下载安装 Git & Node.js

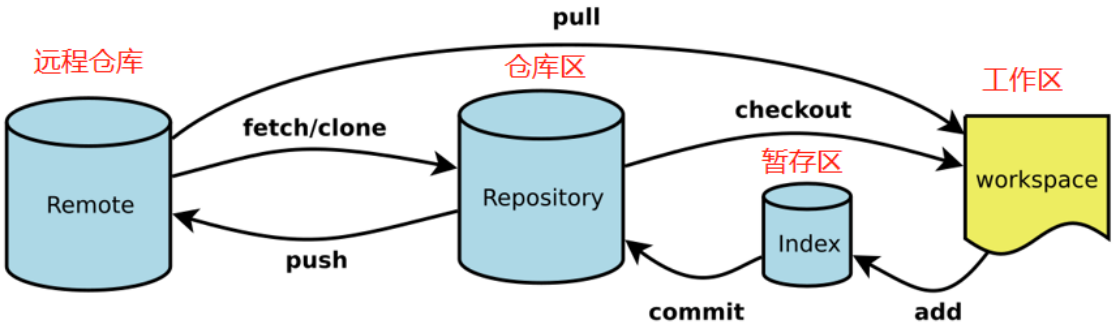
作者：胡萝卜
时间：2020-7-21
内容：

> **1. 下载安装 git，了解 git 的基本语句，创建一个仓库，并写一篇md文章描述 git，以及 git 的安装流程和上传仓库的流程，并把文件上传到 git 仓库中。**
> **2. 安装 nodejs，并用md写一篇文章简要介绍一下 node.js 以及安装流程。**
> **3.安装nginx，用md写一篇文章简要介绍nginx和安装配置流程。**

Part1 Git

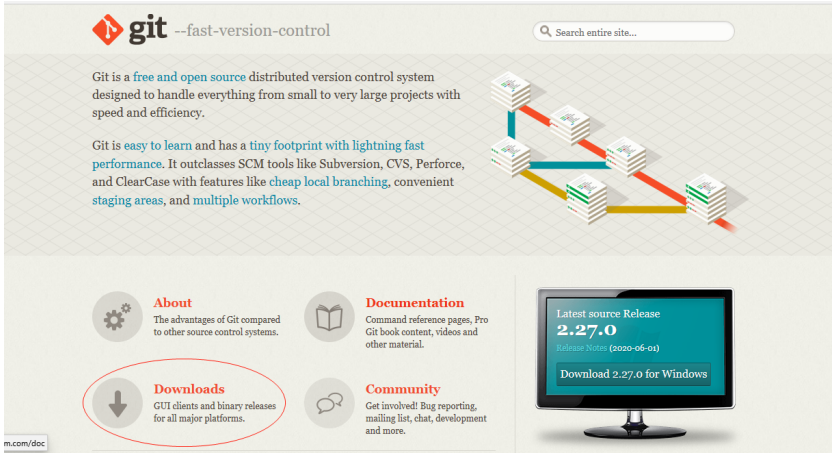
一、介绍

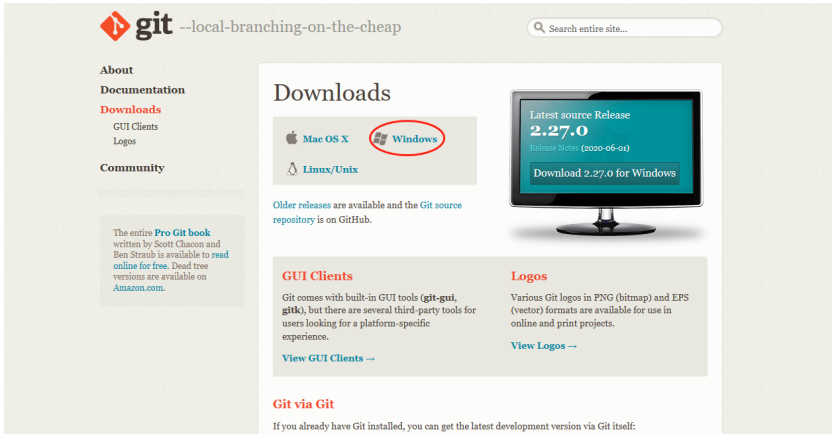
1. Git 是一个开源的分布式版本控制系统，用于敏捷高效地处理任何或小或大的项目。
2. Git 是 Linus Torvalds 为了帮助管理 Linux 内核开发而开发的一个开放源码的版本控制软件。
3. Git 与常用的版本控制工具 CVS, Subversion 等不同，它采用了分布式版本库的方式，不必服务器端软件支持。

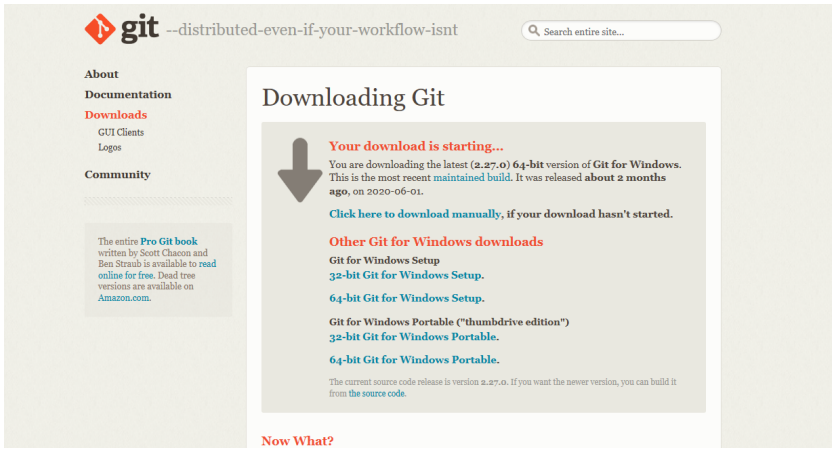


二、安装

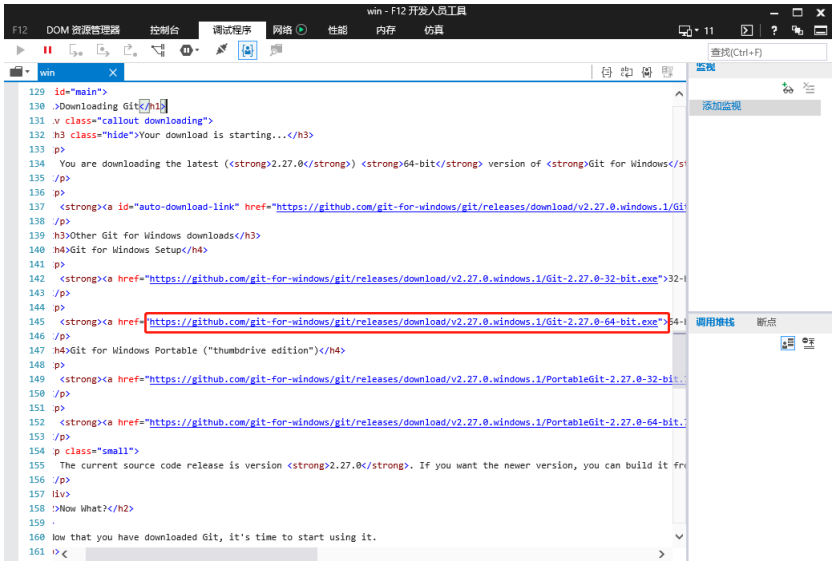
步骤1：进入 git 官网下载 https://git-scm.com

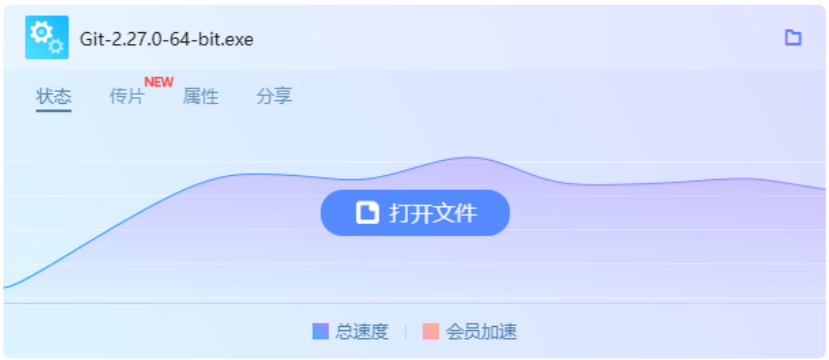






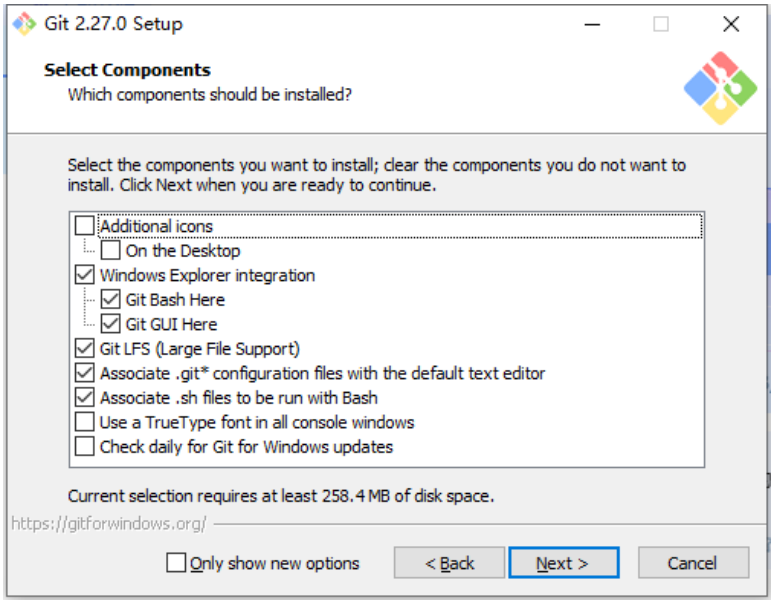
步骤2：若官网下载的 exe 无法运行，可转用迅雷下载

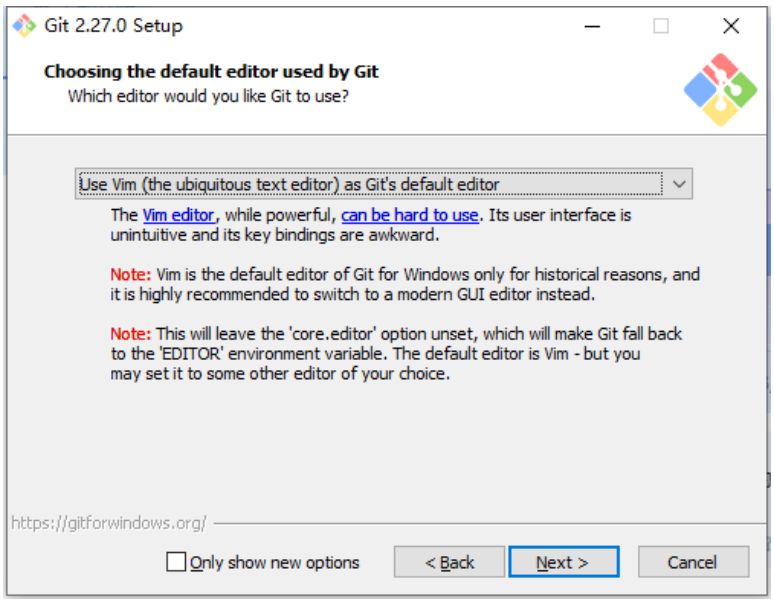


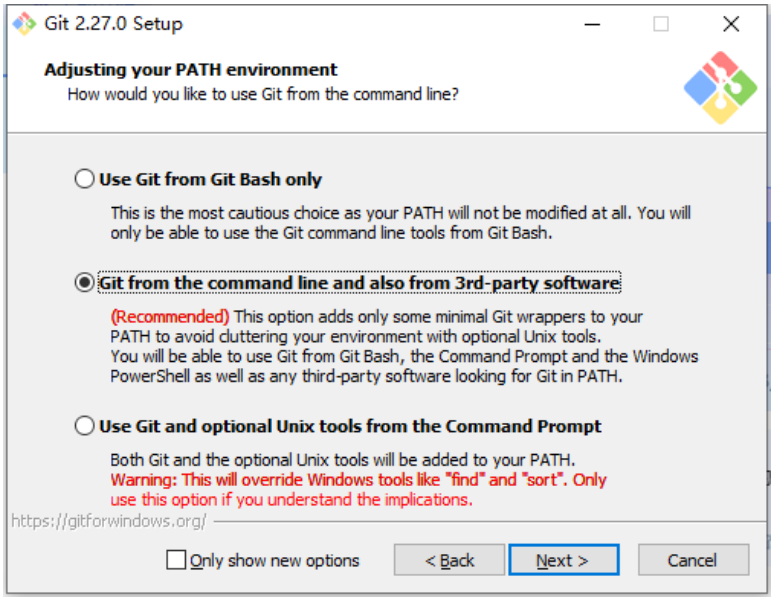


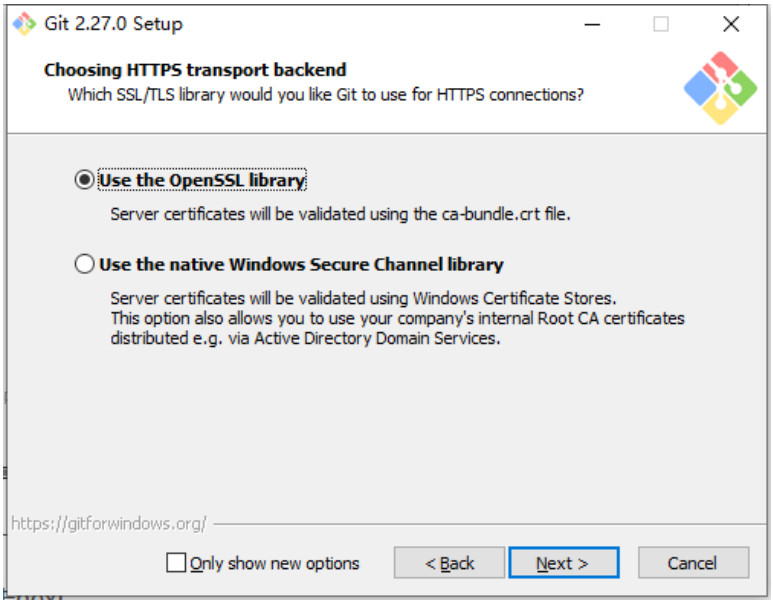
步骤3：开始安装 git

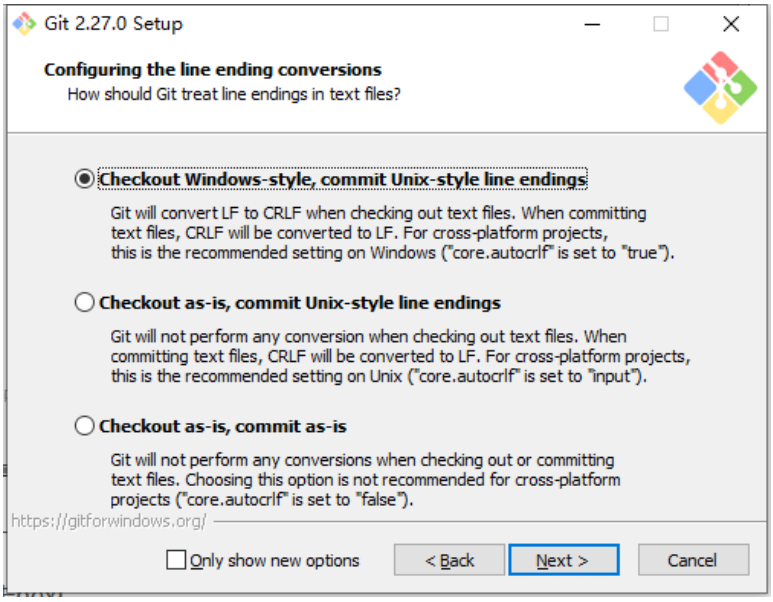

Git 2.27.0 Setup
Information
Please read the following important information before continuing.
When you are ready to continue with Setup, click Next.
GNU General Public License
Version 2, June 1991
Copyright (C) 1989, 1991 Free Software Foundation, Inc.
59 Temple Place - Suite 330, Boston, MA 02111-1307, USA
Everyone is permitted to copy and distribute verbatim copies
of this license document, but changing it is not allowed.
Preamble
The licenses for most software are designed to take away your
freedom to share and change it. By contrast, the GNU General Public
https://gitforwindows.org/
Only show new options
Next >
Cancel

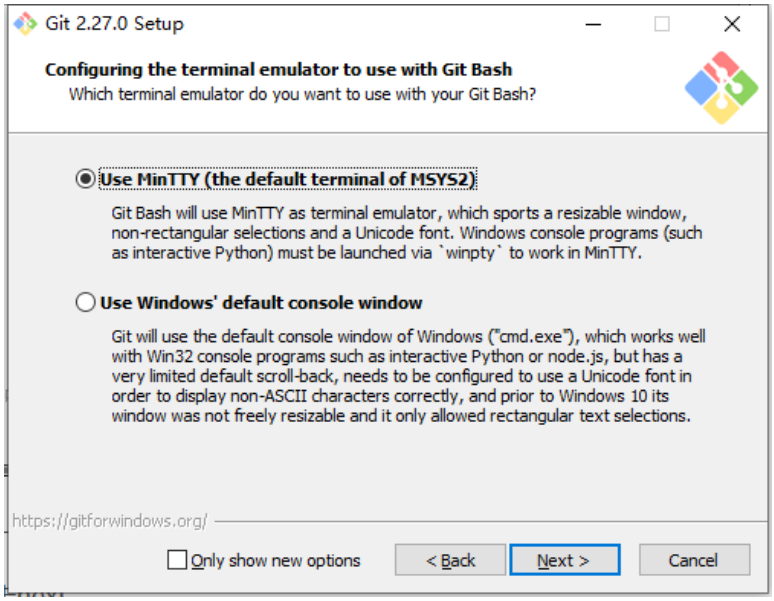

Git 2.27.0 Setup
Select Components
Which components should be installed?
Select the components you want to install; clear the components you do not want to install. Click Next when you are ready to continue.
Additional icons
On the Desktop
Windows Explorer integration
Git Bash Here
Git GUI Here
Git LFS (Large File Support)
Associate .git* configuration files with the default text editor
Associate .sh files to be run with Bash
Use a TrueType font in all console windows
Check daily for Git for Windows updates
Current selection requires at least 258.4 MB of disk space.
https://gitforwindows.org/
Only show new options
< Back
Next >
Cancel

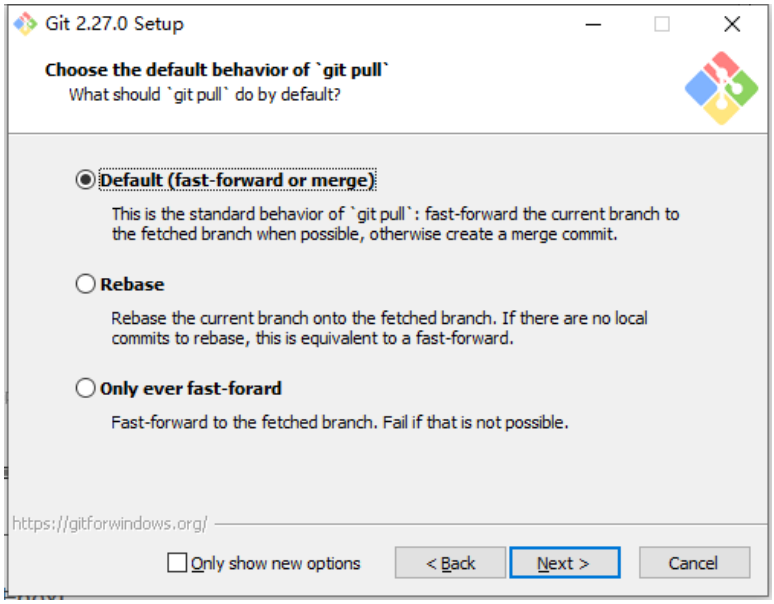

Git 2.27.0 Setup
Choosing the default editor used by Git
Which editor would you like Git to use?
Use Vim (the ubiquitous text editor) as Git's default editor
The Vim editor, while powerful, can be hard to use. Its user interface is unintuitive and its key bindings are awkward.
Note: Vim is the default editor of Git for Windows only for historical reasons, and it is highly recommended to switch to a modern GUI editor instead.
Note: This will leave the 'core.editor' option unset, which will make Git fall back to the 'EDITOR' environment variable. The default editor is Vim - but you may set it to some other editor of your choice.
https://gitforwindows.org/
Only show new options
< Back
Next >
Cancel

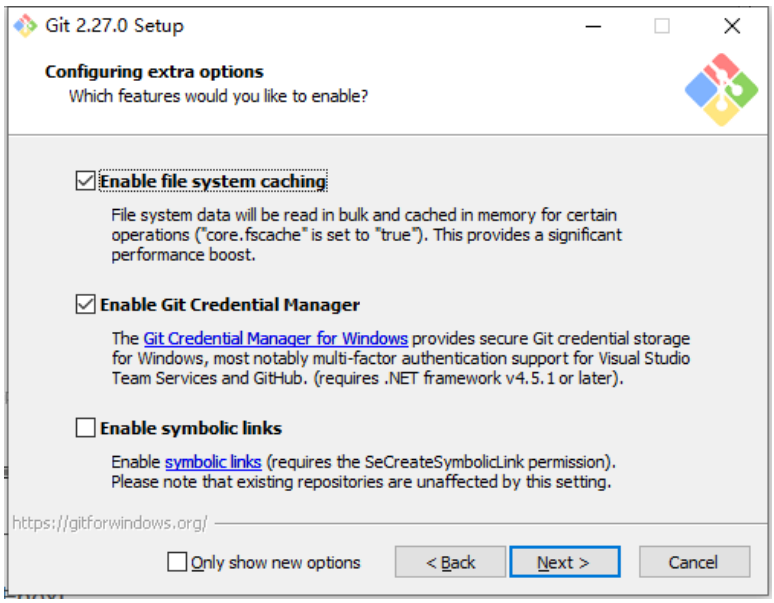

Git 2.27.0 Setup
Adjusting your PATH environment
How would you like to use Git from the command line?
Use Git from Git Bash only
This is the most cautious choice as your PATH will not be modified at all. You will only be able to use the Git command line tools from Git Bash.
Git from the command line and also from 3rd-party software
(Recommended) This option adds only some minimal Git wrappers to your PATH to avoid cluttering your environment with optional Unix tools.
You will be able to use Git from Git Bash, the Command Prompt and the Windows PowerShell as well as any third-party software looking for Git in PATH.
Use Git and optional Unix tools from the Command Prompt
Both Git and the optional Unix tools will be added to your PATH.
Warning: This will override Windows tools like "find" and "sort". Only use this option if you understand the implications.
https://gitforwindows.org/
Only show new options
< Back
Next >
Cancel

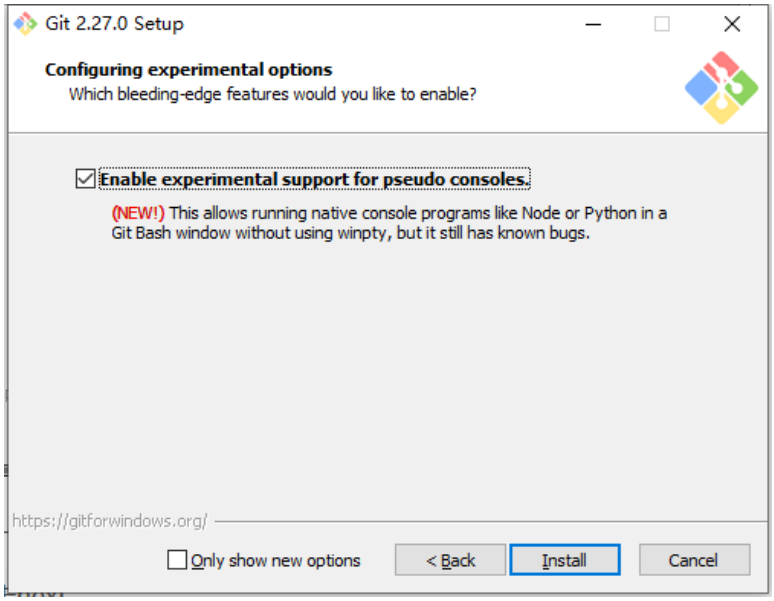

Git 2.27.0 Setup
Choosing HTTPS transport backend
Which SSL/TLS library would you like Git to use for HTTPS connections?
Use the OpenSSL library
Server certificates will be validated using the ca-bundle.crt file.
Use the native Windows Secure Channel library
Server certificates will be validated using Windows Certificate Stores.
This option also allows you to use your company's internal Root CA certificates distributed e.g. via Active Directory Domain Services.
https://gitforwindows.org/
Only show new options
< Back
Next >
Cancel


Git 2.27.0 Setup
Configuring the line ending conversions
How should Git treat line endings in text files?
Checkout Windows-style, commit Unix-style line endings
Git will convert LF to CRLF when checking out text files. When committing text files, CRLF will be converted to LF. For cross-platform projects, this is the recommended setting on Windows ("core.autocrlf" is set to "true").
Checkout as-is, commit Unix-style line endings
Git will not perform any conversion when checking out text files. When committing text files, CRLF will be converted to LF. For cross-platform projects, this is the recommended setting on Unix ("core.autocrlf" is set to "input").
Checkout as-is, commit as-is
Git will not perform any conversions when checking out or committing text files. Choosing this option is not recommended for cross-platform projects ("core.autocrlf" is set to "false").
https://gitforwindows.org/
Only show new options
< Back
Next >
Cancel

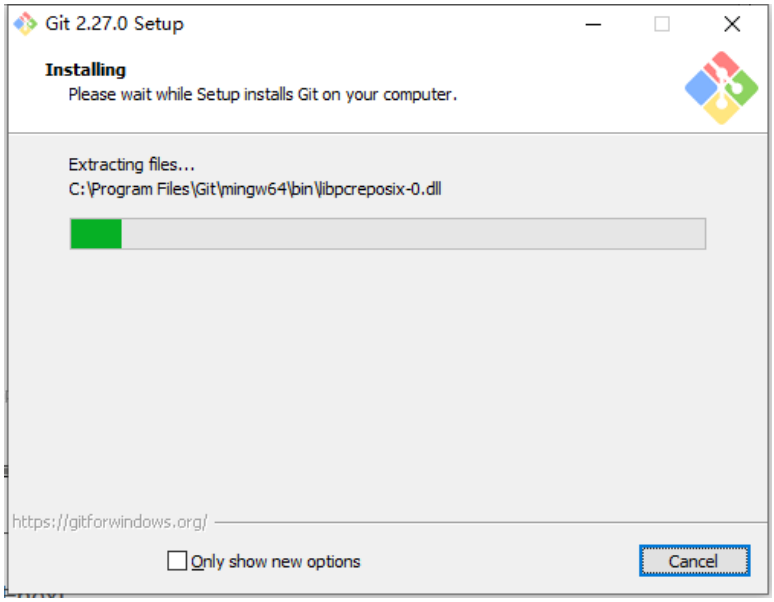

Git 2.27.0 Setup
Configuring the terminal emulator to use with Git Bash
Which terminal emulator do you want to use with your Git Bash?
Use MinTTY (the default terminal of MSYS2)
Git Bash will use MinTTY as terminal emulator, which sports a resizable window, non-rectangular selections and a Unicode font. Windows console programs (such as interactive Python) must be launched via `winpty` to work in MinTTY.
Use Windows' default console window
Git will use the default console window of Windows ("cmd.exe"), which works well with Win32 console programs such as interactive Python or node.js, but has a very limited default scroll-back, needs to be configured to use a Unicode font in order to display non-ASCII characters correctly, and prior to Windows 10 its window was not freely resizable and it only allowed rectangular text selections.
https://gitforwindows.org/
Only show new options
< Back
Next >
Cancel


Git 2.27.0 Setup
Choose the default behavior of `git pull`
What should `git pull` do by default?
Default (fast-forward or merge)
This is the standard behavior of `git pull`: fast-forward the current branch to the fetched branch when possible, otherwise create a merge commit.
Rebase
Rebase the current branch onto the fetched branch. If there are no local commits to rebase, this is equivalent to a fast-forward.
Only ever fast-forard
Fast-forward to the fetched branch. Fail if that is not possible.
https://gitforwindows.org/
Only show new options
< Back
Next >
Cancel

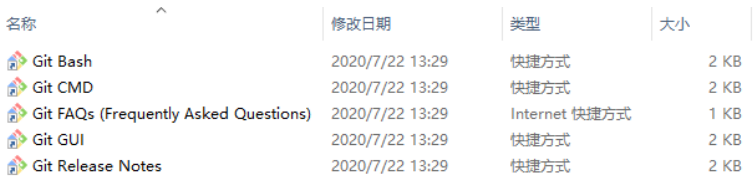

Git 2.27.0 Setup
Configuring extra options
Which features would you like to enable?
Enable file system caching
File system data will be read in bulk and cached in memory for certain operations ("core.fscache" is set to "true"). This provides a significant performance boost.
Enable Git Credential Manager
The Git Credential Manager for Windows provides secure Git credential storage for Windows, most notably multi-factor authentication support for Visual Studio Team Services and GitHub. (requires .NET framework v4.5.1 or later).
Enable symbolic links
Enable symbolic links (requires the SeCreateSymbolicLink permission). Please note that existing repositories are unaffected by this setting.
https://gitforwindows.org/
Only show new options
< Back
Next >
Cancel



步骤4：正在安装 git

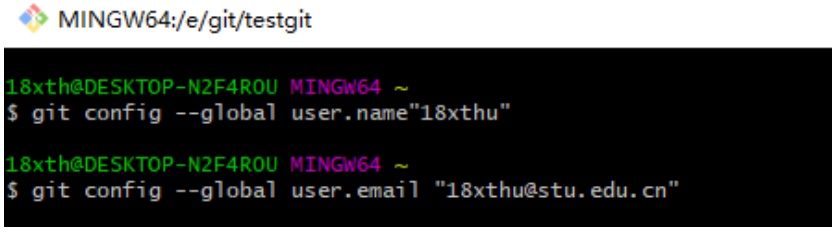


步骤5：git 安装完成



名称	修改日期	类型	大小
Git Bash	2020/7/22 13:29	快捷方式	2 KB
Git CMD	2020/7/22 13:29	快捷方式	2 KB
Git FAQs (Frequently Asked Questions)	2020/7/22 13:29	Internet 快捷方式	1 KB
Git GUI	2020/7/22 13:29	快捷方式	2 KB
Git Release Notes	2020/7/22 13:29	快捷方式	2 KB

步骤6：填写用户名和邮箱作为标识



三、基本操作

1. **git init**：初始化，把目录变成 git 可以管理的仓库。

2. **git add xxx(文件名)**：把文件添加到暂存区。
3. **git commit**：把文件提交到仓库。
4. **git status**：查看暂存区状态，是否有文件未提交。
5. **git diff xxx(文件名)**：查看文件修改的内容。
6. **git log**：查看历史记录。
7. **git reset --hard**：回退。
8. **git reflog**：获得命令的版本号。
9. **git checkout -- xx(文件名)**：可以丢弃工作区的修改，从而撤销命令。

四、上传仓库

步骤1：在 E盘 的 git 目录下新建一个 testgit 版本库

```
18xth@DESKTOP-N2F4ROU MINGW64 ~
$ cd E:

18xth@DESKTOP-N2F4ROU MINGW64 /e
$ cd git

18xth@DESKTOP-N2F4ROU MINGW64 /e/git
$ mkdir testgit

18xth@DESKTOP-N2F4ROU MINGW64 /e/git
$ cd testgit

18xth@DESKTOP-N2F4ROU MINGW64 /e/git/testgit
$ pwd
/e/git/testgit

18xth@DESKTOP-N2F4ROU MINGW64 /e/git/testgit
$
```

步骤2：通过命令 git init 把这个目录变成 git 可以管理的仓库

```
18xth@DESKTOP-N2F4ROU MINGW64 /e/git/testgit
$ git init
Initialized empty Git repository in E:/git/testgit/.git/

18xth@DESKTOP-N2F4ROU MINGW64 /e/git/testgit (master)
$
```

步骤3：在版本库 testgit 目录下新建一个记事本文件 readme.txt，内容为：11111111

```
18xth@DESKTOP-N2F4ROU MINGW64 /e/git/testgit (master)
$ touch readme.txt
```

```
18xth@DESKTOP-N2F4ROU MINGW64 /e/git/testgit (master)
$ cat readme.txt
1111111111
```

步骤4：使用命令 git add readme.txt 添加到暂存区里面去

```
18xth@DESKTOP-N2F4ROU MINGW64 /e/git/testgit (master)
$ git add readme.txt
```

步骤5：用命令 git commit 告诉 git，把文件提交到仓库

```
18xth@DESKTOP-N2F4ROU MINGW64 /e/git/testgit (master)
$ git commit -m "readme.txt提交"
[master (root-commit) 6d570b7] readme.txt鎖愜氮
 1 file changed, 1 insertion(+)
 create mode 100644 readme.txt
```

步骤6：通过命令 git status 来查看是否还有文件未提交
若无，证明刚才的 readme.txt 文件已经提交

```
18xth@DESKTOP-N2F4ROU MINGW64 /e/git/testgit (master)
$ git status
On branch master
nothing to commit, working tree clean
```

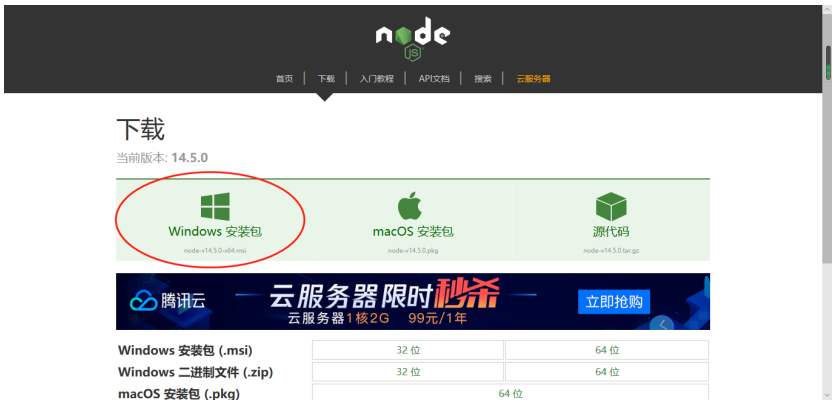
Part2 Node.js

一、介绍

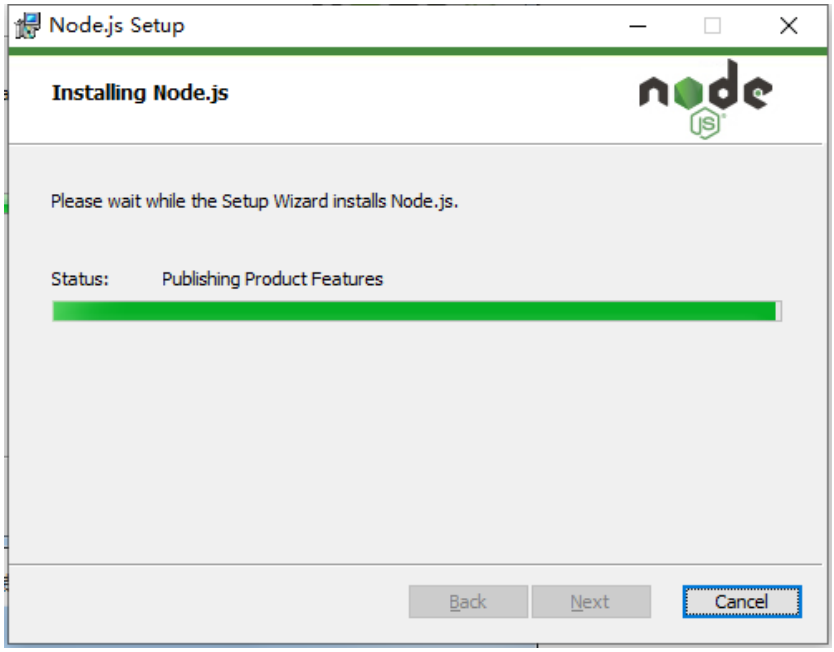
1. Node.js 是一个基于Chrome JavaScript 运行时建立的一个平台。
2. Node.js是一个事件驱动I/O服务端JavaScript环境，基于Google的V8引擎，V8引擎执行Javascript的速度非常快，性能非常好。
3. **简单的说 Node.js 就是运行在服务端的 JavaScript。**

二、安装

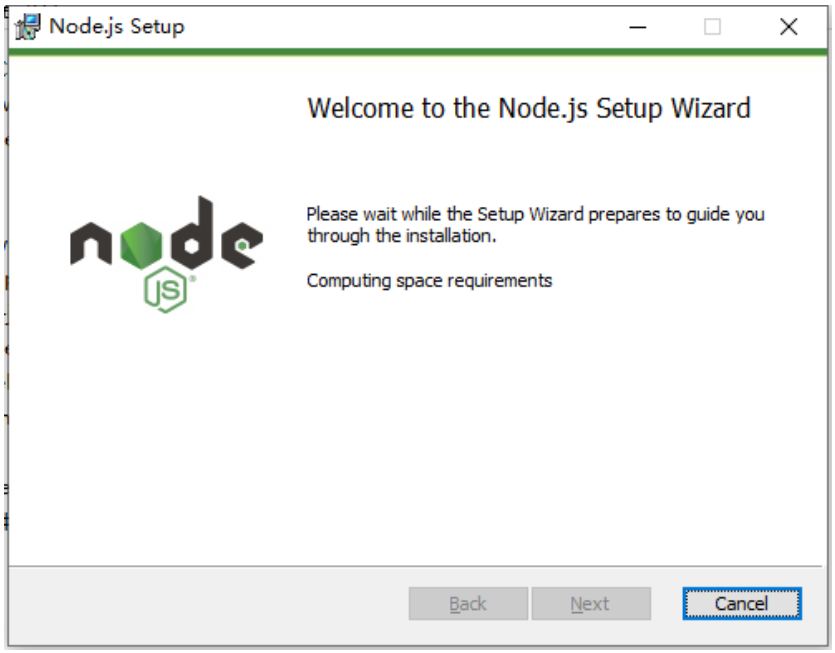
步骤1：进入 node.js 官网下载 http://nodejs.cn/

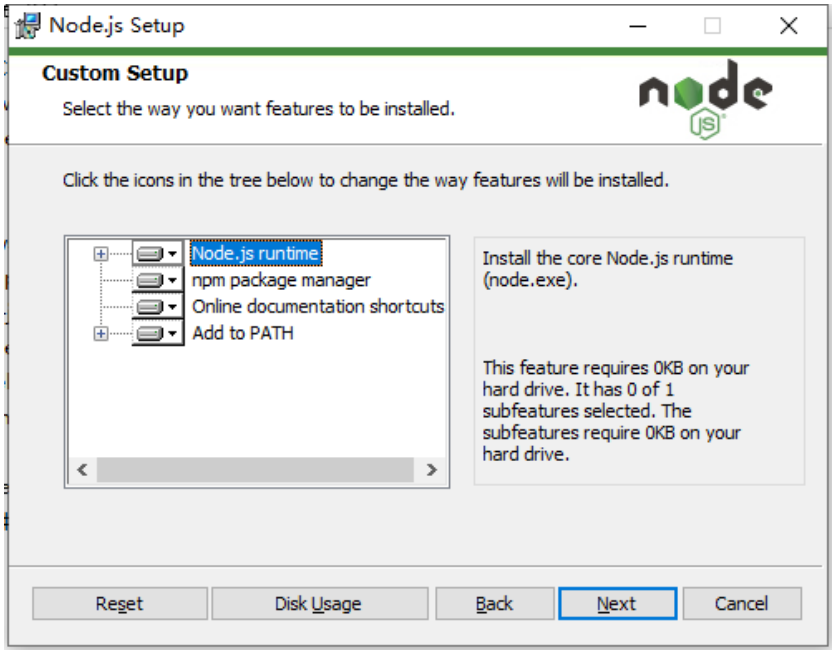


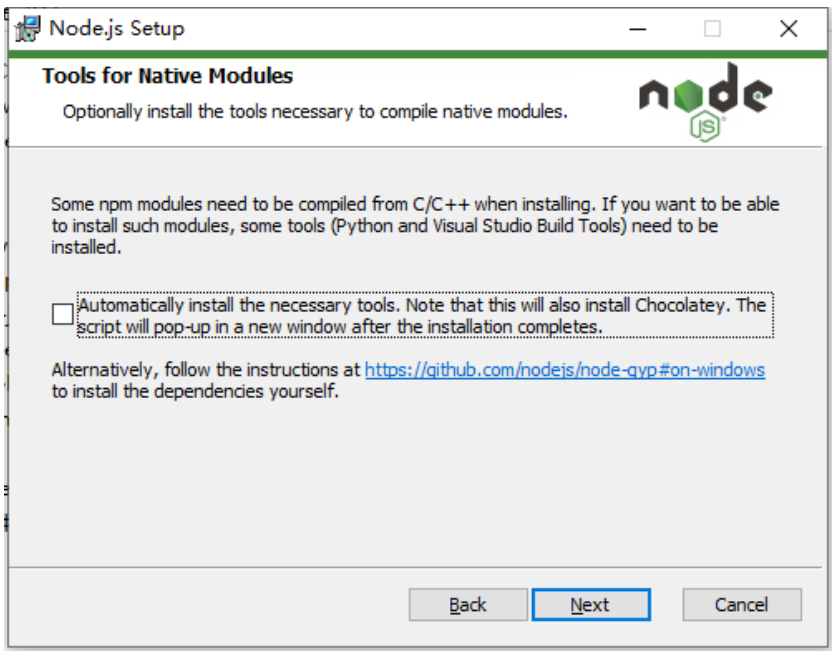
步骤2：运行下载的 node-v14.5.0-x64.msi



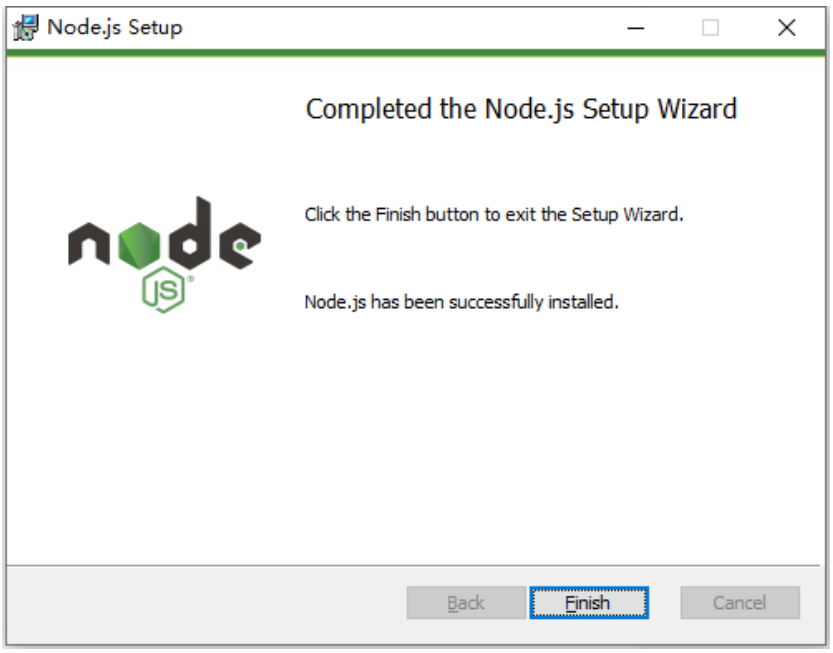
步骤3：开始安装 node.js



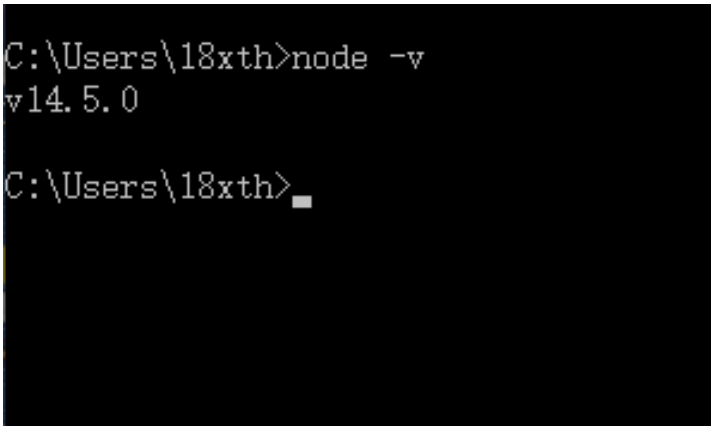




步骤4：node.js 安装完成



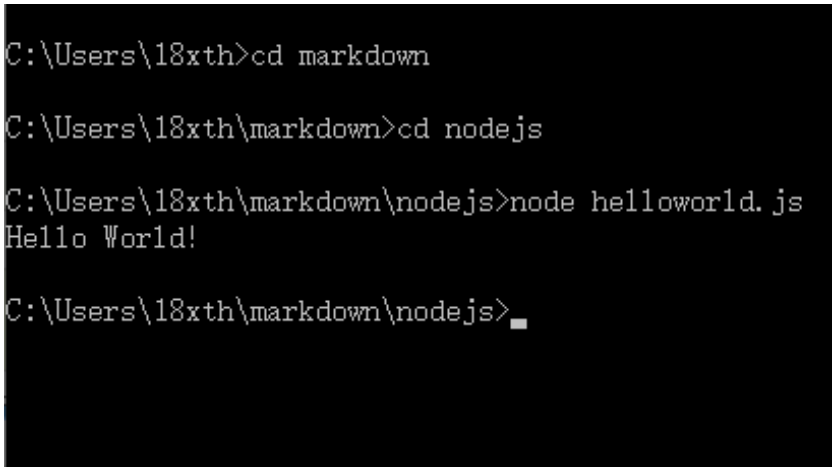
步骤5：在终端查看 node.js 的版本



三、运行程序

方式1：脚本模式——在 node.js 控制端口运行

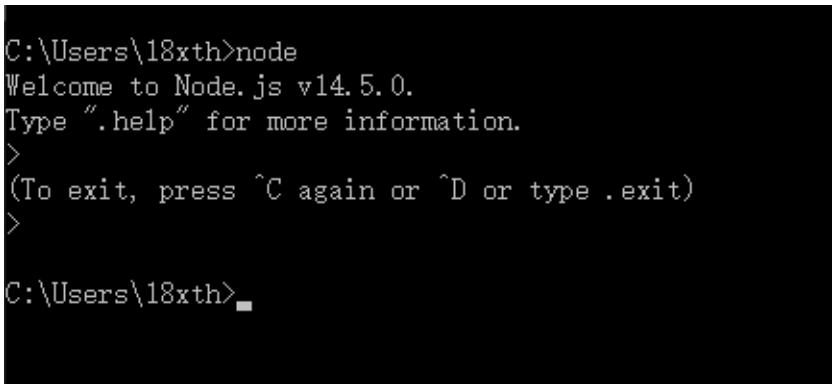
方式2：使用 node 运行来 .js 文件



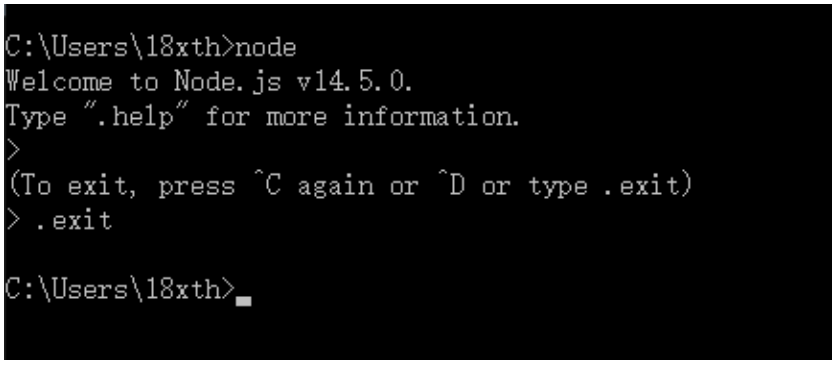
方式3：交互模式——在终端输入 node 进入命令交互模式，输入一条代码语句后立即执行并显示结果

四、退出 node 交互模式

方法1：两次输入 Ctrl -C



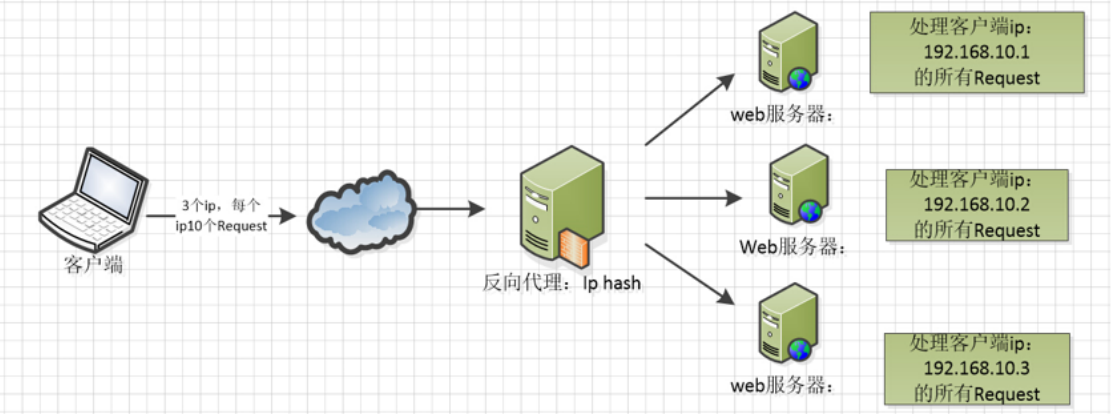
方法2：先输入Ctrl -C，再输入.exit



Part3 Nginx

一、介绍

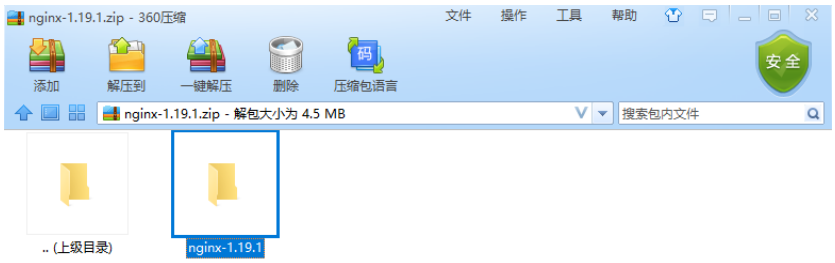
1. Nginx("engine x") 是一款是由俄罗斯的程序设计师 Igor Sysoev 所开发高性能的 Web和 反向代理服务器，也是一个 IMAP/POP3/SMTP 代理服务器。
2. 在高连接并发的情况下，Nginx 是 Apache 服务器不错的替代品。
3. Nginx 也是一种轻量级的 Web服务器，可以作为独立的服务器部署网站（类似Tomcat）。它高性能和低消耗内存的结构受到很多大公司青睐，如淘宝网站架设。



二、安装

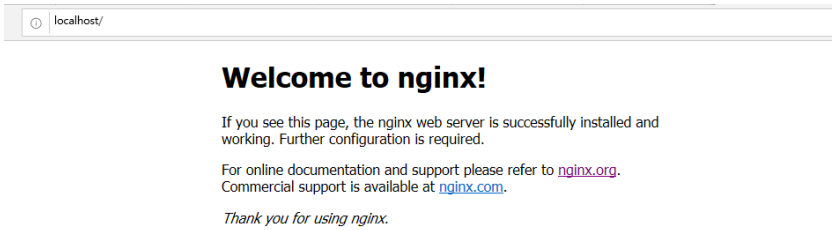
步骤1：进入 nginx 官网下载　http://nginx.org/

步骤2：解压 nginx-1.19.1 包

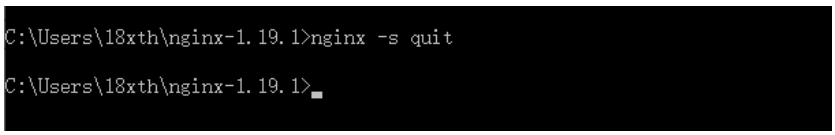


步骤3：终端启动 nginx

步骤4：打开浏览器访问刚才的域名及端口 http://localhost:80 检查是否成功



步骤5：完整有序的关闭 nginx



三、常用命令

1. **start nginx**：启动nginx。
2. **nginx -s quit**：优雅停止nginx，有连接时会等连接请求完成再杀死worker进程。
3. **nginx -s reload**：优雅重启，并重新载入配置文件nginx.conf。
4. **nginx -s reopen**：重新打开日志文件，一般用于切割日志。
5. **nginx -v**：查看版本。
6. **nginx -t**：检查nginx的配置文件。
7. **nginx -h**：查看帮助信息。
8. **nginx -c filename**：指定配置文件。

四、配置文件

1. 文件结构

```
...                 #全局块
```

```
events {         #events块
   ...
}

http      #http块
{
    ...   #http全局块
    server        #server块
    {
        ...       #server全局块
        location [PATTERN]   #location块
        {
            ...
        }
        location [PATTERN]
        {
            ...
        }
    }
    server
    {
      ...
    }
    ...     #http全局块
}
```

- **全局块**：配置影响nginx全局的指令。一般有运行nginx服务器的用户组，nginx进程pid存放路径，日志存放路径，配置文件引入，允许生成worker process数等。
- **events块**：配置影响nginx服务器或与用户的网络连接。有每个进程的最大连接数，选取哪种事件驱动模型处理连接请求，是否允许同时接受多个网路连接，开启多个网络连接序列化等。
- **http块**：可以嵌套多个server，配置代理，缓存，日志定义等绝大多数功能和第三方模块的配置。如文件引入，mime-type定义，日志自定义，是否使用sendfile传输文件，连接超时时间，单连接请求数等。
- **server块**：配置虚拟主机的相关参数，一个http中可以有多个server。
- **location块**：配置请求的路由，以及各种页面的处理情况。

2. 默认的 nginx 配置文件 nginx.conf

```
#user  nobody;
worker_processes  1;

#error_log  logs/error.log;
#error_log  logs/error.log  notice;
#error_log  logs/error.log  info;

#pid        logs/nginx.pid;


events {
    worker_connections  1024;
}
```

```
http {
    include       mime.types;
    default_type  application/octet-stream;

    #log_format  main  '$remote_addr - $remote_user [$time_local] "$request" '
    #                  '$status $body_bytes_sent "$http_referer" '
    #                  '"$http_user_agent" "$http_x_forwarded_for"';

    #access_log  logs/access.log  main;

    sendfile        on;
    #tcp_nopush     on;

    #keepalive_timeout  0;
    keepalive_timeout  65;

    #gzip  on;

    server {
        listen       80;
        server_name  localhost;

        #charset koi8-r;

        #access_log  logs/host.access.log  main;

        location / {
            root   html;
            index  index.html index.htm;
        }

        #error_page  404              /404.html;

        # redirect server error pages to the static page /50x.html
        #
        error_page   500 502 503 504  /50x.html;
        location = /50x.html {
            root   html;
        }

        # proxy the PHP scripts to Apache listening on 127.0.0.1:80
        #
        #location ~ \.php$ {
        #    proxy_pass   http://127.0.0.1;
        #}

        # pass the PHP scripts to FastCGI server listening on 127.0.0.1:9000
        #
        #location ~ \.php$ {
        #    root           html;
        #    fastcgi_pass   127.0.0.1:9000;
        #    fastcgi_index  index.php;
        #    fastcgi_param  SCRIPT_FILENAME  /scripts$fastcgi_script_name;
        #    include        fastcgi_params;
        #}

        # deny access to .htaccess files, if Apache's document root
```

```
        # concurs with nginx's one
        #
        #location ~ /\.ht {
        #    deny  all;
        #}
    }


    # another virtual host using mix of IP-, name-, and port-based configuration
    #
    #server {
    #    listen       8000;
    #    listen       somename:8080;
    #    server_name  somename  alias  another.alias;

    #    location / {
    #        root   html;
    #        index  index.html index.htm;
    #    }
    #}


    # HTTPS server
    #
    #server {
    #    listen       443 ssl;
    #    server_name  localhost;

    #    ssl_certificate      cert.pem;
    #    ssl_certificate_key  cert.key;

    #    ssl_session_cache    shared:SSL:1m;
    #    ssl_session_timeout  5m;

    #    ssl_ciphers  HIGH:!aNULL:!MD5;
    #    ssl_prefer_server_ciphers  on;

    #    location / {
    #        root   html;
    #        index  index.html index.htm;
    #    }
    #}
}
```

参考资料

Git
Node.js
Nginx

(完)