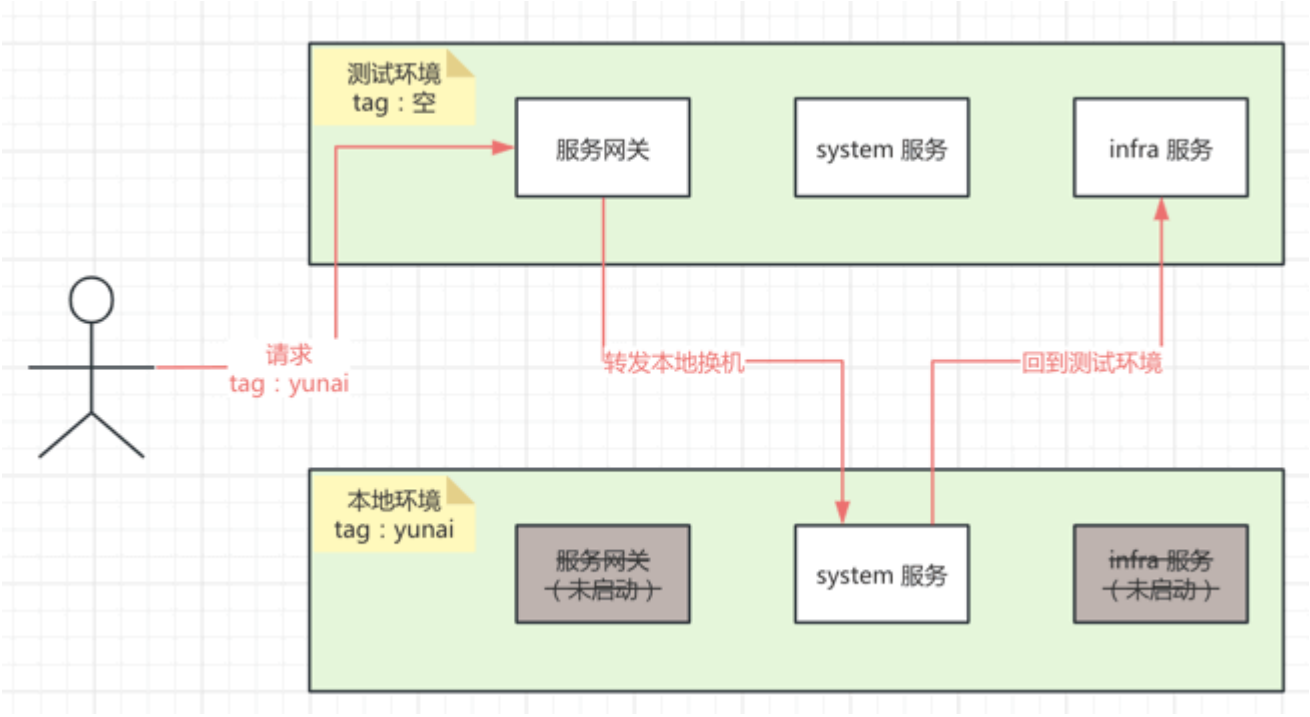
芋道源码 2023-07-28

微服务调试（必读）

1. 多环境 env 组件

在微服务架构下，多服务的调试是个非常大的痛点：在大家使用 **同一个** 注册中心时，如果多个人在本地启动 **相同** 服务，可能需要调试的一个请求会打到其他开发的本地服务，实际是期望达到自己本地的服务。
一般的解决方案是，使用 **不同** 注册中心，避免出现这个情况。但是，服务一多之后，就会产生新的痛点，同时本地启动所有服务很占用电脑内存。
因此，我们实现了 `yudao-spring-boot-starter-env` 组件，通过 Tag 给服务打标，实现在使用 **同一个** 注册中心的情况下，本地只需要启动需要调试的服务，并且保证自己的请求，必须达到自己本地的服务。如下图所示：



- 测试环境：启动了 gateway、system、infra 服务；本地环境：只启动了 system 服务
- 请求时，带上 `tag = yunai` ，优先请求本地环境 + `tag = yunai` 的服务：
  - ① 由于本地未启动 gateway 服务，所以请求打到测试环境的 gateway 服务
  - ② 由于请求 `tag = yunai` ，所以转发到本地环境的 system 服务
  - ③ 由于本地未启动 infra 服务，所以转发回测试环境的 infra 服务

2. 功能演示

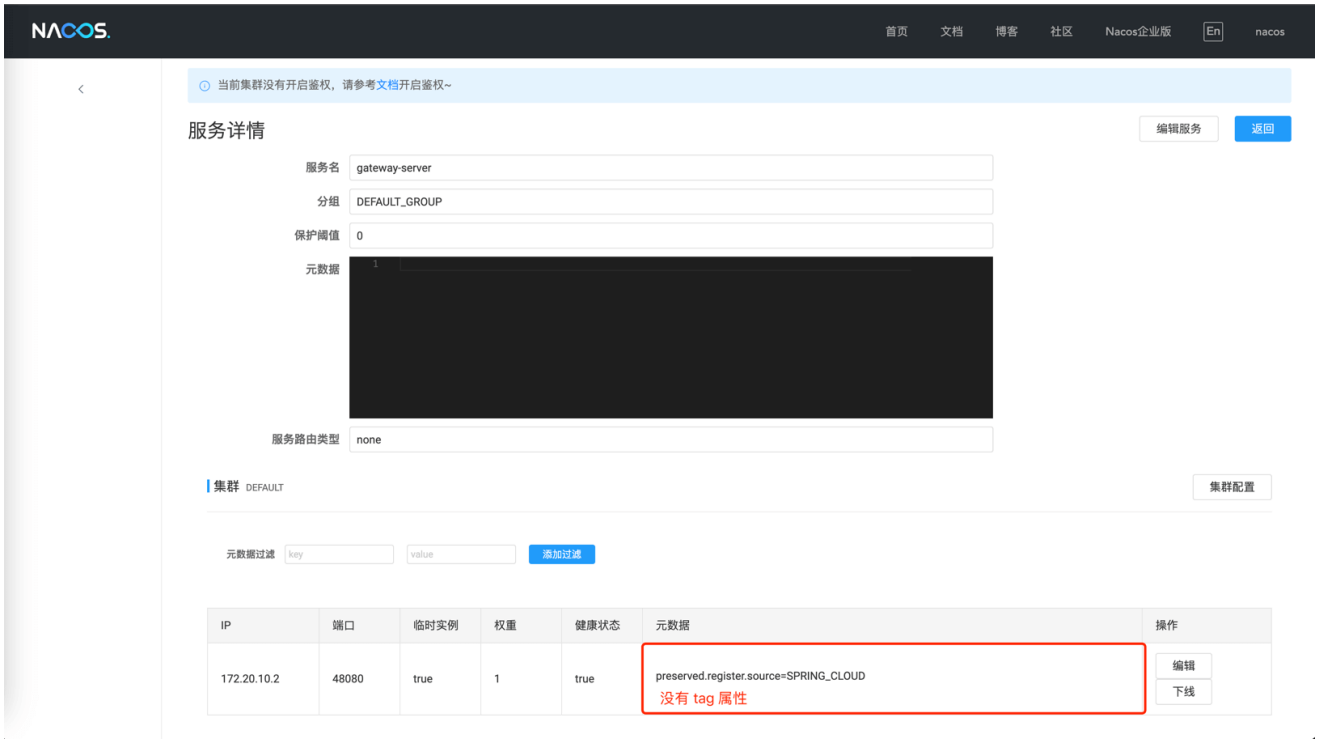
在本地模拟，启动三个进程，如下图所示：

- `tag` 为空的 gateway、system 服务
- `tag` 为本机 `hostname` （例如说我本地是 `Yunai.local` ）的 system 服务

> **注意！！！**
>
> `hostname` 是你的主机名：
>
> - Windows 在 cmd 里输入 hostname
> - MacOS 在 terminal 里输入 hostname

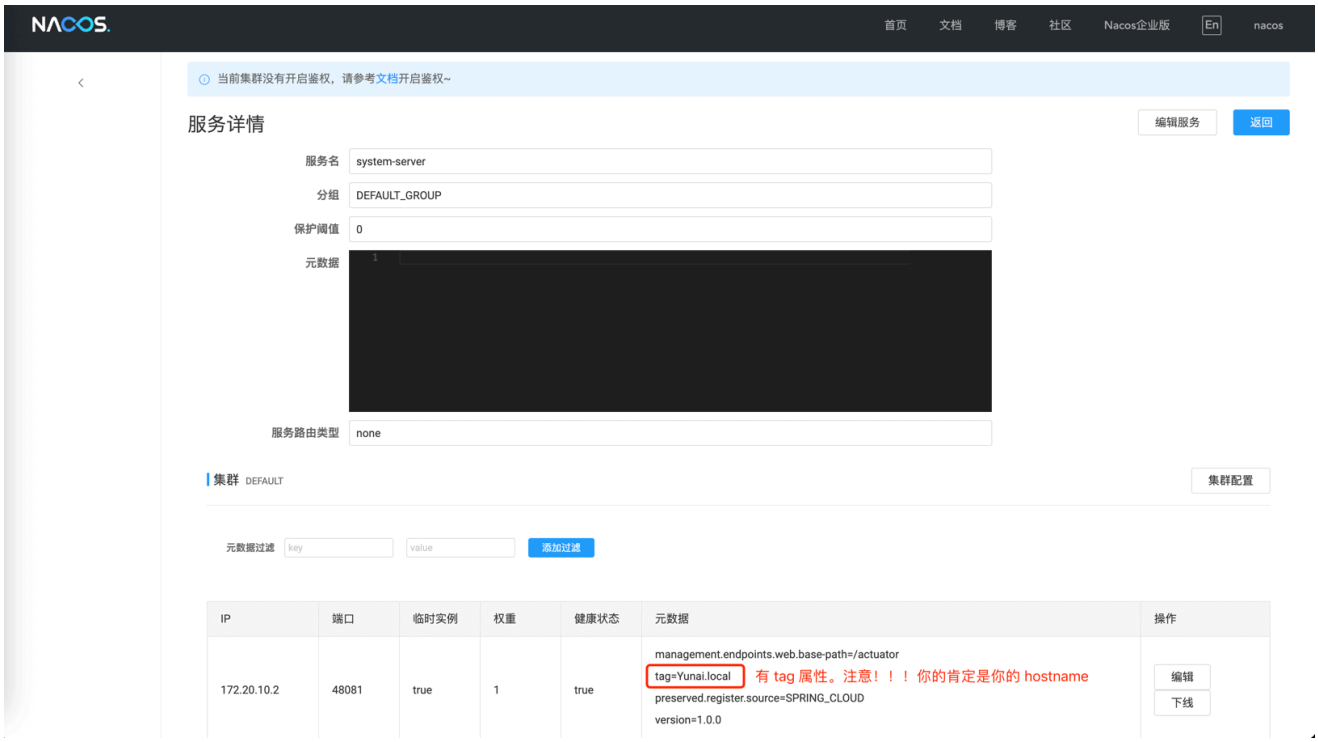
第一步，启动 gateway 服务

直接运行 GatewayServerApplication 类，启动 gateway 服务。此时，我们可以在 Nacos 看到该实例，它是没 `tag` 属性。如下图所示：

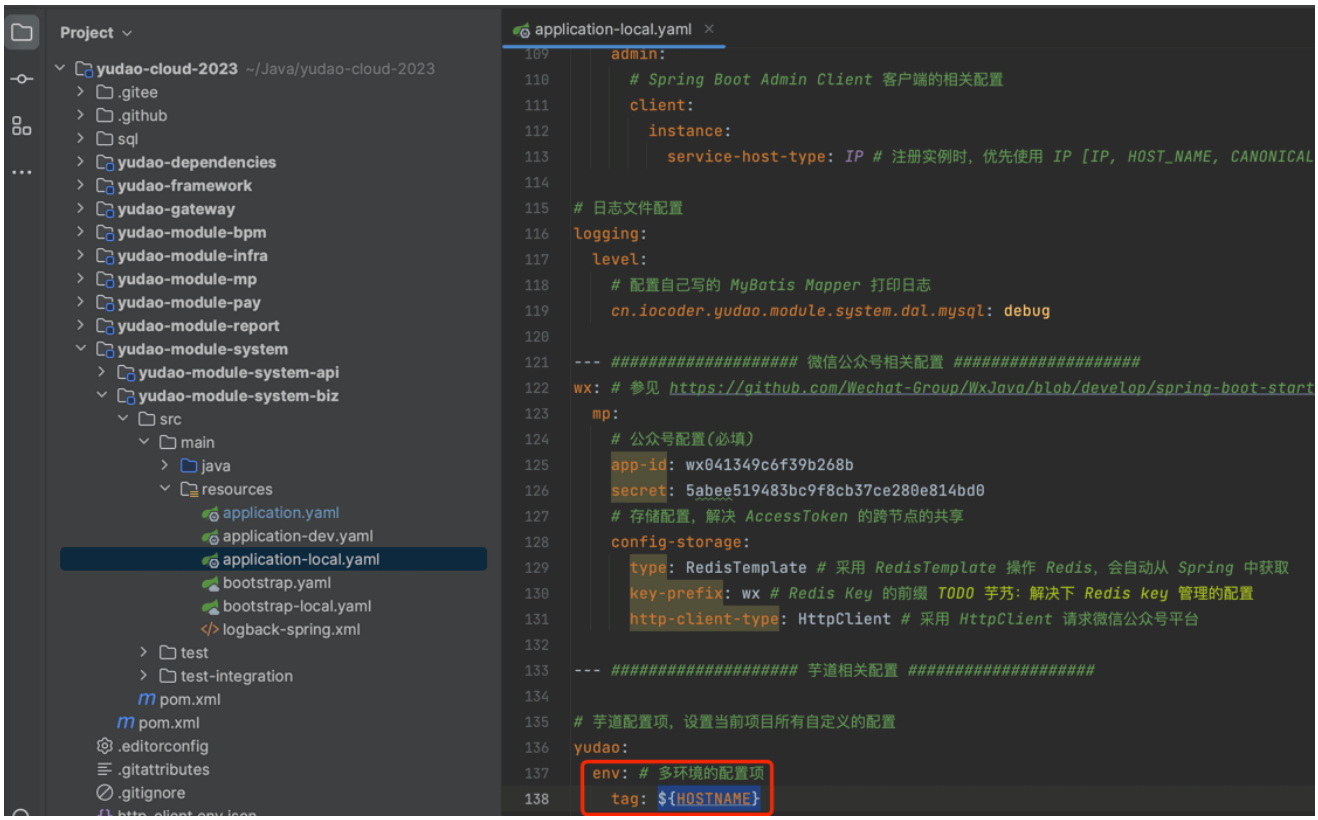


第二步，启动 system 服务【有 tag】

运行 SystemServerApplication 类，启动 system 服务。此时，我们可以在 Nacos 看到该实例，它是有 `tag` 属性。如下图所示：

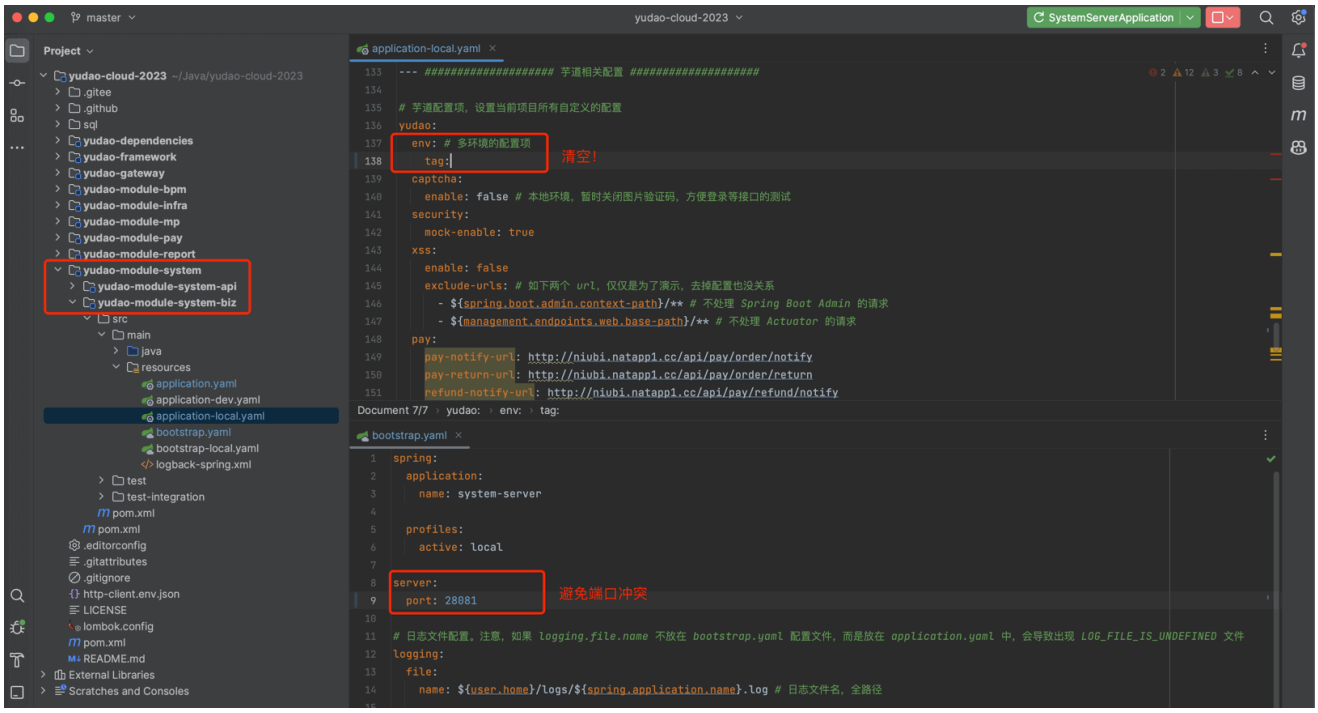


因为我们默认在 `application-local.yaml` 配置文件里，添加了 `yudao.env.tag` 配置项为 `${HOSTNAME}` 。如下图所示：

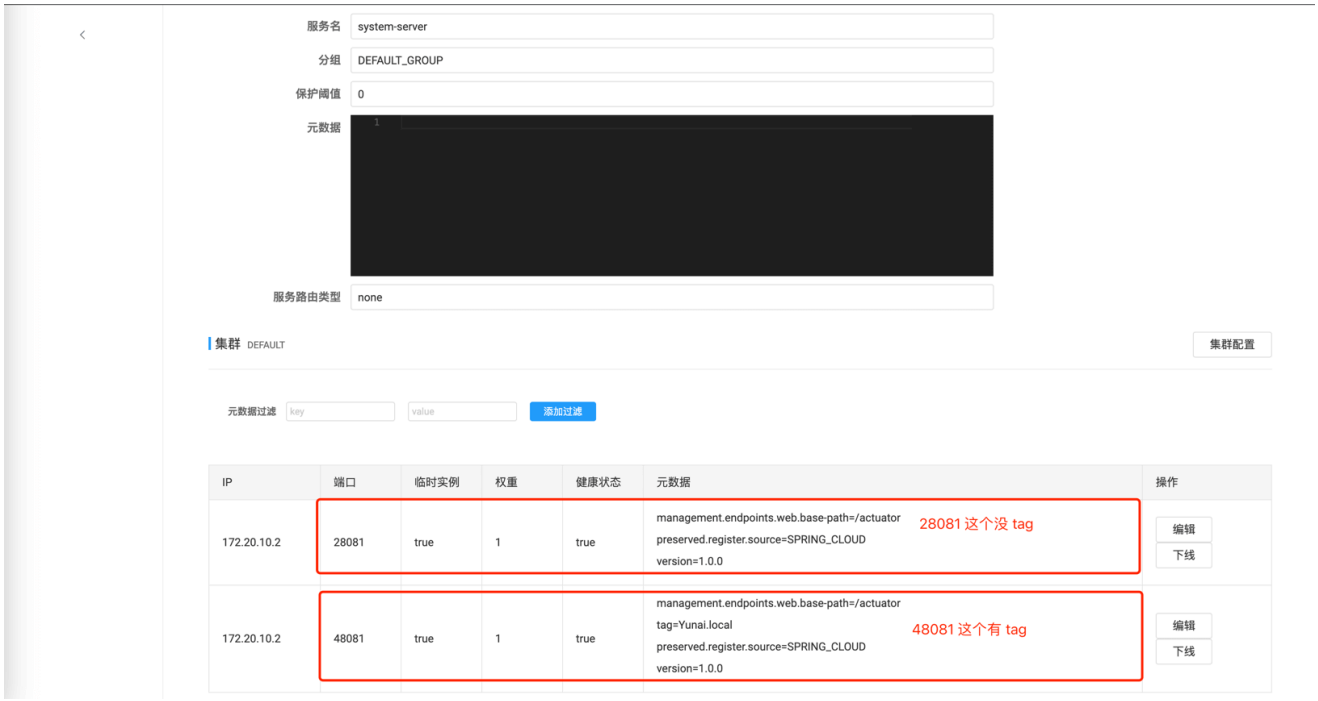


第三步，启动 system 服务 【无 tag】

① 修改 system 服务的端口为 28081， `yudao.env.tag` 配置项为空。如下图所示：

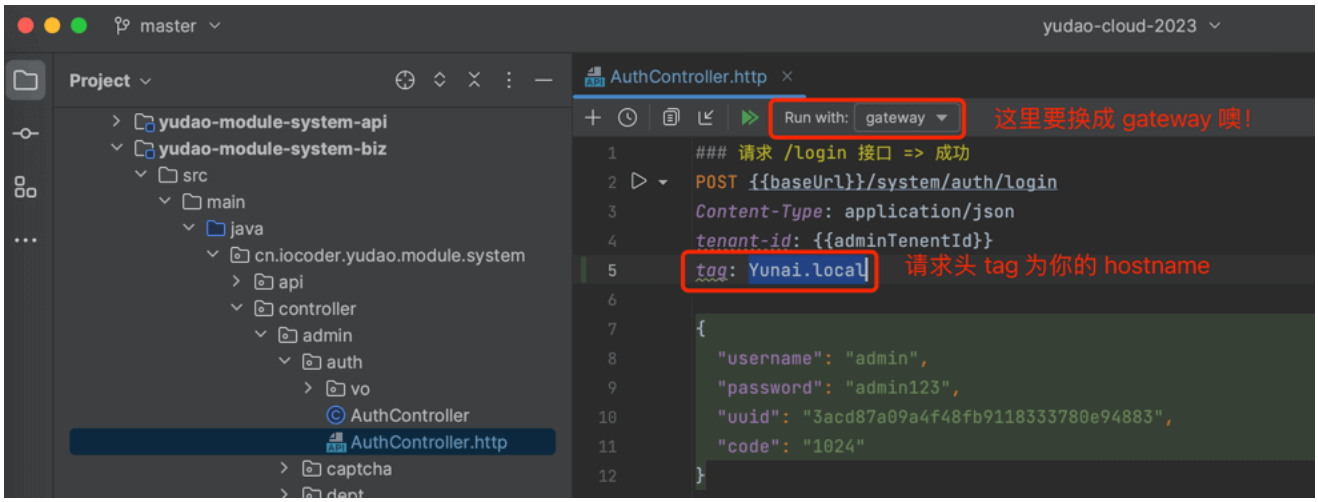


② 再一个 SystemServerApplication，**额外**启动 system 服务。此时，我们可以在 Nacos 看到该实例，它是没 `tag` 属性。如下图所示：



第四步，请求测试

① 打开 `AuthController.http` 文件，设置第一个请求的 `tag` 为 `Yunai.local` （要替换成你的 hostname），如下图所示：



② 点击前面的绿色小箭头，发起请求。从 IDEA 控制台的日志可以看到，只有有 tag 的 system 服务才会被调用。

你可以多点几次，依然是这样的情况噢！

3. 实现原理

① 在服务注册时，会将 `yudao.env.tag` 配置项，写到 Nacos 服务实例的元数据，通过 EnvEnvironmentPostProcessor 类实现。

② 在服务调用时，通过 EnvLoadBalancerClient 类，筛选服务实例，通过服务实例的 `tag` 元数据，匹配请求的 `tag` 请求头。

③ 在网关转发时，通过 GrayLoadBalancer 类，效果和 EnvLoadBalancerClient 一致。

4. 未来的拓展

除了微服务调试比较麻烦外，MQ 消息队列的消费调试也是个麻烦的事儿，未来也会进行支持。实现的效果如下：

- 本地发送的 MQ 消息，优先被本地启动的消费者进行处理，方便调试。
- 如果本地没有启动消费者，则被测试环境的消费者进行处理，避免一致不被消费。

← 验证码 注册中心 Nacos→