**AD.BASE AD\_I Interface**

# 双类转换接口

IBase以及IBaseMap通过To/From函数互相转换，IBase应被用于运行时的实体或数据，IBaseMap应被用于保存数据时提供最佳的支持

两个接口都有对应的泛型版本，意味着更加明确的目标和严格的约束

**ToMap** 将IBase自身转换为IBaseMap

**FromMap** 以IBaseMap的内容初始化IBase的内容

**ToObject** 将IBaseMap自身转换为IBase

**FromObject** 以IBase的内容初始化IBaseMap的数据

**Serialize** 将IBaseMap转换为可保存的字符串，用于Deserialize解析

**Desrialize** 从源字符串解析出应用于自身的数据，其应为Serialize生成

**AD.BASE AD\_S Implementation**

# AD异常类

**ADExcption**会在生成的同时记录事件，作为AD中常用的异常类型

除此之外没有特别特殊的特点

**NullArchitecture**代表没有注册进架构或对应架构已被销毁

# AD架构接口

**IAnyArchitecture** 所有AD架构接口的最底层，并且不具备任何内容和实现

**ICanInitialize** 拥有一个**Init**函数，默认用于初始化或重置类

**ICanGetArchitecture** （旧版本）通过两个函数设置或获取**IADArchitecture**目标

（目前推荐使用）Architecture {get; set; }

**ICanGetSystem** 实现以支持获取架构中System

**ICanGetModel** 实现以支持获取架构中Model

**ICanGetController** 实现以支持获取架构中Controller

**ICanSendCommand** 实现以支持在架构中发送Command

**ICanMonitorCommand**

实现以支持在Command Diffusing过程中激活**OnCommandCall函数**

**IADModel** 实现以支持在架构中作为数据类

**IADSystem** 实现以支持在架构中作为系统类

**IADController** 实现以支持在架构中作为控制器类

**IADCommand** 实现以支持在架构中作为命令类

**IADMessage** 被用于架构内收集消息，具有一个返回字符串的**What**函数

# AD架构标准实现

**ADModel** 实现**IADModel**并提供了默认的实现

**ADSystem** 实现**IADSystem**并提供了默认的实现

**MonoSystem** 实现**IADSystem**并继承了**MonoBehaviour**

**ADController**

实现**IADController**并继承了**MonoBehaviour**,能够在OnDestroy（保护的虚函数）中将自身解除在架构中的注册关系

**ADCommand** 实现**IADCommand**并提供了默认的实现

**Vibration** 继承**ADCommand**并用于Command Diffusing

**ADMessage** 实现**IADMessage**并提供了默认的实现，记录生成的时间

**ADMessageRecord**

一个默认的消息收集器，在默认的**ADArchitecture**实现中被使用，实现了**IADModel**