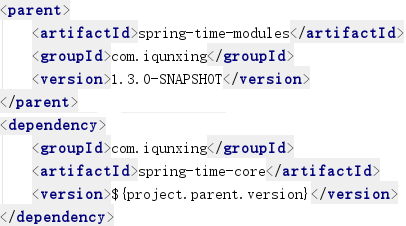


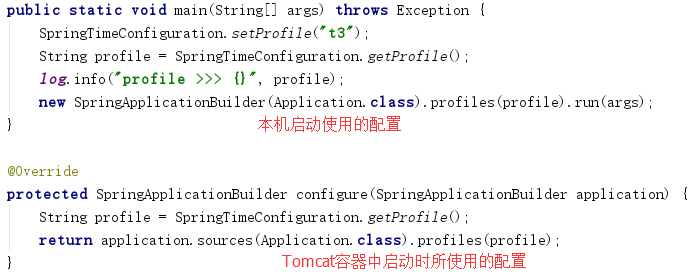
SpringTime是群星金融微服务底层框架，是以Spring Framework和Spring Cloud为核心的，是JavaEE世界中的主流技术选型，最佳实践的总结。群星系统中，几乎所有基于Spring Cloud框架的服务都会引用SpringTime模块。如图所示，SpringTime可以划分为一下几个模块

1. api - 后台微服务之间互相调用的api
2. base - 通用的基础对象，常量，以及基础异常类
3. utils - 常用的工具类
4. rabbit - 定义rabbitmq的事件和传播对象
5. core - 结合Spring Framework和Spring Cloud，提供微服务底层统一框架

### SpringTime使用

要使用SpringTime，首先需要在pom文件中添加包的引用。



然后在启动类上加上启动注解标签（@EnableSpringTime）。除此之外，还需要在微服务启动类中加入环境配置的代码。

## api

这个模块主要包括后台微服务之间互相调用，每个微服务供内部调用的接口都在这里。采用了Spring Framework中的Feign的方式。每个类上方都会有注解。

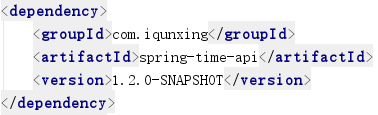
@FeignClient(value = **"loan"**, url = **"${service.uri}"**, configuration = SpringTimeFeignConfiguration.**class**)

这里的Value表示暴露出接口的微服务的服务名，url表示微服务的访问地址，configuration中封装了feign拦截器（包括调用过程中的错误的处理等）。

假如微服务已经注册到注册中心，则可以不需要url，直接根据value中的服务名找到微服务，如果微服务没有注册到注册中心上，可以通过url来调用接口。

### api使用

如果要调用SpringTime模块中定义的接口，首先，要在该项目的pom文件中添加SpringTime及api模块。

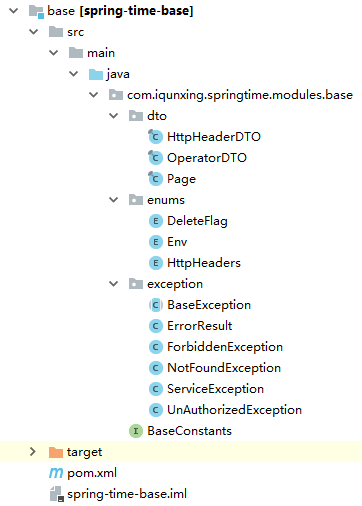


值得一提的是，api模块的版本号，不一定跟SpringTime的版本号相同。

之后在class中注入这个interface，就可以调用了。

## base

定义了一些基础类，常量，以及基础异常类，具体使用方法与api的使用方法类似。



## utils

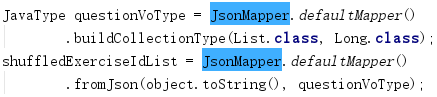
这个模块下，是一些封装的常用的工具类，目前有两个，一个是JsonMapper，一个是BeanMapper。

### JsonMapper

JsonMapper封装了com.fasterxml.jackson下的方法，便于开发人员日常操作Json格式的数据。

其中的toJson和fromJson这两个方法，可以轻松的将Object和String互相转换。

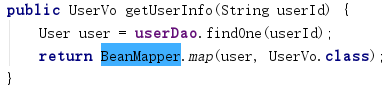
使用fromJson的时候，如果想把Json字符串转换成复杂的Object（类似HashMap<String，String>）的时候，可以使用buildCollectionType和buildMapType方法，获得JavaType，并将String转换成该JavaType。示例代码如下：



### BeanMapper

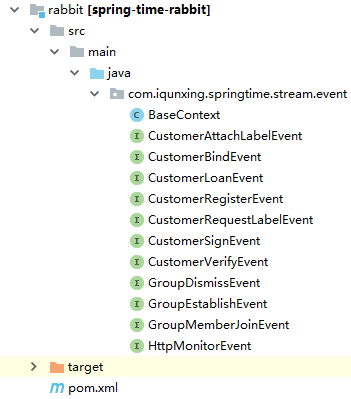
BeanMapper封装了ma.glasnost.orika下的方法，便于开发人员日常操作Bean的赋值。

其中的map，mapList这些方法，可以轻松的将object的各项参数赋给另一个有相同属性的object，或者批量操作。示例代码如下：



## rabbit

定义消息事件场景和传播上下文



## core

core模块是SpringTime中最核心的模块，主要结合Spring Framework和Spring Cloud，提供微服务底层统一框架。

### @EnableSpringTime

用于在其他微服务中引入SpringTime的注解。

### SpringTimeConfiguration

### 

@ComponentScan是扫描加载bean的注解

@Configuration是扫描加载本class中的bean的注解

@EnableFeignClients是用于开启feign调用功能的注解

@EnableRetry开启重试机制的注解

@EnableDiscoveryClient是服务发现注册的注解

@EnableSwagger2开启Swagger2的注解

除了这些注解，该类中还定义了restTemplate，ErrorPageFilter和环境配置设置的方法。

SpringTime模块中的环境设置，在每个接入的微服务中都会用到。它的作用是让每个微服务都接入到对应的环境。

### exception

定义统一异常处理类，集中处理异常，并打印日志。

### feign

feign拦截器，统一管理userId和tk等数据，以及对feign调用时产生的异常做处理。

### health

自定义的健康检查接口。

### interceptor

统一管理当前用户tk，当前userId，当前customerId等信息的方法（可以通过这个方法在各个非public的微服务中获取当前用户信息），与SpringTimeFeignConfiguration联合使用。获取当前userId的示例代码入下所示：

HttpHeaderAttributes.*get*(HttpHeaderDTO::getUserId)

### logback

日志配置。

### operator

Redis中操作人员的库，用于单点/多点登录，和操作人员权限变更时自动登出。

### redis

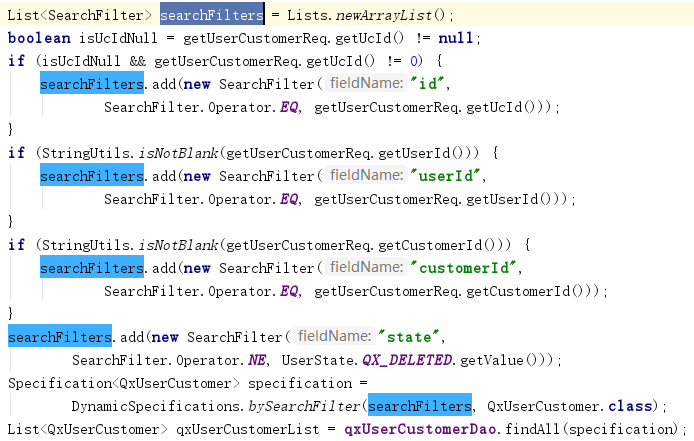
Redis的基础配置，RedisTemplate和RedisCacheManager等Bean的初始化。

### swagger

Swagger的基础配置。

### jpa

群星自己封装的JPA查询工具类，为数据库的查询提供了部分便利。具体使用方式如下图所示：



### ribbon

ribbon是Spring Cloud自带的负载均衡。具体参考Spring Cloud文档。

### validate

endpoint传入参数统一校验。