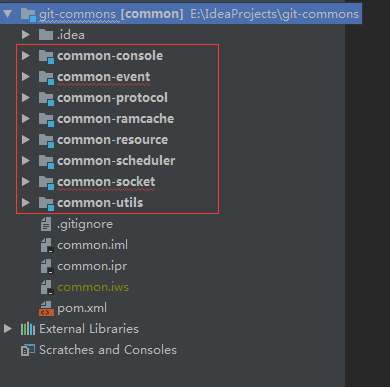
# 1 Commons

## 1.1 commons组成

Commons模块分为几个子模块：

图1



common-console：应用控制台。

common-event：简单的事件系统。

common-protocol：对象传输协议。

common-ramcache：内存缓存。

common-resource：静态资源清理。

common-scheduler：定时任务支持。

common-socket：通信支持模块。

common-utils：基础模块类。

## 1.2 common-console

common-console包括命令，控制台等信息。

### 1.2.1 Command

Command是一个接口，表示抽象命令。一条抽象命令有自己的命令名称，命令描述，命令的执行过程。

图1

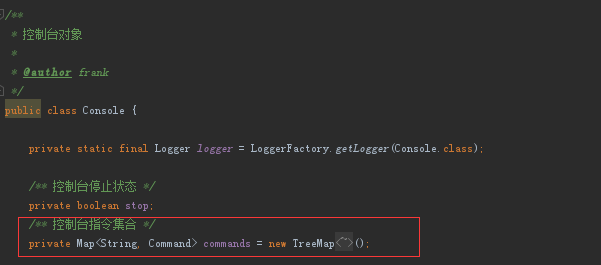


### 1.2.2 Console

Console是控制台对象，是一个类。

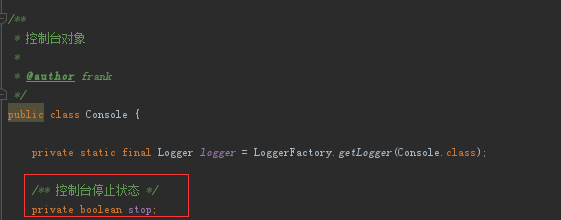
1.一个控制台（Console）由很多条命令（Command）组成，且这些命令之间都有顺序。控制台用一个TreeMap来管理这些命令，key是命令名称，value是命令对象（Command类型）。

图1



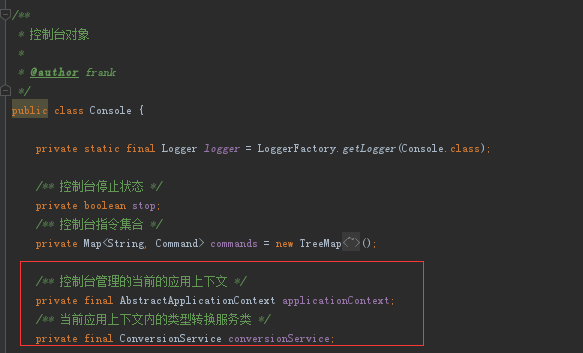
1. 一个控制台有自己的停止状态（stop），表示这个控制台是否已经关闭。

图2



1. 一个控制台管理一个应用上下文（applicationContext），以及管理这个应用上下文的类型转换服务类（conversionService）。

图3



1. 停止控制台的命令。

图1

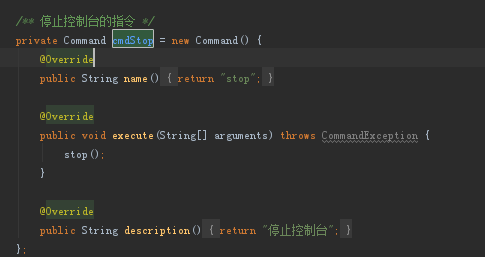
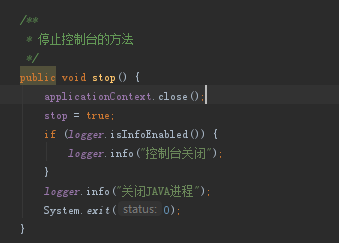


图2（stop()实际上关闭它管理的上下文，并且将关闭状态设置成true）



1. 启动控制台。实际上是创建了一个线程去执行控制台中的命令集合。

图1



1. 列出控制台所有命令的命令。有一条cmdList命令，实际上遍历控制台中管理的命令集合commands，列出所有的命令。

图1



### 1.2.3 @ConsoleBean

ConsoleBean是一个自定义注解，表示这个类是一个控制台。

这个注解加在类或接口上（ElementType.Type）

图1

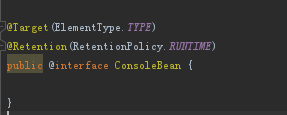
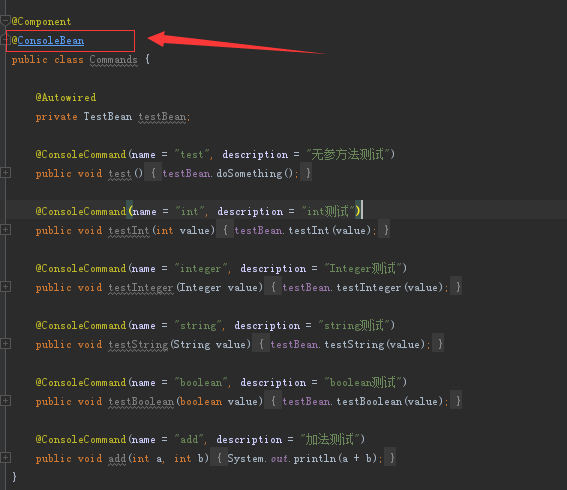


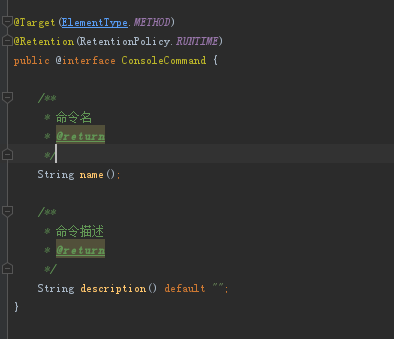
图2



### 1.2.4 @ConsoleCommand

ConsoleCommand是一个注解，表示这个方法是一条命令。

图1



### 1.2.5 ConsoleRunner

ConsoleRunner是一个线程，构造时传入一个控制台对象（Console类型），这个线程会读取用户的输入，然后从控制台中获取这条命令的具体内容，将用户的输入赋值到命令后执行。

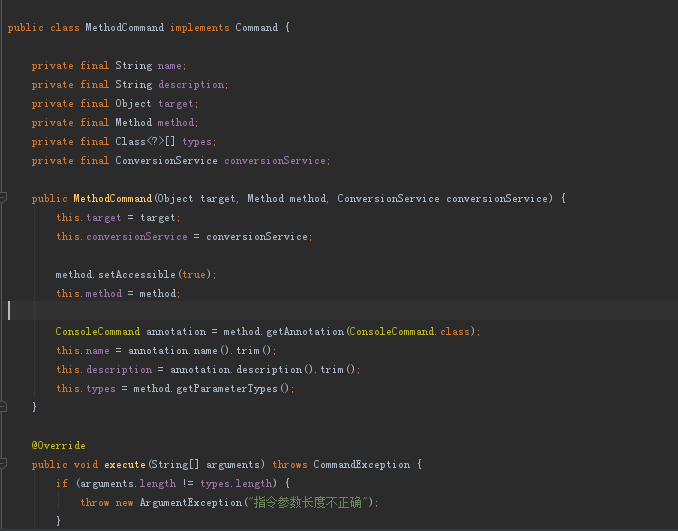
图1



### 1.2.6 MethodCommand

MethodCommand是Command抽象命令的一个实现子类。其中的ConversionService是用来进行参数类型转换的。

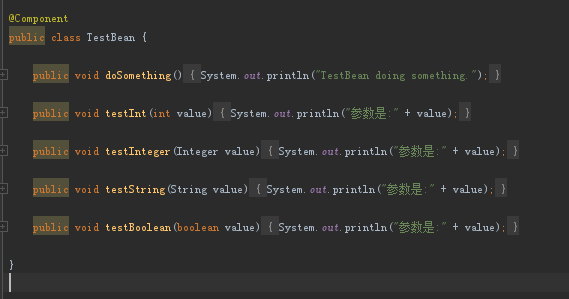
图1



### 1.2.7 用法

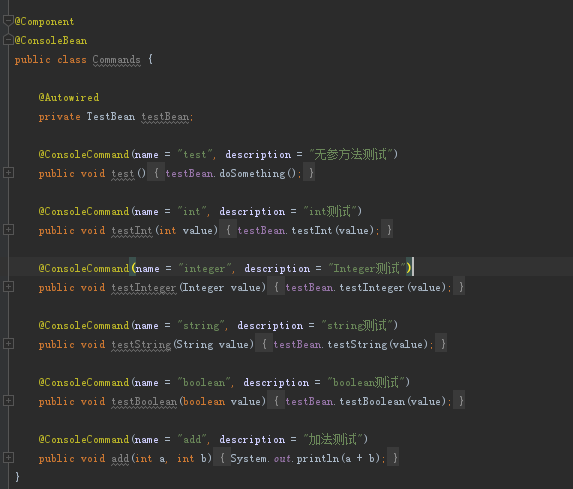
1.首先有一个待测试的bean。

图1



1. 在待测试的bean外面定义一个普通的外层bean（使用@Component），然后将这个外层bean加入@ConsoleBean注解，外层bean的方法加入@ConsoleCammand注解，外层bean的方法底层调用待测试bean的方法。

图1



1. 创建一个控制台（Console类），传入上下文环境，然后启动控制台。控制台创建时会扫描所有带有ConsoleBean注解的类。

图1



图2

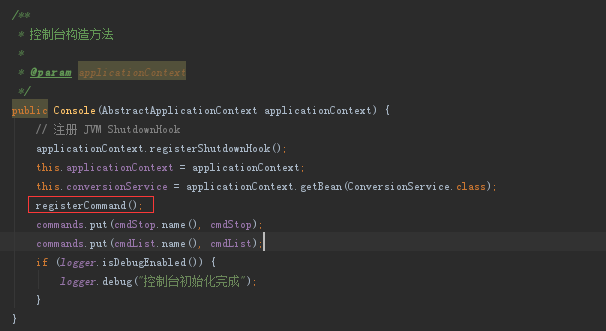
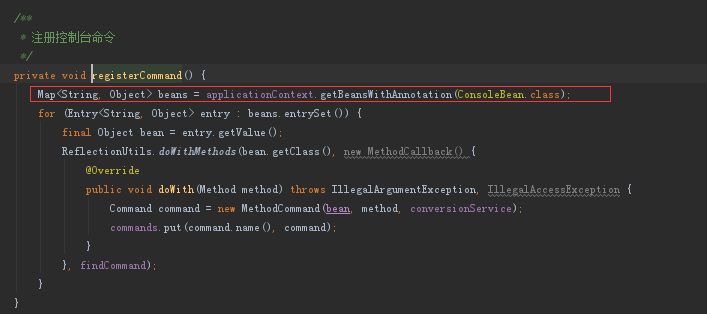


图3



## 1.3 common-event

common-event其实就是生产者-消费者的一个应用，生产者将事件放到事件总线上，然后事件接收者从事件总线上取事件处理。当订阅了该事件的事件接收者全都处理完该事件后，该事件从事件总线上消失。

### 1.3.1 Receiver

Receiver是事件接收者，有一个onEvent方法来处理事件。

图1



### 1.3.2 Event

Event是事件，有事件头和事件体两部分。

图1



### 1.3.3 EventBus

EventBus是一个接口，表示事件总线，包括同步发送事件syncPost，异步发送事件post，注册某个事件的事件接收者，撤销某个事件的事件接收者等方法。

图1



### 1.3.4 AbstractReceiver

AbstractReceiver是事件的接收者，实现了Receiver接口。

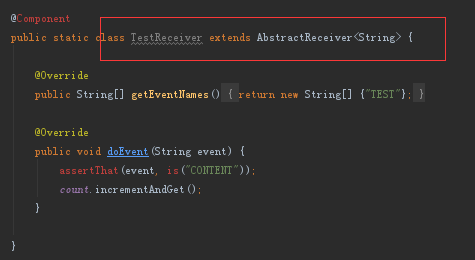
图1



### 1.3.5 用法

1.定义消息接收者，继承抽象消息接收者（AbstractReceiver）。

图1

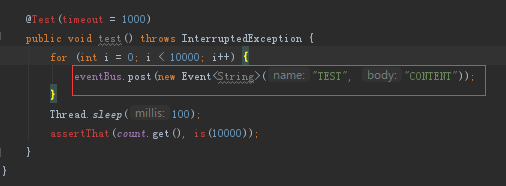


1. 定义一个消息总线的域，生产者将事件发送到消息总线上。

图2



图3

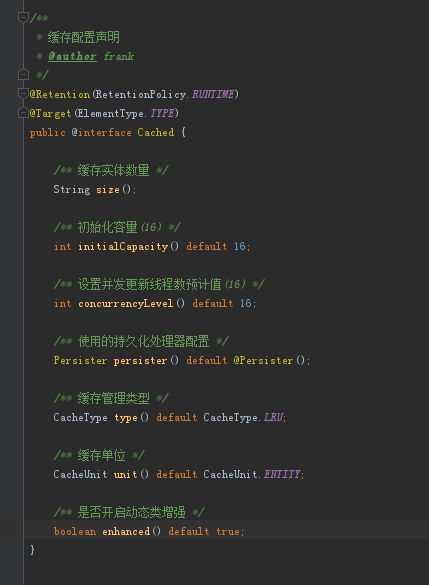


## 1.4 common-ramcache

### 1.4.1 @Cached

Cached是一个注解，表示缓存的容量，缓存的类型等。

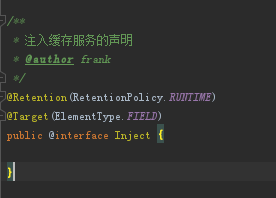
图1



### 1.4.2 @Inject

Inject是注入缓存服务的声明。

图1



### 1.4.3 IndexValue

IndexValue是索引值对象，是一个类，包括索引名和索引值。

图1



图2（使用valueOf创建了IndexValue对象，valueOf方法进行了安全检测，保证name不能为空，当然也可以写在构造函数中，只是习惯不同）

（注意：valueOf一般是给定参数，然后返回一个原始对象）



图3（构造方法是私有的）

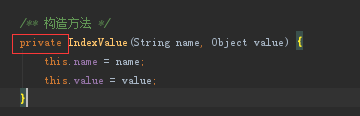
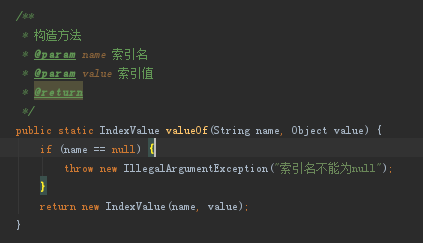


图4



### 1.4.4 IEntity

IEntity是实体标志接口。

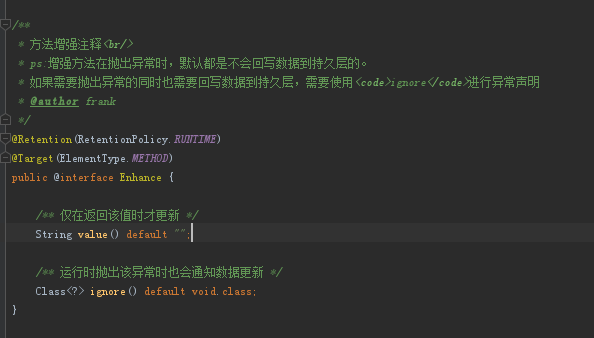
图1



### 1.4.5 @Enhance

Enhance是方法增强注解。

图1



### 1.4.6 Enhancer

Enhancer是实体类增强器接口，用来将实体类增强。

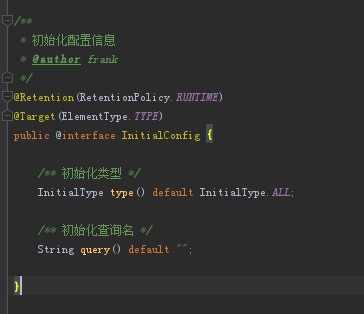
图1



### 1.4.7 @InitConfig

InitConfig是初始化的配置信息。

图1



### 1.4.8 BasicItem

BasicItem是测试道具实体，一个道具有id，拥有者，数量。

图1



# 2 xxarpg

## 2.1 ConsoleCommand

ConsoleCommand是运维管理指令，是一个接口，里面包含了很多个指令（Command）。

图1



## 2.2 ConsoleModule

ConsoleModule封装了指令（Command）的类型。

图1

