亿点汇通CI操作手册

北京亿点汇通网络科技有限公司

编写目的：为新人更快的融入亿点汇通集体开发环境

编写日期：2018年9月13日

2018

[一、 说明 1](#_Toc525749312)

[二、 注册 1](#_Toc525749313)

[三、 登录 1](#_Toc525749314)

[四、 Jenkins工作界面 1](#_Toc525749315)

[4.1导航 2](#_Toc525749316)

[4.2菜单栏 3](#_Toc525749317)

[4.3构建执行状态 4](#_Toc525749318)

[4.4视图列表 5](#_Toc525749319)

[4.4.1编译宜店项目 5](#_Toc525749320)

[4.4.2部署宜店项目 6](#_Toc525749321)

[4.4.3拉分支 7](#_Toc525749322)

[4.5搜索栏 7](#_Toc525749323)

[五、 开发中CI的应用 8](#_Toc525749324)

[5.1功能分支 9](#_Toc525749325)

[5.1.1何时创建功能分支 9](#_Toc525749326)

[5.1.2创建功能分支过程 9](#_Toc525749327)

[5.1.3功能分支使用完后的操作 10](#_Toc525749328)

[5.2 release分支 10](#_Toc525749329)

[5.2.1何时创建release分支 10](#_Toc525749330)

[5.2.2创建release分支过程 10](#_Toc525749331)

[5.2.3release分支使用完后的操作 13](#_Toc525749332)

[5.3改bug分支 14](#_Toc525749333)

[5.3.1何时创建Hotfixes分支 14](#_Toc525749334)

[5.3.2创建Hotfixes分支过程 14](#_Toc525749335)

[5.3.3Hotfixes分支使用完后的操作 16](#_Toc525749336)

[5.4 打Tag 16](#_Toc525749337)

[5.4.1何时打Tag 16](#_Toc525749338)

[5.4.2创建Tag过程 16](#_Toc525749339)

[5.4.3为何要打Tag 18](#_Toc525749340)

[5.4.4如何使用Tag切换版本 18](#_Toc525749341)

[5.5测试代码覆盖率 19](#_Toc525749342)

[5.5.1何时使用Test-Coverage 19](#_Toc525749343)

[5.5.2如何部署Test-Coverage 19](#_Toc525749344)

[5.5.3如何使用Test-Coverage 20](#_Toc525749345)

[六、 分支合并 21](#_Toc525749346)

[6.1何时需要合并分支 21](#_Toc525749347)

[6.2合并分支的过程 21](#_Toc525749348)

[6.3合并分支时冲突的解决办法 22](#_Toc525749349)

[七、 发布版本 24](#_Toc525749350)

[7.1何时发布新版本 24](#_Toc525749351)

[7.2发布版本的操作流程 24](#_Toc525749352)

[7.3部署失败如何处理 25](#_Toc525749353)

# 说明

git是什么？

版本控制（Version control）:顾名思义，版本控制系统是任何能让你了解到一个文件的历史，以及它的发展过程的系统。

Jenkins是什么？

Jenkins 是一个可扩展的持续集成引擎。

通常我们称它叫做CI（Continuous integration）。

# 注册

**Hint：**亿点汇通的所有账号注册均由运维来完成注册和分配，如无账号可去询问运维人员。

# 登录

Gogs目前登录地址为：<http://git.188yd.com:8000/user/login>

Jenkins目前登录地址为：<http://192.168.1.223:8080/login?from=%2F>

如果地址变更，请找运维人员索要新的登录地址。

使用登录地址，直接登录。

# Jenkins工作界面

登录进入后可以看到一个Jenkins的工作界面，如下图7.1所示：

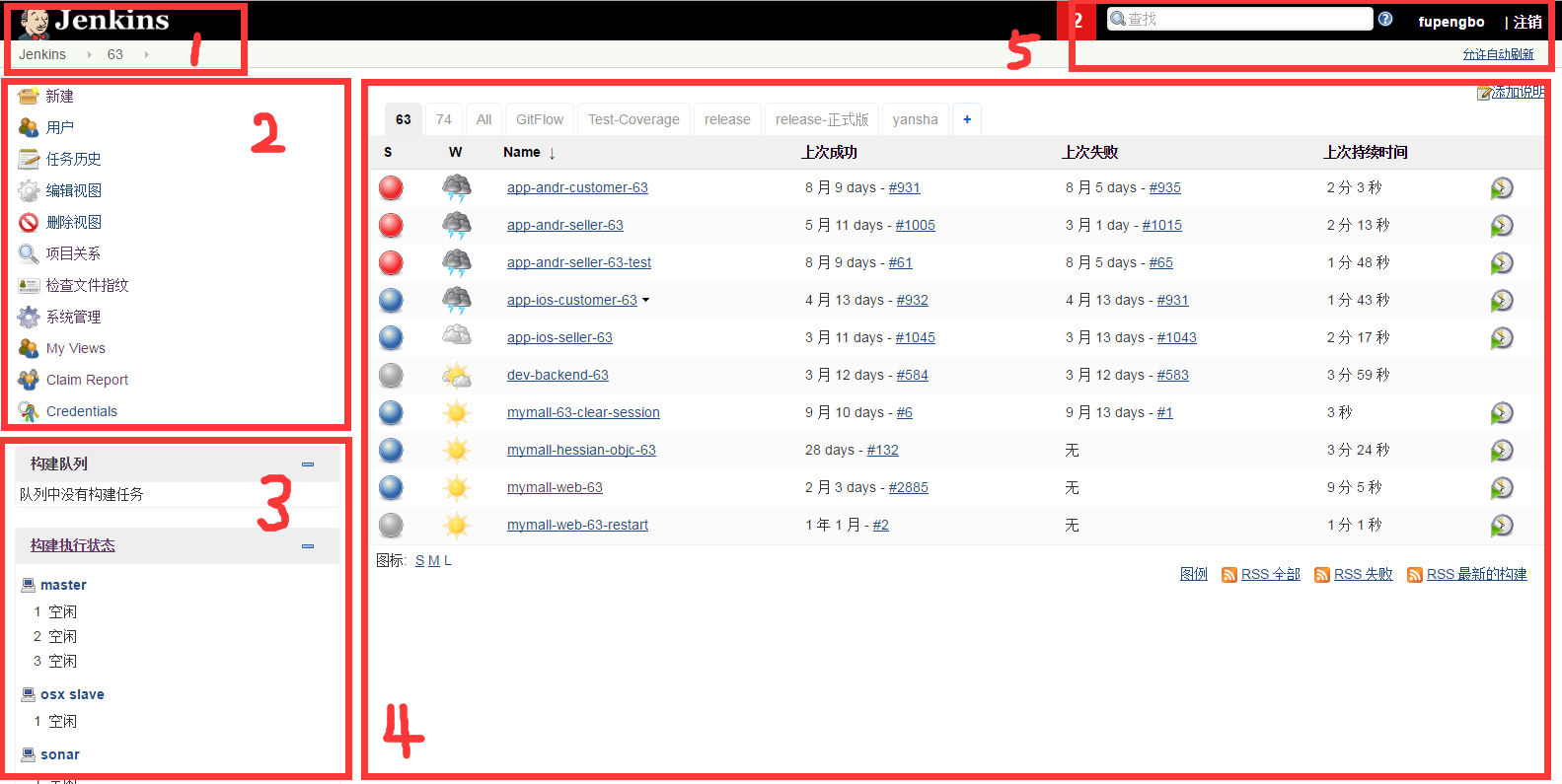


图4.1

Jenkins的工作界面大致可以分为5个区域，分别是：

## 4.1导航

第一个区域：

4.1.1点击字样大的Jenkins是统一返回键。

4.1.2鼠标悬浮在Jenkins的小图标上点击小图标浏览菜单栏：



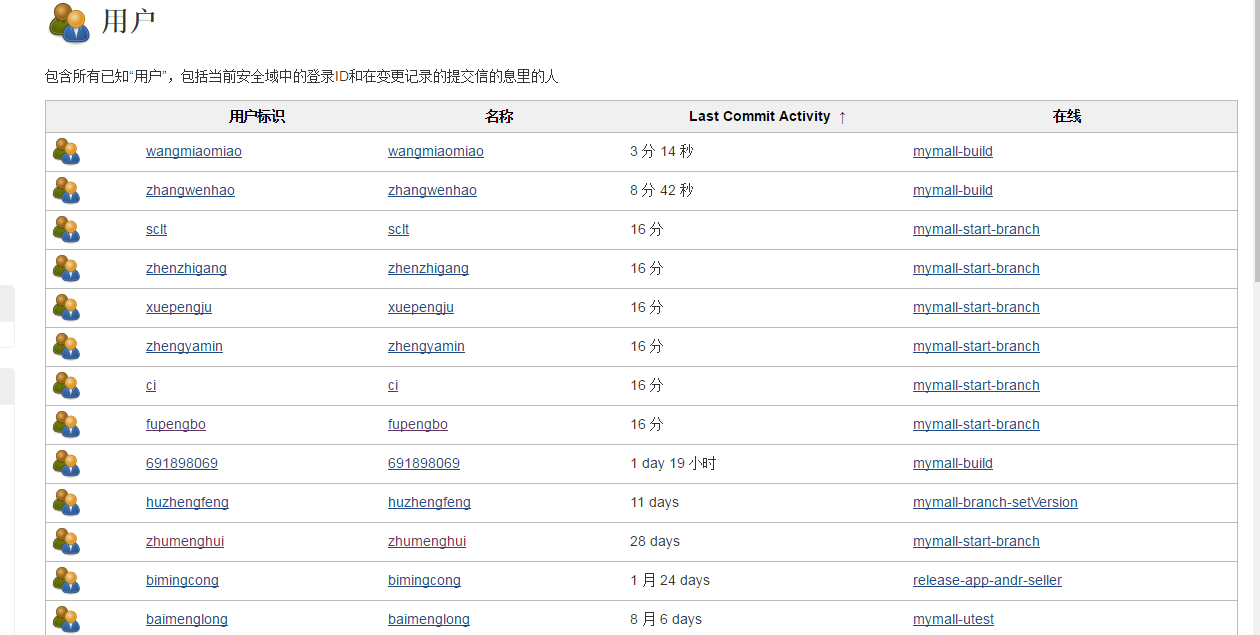
这个功能和第二个区域的功能是一样的，在第二个区域详细介绍。

## 4.2菜单栏

第二个区域：

如图4.1所示，菜单栏包括：

* + - * + 新建：新建一个Job，新入职暂时用不到，略。
        + 用户：点击用户可以看到当前所有用户及其状态等一系列信息。



* + - * + 任务历史：

可以看到所有job的工作记录



* + - * + 项目关系、检查文件指纹、Claim Report、 Credentials

略。

* + - * + 系统管理

一些系统配置和插件管理以及系统信息，负载统计在这里可以配值。

## 4.3构建执行状态

在这个区域可以查看当前Job的执行状态。



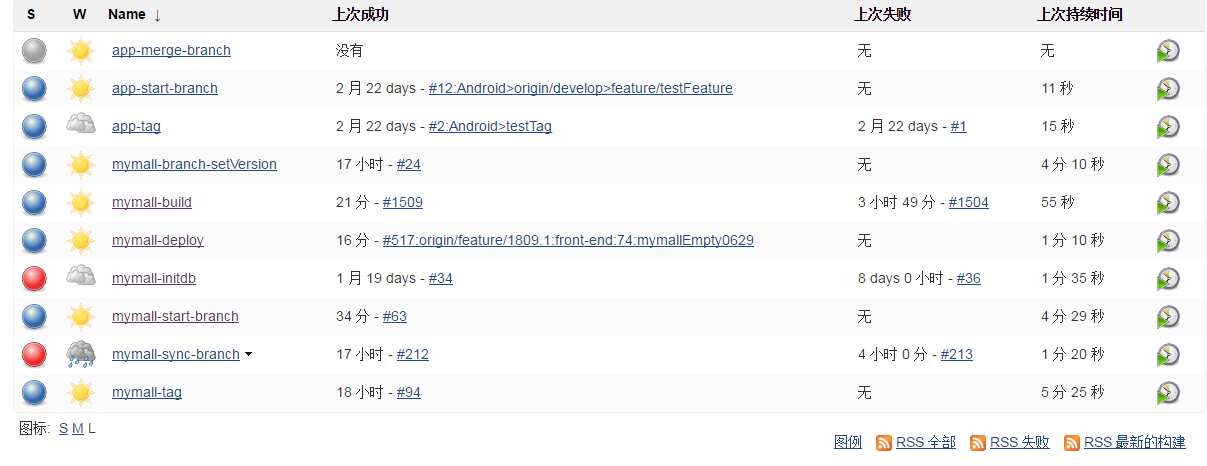
## 4.4视图列表

图4.1中的第四个区域，这个是我们工作中常用的到区域。

这个区域的顶部是各个视图的名字，我们的宜店项目在GitFlow视图中。



点击视图名字，可以看到该视图底下的job列表



其中s代表工作状态

W代表描述

Name是job的名字，可以通过点击名字进入该job中执行操作。

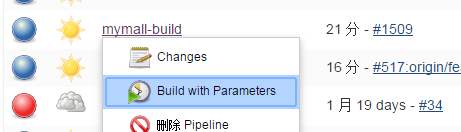
后面列是否成功和持续时间。

### 4.4.1编译宜店项目

当我们提交完代码后，需要部署时，则需要先在jenkens上编译项目。

就目前的job而言，编译项目需要如下操作：

* 1. 鼠标悬浮在mymall-build上面，点击配置并构建

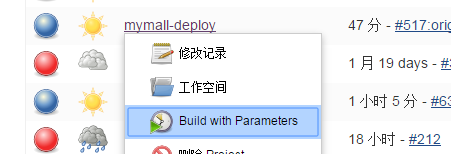


* 1. 选择分支和模块名称以及其他选项，选择后点击开始编译。



### 4.4.2部署宜店项目

1. 编译完后，鼠标悬浮在mymall-deploy上，点击配置并构建

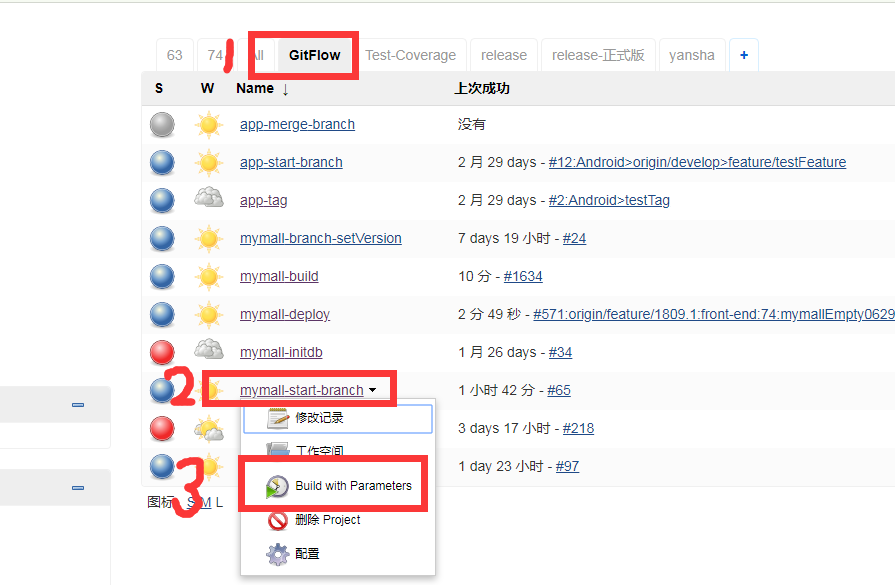


1. 选择分支和部署环境以及其他选项，选择后点击开始编译。



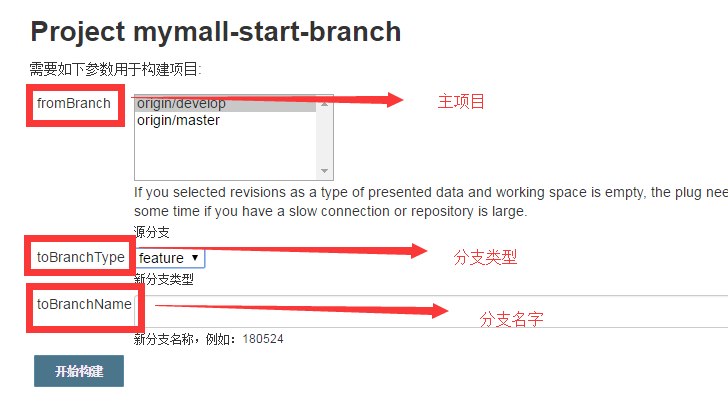
### 4.4.3拉分支

当我们工作中需要使用新的分支时，需要在jenkins上拉取分支，步骤如下：



1. 进入jenkens首页面点击GitFlow视图
2. 鼠标悬浮到mymall-start-branch上点击向下的三角标
3. 点击Build with Parameters

选择从那个项目中拉取分支以及分支类型和分支名字，一般来说分支名字依照开发功能的特点或时间命名。



## 4.5搜索栏

通过搜索栏我们可以快速查找到自己想要的job。



# 开发中CI的应用

图5.1很重要：

Branching model and Release management 
master 
hotfixes 
release 
branches 
develop 
feature 
branches 
Tag 
0.3 
Severe Eug 
fixed for 
production: 
hotfu 0.3.1 
Inco?porate 
bugfix iri 
develop 
Major for 
next 
Time 
Tao 
0.3.1 
Oily 
bugfixestt' 
1.0 
Bugfvxes from release branch rnay be 
merged back into 
develop 
Start oftelease 
branch 
2.0 
Start Of release branch for 
1.0 
this point 
on. next release: 
2.0 
tea 

图5.1

**Master 主干：保持一个稳定的版本，只用来发版和打Tag，不做开发使用。**

**Develop 开发分支：不稳定的版本，主要用来开发使用，开发完成后向Master合并。**

## 5.1功能分支

### 5.1.1何时创建功能分支

当开发在redmine上接到产品指过来的开发新功能任务时，则需要在Develop上拉取一个Feature分支搭建开发环境用于开发新功能使用。

### 5.1.2创建功能分支过程

创建功能分支需要进行以下步骤：

从Develop中拉取一个分支，如图5.1.1所示：

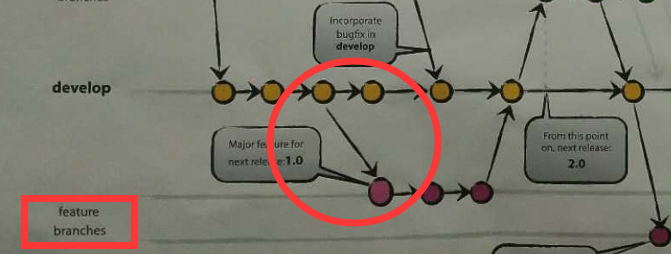


图5.1.1

拉取功能分支在Jenkens上的操作：

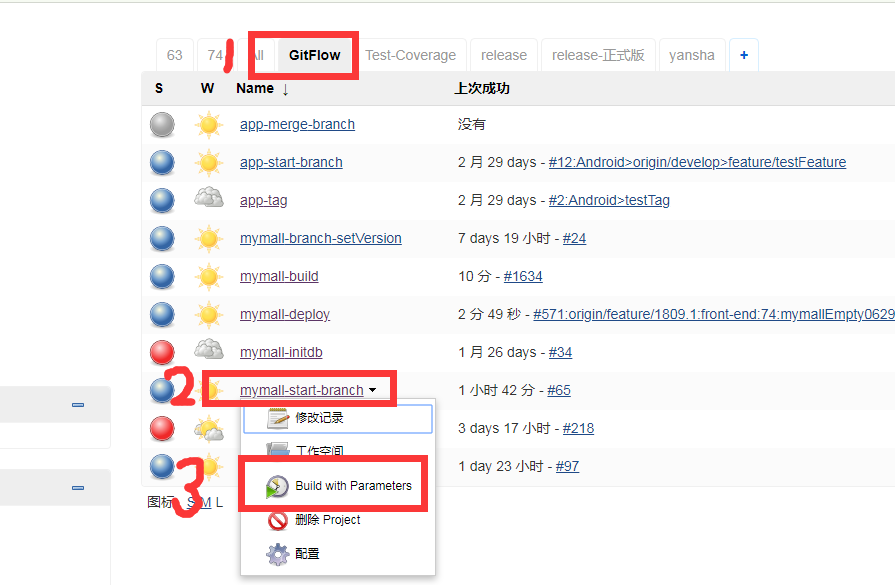
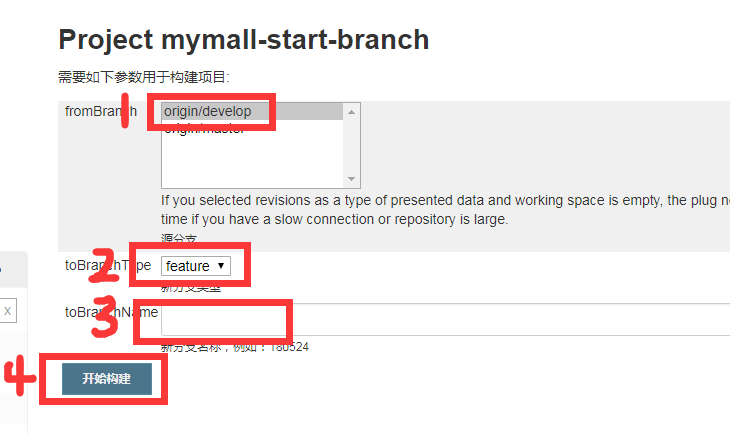


图5.1.2

1. 进入jenkens首页面点击GitFlow视图
2. 鼠标悬浮到mymall-start-branch上,点击三角标
3. 点击Build with Parameters

上步骤操作完之后会跳出以下页面 图5.1.3

图5.1.3

1. 选择develop，为什么选择develop，参考图5.1.3
2. 类型选择feature
3. 分支名字一般按时间或开发功能名字（英文）命名，见名知意即可。
4. 点击开始构建

当以上步骤操作完并构建成功后，打开你的idea，选择已拉取好的分支并检出。

### 5.1.3功能分支使用完后的操作

当我们完成功能开发并自测通过，将目前开发的feature分支合并到develop中，之后这个feature分支就被放弃掉。

## 5.2 release分支

### 5.2.1何时创建release分支

当开发完新功能之后，将从develop中拉取一个release分支用于搭建测试环境，供测试人员测试。测试期间提出的bug在release分支上修改。

### 5.2.2创建release分支过程

从Develop中拉取一个分支，如图5.2.1所示：

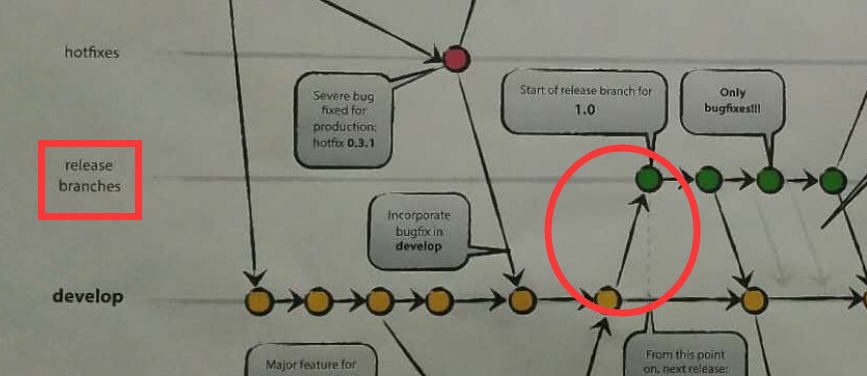


图5.2.1

拉取分支在Jenkens上的操作：

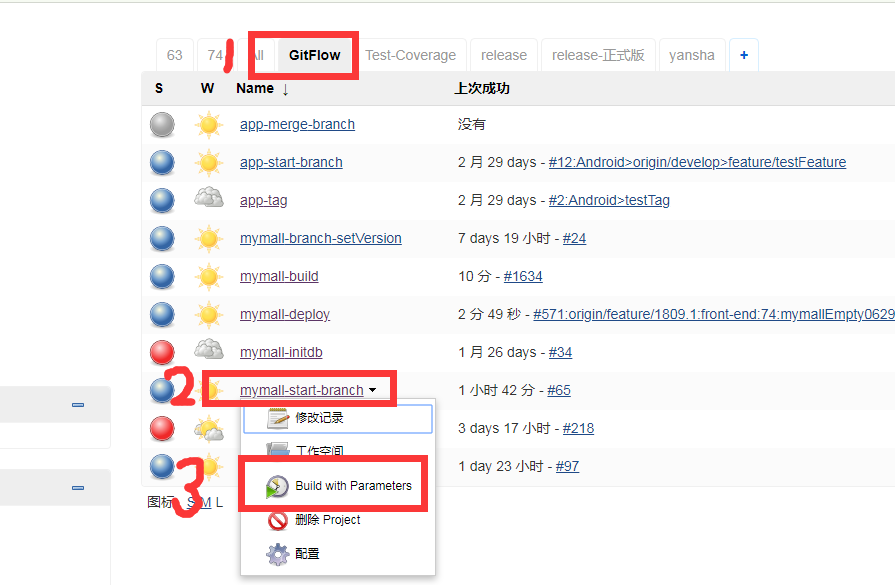


图5.2.2

1.进入jenkens首页面点击GitFlow视图

2.鼠标悬浮到mymall-start-branch上点击向下的三角标

3.点击Build with Parameters

上步骤操作完之后会跳出以下页面 如图5.2.3

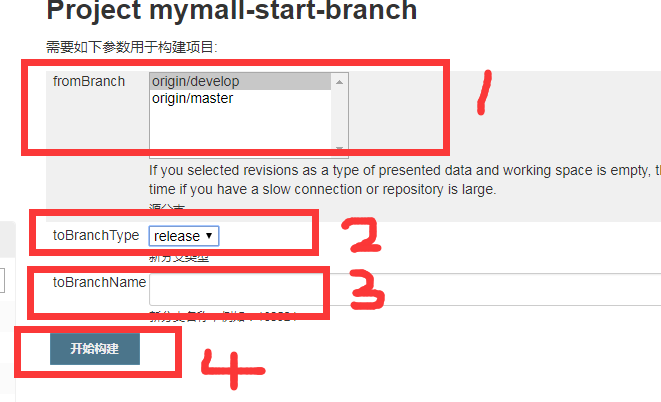


图5.2.3

1. 选择develop
2. 类型选择release
3. 名字按测试功能名字命名（英文）或是时间戳
4. 点击构建

完成上步骤构建好分支后，编译并部署项目，按如下步骤操作：

编译项目

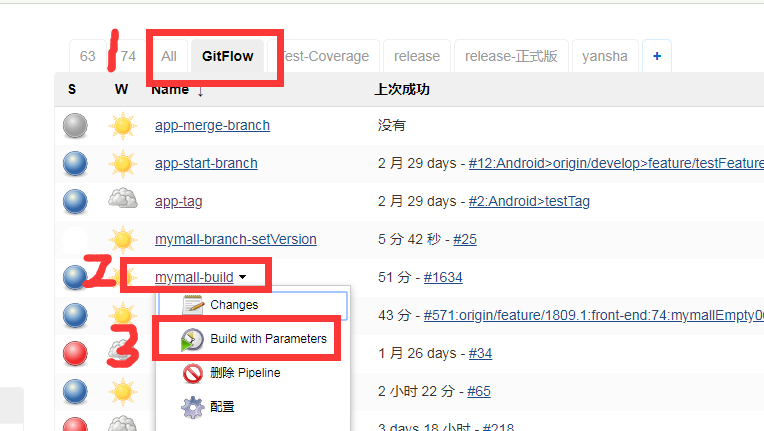


图5.2.4

1. 选择GitFlow视图
2. 选择mymall-build，鼠标悬浮在上面点击小三角
3. 点击build with Parameters

操作完上面步骤之后会出现如下视图，按步骤操作：

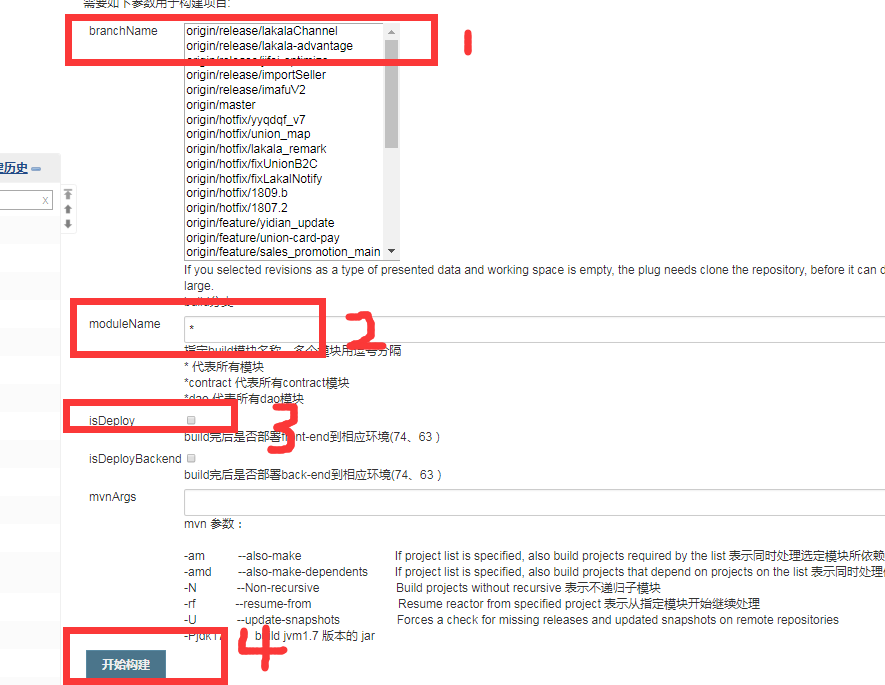


图5.2.5

1. 选择需要编译的分支
2. 选择模块编译，可快速完成编译，但一般不用
3. 编译完成部署项目
4. 点击开始构建

部署项目，如果第二步没有点击编译并部署，则在这一步进行部署操作：

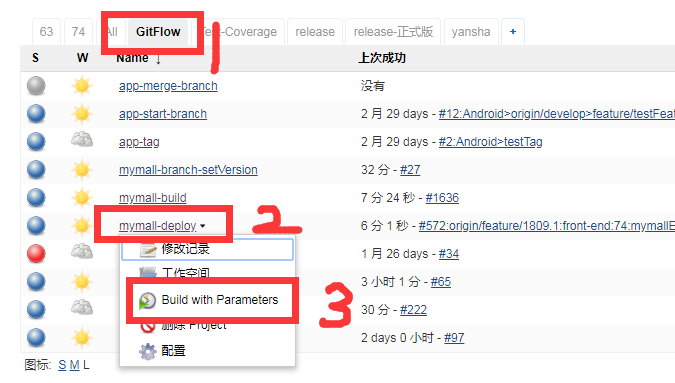


图5.2.6

1. 打开jenkens首页点击GitFlow
2. 鼠标悬浮到mymall-deploy上点击三角标
3. 点击Build with Parameters

上述操作完之后，会跳出如下视图：

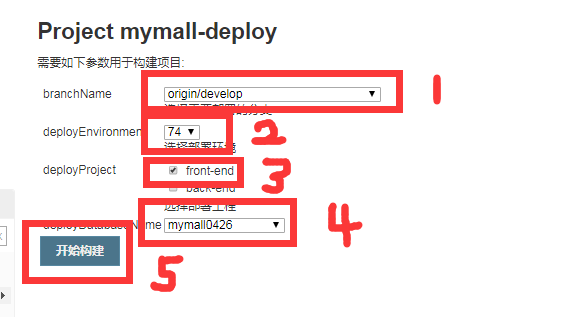


图5.2.7

1. 选择需要部署的分支
2. 选择74环境
3. 选择front-end，如果需要部署管理后台的话吧back-end勾上，是多选项。
4. 选择目标数据库，这个可能会变，具体问此次开发功能的负责人。
5. 开始构建

完成上面步骤之后可以通知测试同事测试功能，当测试同事提出bug 之后，，需要我们切换到第一步我们拉取的release分支进行修改bug。

### 5.2.3release分支使用完后的操作

当所有bug修改并通过测试后，将Release分支合并到Master和Develop中。

当上述三步完成后，给Master打Tag发布新的版本，发布版本之后这个预发布分支就被放弃掉。

## 5.3改bug分支

### 5.3.1何时创建Hotfixes分支

创建这个分支用来修复在正式环境里发现的bug，只用来修复指定的bug，不做其他事情。修复完bug之后这个分支就被放弃掉。

### 5.3.2创建Hotfixes分支过程

具体操作步骤如下：

在master上拉取分支，如图5.3.1：

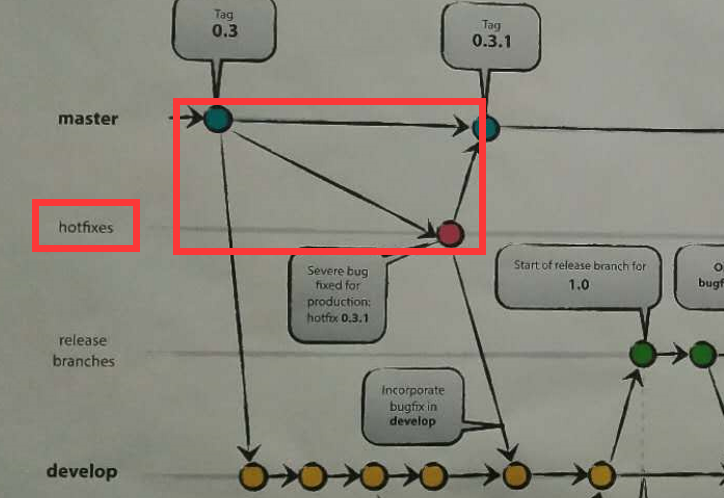


图5.3.1

拉取分支在Jenkens上的具体操作：

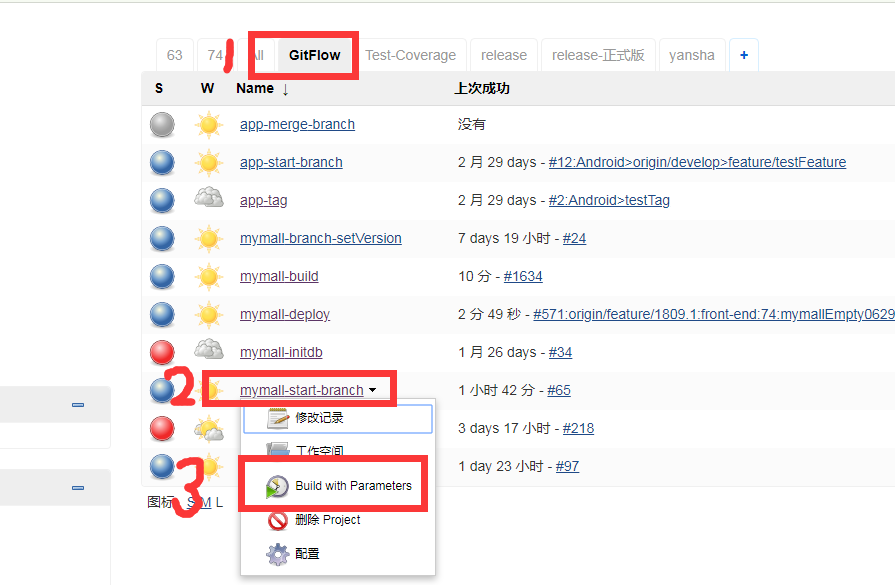


图5.3.2

1.进入jenkens首页面点击GitFlow视图

2.鼠标悬浮到mymall-start-branch上点击向下的三角标

3.点击Build with Parameters

上步骤操作完之后会跳出以下页面 如图8.3.3

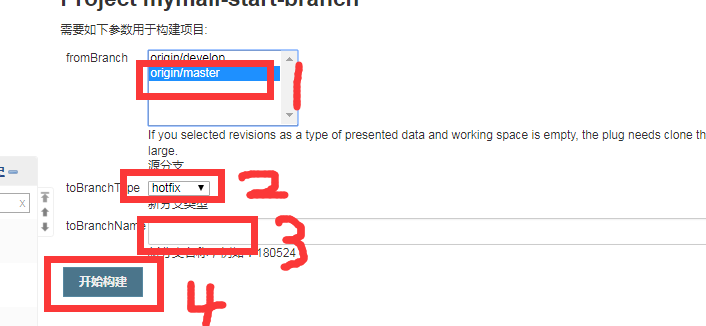


图5.3.3

1. 选择master，因为是修改正式环境的bug所以这步选择master
2. 类型选择hotfix
3. 分支名字，按修改bug的特征（英文）或时间命名
4. 点击构建

构建完成之后，在idea上检出这个分支，在该分支上修改bug。修改完bug之后将该分支部署到测试环境测试。

编译并部署项目，按如下步骤操作：

编译项目

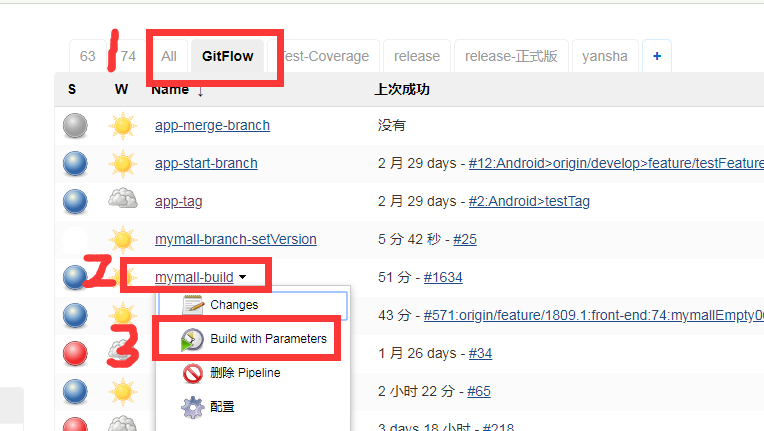


图5.3.4

1. 选择GitFlow视图
2. 选择mymall-build，鼠标悬浮在上面点击小三角
3. 点击build with Parameters

操作完上面步骤之后会出现如下视图，按步骤操作：

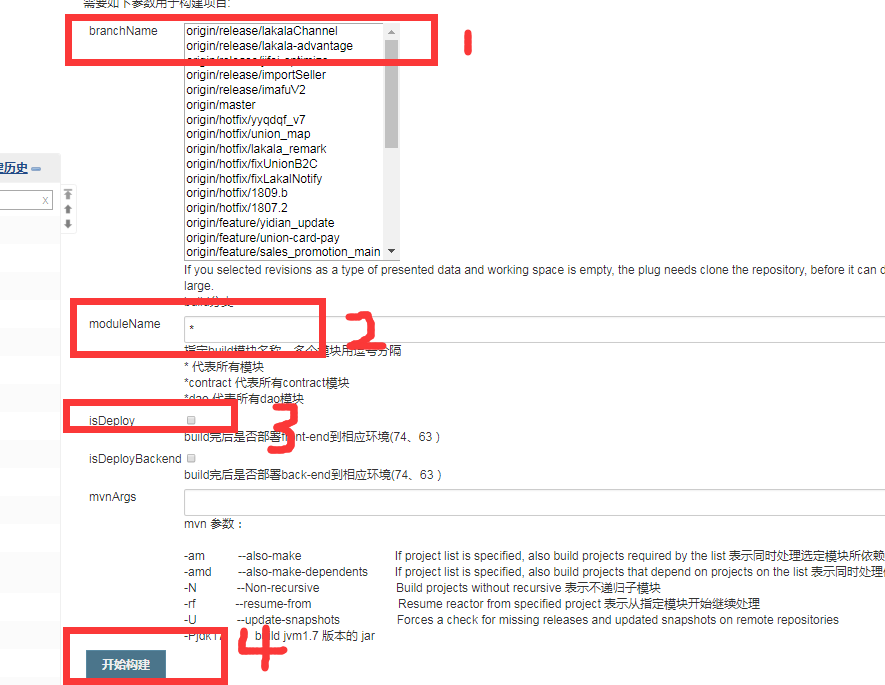


图5.3.5

1. 选择需要编译的hotfix分支
2. 选择模块编译，可快速完成编译，但一般不用
3. 编译完成部署项目
4. 点击开始构建

### 5.3.3Hotfixes分支使用完后的操作

部署完项目之后，通知测试人员测试。测试通过之后，将这个hotfix分支合并到develop

和master中。

给Master打Tag发布新的版本，发布版本之后这个hotfixes分支就被放弃掉。

## 5.4 打Tag

### 5.4.1何时打Tag

当需要发布新版本的时候给Master打Tag。

### 5.4.2创建Tag过程

Tag是在master上创建，如图5.4.1

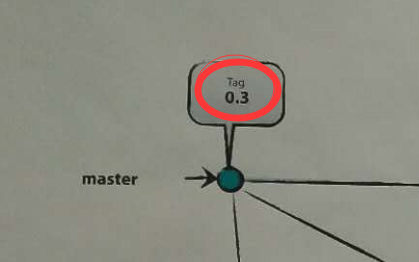


图5.4.1

具体的操作步骤在jenkins上操作，看图5.4.2：



图5.4.2

1. 打开jenkins首页，点击视图GitFlow
2. 鼠标悬浮在mymall-tag上，点击三角
3. 点击Build with Parameters

以上步骤操作完之后，跳出如下视图:

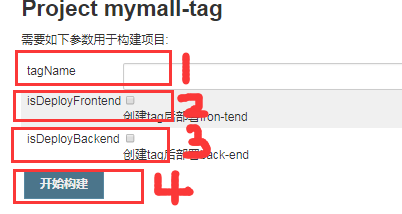


图5.4.3

1. Tag名称，命名是当前发布版本的版本号
2. 创建Tag后非后台管理部署
3. 创建Tag后后台管理部署
4. 构建

### 5.4.3为何要打Tag

创建Tag的目的是什么？这个问题想一分种，然后再点击查看下面答案。

1.当我们一个版本开发完成后，项目到达一个稳定版本，我们会做项目稳定版本备份。

2.给项目打Tag可以做到版本备份与切换

场景：突然发现v1.0.1版本效果没有v1.0.0效果好，需要切换v1.0.0。

### 5.4.4如何使用Tag切换版本

当我们需要切换版本时，步骤如下：



图5.4.4

1. 点击release视图
2. 鼠标悬浮到release-deploy，点击三角
3. 点击Build with Parameters

完成上述操作步骤之后，跳出如下视图,继续操作：



图5.4.5

1. 选择需要回退的版本
2. 勾选则部署商户后台
3. 勾选则部署商户管理后台
4. 构建

构建完成之后，发布版本。

## 5.5测试代码覆盖率

### 5.5.1何时使用Test-Coverage

将release分支部署到测试环境之后，想要查看测试人员在测试过程中覆盖了多少功能则需要使用Test-Coverage。

### 5.5.2如何部署Test-Coverage

Test-Coverage配置是写死的，我们部署测试环境同时就已经部署好Test-Coverage，不过我们可以重置Test-Coverage，操作部署如下：

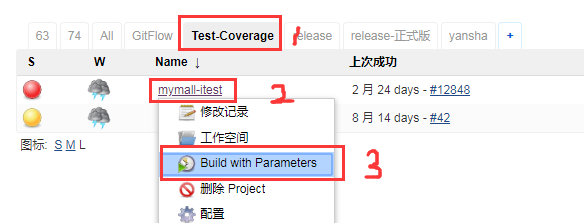


图5.5.1

1. 打开jenkins首页，点击Test-Coverage
2. 鼠标悬浮到mymall-itest上，点击三角
3. 点击Build with Parameters

完成上述操作后，弹出视图框，操作步骤如下：



图5.5.2

1. 勾选Reset
2. 点击构建

### 5.5.3如何使用Test-Coverage

当我们需要查看测试人员在测试的时候覆盖多少代码区域，需要进行以下操作：



图5.5.3

1. 打开Jenkins点击Test-Coverage
2. 点击带有下划线的百分比

完成以上操作之后，跳出新的视图：

如图5.5.4所示，在这个视图了里我们选择我们需要查看的包

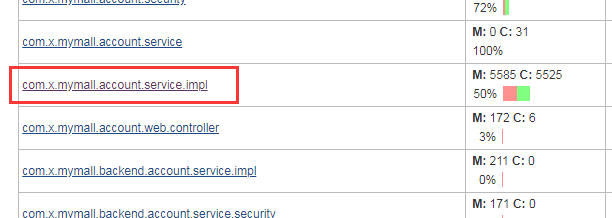


图5.5.4

选择完包之后选择我们需要查看的类名，如图5.5.5：

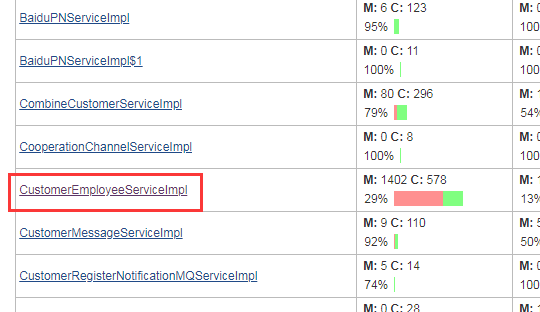


图5.5.5

选择完类之后选择需要查看的方法名，点击进5165会跳出如下视图，如图5.5.6：

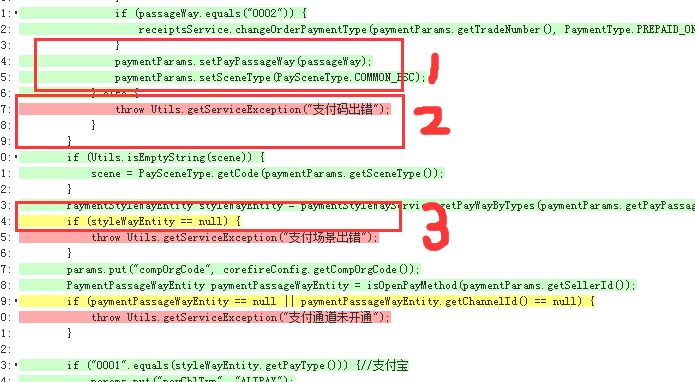


图5.5.6

1. 图5.5.6中所有被绿色覆盖的区域是测试时已经走过的代码。
2. 图5.5.6中所有被黄色覆盖的区域是测试时未完全走完的代码。
3. 图5.5.6中所有被红色覆盖的区域是测试时未走过的代码。

# 分支合并

## 6.1何时需要合并分支

在第五章提到的创建各个分支，当我们使用分支完成任务之后，则需要将分支合并。

## 6.2合并分支的过程

先切换到目标分支，例如图6.1中的master分支

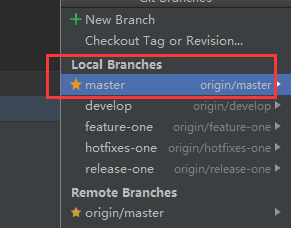


图6.1

切换完成后，选择需要合并的分支，假设要合并feature-one，如图6.2

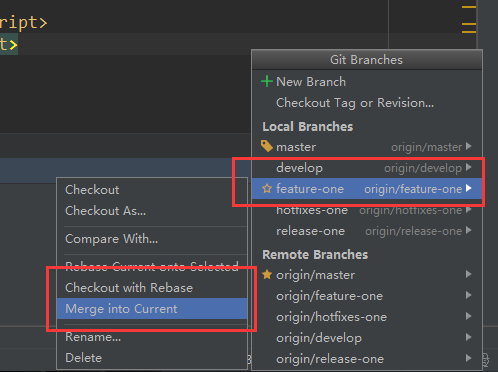


图6.2

1. 选择需要合并的分支
2. 选择 Merge into Current

如果没有文件冲突，则合并完成。

## 6.3合并分支时冲突的解决办法

当合并时出现冲突会跳出提示框，如图6.3

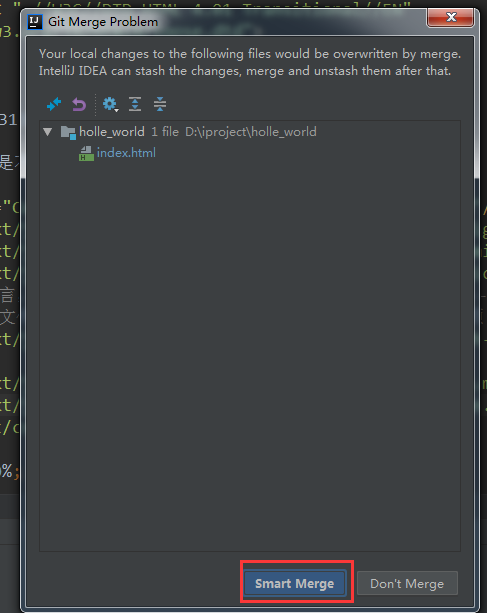


图6.3

点击Smart merge

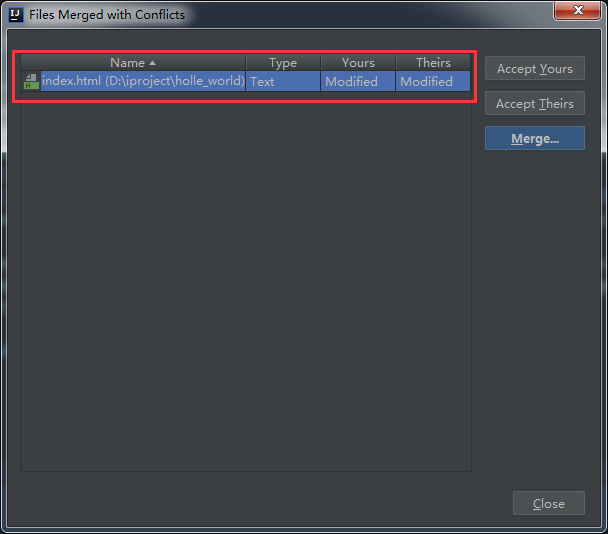


图6.4

点击查看冲突的内容，手动合并内容。

# 发布版本

## 7.1何时发布新版本

发布新版本的时间一般是由测试负责人和开发负责人制定。

## 7.2发布版本的操作流程

当测试和开发负责人同意发版，并且master打完Tag之后

进行发版操作，具体操作如下：



图7.1

1. 选择这次发版打的Tag
2. 部署商户后台则勾选2
3. 部署商户管理后台则勾选3
4. 构建

完成以上操作步骤之后，继续以下操作步骤：

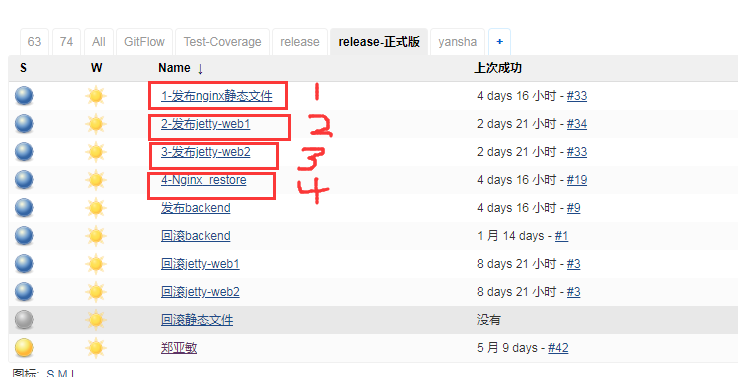


图7.2

1. 同步静态文件
2. 将新的版本部署到jetty-web1中，此时jetty-web2承担所有的工作。
3. 将新的版本部署到jetty-web2中，此时jetty-web1承担所有的工作。
4. 恢复负载。

如果需要发布后台，则点击图7.1中 发布backend直接构建。

## 7.3部署失败如何处理

当部署失败或遇到需要返回之前版本的状况时需要回滚操作，步骤如下：



图7.3

1. 回滚静态文件
2. 回滚jetty-web2
3. 回滚jetty-web1

如果部署后台出现问题，则点击回滚backend，回退到上一版本。