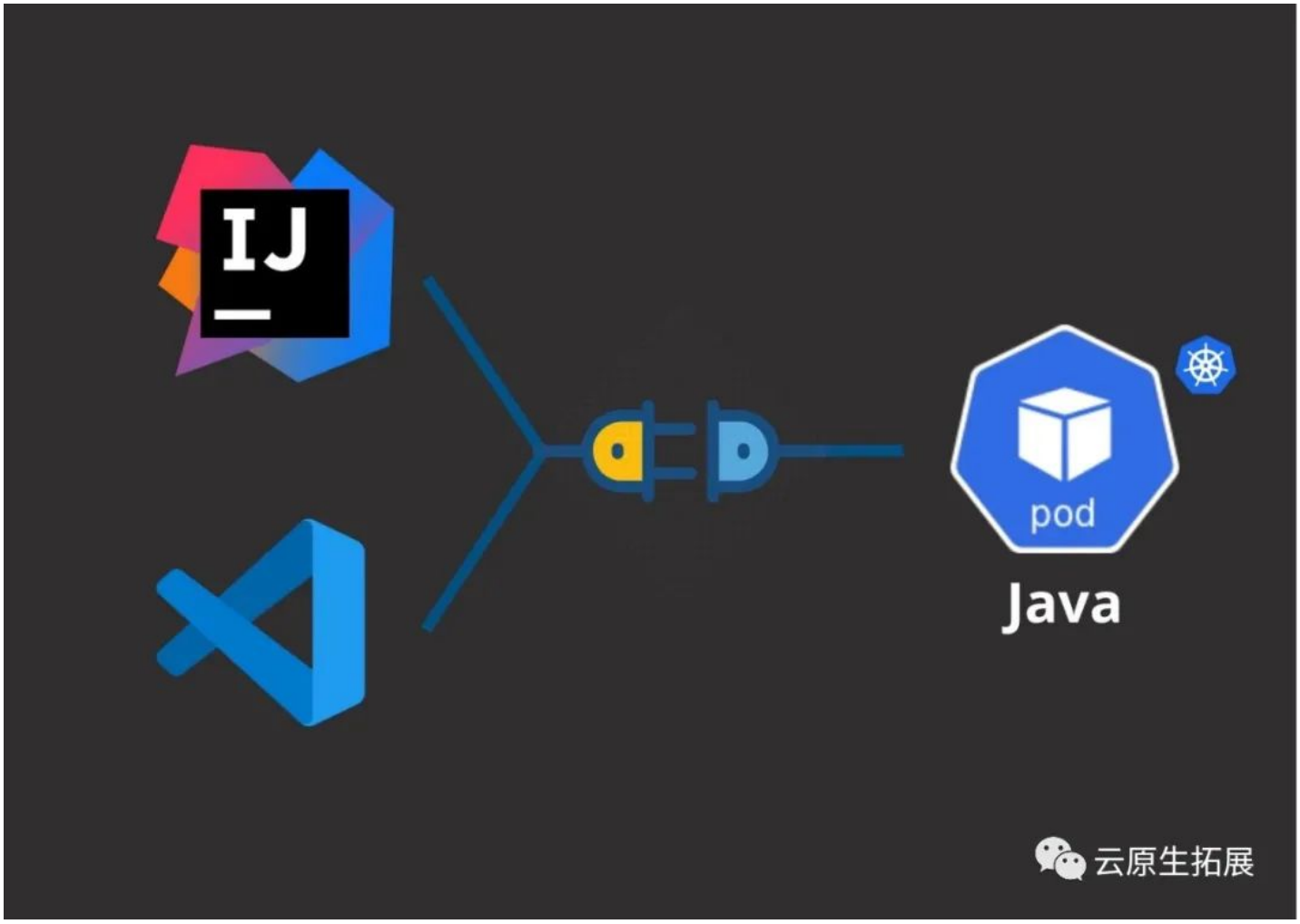
27Kubernetes 系列（二十四）如何远程 debug kubernetes pod 中的 java 程序

Kubernetes 系列（二十四）如何远程 debug kubernetes pod 中的 java 程序

这是一篇关于如何使用Intellij或VScode调试运行在Kubernetes Pod中的Java应用程序的快速文章。



项目配置

假设我们有一个Java应用程序，它提供了一些REST端点，您希望使用IDE对其进行调试。

让我们在Spring Boot中创建一个简单的应用程序，它提供一个简单的REST端点。

```
@RestController
public class WebController {
    @GetMapping("/")
    public ResponseEntity get(){
        return ResponseEntity.ok("All Works fine");
    }
}
```

现在，我们将使用谷歌的JIB maven 插件创建这个应用程序的 docker 镜像。

```
<plugin>
    <groupId>com.google.cloud.tools</groupId>
    <artifactId>jib-maven-plugin</artifactId>
    <version>3.2.1</version>
    <configuration>
        <from>
            <image>openjdk:17</image>
        </from>
        <to>
            <image>ghcr.io/amrutprabhu/${project.name}:${project.version}</image>
        </to>
    </configuration>
    <executions>
        <execution>
            <phase>verify</phase>
            <goals>
                <goal>build</goal>
            </goals>
        </execution>
    </executions>
</plugin>
```

还有其他插件可以dockerize您的应用程序，您可以在我的文章 这里 （ https://refactorfirst.com/3-ways-to-create-spring-boot-docker-images ）中探索一些流行的插件。

现在，当我们运行 mvn clean verify 时，docker 镜像将被构建并推送到 GitHub 存储库中。

部署应用到 Kubernetes 中

假设您已经有一个Kubernetes集群来部署您的应用程序。如果没有，可以使用 https://k3s.io 运行本地Kubernetes集群。

我们将使用这个K3s集群部署我们的应用程序。

为了部署我们的应用程序，我们将创建一个简单的应用，其部署定义如下所示。

```yaml
apiVersion: apps/v1
kind: Deployment
metadata:
  name: application-deployment
spec:
  selector:
    matchLabels:
      app: application
  template:
    metadata:
      labels:
         app: application
    spec:
      containers:
        - image: ghcr.io/amrutprabhu/remote-application:1.0.0-SNAPSHOT
          imagePullPolicy: Always
          ports:
            - name: http
              containerPort: 8080
              protocol: TCP
            - name: debug-port
              containerPort: 5005
              protocol: TCP
          env:
            - name: JAVA_TOOL_OPTIONS
              value: "-Xdebug -agentlib:jdwp=transport=dt_socket,address=0.0.0.0:5005,server=y,suspend=n"
```

对我们来说，最重要的是部署中设置的环境变量 `JAVA_TOOL_OPTIONS` 。

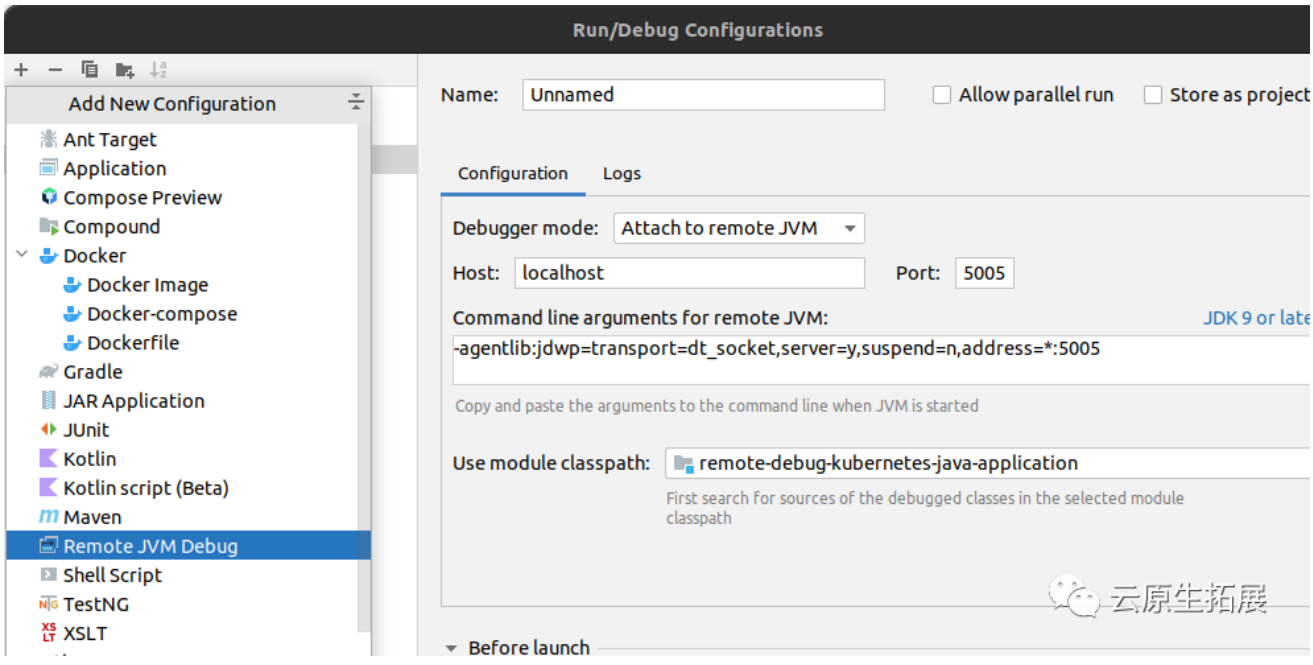
由于我们使用的是OpenJDK 基础镜像，JVM 将获取这个环境变量，以允许您将一个调试器附加到端口 `5005` 。

部署应用程序之后，需要将端口转发到 `5005` 以附加调试器。

```
kubectl port-forward <your pod name> 5005:5005
```

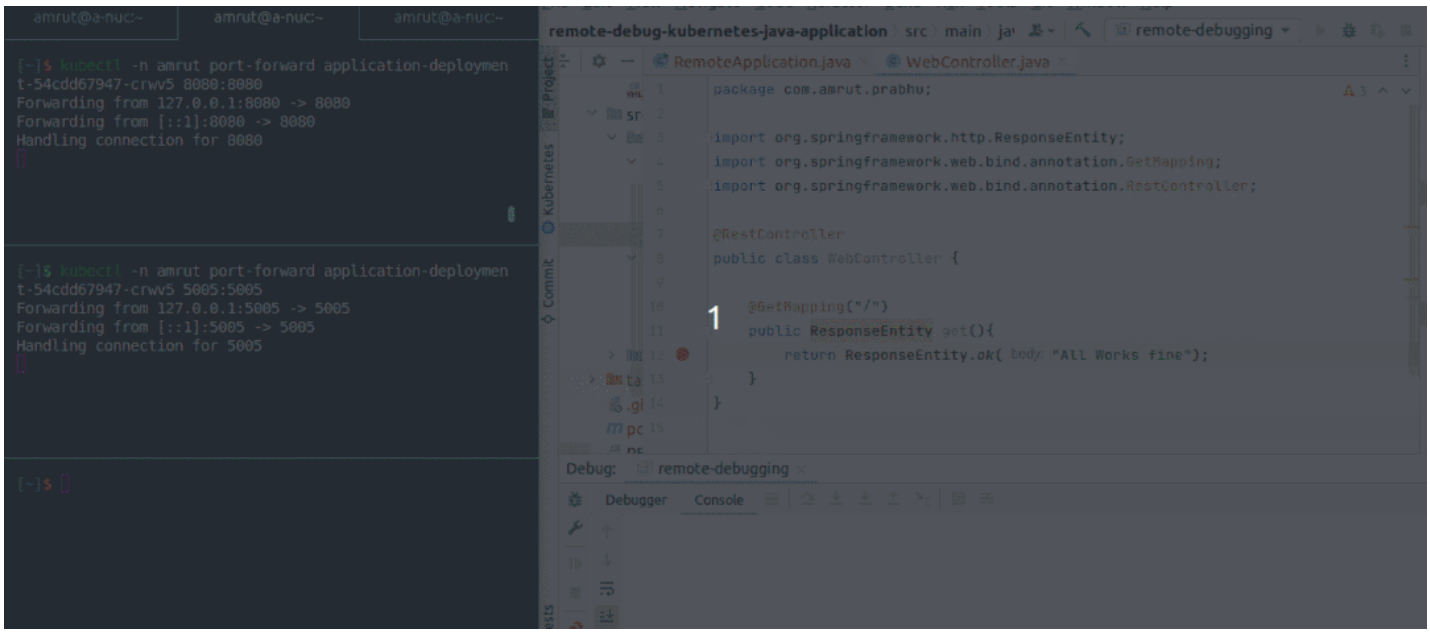
使用 Intellij 附加远程调试器

要附加调试器，转到右下角的运行部分并添加“Remote JVM Debug”运行配置。



可以看到，上面显示的命令行参数与我们在部署文件中指定为环境变量的值相同。

就是这样。现在可以运行配置并附加调试器。



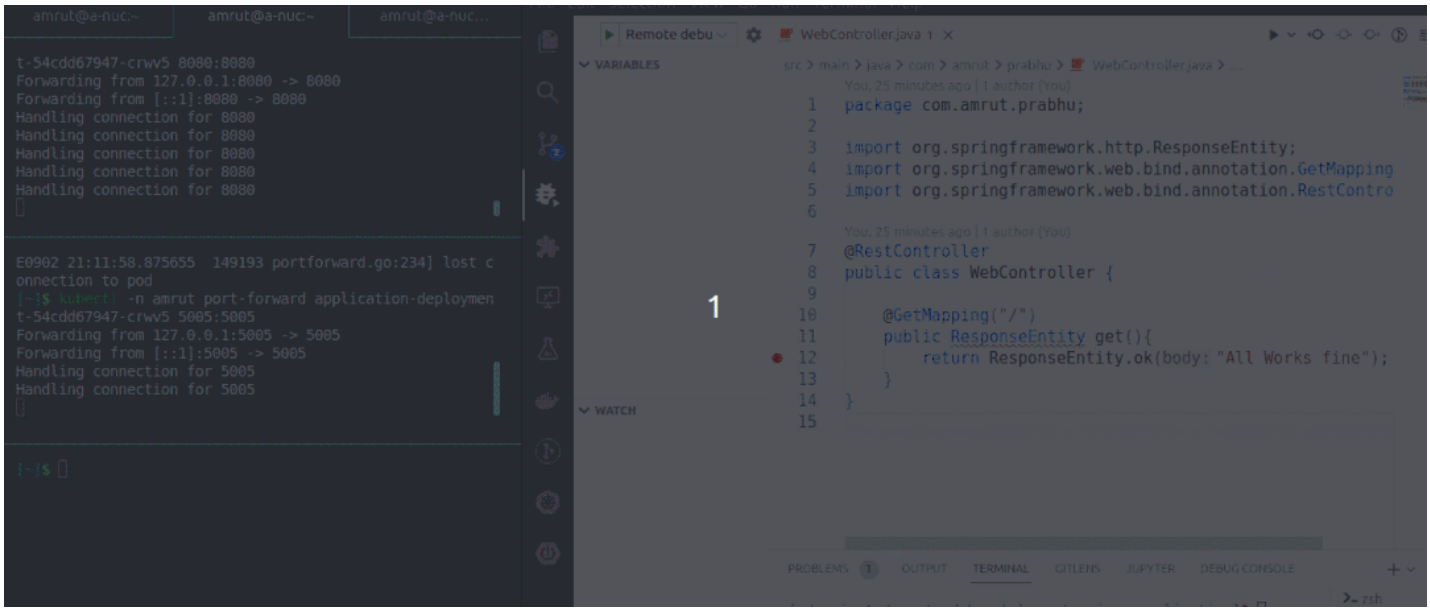
使用 VSCode 附加远程调试器

要使用 VScode 附加一个远程调试器，我们需要添加如下所示的启动配置

```
{
"version": "0.2.0",
"configurations": [    {
      "type": "java",
      "name": "Remote debugging RemoteApplication", // name for you your configuration
      "request": "attach",
      "hostName": "localhost",
      "projectName": "remote-application", //  your java project
      "port": "5005"    // port to attach to
    }
  ]
```

要添加这个启动配置，点击左边栏的“Run and Debug”，然后点击顶部的齿轮图标打开“launch.json”。

完成此操作后，启动配置，将附加调试器。添加断点并在端点上向调试器发送请求以暂停执行，如下所示。



欢迎关注我的公众号“**云原生拓展**”，原创技术文章第一时间推送。