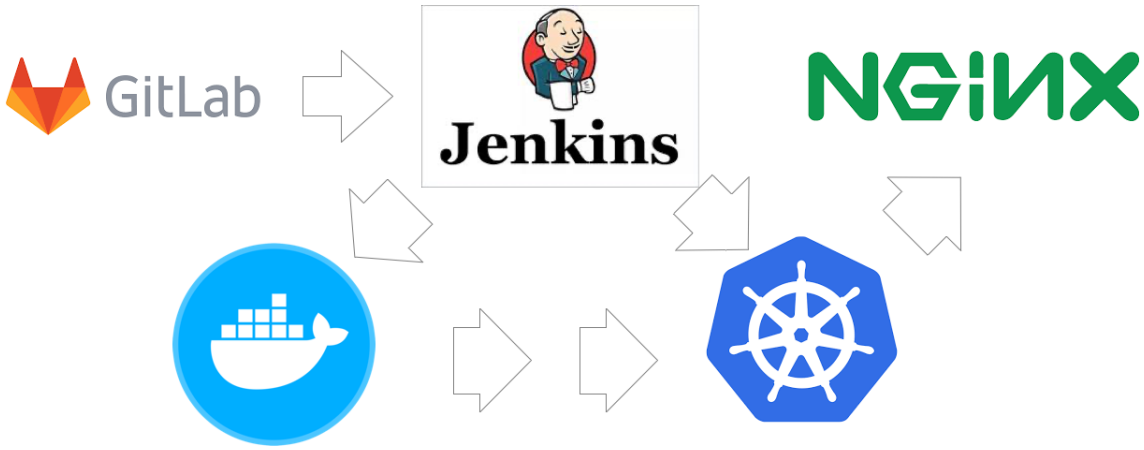
容器虚拟化技术和自动化部署



k8s快速入门之资源文件

1.idea安装k8s插件

1.1idea插件官网地址

```
https://plugins.jetbrains.com/

kubernetes地址：
https://plugins.jetbrains.com/plugin/10485-kubernetes
```

1.2查找对应自己idea版本的k8s插件信息

```
help->about->查看idea内部版本信息  一定要注意版本信息，否则无法安装
```

1.3离线安装k8s插件

```
因国外网站网速较慢，在线安装有安装失败的危险。推荐大家下载idea对应版本的插件后，进行离线安装

193.5662.65
settings->plugins->Install Plugin from Disk->插件安装目录

安装完成后重启idea开发工具
```

2.idea配置SSH客户端

目标：在idea中打开终端操作k8s集群master节点。

2.1idea配置

```
settings->Tools->SSH Configurations->新建
```

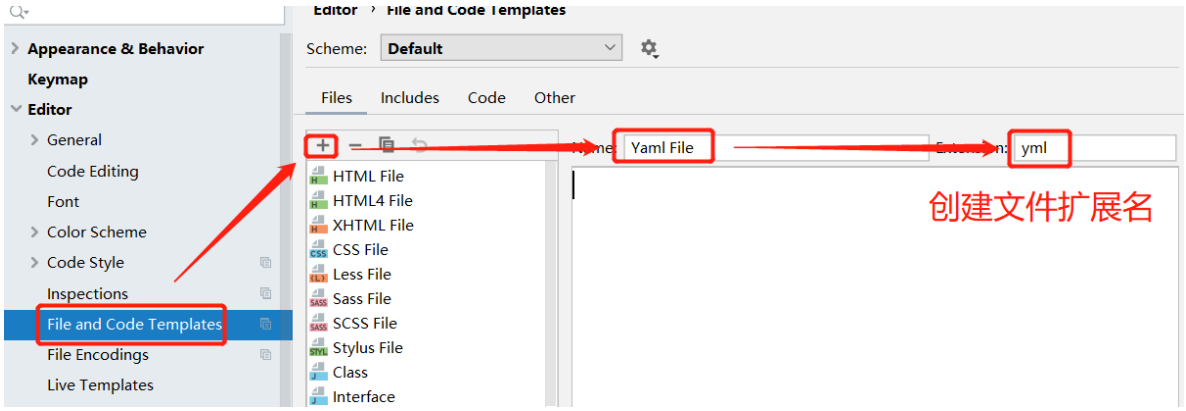
2.2使用SSH客户端

```
Tools->Start SSH session->选择我们刚刚配置的ssh客户端名称
```

2.3新建yml类型文件

idea默认没有yml文件类型。可以通过new->file->手工输入*.yml创建yml类型文件。也可以通过配置增加yml类型文件。

```
settings->Edtior->File and Code Template->file->+(新建)
```



3.Remote Host

目标：将idea工程中的文件上传k8s集群master节点。

3.1idea配置

```
Tools->Deployment->Configurations->配置Remote Host
```

3.2使用Remote Host

```
可以将本工程中的文件上传k8s集群
```

4.NameSpace

4.1创建NameSpace

```
操作指南：
settings->Editor->Live Template->Kubernetes->查看自动生成的模板信息内容
```

4.1.1lagounamespace.yml

在文件中输入kres，根据模板快速生成yml文件信息

```yaml
apiVersion: v1
kind: Namespace
metadata:
  name: lagou
```

通过idea的Remote Host快速将yml文件上传k8s集群进行测试

```
mkdir -p /data/namespaces
cd /data/namespaces

kubectl apply -f lagounamespace.yml
```

4.2删除NameSpace

```
kubectl delete -f lagounamespace.yml
```

5.pod

5.1创建pod

在idea工程resource/pod/tomcatpod.yml

```yaml
apiVersion: v1
kind: Pod
metadata:
  name: tomcat9
  labels:
    app: tomcat9
spec:
  containers:
    - name: tomcat9
      image: tomcat:9.0.20-jre8-alpine
      imagePullPolicy: IfNotPresent
  restartPolicy: Always
```

5.2镜像下载策略、重启策略

```
imagePullPolicy:
    Always:总是拉取 pull
    IfNotPresent:如果本地有镜像，使用本地，如果本地没有镜像，下载镜像。
    Never:只使用本地镜像，从不拉取
```

```
restartPolicy:
    Always：只要退出就重启。
    OnFailure：失败退出时（exit code不为0）才重启
    Never：永远不重启
```

5.3运行pod

```
kubectl apply -f tomcatpod.yml
```

5.4测试pod

```
curl 10.81.58.196:8080
```

5.5删除pod

```
kubectl delete -f tomcatpod.yml
```

6.deployment

6.1创建deployment

在idea工程resource/deployment/tomcatdeployment.yml

```yaml
apiVersion: apps/v1
kind: Deployment
metadata:
  name: tomcat-deployment
  labels:
    app: tomcat-deployment
spec:
  replicas: 3
  template:
    metadata:
      name: tomcat-deployment
      labels:
        app: tomcat
```

```
    spec:
      containers:
        - name: tomcat-deployment
          image: tomcat:9.0.20-jre8-alpine
          imagePullPolicy: IfNotPresent
      restartPolicy: Always
  selector:
    matchLabels:
      app: tomcat
```

matchLabels

```

总结：
在Deployment中必须写matchLables
在定义模板的时候必须定义labels,因为Deployment.spec.selector是必须字段,而他又必须和template.labels对应
```

6.2运行deployment

```
kubectl apply -f tomcatdeployment.yml
```

6.3控制器类型

控制器名称	作用
Deployment	声明式更新控制器，用于发布无状态应用
ReplicaSet	副本集控制器，用于对Pod进行副本规模 扩大或剪裁
StatefulSet	有状态副本集，用于发布有状态应用
DaemonSet	在k8s集群每一个Node上运行一个副本， 用于发布监控或日志收集类等应用
Job	运行一次性作业任务
CronJob	运行周期性作业任务

6.4Deployment控制器介绍

具有上线部署、滚动升级、创建副本、回滚到以前某一版本（成功/ 稳定）等功能。
Deployment包含ReplicaSet，除非需要自定义升级功能或者根本不需要升级Pod，否则还是建议使用Deployment而不直接使用ReplicaSet 。

6.5删除Deployment

```
kubectl delete -f tomcatdeployment.yml
```

7.service

7.1创建service

在idea工程resource/service/tomcatservice.yml

```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: tomcat-deploy
  labels:
    app: tomcat-deploy
spec:
  replicas: 1
  template:
    metadata:
      name: tomcat-deploy
      labels:
        app: tomcat-pod
    spec:
      containers:
        - name: tomcat-deploy
          image: tomcat:9.0.20-jre8-alpine
          imagePullPolicy: IfNotPresent
          ports:
            - containerPort: 8080
      restartPolicy: Always
  selector:
    matchLabels:
      app: tomcat-pod
---
apiVersion: v1
kind: Service
metadata:
  name: tomcat-svc
spec:
  selector:
    app: tomcat-pod
  ports:
    - port: 8888
      targetPort: 8080
      nodePort: 30088
      protocol: TCP
  type: NodePort
```

7.1.0service的selector

```
请各位小伙伴注意：
service.spec.selector.app选择的内容仍然是template.label.app内容。而不是我们deployment控制器的label内容
```

7.1.1Service类型

```
ClusterIP：默认，分配一个集群内部可以访问的虚拟IP
NodePort：在每个Node上分配一个端口作为外部访问入口
LoadBalancer：工作在特定的Cloud Provider上，例如Google Cloud，AWS，OpenStack
ExternalName：表示把集群外部的服务引入到集群内部中来，即实现了集群内部pod和集群外部的服务进行通信
```

7.1.2Service参数

```
port ：访问service使用的端口
targetPort ：Pod中容器端口
NodePort： 通过Node实现外网用户访问k8s集群内service(30000-32767)
```

7.2运行service

```
kubectl apply -f tomcatservice.yml
```

7.2删除service

```
kubectl delete -f tomcatservice.yml
```