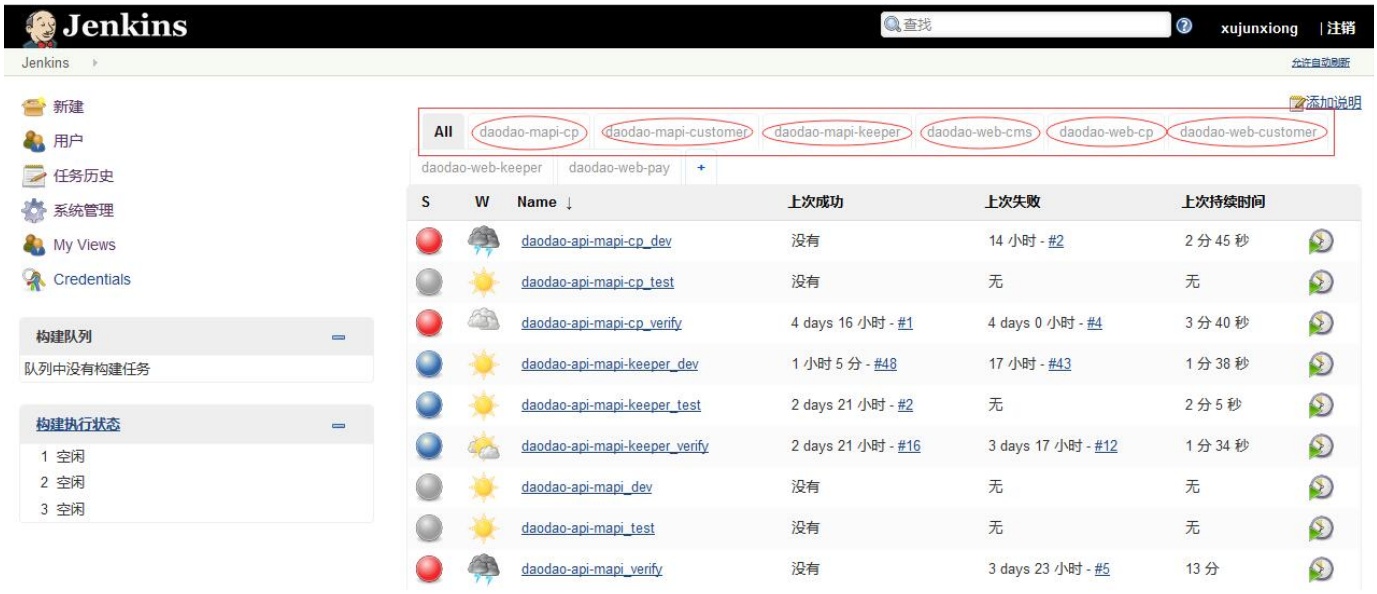
jenkins的正确打开方式

登陆jenkins

访问 jenkins.dongdaodao.com，输入用户密码

项目总览



每个项目视图都会有三个project，分别是dev、test、verify环境的project，这样命令是为了区分环境，workspace的命名就是以project名称来定

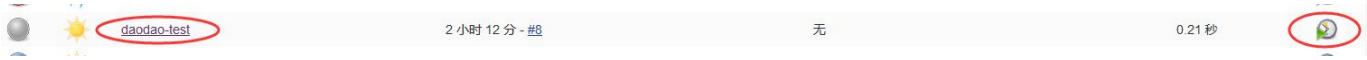
义的

配置项目（无特殊情况请勿擅自修改配置）

如何构建项目？

首先打开需要构建的项目，然后输入构建参数进行构建即可

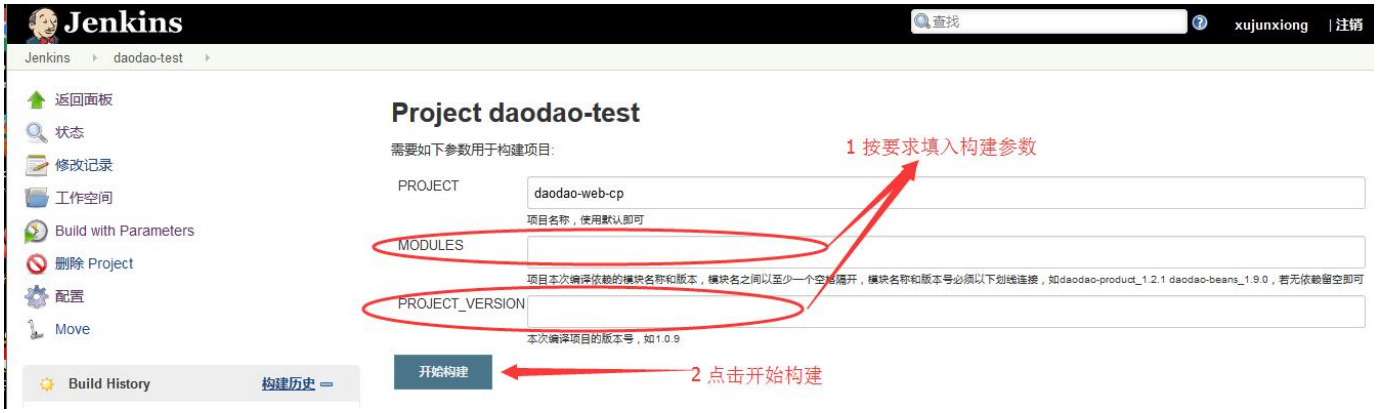
1）选择构建项目，点击项目名称 或 点击项目最右边的构建按钮（如下图红框所示）进行参数构建



2）若点击项目名称，则如下图所示，需要再次点击如下图中的Build with Parameters 按钮输入参数

3）若直接点击构建按钮则直接显示构建参数输入框，按照文字提示补充构建参数，点击构建即可开始项目构建

3

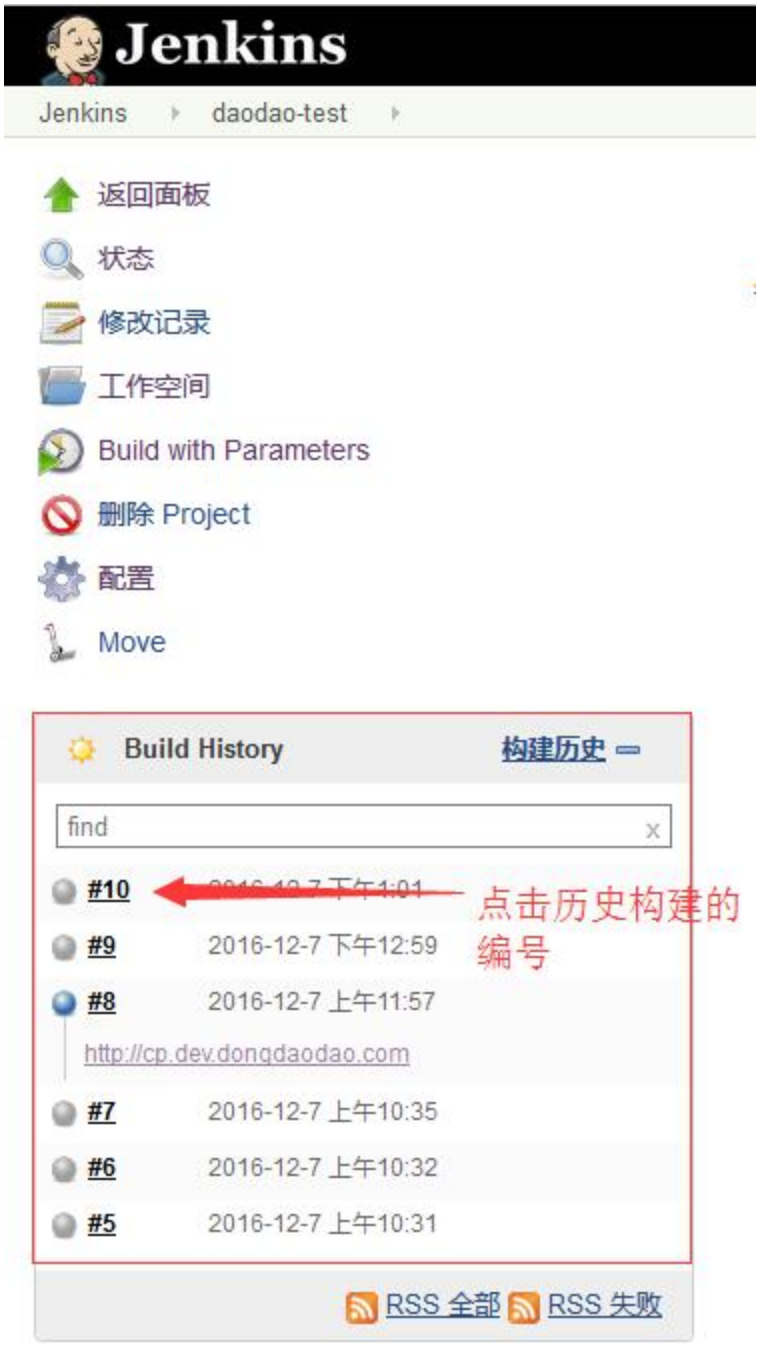


添加参数后效果如图所示

提醒：这种参数化的构建方式每次构建勾需要手动输入参数，如果参数不变需要重复构建，我们可以复制上一次的构建的参数来的粘贴到本次构建中使用

查看历史构建输入的参数？

如下图两种方式打开都能获取历史构建参数


Jenkins
Jenkins
daodao-test
返回面板
状态
修改记录
工作空间
Build with Parameters
删除 Project
配置
Move
Build History
构建历史
find
#10
点击历史构建的编号
#9 2016-12-7 下午12:59
#8 2016-12-7 上午11:57
http://cp.dev.dongdaodao.com
#7 2016-12-7 上午10:35
#6 2016-12-7 上午10:32
#5 2016-12-7 上午10:31
RSS 全部
RSS 失败


Build History
构建历史
点击历史构建编号的三角符号
find
#10
2016-12-7 下午1:01
变更记录
Console Output
编辑编译信息
删除本次生成
Parameters
点击左边的按钮
Parameters
RSS 失败


Jenkins
Jenkins
daodao-test
#10
Parameters
返回到工程
状态集
变更记录
Console Output
编辑编译信息
删除本次生成
Parameters
Parameters
点击框内按钮查看构建参数
前一次构建

如何查看构建日志？

编译的时候想看日志，可以点击项目名称编号的的三角符号，然后点击console output按钮查看构建日志

jenkins编译部署脚步：大家大概知道脚本的逻辑即可，脚本再做了较多的条件判断，是为了节省编译时间，跳过如本地代码已经是最新且已经编译过的模块，所以各位尽可能不要清理工作空间，特殊情况除外

```
#!/bin/bash
# 读取环境变量
. /etc/profile
# 读取jenkins脚本配置变量
. /tmp/.project_deploy_var
```

```
# 在读取完后删除文件，这样避免阻塞其他项目的构建，也避免变量冲突
rm -fr /tmp/.project_deploy_var
line="================================================================================"

## 克隆项目函数
clone_project() {
    cd $project_path
    if ! [ -a $project_path/$i ]; then
        echo  -e "[INFO]\n[INFO] $line\n[INFO] Git clone '$i' ...\n[INFO] $line"
        if ! git clone git@192.168.3.173:server/$i.git &> /dev/null; then
            echo  -e "[ERROR]\n[ERROR] $line\n[ERROR] Git clone '$i' FAILED\n[ERROR] $line"
            exit 2
        fi
    fi
}

## git拉取代码和maven编译函数
gitPull_maven_codes() {
echo  -e "[INFO]\n[INFO] $line\n[INFO] Git pull '$i' on branch $branch \n[INFO] $line"
cd $project_path/$i
if git fetch &> /dev/null && git checkout $branch &> /tmp/gitpull.log && git pull &> /tmp/gitpull.log; then
    cat /tmp/gitpull.log
 # 若当前编译的模块是项目本身，则无论是否拉取到新代码都需要重新编译一次
 if [[ $i == $project ]]; then
        echo -e "[INFO]\n[INFO] $line\n[INFO] Project '$i' maven start\n[INFO] $line"
        if maven_project; then
            echo -e "[INFO] $line\n[INFO] Project '$i' maven SUCCESS\n[INFO] $line"
        else
            echo -e "[ERROR] $line\n[ERROR] Project '$i' maven FAILED\n[ERROR] $line"
            exit 2
        fi
    else
  #
判断拉代码的结果，如果模块代码本地已经是最新，判断是否已经编译过（对应jar包是否存在），没有jar包则需要编译，有则跳过编译
  if grep -q "Already up-to-date" /tmp/gitpull.log; then
            if [ -a $project_path/$i/target/$i-*-SNAPSHOT.jar ] || ls $project_path/$i/*/target/*.jar &> /dev/null;
then
                echo -e "[INFO]\n[INFO] $line\n[INFO] Project '$i' already up-to-date and already biuld\n[INFO]
$line"
            else
                echo -e "[INFO]\n[INFO] $line\n[INFO] Project '$i' maven start\n[INFO] $line"
                if maven_project; then
                    echo -e "[INFO] $line\n[INFO] Project '$i' maven SUCCESS\n[INFO] $line"
                else
                    echo -e "[ERROR] $line\n[ERROR] Project '$i' maven FAILED\n[ERROR] $line"
                    exit 2
                fi
            fi
  # 判断拉代码的结果，若拉取到新的代码，则执行编译操作
   elif grep -q "changed" /tmp/gitpull.log; then
            echo -e "[INFO]\n[INFO] $line\n[INFO] Project '$i' maven start\n[INFO] $line"
            if maven_project; then
                echo -e "[INFO] $line\n[INFO] Project '$i' maven SUCCESS\n[INFO] $line"
            else
                echo -e "[ERROR] $line\n[ERROR] Project '$i' maven FAILED\n[ERROR] $line"
                exit 2
            fi
```

```
        fi
    fi
else
    echo -e "[ERROR] $line\n[ERROR] Git pull '$i' FAILED\n[ERROR] $line"
    cat /tmp/gitpull.log
    exit 2
fi
}

## maven 构建命令函数
maven_project() {
    cd $project_path/$i
    $mvn_sh
}

# 调用克隆函数和编译函数代码进行项目构建
for i in $modules "$project"_"$version"; do
    branch=develop-`echo $i | awk -F"_" '{print $2}'`
    i=`echo $i | awk -F"_" '{print $1}'`
    clone_project
    gitPull_maven_codes
done

# 创建远程服务器部署脚本函数，连接远程服务器创建脚本以调用
create_script() {
ssh $server_host 'cat << END > /tmp/.'$project'_deploy.sh
#!/bin/bash
project='$project'
# 读取环境变量
. /etc/profile
# 读取打印tomcat日志命令函数
. /usr/local/.prinlog

# 请求项目的测试页面判断是否部署成功
check_url() {
declare -i n=1
sleep 30
until [ \$n -gt 8 ]; do
        if curl --head '$domain'/test.jsp | grep "200 OK" ; then
            return 0
        else
            let n++
            sleep 5
            [ \$n -eq 8 ] && return 6 # test page request failed
        fi
done
}
# 备份项目旧代码
if [ -a /deploy/'$project' ]; then
    if ! mv /deploy/'$project' /deploy/bak/'$project'_\`date +%F-%T\`; then
        echo "[ERROR] --- Backup '$project' failed"
 fi
fi
# 解压新代码
if tar xf /tmp/'$project'.tar -C /deploy; then
   echo "[INFO] --- Create test page for '$project'"
   if echo "test page" > /deploy/'$project'/test.jsp; then
        echo "[INFO] --- Restart '$project' tomcat service"
        # 设定日志启动时间以在启动失败时打印日志
```

```
  logdtime=\`date +%H:%M\`
     logdate=\`date +%d-%b-%Y\`
     service '$project' restart &> /dev/null
  fi
  # tomcat启动后请求测试页面
 if check_url; then
     exit 0
   else
      echo "[ERROR] --- Test page request failed, print tomcat log ---"
       print_log
       exit 2
   fi
else
 echo "[ERROR] --- Unpack '$project' failed"
  exit 2
fi
END'
}

## 测试和开发环境调用的代码部署脚本，将代码scp到对应环境服务器，调用远程服务器的部署脚本
test_dev_script() {
cd $project_path/$project/target
echo  -e "[INFO]\n[INFO] $line\n[INFO] Project '$project' deploy start \n[INFO] $line"
echo "[INFO] --- Target server: $server_host, domain: $domain ---"
echo "[INFO] --- Packing project '$project' ..."
# 归档新构建的代码目录
if tar cf $project.tar $project; then
       echo "[INFO] --- Copying project '$project.tar' to remote host $server_host ..."
  # 将代码复制到远程服务器
   if scp $project.tar $server_host:/tmp; then
              echo "[INFO] --- Creating deploy script on remote host $server_host ..."
              # 连接远程服务器创建部署脚本
    if create_script; then
                    echo "[INFO] --- Executing deploy script on remote host $server_host ..."
      # 远程执行部署脚本
     if ssh $server_host bash -x /tmp/.$project"_"deploy.sh; then
                echo  -e "[INFO]\n[INFO] $line\n[INFO] Deploy '$project' on host $server_host SUCCESS \n[INFO]
$line"
                    else
                echo  -e "[EERROR]\n[ERROR] $line\n[ERROR] Deploy '$project' on host $server_host FAILED
\n[ERROR] $line"
                exit 2
                    fi
              else
                    echo "[ERROR] --- Create script on host $server_host failed"
                    exit 2
              fi
       else
              echo "[ERROR] --- Copy project '$project.tar' to host $server_host  failed"
              exit 2
       fi
else
       echo "[ERROR] --- Pack project '$project' failed"
       exit 2
fi
}

# 环境域名匹配，部署后请求测试页面使用
roles="$project"_"$env"
```

```
case $roles in
    daodao-api-mapi_dev )
        domain=mapi.dev.dongdaodao.com
        ;;
    daodao-api-mapi_test )
        domain=mapi.test.dongdaodao.com
        ;;
    daodao-api-mapi-keeper_dev )
        domain=mapi-k.dev.dongdaodao.com
        ;;
    daodao-api-mapi-keeper_test )
        domain=mapi-k.test.dongdaodao.com
        ;;
    daodao-api-mapi-cp_dev )
       domain=mapi-cp.dev.dongdaodao.com
        ;;
    daodao-api-mapi-cp_test )
       domain=mapi-cp.test.dongdaodao.com
        ;;
    daodao-web-customer_dev )
        domain=c.dev.dongdaodao.com
        ;;
    daodao-web-customer_test )
        domain=c.test.dongdaodao.com
        ;;
    daodao-web-keeper_dev )
        domain=k.dev.dongdaodao.com
        ;;
    daodao-web-keeper_test )
        domain=k.test.dongdaodao.com
        ;;
    daodao-web-cp_dev )
        domain=cp.dev.dongdaodao.com
        ;;
    daodao-web-cp_test )
        domain=cp.test.dongdaodao.com
        ;;
    daodao-web-cms_dev )
        domain=cms.dev.dongdaodao.com
        ;;
    daodao-web-cms_test )
        domain=cms.test.dongdaodao.com
        ;;
    daodao-web-official_dev )
        domain=official.dev.dongdaodao.com
        ;;
    daodao-web-official_test )
        domain=official.test.dongdaodao.com
        ;;
    daodao-web-pay_dev )
        domain="127.0.0.1:8083"
    ;;
    daodao-web-pay_test )
    domain="127.0.0.1:8083"
        ;;
esac

# 调用部署脚本函数
if [ $? -eq 0 ]; then
```

```
    test_dev_script
fi
```