

GIT使用说明

2015-02-11

Intelligent Automation (Zhuhai) Co., Ltd.
www.intelligentgroup.cn

A graphic consisting of three overlapping, semi-transparent blue squares with white outlines, arranged in a triangular pattern. The text "Automation Testing Machining" is centered over the squares in a white, bold, sans-serif font.

Automation Testing Machining

1. Over View
2. Install GIT
3. Client Tools-Source Tree
4. Intelligent GIT Server
5. GIT Command

Over View

GIT主要具有以下特点:

- 适合分布式开发, 强调个体
- 公共服务器压力和数据量都不会太大。
- 速度快、灵活。
- 任意两个开发者之间可以很容易的解决冲突。
- 支持离线工作。

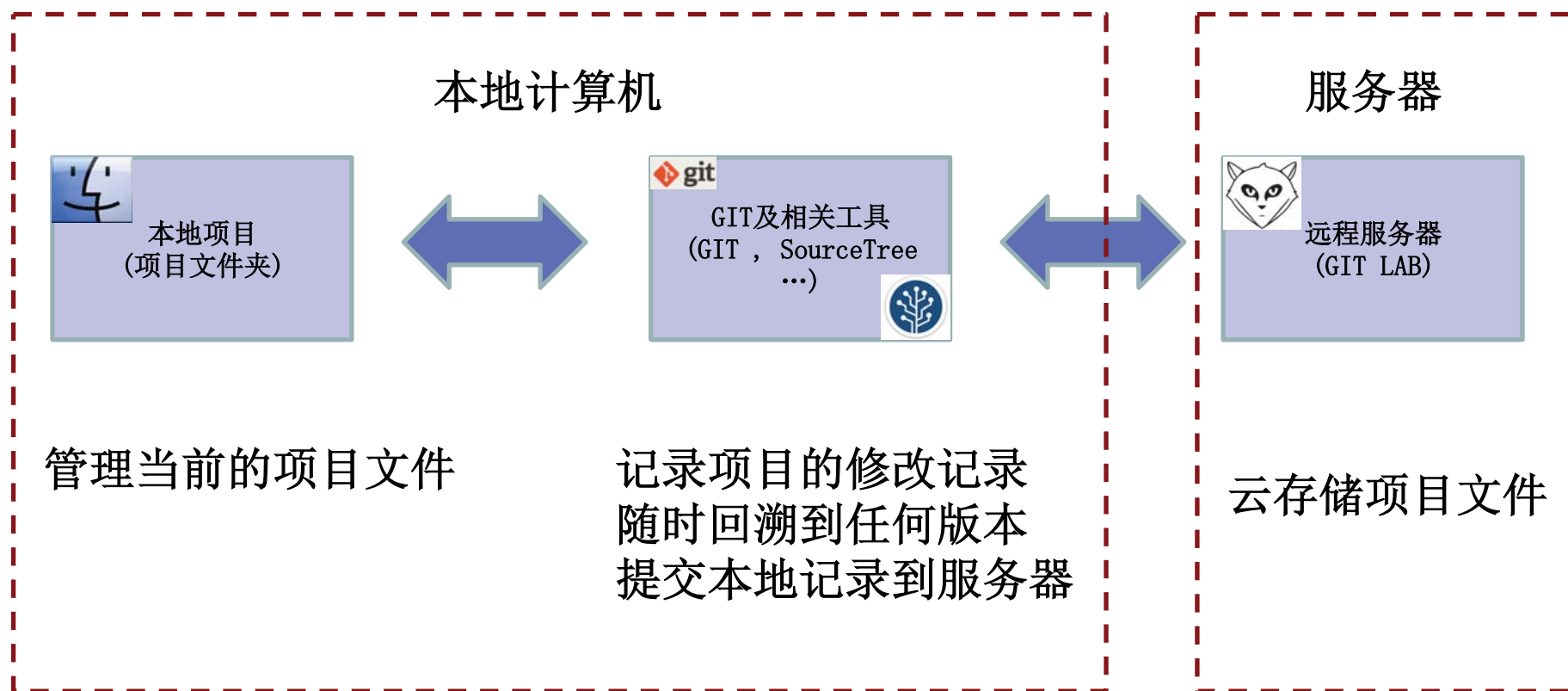
从一般开发者的角度来看, **GIT**有以下功能:

- 1、从服务器上克隆数据库(包括代码和版本信息)到单机上。
- 2、在自己的机器上创建分支, 修改代码。
- 3、在单机上自己创建的分支上提交代码。
- 4、在单机上合并分支。
- 5、新建一个分支, 把服务器上最新版的代码**fetch**下来, 然后跟自己的主分支合并。
- 6、生成补丁, 并把不定发送给主开发者
- 7、看主开发者的反馈, 成员协作解决冲突

从主开发者的角度看, **GIT**有以下功能:

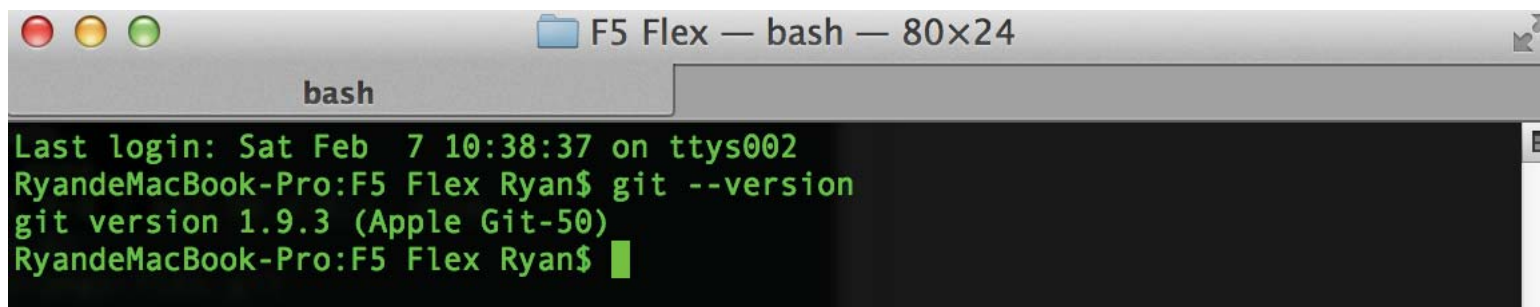
- 1、查看邮件或者通过其它方式查看一般开发者的提交状态。
- 2、打上补丁, 解决冲突
- 3、向公共服务器提交结果, 然后通知所有开发人员。

GIT系统组成



GIT安装 --- 安装GIT

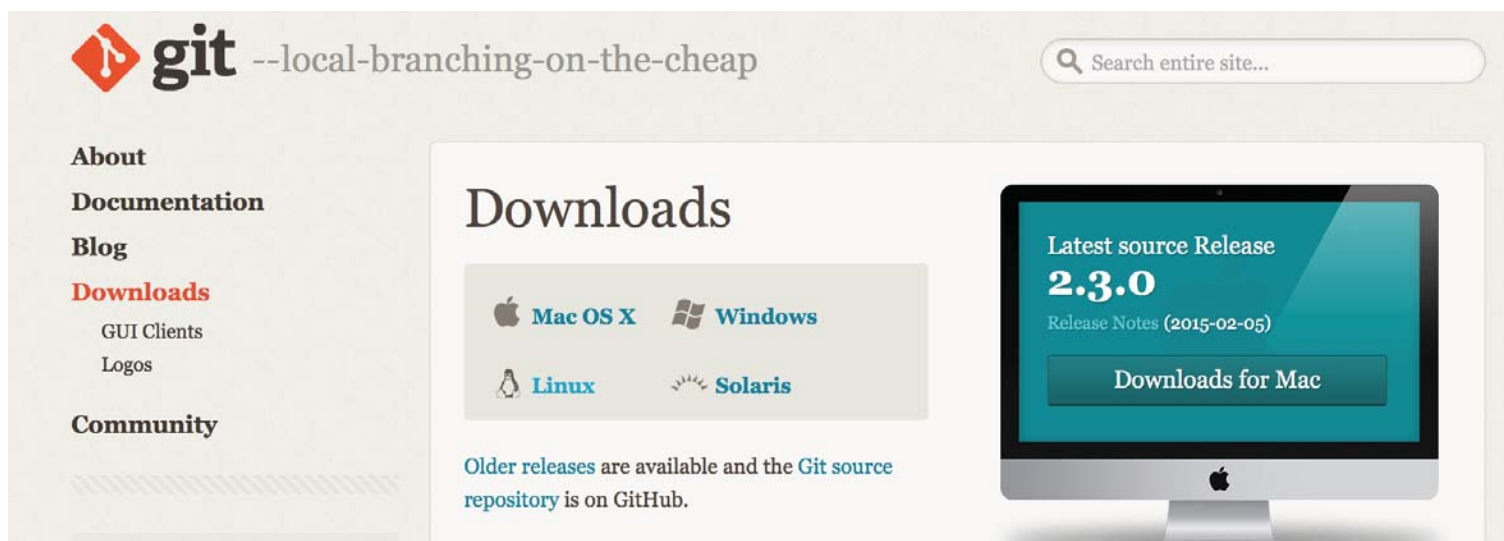
1. Mac OS默认是已经安装完GIT，可以通过终端来运行如下命令来检查：



```
F5 Flex — bash — 80x24
bash
Last login: Sat Feb  7 10:38:37 on ttys002
RyandeMacBook-Pro:F5 Flex Ryan$ git --version
git version 1.9.3 (Apple Git-50)
RyandeMacBook-Pro:F5 Flex Ryan$
```

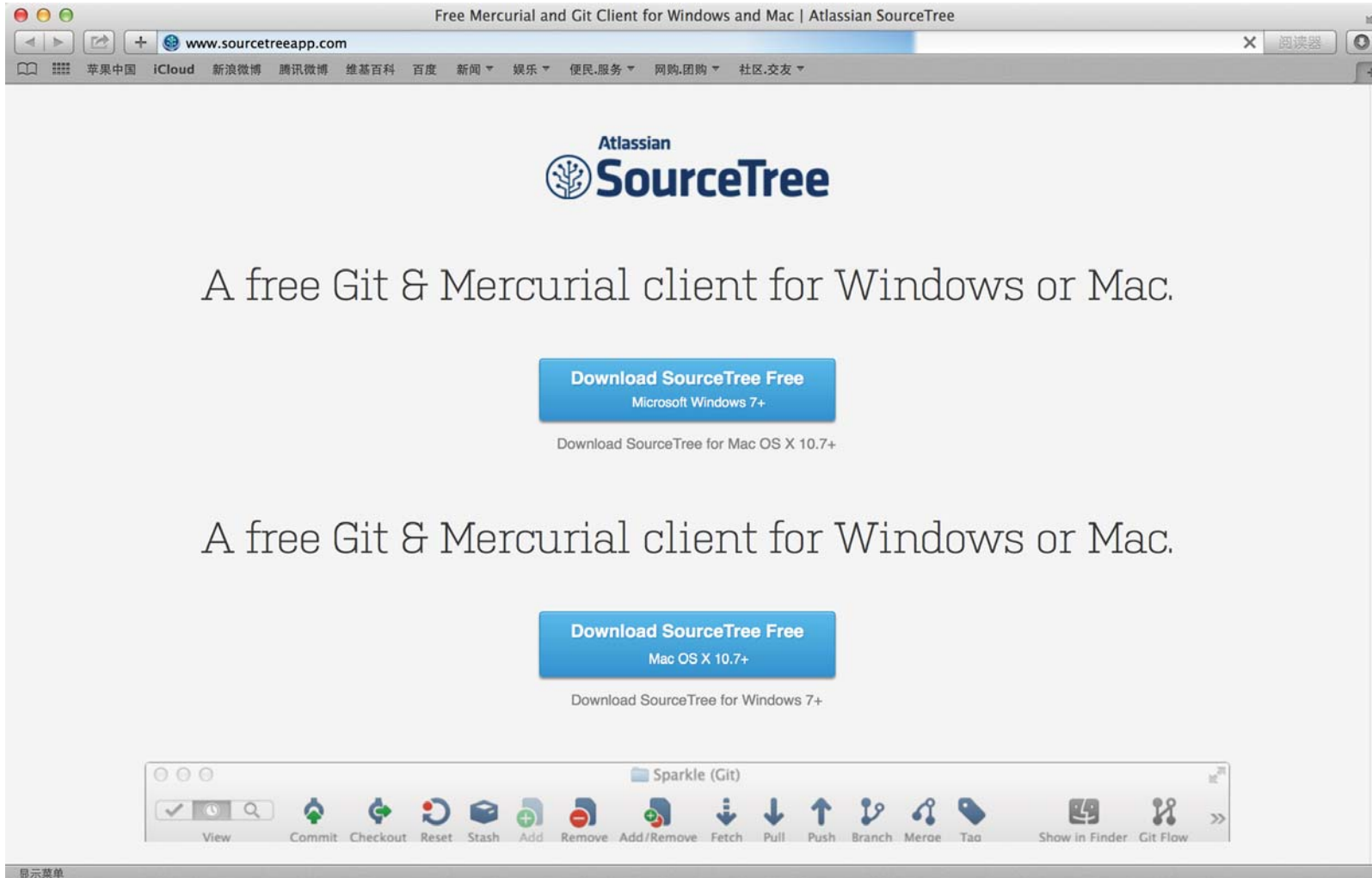
2. 如果没有安装，请从如下连接下载对应版本并安装：

<http://git-scm.com/download/>



GIT安装 --- 客户端工具: SourceTree

- 1.开发人员建议使用**SourceTree**,可以从以下网站免费下载对应的版本:
<http://www.sourcetreeapp.com>



GIT安装 --- 客户端工具: SourceTree

1.打开下载下来的**SourceTree_2.0.3.dmg**文件，按提示将**APP**拖进**Application**文件夹。(Windows默认按提示安装即可)



SourceTree --- 创建新项目

1. 创建一个新项目



新仓库→创建本地仓库



输入存储项目的目录



创建完成

SourceTree --- 添加已有的项目

1. 添加一个已有项目:

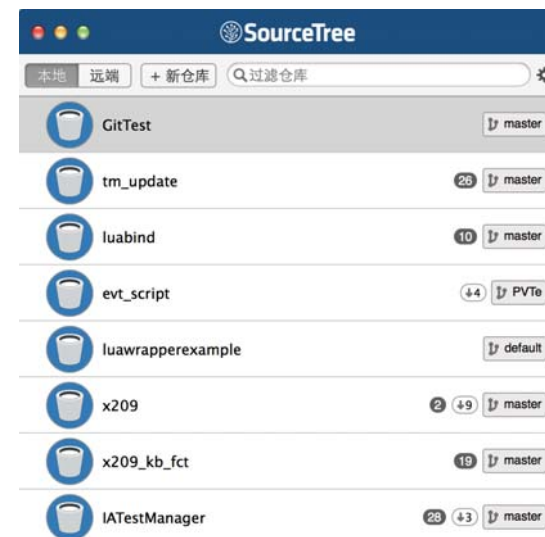
如果有一个已经创建好的项目，可以按如下步骤，将其添加到**GIT**来管理



新仓库→添加存在仓库



选择已存在的目录，
并设置类型为**GIT**



添加完成

SourceTree --- 项目管理与跟踪

双击一个创建好的项目，进入该项目的管理和跟踪页面

The screenshot displays the SourceTree application interface for a project named 'IATestManager (Git)'. The interface is divided into several sections:

- Left Panel:** Contains a sidebar with '文件状态' (File Status), '分支' (Branches), 'REMOTE' (Remotes), 'STASHES' (Stashes), '子模块' (Submodules), and '子树' (Subtrees). The '分支' section is expanded, showing a list of branches including 'origin', 'DVT', 'EVT', 'fixtured', 'HEAD', 'master', and 'Process-DOE'. A red label '分支' points to this section.
- Top Panel:** Displays 'Uncommitted changes' and a list of commits. A red label '当前项目的被修订提交的记录' points to this section.
- Bottom Panel:** Shows a list of files in the project, including 'DFUScripts/DFU.IASeq/index.plist' and 'DFUScripts/DFU.IASeq/Contents/x240_dfu_in760xdfuxevt.lmp'. A red label '被选中提交中，被改动的文件列表' points to this section.
- Right Panel:** Displays the content of the selected file, showing a code snippet for 'function ParseOutResponse(par)'. A red label '被选中的文件里，被改动的内容' points to this section.

The interface also includes a top toolbar with various icons for file operations, a search bar, and a status bar at the bottom.

SourceTree --- 项目管理与跟踪

当对一个项目的文件进行了相关修改好，选择**Uncommitted Changes**

The screenshot displays the SourceTree application interface for the 'IATestManager (Git)' repository. The central pane shows the 'Uncommitted changes' section, which lists several files that have been modified but not yet committed. These files are listed under the '已暂存文件' (Staged) section, including:

- IATestManager/TestBundle/TestBundle/ClassExport.mm
- IATestManager/TestBundle/TestBundle/CTExport-global.mm
- IATestManager/TestBundle/TestBundle/CTExport-lua.mm
- IATestManager/TestBundle/TestBundle/CTExport.cpp
- IATestManager/TestBundle/TestBundle/CTExport.h
- IATestManager/TestBundle/TestBundle/CTExport.pkg
- IATestManager/TestBundle/TestBundle/CTExport_global.pkg

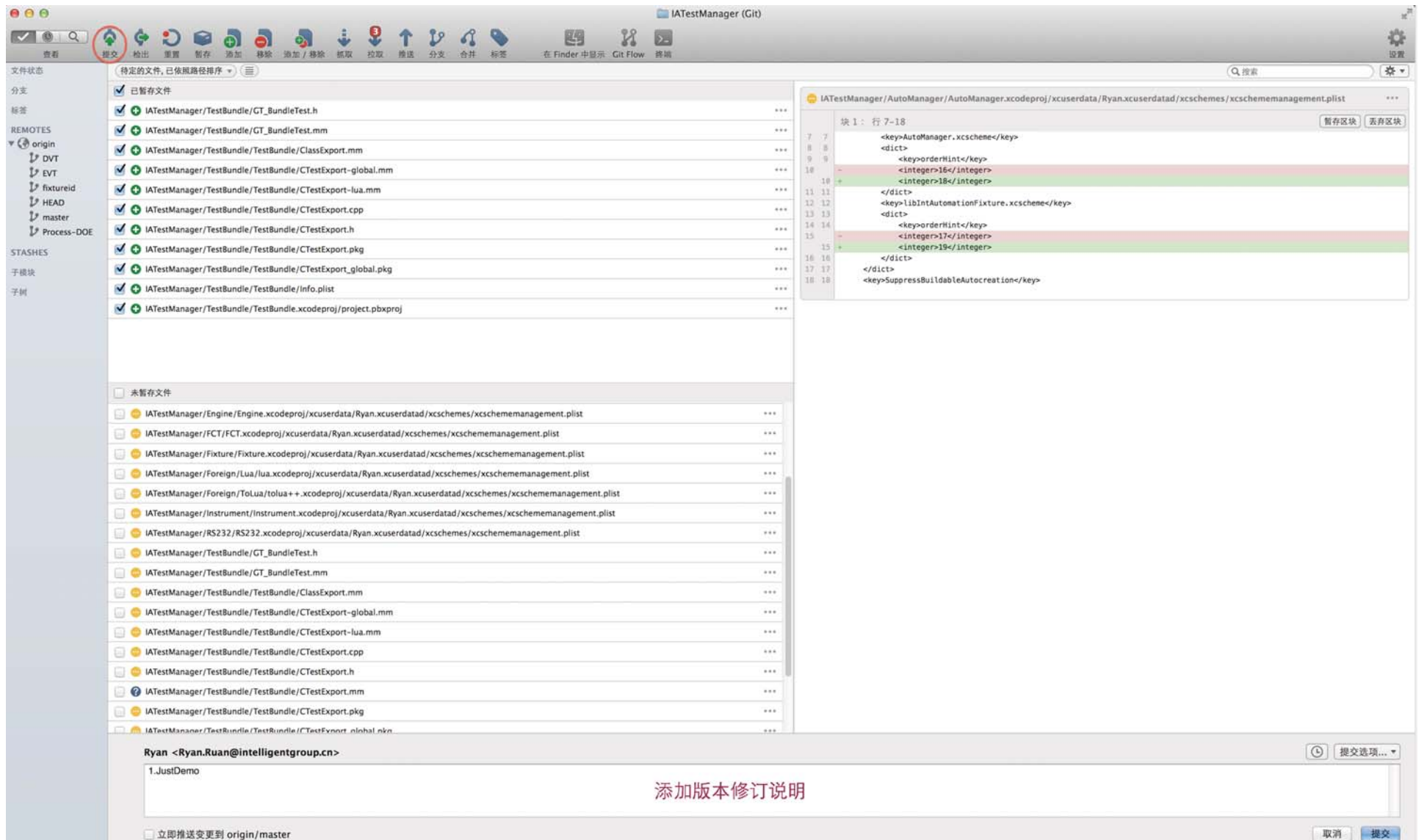
Below these, the '未暂存文件' (Unstaged) section lists various other files, including 'IATestManager/AutoManager/AutoManager.xcodeproj/xcuserdata/Ryan.xcuserdatad/xcschemes/xcschememanagement.plist'. A red annotation points to this file, stating: '在需要存档的修改文件前勾上，该文件将被暂存到上方' (Check the box before the files you want to archive, this file will be staged to the top).

On the right side of the interface, the 'Commit' tab is active, showing a list of commit messages and their authors. A red annotation points to the 'Commit' button, stating: '选择UnCommit Changes,显示当前项目没有被保存的更改' (Select UnCommit Changes, display current project changes not saved).

The bottom right pane shows a preview of the selected file, 'IATestManager/AutoManager/AutoManager.xcodeproj/xcuserdata/Ryan.xcuserdatad/xcschemes/xcschememanagement.plist', displaying its XML content.

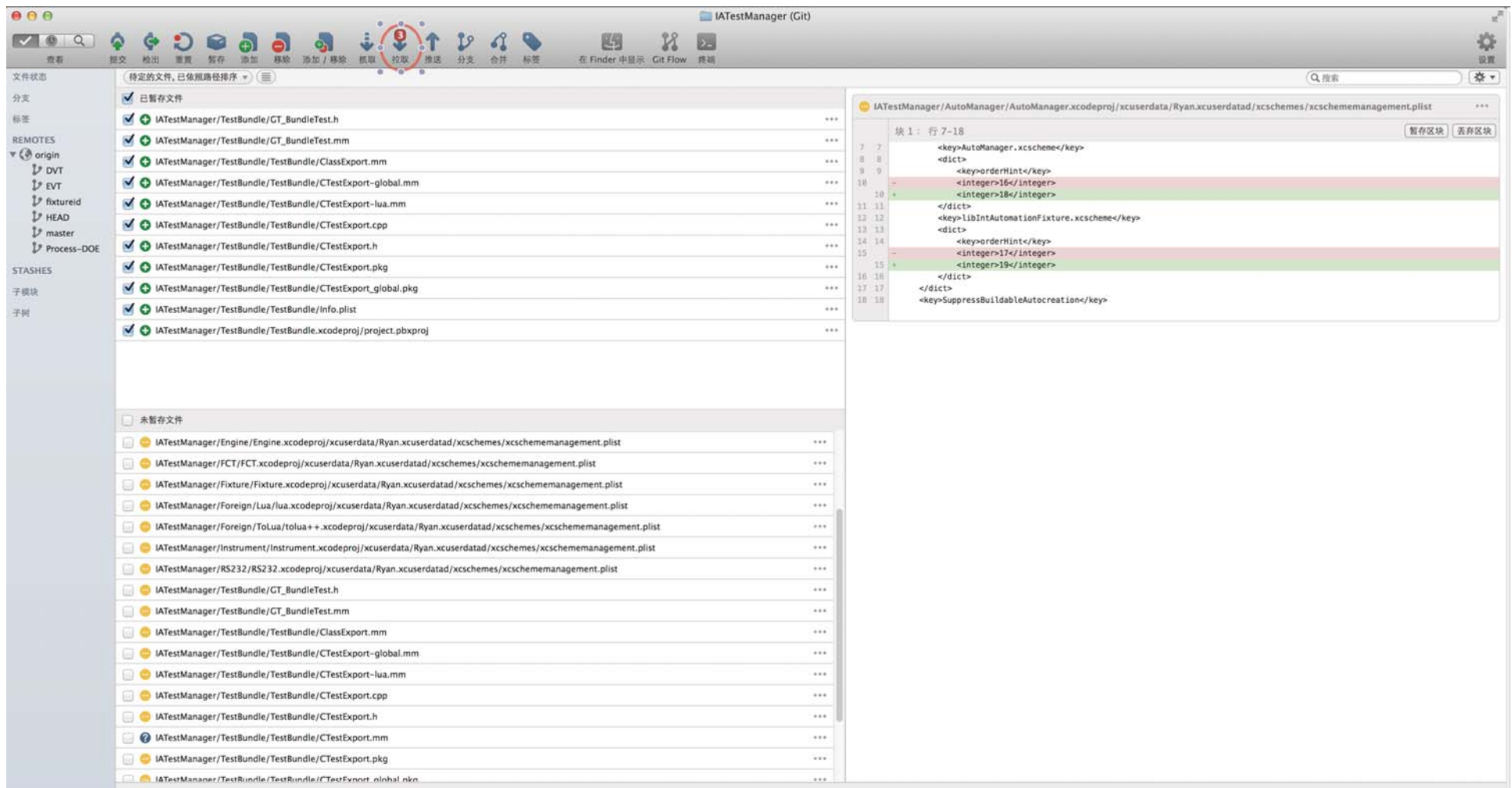
SourceTree --- 项目管理与跟踪

点击提交，并输出如修订说明，完成项目的本地版本存储



SourceTree --- 项目管理与跟踪

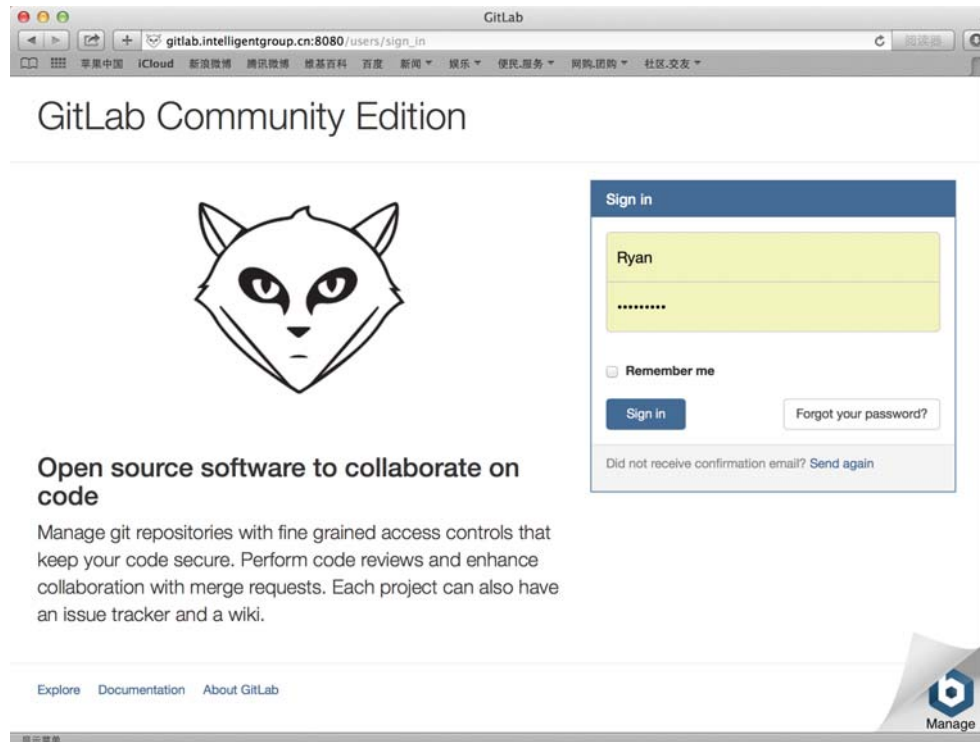
GIT支持多人编辑同一个项目，当别人有提交项目的时候，点击”拉取”即可同步获取到最新的代码



GIT服务器

- 1.前面所有的操作都是基于本地计算机存储
- 2.公司所有软件项目，脚本等都需要存储在公司的**GIT**的服务器上。
- 3.可以通过如下连接访问公司的**GIT**服务器

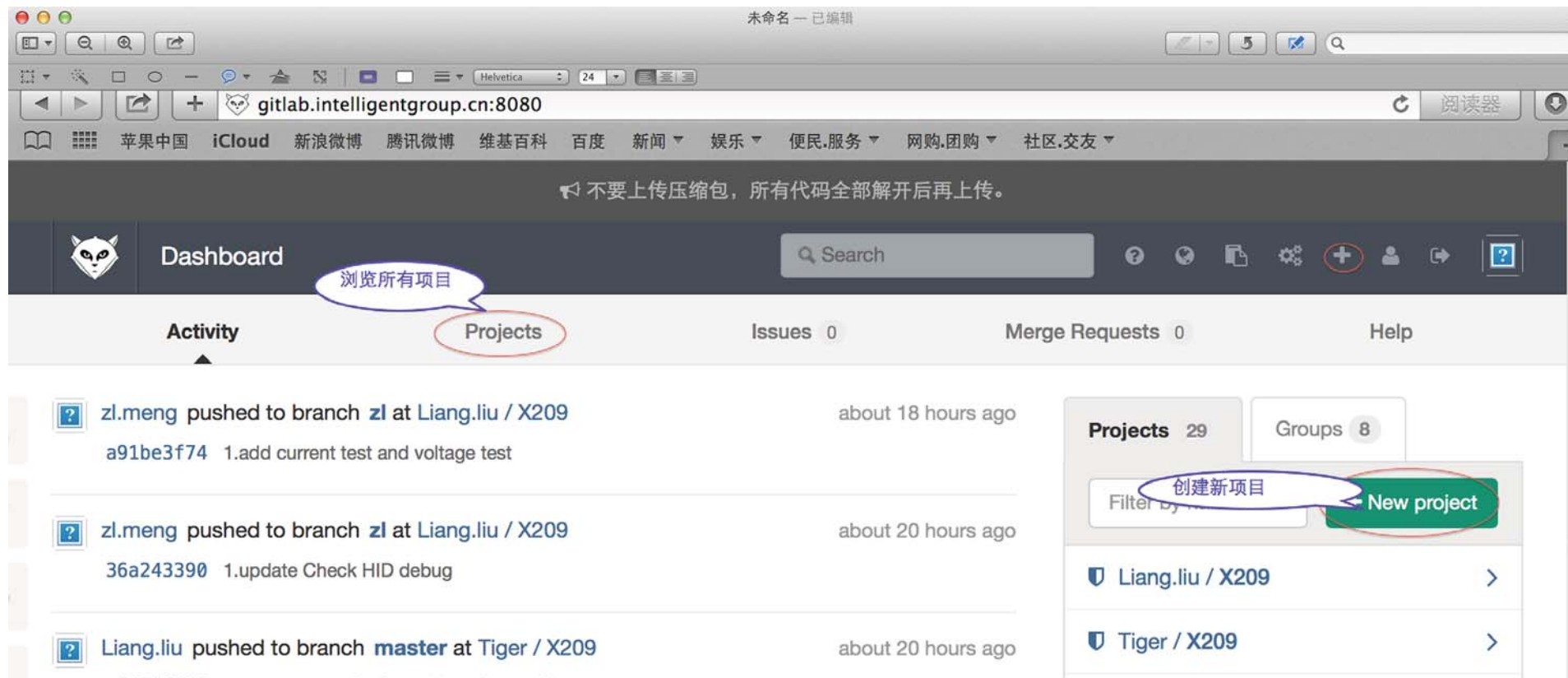
<http://gitlab.intelligentgroup.cn> 或
<http://gitlab.intelligentgroup.cn:8080>



进入登录页面，输入用户名和密码
(如果没有用户名，
请联系我或相关的**leader**)

GIT服务器

完整登录后，可以进行相关项目的浏览，管理，下载等操作。



GIT服务器 --- 创建新项目

要将一个新项目上出到**GIT**服务器，需要现在服务器上建立一个空项目
点击页面右上角的”+”进入创建项目页面：

未命名3 — 已编辑

gitlab.intelligentgroup.cn/projects/new

不要上传压缩包，所有代码全部解开后再上传。

New Project

Project name: TestPri 项目名称

Namespace: Ryan 项目组

☐ Customize repository name?

☐ Import existing repository?

Description (optional): This is a Test Project. 项目描述

Visibility Level (?)

- ☒ Private
Project access must be granted explicitly for each user.
- ☐ Internal
The project can be cloned by any logged in user.
- ☐ Public
The project can be cloned without any authentication.

可见性，默认选Private即可

- 1.Private: 只有设置了权限的人才可以尽心相关操作和浏览
- 2.Internal: 登录的用户才可以进行浏览
- 3.Public: 公开的，即使没有登陆也可以浏览，建议不要设置此选项

Create project 创建项目

Need a group for several dependent projects? Create a group

GIT服务器 --- 创建新项目

点击

Create project

进入项目初始化页面，并拷贝右上的项目连接地址

The screenshot shows the GitLab web interface for a new project named 'Ryan / TestPrj'. The browser address bar shows 'gitlab.intelligentgroup.cn/Ryan/testprj'. The page has a dark header with the project name and a search bar. Below the header, there are tabs for 'Activity', 'Issues 0', 'Merge Requests 0', 'Wiki', and 'Settings'. The main content area shows the project name 'Ryan / TestPrj' with a lock icon. To the right, there are buttons for 'SSH' and 'HTTP', followed by the project URL 'http://192.168.16.10/Ryan/testprj.git' which is circled in red, and a label '项目地址'. Below this, there is a section 'Git global setup:' with a code block containing the commands:

```
git config --global user.name "Ryan"
git config --global user.email "ryan.ruan@intelligentgroup.cn"
```

. This is followed by a section 'Create Repository' with a code block containing the commands:

```
mkdir testprj
cd testprj
git init
touch README
git add README
git commit -m 'first commit'
git remote add origin http://192.168.16.10/Ryan/testprj.git
git push -u origin master
```

. Below that is a section 'Existing Git Repo?' with a code block containing the commands:

```
cd existing_git_repo
git remote add origin http://192.168.16.10/Ryan/testprj.git
git push -u origin master
```

. At the bottom right, there is a red button labeled 'Remove project'.

Activity Issues 0 Merge Requests 0 Wiki Settings

Ryan / TestPrj

SSH HTTP **http://192.168.16.10/Ryan/testprj.git** 项目地址

This is a Test Project. – Edit

Git global setup:

```
git config --global user.name "Ryan"
git config --global user.email "ryan.ruan@intelligentgroup.cn"
```

Create Repository

```
mkdir testprj
cd testprj
git init
touch README
git add README
git commit -m 'first commit'
git remote add origin http://192.168.16.10/Ryan/testprj.git
git push -u origin master
```

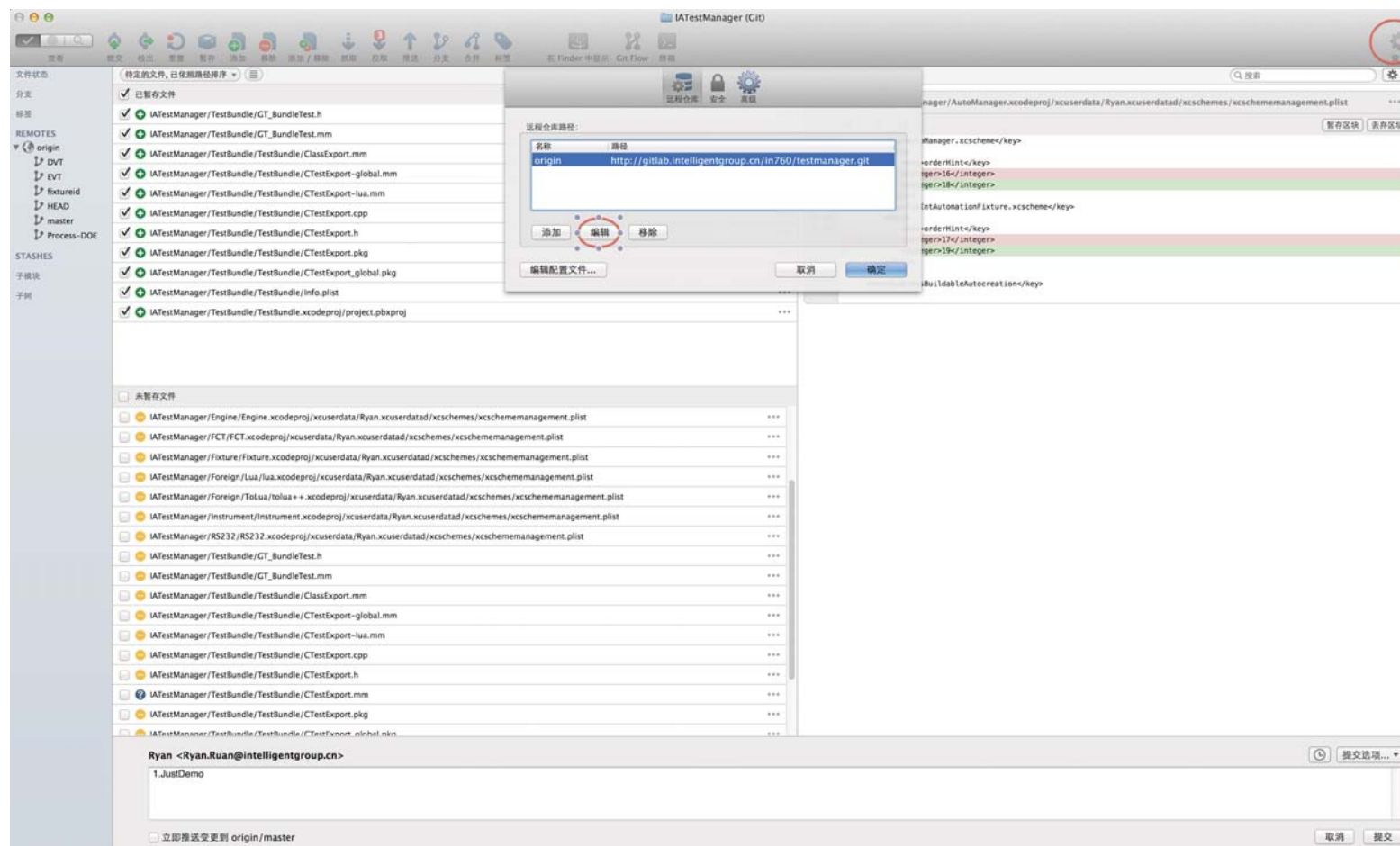
Existing Git Repo?

```
cd existing_git_repo
git remote add origin http://192.168.16.10/Ryan/testprj.git
git push -u origin master
```

Remove project

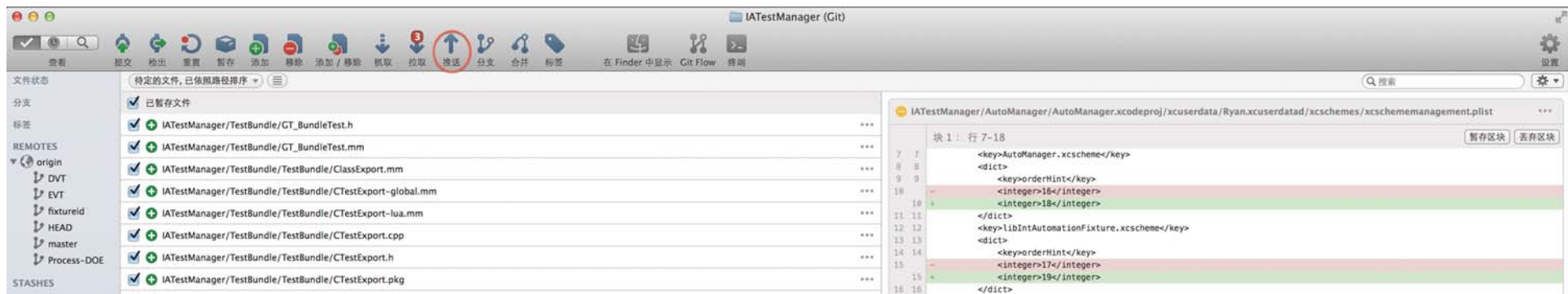
GIT服务器 --- 绑定新项目

在**SourceTree**里修改设置，绑定本地项目与服务器项目
依次点击设置→编辑，再将刚拷贝的项目地址填入



GIT服务器 --- 上传新项目

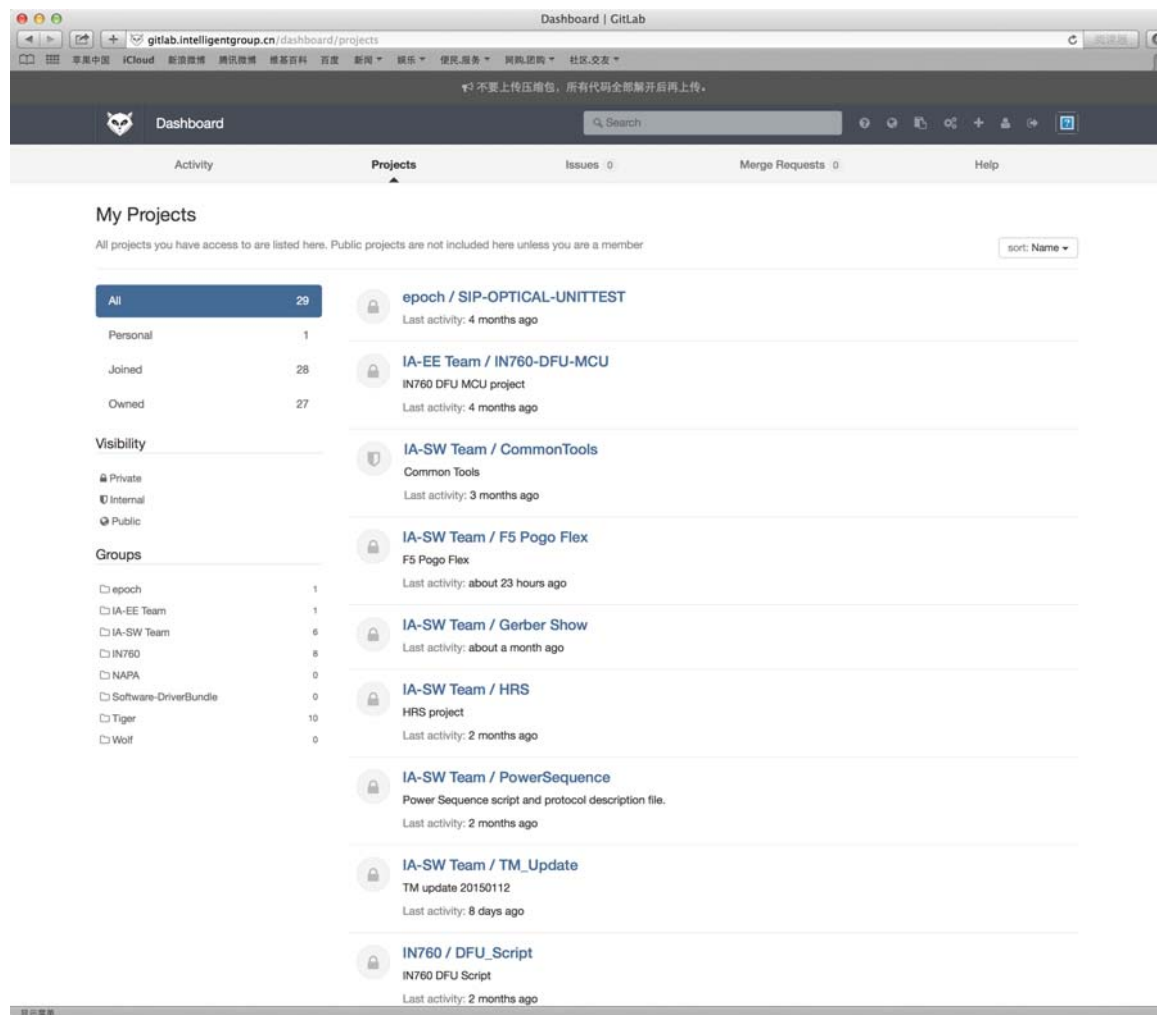
点击推送按钮，完成本地项目到远程项目的上传



推送完成后，刷新远程服务器，将能看到页面被更新。

GIT服务器 --- 浏览服务器项目

点击**Projects**浏览所有项目：

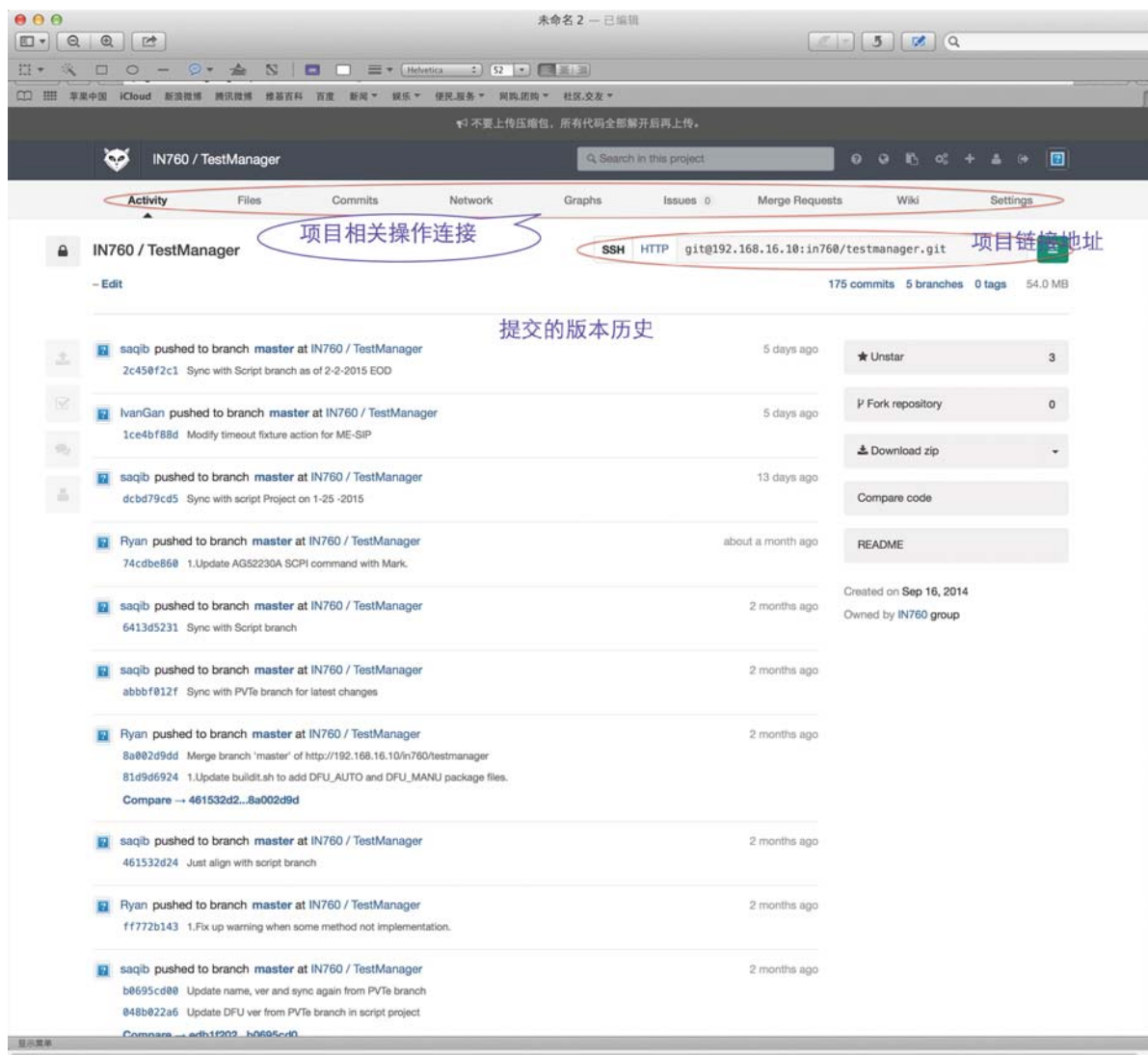


GIT服务器 --- 管理服务器项目

点击需要编辑项目，进入项目管理页面：

可以进行如下操作：

1. 了解所有开发人员的提交过程
2. 点击**File**查看各个分支的文件
3. 点击**Commit**查看某个分支的更新履历
4. 点击**Setting**可以对项目的人员，权限等进行管理



除去**SourceTree**,你也可以通过**GIT**命令直接操作**GIT**，来对项目进行管理，常用**GIT**命令有如下：

查看、添加、提交、删除、找回，重置修改文件

<code>git show</code>	# 显示某次提交的内容 <code>git show \$id</code>
<code>git co -- <file></code>	# 抛弃工作区修改
<code>git co .</code>	# 抛弃工作区修改
<code>git add <file></code>	# 将工作文件修改提交到本地暂存区
<code>git add .</code>	# 将所有修改过的工作文件提交暂存区
<code>git rm <file></code>	# 从版本库中删除文件
<code>git rm <file> --cached</code>	# 从版本库中删除文件，但不删除文件
<code>git reset <file></code>	# 从暂存区恢复到工作文件
<code>git reset -- .</code>	# 从暂存区恢复到工作文件
<code>git reset --hard</code>	# 恢复最近一次提交过的状态，即放弃上次提交后的所有本次修改
<code>git ci <file> git ci . git ci -a</code>	# 将 <code>git add</code> , <code>git rm</code> 和 <code>git ci</code> 等操作都合并在一起做
<code>git ci -am "some comments"</code>	
<code>git ci --amend</code>	# 修改最后一次提交记录
<code>git revert <\$id></code>	# 恢复某次提交的状态，恢复动作本身也创建次提交对象
<code>git revert HEAD</code>	# 恢复最后一次提交的状态

查看文件diff

<code>git diff <file></code>	# 比较当前文件和暂存区文件差异 <code>git diff</code>
<code>git diff <\$id1> <\$id2></code>	# 比较两次提交之间的差异
<code>git diff <branch1>..<branch2></code>	# 在两个分支之间比较
<code>git diff --staged</code>	# 比较暂存区和版本库差异
<code>git diff --cached</code>	# 比较暂存区和版本库差异
<code>git diff --stat</code>	# 仅仅比较统计信息

GIT命令

查看提交记录

```
git log git log <file>
git log -p <file>
git log -p -2
git log --stat
```

```
# 查看该文件每次提交记录
# 查看每次详细修改内容的diff
# 查看最近两次详细修改内容的diff
# 查看提交统计信息
```

查看、切换、创建和删除分支

```
git br -r
git br <new_branch>
git br -v
git br --merged
git br --no-merged
git co <branch>
git co -b <new_branch>
git co -b <new_branch> <branch>
git co $id
git co $id -b <new_branch>
git br -d <branch>
git br -D <branch>
```

```
# 查看远程分支
# 创建新的分支
# 查看各个分支最后提交信息
# 查看已经被合并到当前分支的分支
# 查看尚未被合并到当前分支的分支
# 切换到某个分支
# 创建新的分支，并且切换过去
# 基于branch创建新的new_branch
# 把某次历史提交记录checkout出来，但无分支信息，切换到其他分支会自动删除
# 把某次历史提交记录checkout出来，创建成一个分支
# 删除某个分支
# 强制删除某个分支 (未被合并的分支被删除的时候需要强制)
```

分支合并和rebase

```
git merge <branch>
git merge origin/master --no-ff
git rebase master <branch>
```

```
# 将branch分支合并到当前分支
# 不要Fast-Foward合并，这样可以生成merge提交
# 将master rebase到branch，相当于：
git co <branch> && git rebase master && git co master && git merge <branch>
```


Thank You

for the opportunity...

