# 事件使用手册

[Ellan](http://gameframework.cn/archives/author/jiangyin" \o "由Ellan发布) [事件](http://gameframework.cn/archives/category/module/buildin/event) 围观*4326*次 [留下评论](http://gameframework.cn/archives/264" \l "respond) 编辑日期：2016-09-01 字体：[大](http://gameframework.cn/archives/javascript:;) [中](http://gameframework.cn/archives/javascript:;) [小](http://gameframework.cn/archives/javascript:;)

Game Framework 支持在游戏逻辑监听、抛出事件。

Game Framework 中的很多模块在完成操作后都会抛出内置事件，监听这些事件将大大解除游戏逻辑之间的耦合。

除了这些内置事件，用户也可以[定义自己的游戏逻辑事件](http://gameframework.cn/archives/261)。为了自定义游戏逻辑事件，需要在项目内将 EventId 枚举进行扩展，扩展的事件需要从 UnityGameFramework.Runtime.EventId.GameEventStart 之后开始。

    /// <summary>

    /// 事件类型编号。

    /// </summary>

    public enum EventId

    {

        GameEventStart = UnityGameFramework.Runtime.EventId.GameEventStart,

    }

同时，在游戏逻辑层给 EventComponent 增加一些扩展方法，使得游戏内能够方便的使用这个 EventId 进行操作。

public static class EventExtension

{

// 扩展检查订阅事件方法

  public static bool Check(this EventComponent eventComponent, EventId eventId, EventHandler<GameEventArgs> handler)  {

            return .Check((UnityGameFramework.Runtime.EventId)eventId, handler);

}

  // 扩展订阅事件方法

  public static void Subscribe(this EventComponent eventComponent, EventId eventId, EventHandler<GameEventArgs> handler){

eventComponent.Subscribe((UnityGameFramework.Runtime.EventId)eventId, handler);

  }

// 扩展取消订阅事件方法

  public static void Unsubscribe(this EventComponent eventComponent, EventId eventId, EventHandler<GameEventArgs> handler){

eventComponent.Unsubscribe((UnityGameFramework.Runtime.EventId)eventId, handler);

}

}

下面，以订阅加载数据表加载成功事件为例。

订阅事件：

  EventComponent eventComponent = GameEntry.GetComponent<EventComponent>();

  eventComponent.Subscribe(UnityGameFramework.Runtime.EventId.LoadDataTableSuccess, OnLoadDataTableSuccess);

取消订阅事件：

EventComponent eventComponent = GameEntry.GetComponent<EventComponent>();

eventComponent.Unsubscribe(UnityGameFramework.Runtime.EventId.LoadDataTableSuccess, OnLoadDataTableSuccess);

检查订阅事件：

EventComponent eventComponent = GameEntry.GetComponent<EventComponent>();

bool subscribed = eventComponent.Check(UnityGameFramework.Runtime.EventId.LoadDataTableSuccess,OnLoadDataTableSuccess);

定义事件处理函数：

private void OnLoadDataTableSuccess(object sender, GameEventArgs e)

{

UnityGameFramework.Runtime.LoadDataTableSuccessEventArgs ne = e as UnityGameFramework.Runtime.LoadDataTableSuccessEventArgs;

// Do something.

  Log.Info("Load data table '{0}' success.", ne.DataTableName);

}

抛出事件：

EventComponent eventComponent = GameEntry.GetComponent<EventComponent>();

// 抛出事件，这个操作是线程安全的，即使不在主线程中抛出，也可保证在主线程中回调事件处理函数，但事件会在抛出后的下一帧分发。

eventComponent.Fire(this, new XXXEvent());

// 抛出事件立即模式，这个操作不是线程安全的，事件会立刻分发。

eventComponent.FireNow(this, new XXXEvent());

当事件被抛出后，订阅过此事件的事件处理函数就会得到响应。

**相关 API 参考手册**

* [EventComponent（事件组件）](http://gameframework.cn/api/class_unity_game_framework_1_1_runtime_1_1_event_component.html)
* [GameEventArgs（游戏逻辑事件基类）](http://gameframework.cn/api/class_game_framework_1_1_event_1_1_game_event_args.html)
* 本文固定链接: [http://gameframework.cn/archives/264](http://gameframework.cn/archives/264" \o "事件使用手册)
* 转载请注明: [Ellan](http://gameframework.cn/archives/author/jiangyin" \o "由Ellan发布) 2016年08月31日 于 [Game Framework](http://gameframework.cn/" \o "访问Game Framework) 发表