**代码下载／提交使用说明书V0.8**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 序号 | 版本 | 时间 | 撰写人 | 审核人 | 说明 | 其他 |
| １ | ０．８ | ２０１６－０７－２８ | 刘畅 | 陈刚 | 代码下载／提交 |  |
|  |  |  |  |  |  |  |

## 一、工作常用的几条命令

１．git命令三部曲

１.初始化并克隆manifest

repo init -u git://192.168.0.185/lollipop-x86/manifest.git -b multiwindow

2.下载代码（同步代码）

repo sync

3.创建并切换分支

repo start multiwindow --all

## 二、git来龙去脉

　　什么是版本库呢？版本库又名仓库，英文名repository，我们可以简单理解成一个目录，这个目录里面的所有文件都可以被Git管理起来，每个文件的修改、删除，Git都能跟踪，以便任何时刻都可以追踪历史，或者在将来某个时刻可以“还原”。

git是版本管理工具

## 三、repo与git的关系

repo来自英文：仓库、版本库：Repository

Android使用git作为代码管理工具，开发了Gerrit进行代码审核以便更好的对代码进行集中式管理，还开发了r**epo**命令行工具，对Git部分命令封装，将百多个git库有效的进行组织。

　简而言之，repo是Google公司，在Android项目，对git命令封装，或者git命令简化。

**四、命令详解**

命令１：

repo init -u git://192.168.0.185/lollipop-x86/manifest.git -b multiwindow

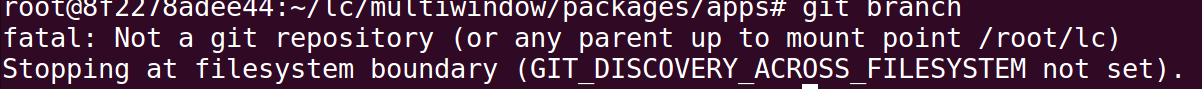
-b:选择一个maniest仓库中的一个特殊的分支multiwindow，也就是multiwindow这个分支

命令２：

repo start multiwindow --all

创建multiwindow分支，并且切换到multiwindow分支来工作。

**五、为什么不同的目录，有的有git仓库，有的没有git仓库，这么诡异？**





问题的关键在于：执行如下命令：**repo start multiwindow --all**

repo start实际是对git checkout –b 命令的封装。为指定的项目或所有项目（若使用—all参数），以清单文件中为设定的分支，创建特性分支。

这条指令与git checkout –b 还是有很大的区别的，git checkout –b 是在当前所在的分支的基础上创建特性分支，而repo start是在清单文件设定分支的基础上创建特性分支。

也就是说：**repo start multiwindow --all**

**创建的分支，建立的git 仓库，目录是特定的**

**这也就是解释了为什么：**

**packages/apps没有仓库；**

**但是packages/apps/Settings是有仓库的。**

**五、multiwindow理解**

命令一：

repo init -u git://192.168.0.185/lollipop-x86/manifest.git -b **multiwindow**

命令二：

repo start **multiwindow** --all

命令一：**multiwindow是说获取服务器**manifest.git 主干下的一个分支**multiwindow下代码**

命令二：**multiwindow是说在本地创建一个multiwindow分支，也就是在本地创建一个multiwindow仓库**

**六、如何提交代码，如何打补丁？**

**提交代码７大步骤**

**第１步：**创建本地**multiwindow**分支子分支lc

**git checkout -b lc**

**第２步：**创建补丁，打补丁

**git apply diff.patch**

**第３步：**提交补丁到本地分支lc

**git commit -a -s**

**第４步：**格式化输出补丁至头节点HEAD

**git format-patch -M HEAD^**

**第５步：**从子分支lc切换到本地分支multiwindow

**git checkout multiwindow**

**第６步：**

**把代码上传到远程服务器的** multiwindow分支上

**git push devorg multiwindow:refs/heads/multiwindow**

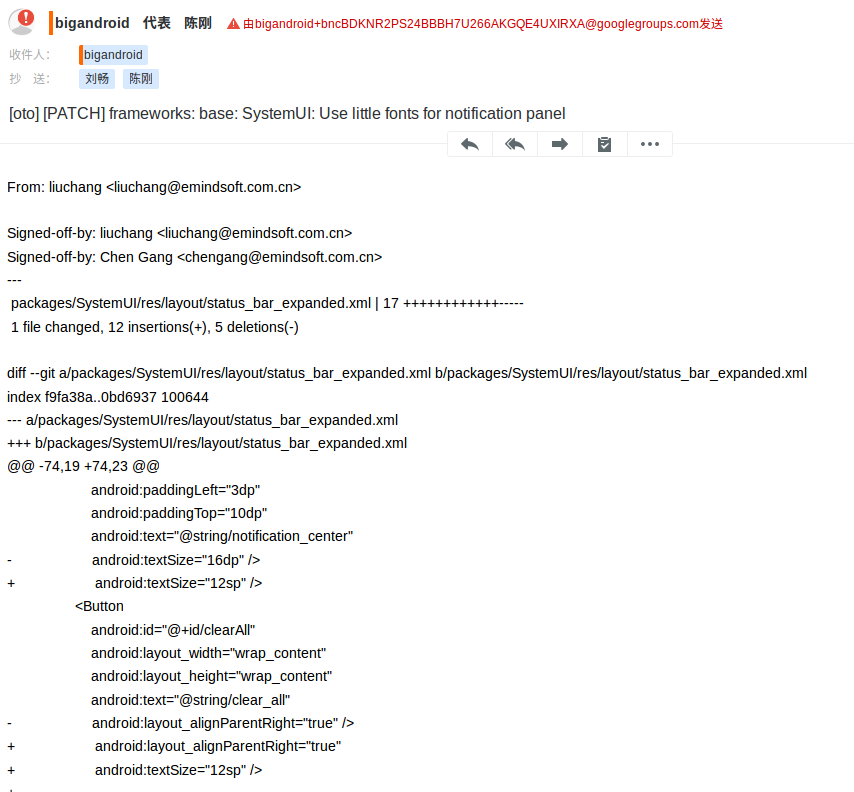
把本地分支**multiwindow**代码，也就是头heads的代码，提交到服务器的**multiwindow**的分支上**。**

**第７步：**

最后一步：发邮件

git send-email --To bigandroid@googlegroups.com 0001-frameworks-base-SystemUI-Use-little-fonts-for-notifi.patch

就是大家在邮箱里，看到的这份邮件了。



**ＰＳ**

**代码分析的时候，经常使用的命令：**

git log

git log -p

git checkout

更多的参数，可以linux 命令上git --help 或者 网上查询。