Apache详解

一、先导概念

web全称：World Wide Web，全球广域网，也称为万维网。

web项目分为如下架构：

C/S架构：全称客户端服务端架构，英文Client/Server。qq、微信、京东app、各种游戏app或者电脑游戏应用等等。服务端主要负责加工处理数据，客户端主要用来展示数据。有些数据是存储在服务端的，比如商品信息，我们换个电脑查看，商品信息还在，有些数据是存储在客户端的，比如聊天记录，我们换个电脑在登录qq，如果不做同步，就没有聊天记录了。

B/S架构：是C/S架构的一种，只是单独拿出来说，浏览器/服务端架构，英文Browser/Server，京东网站、淘宝网站等等，基于浏览器访问的网站都是B/S架构。这种架构的网站非常多，因为成本也比较便宜，不是每个公司都会开发自己的客户端程序，而且如果每个公司都制作自己客户端程序，我们的手机或者电脑要安装很多的客户端，其实很难受，而浏览器就提供了一个统一的入口，我们只需要下载一个浏览器软件，比如谷歌浏览器、火狐浏览器、360浏览器等等，只要打开一个标签页，输入一个网址就能访问一个网站，那么相当于我们一个软件就访问了多个公司的网站，很方便，这就是统一入口的意思。

web发展了几个阶段：

web 1.0阶段：纯静态页面，不能交互，只是用来展示一些内容给大家看。客户端页面使用的HTML语言开发的。

web 2.0阶段：支持用户上传信息，交互，购物。服务端后台开发语言有：php、java、ruby、python等 ，刚开始使用后端开发语言就是为了用它来生成动态页面的，比如一些动态数据的展示等。php、java是早先用的最多的。

web 3.0阶段：移动互联网，衣食住行，出门只带个手机就行。服务端后台开发语言有：php、java、ruby、python等

二、 Apache的简单介绍

Apache(音译为阿帕奇)是世界使用排名第一的Web服务器软件。它可以运行在几乎所有广泛使用的计算机平台上，由于其跨平台和安全性被广泛使用，是最流行的Web服务器端软件之一。

它可以运行在几乎所有广泛使用的计算机平台上。

Apache源于NCSA httpd服务器，经过多次修改，成为世界上最流行的Web服务器软件之一。Apache取自“a patchy server”的读音，意思是充满补丁的服务器，因为它是自由软件，所以不断有人来为它开发新的功能、新的特性、修改原来的缺陷。Apache的特点是简单、速度快、性能稳定，并可做代理服务器来使用。

Apache 软件拥有以下特性：

1.开放源代码。
2.跨平台应用，可运行windoows和大多数linux系统。
3.支持php，python和java等多种网页编辑语言。
4.采用模块化设计。
5.运行非常稳定。
6.具有相较好的安全性。
6.提供用户会话过程的跟踪。
7.拥有简单而强有力的基于文件的配置过程。
8.支持多种方式的HTTP认证。

三、Apache配置文件详解

```
ServerRoot "/etc/httpd"      #Apache服务的根路径

Listen 80   #Apache监听的端口

Include conf.modules.d/*.conf   #加载目录中所有配置文件（以.conf为后缀）

User apache      #指定Apache服务的运行用户
Group apache     #指定Apache服务的运行组


ServerAdmin root@localhost  #指定Apache服务管理员通知邮箱地址


<Directory />   #指定了对根目录("/")的访问控制规则
    AllowOverride none  #是否允许目录下的.htaccess文件生效，此处为不生效，如果要生效请设置为all
    Require all denied  #拒绝所有访问
</Directory>


DocumentRoot "/var/www/html"     #指定网站根目录，当客户端请求访问网站时，服务器会从该目录下寻找对应的文件并将其返回给客户端。

<Directory "/var/www">       #指定了对/var/www目录的访问控制规则
    AllowOverride None  #指定在该目录下不允许使用.htaccess文件覆盖服务器的配置
    Require all granted #指定允许所有的请求访问该目录。
</Directory>

<Directory "/var/www/html">      # 指定了对/var/www/html目录的访问控制规则
    Options Indexes FollowSymLinks  #设置目录下的一些特征，indexes表示当用户访问该目录时，如果找不到index文件,则返回该目录下的文件列表给用户，FollowSymLinks表示服务器允许在此目录中使用符号连接

    AllowOverride None  #指定在该目录下不允许使用.htaccess文件覆盖服务器的配置

    Require all granted #指定允许所有的请求访问该目录。这个设置允许客户端访问/var/www/html目录下的所有文件和子目录。
</Directory>
```

```
<IfModule dir_module>          #指定了一个条件，如果Apache已经加载了dir_module模块，则执行其中的指令。dir_module模块是Apache的核心模块之一，用于处理目录列表和默认索引文件等功能。
    DirectoryIndex index.html    #设置默认的索引文件为index.html，当客户端请求的URL路径没有指定具体文件时，自动使用index.html文件作为默认文件返回给客户端。这个设置用于指定网站的默认首页。
</IfModule>

<Files ".ht*">  #指定了一个条件，该条件限制了所有以".ht"开头的文件的访问权限。
    Require all denied  #设置拒绝所有请求访问以".ht"开头的文件。这个设置用于保护服务器上的Apache配置文件，防止未经授权的访问和恶意修改。
</Files>

ErrorLog "logs/error_log"    #设置错误日志的存放位置

LogLevel warn    #设置日志记录的级别

<IfModule log_config_module>     #指定了一个条件，如果Apache已经加载了log_config_module模块，则执行其中的指令。log_config_module模块是Apache的核心模块之一，用于记录访问日志。
    LogFormat "%h %l %u %t \"%r\" %>s %b \"%{Referer}i\" \"%{User-Agent}i\""
combined    #定义了一个自定义的访问日志格式，称为combined格式。该格式由多个字段组成，包括客户端IP地址、远程登录名、远程用户身份、请求时间、请求的URL路径、HTTP响应状态码、响应的字节数、Referer信息和User-Agent信息等。这个设置用于定义日志记录的格式。

    LogFormat "%h %l %u %t \"%r\" %>s %b" common #定义了另一个访问日志格式，称为common格式。该格式只包含一些基本的字段，包括客户端IP地址、远程登录名、远程用户身份、请求时间、请求的URL路径、HTTP响应状态码和响应的字节数等。

    <IfModule logio_module> #指定了一个条件，如果Apache已经加载了logio_module模块，则执行其中的指令。logio_module模块是Apache的核心模块之一，用于记录响应时间和字节数。

      LogFormat "%h %l %u %t \"%r\" %>s %b \"%{Referer}i\" \"%{User-Agent}i\" %I %O" combinedio #定义了一个自定义的访问日志格式，称为combinedio格式。该格式与combined格式相似，但还包括响应的输入字节数和输出字节数。这个设置用于定义日志记录的格式。

    </IfModule>


    CustomLog "logs/access_log" combined     #设置访问日志路径和使用的格式
</IfModule>

<IfModule alias_module>  #指定了一个条件，如果Apache已经加载了alias_module模块，则执行其中的指令。alias_module模块是Apache的核心模块之一，用于管理URL的别名和重定向。


    ScriptAlias /cgi-bin/ "/var/www/cgi-bin/"    #设置CGI脚本目录位置的别名。

</IfModule> #结束条件指令

<Directory "/var/www/cgi-bin">  #针对/var/www/cgi-bin的设置
    AllowOverride None  #指定在该目录下不允许使用.htaccess文件覆盖服务器的配置
    Options None    #设置目录下的一些特征,这里表示不允许使用任何选项
    Require all granted ##指定允许所有的请求访问该目录
</Directory>
```

```
<IfModule mime_module>  #指定了一个条件，如果Apache已经加载了mime_module模块，则执行其中的指令。mime_module模块是Apache的核心模块之一，用于处理HTTP响应的MIME类型和内容编码。
    TypesConfig /etc/mime.types #设置MIME类型配置文件的路径

    AddType application/x-compress .Z   #AddType 指令可以将给定的文件扩展名映射到指定的内容类型
    AddType application/x-gzip .gz .tgz

    AddType text/html .shtml
    AddOutputFilter INCLUDES .shtml
</IfModule>

AddDefaultCharset UTF-8 #指定默认字符集

<IfModule mime_magic_module>
    MIMEMagicFile conf/magic
</IfModule>
#检查是否已经加载了mime_magic_module模块，并设置MIME类型魔术文件的路径。mime_magic_module模块用于根据文件内容判断文件类型，可以帮助Apache更准确地识别文件类型。

EnableSendfile on

IncludeOptional conf.d/*.conf       #包含其他配置文件
```

.htaccess 文件简介

.htaccess文件是apache的分布式配置文件。他负责相关目录下的网页配置，通过.htaccess可以帮我们实现。必须以ASCII模式上传，并且给其可读性。

.htaccess 可以帮我们实现：网页301重定向，自定义404错误页面，改变文件扩展名、允许/阻止特定的用户或者目录的访问、禁止目录列表、封禁特定 IP 地址的用户，只允许特定 IP 地址的用户，以及使用其他文件作为index文件配置默认文档等功能。

.htaccess文件一般情况下，不应该使用.htaccess文件，除非对主配置文件没有访问权限。用户认证不但能使用.htaccess文件实现，把用户认证写在主配置文件中是完全可行的，而且是一种很好的方法。

任何希望放在.htaccess 文件中的配置，都可以在主配置文件的段中，而且更高效。

避免使用.htaccess文件有两个主要原因：

①性能。如果AllowOverride启用了.htaccess文件，则Apache需要在每个目录查找.htaccess文件，无论是否找到，启用都会导致性能下降。另外，对每一个请求，都需要读取一次.htaccess文件，还有Apache必须在所有上级的目录中查找.htaccess文件，以使所有有效的指令都起作用，例如：

如果请求/www/htdocs/example中的页面，Apache必须查找以下文件：
/.htaccess /www/.htaccess /www/htdocs/.htaccess /www/htdocs/example/.htaccess

②安全。这样会允许用户自己修改服务器的配置，这会导致某些意想不到的修改。

注意：在/www/htdocs/example目录下的.htaccess文件中的放置指令，与在主配置文件中<Directory /www/htdocs/example>段中放置相同的指令，是完全等效的。

Options：目录中将使用哪些服务器特性

```
None  #不启用任何额外特性

Indexes  #当用户访问该目录时，找不到index文件,则返回该目录下的文件列表给用户

FollowSymLinks  #可以在该目录中使用符号连接，这样就访问目录之外的文档
```

Require：目录访问控制

```
<RequireAll>
Require all granted  #允许所有访问
</RequireAll>

<RequireAll>
Require all denied  #拒绝所有访问
</RequireAll>

<RequireAll>
Require ip 192.168.0.1  #仅允许IP：192.168.0.1 访问
</RequireAll>

<RequireAll>
Require all granted
Require not ip 192.168.0.1 #仅禁止IP：192.168.0.1访问
</RequireAll>


#设置用户认证，需要先使用htpasswd命令创建用户密码文件，例如：htpasswd -c
/var/www/html/.htuser tom 如果文件已存在，请不要使用-c选项，否则将覆盖现有文件。
<Directory "/var/www/html/test">
authuserfile "/var/www/html/.htuser"
authname "please input your username and passwd"
authtype basic
<RequireAll>
Require valid-user  #允许认证文件中的用户访问
Require not user tom #禁止tom用户访问
</RequireAll>
</Directory>

#设置用户组认证，需要先使用htpasswd命令创建用户密码文件，例如：htpasswd -c
/var/www/html/.htuser tom 如果文件已存在，请不要使用-c选项，否则将覆盖现有文件。
#然后创建用户组存放文件，例如：vim /var/www/html/.htgroup 然后在该文件内写入用户组信息。格
式为：组名:用户名
<Directory "/var/www/html/test">
authuserfile "/var/www/html/.htuser"
authgroupfile "/var/www/html/.htgroup"
authname "please input your username and passwd"
authtype basic
<RequireAll>
require group g1
</RequireAll>
</Directory>


<RequireAny>    #该标签表示满足任意一个规则即可访问
```

```
Require ip 192.168.228.163
Require ip 192.168.228.1
</RequireAny>
```

Apache日志标识含义

标识	含义
%h	客户端ip
%l（小写的L）	Remote User, 通常为一个减号（“-”）；
%u	Remote user (from auth; may be bogus if return status (%s) is 401)；非为登录访问时，其为一个减号；
%t	服务器收到请求时的时间；
%r	First line of request，即表示请求报文的首行；记录了此次请求的“方法”，“URL”以及协议版本；
%>s	响应状态码；
%b	响应报文的大小，单位是字节；不包括响应报文的http首部；
%{Referer}i	请求报文中首部“referer”的值；即从哪个页面中的超链接跳转至当前页面的；
%{User-Agent}i	请求报文中首部“User-Agent”的值；即发出请求的应用程序；
%I（大写的i）	接收的字节数，要使用这个指令必须启用mod_logio模块。
%O	发送的字节数，要使用这个指令必须启用mod_logio模块。

四、Apache设置虚拟主机

虚拟web主机指的是在同一台服务器中运行多个web站点，其中的每个网站并不会单独占用整个服务器，因此被称为“虚拟”web主机，通过虚拟web主机服务可以充分利用服务器的硬件资源，从而降低网站构建与运行成本。虚拟主机类型包括三种：

（1）：基于域名：每个虚拟主机使用不同的域名，但是IP地址相同，使用最为普及的方式

（2）：基于端口：每个虚拟主机使用不同的TCP端口，但是IP地址相同

（3）：基于IP地址：每个主机的IP地址不同，这种方式需为服务器配置多个网络接口，并且拥有多个IP，一般不使用该方法

```
配置基于域名的虚拟主机
1、cd /var/www/html  #切换到Apache的网站根目录
```

2、mkdir a b #创建两个目录，用于存放虚拟主机的网站页面
3、echo "This is a" > a/index.html;echo "This is b" > b/index.html #创建默认首页文件并写入内容
4、cat a/index.html b/index.html 查看是否成功写入
5、vim /etc/httpd/conf.d/vhosts.conf #进行虚拟主机设置，在该文件中写入下列内容

```
<VirtualHost *:80>
        DocumentRoot "/var/www/html/a"
        ServerName www.a.com
        ErrorLog "logs/err_log"
        CustomLog "logs/access_log" combined
</VirtualHost>
<VirtualHost *:80>
        DocumentRoot "/var/www/html/b"
        ServerName www.b.com
        ErrorLog "logs/err_log"
        CustomLog "logs/access_log" combined
</VirtualHost>
```

6、systemctl restart httpd #重启服务

配置基于端口的虚拟主机
前期准备步骤与基于域名的虚拟主机设置方法相同，次数不再赘述

5、vim /etc/httpd/conf.d/vhosts.conf #进行虚拟主机设置，在该文件中写入下列内容

```
listen 81
listen 82
<VirtualHost *:81>
        DocumentRoot "/var/www/html/a"
        ErrorLog "logs/err_log"
        CustomLog "logs/access_log" combined
</VirtualHost>

<VirtualHost *:82>
        DocumentRoot "/var/www/html/b"
        ErrorLog "logs/err_log"
        CustomLog "logs/access_log" combined
</VirtualHost>
```

6、systemctl restart httpd #重启服务