Git

一、Git的诞生?

很多人都知道，Linus在1991年创建了开源的Linux，从此，Linux系统不断发展，已经成为最大的服务器系统软件了。

Linus虽然创建了Linux，但Linux的壮大是靠全世界热心的志愿者参与的，这么多人在世界各地为Linux编写代码，那Linux的代码是如何管理的呢?

事实是，在2002年以前，世界各地的志愿者把源代码文件通过diff的方式发给Linus，然后由Linus本人通过手工方式合并代码!

你也许会想，为什么Linus不把Linux代码放到版本控制系统里呢？不是有CVS、SVN这些免费的版本控制系统吗？因为Linus坚定地反对CVS和SVN，这些集中式的版本控制系统不但速度慢，而且必须联网才能使用。有一些商用的版本控制系统，虽然比CVS、SVN好用，但那是付费的，和Linux的开源精神不符。

不过，到了2002年，Linux系统已经发展了十年了，代码库之大让Linus很难继续通过手工方式管理了，社区的弟兄们也对这种方式表达了强烈不满，于是Linus选择了一个商业的版本控制系统BitKeeper，BitKeeper的东家BitMover公司出于人道主义精神，授权Linux社区免费使用这个版本控制系统。

安定团结的大好局面在2005年就被打破了，原因是Linux社区牛人聚集，不免沾染了一些梁山好汉的江湖习气。开发Samba的Andrew试图破解BitKeeper的协议（这么干的其实也不只他一个），被BitMover公司发现了（监控工作做得不错！），于是BitMover公司怒了，要收回Linux社区的免费使用权。

Linus可以向BitMover公司道个歉，保证以后严格管教弟兄们，嗯，这是不可能的。实际情况是这样的：

Linus花了两周时间自己用C写了一个分布式版本控制系统，这就是Git！一个月之内，Linux系统的源码已经由Git管理了！牛是怎么定义的呢？大家可以体会一下。

Git迅速成为最流行的分布式版本控制系统，尤其是2008年，GitHub网站上线了，它为开源项目免费提供Git存储，无数开源项目开始迁移至GitHub，包括jQuery，PHP，Ruby等等。

历史就是这么偶然，如果不是当年BitMover公司威胁Linux社区，可能现在我们就没有免费而超级好用的Git了

- 集中式vs分布式

Linus一直痛恨的CVS及SVN都是集中式的版本控制系统，而Git是分布式版本控制系统，集中式和分布式版本控制系统有什么区别呢?

先说集中式版本控制系统，版本库是集中存放在中央服务器的，而干活的时候，用的都是自己的电脑，所以要先从中央服务器取得最新的版本，然后开始干活，干完活了，再把自己的活推送给中央服务器。中央服务器就好比是一个图书馆，你要改一本书，必须先从图书馆借出来，然后回到家自己改，改完了，再放回图书馆。

二、什么是GIT?

> Git 是一个 **免费的开源** 分布式版本控制系统，旨在快速高效地处理从小到大的所有项目。
>
> Git **易于学习**， **占用空间小，性能快如闪电** 。它优于 Subversion、CVS、Perforce 和 ClearCase 等 SCM 工具，具有 **便宜的本地分支** 、方便的 **暂存区** 和 **多个工作流等功能** 。

三、下载GIT

- 官网地址: **https://git-scm.com/**
- 安装文档教程:
  - **https://blog.csdn.net/ichen820/article/details/113891601?utm_term=git%E5%AE%89%E8%A3%85%E6%95%99%E7%A8%8B%E6%9C%80%E6%96%B0%E7%89%88%E6%9C%AC&utm_medium=distribute.pc_aggpage_search_result.none-task-blog-2~all~sobaiduweb~default-1-113891601&spm=3001.4430**

四、GIT工作区域介绍

git 记录文件内容的变化

git status 查看仓库的状态

git init 将工作区变成本地git仓库。只有变成git的本地仓库我们才能使用git的命令 将工作区的内容提交到暂存区,再将暂存区的内容提交到本地仓库

工作区 git add 文件名称/文件夹名称/. (这个点表示该文件夹内的所有文件) 暂存区 git commit -m "" 本地仓库

如果将工作区的内容提交到本地仓库,那么git就可以记录文件内容的变化

- 设置全局的用户名和邮箱

  在任何项目内都可以使用设置的全局的用户名和邮箱

```
//设置的全局用户名
git config --global user.name 用户名(英文)

//设置的全局的邮箱地址
git config --global user.email 邮箱地址
```

- 设置局部的用户名和邮箱

只能在当前的项目内用当前设置的用户名和邮箱

```
//设置的全局用户名
git config  user.name 用户名(英文)

//设置的全局的邮箱地址
git config  user.email 邮箱地址
```

全局:

任意地方都可以访问

局部:

在一定的范围内可以访问

五、操作流程

1. 鼠标右键 -》 选择git bash here ->. 显示小弹窗
2. 在桌面创建一个文件夹 ,英文名称
3. cd 进入到刚才创建的文件夹
4. 把当前的文件夹变成git仓库 ,只有变成git仓库,git才能记录文件内容的变化

```
git init
```

5. 创建一个文件,写上内容
6. 将文件提交到暂存区

```
git add 文件名/.
```

7. 将暂存区的内容提交到本地仓库

```
git commit -m "这次提交的描述"
```

8. 查看提交的历史

```
git log
```

六、总结

Git命令

1. 初始化git本地仓库

```
git init
```

2. 设置签名信息(用户名和邮箱)
   - 全局

```
//设置用户名
git config --global user.name 用户名(建议是英文)

//设置邮箱
git config --global user.email 邮箱
```

   - 局部

```
//设置用户名
git config user.name 用户名(建议是英文)

//设置邮箱
git config user.email 邮箱
```

   - 查看设置的用户名和邮箱
     - 第一种
       - 输入 `git config --global user.name`
       - 输入 `git config --global user.email`
     - 第二种
       - cat ~/.gitconfig
3. 查看仓库的状态

```
git status
```

   - 如果显示文件是红色,则表示没有提交到暂存区
   - 如果显示的是绿色,则表示提交到暂存区,但是没有提交到本地仓库
4. 将工作区的内容提交到暂存区

```
git add 文件名 或者 文件夹名称 或者 . (点表示当前文件夹内的所有文件)
```

5. 将暂存区的内容提交到本地仓库

```
git commit -m "描述"
```

6. 查看提交的历史记录

```
git log
```

7. 查看回退的记录

```
git reflog
```

8. 版本前进或者会退

- 索引值(哈希值)

```
git reset --hard 某个版本的索引值(哈希值)
```

- ^ (异或)符号

```
git reset --hard HEAD^ (一个^表示回退一步)
```

- ~符号

```
git reset --hard HEAD~n (n表示回退的步数)
```

Linux命令

1. 进入到某个文件夹

```
cd 文件夹名称
```

2. 查看当前文件夹内的文件

```
ls
```

3. 查看当前当前文件夹内的文件以及隐藏文件

```
ls -a
```

4. 创建一个文件夹

```
mkdir 文件夹名称
```

5. 创建一个文件

```
touch 文件名称
```

6. 编辑一个文件
   - 使用编辑器
   - 使用git内置的vim编辑器

```
vim 文件名称

i //可以输入内容

按esc键  然后输入:wq 表示保存并退出
```

7. 查看文件的内容

```
cat 文件名
```