Web 基础-day01-课堂笔记

1. Git 工具

1.1 【简单了解】Git 功能简介

Git 是一个开源的项目源代码管理工具，其主要功能就是：代码整合和版本控制，可以方便的支持项目开发过程中，多人协同开发及源代码的版本控制。

软件安装(ubuntu):

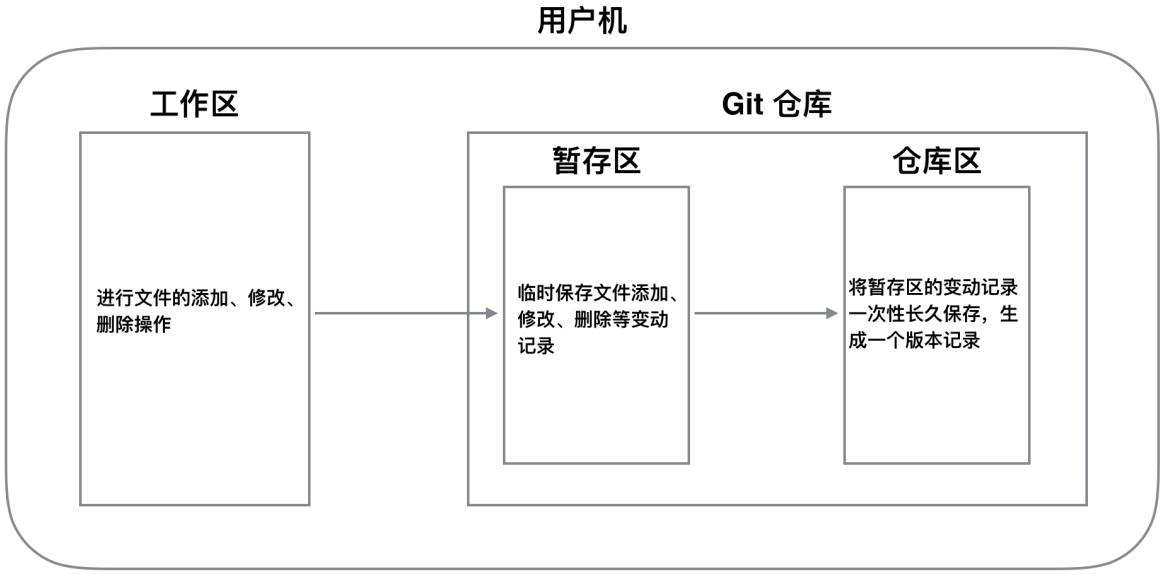
```
apt-get install git
```

1.2 【熟练常握】Git 仓库创建

Git 仓库创建命令：

```
git init
```

1.3 【基本理解】Git 仓库的三个区域



1）工作区(workspace)

Git 中的工作区指的就是我们平时写代码的区域，其实就是项目目录下除 .git 目录以外的地方。工作区的主要作用就是平时开发时对文件进行添加、修改、删除等操作。

2）暂存区(index)

Git 中临时存储文件内容变动的一个区域，我们称为暂存区。这个区域我们一般看不到，但它实际存在。我们可以把文件和代码变动记录临时保存在这里，也可以撤销。主要是为了让用户能够把内容变动进行长久存储前，可以进行筛查、反悔等操作而创建的区域。

3）仓库区(Repository)

Git 中本地进行长久存储内容变动的一个区域。主要用于长久存储一个小阶段的成果，防止丢失。我们一般把暂存区的内容存储到这里。暂存区的内容存储到本地仓库区后，暂存区的内容将会销毁。往仓库区的每一次存储称为一次提交(commit)，即生成了一个内容变动版本记录，仓库区保存可以多个版本记录的信息，并且可以在不同的版本中进行切换。

1.4 【熟练常握】Git 代码提交

Git 代码提交过程，一次本地仓库代码提交其实分为 2 步：

1）第一步：将工作区的文件变动记录添加到暂存区，进行暂存

```
git add 文件1, 文件2, ...
或
git add 目录1, 目录2, ...
```

2）第二步：将暂存区保存的变动记录一次性提交到仓库区，生成一个版本记录

```
git commit -m '提交说明信息'
```

查看 Git 仓库变化状态：

```
git status
```

查看 Git 提交版本记录：

```
git log
```

1.5 【熟练常握】代码改动撤销

对于工作区文件改动之后的撤销，分为两种情况：

1）情况1：改动了工作区文件，但是未将变动记录添加到暂存区

```
git checkout -- <file>…
```

2）情况2：改动了工作区文件，并且已将变动记录添加到暂存区

```
git reset HEAD <file>…
git checkout -- <file>…
```

1.6 【熟练常握】Git 版本回退

进行 Git 版本回退时，可以采用如下两种方式：

```
# HEAD：表示 Git 版本记录中的当前版本
# HEAD^：表示 Git 版本记录中的当前版本的上一个版本
# HEAD^^：表示 Git 版本记录中的当前版本的上上个版本，依次类推...
# HEAD~1：表示 Git 版本记录中的当前版本的上一个版本
# HEAD~2：表示 Git 版本记录中的当前版本的上上个版本，依次类推...
git reset --hard 回退版本
或
# 对应版本的编号可以通过 git log 或 git reflog 查看
git reset --hard 版本编号
```

1.7 【基本常握】github 远程仓库操作

克隆远程仓库：

```
git clone 远程代码仓库地址
```

本地仓库代码推送到远程仓库：

```
git push
```

拉取远程仓库代码到本地仓库：

```
git pull
```

2. Redis 数据库

2.1 【基本了解】Redis 简介

Redis 是 NoSQL 技术阵营中的一员，是一个使用 ANSI C语言编写、支持网络、可基于内存亦可持久化的日志型、Key-Value 非关系型数据库，并提供多种语言的 API。Redis 在开发时，经常用来做缓存，提高数据的读写效率。

2.2 【基本了解】Redis 安装、配置和启动

Redis 数据库安装：

```
apt-get install redis
```

Redis 配置文件：

```
/etc/redis/redis.conf
```

Redis 数据库启动、停止、重启操作：

```
systemctl start redis
systemctl stop redis
systemctl restart redis
systemctl status redis
```

2.3 【基本常握】Redis 数据存储操作命令

redis 命令查询参考文档：http://doc.redisfans.com/

Redis 数据库中没有表的概念，存储的数据都是 Key-Value 数据结构，每条数据都是一个键值对，Key 都是字符串，具体 Value 分为 5 种数据类型：

1）string：字符串

```
# 结构类似于 Python 中的字符串
Key: "字符串值"
```

2）list：列表

```
# 结构类似于 Python 中的列表
Key: ["元素1", "元素2", ...]
```

3）hash：哈希

```
# 结构类似于 Python 中的字典
Key: {"字段名1": "值1", "字段名2": "值2", ...}
```

4）set：集合

```
# 元素无序，且唯一
Key: ("元素1", "元素2", ...)
```

5）zset：有序集合

```
# 元素有序，且唯一
Key: ("元素1", "元素2", ...)
```

string 操作命令：

```
# 设置键值，如果设置的键不存在则为添加，如果设置的键已经存在则修改
set key value
# 获取指定 key 对应的 value 值
get key
# 设置键值及过期时间，以秒为单位
setex key seconds value
# 一次设置多个 key-value 数据
mset key1 value1 key2 value2 ...
# 一次获取指定多个 key 对应的 value 值
mget key1 key2 ...
# 向指定 key 对应的 value 值后追加内容
append key value
```

list 操作命令：

```
# 从列表左侧插入数据
lpush key value1 value2 ...
# 从列表右侧插入数据
rpush key value1 value2 ...
# 在列表指定元素的前或后插入新元素
linsert key before或after 现有元素 新元素

# 获取列表里指定范围内的元素
# 命令说明：
# start、stop为元素的下标索引
# 索引从左侧开始，第一个元素为0
# 索引可以是负数，表示从尾部开始计数，如 -1 表示最后一个元素的索引
lrange key start stop
# 设置列表指定索引位置的元素值
lset key index value

# 删除列表中的指定元素
```

```
# 将列表中前 count 次出现的值为 value 的元素移除
# count > 0：从头往尾移除
# count < 0：从尾往头移除
# count = 0：移除所有
lrem key count value

# 只保留列表指定范围内的元素，不在范围内的会被删除
# 命令说明：
# start、stop为元素的下标索引
ltrim key start stop
```

hash 操作命令：

```
# 设置 hash 中单个属性和值
hset key field value
# 一次设置 hash 中多个属性和值
hmset key field1 value1 field2 value2 ...
# 获取指定 hash 键所有的属性
hkeys key
# 获取 hash 中指定单个属性的值
hget key field
# 一次获取 hash 中指定的多个属性的值
hmget key field1 field2 ...
# 获取指定 hash 键所有属性的值
hvals key
# 删除 hash 指定的属性，属性对应的值会被一起删除
hdel key field1 field2 ...
```

set 操作命令：

```
# 向指定 set 集合中添加元素
sadd key member1 member2 ...
# 获取指定 set 集合中的所有元素
smembers key
# 删除 set 集合中的指定元素
srem key member1 member2 ...
```

zset 操作指令：

```
# 向指定 zset 集合中添加元素
zadd key score1 member1 score2 member2 ...

# 获取 zset 集合中指定范围内的数据
# start、stop为元素的下标索引
# 索引从左侧开始，第一个元素为0
# 索引可以是负数，表示从尾部开始计数，如 -1 表示最后一个元素的索引
zrange key start stop
```

```
# 获取 zset 集合中 score 在 min 和 max 之间的元素
zrangebyscore key min max
# 获取 zset 集合中指定元素的 score
zscore key member

# 删除 zset 集合中的指定元素
zrem key member1 member2 ...
# 删除 zset 集合中 score 在 min 和 max 之间的元素
zremrangebyscore key min max
```

key 操作指令：

```
# 显示当前 Redis 数据库中有哪些 key，pattern 支持正则表达式
keys pattern
# 判断 key 是否存在，存在返回1，不存在返回0
exists key
# 查看 key 对应的 value 的类型
type key
# 删除指定 key 及其对应的 value
del key1 key2 ...
# 设置指定 key 的过期时间
# 如果没有指定过期时间则一直存在，直到使用 del 删除
expire key seconds
# 查看指定 key 的剩余有效时间
ttl key
```

2.4 【基本常握】Python 操作 Redis 数据库

使用 Python 语言去操作 Redis 数据库，需要安装 redis-py 模块，安装命令如下：

```
pip install redis==3.5.3
```

redis-py 官方文档地址：https://redis-py.readthedocs.io/en/stable/index.html

基本使用：

使用 redis-py 模块去操作 Redis 数据库，需要首先创建一个 Redis 类的对象，链接到 Redis 数据库。

```
from redis import Redis
# 创建 Redis 数据库链接
redis_conn = Redis(host=<Redis 服务端IP>, port=<Redis 服务端PORT>, db=<Redis 数据库编号>)
# 参数说明：
# host：默认值为 127.0.0.1
# port：默认值为 6379
# db：默认值为 0
```

创建了 Redis 对象之后，即可通过调用该对象的方法对 Redis 数据库中的各种数据类型进行操作。

3. 课堂补充内容

3.1 【简单了解】pip install 和 apt-get install 的区别?

pip install ...：安装 python 的第三方包

apt-get install ...：安装一个软件，和 python 无关

3.2 【简单了解】vi 和 vim 基本使用

vi 和 vim 的关系相当于 IPhone 和 IPhone plus 的关系，vim 是 vi 的升级版，但其使用方式是一样的。

Ubuntu 系统中提供的 vim 操作教程打开方式：

```
# 打开 ubuntu 系统中的 vim 操作教程
vimtutor zh_CN
```

vi 和 vim 数据保存的两种方式：

```
1）esc 回到命令模式，先按：，然后输入 wq
2）esc 回到命令模式，直接按 shift + zz
```

3.3 【简单了解】PyCharm 结合 Git 管理项目代码

PyCharm 结合 Git 管理项目代码时：

红色的文件：表示该文件还未添加到暂存区，

绿色的文件：表示该文件以及添加到暂存区，

蓝色的文件：表示表示该文件的内容发了改动

PyCharm 代码提交快捷键：Ctrl + k

PyCharm 本地代码提交记录推送到远程仓库快捷键：Ctrl + Shift + K