# AggregateFamework Reference

**作者**

     changmingxie

介绍

什么是AggregateFramework

     Aggregate Framework是为方便开发人员运用DDD和CQRS思想来构建复杂的、可扩展的Java企业应用系统而提供的Java技术框架。该框架提供了Aggregate、Repository、Domain Event等构建块的实现；使用DomainEvent，借助于内建的Disruptor组件，AggregateFramework可使开发人员方便的实现高性能SEDA架构。此外，该框架支持与Spring集成，提供使用 annotation的方式让开发人员方便地为Domain Event定义一个或多个事件处理, 同时可指定事件处理是同步还是异步触发，是否需要重试，以及重试策略; 使用Spring事务管理器管理事务时，支持Unit Of Work数据访问模式以及内建一级缓存以提高访问性能。

Aggregate

     Aggregate(聚合)定义了一组具有内聚关系的相关对象的集合，是一致性修改数据的单元。聚合里包含有聚合根，实体、值对象。AggregateFramework为聚合定义了接口和抽象父类，开发人员实现领域模型时需继承这些抽象父类。

     在Aggregate Framework中，定义了聚合对象的接口及抽象父类,类层次结构如下。聚合最核心的接口是DomainObject，聚合对象都实现该接口。另一个接口是AggregteRoot，该接口继承自 DomainObject,  聚合根实现该接口。AbstractDomainObject抽象类实现了DomainObject接口, 并提供了对部分方法的重载。AbstractAggregateRoot继承AbstractDomainObject，在 AbstractDomainObject上新增了对Event的支持。为方便定义聚合对象，AbstractSimpleAggregateRoot和 AbstractSimpleDomainObject分别对AbstractAggregateRoot和AbstractDomainObject进 行了简单实现，提供对接口的默认实现，让开发人员不需关系内部事件注册及发布机制。

achine generated alternative text:
pkg 
o 
Domain Object 
Agg regateRoot 
Abs

Repository

     Repository用于协调领域和数据映射层，利用类似于集合的接口来访问领域对象，每个聚合根有一个对应的Repository, 用于对这个聚合对象的数据访问。Aggregate Framework定义Repository相关的接口及抽象父类，开发人员在实现聚合根的Repository时需要继承这些抽象父类。

     在Aggregate Framework中，定义了Repository对象的接口及抽象父类,类层次结构如下。在Repository构建块中，最核心的接口莫过于Repository。它是个标记接口，CrudRepository继承该接口，并为聚合对象提供 了专门的CRUD方法。AggreateRepository继承CrudRepository, 限制操作的实体是聚合根。AbstractAggregateRepository提供了对AggregateReposiotry的部分实现，将事件的 发布进行了简单处理。TraversalAggregateRepository继承了AbstractAggregateRepository，同时实 现了对聚合里对象的成员变量进行遍历并调用对应的DAO方法进行CRUD操作。DaoAwareAggregateRepository继承自 TraversalAggregateRepository，实现了基于Spring获取DAO依赖实现聚合对象的遍历CRUD操作。

achine generated alternative text:
pkg 
o 
Repository 
CrudRe ository 
Agg re

 Domain Event

     Domain Event是领域中已经发生的事件，当领域对象的状态发生了改变，便会产生Domain Event。

     在AggregateFramework中聚合根的抽象父类 AbstractAggregateRoot中定义了注册Domain Event的方法 apply(Object eventPayload), 当领域对象的改变其状态后，可使用apply方法来注册一个Domain Event，以表示产生了一个Domain Event。当调用聚合根的Repository方法save来保存聚合对象时，框架会将注册的Domain Event发布出去。

使用指南

     本指南使用一个示例来阐述如何使用AggregateFramework开发。示例的领域模型如下图，Order和Payment是聚合根，OrderLine是Order聚合里的实体。

achine generated alternative text:
pkg 
< <aggregate root>> 
Order 
- totalAm

    在示例里，将阐述两个业务流程，一个是下订单流程，另一个是第三方支付通知支付成功流程。

     下订单过程为：创建Order聚合对象，创建后将发布OrderPlaced事件，PaymentHandler接受OrderPlaced事件，并为创建的Order聚合对象产生一个Payment。下面是流程的序列图。

achine generated alternative text:
sd PlaceOrder 
1: buildOr ero : order 
1.1

     第三方支付通知支付成功过程为：根据支付订单号获取Payment, 更新Payment为Confirmed状态，更新状态后将发布PaymentConfirmed事件，OrderHandler接受PaymentConfirmed事件，并将对应的Order更新为confirm状态。下面是流程的序列图。

achine generated alternative text:
sd Notify 
1: ndByPaymentNo() : pa ment 
2

1. 定义Aggregate类

     聚合根类继承AbstractAggregateRoot抽象类，聚合实体类继承AbstractDomainObject抽象类，聚合根引用聚合实体，聚合根和聚合实体的关系可以是1:0..1或是1：0..\*。为简化实现，AggregateFramework里提供了对AbstractAggregateRoot和AbstractDomainObject的缺省实现类，分别为AbstractSimpleAggregateRoot类和AbstractSimpleDomainObject类。

     示例中，Order和Payment是聚合根，继承AbstractSimpleAggregateRoot类，Order和Payment都提供confirm方法以改变对象的状态。OrderLine是Order聚合对象里的实体, Order与OrderLine是一对多的关系，Order.orderLines实现关联多个OrderLine，同时OrderLine.order反向关联一个Order。

1. 注册Domain Event

当领域对象的状态改变后，可注册Domain Event。在AbstractAggregateRoot里提供了apply方法用于注册Domain Event。

     在示例中，在Order的confirm方法里，改变了Order的状态后，调用apply方法注册一个OrderPlacedEvent事件, 在Payment的confirm方法里，改变了Payment的状态后，调用apply方法注册一个PaymentConfirmedEvent事件。

1. 定义Repository

  每个聚合根对应一个Repository。聚合根的Repository类继承AbstractAggregateRepository类。通常应用系统使用DAO模式进行数据访问，每个实体类对应一个DAO,AggregateFramework提供了另外的抽象类TraversalAggregateRepository类和DaoAwareAggregateRepository类以支持实体类底层使用DAO模式进行数据访问。

     在示例中，OrderRepository和PaymentRepository继承DaoAwareAggregateRepository。使用OrderRepository和PaymentRepository进行数据访问操作时，AggregateFramework将CRUD操作委派给聚合里实体的DAO类进行数据访问。定义DAO将在下面介绍。

1. 定义DAO

AggregateFramework为DAO模式提供了支持, 下图为DAO类结构图。每个领域实体对应一个DAO类，聚合根对应的DAO类继承AggregateRootDao或是CollectiveAggregateRootDao，聚合里的实体类对应的DAO继承DomainObjectDao或CollectiveDomainObjectDao。 推荐使用CollectiveAggregateRootDao和CollectiveDomainObjectDao,相比另外两个DAO增加了insertAll,updateAll,deleteAll和findByIds集合操作方法以实现对实体对象的批量CRUD操作。

  示例中，OrderDao和PaymentDao接口类继承了CollectiveAggregateRootDao类，OrderLineDao接口类继承CollectiveDomainObjectDao类。示例使用mybatis的MapperFactoryBean自动生成DAO接口类的实例。

1. 定义事件处理方法

领域实体当状态发生了改变后会发出Domain Event。使用@EventHandler在Spring容器管理的类的方法上来声明领域事件处理方法，方法参数需有一个为Domain Event类型的参数以表明领域事件处理方法处理这个Domain Event。@EventHandler里有两个属性，一个是asynchronize, 默认是false, 表示事件处理方法将和对领域实体的更改在同一个线程中执行，如果设置为true, 则表示事件处理方法和对领域实体操作在不同线程执行，即异步执行。另一个属性是postAfterTransaction, 当使用TransactionManager管理事务时有效，默认是false, 表示事件处理方法和对领域实体的更改操作在同一个事务中，如果是true, 则事件处理方法将在对领域实体更改操作结束后执行。

PaymentHandler和OrderHandler类定义了事件处理方法。

1. 设置重试策略

与@EventHandler配合使用还有一个注解@Retryable, 类似于spring-retry，该注解用来设置事件处理方法的重试策略, 比如重试次数，对哪些异常进行重试，对哪些异常不重试，重试间隔。