Apollo

这是一篇配置中心工具`Apollo`的一篇知识点记录文章，参考github中的指导文档；此文章不再赘述太多，只记录难点、易遗忘点；

1. AppId配置

1. `classpath:/META-INF/app.properties`配置文件中配置
2. 系统属性（`System Property`）中配置
   - `-Dapp.id=YOUR-APP-ID`

2. Environment

2.1 配置方式

1. **Java System Property**

- 可以通过Java的System Property `env`来指定环境
- 在Java程序启动脚本中，可以指定`-Denv=YOUR-ENVIRONMENT`
  - 如果是运行jar文件，需要注意格式是`java -Denv=YOUR-ENVIRONMENT -jar xxx.jar`
- 注意key为全小写

2. **System Environment**

- 还可以通过操作系统的System Environment `ENV`来指定
- 注意key为全大写

3. **config file**

- 最后一个推荐的方式是通过配置文件来指定`env=YOUR-ENVIRONMENT`
  - 对于Mac/Linux，文件位置为`/opt/settings/server.properties`
  - 对于Windows，文件位置为`C:\opt\settings\server.properties`

2.2 目前Environment支持的值（大小写不敏感）

- **DEV** (Development environment)、**FAT** (Feature Acceptance Test environment)、**UAT** (User Acceptance Test environment)、**PRO** (Production environment)

3. 集群

3.1 配置方式

1. `-Dapollo.cluster=SomeCluster`
2. 配置文件
   - 对于Mac/Linux，文件位置为 `/opt/settings/server.properties`
   - 对于Windows，文件位置为 `C:\opt\settings\server.properties`

3.2 集群配置优先级

- `apollo.cluster` > `IDC` > `default`

4. 本地缓存

4.1 缓存路径

- **Mac/Linux**: `/opt/data/{*appId*}/config-cache`
- **Windows**: `C:\opt\data\{*appId*}\config-cache`

4.2 文件命名格式

- `{appId}+{cluster}+{namespace}.properties`
  - `namespace` 就是应用使用的配置 `namespace`，一般是 `application`

5. 配置项

- `apollo.refreshInterval` - 客户端定时拉取配置的频率，单位为分钟，默认5分钟；
- `env=Local` - 设置本地开发模式
  - 本地开发模式 - Apollo只会从本地文件（缓存配置文件）读取配置信息，不会从Apollo服务器读取配置
  - 需要事先在对应目录下准备配置文件，并遵循 `{appId}+{cluster}+{namespace}.properties` 的命名格式
    - **Mac/Linux**: /opt/data/{*appId*}/config-cache
    - **Windows**: C:\opt\data{*appId*}\config-cache

6. 分布式部署指南

6.1 ApolloPortalDB.ServerConfig

- `apollo.portal.envs` - **可支持的环境列表**

- `organizations` - **部门列表**

  格式：

  ```
  [{"orgId":"TEST1","orgName":"样例部门1"},
  {"orgId":"TEST2","orgName":"样例部门2"}]
  ```

- `superAdmin` - **Portal超级管理员**

  多个账号以英文逗号分隔(,)

- `consumer.token.salt` - consumer token salt

- `wiki.address`

  portal上"帮助"链接的地址，默认是Apollo github的wiki首页

- `admin.createPrivateNamespace.switch` - **是否允许项目管理员创建private namespace**

-

6.2 ApolloConfigDB.ServerConfig

- `eureka.service.url` - Eureka服务Url

  > 不管是apollo-configservice还是apollo-adminservice都需要向eureka服务注册，所以需要配置eureka服务地址。 按照目前的实现，apollo-configservice本身就是一个eureka服务，所以只需要填入apollo-configservice的地址即可，如有多个，用逗号分隔（注意不要忘了/eureka/后缀）

  > 需要注意的是每个环境只填入自己环境的eureka服务地址，比如FAT的apollo-configservice是1.1.1.1:8080和2.2.2.2:8080，那么，在FAT环境的ApolloConfigDB.ServerConfig表中设置eureka.service.url为：
  >
  > ```
  > http://1.1.1.1:8080/eureka/,http://2.2.2.2:8080/eureka/
  > ```

- `namespace.lock.switch` - **一次发布只能有一个人修改开关，用于发布审核**

  这是一个功能开关，如果配置为true的话，那么一次配置发布只能是一个人修改，另一个发布

- config-service.cache.enabled - **是否开启配置缓存**

  默认false，开启后config service会缓存加载过的配置信息，从而加快后续配置获取性能，但消耗

6.3 将Config Service和Admin Service注册到单独的Eureka Server上

6.4 安装方式

安装方式有2中：直接下载安装包、源码编译后安装