docker安装apollo

```
#下载docker-compose.yml和sql 文件夹到本地目录，如 docker-quick-start。
- docker-quick-start
  - docker-compose.yml
  - sql
    - apolloconfigdb.sql
    - apolloportaldb.sql

```

通过浏览器可以访问授权中心8070、8080端口，但是只有在同一网桥内部的程序可以成功获取Apollo中的配置信息，其他服务器或者网桥以外的程序无法成功获取配置需信息。Apollo配置中心通过8080端口提供配置获取接口，其接口依赖Spring Eureka，默认情况下，Eureka使用的是网桥中该节点的IP地址端口号，以其他IP地址访问是不被允许的。

首先对Apollo提供的docker-compose文件做如下改动：

1. 为apollo-quick-start容器增加环境变量，配置实例地址为容器所在宿主机IP地址。

```
version: '2'

services:
  apollo-quick-start:
    image: nobodyiam/apollo-quick-start
    container_name: apollo-quick-start
    environment:
      EUREKA_INSTANCE_IP_ADDRESS: '192.168.0.103'
    depends_on:
      - apollo-db
    ports:
      - "8080:8080"
      - "8070:8070"
      - "8090:8090"
    links:
      - apollo-db
    #environment:
      #JAVA_OPTS: '-Xms100m -Xmx1000m -Xmn100m -Xss256k -XX:MetaspaceSize=10m -XX:MaxMetaspaceSize=250m'
      #APOLLO_CONFIG_DB_USERNAME: 'root'
      #APOLLO_CONFIG_DB_PASSWORD: 'apollo'
      #APOLLO_PORTAL_DB_USERNAME: 'root'
```

```
      #APOLLO_PORTAL_DB_PASSWORD: 'apollo'

  apollo-db:
    image: mysql:5.7
    container_name: apollo-db
    environment:
      TZ: Asia/Shanghai
      MYSQL_ALLOW_EMPTY_PASSWORD: 'yes'
      #MYSQL_ROOT_PASSWORD: 'apollo'
    depends_on:
      - apollo-dbdata
    ports:
      - "13306:3306"
    volumes:
      - ./sql:/docker-entrypoint-initdb.d
    volumes_from:
      - apollo-dbdata

  apollo-dbdata:
    image: alpine:latest
    container_name: apollo-dbdata
    volumes:
      - /var/lib/mysql

```

在docker-quick-start目录下执行docker-compose up，第一次执行会触发下载镜像等操作，需要耐心等待一些时间。

出现如下情况启动成功

```
nt(s) are active.
apollo-db              | 2022-12-10T18:14:41.652621Z 0 [Note] InnoDB: 32 non-redo rollback segment(s) are active.
apollo-db              | 2022-12-10T18:14:41.654174Z 0 [Note] InnoDB: 5.7.40 started; log sequence number 12662157
apollo-db              | 2022-12-10T18:14:41.655318Z 0 [Note] Plugin 'FEDERATED' is disabled.
apollo-db              | 2022-12-10T18:14:41.655640Z 0 [Note] InnoDB: Loading buffer pool(s) from /var/lib/mysql/ib_buffer_pool
apollo-db              | 2022-12-10T18:14:41.680327Z 0 [Note] Found ca.pem, server-cert.pem and server-key.pem in data director
y. Trying to enable SSL support using them.
apollo-db              | 2022-12-10T18:14:41.680397Z 0 [Note] Skipping generation of SSL certificates as certificate files are
present in data directory.
apollo-db              | 2022-12-10T18:14:41.680410Z 0 [Warning] A deprecated TLS version TLSv1 is enabled. Please use TLSv1.2
or higher.
apollo-db              | 2022-12-10T18:14:41.680414Z 0 [Warning] A deprecated TLS version TLSv1.1 is enabled. Please use TLSv1.
2 or higher.
apollo-db              | 2022-12-10T18:14:41.681657Z 0 [Warning] CA certificate ca.pem is self signed.
apollo-db              | 2022-12-10T18:14:41.681991Z 0 [Note] Skipping generation of RSA key pair as key files are present in d
ata directory.
apollo-db              | 2022-12-10T18:14:41.698182Z 0 [Note] Server hostname (bind-address): '*'; port: 3306
apollo-db              | 2022-12-10T18:14:41.699115Z 0 [Note] IPv6 is available.
apollo-db              | 2022-12-10T18:14:41.700622Z 0 [Note]   - '::' resolves to '::';
apollo-db              | 2022-12-10T18:14:41.703298Z 0 [Note] Server socket created on IP: '::'.
apollo-db              | 2022-12-10T18:14:41.708860Z 0 [Warning] Insecure configuration for --pid-file: Location '/var/run/mysq
ld' in the path is accessible to all OS users. Consider choosing a different directory.
apollo-db              | 2022-12-10T18:14:41.728971Z 0 [Note] InnoDB: Buffer pool(s) load completed at 221211  2:14:41
apollo-db              | 2022-12-10T18:14:41.771421Z 0 [Note] Event Scheduler: Loaded 0 events
apollo-db              | 2022-12-10T18:14:41.771884Z 0 [Note] mysqld: ready for connections.
apollo-db              | Version: '5.7.40'  socket: '/var/run/mysqld/mysqld.sock'  port: 3306  MySQL Community Server (GPL)
apollo-quick-start     | Waiting for config service startup.......
apollo-quick-start     | Config service started. You may visit http://localhost:8080 for service status now!
apollo-quick-start     | Waiting for admin service startup..
apollo-quick-start     | Admin service started
apollo-quick-start     | ==== starting portal ====
apollo-quick-start     | Portal logging file is ./portal/apollo-portal.log
apollo-quick-start     | Application is running as root (UID 0). This is considered insecure.
apollo-quick-start     | Started [264]
apollo-quick-start     | Waiting for portal startup.....
apollo-quick-start     | Portal started. You can visit http://localhost:8070 now!
```

访问8080端口，会发现此处显示宿主机IP地址，则修改成功：

Instance Info

Name	Value
ipAddr	192.168.0.103
status	UP

访问apollo配置中心：localhost:8070 查看查看管理员工具->系统信息

系统信息

系统版本: java-2.0.1

环境列表来自于 ApolloPortalDB.ServerConfig 中的 apollo.portal.envs 配置，可以到 系统参数页面配置，更多信息可以参考分布式部署指南中的 apollo.portal.envs - 可支持的环境列表章节。

Meta Server 地址展示了该环境配置的 Meta Server 信息，更多信息可以参考分布式部署指南中的配置 apollo-portal 的 meta service 信息章节。

环境: DEV

Active: true

Meta Server 地址: http://localhost:8080

Config Services

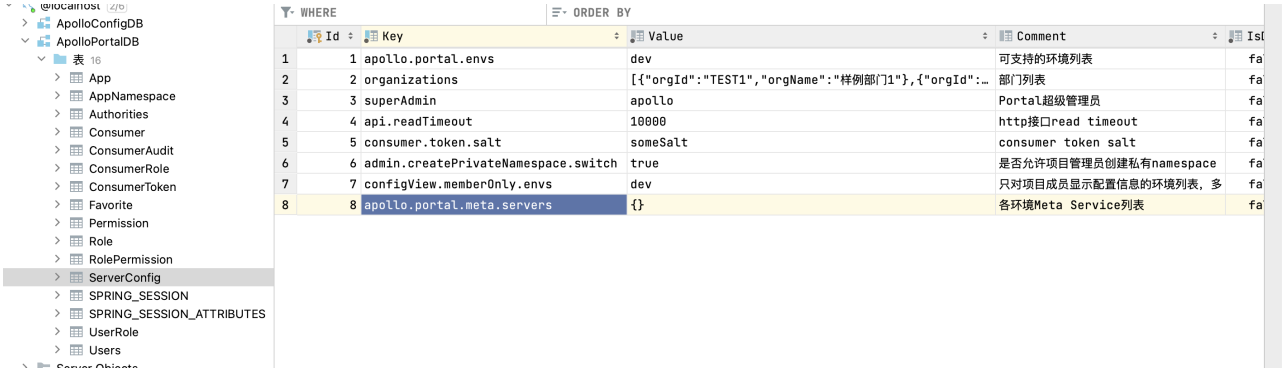
Name	Instance Id	Home Page Url	Check Health
APOLLO-CONFIGSERVICE	8b2a4e92abc1:apollo-configservice:8080	http://192.168.0.103:8080/	Check

Admin Services

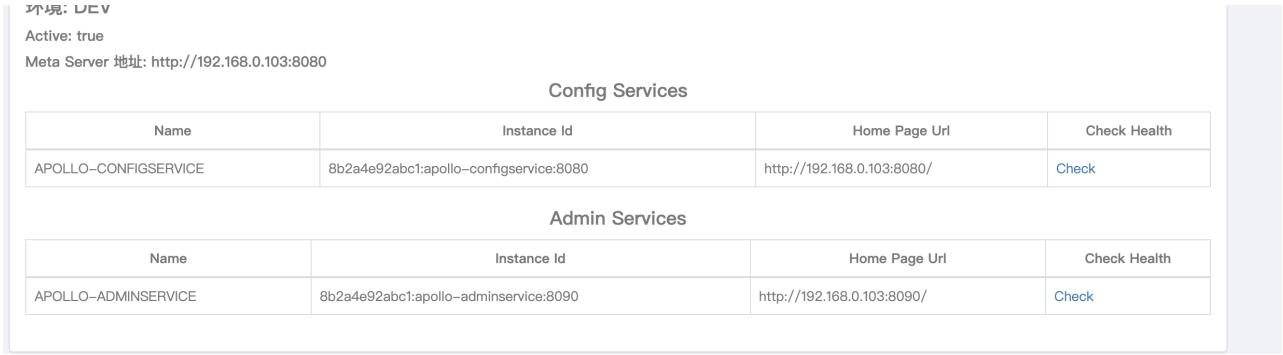
Name	Instance Id	Home Page Url	Check Health
APOLLO-ADMINSERVICE	8b2a4e92abc1:apollo-adminservice:8090	http://192.168.0.103:8090/	Check

发现Config Services，Admin Service信息配置正确，已经配置为宿主机的IP地址

Meta serve地址并不是宿主机的ip地址。此时可以连接数据库localhost:13306

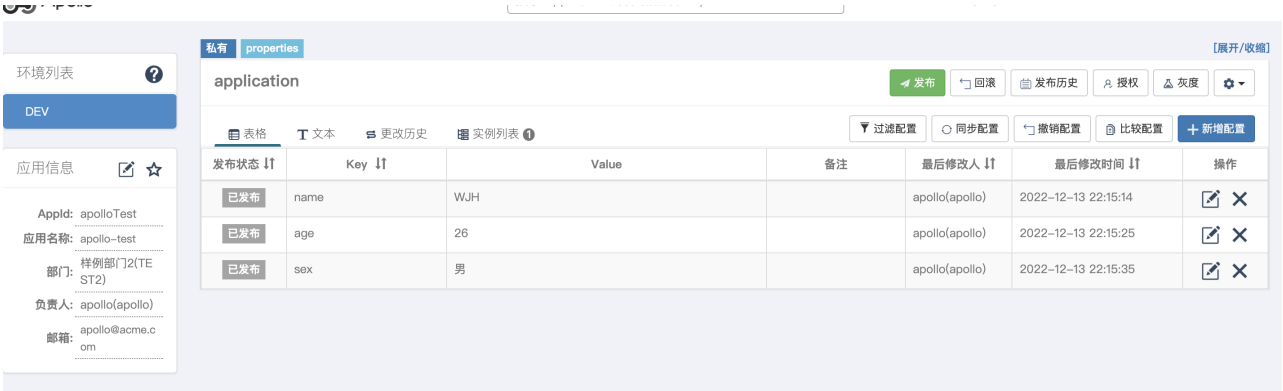


将数据库ApolloPortalDB的ServerConfig表中Key=apollo.portal.meta.servers项中value改为{"DEV":"http://【宿主机IP】:8080"}并重新启动容器



这回Meta Server的地址变成了正确的宿主机ip地址

添加配置项，并发布



启动客户端服务，查看apollo配置参数

```
    1 个用法
    @Value ("${name:AA}")
    private String name;
    1 个用法
    @Value ("${sex:F}")
    private String sex;

    1 个用法
    @Value ("${age:13}")
    private String age;


    新 *
    @GetMapping("apollo1")
    public void testApollo1(){
        log.info ("name: {}",name);
        log.info ("sex: {}",sex);
        log.info ("age: {}",age);
    }
```

```
Application
:33:07.247  INFO 94289 --- [io-30001-exec-1] o.a.c.c.C.[Tomcat].[localhost].[/]       : Initializing Spring DispatcherServlet 'dispatcl
:33:07.248  INFO 94289 --- [io-30001-exec-1] o.s.web.servlet.DispatcherServlet        : Initializing Servlet 'dispatcherServlet'
:33:07.265  INFO 94289 --- [io-30001-exec-1] o.s.web.servlet.DispatcherServlet        : Completed initialization in 16 ms
:33:07.319  INFO 94289 --- [io-30001-exec-1] c.g.a.s.controller.ApolloController      : name: WJH
:33:07.319  INFO 94289 --- [io-30001-exec-1] c.g.a.s.controller.ApolloController      : sex: 男
:33:07.320  INFO 94289 --- [io-30001-exec-1] c.g.a.s.controller.ApolloController      : age: 26
```

控制台已经打印出配置中心配置的信息，当我们修改配置中心信息将age修改为18岁，服务会监听到配置中心修改的信息

```
2022-12-13 22:33:07.319  INFO 94289 --- [io-30001-exec-1] c.g.a.s.controller.ApolloController      : name: WJH
2022-12-13 22:33:07.319  INFO 94289 --- [io-30001-exec-1] c.g.a.s.controller.ApolloController      : sex: 男
2022-12-13 22:33:07.320  INFO 94289 --- [io-30001-exec-1] c.g.a.s.controller.ApolloController      : age: 26
2022-12-13 22:34:39.397  INFO 94289 --- [Apollo-Config-1] c.f.a.s.p.AutoUpdateConfigChangeListener : Auto update apollo changed value successfully, new value: 18岁,
 key: age, beanName: apolloController, field: com.geektime.apollo.springbootapollo.controller.ApolloController.age
```

再次调用，打印

```
2022-12-13 22:34:39.397  INFO 94289 --- [Apollo-Config-1] c.f.a.s.p.AutoUpdateConfigChangeListener : Auto update apollo changed value successfully, new value: 18岁,
 key: age, beanName: apolloController, field: com.geektime.apollo.springbootapollo.controller.ApolloController.age
2022-12-13 22:35:39.918  INFO 94289 --- [io-30001-exec-4] c.g.a.s.controller.ApolloController      : name: WJH
2022-12-13 22:35:39.926  INFO 94289 --- [io-30001-exec-4] c.g.a.s.controller.ApolloController      : sex: 男
2022-12-13 22:35:39.926  INFO 94289 --- [io-30001-exec-4] c.g.a.s.controller.ApolloController      : age: 18岁
```

发现配置已经更改

通过API方式调用获取默认namespace下的属性

```java
/**
 * 通过API调用
 */
邬锦华 *
@GetMapping("apollo2")
public void testApollo2(){
    Config config = ConfigService.getAppConfig();
    String name1 = config.getProperty( s: "name", s1: "");
    String age = config.getProperty( s: "age", s1: "");
    String sex = config.getProperty( s: "sex", s1: "");
    log.info ("name: {}",name1);
    log.info ("age: {}",age);
    log.info ("sex: {}",sex);

}

/**
```

```
io-30001-exec-1] o.a.c.c.C.[Tomcat].[localhost].[/]       : Initializing Spring DispatcherServlet 'dispatcherServlet'
io-30001-exec-1] o.s.web.servlet.DispatcherServlet        : Initializing Servlet 'dispatcherServlet'
io-30001-exec-1] o.s.web.servlet.DispatcherServlet        : Completed initialization in 10 ms
io-30001-exec-1] c.g.a.s.controller.ApolloController      : name: WJH
io-30001-exec-1] c.g.a.s.controller.ApolloController      : age: 18岁
io-30001-exec-1] c.g.a.s.controller.ApolloController      : sex: 男
```

获取yaml/yml格式的namespace

```java
/**
 * yaml/yml格式的namespace
 */
邬锦华 *
@GetMapping("apollo3")
public void testApollo3(){
    Config config = ConfigService.getConfig( namespace: "testspace.yaml");
    String property = config.getProperty ( key: "mybatis-plus.mapper-locations", defaultValue: "");
    String property2 = config.getProperty ( key: "mybatis-plus.global-config.db-config.id-type", defaultValue: "");
    log.info ("property: {}",property);
    log.info ("property2: {}",property2);

}
}
```

```
001-exec-1] o.a.c.c.C.[Tomcat].[localhost].[/]       : Initializing Spring DispatcherServlet 'dispatcherServlet'
001-exec-1] o.s.web.servlet.DispatcherServlet        : Initializing Servlet 'dispatcherServlet'
001-exec-1] o.s.web.servlet.DispatcherServlet        : Completed initialization in 6 ms
001-exec-1] c.g.a.s.controller.ApolloController      : property: classpath*:/mapper/**/*.xml
001-exec-1] c.g.a.s.controller.ApolloController      : property2: AUTO
```