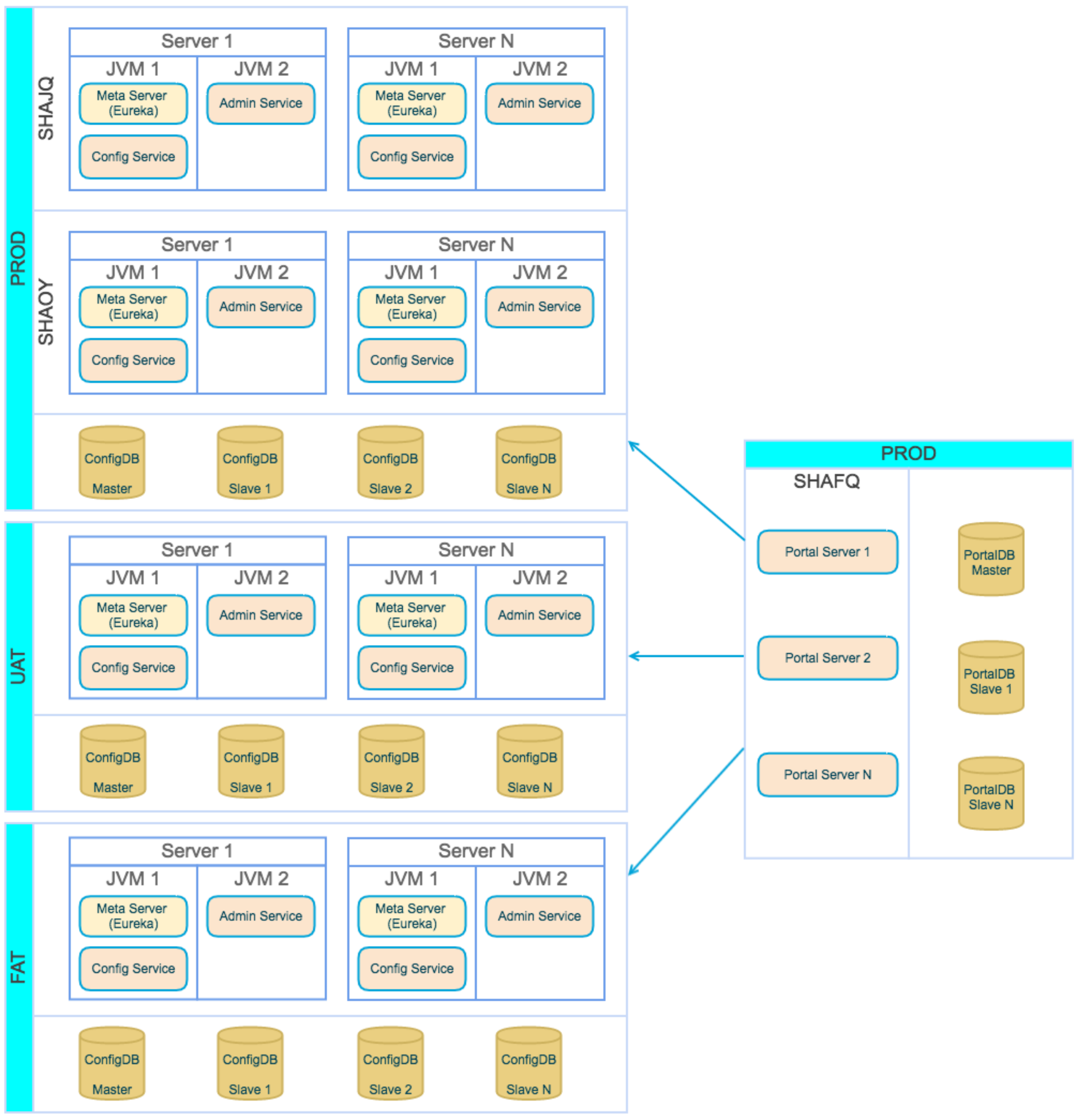
Apollo Configuration Guide

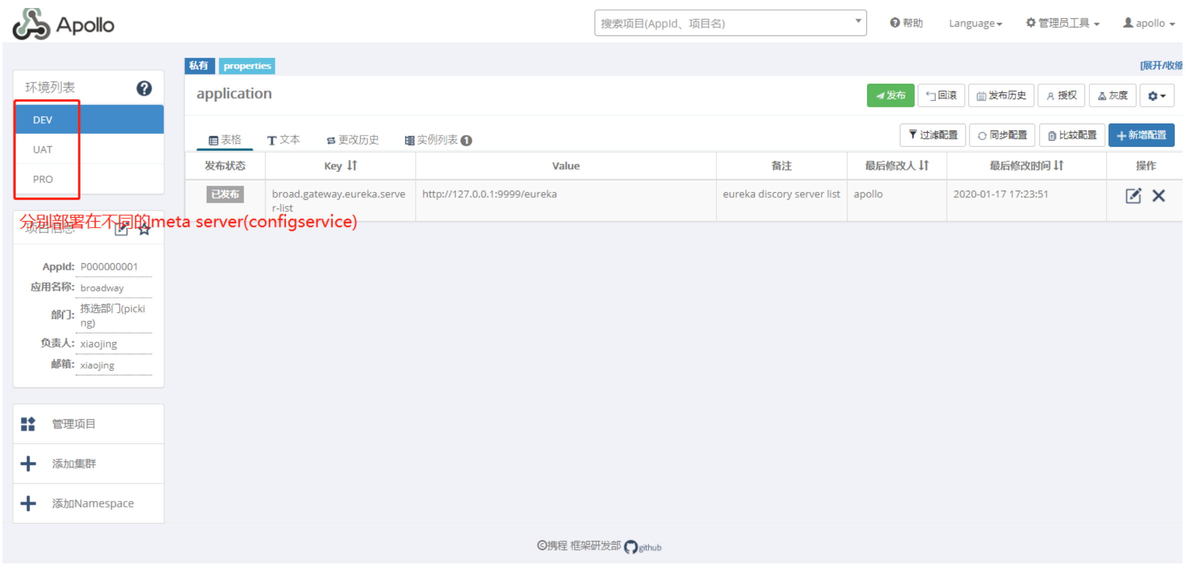
> 目前市面上流行的三大配置中心框架：[Spring CLoud Config](#)、[Alibaba Nacos](#) 以及[携程apollo](#)，我们相应架构组号召，就使用Apollo吧。

Work Flow



简单解释：

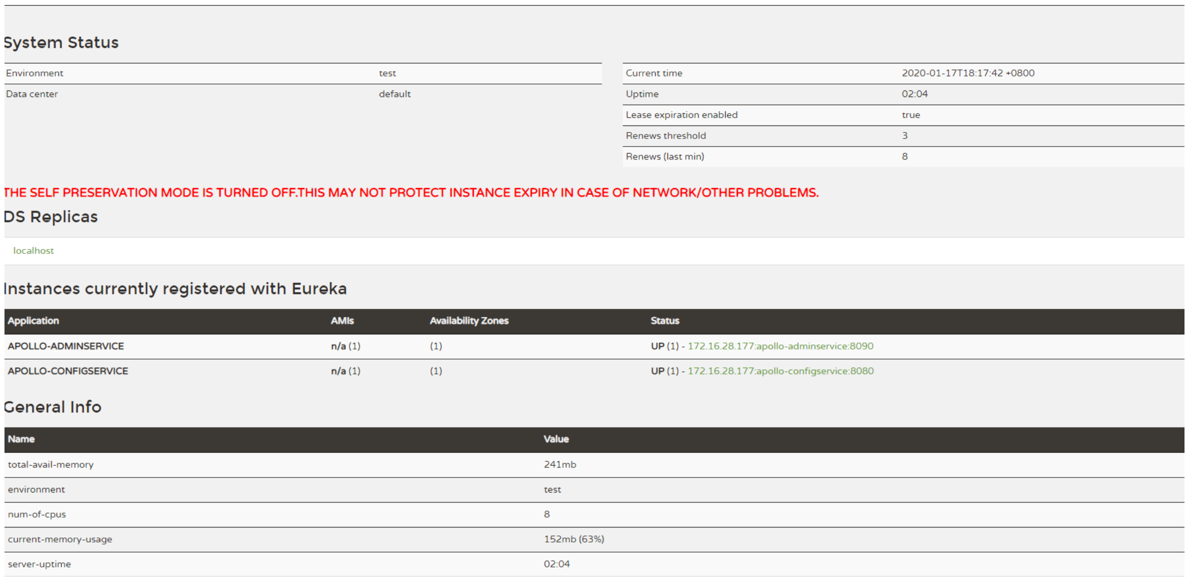
上图中有三套环境`FAT`、`UAT`和`PROD`，每一套环境都部署了2套`Configservice` 和 `Adminservice`，使用统一的`Portal Server Cluster`对所有环境进行配置管理。如我们自己的配置环境：



AppId:	P000000001
应用名称:	broadway
部门:	拣选部门(picking)
负责人:	xiaojing
邮箱:	xiaojing

发布状态	Key	Value	备注	最后修改人	最后修改时间	操作
已发布	broad.gateway.eureka.server-list	http://127.0.0.1:9999/eureka	eureka discory server list	apollo	2020-01-17 17:23:51	

Meta-Server（Configservice）说白一点，就是`Eureka discovery`,每一套环境对应不同的一套meta server，以此来实现环境隔离，如下图：



System Status

Environment	test
Data center	default

Current time	2020-01-17T18:17:42 +0800
Uptime	02:04
Lease expiration enabled	true
Renews threshold	3
Renews (last min)	8

THE SELF PRESERVATION MODE IS TURNED OFF.THIS MAY NOT PROTECT INSTANCE EXPIRY IN CASE OF NETWORK/OTHER PROBLEMS.

DS Replicas

localhost

Instances currently registered with Eureka

Application	AMIs	Availability Zones	Status
APOLLO-ADMINSERVICE	n/a (1)	(1)	UP (1) - 172.16.28.177:apollo-adminservice:8090
APOLLO-CONFIGSERVICE	n/a (1)	(1)	UP (1) - 172.16.28.177:apollo-configservice:8080

General Info

Name	Value
total-avail-memory	241mb
environment	test
num-of-cpus	8
current-memory-usage	152mb (63%)
server-uptime	02:04

```
 #因为资源有限，因此我配置的三套环境使用同一台数据库，无法实现环境隔离，但是不影响展示效果。
 local.meta=http://localhost:8080
 2 dev.meta=http://localhost:8080
 4 uat.meta=http://localhost:8080
 6 pro.meta=http://localhost:8080
```

如图2所示。

Configure Server

apollo 服务端主要有三个Spring Boot项目 和两个数据库组成：

- **apollo-configservice（默认端口：8080）**

  > 提供配置的读取、推送等功能，服务对象是Apollo客户端

- **apollo-adminservice（默认端口：8090）**

  > 提供配置的修改、发布等功能，服务对象是Apollo Portal（管理界面）

- **apollo-portal（默认端口：8070）**

> 通过域名访问Meta Server获取Admin Service服务列表（IP+Port），而后直接通过IP+Port访问服务，同时在Portal侧会做load balance、错误重试

- apolloportaldb
- apolloconfigdb

因为Apollo官方文档足够详细，想要了解的同学直接Apollo官网传送门

但是又因为官方文档太过详细，以至于如果只想部署的同学可能觉得稍显繁琐，因此，我这里直接开始部署服务端，就废话少说了。

1. 第一步，下载https://github.com/ctripcorp/apollo/releases
2. 第二步，下载`adminservice`、`configservice`和`portal`三个zip包之后，上传到服务器。
3. 第三部，在服务器中安装mysql,并创建数据库，脚本传送门
4. 第四步，分别配置三个Springboot服务并启动，主要配置点有3处：
    1. 数据库配置

       ```
         1 # DataSource
         2 spring.datasource.url = jdbc:mysql://localhost:3306/ApolloPortalDB?characterEncoding=utf8
         3 spring.datasource.username = wr
         4 spring.datasource.password = wr
       ```

    2. 日志路径
        - (`/根路径/apollo-xxxx.conf`)

          ```
            1 MODE=service
            2 PID_FOLDER=.
            # 这里
            3 LOG_FOLDER=/home/jing/software/apollo-portal-1.5.1/logs/100003173/
          ```

        - /script/startup.sh

          ```
            1 #!/bin/bash
            2 SERVICE_NAME=apollo-portal
          3 ## Adjust log dir if necessary
            #这里
            4 LOG_DIR=/home/jing/software/apollo-portal-1.5.1/logs/100003173
            5 ## Adjust server port if necessary
            6 SERVER_PORT=${SERVER_PORT:=8070}
          ```

    3. 配置apollo-portal的meta service信息

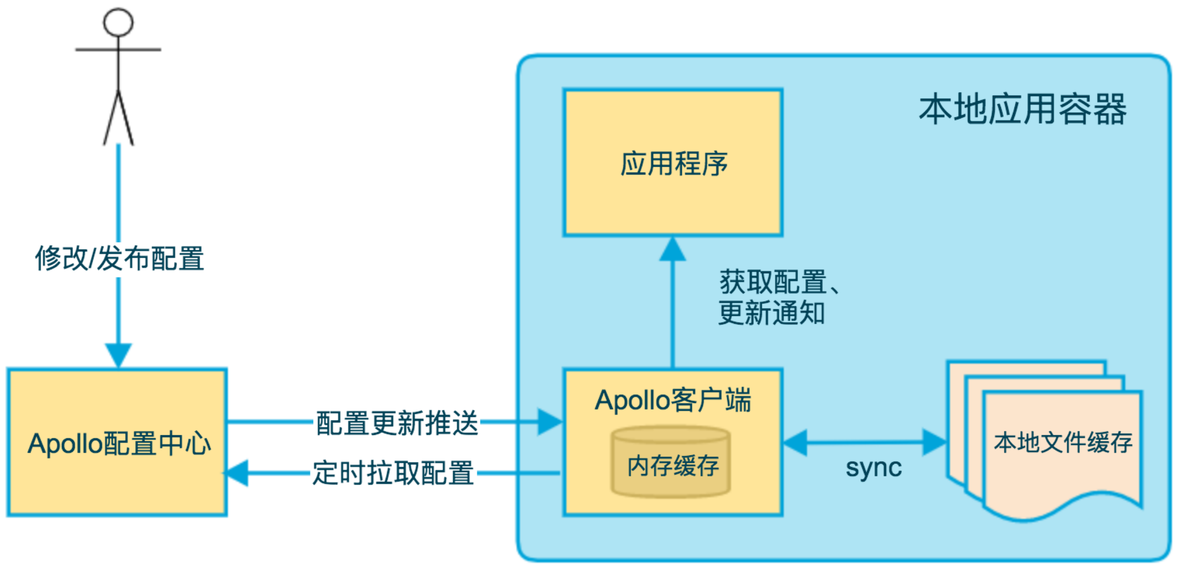
       ```
         local.meta=http://localhost:8080
         2 dev.meta=http://localhost:8080
         4 uat.meta=http://localhost:8080
         6 pro.meta=http://localhost:8080
       ```

       这里也得修改`apolloconfigdb数据库中的表serverconfig`中`apollo.portal.envs`的配置为：`dev,uat,prod`

然后apollo server就配置好了，分别启动三个服务即可！访问http://172.16.28.177:8070/

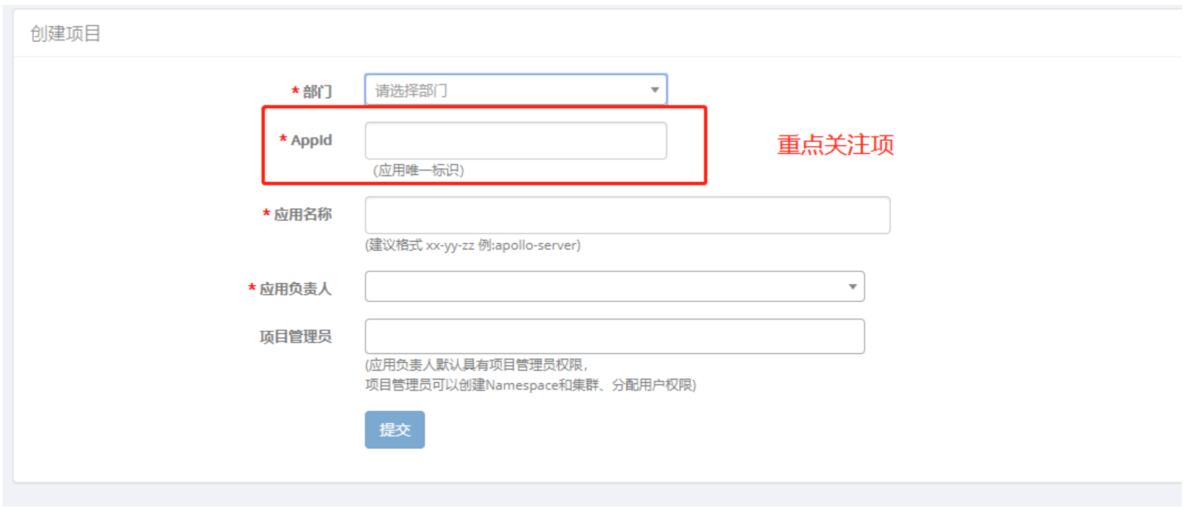
```
jing@sysdep:~/software/apollo-adminservice-1.5.1$ ps -ef|grep apollo
jing     25348     1 19 08:12 pts/2    00:01:31 /home/jing/.jenv/versions/1.8/bin/java -Dsun.misc.URLClassPath.disableJarChecking=true -XX:ParallelGCThreads=4 -XX:MaxTenuringThreshold=9 -XX:+DisableExplicitGC -XX:+ScavengeBeforeFullGC -XX:SoftRefLRUPolicyMSPerMB=0 -XX:+ExplicitGCInvokesConcurrent -XX:+HeapDumpOnOutOfMemoryError -XX:-OmitStackTraceInFastThrow -Duser.timezone=Asia/Shanghai -Dclient.encoding.override=UTF-8 -Dfile.encoding=UTF-8 -Djava.security.egd=file:/dev/./urandom -Dserver.port=8080 -Dlogging.file=/home/jing/software/apollo-configservice-1.5.1/logs/100003171/apollo-configservice.log -XX:HeapDumpPath=/home/jing/software/apollo-configservice-1.5.1/logs/100003171/HeapDumpOnOutOfMemoryError/ -XX:+UseParNewGC -Xloggc:/home/jing/software/apollo-configservice-1.5.1/logs/100003171/gc.log -XX:+PrintGCDetails -XX:+UseConcMarkSweepGC -XX:+UseCMSCompactAtFullCollection -XX:+UseCMSInitiatingOccupancyOnly -XX:CMSInitiatingOccupancyFraction=60 -XX:+CMSClassUnloadingEnabled -XX:+CMSParallelRemarkEnabled -XX:CMSFullGCsBeforeCompaction=9 -XX:+CMSClassUnloadingEnabled -XX:+PrintGCDateStamps -XX:+PrintGCApplicationConcurrentTime -XX:+PrintHeapAtGC -XX:+UseGCLogFileRotation -XX:NumberOfGCLogFiles=5 -XX:GCLogFileSize=5M -jar /home/jing/software/apollo-configservice-1.5.1/apollo-configservice.jar
jing     27274     1 37 08:17 pts/2    00:01:23 /home/jing/.jenv/versions/1.8/bin/java -Dsun.misc.URLClassPath.disableJarChecking=true -XX:ParallelGCThreads=4 -XX:MaxTenuringThreshold=9 -XX:+DisableExplicitGC -XX:+ScavengeBeforeFullGC -XX:SoftRefLRUPolicyMSPerMB=0 -XX:+ExplicitGCInvokesConcurrent -XX:+HeapDumpOnOutOfMemoryError -XX:-OmitStackTraceInFastThrow -Duser.timezone=Asia/Shanghai -Dclient.encoding.override=UTF-8 -Dfile.encoding=UTF-8 -Djava.security.egd=file:/dev/./urandom -Dserver.port=8090 -Dlogging.file=/home/jing/software/apollo-adminservice-1.5.1/logs/100003172/apollo-adminservice.log -XX:HeapDumpPath=/home/jing/software/apollo-adminservice-1.5.1/logs/100003172/HeapDumpOnOutOfMemoryError/ -XX:+UseParNewGC -Xloggc:/home/jing/software/apollo-adminservice-1.5.1/logs/100003172/gc.log -XX:+PrintGCDetails -XX:+UseConcMarkSweepGC -XX:+UseCMSCompactAtFullCollection -XX:+UseCMSInitiatingOccupancyOnly -XX:CMSInitiatingOccupancyFraction=60 -XX:+CMSClassUnloadingEnabled -XX:+CMSParallelRemarkEnabled -XX:CMSFullGCsBeforeCompaction=9 -XX:+CMSClassUnloadingEnabled -XX:+PrintGCDateStamps -XX:+PrintGCApplicationConcurrentTime -XX:+PrintHeapAtGC -XX:+UseGCLogFileRotation -XX:NumberOfGCLogFiles=5 -XX:GCLogFileSize=5M -jar /home/jing/software/apollo-adminservice-1.5.1/apollo-adminservice.jar
jing     28964     1 99 08:19 pts/3    00:01:00
/home/jing/.jenv/versions/1.8/bin/java -
Dsun.misc.URLClassPath.disableJarChecking=true -XX:ParallelGCThreads=4 -
XX:MaxTenuringThreshold=9 -XX:+DisableExplicitGC -XX:+ScavengeBeforeFullGC -
XX:SoftRefLRUPolicyMSPerMB=0 -XX:+ExplicitGCInvokesConcurrent -
XX:+HeapDumpOnOutOfMemoryError -XX:-OmitStackTraceInFastThrow -
Duser.timezone=Asia/Shanghai -Dclient.encoding.override=UTF-8 -
Dfile.encoding=UTF-8 -Djava.security.egd=file:/dev/./urandom -
Dserver.port=8070 -Dlogging.file=/home/jing/software/apollo-portal-
1.5.1/logs/100003173/apollo-portal.log -
XX:HeapDumpPath=/home/jing/software/apollo-portal-
1.5.1/logs/100003173/HeapDumpOnOutOfMemoryError/ -XX:+UseParNewGC -
Xloggc:/home/jing/software/apollo-portal-1.5.1/logs/100003173/gc.log -
XX:+PrintGCDetails -XX:+UseConcMarkSweepGC -XX:+UseCMSCompactAtFullCollection
-XX:+UseCMSInitiatingOccupancyOnly -XX:CMSInitiatingOccupancyFraction=60 -
XX:+CMSClassUnloadingEnabled -XX:+CMSParallelRemarkEnabled -
XX:CMSFullGCsBeforeCompaction=9 -XX:+CMSClassUnloadingEnabled -
XX:+PrintGCDateStamps -XX:+PrintGCApplicationConcurrentTime -
XX:+PrintHeapAtGC -XX:+UseGCLogFileRotation -XX:NumberOfGCLogFiles=5 -
XX:GCLogFileSize=5M -jar /home/jing/software/apollo-portal-1.5.1/apollo-
portal.jar
```

Configure Client



客户端使用分步骤：访问（http://172.16.28.177:8070/）

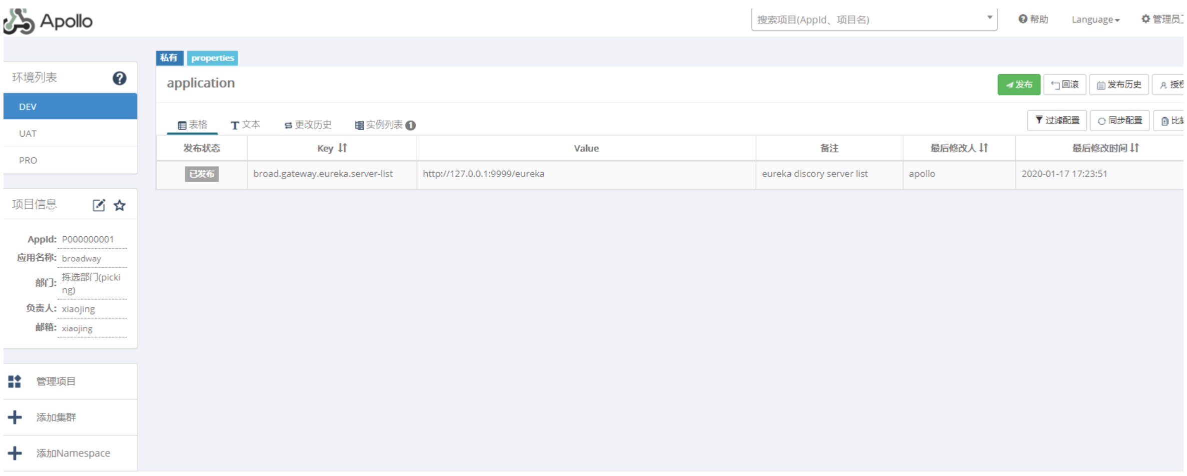
1. 第一步：创建项目



创建项目

* 部门	请选择部门
* AppId	(应用唯一标识)
* 应用名称	(建议格式 xx-yy-zz 例:apollo-server)
* 应用负责人	
项目管理员	(应用负责人默认具有项目管理员权限, 项目管理员可以创建Namespace和集群、分配用户权限)

创建之后，如下图



application

发布状态	Key	Value	备注	最后修改人	最后修改时间
已发布	broad.gateway.eureka.server-list	http://127.0.0.1:9999/eureka	eureka discory server list	apollo	2020-01-17 17:23:51

项目信息

AppId:	P000000001
应用名称:	broadway
部门:	拣选部门(picking)
负责人:	xiaojing
邮箱:	xiaojing

具体查看使用指南传送门

上面不是我要说的重点，重点是如何在我们项目中使用，以broadway-gateway为例。

- 第一步：创建bootstrap.yml，在application.yml加载之前要加载配置属性

```yaml
app:
#重点关联，id必须是唯一的，每个service各自不同
  id: P000000001
apollo:
  meta: http://172.16.28.177:8080/ # apollo-configservice
  bootstrap:
    enabled: true
    namespaces: application
```

- 第二步：使用配置的变量

```yaml
eureka:
  instance:
    prefer-ip-address: true
  client:
    service-url:
    # 冒号后面的baidu.com是默认值，如果无法连接apollo,可以使用默认值防止出错
      defaultZone: ${broad.gateway.eureka.server-list:http://baidu.com}
```

运行gateway,可以看到如下结果：

```
2020-01-17 17:06:56.623  INFO 12444 --- [           main]
c.c.f.f.i.p.DefaultApplicationProvider   : App ID is set to P000000001 by
app.id property from System Property
2020-01-17 17:06:56.634  INFO 12444 --- [           main]
c.c.f.f.i.p.DefaultServerProvider        : Environment is set to null.
Because it is not available in either (1) JVM system property 'env', (2) OS
env variable 'ENV' nor (3) property 'env' from the properties InputStream.
2020-01-17 17:06:56.711  INFO 12444 --- [           main]
c.c.f.a.i.DefaultMetaServerProvider      : Located meta services from
apollo.meta configuration: http://172.16.28.177:8080/!
2020-01-17 17:06:56.719  INFO 12444 --- [           main]
c.c.f.apollo.core.MetaDomainConsts       : Located meta server address
http://172.16.28.177:8080/ for env UNKNOWN from
com.ctrip.framework.apollo.internals.DefaultMetaServerProvider
2020-01-17 17:06:57.761  INFO 12444 --- [           main]
trationDelegate$BeanPostProcessorChecker : Bean
'org.springframework.cloud.autoconfigure.ConfigurationPropertiesRebinderAut
oConfiguration' of type
[org.springframework.cloud.autoconfigure.ConfigurationPropertiesRebinderAut
oConfiguration$$EnhancerBySpringCGLIB$$bc952272] is not eligible for
getting processed by all BeanPostProcessors (for example: not eligible for
auto-proxying)

  .   ____          _            __ _ _
 /\\ / ___'_ __ _ _(_)_ __  __ _ \ \ \ \
( ( )\___ | '_ | '_| | '_ \/ _` | \ \ \ \
 \\/  ___)| |_)| | | | | || (_| |  ) ) ) )
  '  |____| .__|_| |_|_| |_\__, | / / / /
 =========|_|==============|___/=/_/_/_/
 :: Spring Boot ::       (v2.1.10.RELEASE)
...
```

可以看到，在springboot启动之前，apollo首先加载配置信息了～

相关profile信息，后续再表，先改bug了～～