11111111111111111111111111111

虚拟环境：

Q1：virtualenv是什么？为什么需要？常用的命令有哪些？

python 虚拟环境配置工具

1. virtualenv通过创建独立Python开发环境的工具, 来解决依赖、版本以及间接权限。

virtualenv创建一个拥有自己安装目录的环境, 这个环境不与其他虚拟环境共享库, 能够方便的管理python版本和管理python库。

1. 如果我们要同时开发多个应用程序，那这些应用程序都会共用一个Python，就是安装在系统的Python 3。如果应用A需要jinja 2.7，而应用B需要jinja 2.6，就可以通过virtualenv为每个应用创建一套独立的运行环境。virtualenv为应用提供了隔离的Python运行环境，解决了不同应用间多版本的冲突问题。
2. 先进入项目根目录 输入env\Scripts\activate，即可进入（env）虚拟环境，常用的命令有pip python等，若要关闭虚拟环境，使用deactivate即可退出。

Q2：pyenv 是什么？什么场景下需要用到这个东西？它和 virtualenv 的关系是什么？为什么有了 virtualenv 还要 pyenv ?

Pyenv是python版本环境切换工具，但是不支持Windows

Windows上由pywin替代，但pywin功能较少

Q3：virtualenvwrapper 是什么？为什么要有这个东西？和 virtualenv 的关系是什么？有什么常用命令？

使用shell开发，是对virtualenv的一个封装，简化了virtualenv命令的使用，并且进行了一些功能扩展

Shell是用来解决用户如何与操作系统通信的问题。

可以把 shell 理解为 命令解释器

Web框架：

Q1：框架是什么概念？为什么要有框架这个东西？用框架和不用框架有什么区别？

围绕web应用的所有问题中，两个问题尤为突出：

我们如何将请求的 URL 映射到处理它的代码上？

我们怎样动态的构造请求的HTML返回给客户端，HTML中带有计算得到的值或者从数据库中取出来的信息？

每个 web 框架都以某种方法来解决这些问题，也有很多不同的解决方案。

Web框架的目的：向程序员隐藏了处理HTTP请求和响应相关的基础代码。

Python web框架都以相同的方式工作，他们接收HTTP请求，分派代码，产生HTML，创建带有内容的HTTP响应。

Q2：flask 和 django , django 和 tornado 的区别是什么？为什么有了 django 又出了 flask （ flask 相比于 django 有什么优势）？

Django包含ORM（对象关系映射），目的有：1、将python的class与数据库表建立映射，剥离出不同数据库引擎直接的差异。功能齐全。

Flask是一个微框架，不包括ORM。

Http相关：

Q1：什么是协议？协议在客户端和服务器之间充当了什么样的角色？

Q2：Cookies 是什么？ HTTP 协议为什么要有这么东西？ Cookies 一般存储在什么地方？我们日常浏览网页哪些地方或者哪些功能使用到了 Cookies ？

因为[HTTP协议](https://zh.wikipedia.org/wiki/HTTP)是无状态的，即[服务器](https://zh.wikipedia.org/wiki/%E6%9C%8D%E5%8A%A1%E5%99%A8)不知道用户上一次做了什么，这严重阻碍了[交互式Web应用程序](https://zh.wikipedia.org/w/index.php?title=%E4%BA%A4%E4%BA%92%E5%BC%8FWeb%E5%BA%94%E7%94%A8%E7%A8%8B%E5%BA%8F&action=edit&redlink=1)的实现。在典型的网上购物场景中，用户浏览了几个页面，买了一盒饼干和两瓶饮料。最后结帐时，由于HTTP的无状态性，不通过额外的手段，服务器并不知道用户到底买了什么。 所以Cookie就是用来绕开HTTP的无状态性的“额外手段”之一。服务器可以设置或读取Cookies中包含信息，借此维护用户跟服务器[会话](https://zh.wikipedia.org/wiki/%E4%BC%9A%E8%AF%9D_(%E8%AE%A1%E7%AE%97%E6%9C%BA%E7%A7%91%E5%AD%A6))中的状态。

Cookie另一个典型的应用是当登录一个网站时，网站往往会请求用户输入用户名和密码，并且用户可以勾选“下次自动登录”。如果勾选了，那么下次访问同一网站时，用户会发现没输入用户名和密码就已经登录了。这正是因为前一次登录时，服务器发送了包含登录凭据（用户名加密码的某种[加密](https://zh.wikipedia.org/wiki/%E5%8A%A0%E5%AF%86)形式）的Cookie到用户的硬盘上。第二次登录时，（如果该Cookie尚未到期）浏览器会发送该Cookie，服务器验证凭据，于是不必输入用户名和密码就让用户登录了。

1. Cookie会被附加在每个HTTP请求中，所以无形中增加了流量。
2. 由于在HTTP请求中的Cookie是明文传递的，所以安全性成问题。（除非用[HTTPS](https://zh.wikipedia.org/wiki/HTTPS)）
3. Cookie的大小限制在4KB左右。对于复杂的存储需求来说是不够用的。

Cookie总是保存在客户端中，按在客户端中的存储位置，可分为内存Cookie和硬盘Cookie。

内存Cookie由[浏览器](https://zh.wikipedia.org/wiki/%E6%B5%8F%E8%A7%88%E5%99%A8)维护，保存在[内存](https://zh.wikipedia.org/wiki/%E5%86%85%E5%AD%98)中，浏览器关闭后就消失了，其存在时间是短暂的。硬盘Cookie保存在[硬盘](https://zh.wikipedia.org/wiki/%E7%A1%AC%E7%9B%98)里，有一个过期时间，除非用户手工清理或到了过期时间，硬盘Cookie不会被删除，其存在时间是长期的。所以，按存在时间，可分为非持久Cookie和持久Cookie。

Q3：Cookies 和 Session 的区别？（ session 概念暂时不理解可以略过）

Session是在服务端保存的一个数据结构，用来跟踪用户的状态，这个数据可以保存在集群、数据库、文件中；  
Cookie是客户端保存用户信息的一种机制，用来记录用户的一些信息，也是实现Session的一种方式。

cookies服务端如何识别特定的客户，session用于识别具体的用户，保存在服务端，记录用户状态。

Q4：HTTP 常见的错误码有哪些？在浏览器的开发者工具中能找到吗？

HTTP协议状态码表示的意思主要分为五类 ,大体是 :  
  
1×× 　　保留  
2×× 　　表示请求成功地接收  
3×× 　　为完成请求客户需进一步细化请求  
4×× 　　客户错误  
5×× 　　服务器错误

客户端错误：

400 Bad Request  
因为错误的语法导致服务器无法理解请求信息。

401 Unauthorized    
如果请求需要用户验证。回送应该包含一个WWW-Authenticate头字段用来指明请求资源的权限。

402 Payment Required   
保留状态码。

403 Forbidden   
服务器接受请求，但是被拒绝处理。

404 Not Found   
服务器已经找到任何匹配Request-URI的资源。

405 Menthod Not Allowed   
Request-Line 请求的方法不被允许通过指定的URI。

406 Not Acceptable  
客户端浏览器不接受所请求页面的 MIME 类型。  
407 Proxy Authentication Required  
要求进行代理身份验证。  
408 Reqeust Timeout   
客服端没有提交任何请求在服务器等待处理时间内。

409 Conflict

410 Gone

411 Length Required   
服务器拒绝接受请求在没有定义Content-Length字段的情况下。

412 Precondition Failed  
前提条件失败。  
413 Request Entity Too Large   
服务器拒绝处理请求因为请求数据超过服务器能够处理的范围。服务器可能关闭当前连接来阻止客服端继续请求。

414 Request-URI Too Long   
服务器拒绝服务当前请求因为URI的长度超过了服务器的解析范围。

415 Unsupported Media Type   
服务器拒绝服务当前请求因为请求数据格式并不被请求的资源支持。

416 Request Range Not Satisfialbe  
所请求的范围无法满足。  
417 Expectation Failed  
执行失败。

服务器端错误：

500 Internal Server Error  
服务器遭遇异常阻止了当前请求的执行

501 Not Implemented   
服务器没有相应的执行动作来完成当前请求。

502 Bad Gateway  
Web 服务器用作网关或代理服务器时收到了无效响应。  
503 Service Unavailable   
因为临时文件超载导致服务器不能处理当前请求。

504 Gateway Timeout  
网关访问超时。  
505 Http Version Not Supported  
HTTP 版本不受支持。

Q5：HTTP 协议中对请求和响应的定义？

Q6：Web 开发框架和 HTTP 协议的关系是什么？

Q7：什么是域名？域名和IP地址的关系是什么？为什么有了IP地址还要域名？ DNS 又是什么？ DNS 在域名和IP地址之间充当了什么角色？

Q8：https://www.baidu.com/s? wd=%E6%85%95%E8%AF%BE%E7%BD%91&rsv\_spt=1&rsv\_iqid=0xd51ee8ab00005edc&issp=1&f=8&rsv\_bp=0&rsv\_idx=2&ie=utf- 8&tn=baiduhome\_pg&rsv\_enter=1&rsv\_sug3=8&rsv\_sug1=6&rsv\_sug7=100&rsv\_t=1287WG5bMXS8Ru9oVfbynyerbC60HFka9eISwAFlmjHC3YMLZtfIksCnch7vwyZGoLMJ&rsv\_sug2=0&inputT=1341&rsv\_sug4=1341 这么长的一个 URL 路由中，哪一段是域名？还是这一段 URL 全是域名？

在浏览器中键入：

http://www.sohu.com/question/index.htm

访问搜狐网页，简述浏览器进行域名解析，建立客户／服务器连接与实现 WWW网页浏览的工作全过程。

[答] URL是统一资源定位符，其一般格式为：协议名：//主机域名/路径名

HTTP协议为超文本传输协议，即WWW使用的协议

简述工作过程如下： 用户在浏览器的 URL 域键入：

http://www.sohu.com/question/index.htm 后，

浏览器首先从URL中抽取出网页所在站点的主机域名www.sohu.com，通过域名系统解析，获知对方站点的IP地址，然后和该IP地址的WEB服务器建立TCP连接,连接对方服务器的端口号为80；连接建立起来后，由于URL中协议名为HTTP,因而用HTTP协议与该站点(WEB服务器)进行交互：浏览器先向该WEB服务器发出HTTP请求报文，请求WEB服务器发送路径名/question/index.htm指定的页，请求报文中的命令行为：

GET /question/index.htm HTTP/1.0

服务器响应请求，从磁盘上取得所要的网页，构建状态行等响应报文头，并将取得的网页封装在该HTTP响应报文中，传回给浏览器，然后双方断开TCP连接。当浏览器取得HTTP响应报文后，解封出网页，然后按HTML的相应格式进行显示。

数据库：

Q1：网站的运行是否离不开数据库？有无数据库的区别是什么？为什么要用数据库存储数据（有什么好处）？

Q2：在我们日常浏览的网页中，哪些数据一般会放在数据库中进行存储？哪些数据不需要？为什么？

Q3：在我们开发实现的网站中，数据库扮演了什么角色？SQL中的DML语句是指哪些语句？分别表示对数据库中数据如何操作？日常浏览网页过程中，我 们的哪些行为会导致数据库需要执行上边提到的DML语句？举出实例。

DML（data manipulation language）是数据操纵语言：它们是SELECT、UPDATE、INSERT、DELETE，就象它的名字一样，这4条命令是用来对数据库里的数据进行操作的语言。

增删查改语句，对数据进行增删查改。

日常浏览网页，我们添加、删除、修改、查询记录条目会导致数据库执行DML语句。

比如要选课时报名，需要查看课程信息，点击查看，页面即展示有关课程内容，选好课程之后删除，即取消选课等等。

DDL（data definition language）是数据定义语言：DDL比DML要多，主要的命令有CREATE、ALTER、DROP等，DDL主要是用在定义或改变表（TABLE）的结构，数据类型，表之间的链接和约束等初始化工作上，他们大多在建立表时使用。

DCL（DataControlLanguage）是数据库控制语言：是用来设置或更改数据库用户或角色权限的语句，包括（grant,deny,revoke等）语句。

Q4：Mysql 和 Sqllite 的区别是什么？

Sqlite是单机数据库，功能简约，小型化，追求最大磁盘效率。

MySQL是完善的服务器数据库。功能全面，综合化，追求最大并发效率。

MySQL适用于服务器数据量大、功能多的情况，需要安装。如网站访问量比较大的，sqlite是一款小型数据库，简单一致性好，无需安装，只需要几个文件。

Sqlite安装方便，适合做简单的测试应用，适合快速开发 适合嵌入式开发，没有用户管理，没有完整的数据库功能。

MySQL适合做产品，有管理用户权限等功能，有完整的数据库功能。

Q5：什么是数据库范式？举出实例表明数据库范式在实际开发表设计中的应用。（进阶了解）

范式是对关系的不同数据依赖程度的要求，即符合某种级别的关系模式的集合，满足不同程度要求的为不同范式

通过模式分解将一个低级范式转换为若干个高级范式的过程称作规范化（概念的纯粹化）

“不好”的关系模式会存在问题

不可表示：插入异常、删除异常

冗余：数据冗余、更新复杂

属性间存在某种数据依赖

范式：对关系的数据依赖程度的要求

版本控制工具：

A1：注册 Coding 和 GitHub 账号（如果没有）。

Q1：版本控制工具是什么？为什么在开发过程中需要这些工具？在 Git 出现前，人们是如何进行版本控制的？使用 Git 的优点是什么？

版本控制可以确保在软件项目开发中，不同的开发人员所涉及的同一文档都得到更新。

版本控制的目的是实现开发团队并行开发，提高开发效率的基础，其目的是在于对软件开发进程中文件/目的的发展过程提供有效的追踪手段，保证在需要时可以回到旧的版本，避免文件的丢失，修改的丢失和相互覆盖。

版本控制的功能在于跟踪记录整个软件的开发过程，包括软件本身和相关文档，以便对不同阶段的软件以及相关文档进行表示并进行差别分析，对软件代码进行可撤销的修改，便于汇总不同开发人员所做的修改，辅助协调和管理软件开发团队。

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！

Git出现之前有CVS和SVN这些免费的版本控制系统，但是集中式的版本控制不但速度慢，而且必须联网才能使用，有一些商用的版本控制系统，虽然好用，但是需要付费，和开源精神不符，到了02年，Linux系统已经发展了10年，代码库之大很难通过手工方式管理，于是Linus选择了BitKeeper这个商用的版本控制工具，（Bitkeeper授权Linux社区免费使用了一段时间），但后来由于Andrew试图破解Bitkepper的协议，被发现后Bitkeeper要收回Linux社区的免费使用权，于是Linus花两周时间自己用C写了一个分布式版本控制系统，即Git。Git迅速成为最流行的分布式版本控制系统，为开源项目免费提供Git存储。

集中式版本控制必须联网才能工作，版本库集中存放在中央服务器，提交文件受到网速控制。

分布式版本控制，每个人电脑上都是一个完整的版本库，工作时无需联网等。Git还具有分支管理这个强大的功能。

A3：阅读Git教程，熟悉你觉得会常用的命令。

mkdir sample 创建一个新的目录

cd sample 进入该目录

git init 把该目录变成git可以管理的仓库

把文件添加到该目录下

git add test.txt 把文件添加到仓库

git commit –m“some discribtion”把文件提交到仓库 –m后面输入的是本次提交的说明（此外commit可以一次提交多个文件）

git status 修改文件后查看结果 掌握仓库当前状态

git diff 查看之前对提交文件的操作 掌握修改内容

git log 显示从最近到最远的提交日志

git log –-pretty=oneline 只显示一行输出

git reset –hard commit\_id 可以进行版本穿梭 commit\_id仅需前几位即可

git reflog 查看命令历史 回到未来

git checkout – filename 让文件回到最近一次git comiit /git add时的状态

git reset HEAD filename 可以把暂存区的修改撤销

rm filename 可以直接删除某文件

git rm filename删除文件 并且git commit

git checkout – filename 恢复到最新版本

git push origin master 本地提交之后push到远程仓库

git remote add origin [git@github.com:aryayin/repo-name.git](mailto:git@github.com:aryayin/repo-name.git)关联一个远程库

git clone [git@github.com:aryayin/repo-name.git](mailto:git@github.com:aryayin/repo-name.git) 根据远程库克隆一个本地库

然后cd repo-name 开始之前的那些操作——

git checkout –b dev 创建dev分支 然后切换到dev分支 git checkout加上-b表示创建并切换

相当于git branch dev git checkout dev

git branch 查看当前分支

git checkout xxx 切换分支

git merge xxx合并分支

git branch –d xxx删除分支

git stash可以把当场的工作储存起来，以后恢复现场继续工作

然后确定要在哪个分支修复bug 假设要在master上修复，就从master创建临时分支，使用 git checkout master,git checkout –b issue-01 然后进行add commit

最后切换回master分支，并完成合并 最后删除issue-01分支

git stash app恢复后 stash内容并不删除

用git stash drop删除

git stash pop 恢复的同时把stash内容也删了 可用git stash list查看

git branch –D feature-vulcan强行删除没有被合并的分支

git tag v1.0打一个新标签 打在最新的commit上

git tag 查看标签

git tag v0.9 commit\_id打在历史的提交上

可以用git show <tagname>查看标签信息

git tag –d v0.1可以删除

git push origin <tagname>

git push origin –tags

删除： 先从本地删除

git push origin:refs/tags/v0.9

前端相关：

Q1： HTML 、 CSS 、 JavaScript 分别表示在前端网页中扮演什么角色？

HTML：超文本标记语言，是为“网页创建和其他可在网页浏览器中看到的信息” 设计的一种标记语言。

CSS：层叠样式表，是一种用来表现HTML或XML等文件样式的计算机语言。

JavaScript：一种直译式脚本语言，是一种动态类型、弱类型、基于原理的语言，内置支持类型。

每个网页都是一个html文档，由浏览器解释呈现的，CSS是html中用来布局表现的，JavaScript是html中用来动态交互的脚本。

好比html是一个人，css是给人穿上了衣服，js是让其他人认为这个人是活的，而不是死的。

Q2：常见的前端开发框架有哪些？

AngularJS jQuery bootstrap adminLTE Gentelella

Q3：在前端开发中，使用框架和不使用框架的区别？

前端框架一般指用于简化网页设计的框架，如jquery等，这些框架封装了一些功能，比如html文档操作，漂亮的各种控件/按钮（表单）等。

前端框架的好处：对于互联网产品而言，前端的用户体验往往决定了一款产品的竞争力和吸引力，对于前端工程师，他们既要为网站上提供的产品和服务实现一流的Web页面，优化代码，保证体验最佳，更要考虑与市面上各大浏览器的兼容性，还需估计Web前端表现层和前后端交互的架构，以及模块化、通用类库 框架编写等一系列问题。由此提升前端开发的效率，降低开发成本和周期。

前端在基础架构上迫切的需要组件化开发和资源管理，而解决资源管理的方法即需要：

一个通用的资源表生成工具+基于表的资源加载框架，运用框架可以提高开发效率和运行性能。前端建设的第一任务是根据项目特征进行技术选型。框架解放了生产力，合理的技术选型可以为项目节省许多工程量。框架功能强大，使得前端的开发变得快速。