[Linux笔记](#_Toc515176666)

[Git笔记](#_Toc515176667)

[MongoDB笔记](#_Toc515176668)

[Nginx笔记](#_Toc515176669)

[NPM笔记](#_Toc515176670)

[Gulp笔记](#_Toc515176671)

[Bower笔记](#_Toc515176672)

[Gradle笔记](#_Toc515176673)

[JSON笔记](#_Toc515176674)

# Linux笔记

1. 指令

|  |  |  |
| --- | --- | --- |
| sudo | 以系统管理者的身份执行指令 |  |
| systemctl poweroff | 系统关机 |  |
| systemctl reboot | 重新启动 |  |
| systemctl suspend | 进入暂停模式 |  |
| systemctl hibernate | 进入休眠模式 |  |
| systemctl rescue | 强制进入救援模式 |  |
| systemctl emergency | 强制进入紧急救援模式 |  |
| systemctl set-default graphical.target |  |  |
| ps –l | 仅观察自己的 bash 相关进程： |  |
| ps aux | 观察系统所有进程： |  |
| cd .. | 返回上级目录 |  |
| rpm -q < package name> | 查询一个包是否被安装 |  |
| rpm -qa | sort | 查询系统安装的所有软件包 |  |
| rpm -ql< package name> | 列出软件包安装的文件**和位置** |  |
| rm –rf /home/test | 删除test目录下所有文件 | 包括test目录 |
| rm -rf /home/test/\* | 删除test目录下所有文件 | **保留**test目录  -f：强制删除;-r：递归的删除目录下面的文件以及子目录下文件 |
| rm –rf !(a|b) | 删除除a和b外所有文件，必须在当前目录下执行 | 如果提示“-bash: !: event not found ” shopt -s extglob 来解决 |
| find a/\* [-not] –name b | xargs rm –rf | 可选参数[-not]存在时，删除a文件下所有除b外的所有其他文件 | b可为文件夹名，若b为文件夹时，b底下的文件也会被删除 |
| find /usr/sam/\* -path  /usr/sam/dir1 –prune -o –print | 在/usr/sam目录下查找不在dir1子目录之内的所有文件 |  |
| mkdir 文件夹名称 | 新建文件夹 |  |
| vi文件名称 | 修改此文件 | 按i键进入编辑模式，按esc退出编辑模式；在浏览模式中输入:qw 保存退出，:q直接退出 |
| vi 新文件名 | 创建文本文件 |  |

2. 功能提示

* cd .. 返回上级目录

.表示当前目录；..表示上级目录；./aaa/ 表明的就是当前目录下面的aaa目录

* vncserver 启动vncserver服务

vncpasswd 修改vnc远程登录密码

systemctllist-unit-files | grep vnc 查看vnc服务

Systemctlenablevncserver@serivce 启用服务

* bash环境下通配符：



* bash环境下特殊符号：



* 管线命令





* 管线命令仅会处理 standard output，对于 standard error output 会予以忽略
* 管线命令必须要能够接受来自前一个指令的数据成为 standard input 继续处理才行。

# Git笔记

1. 指令

|  |  |  |
| --- | --- | --- |
| git config --global http.sslVerify false | 设置全局变量http.sslVerify为false |  |
| git pull | = git fetch + git merge |  |
| git push --force origin | 强制推送 |  |
| git reset --hard 8618818（版本号） | 回退到某一版本 |  |

1. 功能提示



* webstorm删除github上的文件
* 先commit到本地仓库，再push到GitHub上
* 强制覆盖本地文件（本地文件有更改）
* 本地先revert到上一个版本，然后git pull

# MongoDB笔记

下载地址参考：<https://www.mongodb.org/dl/win32/i386>

1. 安装 MongoDB服务
2. 新建数据库根文件夹mkdir c:\data
3. 新建存放数据和日志文件的目录mkdir c:\data\dbmkdir c:\data\log
4. 创建一个配置文件位于 C:\mongodb\mongod.cfg，内容如下

systemLog:

destination: file

path: c:\data\log\mongod.log

storage:

dbPath: c:\data\db

1. 通过执行mongod.exe，使用--install选项来安装服务，使用--config选项来指定之前创建的配置文件。

C:\mongodb\bin\mongod.exe --config "C:\mongodb\mongod.cfg" –install

1. 启动MongoDB服务net start MongoDB

关闭MongoDB服务net stop MongoDB

移除 MongoDB服务C:\mongodb\bin\mongod.exe --remove

1. 指令

|  |  |  |
| --- | --- | --- |
| mongod –dbpath数据库存放目录 | 启动数据库 | 在系统Shell下 |
| mongo | 进入后台管理Shell | 在系统Shell下 |
| use DATABASE\_NAME | 数据库存放目录 | 以下指令在后台管理Shell下 |
| show dbs | 查看所有数据库 |  |
| db.createCollection(name) | 新建集合 |  |
| db.dropDatabase() | 删除数据库 | 需先use切换入要删的数据库 |
| db.collection.drop() | 删除集合 |  |
| db.原集合名.renameCollection(新集合名) | 修改集合名称 |  |
| show collections | 查看所有集合 |  |

# Nginx笔记

1. CentOS 7上安装Nginx

1. 添加CentOS 7 EPEL仓库

sudo yum install epel-release

2. 安装Nginx

sudo yum install nginx

3. 启动Nginx

sudo systemctl start nginx

4. 如果您正在运行防火墙，请运行以下命令以允许HTTP和HTTPS通信：

sudo firewall-cmd --permanent --zone=public --add-service=http

sudo firewall-cmd --permanent --zone=public --add-service=https

sudo firewall-cmd --reload

5. 如果想在系统启动时启用Nginx。请输入以下命令：

sudo systemctl enable nginx

2. 指令

|  |  |  |
| --- | --- | --- |
| nginx –s reload | 重启命令 |  |
| nginx –t | 检查配置文件是否正确 |  |

3. 功能提示

* nginx 更改配置文件后需要重启，重启命令：nginx –s reload

# NPM笔记

1. 指令

|  |  |  |
| --- | --- | --- |
| npm config get prefix | 查看npm全局仓库路径 |  |
| npm config set prefix | 设置npm全局仓库路径 |  |
| npm view jquery versions | 查看jquery模块全部版本 |  |
| npm install gulp-sourcemaps --save | 安装gulp-sourcemaps模块并写入package.json |  |

2. 功能提示

* package.json版本(^和~区别)
* 波浪符号（~）：他会更新到当前minor version（也就是中间的那位数字）中最新的版本
* 插入符号（^）：将会把当前库的版本更新到当前major version（也就是第一位数字）中最新的版本

# Bower笔记

1. 指令

|  |  |  |
| --- | --- | --- |
| bower list | 列出当前安装的版本和最新版本，以及各个安装包的依赖 |  |
| bower info jquery | 查询jQuery有哪些版本 |  |
| bower install bootstrap --save | 安装bootstrap并更新至bower.json中的Dependencies |  |
| bower install bootstrap –save-dev | 安装bootstrap并更新至bower.json中的devDependencies |  |
| bower install --force | 强制安装 |  |

2. 功能提示

* bower.json文件解释

{

"name":"", //必须，如果需要注册包，则该包名唯一。

"description":"", //可选，包描述

"main":[], //可选，入口文件，bower本身不使用，供第三方构建工具会使用,每种文件类型只能有一个。

"ignore":[], //可选，文件或目录列表。bower安装的时候将忽略该列表中的文件,bower是从git仓库或压缩包下载一个包，里面的文件并不一定全部需要。

"dependencies":[], //依赖包，name:value，value可以是包的semverrange(版本号范围)，也可以直接是一个包的git地址或压缩包地址。

"devDependencies":[], //开发依赖包，仅仅在开发过程中测试或者编译文档用，部署生产环境是不需要。

"resolutions":[], //包引用冲突自动使用该模块指定的包版本,格式和dependencies 相同

"overrides" :[ //可以覆盖一个包中的默认设置，比如main里面设定的入口文件

"package-name":{ //这样可以根据需要，让第三方工具只打包需要的文件。

"main":[]

}

],

"moduleType":"", //可选，指定包采用那种模块化方式(globals,amd,node,es6,yui)

"private":Boolean, //是否公开发布当前包,如果只是使用bower来管理项目的包，设置为true.

"license":"", //授权方式(GPL-3.0,CC-BY-4.0.....)

"keywords":[], //可选，方便注册后容易被其他人搜索到。

"authors":[], //作者列表

"homepage":[], //主页，包介绍页

"repository":{ //包所在仓库。

"type": "git",

"url": "git://github.com/foo/bar.git"

},

｝

# Gulp笔记

1. 语法

* gulp.src(globs[, options])
* gulp.src (['./src/\*\*', '**!**\*\*/node\_modules/\*\*'])**→**感叹号表示不包括第二个参数代表的内容
* gulp.src ('./src/\*\*/\*.js', {base:'./src/'})**→**base作用：'./src/\*\*/\*.js'路径去除'./src/'得到的路径加到gulp.dest里的路径后面得出最终的目的路径

# Gradle笔记

# JSON笔记

1. 注意事项

* 基本规则
* JSON官网规定，不管是键或值都用**双引号**引起来
* json文件里的换行符\r\n
* json文件里不允许加注释
* 要描述的一个对象, (里面是这个对象的属性/函数!!) 那么用大括号, 表示对象; 如果你要描述的东西是多个并列的值, 那么用数组用中括号的方式!!!