# 测试环境及一键部署

## 使用JDOS构建所需容器

### 申请系统

系统是个大范围，可以对应一个部门也可以对应一个团队。**申请系统必须张珉审批。**

一旦申请成功，后面的JDOS上所有的工作将不再需要请求别人审批，系统管理员可以自批。

目前申请的系统名称是 airplane (如下图) 但是可以不仅仅只做这一个业务。

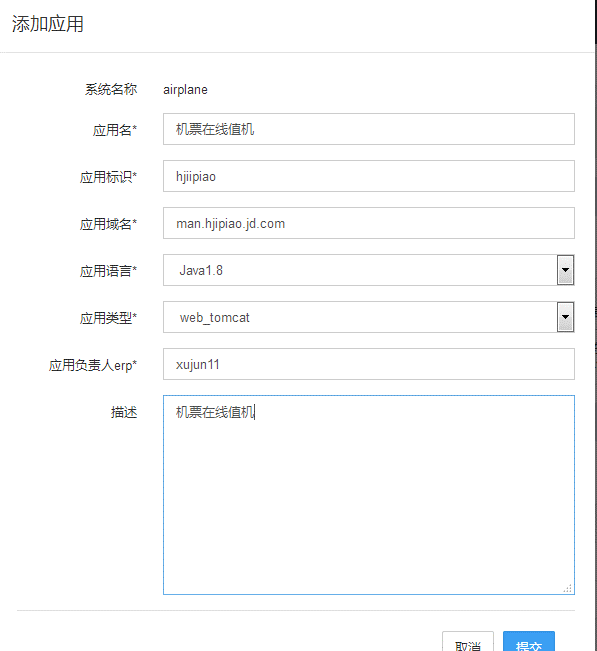


### 创建应用

一个应用对应一个POOL，应用由多个分组构成。分组就是容器的集群，里包含若干容器。

所谓集群就是一组容器提供相同的服务，一般也具有相同的配置。

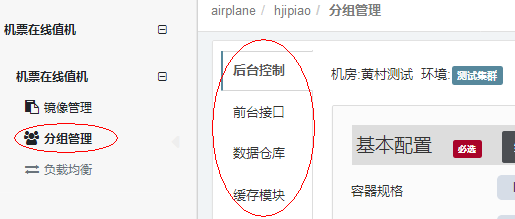
添加应用时注意 【应用标识】和【应用域名】不重复就行了。应用类型选择web\_tomcat，其他字段可以随便填写。【应用负责人erp】具有分组和容器的所有操作权限**包括审批**。



注意这里【应用语言】也可以随意选择。因为我们的容器只需要一个干净的centos镜像，jdk都不是必须的。但是JDOS目前没有提供其他的选项，所以每个容器里面都带一个jdk。

应用创建好后，只需要【分组管理】，【镜像管理】和【负载均衡】请都无视。

接下来我们要创建创建【后台控制】【前台接口】【数据仓库】【缓存模块】这几个分组。

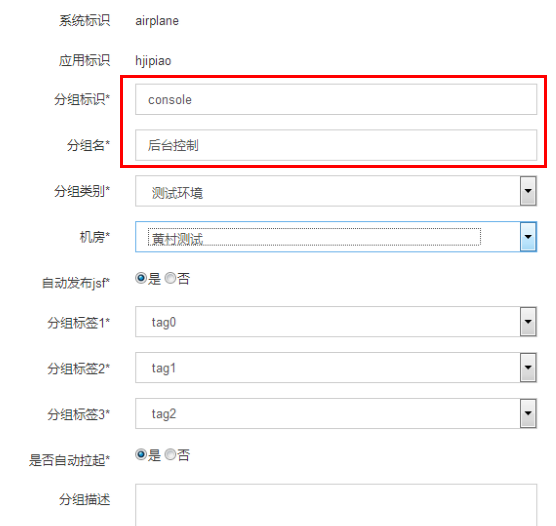


容器就是提供特定服务的虚拟主机，如MySQL数据库服务、WEB服务、缓存服务等。

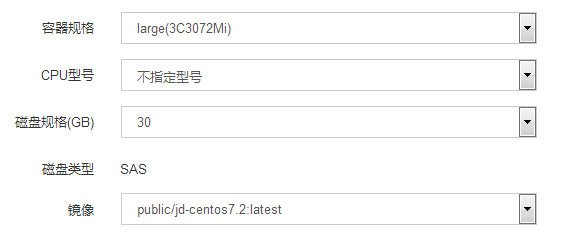
虽然可以在一台主机上通过不通端口来提供多种服务。但是**功能单一**的容器还是有诸多好处的。例如集群的扩容、批量配置、自动部署。

### WEB容器

web容器也就是基于tomcat的包括【前台接口】和【后台控制】两个

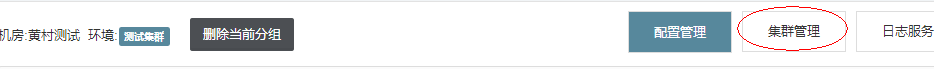


如图填写【分组标识】和【分组名】，其他字段默认即可。这里自动发布jsf可以选择【是】



【基本配置】中【容器规格】表示CPU和内存资源，镜像选择public/jd-centos7.2:latest

后面的高级配置不用管。然后点击【集群管理】tab页进行容器创建



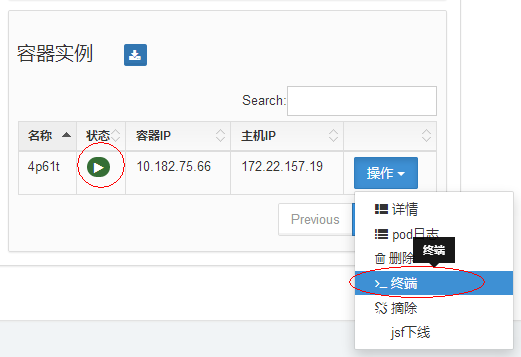
在集群页面点击【创建集群】，【容器个数】填写1，就提出了创建集群的申请。

**该申请需要应用管理员**



管理员到【个人中心】进行审批，审批通过后进行【集群上线】。

可以看到下方的【容器实例】，当状态是绿色就表示容器启动成功了。我们可以使用终端登录上去进行操作。

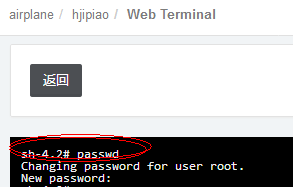


注意这里我们直接 ssh [root@10.182.75.66](mailto:root@10.182.75.66) 是不能登录到容器上的。

目前办公室的网段没有开通这个。文档上说**可以通过192.168.x.x网段跳转**。

正好目前我们有**airplane@192.168.170.45** 我们可以登录到这台机器上再ssh到容器。

如果操作不复杂也可以直接用这里的【web终端】，第一次必须用web终端因为没有设置root的密码。在web终端里使用 **passwd** 命令设置 root的密码（不设置也可以）。如下图。



目前我所有的密码都设置为“**lucky**”

### MySQL容器

MySQL容器我这里取名叫【数据仓库】，它和其他容器的区别是镜像选择 **public/mysql-5.5.38**



这里注意我们不能选择public/jd-centos7.2:latest 因为那样我们要安装mysql-server。但是目前**容器上不能访问外网服务**。包括centos的镜像设置京东自己的外网镜像。这样就有个问题，除了maven仓库外很多仓库都要访问外网。例如nodejs-npm、ruby-gem，python-pip。如果我们的应用需要访问这次东东，可以使用http代理。使用如下设置即可。

|  |
| --- |
| export http\_proxy=http://192.168.180.101:11128  export https\_proxy=http://192.168.180.101:11128 |

但是注意这只能暂时在会话中的设置，而**不宜保存在配置文件里**。用代理安装完软件后，需清除掉设置或者重新开启终端。将来jdos或许开放自制镜像的功能，我们就能上传所需的docker镜像，在容器里尽情的玩耍。

MySQL容器搭建好后接下来创建数据库，创建用户，给用户授权。然后就可以在本机用MySQL客户端来访问容器里的MySQL数据库了。

|  |
| --- |
| mysql  > create user ‘moon\_rw’@’%’ identified by ‘1YVaccCuHJAL.E:’  > create database airplane\_moon default charset utf8 collate utf8\_general\_ci;  > grant all privileges on airplane\_moon.\* to ‘moon\_rw’@’%’;  > flush privileges; |

**这里MySQL服务对外的端口默认为3358 而不是 3306**

这里注意下 MySQL 安装在 /export/servers/mysql 目录下但是可执行文件不在PATH里。所以我们可以调整下将以下命令写入 /etc/profile 文件中

|  |
| --- |
| export PATH=$PATH:/export/servers/mysql/bin |

然后再用命令使其生肖

|  |
| --- |
| source /etc/profile |

### JIM缓存容器

首先根据WEB容器的方式来创建容器。然后再根据文档来搭建。

参考 <http://jpcloud.jd.com/pages/viewpage.action?pageId=15342260>

注意在我们代码的filter-dev.properties里 jimdb.config.jimurl 要有如下配置

|  |
| --- |
| jimdb.config.jimurl=jim://2870414099454224775/3 |

这次容器上的hosts配置如下

|  |
| --- |
| # Kubernetes-managed hosts file.  127.0.0.1 localhost  ::1 localhost ip6-localhost ip6-loopback  fe00::0 ip6-localnet  fe00::0 ip6-mcastprefix  fe00::1 ip6-allnodes  fe00::2 ip6-allrouters  10.182.75.69 hjipiao-e96f1619-2468888727-5jb20  **192.168.150.61 cfs.jim.jd.local**  **10.182.75.69 redis.local** |

## 修改dev配置使其指向新环境

上述容器搭建完成后我们需要修改代码让依赖的资源指向容器。

filter-dev.properties 文件修改

|  |
| --- |
| airplane.moon.mysql.jdbc.url=\  jdbc:mysql://10.182.75.68:3358/airplane\_moon?autoReconnect=true  jimdb.config.jimurl=jim://2870414099454224775/3 |

注意自己本地的hosts文件也要修改redis.local

|  |
| --- |
| **10.182.75.69 redis.local**  192.168.150.61 cfs.cache.jd.local  192.168.150.61 cfs.jim.jd.local  192.168.150.61 adm.cache.360buy.com  192.168.150.151 erp1.360buy.com  192.168.150.121 i.jsf.jd.com  10.190.161.237 zk.m.jd.com  192.168.151.143 g.jsf.jd.test  **192.168.104.121 autodeploy.jd.com**  #127.0.0.1 man.hjipiao.jd.com  **10.182.75.66 man.hjipiao.jd.com**  **10.182.75.67 hjipiao.jd.com** |

这样我们在部署完成后就可以访问man.hjipiao.jd.com

## 使用autodeploy一键部署

### 产品运维

我们没有使用JDOS提供的镜像打包war。而只是用了centos基础容器

WEB容器中部署前后台应用我们使用 <http://autodeploy.jd.com>

首先选择【产品运维】> 【部署测试环境】进行配置



如图进入如下页面。【部门和产品线】一个POOL就是一个产品线（包含接口和后台）。例如我建立了airplane 这个产品线。**但是我们没有权限将ERP帐号跟产品线绑定起来。**



**所以接下来大家需要JDme上找 huwenping（胡文萍） 申请得到操作 产品线的权限。**



然后进入【服务器IP和域名】录入我们搭建的WEB容器的IP。这里有两个分组。

所以我录入了10.182.75.66 (后台控制) 以及 10.182.75.67 (前台接口)



然后进入【产品线和域名】进行配置，如下图。这里git仓库要选择私有库。输入自己的erp帐号密码进行拉取。【war包目录名】选择war包所在子项目的target子目录，相对项目根路径。例如本例中填写 **airplane-moon-man-web/target**



设置完成后如下所示

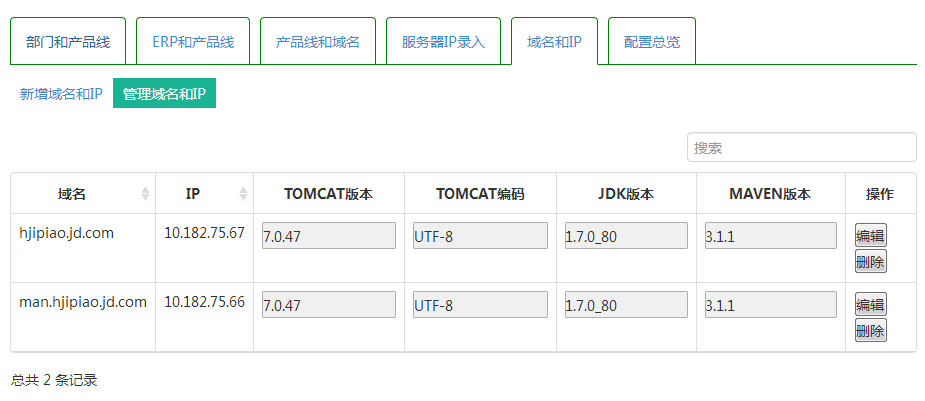


接下来设置【域名和IP】即配置域名和IP的关系，如下所示



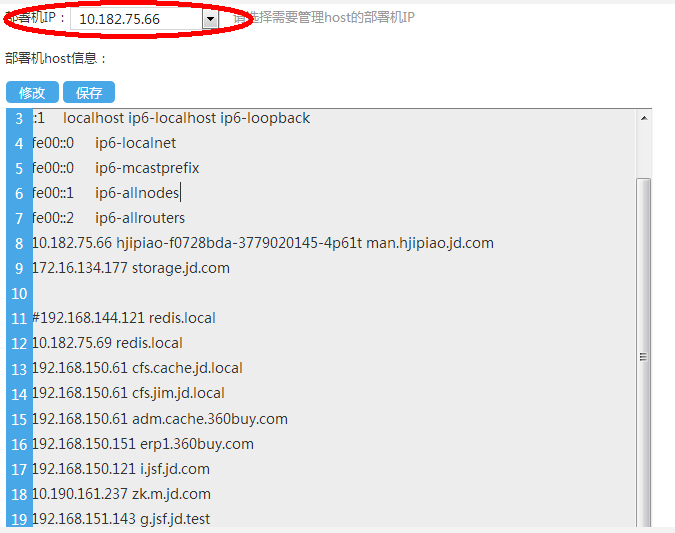
这里tomcat 我们选择 tomcat7，jdk选择jdk1.7.x，maven选择3.x.x

配置结果如下图所示



接下来我们还要去配置WEB容器上的hosts文件。autodeploy也提供了界面来做。

点击【产品运维】的【Host管理】进入容器的Hosts的配置



可以分别对两个WEB容器进行配置，为两者添加相同的配置。

这里主要是账户、zk、jimdb和jsf的配置。注意redis.local指向容器中的缓存模块

到这里为止我们已经完成了autodeploy的配置工作接下来我们进行一键部署

### 一键部署



点击【一键部署】进入操作画面，然后按照【步骤】中的说明在web终端中操作容器。

执行步骤中的1、2、3步。注意这里的容器是指WEB容器，MySQL和JIMDB的容器不管。

|  |
| --- |
| 1. root账号登录测试服务器  2. 执行 echo 192.168.104.121 autodeploy.jd.com >> /etc/hosts  3. JDOS上申请的容器执行 source /dev/stdin <<< "$(curl http://autodeploy.jd.com/resource/shell/install\_software\_jdos.sh)"  4. 由管理员登录autodeploy.jd.com增加站点、域名等配置即可进行部署 |

接下来就可以创建发布任务了。如下图添加所需的应用，注意这里勾选强制打包。

我目前是加了个分支 xujun-autodeploy-jdos 来指向配置的容器。这里发布的分支名称只能是**字母数字和下划线中划线**，所以xujun/autodeploy-jdos就不能成功发布。



勾选列表中任务，点击【批量部署】，直到出现【部署成功】后，恭喜你你完成了自动发布。

## 后记

JDOS环境是用来管理轻量级容器Docker的平台，这个平台给了我们足够的计算资源。这些计算资源除了用来搭建项目的测试环境外，还可以利用来学习一些新的框架、开源项目，特别是基于网络的应用，分布式应用，模拟集群压力测试。例如可以自己搭建一个小型的zookeeper集群，hadoop计算，spark实时，elasticsearch搜索，tensorflow人工智能研究。（恰好它自带的镜像就有tensorflow的）