#### CAScheduler

###### 类说明

CrossApp中的任务调度，也就是通常所说的定时器。

###### 基类

[CCObject](../CCObject.docx)

###### 属性

|  |  |  |
| --- | --- | --- |
| 访问修饰符 | 属性名 | 说明 |
| public | [Target](#Target) | 定时器所属对象 |

###### 方法

|  |  |  |
| --- | --- | --- |
| 访问修饰符 | 方法名 | 说明 |
| public | [schedule](#schedule) | 启动定时器 |
| public | [schedule](#schedule1) | 方法重载，启动定时器 |
| public | [unschedule](#unschedule) | 移除定时器 |
| public | [unscheduleAllForTarget](#unscheduleAllForTarget) | 移除指定对象的所有定时器 |
| public | [unscheduleAll](#unscheduleAll) | 移除当前对象的所有定时器 |
| public | [isScheduled](#isScheduled) | 是否存在指定对象的定时器 |
| public | [getScheduler](#getScheduler) | 获取一个CAScheduler单例 |
| public | [pauseTarget](#pauseTarget) | 暂停指定对象的定时器 |
| public | [resumeTarget](#resumeTarget) | 恢复指定对象的定时器 |
| public | [isTargetPaused](#isTargetPaused) | 指定对象定时器是否被暂停 |
| public | [resumeTargets](#resumeTargets) | 恢复对象集合的所有定时器 |

###### 属性介绍

**Target**

类型：CCObject\*

解释：定时器的调度者，每个定时器都需要一个CCObject\*对象来启动。

###### 方法介绍

**static void** **schedule(SEL\_SCHEDULE pfnSelector, CCObject \*pTarget, float fInterval,**

**unsigned int repeat, float delay, bool bPaused)**

返回值：void

参数：

|  |  |  |
| --- | --- | --- |
| 类型 | 参数名 | 说明 |
| SEL\_SCHEDULE | pfnSelector | 定时器的回调函数 |
| CCObject\* | pTarget | 定时器所属对象 |
| float | fInterval | 调度定时器的时间间隔 |
| unsigned int | repeat | 除第一次外，调度的次数 |
| float | delay | 第一次调度前的延迟时间 |
| bool | bPaused | 定时器是否为暂停状态 |

示例：

progress = CAProgress::create();

progress->setFrame(CCRect(winRect.size.width\*0.5-100,winRect.size.height\*0.5+100,200,16));

progress->setProgresstrackColor(ccYELLOW);

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

CAScheduler::schedule(schedule\_selector(FifthViewController::changeValue),this,1,3,4,false);

void FifthViewController::changeValue(float interval)

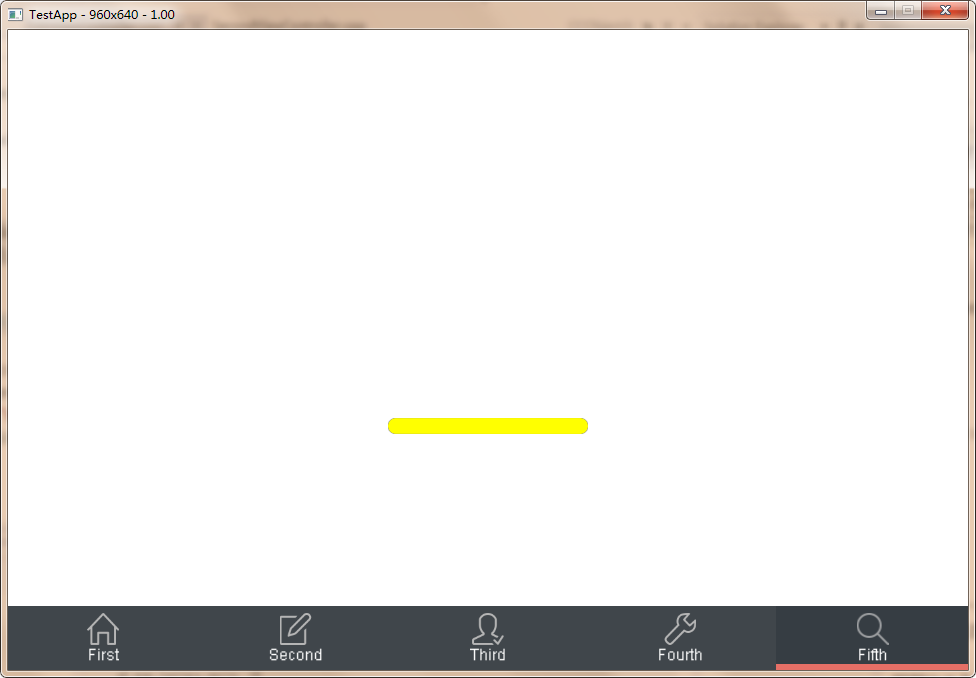
{

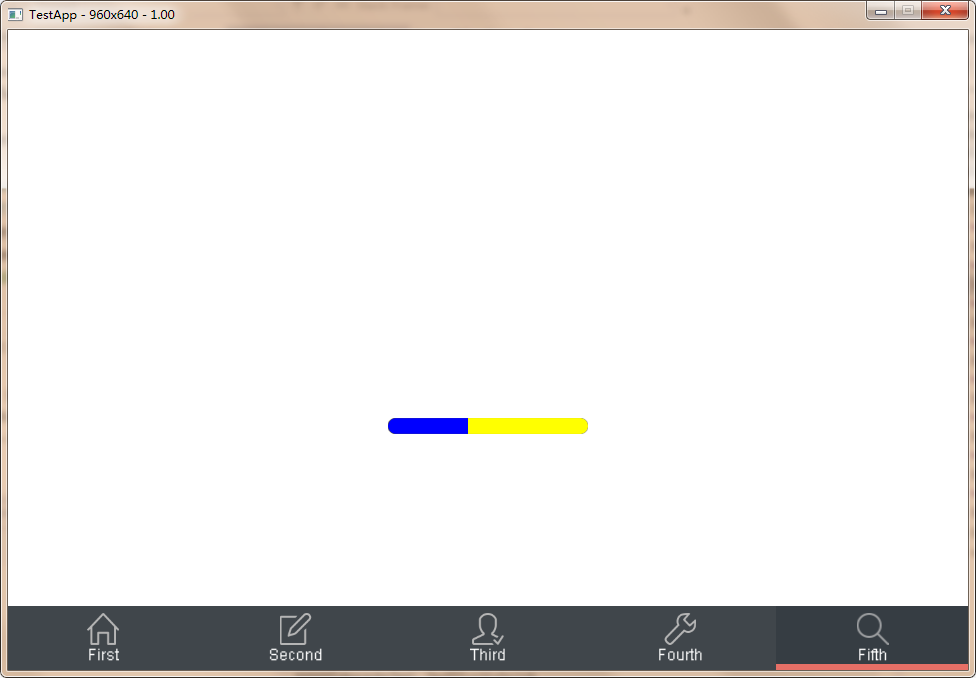
float currentValue = progress->getProgress();

progress->setProgress(currentValue+0.1);

}

设置一个定时器，调度时间间隔为1秒，延迟4秒调度，除去第一次调度外，重复调度3次。每次调度，进度条的值增加百分之10，所以进度条的值在4次调度为百分之四十。





**static void** **schedule(SEL\_SCHEDULE pfnSelector, CCObject \*pTarget, float fInterval, bool bPaused)**

返回值：void

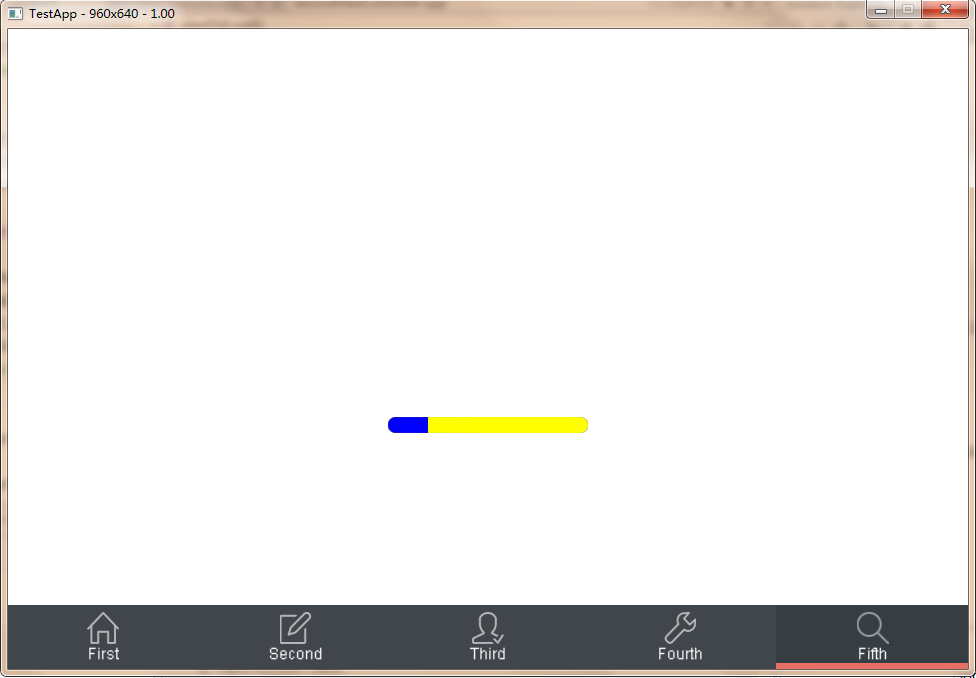
参数：

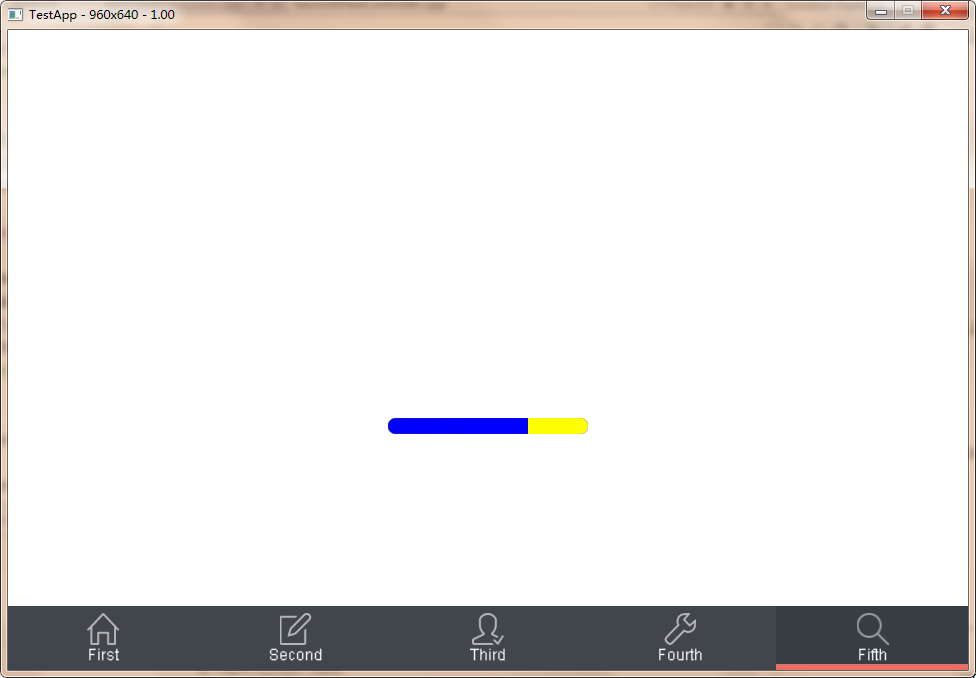
|  |  |  |
| --- | --- | --- |
| 类型 | 参数名 | 说明 |
| SEL\_SCHEDULE | pfnSelector | 定时器的回调函数 |
| CCObject\* | pTarget | 定时器所属对象 |
| float | fInterval | 调度定时器的时间间隔 |
| bool | bPaused | 定时器是否为暂停状态 |

示例：

CAScheduler::schedule(schedule\_selector(FifthViewController::changeValue),this,1,false);

设置了一个定时器，每隔1秒调度一次，进度条的值也会每隔1秒增加百分之10。





**static void** **unschedule(SEL\_SCHEDULE pfnSelector, CCObject \*pTarget)**

返回值：void

参数：

|  |  |  |
| --- | --- | --- |
| 类型 | 参数名 | 说明 |
| SEL\_SCHEDULE | pfnSelector | 定时器的回调函数 |
| CCObject\* | pTarget | 定时器所属对象 |

**static void** **unscheduleAllForTarget(CCObject \*pTarget)**

返回值：void

参数：

|  |  |  |
| --- | --- | --- |
| 类型 | 参数名 | 说明 |
| CCObject\* | pTarget | 定时器所属对象 |

**static void** **unscheduleAll(void)**

返回值：void

**static bool** **isScheduled(SEL\_SCHEDULE pfnSelector, CCObject \*pTarget)**

返回值：bool

参数：

|  |  |  |
| --- | --- | --- |
| 类型 | 参数名 | 说明 |
| SEL\_SCHEDULE | pfnSelector | 定时器的回调函数 |
| CCObject\* | pTarget | 定时器所属对象 |

**static CAScheduler\*** **getScheduler()**

解释：返回一个CAScheduler单例对象

**void** **pauseTarget(CCObject \*pTarget)**

返回值：void

参数：

|  |  |  |
| --- | --- | --- |
| 类型 | 参数名 | 说明 |
| CCObject\* | pTarget | 定时器所属对象 |

示例：

CAScheduler::getScheduler()->pauseTarget(this);

当调用此方法后会暂停当前对象的所有定时器。

**void** **resumeTarget(CCObject \*pTarget)**

返回值：void

参数：

|  |  |  |
| --- | --- | --- |
| 类型 | 参数名 | 说明 |
| CCObject\* | pTarget | 定时器所属对象 |

示例：

CAScheduler::getScheduler()->pauseTarget(this);

当调用此方法后会恢复当前对象中被暂停的所有定时器

**bool** **isTargetPaused(CCObject \*pTarget)**

返回值：bool

参数：

|  |  |  |
| --- | --- | --- |
| 类型 | 参数名 | 说明 |
| CCObject\* | pTarget | 定时器所属对象 |

示例：

bool isPaused=CAScheduler::getScheduler()->isTargetPaused(this)；

返回一个布尔值，判断当前对象的所有定时器是否被暂停。

**void** **resumeTargets(CCSet\* targetsToResume)**

返回值：void

参数：

|  |  |  |
| --- | --- | --- |
| 类型 | 参数名 | 说明 |
| CCSet\* | targetsToResume | 定时器对象集合 |