#### CAControl

###### 类说明

[CAButton](CAControl/CAButton.docx)按钮的基类，主要用于定义按钮的状态和回调事件，并响应按钮的触摸事件。通常按钮有普通、高亮、选中与禁止点击四种状态。有点击、重复点击，点击后在按钮范围内滑动、点击后再按钮外部滑动、点击并在按钮内部抬起与点击按下再任意地方抬起六种按钮回调事件触发的条件。

###### 基类

[CAView](../CAView.docx)

###### 属性

|  |  |  |
| --- | --- | --- |
| 访问修饰符 | 属性名 | 说明 |
| protected | [ControlState](#ControlState) | 按钮的状态 |
| protected | [TouchEnabled](#TouchEnabled) | 按钮是否可触摸 |
| public | [Target](#Target) | 按钮的点击事件 |

###### 方法

|  |  |  |
| --- | --- | --- |
| 访问修饰符 | 属性名 | 说明 |
| public | [setControlStateNormal](#setControlStateNormal) | 设置为正常状态 |
| public | [setControlStateHighlighted](#setControlStateHighlighted) | 设置为高亮状态 |
| public | [setControlStateDisabled](#setControlStateDisabled) | 设置为不可选状态 |
| public | [setControlStateSelected](#setControlStateSelected) | 设置为选中状态 |
| public | [addTarget](#addTarget) | 添加回调事件 |
| public | [removeTarget](#removeTarget) | 移除回调事件 |

###### 属性介绍

**ControlState**

类型：CAControlState

解释：按钮的状态，枚举类型，包括正常、高亮、不可选、选中和全状态五种状态。这里所谓的全状态，即我们对按钮进行操作时，为了方便，无须对按钮的各个状态进行设置，只需设置一个全状态，即可将当前的操作应用到按钮的左右状态，get/set{}。

**TouchEnabled**

类型：bool

解释：按钮是否可触摸，get/set{}。

**Target**

类型：CCObject\*

解释：按钮点击时的回调事件

###### 方法介绍

**void** **setControlStateNormal()**

返回值：void

说明：设置按钮为正常状态，这也是按钮的默认状态。

**void** **setControlStateHighlighted()**

返回值：void

说明：设置按钮为高亮状态

**void** **setControlStateDisabled()**

返回值：void

说明：设置按钮为不可选状态

**void** **setControlStateSelected()**

返回值：void

说明：设置按钮为选中状态

**virtual void** **addTarget(CCObject\* target, SEL\_CAControl selector, CAControlEvents event)**

返回值：void

说明：为按钮添加回调事件

参数：

|  |  |  |
| --- | --- | --- |
| 类型 | 参数名 | 说明 |
| CCObject\* | target | 目标对象 |
| CCObject::\*SEL\_CAControl | selector | 函数回调器，函数指针，响应按钮点击事件，返回值为void。 |
| CAControlEvents | event | 触发事件的条件，枚举类型，包括6种触发条件。 |

示例：

void NewViewController::viewDidLoad()

{

CCRect winRect = this->getView()->getBounds();

CCLabelTTF\* viewText = CCLabelTTF::create("ViewController", "", 32);

viewText->setColor(ccGREEN);

viewText->setCenter(CCRect(winRect.size.width\*0.5,winRect.size.height\*0.5,0,0));

this->getView()->addSubview(viewText);

CAButton\* button = CAButton::createWithFrame(CCRect(0, 0, 0, 0),

CAButtonTypeCustom);

button->setFrame(CCRect(winRect.size.width\*0.5-50, winRect.size.height\*0.5+25, 100, 50));

button->addTarget(this, CAControl\_selector(NewViewController::buttonCallback),

CAControlEventTouchUpSide);

this->getView()->addSubview(button);

}

void NewViewController::buttonCallback(CAButton\* btn, CCPoint point)

{

NewViewController\* viewController = new NewViewController();

viewController->init();

viewController->setNavigationBarItem(CANavigationBarItem::create("The next view"));

viewController->setTitle("New View");

this->getNavigationController()->pushViewController(viewController,true);

viewController->autorelease();

}

函数回调器SEL\_CAControl的函数原型包含两个参数，CAControl\*和CCPoint，前者代表当前事件的触发者，后者代表当前触摸的坐标，所以回调函数必须包含这两个参数。

**virtual void** **removeTarget(CCObject\* target, SEL\_CAControl selector, CAControlEvents event)**

返回值：void

说明：移除按钮的回调事件