示状态

raycast

```
public override bool raycast
```
射线开关【影响子集】
用途：用于设置该控件下的所有子物体的射线检测是否响应

alpha

```
public override float alpha
```
透明度开关【影响子集】
用途：用于设置该控件下所有子物体的透明度

interact

```
public override bool interact
```
交互开关【影响子集】
用途：用于设置该控件下所有子物体的交互操作是否响应

Initialize()

```
public override void Initialize()
```
初始化函数【此函数为虚函数】
用途：重写此函数个性化初始化操作

GetContainerObject()

public override IMCUIContainer GetContainerObject()
获得容器函数【此函数为虚函数】
用途：获得自身

AddControl(IMCUIBehaviour control)

public virtual void AddControl(IMCUIBehaviour control)
添加子物体函数，参数 control 为需要添加的子物体【此函数为虚函数】
用途： 向容器添加子物体

AddControl(IMCUIBehaviour control, Vector3 pos)

public virtual void AddControl(IMCUIBehaviour control, Vector3 pos)
添加子物体函数，参数 control 为需要添加的子物体，pos 为相对于父级的 anchoredPosition3D 位置【此函数为虚函数】
用途：向容器添加子物体

AddControls(List<IMCUIBehaviour> list)

public virtual void AddControls(List<IMCUIBehaviour> list)
添加子物体集合函数，参数 list 为需要添加的子物体集合【此函数为虚函数】
用途：向容器添加子物体

AddControls(List<IMCUIBehaviour> list, List<Vector2> posList)

public virtual void AddControls(List<IMCUIBehaviour> list, List<Vector2> posList)
添加子物体集合函数，参数 list 为需要添加的子物体集合，posList 为 list 元素对应的相对于父级的 anchoredPosition3D 位置【此函数为虚函数】
用途：向容器添加子物体

RemoveControl(IMCUIBehaviour control)

移除子物体函数，参数 control 为需要被移除的子物体【此函数为虚函数】
用途：移除 controls 数组中对应的元素

RemoveControlAndDestroy(IMCUIBehaviour control)

移除并销毁子物体函数，参数 control 为需要被移除并销毁的子物体【此函数为虚函数】
用途：移除并销毁 control 数组中对应的元素

RemoveAllControl()

```
public virtual void RemoveAllControl()
```

移除全部子物体函数【此函数为虚函数】
用途：移除 controls 数组中全部元素

RemoveAllControlAndDestroy()

```
public virtual void RemoveAllControlAndDestroy()
```

移除并销毁全部子物体函数【此函数为虚函数】
用途：移除并销毁 contorls 数组中的全部元素

FindControl<T>()

```
public virtual T FindControl<T>()where T:IMCUIBehaviour
```

查找控件函数，泛型 T 为查找的控件类型【通过正向查找，返回第一个符合条件的元素】【此函数为虚函数】
用途：查找子物体

FindControlByCustomID(string customID)

```
public virtual IMCUIBehaviour FindControlByCustomID(string customID)
```

通过控件 CustomID 查找函数，参数 customID 为控件控件的 customID【通过正向查找，返回第一个符合条件的元素】【此函数为虚函数】
用途：查找子物体

FindControlByCustomID<T>(string customID)

```
public virtual T FindControlByCustomID<T>(string customID) where T : IMCUIBehaviour
```

通过控件 CustomID 查找函数，泛型 T 为查找的控件类型，customID 为控件控件的 customID【通过正向查找，返回第一个符合条件的元素】【此函数为虚函数】
用途：查找子物体

FindControlByName(string name)

```
public virtual IMCUIBehaviour FindControlByName(string name)
```

通过控件 name 查找函数，参数 name 为控件的 name 变量值【gameObject.name】【通过正向查找，返回第一个符合条件的元素】【此函数为虚函数】
用途：查找子物体

FindControlByName<T>(string name)

public virtual T FindControlByName<T>(string name) where T : IMCUIBehaviour
通过控件 name 查找函数，泛型 T 为查找的控件类型，参数 name 为控件的 name 变量值【gameObject.name】【通过正向查找，返回第一个符合条件的元素】【此函数为虚函数】
用途：查找子物体

FindControlByNameAndCustomID(string name, string customID)

public virtual IMCUIBehaviour FindControlByNameAndCustomID(string name, string customID)
通过控件 name 与 customID 查找函数，参数 name 为控件的 name 变量值【gameObject.name】，customID 为控件控件的 customID【通过正向查找，返回第一个符合条件的元素】【此函数为虚函数】
用途：查找子物体

FindControlByNameAndCustomID<T>(string name, string customID)

public virtual T FindControlByNameAndCustomID<T>(string name, string customID) where T : IMCUIBehaviour
通过控件 name 与 customID 查找函数，泛型 T 为控件的类型，参数 name 为控件的 name 变量值【gameObject.name】，customID 为控件控件的 customID【通过正向查找，返回第一个符合条件的元素】【此函数为虚函数】
用途：查找子物体

FindControls<T>()

public virtual List<T> FindControls<T>() where T : IMCUIBehaviour
查找子物体集合函数，泛型 T 为控件类型【此函数为虚函数】
用途：查找子物体集合

FindControlsByCustomID(string customID)

public virtual List<IMCUIBehaviour> FindControlsByCustomID(string customID)
通过控件 customID 查找函数，参数 customID 为控件的 customID【此函数为虚函数】
用途：查找子物体集合

FindControlsByCustomID<T>(string customID)

public virtual List<T> FindControlsByCustomID<T>(string customID) where T : IMCUIBehaviour
通过控件 customID 查找函数，泛型 T 为控件类型，参数 customID 为控件的 customID【此函数为虚函数】
用途：查找子物体集合

FindControlsByName(string name)

```
public virtual List<IMCUIBehaviour> FindControlsByName(string name)
```

通过控件 name 查找函数，参数 name 为控件的 name 值【gameObject.name】【此函数为虚函数】
用途：查找子物体集合

FindControlsByName<T>(string name)

```
public virtual List<T> FindControlsByName<T>(string name) where T : IMCUIBehaviour
```

通过控件 name 查找函数，泛型 T 为控件类型，参数 name 为控件的 name 值【gameObject.name】【此函数为虚函数】
用途：查找子物体集合

FindControlsByNameAndCustomID(string name, string customID)

```
public virtual List<IMCUIBehaviour> FindControlsByNameAndCustomID(string name, string customID)
```

通过控件 name 与 custiomID 查找函数，参数 name 为控件的 name 值【gameObject.name】，customID 为控件的 customID【此函数为虚函数】
用途：查找子物体集合

FindControlsByNameAndCustomID<T>(string name, string customID)

```
public virtual List<T> FindControlsByNameAndCustomID<T>(string name, string customID) where T : IMCUIBehaviour
```

通过控件 name 与 custiomID 查找函数，泛型 T 为控件类型，参数 name 为控件的 name 值【gameObject.name】，customID 为控件的 customID【此函数为虚函数】
用途：查找子物体集合

FindControlByIntanceID(int intanceID)

```
public virtual IMCUIBehaviour FindControlByIntanceID(int intanceID)
```

通过控件 InstanceID 查找函数，参数 instanceID 为控件 intanceID【此函数为虚函数】
用途：查找子物体

InitializeChildControls()

```
protected virtual void InitializeChildControls()
```

获取子物体函数【此函数为虚函数】
用途：？？？

GetChildControls()

```
public List<IMCUIBehaviour> GetChildControls()
```

获取子物体集合函数
用途：获取子物体集合

CanvasGroupSwitch(bool m_switch)

```
protected void CanvasGroupSwitch(bool m_switch)
```

画布组开关函数，参数 m_switch 为开关变量
用途：设置当前组件画布组开关，【包括射线，交互，透明度】

MoveHouse(IMCUIBehaviour control, IMCUIContainer form, IMCUIContainer to, Vector2 pos, int tabViewIndex = 0)

```
public static void MoveHouse(IMCUIBehaviour control, IMCUIContainer form, IMCUIContainer to, Vector2 pos, int tabViewIndex = 0)
```

搬家函数，将 control 对象从 form 容器对象中移至 to 容器对象中【如果 control 不存在于 form 容器 controls 数组中，程序不会执行任何操作。如果 to 容器是 tabView 类型容器，还需要指定 tabViewIndex 传入具体的 tabcontent 索引。】【此函数为静态函数】
用途：子物体批量更换父级时调用此函数

IMCForm

```
public class IMCForm : IMCUIContainer
```

canvas

```
public IMCCanvas canvas;
```

Canvas 变量
用途：获取当前 form 的 IMCCanvas

showOnAwake

```
public bool showOnAwake
```

开始立刻显示属性
用途：此属性决定程序运行后是否自动显示当前 form

showTargetPosition

```
public Vector3 showTargetPosition
```

Show 状态目标位置属性
用途：show 状态时 form 的显示的 anchoredPosition3D 位置

closeTargetPosition

```
public Vector3 closeTargetPosition
```

Close 状态目标位置属性
用途：Close 状态时 form 的显示的 anchoredPosition3D 位置

isBlockerToggle

```
public bool isBlockerToggle
```

是否需要阻挡器属性
用途：show 状态时是否需要阻挡器

isDontDestroy

```
public bool isDontDestroy = false;
```

是否跳转场景销毁变量
用途：为 true 时，跳转场景后 form 不销毁

blockerShowStyle

```
public IMCBlocker.ShowStyleEnum blockerShowStyle = IMCBlocker.ShowStyleEnum.Transparent;
```

阻拦器显示风格枚举变量【当 isBlockerToggle 为 true 时才会有用】
用途：设置阻拦器显示风格

InitializeAction

```
public UnityAction InitializeAction;
```

初始化委托
用途：初始化时需要执行代码绑定此委托

ShowAction

```
public UnityAction ShowAction;
```

显示委托
用途：show 状态时需要执行代码绑定此委托

CloseAction

```
public UnityAction CloseAction;
```

关闭委托
用途：close 状态时需要执行代码绑定此委托

UnInitAction

```
public UnityAction UnInitAction;
```

注销委托
用途：注销状态时需要执行代码绑定此委托

blockerCallBack

```
public UnityAction blockerCallBack;
```

阻拦器回调委托
用途：阻拦器触发后需要执行代码绑定此委托

showCallBack

```
public UnityEvent showCallBack;
```

显示事件
用途：show 状态时需要执行代码绑定此事件

closeCallBack

```
public UnityEvent closeCallBack;
```

关闭事件
用途：close 状态时需要执行代码绑定此事件

Show()

```
public virtual void Show()
```

显示函数【此函数为虚函数】
用途：需要显示控件时调用此函数

Show(Vector3 showTargetPosition, bool blockerToggle = false, IMCBlocker.ShowStyleEnum showStyle = IMCBlocker.ShowStyleEnum.Transparent, UnityEvent showCallBack = null, UnityAction blockerCallBack = null)

```
public virtual void Show(Vector3 showTargetPosition, bool blockerToggle = false, IMCBlocker.ShowStyleEnum showStyle = IMCBlocker.ShowStyleEnum.Transparent, UnityEvent showCallBack = null, UnityAction blockerCallBack = null)
```

显示函数,参数 showTargetPosition 为 show 状态时控件的 anchoredPosition3D 位置,blockerToggle 为是否需要阻拦器，showStyle 为阻拦器显示状态时的枚举【blockerToggle 为 flase 时不生效】，showCallBack 为显示控件之后的回调委托，blockerCallBack 为阻拦器被触发后的回调委托【blockerToggle 为 flase 时不生效】【此函数为虚函数】
用途：需要显示控件时调用此函数

Close()

```
public virtual void Close()
```

关闭函数【此函数为虚函数】
用途：需要关闭控件时调用此函数

Close(Vector3 closeTargetPosition, UnityEvent closeCallBack = null)

```
public virtual void Close(Vector3 closeTargetPosition, UnityEvent closeCallBack = null)
```

关闭函数，参数 closeTargetPosition 为 close 状态时控件的 anchoredPosition3D 位置，closeCallBack 为阻拦器被触发后的回调委托【此函数为虚函数】
用途：需要关闭控件时调用此函数

UnInit(int time = 0)

```
public override void UnInit(int time = 0)
```

注销函数，参数 time 为延迟销毁时间【此函数为虚函数】
用途：注销/销毁控件时调用此函数

Awake()

```
protected override void Awake()
```

Awake 函数，重写此函数，更改控件类型枚举【此函数为虚函数】
用途：在 Awake 时需要执行代码重写此函数

Initialize()

```
public override void Initialize()
```

初始化函数，调用 controls 数组中元素的初始化函数【此函数为虚函数】
用途：在初始化时需要执行代码重写此函数

OnDestroy()

```
protected override void OnDestroy()
```

在销毁之前函数【此函数为虚函数】
用途：在销毁之前需要执行代码重写此函数

IMCScrollRect

public class IMCScrollRect : IMCUIContainer, IInitializePotentialDragHandler, IBeginDragHandler, IEndDragHandler, IDragHandler, IScrollHandler, ICanvasElement, ILayoutElement, ILayoutGroup, IPointerDownHandler

parentScrollRect

public IMCScrollRect parentScrollRect
父级 ScrollRect 属性
用途：多层级 ScrollRect 嵌套赋值此属性

Awake()

protected override void Awake()
Awake 函数，更改控件类型枚举【此函数为虚函数】
用途：在 Awake 时需要执行代码重写此函数

InitializeChildControls()

protected override void InitializeChildControls()
初始化获取 controls 元素函数【此函数为虚函数】
用途：？？？

Initialize()

初始化函数，调用 controls 数组中元素的初始化函数【此函数为虚函数】
用途：在初始化时需要执行代码重写此函数

IMCTabView

```
public class IMCTabView : IMCUIContainer
```

toggles

```
public List<IMCTabViewToggle> toggles = new List<IMCTabViewToggle>();
```

Toggle 集合
用途：增删 toggles 集合元素，查找指定元素

contents

```
public List<IMCUIBehaviour> contents = new List<IMCUIBehaviour>();
```

contents 集合
用途：增删 contents 集合元素，查找指定元素

Awake()

```
protected override void Awake()
```

Awake 函数，更改控件类型枚举【此函数为虚函数】
用途：在 Awake 时需要执行代码重写此函数

OnEnable()

```
protected override void OnEnable()
```

在激活之前函数，对 toggles 集合和 contents 进行初始化操作【此函数为虚函数】
用途：在控件激活之前需要执行代码重写此函数

TabViewControlShowOrClose(bool nuclearBombSwitch, IMCTabViewToggle tvt)

```
public void TabViewControlShowOrClose(bool nuclearBombSwitch, IMCTabViewToggle tvt)
```

选项卡视图控制显示或关闭函数，参数 nucleraBombSwitch 为每个 toggle 的 ison 值，tvt 为指定的 IMCTabViewToggle 类型控件。此函数是自定义广播函数【此函数为内部调用函数】
用途：？？？

AddControl(int index, IMCUIBehaviour control)

```
public void AddControl(int index, IMCUIBehaviour control)
```

添加 control 元素函数，参数 index 为 contents 集合索引，controls 为需要添加的控件
用途：动态添加 controls 控件调用此函数

AddControl(int index, IMCUIBehaviour control, Vector3 pos)

```
public void AddControl(int index, IMCUIBehaviour control, Vector3 pos)
```

添加 control 元素函数，参数 index 为 contents 集合索引，controls 为需要添加的控件，pos 为 controls 的 anchoredPosition3D 位置
用途：动态添加 controls 控件调用此函数

AddControl(IMCUIBehaviour content, IMCUIBehaviour control)

```
public void AddControl(IMCUIBehaviour content, IMCUIBehaviour control)
```

添加 control 元素函数，参数 content 为 contents 集合中的元素，controls 为需要添加的控件
用途：动态添加 controls 控件调用此函数

AddControl(IMCUIBehaviour content, IMCUIBehaviour control, Vector2 pos)

```
public void AddControl(IMCUIBehaviour content, IMCUIBehaviour control, Vector2 pos)
```

添加 control 元素函数，参数 content 为 contents 集合中的元素，controls 为需要添加的控件，pos 为 controls 的 anchoredPosition3D 位置
用途：动态添加 controls 控件调用此函数

InitializeChildControls()

```
protected override void InitializeChildControls()
```

初始化获取 controls 元素函数【此函数为虚函数】
用途：？？？

GetCurrentOperationContentGameObjectIndex()

```
public int GetCurrentOperationContentGameObjectIndex()
```

获取当前正在执行的 content 控件的索引函数
用途：获取当前显示的 content 控件索引

SetShowContent(int index)

```
public void SetShowContent(int index)
```

设置显示 content 函数，参数 index 为需要显示的 contents 集合索引
用途：动态设置 content 的显示

IMCGroup

```
public class IMCGroup : IMCUIContainer
```

【一个特殊的容器类型控件，因为 Form 不能嵌套 Form，所以才会有 Group 容器控件，继承容器类型所有特性】

Awake()

```
protected override void Awake()
```

Awake 函数，更改控件类型枚举【此函数为虚函数】

用途：在 Awake 时需要执行代码重写此函数

IMCBlocker

```
public class IMCBlocker : IMCUIBehaviour, IPointerClickHandler
```

Instance

```
public static IMCBlocker Instance
```

单例属性【此属性为只读属性】
用途：调用非静态函数、属性

Is_blockerPointerClick

```
public bool Is_blockerPointerClick
```

阻拦器是否完成点击属性【此属性为只读属性】
用途：判断阻拦器是否执行点击

ShowStyleEnum

```
public enum ShowStyleEnum
```

显示风格枚举
用途：设置阻拦器的显示风格

Create(List<IMCUIBehaviour> targetUis, ShowStyleEnum showStyle, UnityAction callback = null, bool isDestroy = true, Transform parent = null)

```
public IMCBlocker Create(List<IMCUIBehaviour> targetUis, ShowStyleEnum showStyle, UnityAction callback = null, bool isDestroy = true, Transform parent = null)
```

创建函数,参数 targetUis 为需要被阻拦器呈现的 ui 集合,showStyle 为阻拦器的显示风格,callback 为触发阻拦器的回调委托，isDestroy 为触发阻拦器后是否销毁函数，parent 为生成在指定控件下成为子物体
用途：创建阻拦器控件

Awake()

```
protected override void Awake()
```

Awake 函数，更改控件类型枚举，更改容器类型枚举【此函数为虚函数】
用途：在 Awake 时需要执行代码重写此函数

OnDestroy()

```
protected override void OnDestroy()
```

在销毁之前函数【此函数为虚函数】
用途：在销毁之前需要执行代码重写此函数

RemoveStack(IMCUIBehaviour removeTarget)

```
public void RemoveStack(IMCUIBehaviour removeTarget)
```

移除栈元素函数，参数 removeTarget 为需要移除的指定控件
用途：动态移除指定元素

Screenshot()

```
public Sprite Screenshot()
```

截屏函数
用途：截屏

IMCDebug

public class IMCDebug : IMCForm
【呼出手势：同一位置双击屏幕，随后点击屏幕从右往左滑动一段距离，然后再某一位置连续点击五下 AA←AAAAA】【此控件为 resource 加载，使用前确保 resource 文件夹中有此控件预制体】

Instance

public static IMCDebug Instance
单例属性【此属性为只读属性】
用途：调用非静态函数、属性

debugDatas

public List<IMCDebugCreateData> debugDatas;
debugDatas 集合【0 debug 1 error 2 warning】
用途：？？？

Log(string content, GameObject go = null)

public static void Log(string content, GameObject go = null)
打印日志函数，参数 content 为字符内容，go 为目标组件【此函数为静态函数】
用途：打印日志

LogError(string content, GameObject go = null)

public static void LogError(string content, GameObject go = null)
打印错误日志函数，参数 content 为字符内容，go 为目标组件【此函数为静态函数】
用途：打印错误日志

LogWarning(string content, GameObject go = null)

public static void LogWarning(string content, GameObject go = null)
打印警告日志函数，参数 content 为字符内容，go 为目标组件【此函数为静态函数】
用途：打印警告日志

Show()

public override void Show()
显示函数【此函数为虚函数】
用途：显示控件

Close()

```
public override void Close()
```

关闭函数【此函数为虚函数】
用途：关闭控件

Clear()

```
public void Clear()
```

清除函数，清除当前显示的 content 控件内容【此函数为按钮绑定函数】
用途：清除 IMCDebug 控制台内容

AddFontSize()

```
public void AddFontSize()
```

放大字体函数，放大当前显示的 content 空间字体【此函数为按钮绑定函数】
用途：放大 IMCDebug 控制台字体内容

ReduceFontSize()

```
public void ReduceFontSize()
```

缩小字体函数，放大当前显示的 content 空间字体【此函数为按钮绑定函数】
用途：缩小 IMCDebug 控制台字体内容

tabView

```
public IMCTabView tabView;
```

tabView 控件，【按钮操作台容器】
用途：？？？

IMCDragMove

```
public class IMCDragMove : IMCUIBehaviour, IPointerDownHandler, IDragHandler
```

panelRectTransform

```
public RectTransform panelRectTransform;
```

面板对象，被移动物体
用途：设置被拖拽物体

rangeLimit

```
public bool rangeLimit = true;
```

范围限制变量
用途：设置范围限制

IMCUIAnimation

```
public class IMCUIAnimation : MonoBehaviour
```

MoveTo(IMCUIBehaviour obj, Vector3 targetPos, float time, UnityAction callback = null)

```
public IEnumerator MoveTo(IMCUIBehaviour obj, Vector3 targetPos, float time, UnityAction callback = null)
```

位移协程，参数 obj 为需要位移的控件，targetPos 为需要位移的 anchoredPosition3D 位置，time 为移动时间（秒），callback 为移动到 targetPos 位置时回调委托
用途：位移

RotationTo(IMCUIBehaviour obj, Vector3 targetAngle, float time, UnityAction callbacke = null)

```
public IEnumerator RotationTo(IMCUIBehaviour obj, Vector3 targetAngle, float time, UnityAction callbacke = null)
```

旋转协程，参数 obj 为需要旋转的控件，targetAngle 为需要旋转的角度，time 为旋转时间(秒），callback 为旋转到 targetAngle 角度时回调委托
用途：旋转

ScaleTo(IMCUIBehaviour obj, Vector3 targetScale, float time, UnityAction callback = null)

```
public IEnumerator ScaleTo(IMCUIBehaviour obj, Vector3 targetScale, float time, UnityAction callback = null)
```

缩放协程，参数 obj 为需要缩放的控件，targetScale 为需要缩放的大小，time 为缩放时间(秒），callback 为缩放到 targetScale 大小时回调委托
用途：缩放

AlphaTo(IMCUIBehaviour obj, float alpha,float time,UnityAction callback=null)

```
public IEnumerator AlphaTo(IMCUIBehaviour obj, float alpha,float time,UnityAction callback=null)
```

透明度协程【透明度渐变】，参数 obj 为需要缩放的控件，alpha 为目标透明度值，time 为缩放时间（秒），callback 为渐变到 alpha 时回调委托
用途：透明度渐变

IMCUIResouceManager

```
public class IMCUIResouceManager : MonoBehaviour
```

LoadControl<T>(string path, Transform parent = null)

```
public static T LoadControl<T>(string path, Transform parent = null)where T:IMCUIBehaviour
```

加载控件函数，泛型 T 为 IMCUIBehaviour 限制类型控件，参数 path 为加载路径，parent 为设置指定父级【加载控件的 gameObject.name 要和泛型 T 的类型名相同】【该函数为静态函数】
用途：加载控件

LoadControl(string path, string name, Transform parent = null)

```
public static GameObject LoadControl(string path, string name, Transform parent = null)
```

加载控件函数，参数 path 为加载路径，name 为具体的控件 name【gameObject.name】，parent 为设置指定父级【该函数为静态函数】
用途：加载控件

LoadControl(string pathAndName, Transform parent = null)

```
public static GameObject LoadControl(string pathAndName, Transform parent = null)
```

加载控件函数，参数 pathAndName 为加载路径和控件 name【gameObject.name】，parent 为设置指定父级【该函数为静态函数】
用途：加载控件