Node.js笔记

目录

[Node.js笔记 1](#_Toc15995022)

[NodeJs和JavaScript有什么区别？ 1](#_Toc15995023)

[Linux的分类 1](#_Toc15995024)

[常见的虚拟机 2](#_Toc15995025)

[FHS协议 2](#_Toc15995026)

[根目录路径 2](#_Toc15995027)

[文件颜色代表的含义 2](#_Toc15995028)

[基本命令 3](#_Toc15995029)

[文件权限 4](#_Toc15995030)

[NodeJsAPI 4](#_Toc15995031)

[Net: 4](#_Toc15995032)

[文件读取异步和同步的区别 6](#_Toc15995033)

## NodeJs和JavaScript有什么区别？

NodeJs是个平台，JavaScript是个语言。

编译过程分为词法分析、语法分析、语义分析三部分。

词法分析：识别关键字、标识符、分界符，运算符的过程。

语法分析：将代码转化为命令语句或者短语。

语义分析：生成操作系统能够执行的程序；程序作用在操作系统上。

NodeJS是在浏览器大战时期出现的。

世界上第一个普遍使用并且可以展示图片的浏览器是NCSA Mosaic 作者：蒂姆伯纳斯·李（万维网之父）

NodeJs特点：高性能、异步IO、事件驱动、多线程，并且由线程池管理。

## Linux的分类

1. Ubuntu：界面华丽，完善的管理系统和软件支持 分为用户端和服务端
2. CentOS：很多公司在生产环境部署CentOS， fromRHEL。(开源免费的)

RHEL：Red Hat Enterprise Linux(小红帽商业Linux)

1. Debian：内核非常小，非常稳定。适用于系统硬盘空间少，内存小的机器。
2. RedHat：RedHat基于Linux,遵守GPL（开源协议）小红帽不能用于商业用途,于是在小红帽的版本上拷贝了一份就是CentOS。

常见的虚拟机：VM Ware(收费)和 Oracle VM VirtualBox(免费)

FHS协议:File System Hierarchy Standard(文件系统层次化标准)

根目录路径：

1. /usr(UNIX Software Resource) 和软件安装有关

2. /var(variable) 和系统运行过程相关

3. /bin 文件夹下都是可执行的

4. /boot 和系统开机相关的目录

5. /dev 和设备相关的目录 访问设备目录就是访问设备

6. /etc 存放配置文件（比较重要）

7. /home 用户目录

8. ~ 当前用户的主文件夹

9. /root 超级管理用户目录

10. /lib 存放Linux系统的函数库

11. /Media 存放一些可以删除的设备（如U盘、光盘）

12. /mnt 和Mdeia类似

13. /opt 存放第三方软件目录（比较常用）

14. /sbin 存放一些系统管理员执行的文件

15. /srv 存放用户主动生产出来的数据

16. /tpm 临时文件

17. /proc 虚拟文件系统，不在硬盘上存着的，存在内存里；存放着系统内核、进程、外部设备状态、状态网络（不占磁盘空间）

18. /sys 也是虚拟文件系统，存放着和系统相关、和内核相关的信息

19. /Lost+found 当文件系统发生错误时，把一些丢失的文件碎片存起来，让用户有可能把丢失数据补回来。

文件颜色代表的含义：

蓝色表示目录；

绿色表示可执行文件；

红色表示压缩文件；

浅蓝色表示链接文件；

白色表示其他文件；

黄色表示设备文件，包括block, char, fifo。；

基本命令：

Cd：cd 文件夹名 (进入文件夹)

cd .. 回到上一级目录

cd ~ 回到当前用户的主文件夹

cd ../bin 进入上一级目录下的bin

ls： 当前目录

ls –l 列出目录详情 ll是ls –l的缩写

ls –al 可以看到隐藏文件 Linux的隐藏文件以‘.’开头

mkdir： 创建路径

mkdir –p 路径/路径 深层创建路径

mkdir –m 777 路径 创建路径并设置权限

rmdir： 删除路径

rmdir 路径 只能删除空路径

rm –rf 路径 删除路径下的所有内容（慎用）

pwd： 以绝对路径的方式显示用户当前工作目录

useradd： 创建用户（用户必须至少属于一个用户组）创建用户时会自动创建一个用户组

useradd –G 用户组名 用户 将用户创建到指定用户组中

userdel：删除用户

passwd 用户名 设置用户密码

usermod –g 组名 用户名 修改用户的主要组

id 用户名 查看用详细信息

su 用户名 切换用户

groupadd：创建用户组

groupdel：删除用户组

cat：查看文件

cat /etc/passwd 查看用户

cat /etc/group 查看用户组

ip addr：查看IP地址

lscpu：查看CPU类型

df：查看磁盘空间

df –h 磁盘空间

df –I 索引空间

ps aux：系统上启的哪些服务

ps aux |grep “服务名” 查看服务是否开启

top：这台机器上资源占用的情况

systemctl disable irewalld ： 关闭防火墙

文件权限：

第一个字符：路径（D）还是文件（-）

R:读 都可以用-表示 也可以分别用数字表示4，拥有所有全是就是把三个权限加起来7

W:写 2

X:执行 1

分三组，每组三个字符，第一组：当前用户所属的权限；第二组：当前所属组的权限；第三组：其他用的权限。

## NodeJsAPI

Net:

const net=require('net'); 引用net模块

const socket=net.connect(12306,'127.0.0.1'); 创建连接，并连接到本机的12306端口。

createConnection和connect都是创建连接，createConnection底层用的是connect,所以直接用connect。

事件：

connect：客户端连接事件

socket.on('connect', () => {

console.log('客户端已连接到服务器');

});

data：是否有数据

socket.on('data', (data) => {

console.log(data.toString());

});

end：客户端关闭事件

socket.on('end', () => {

console.log('客户端关闭');

});

timeout：客户端超时事件

error：客户端报错事件

close：客户端关闭事件

属性：

remoteAddress：远端地址

remotePort：远端端口

localAddress：本地地址

loaclPort：本地端口

方法：

setTimeout：在 timeout 毫秒不活动之后将其设置为超时状态

write：发送数据

setEncoding：设置作为可读流（Readable Stream）的编码

end：关闭客户端

const sever = net.createServer(); 创建服务

事件：

listening：是否监听端口事件

sever.on('listening', () => {

console.log('服务已启动');

});

connection：连接事件

sever.on('connection', (scoket) => {

console.log('有新的连接');

});

close：服务器关闭事件（不常用）

error：服务器报错事件

方法：

sever.listen(12306,'127.0.0.1'); 监听地址及端口

sever.close() 服务器关闭

sever.address() 服务器地址需要在有回调的方法中使用

## 文件读取异步和同步的区别

异步用另外一个线程去执行，主线程继续执行,同步(sync)没执行完，后面的无法进行。

如果两件事没有依赖关系就尽量用异步，有必然联系的就用同步。

文件异步读写法:

fs.readFile(‘文件路径’，function(error,data){

有错误就没有数据，有数据就没有错误。

})

Options.mode:0o666 0o代表八进制

第一个数代表：文件所有者的权限

第二个数代表：同组用户的权限

第三个数代表：非同组用户的权限

R(可读) W(可写) X(可执行)

4 2 1