Git常用命令

• 1 `git init`

1. 作用

   创建一个本地版本库,让git能够管理本地目录

2. 执行完成 `git init` 之后，本地目录变化
   - 在当前目录下创建一个 `.git` 的隐藏目录；
   - 这个 `.git` 隐藏目录就相当于本地版本库；
   - 默认创建了一个 `master` 分支（主干）；

• 2 `git status`

1. 作用

   查看版本库文件状态

2. 文件的颜色变化

   红色：未添加到暂存区的文件

   哪些文件是红色的？？

   1. 新增加文件
   2. 修改的文件（修改已经被git管理过的文件）

   执行 `git add 文件` ，可以将**红色**变成**绿色**

   绿色：添加到暂存区的文件，没有提交到版本库

   执行 `git commit -m "提交日志" 文件` ，将文件添加到本地版本库

• 3 `git add`

1. 作用

将文件添加到暂存区

2. 常用方式
    1. 将单个文件添加到暂存区

        `git add 文件名`

    2. 将所有未添加到暂存区的文件添加到暂存区

        `git add .`

        `git add *`

        `git add -A`

4 `git commit`

1. 作用

    将暂存区的文件提交到本地版本库

2. 常用方式

    `git commit -m "msg" 文件`

    如果后边不加具体文件，表示将暂存区所有文件提交到版本库

    `git commit -am "msg" 文件`

    可以将修改的版本库文件直接提交到本地版本库

5 `git config`

1. 作用

    配置git配置文件

2. Git的三类配置
    1. 本地版本库配置
    2. 用户级别配置（全局配置 - 所有本地版本库配置）
    3. 系统级别配置

配置级别	命令	配置文件	优先级
本地版本库	git config --local	.git/config	最高
用户级别	git config --global	~/.gitconfig	其次
系统配置	git config --system	安装目录/etc/gitconfig	最低

5.1 添加配置方式

```
git config --local user.name 'et'
```

```
git config --local user.email 'et@et.com'
```

注意：`--local` 可以替换成 `--global` 或者 `--system`

5.2 查看配置

- `git config --local --list`
- `git config --local -l`

5.3 删除配置 - 了解

- `git config --local --unset user.name`

5.4 修改配置 - 了解

- `git config --local --replace-all user.name 'etoak'`

6 `git log`

1. 作用

   查看提交的历史记录

2. 常用方式

1. `git log`

   该命令可以输出commit hash值、作者、时间、提交message

2. `git log --oneline`

   仅输出commit hash值和提交message

```
lenovo@lenovo-PC MINGW64 /d/git-course/emp (master)
$ git log --oneline
c74a43c (HEAD -> master) 修改Hello.java
0831bb8 add Hello.java
```

3. `git log --graph`

   以点线图展示提交历史记录

```
$ git log --graph
* commit 7895ce11fc2b9c001881b5715c07a26ec322ec17 (HEAD -> master,
| Author: zs <zs@et.com>
| Date:   Sat Jan 23 16:05:02 2021 +0800
|
|     add B.java
|
* commit 3c24d118dec37c0955f267850a9e4ec0c488e8bf
  Author: zs <zs@et.com>
  Date:   Sat Jan 23 15:44:26 2021 +0800

      add A.java Hello.java
```

4. `git log --pretty=format:'%h %s'`

   控制显示记录格式

   常用格式占位符写法

```
%H：提交对象(commit)的完整hash值
%h：提交对象的缩略hash值(前7位-能够唯一确定一个提交对象)
%s：显示提交message
%an：显示作者
%ae：显示邮件地址
%cd：显示提交日期
```

- 7 `git reset`

- 7.1 作用

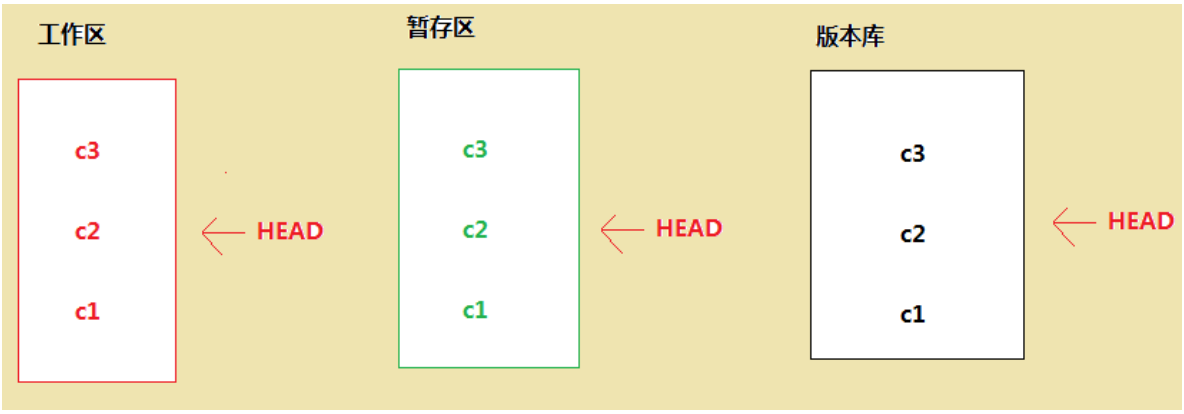
将版本库内容重置到某个commit

- 7.2 `git reset --hard <commit>`

重置本地版本库

重置暂存区

重置工作区（ 比较危险 ）

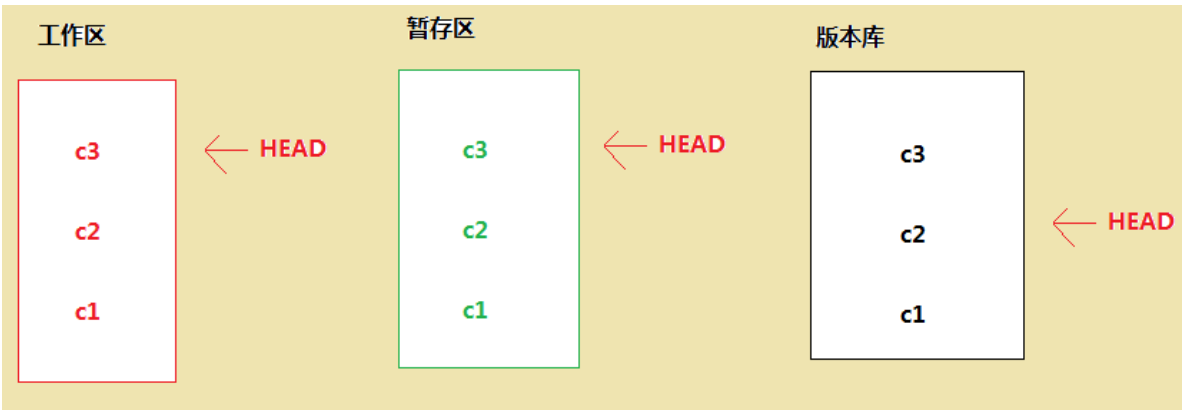


- 7.3 `git reset --soft <commit>`

重置版本库

不重置暂存区

不重置工作区

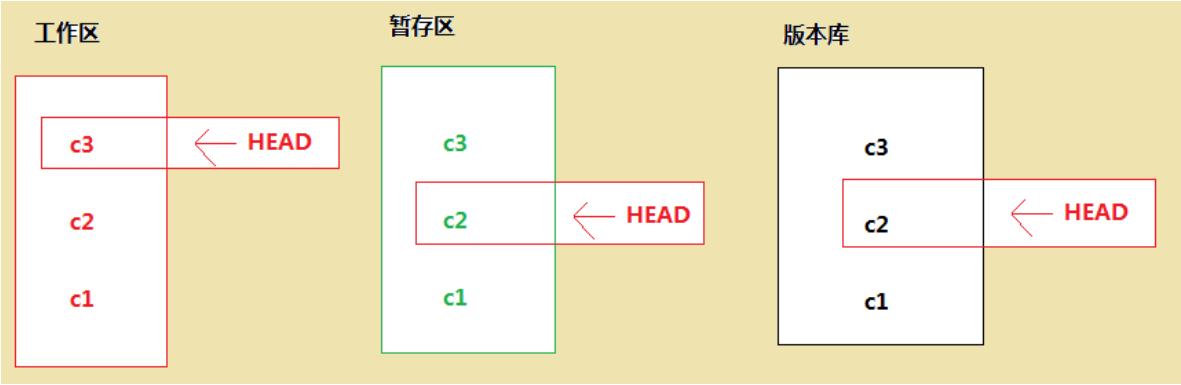


- 7.4 `git reset --mixed <commit>`

重置版本库

重置暂存区

不重置工作区



• 8 `git reflog`

- 作用

  可以查看所有分支的所有操作记录（包括已经被删除的commit记录和reset的操作)

• 9 `git diff`

- 作用

  比较文件差异

- 9.1 比较工作区和暂存区差异

```
git diff <file..>
```

- 9.2 比较工作区和版本库差异

```
git diff HEAD <文件..>
```

如果最后不指定文件，表示比较所有文件差异

- 9.3 比较暂存区和版本库差异

```
git diff --cached <文件..>
```

```
git diff --staged <文件..>
```

10 撤销暂存区修改

```
git reset HEAD 文件
```

或者

```
git restore --staged 文件
```

注意：`git restore` 是git 2.23版本之后出现的命令

- **帮助理解**

  在执行上面命令之后，

  使用 `git status` 查看状态，

  文件的颜色由**绿色**变为**红色**

11 撤销工作区修改

```
git checkout 文件
```

或者

```
git restore 文件
```

12 `git revert`

- 作用

  撤销某个commit

  在执行撤销之后，并没有删除这个commit，而是git会在版本库重新创建一个commit；

- **13 `git mv`**
  - 修改git版本库中的文件名称

    ```
    git mv A.java B.java
    ```

- **14 `git rm`**
  - 删除版本库中的文件

    ```
    git rm B.java
    ```