

# 基于 git/gitLab 的版本管理

# 常见的 git 操作问题

星期五 下午 3:45

git 整理的问题：

- 1、git 代码管理的流程，init、add、commit、pull、push、rollback 等 **开发流程**
- 2、多人联合开发一个项目，代码如何管理，主干和分支管理 **分支管理**
- 3、多人维护同一个项目，代码如何管理？是在同一个分支上进行开发，每个人在提交前 pull，还是在不同分支上开发，然后 merge 分支  
本地修改不想 push 到远程仓库，但是想从远程仓库上 pull 最新的代码，本地分支的改动会丢失吗？如何避免 **分支管理/合并操作**
- 4、同时提交对一个文件的修改版本，主干选择合并 A，B 应该如何操作 **合并操作**
- 5、常见的 pull 失败和 push 冲突问题 **合并操作**
- 6、clone 代码出现：clone succeed, but checkout failed **百度搜一下，windows 下文件路径过长？**

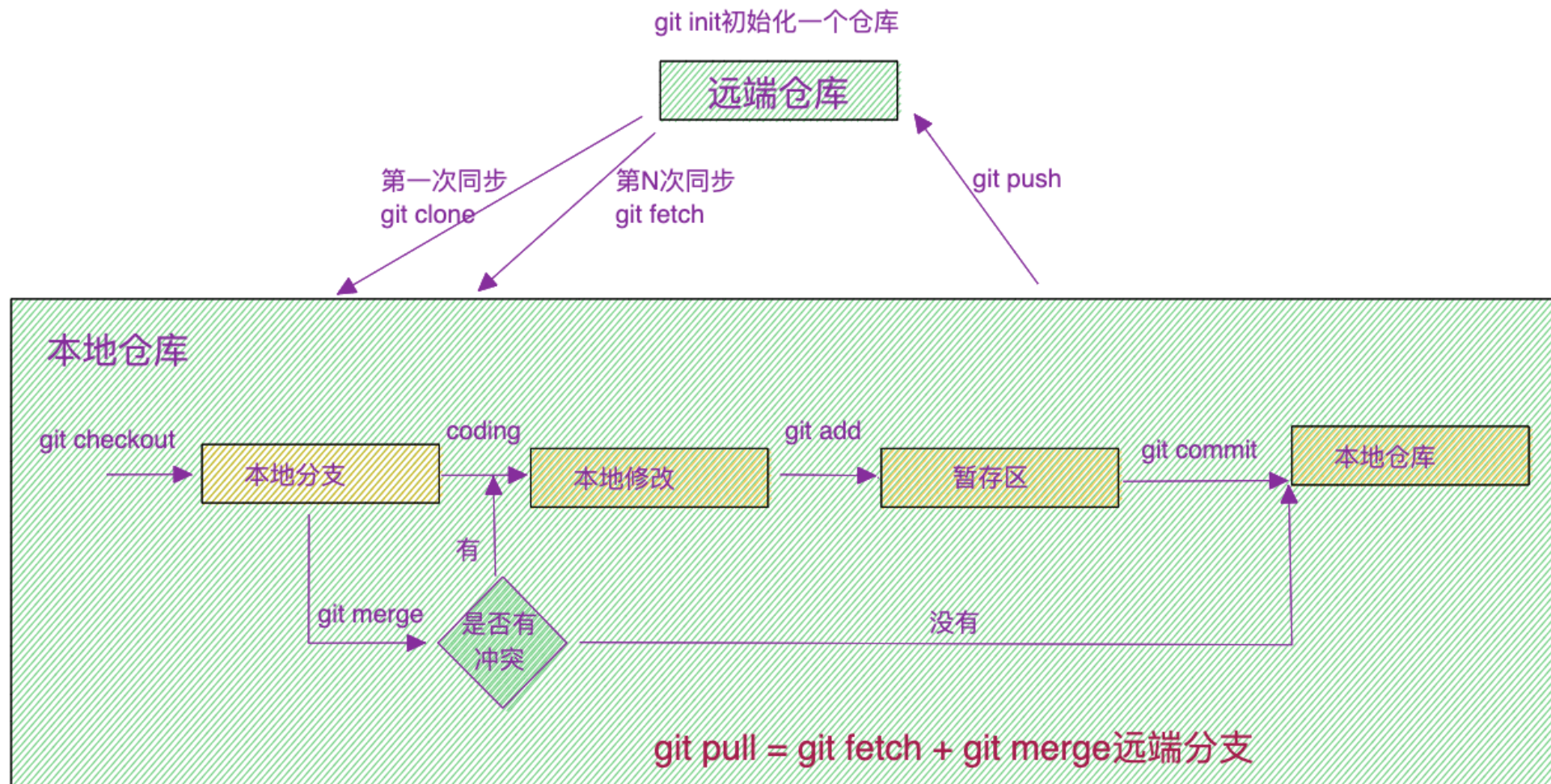
# 本次分享的内容

从以下 5 个方面介绍常用的 git 开发规范及 gitlab 的一些操作设置

- 常见的 git 操作/流程
- 分支策略
- commit 规范
- 合并规范
- 发版操作

# 常见的 git 操作

PS: 本地仓库 = .git(所有提交的快照) + 本地文件



## 常见的 git 操作问题-回退/撤销

- `git revert commitId`: 新增一笔提交, 内容为插撤销对应的修改内容
- `git reset --hard commitId`: 直接将当前分支的 head 指向之前的某一笔提交
- `git rebase -i commitId`: 针对当前提交到指定提交之间的所有提交重新进行编辑

# 常见的 git 操作问题-简写

配置文件目录：

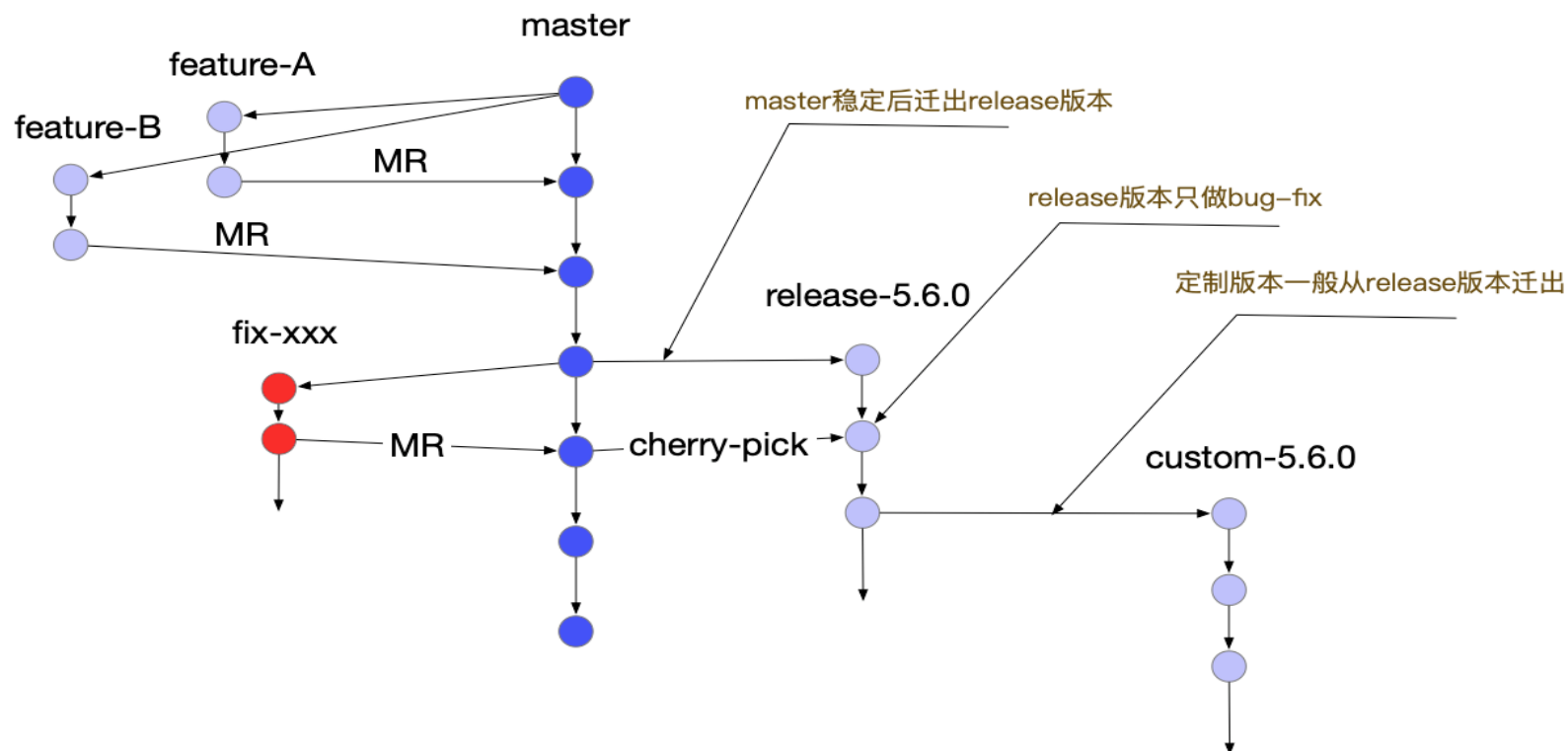
~/**.gitconfig**

```
[user]
    name = haoke
    email = haoke@i-i.ai

[alias]
    st = status
    br = branch
    cm = commit
    re = rebase
    cp = cherry-pick
    l = log
    f = fetch
    ps = push
    pl = pull
    co=checkout
```

# 分支策略

需求驱动开发+ 维持固定主线



## 分支策略-分支说明

1. `master` - 主干分支，长期迭代，稳定分支，需求和 bug-fix 都合入该分支

权限：不能删除，不能 push，需要 merge request 合入

2. `feature-xxx` - 功能开发分支，短期迭代，从 release 分支切出，- 在最后的 merge request 前，需要通过命令 `git rebase` 对应分支 合并 master 分支 - 开发测试后合入 master 分支

权限：分支开发完毕后需要删除，可直接 push 代码到该分支

3. `bugfix-xxx` - 修复分支，短期分支 - 可以通过 merge request 先合入 master 或 release 分支，再 cherry-pick 到需要的分支

权限：分支开发完毕后需要删除，可直接 push 代码到该分支



## 分支策略-分支说明-只有唯一主线的分支策略

4. `dev` - 长期开发分支，供开发人员使用，feature 分支从该分支迁出进行开发，解决有固定迭代周期的多需求开发合入问题

权限：不能删除，不能 push，feature 分支需要 merge request 合入 dev

5. `qa` - 测试分支，供测试人员进行功能验证，解决有固定迭代周期的多需求开发合入问题

权限：不能删除，不能 push，一般从通过 merge request 合入 dev 分支

## 分支策略-分支说明-有定制和版本维护需求的策略

6. `release-xxx` - 发布分支（可选，针对需要长期维护的特定版本）

权限：不能删除，不能 push，需要 merge request 合入，在发版本后打 tag，符合版本号规范，如 **release-5.6.0**

7. `custom-xxx` - 定制分支（可选，针对特定客户或产品的分支，与 `release-xxx` 类似，但不具备版本延续性）

权限：不能删除，不能 push，需要 merge request 合入，发版后打 tag，符合版本号规范，如 **\*\*custom-yancao-5.6.0**，\*\*定制分支迭代后需要持续增加版本号

## 分支策略-gitlab 分支操作-分支权限

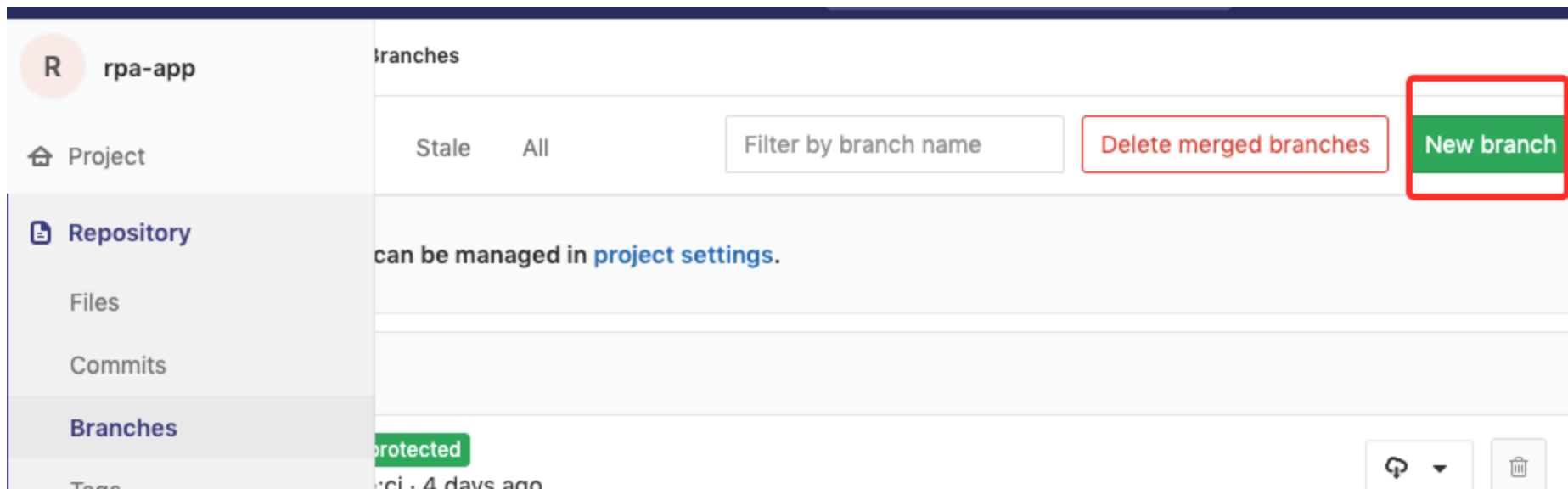
设置路径：项目 -> Settings -> Repository -> Protected Branches

对分支进行权限设置, 特定分支名 和 固定开头 的分支设定可以推送/合并的人员

Protected branch (27)	Last commit	Allowed to merge	Allowed to push	
develop-*	26 matching branches	Maintainers ▼	Maintainers ▼	Unprotect
master <span>default</span>	466a33a9 1 month ago	Maintainers ▼	No one ▼	Unprotect

# 分支策略-gitlab 分支操作-分支创建

设置了权限的分支只能由有权限的人本地或在 web 端创建



# commit 规范

## 提交格式

```
<type>(<scope>): <subject>  
<BLANK LINE>  
<body>  
<BLANK LINE>  
<footer>
```

参考:Angular 提交信息规范

提交类型指定为下面其中一个:

1. `build`: 对构建系统或者外部依赖项进行了修改
2. `ci`: 对CI配置文件或脚本进行了修改
3. `docs`: 对文档进行了修改
4. `feat`: 增加新的特征
5. `fix`: 修复 bug
6. `perf`: 提高性能的代码更改
7. `refactor`: 既不是修复 bug 也不是添加特征的代码重构
8. `style`: 不影响代码含义的修改, 比如空格、格式化、缺失的分号等
9. `test`: 增加确实的测试或者矫正已存在的测试

# commit 规范-实例

```
git commit -m '[feat]: 添加新的功能'
```

**feat:** 支持中枢变量备注显示

罗(周骐) authored 6 hours ago

**fix:** 单位输入框模板 添加变量

shenping authored 7 hours ago

**fix:** --bug=1014575 --user=taozhizhu sdk的依赖包不存在时

桃之助 authored 2 days ago

**fix:** --bug=1014990 修复备注300字不能添加

罗(周骐) authored 7 hours ago

**feat:** 全局变量加备注

罗(周骐) authored 1 day ago

**fix:** 依赖包-元素依赖包新版元素的切割版本号是1.1.9.0

桃之助 authored 8 hours ago

# commit 规范

commit 的信息通过特殊约定，实现与缺陷跟踪系统的联动

```
// --story --user --bug等设置与tapd上的信息关联
```

```
git commit -m '[feat]: --story=*** --user=*** 添加新的功能'
```

 **[ID1006384] 【Bug转需求】循环数据表内容转化源代码后，切换内容改变**  

原始缺陷ID: [1013173](#)

详细信息 子需求 0 缺陷 0 任务 1 测试用例 0 变更历史 13 [Gitlab提交 \(2\)](#) 

关联源码提交



lab助手

**feat: --story=1006384 --user=haoke** 添加数据表内容转组件

提交于: 2021-09-17 16:56:37 源码提交人:haoke 版本库:  [rpa-app](#) feat-flowchart-queue 版本号: 0ebf

-  M [rpa-studio/resources/compiler/Transformer.py](#)
-  M [rpa-studio/resources/compiler/main.py](#)
-  M [rpa-studio/src/editor/task-editor/base-components/logic\\_control.ts](#)
-  M [rpa-studio/src/platform/compiler/browser/SemanticAnalysis.ts](#)

# 合并规范

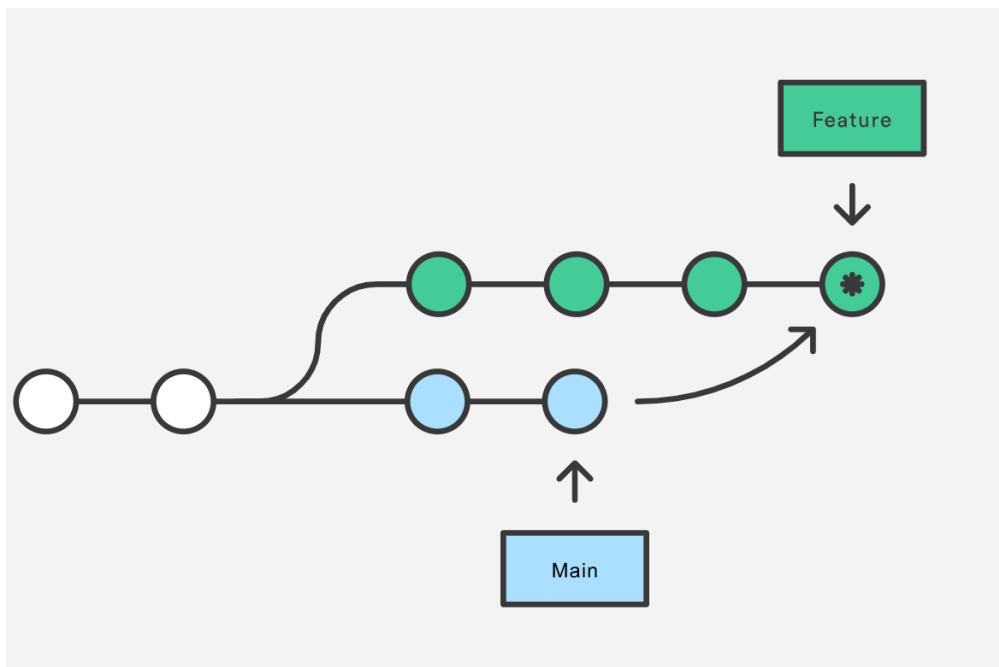
合并代码的目的：

- 1、解决与远端代码的冲突
- 2、本地代码比远端代码新，即将修改置于远端代码之上（git rebase 可以修改顺序）



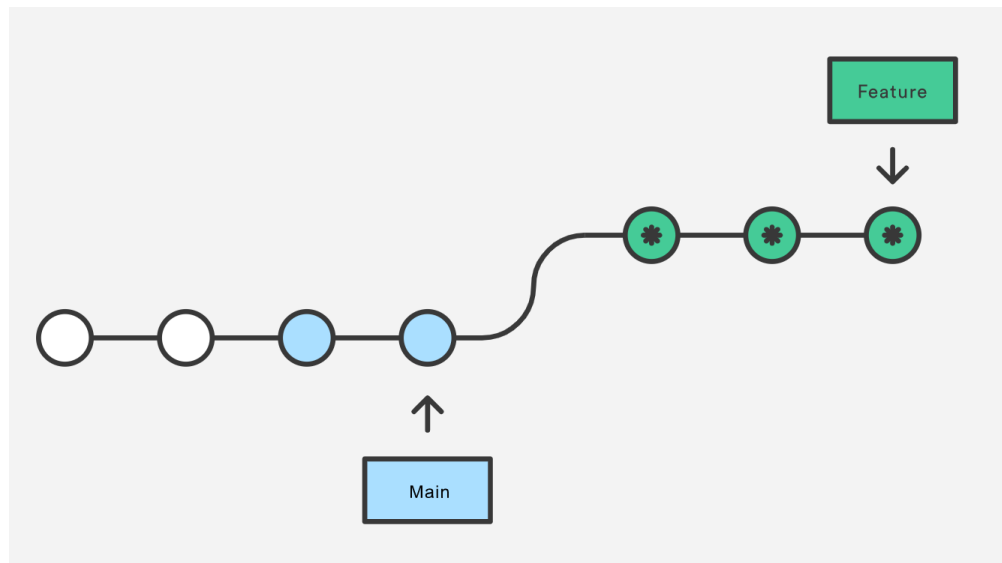
# 合并规范-本地合并

- `git pull = git fetch + git merge`
- `git merge`: 按提交时间合并两个分支, 保留分叉形式, 最后一个一笔 merge 提交



# 合并规范-本地合并

- git rebase: 变基操作，所有提交放在目标分支，没有多余提交



# 合并规范-本地合并冲突

不同开发人员修改同一分文件肯定会存在冲突问题，git 合并中肯定会出现冲突

- 需要在 `git merge` / `git rebase` 中手动解决冲突，再进行提交
- 后提交的人需要解决冲突，在解决过程中对代码有疑问的需要与当事人一起 review

# 合并规范-merge request

通过 merge request 解决乱推代码，并在 review 中提高代码提交质量

The screenshot shows the 'Merge Requests' page for a project named 'test-pipeline'. The left sidebar contains navigation links: Project, Repository, Issues (0), Merge Requests (1), CI / CD, and Operations. The main content area has a breadcrumb 'ued > test > test-pipeline > Merge Requests' and a header with filters: 'Open 1', 'Merged 1', 'Closed 0', and 'All 2'. There are buttons for 'Edit merge requests' and 'New merge request' (highlighted with a red box and the label '发起MR'). Below the filters is a search bar 'Search or filter results...' and a 'Created date' dropdown. A single merge request is listed, titled 'testa', with the description '!2 · opened 1 day ago by 浩克 (刘智刚)' and a 'factory' label. This entry is highlighted with a red box and the label '待合并的merge request'. To the right of the entry are status icons (a green checkmark, a green circle with a cross, and a speech bubble with '0') and the text 'updated 1 day ago'.

ued > test > test-pipeline > Merge Requests

发起MR

Open 1 Merged 1 Closed 0 All 2

Edit merge requests New merge request

Search or filter results... Created date

testa  
!2 · opened 1 day ago by 浩克 (刘智刚) factory


待合并的merge request


updated 1 day ago



# 合并规范-CR

通过 merge request 解决乱推代码，并在 review 中提高代码提交质量




## 旺店通core 增加日志打印 去除默写报警抛出

**Request to merge** [jinyun-wangdiantong](#) into [develop-5.9.9](#)  
The source branch is [18 commits behind](#) the target branch

[Open in Web IDE](#) [Check out branch](#) 

**Merge** ☒ Remove source branch ☒ Squash commits  [Modify commit message](#)




*You can merge this merge request manually using the [command line](#)*

 0  0 

[Discussion](#) 0 [Commits](#) 3 **[Changes](#)** 2

Changes between [latest version](#) and [develop-5.9.9](#)

Showing [2 changed files](#) with [3 additions](#) and [1 deletion](#) [Hide whitespace changes](#) [Inline](#) [Side-by-side](#)

  [src/II.RPA.Plugin.WdtErp/Libs/WdtErpCore.dll](#) 

[View file @ 64a600c6](#)

by 浩克

# 发版操作

## 发版：

- 合入 master
- git tag 打标签

## 线上回滚操作

- git checkout tagId

# 本次分享总结

- 常见的 git 操作/流程: git fetch/git merge/git push/git rebase/ git revert
- commit 规范: feat: --bug=xxxx --user=xxx 添加新功能(或放入缺陷地址+标题)
- 分支策略: gitlab flow, 保留统一分支, 设定特定分支权限, 通过 feature/bugfix 分支开发
- 合并规范: 本地 merge/rebase 操作, 提交 merge request 进行合并, 在 gitlab 进行 code review
- 发版操作: git tag 打标签

