**Git原理 - Git对象[commit对象]**

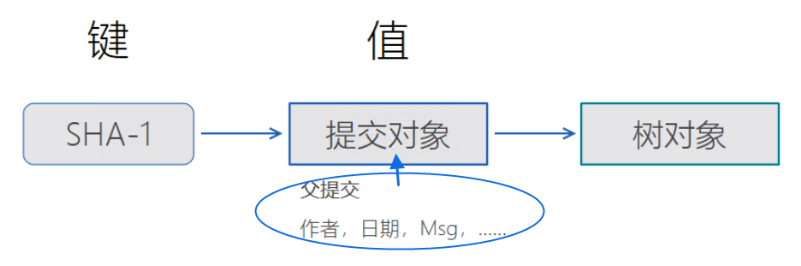
**1、Commit对象介绍**

现在来介绍最后一种Git对象commit对象，也叫提交对象。

提交对象可以理解为是对树对象的一层封装，提交信息包括基于当前暂存区中索引文件生成的tree对象，还有包含了提交时间，提交者信息，作者信息，以及提交备注等内容，更重要的是里面还包含了父提交的ID，由此就可以形成Git提交的有向无环图。（是链式的关系，把所有commit对象关联起来）

即：commit对象通常指向一个 tree 对象，并且封装了文件的提交时间，提交者信息，作者信息，提交备注，以及父提交引用等数据。

下面是commit对象的存储结构



**2、Commit对象说明**

我们通过练习来说明commit对象，接着用前面Tree对象的本地版本库。

**（1）创建一个commit对象**

我们可以通过调用commit-tree命令创建一个提交对象，为此需要指定一个树对象的SHA-1值，以及该提交的父提交对象。

|  |
| --- |
| 说明：使用commit-tree命令来创建提交对象，一般都需要和父提交进行关联，如果是第一次将暂存区的文件索引数据提交到本地版本库，那么该提交操作就不需要指定父提交对象。 |

**1）我们可以先查看一下此时Git本地库中的对象，如下**：

|  |
| --- |
| Bash .git/objects/01/ab2a43b1eb150bcf00f375800727df240cf653 # 第三个tree树对象 .git/objects/0c/1e7391ca4e59584f8b773ecdbbb9467eba1547 # test.txt第二个版本（blob对象） .git/objects/16/3b45f0a0925b0655da232ea8a4188ccec615f5 # 第二个tree树对象 .git/objects/83/baae61804e65cc73a7201a7252750c76066a30 # test.txt第一个版本（blob对象） .git/objects/d8/329fc1cc938780ffdd9f94e0d364e0ea74f579 # 第一个tree树对象 .git/objects/fa/49b077972391ad58037050f2a75f74e3671e92 # new.txt第一个版本（blob对象） |

**2）我们通过第一个树对象，创建一个commit对象**

|  |
| --- |
| Bash # 1.做提交操作，创建一个commit对象 L@DESKTOP-T2AI2SU MINGW64 /j/git-repository/git\_learning (master) $ echo 'first commit' | git commit-tree d8329f 3ceba95d3cd9cce982d31e41e3b995ece72f755d  # 2.确定该对象类型 L@DESKTOP-T2AI2SU MINGW64 /j/git-repository/git\_learning (master) $ git cat-file -t 3ceba95d3c commit  # 3.查看该对象内容 L@DESKTOP-T2AI2SU MINGW64 /j/git-repository/git\_learning (master) $ git cat-file -p 3ceba95d3c tree d8329fc1cc938780ffdd9f94e0d364e0ea74f579 author sun\_wk <sun\_wk@126.com> 1618190880 +0800 committer sun\_wk <sun\_wk@126.com> 1618190880 +0800  first commit |

说明：

* tree：表示该commit对象所指向的tree对象的索引
* author：表示该文件的作者。
* committer：表示该文件的提交者。
* first commit：这段文本是提交备注。（备注与前面留空一行）
* 因为是第一次进行commit提交操作，所以没有父提交信息。
* 1618190880 +0800：表示时间，一个时间戳。

|  |
| --- |
| 即：commit对象的格式很简单：指明了该时间点项目快照的顶层树对象、作者/提交者信息（从 Git 设置的 user.name和 user.email中获得)，以及当前时间戳、留空一行，最后是提交注释。  提示：git commit-tree命令不但生成了提交对象，而且会将对应的快照（树对象）提交到本地库中。 |

**（2）创建第二个commit对象**

根据第二个tree对象和第一个commit对象，来创建第二个commit对象。

通过-p选项指定父提交对象。

|  |
| --- |
| Bash # 1.创建第二个commit对象 L@DESKTOP-T2AI2SU MINGW64 /j/git-repository/git\_learning (master) $ echo 'second commit' | git commit-tree 163b45f0a09 -p 3ceba95d3cd9cc 60e1c209e9de87314ec47cf28e61de8df5362fe6  # 2.查看该对象内容 L@DESKTOP-T2AI2SU MINGW64 /j/git-repository/git\_learning (master) $ git cat-file -p 60e1c209e9de8 tree 163b45f0a0925b0655da232ea8a4188ccec615f5 parent 3ceba95d3cd9cce982d31e41e3b995ece72f755d author sun\_wk <sun\_wk@126.com> 1618193286 +0800 committer sun\_wk <sun\_wk@126.com> 1618193286 +0800  second commit |

|  |
| --- |
| **提交对象的格式很简单：**  它先指定一个顶层树对象，代表当前项目快照；  然后是可能存在的父提交；  之后是作者/提交者信息（依据你的 user.name 和 user.email 配置来设定，外加一个时间戳）；  留空一行，最后是提交注释。 |

第三个commit提交，同上，这里就不演示了。

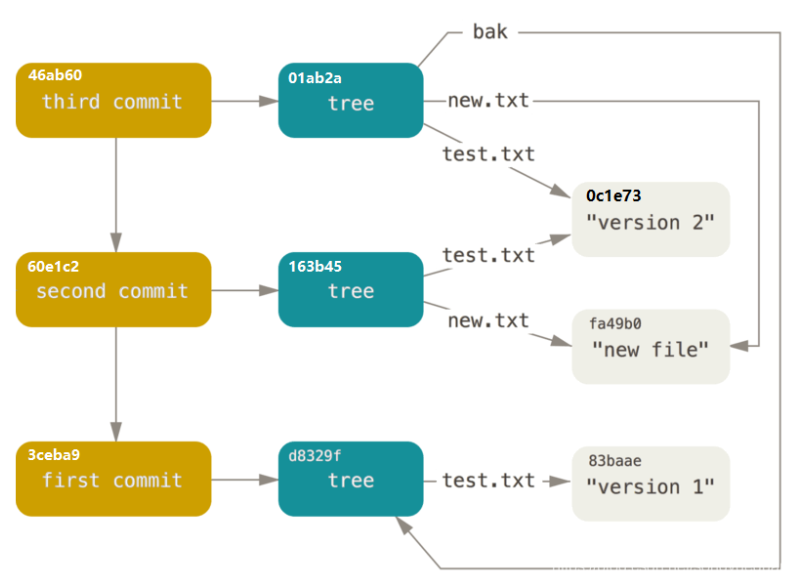
**3、本地库中对象之间的关系**

我们可以查看一下此时Git本地库中的对象

|  |
| --- |
| Bash .git/objects/01/ab2a43b1eb150bcf00f375800727df240cf653 # 第三个tree树对象 .git/objects/0c/1e7391ca4e59584f8b773ecdbbb9467eba1547 # test.txt第二个版本（blob对象） .git/objects/16/3b45f0a0925b0655da232ea8a4188ccec615f5 # 第二个tree树对象 .git/objects/3c/eba95d3cd9cce982d31e41e3b995ece72f755d # 第一个commit提交对象 .git/objects/46/ab608799a0e65e970b67b9b52f6c1407c39036 # 第三个commit提交对象 .git/objects/60/e1c209e9de87314ec47cf28e61de8df5362fe6 # 第二个commit提交对象 .git/objects/83/baae61804e65cc73a7201a7252750c76066a30 # test.txt第一个版本（blob对象） .git/objects/d8/329fc1cc938780ffdd9f94e0d364e0ea74f579 # 第一个tree树对象 .git/objects/fa/49b077972391ad58037050f2a75f74e3671e92 # new.txt第一个版本（blob对象） |

可以从上面看到，此时的本地版本库中共有9个对象，三个blob对象，三个tree对象，三个commit对象。

他们之间的关系如下图：



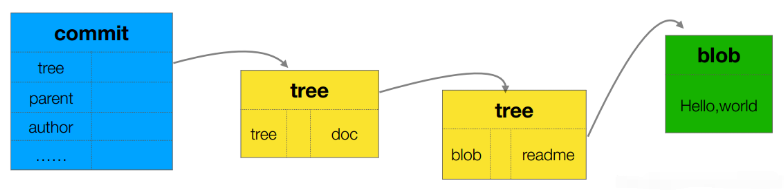
**4、总结**

1. 提交是我们经常使用的Git动作，每次提交操作都指向一个树对象，同时会产生一个commit对象。
2. 即：一个commit对象包含了一个tree对象，这个tree对象记录了在那个时间点，项目包含了什么文件夹和什么文件。
3. 一个提交对象可以有一个或者多个父提交。
4. 每次commit操作都会基于当前索引文件index新建tree对象。那么当前索引文件，是在上次提交的基础上更新来的，所以每次提交产生的commit对象，与其他的commit对象，都有前后关系或者称为父子关系。
5. 对于我们来说，不需要直接访问blob对象和tree对象，我们直接访问commit对象就可以了。
6. 即：commit对象对应的tree对象下面，又包含了小的tree对象和blob对象，子的tree对象一层层展开，最后叶子节点就是一个个blob对象，也就是一个个文件。

|  |
| --- |
| 到这里，我们就能够清楚的了解，什么叫一个Git版本。tree对象才是一次项目版本的快照，提交对象是对tree对象的一次封装。  即：   * 项目的快照就是一个树对象。 * 项目的版本就是一个提交对象。   而且Git的每一个版本，存储的不是增量，而存储的是当前项目的快照。同时objects目录中相当于存放了项目的所有历史记录，回滚就相当的方便了，找到对应的commit对象的hash就可以了。 |

**5、练习**

请问下图中包含多少个tree对象和blob对象？



一共包含两个tree对象，一个blob对象，一个commit对象。

说明：

* 一个commit对象一定对应一个tree对象（这个tree对象应该是一个完整项目仓库的快照）
* doc目录下有一个blob对象，也就是redme文件。

**6、本文用到的命令总结**

Git底层命令：

* git commit-tree：生成一个commit对象。
* git cat-file -t 键：查看Git对象的类型。
* git cat-file -p 键：查看Git对象的内容。

|  |
| --- |
| 参考：   * [https://blog.csdn.net/songyuequan/article/details/85862415](https://links.jianshu.com/go?to=https%3A%2F%2Fblog.csdn.net%2Fsongyuequan%2Farticle%2Fdetails%2F85862415) * [https://www.cnblogs.com/jasongrass/p/10582465.html](https://links.jianshu.com/go?to=https%3A%2F%2Fwww.cnblogs.com%2Fjasongrass%2Fp%2F10582465.html) |

作者：繁华似锦Fighting 链接：https://www.jianshu.com/p/08f45abf508f 来源：简书 著作权归作者所有。商业转载请联系作者获得授权，非商业转载请注明出处。