

# 创建rest服务工程

- <https://spring.io/guides/gs/rest-service/> 参考这个文档做基于spring boot的rest服务
- 我做例子的代码在这: <https://github.com/caishu/gs-sample.git>
- 我是从这里开始创建spring boot工程的: <https://start.spring.io/> 记得packaging选jar; 另外, 选上Actuator是个好的选择, 虽然我没有选
- 这步在mac或者linux做都可以; 不同之处在于, mac有图形环境, 编辑文件可能会方便些
- 验证的方法是mvn clean package可以BUILD SUCCESS
- GreetingController.java文件20行, 我加入了获取HOSTNAME的代码, 用来后期验证负载均衡功能

# 部署rest服务

- 我使用S2I的方式部署rest服务，主要是为了演示用jar和war的方式处理spring boot都可以，差别不大
- 参考文档：<https://blog.codecentric.de/en/2016/03/deploy-spring-boot-applications-openshift/>
- `oc new-app codecentric/springboot-maven3-centos~https://github.com/caishu/gs-sample.git` 这个是在linux上执行的；注意用developer用户执行（如果不确认自己是哪个用户，可以用`oc whoami`看看自己是谁；或者直接`oc login -u developer`再登录一次）
- 这步会用springboot-maven3-centos这个s2i镜像，创建一个openshift的服务
- `oc expose svc gs-sample` 。这步是给刚才部署的服务添加一个route
- 这时候从mac浏览器访问 `http://gs-sample-myproject.127.0.0.1.nip.io/greeting`，就能看到返回的json串了。这里，注意这个域名，是从mac访问linux时候用的。这个域名需要解析成127.0.0.1，如果解析错误，可以在mac的/etc/hosts里边加一个记录
- 查看gs-sample服务的详情：<https://127.0.0.1:8443/console/project/myproject/browse/services/gs-sample?tab=details> 。这里可以看到Hostname: gs-sample.myproject.svc 这个域名是openshift内部pod之间访问时候用的。这个需要记下来（实际这个是自动生成的，规则是service\_name+project\_name+'svc'），一会创建的另一个消费这个REST的程序里用这个地址