

Yarn学习笔记

书栈(BookStack.CN)

目 录

致谢

概览

理解概念

使用流程

命令列表

从NPM客户端迁移

Yarn命令 vs NPM命令

各种依赖类型

致谢

当前文档《Yarn学习笔记》由 进击的皇虫 使用 书栈(BookStack.CN) 进行构建,生成于 2018-07-21。

书栈(BookStack.CN) 仅提供文档编写、整理、归类等功能,以及对文档内容的生成和导出工具。

文档内容由网友们编写和整理,书栈(BookStack.CN) 难以确认文档内容知识点是否错漏。如果您在阅读文档获取知识的时候,发现文档内容有不恰当的地方,请向我们反馈,让我们共同携手,将知识准确、高效且有效地传递给每一个人。

同时,如果您在日常工作、生活和学习中遇到有价值有营养的知识文档,欢迎分享到 书栈(BookStack.CN),为知识的传承献上您的一份力量!

如果当前文档生成时间太久,请到 书栈(BookStack.CN) 获取最新的文档,以跟上知识更新换代的步伐。

文档地址: <http://www.bookstack.cn/books/learning-yarn>

书栈官网: <http://www.bookstack.cn>

书栈开源: <https://github.com/TruthHun>

分享,让知识传承更久远! 感谢知识的创造者,感谢知识的分享者,也感谢每一位阅读到此处的读者,因为我们都将成为知识的传承者。

概览

Yarn学习笔记。

Yarn作为npm客户端，具有明显的优势：1、很快速；2、很安全；3、超可靠。

- 作者：Shellway Ho，转载请注明出处！
- 来源：<http://docs.shellway.cn/learning-yarn/>

理解概念

yarn: /jɑ:n/, 纱线, 奇谈, 故事。

Yarn为Node.js平台的代码包管理器。类似于知名的npm包管理器，实际是npm客户端。

特点：快速、安全、可靠。1、离线模式：依赖包只要被装过一次，就会被缓存到本机，再次安装时直接从缓存中读取。2、高确定性：无论安装顺序如何，同样的依赖包都将会以完全一样的方式安装。3、网络性能优异：将请求高效队列化，

Yarn可以将安装时间从数分钟减少至几秒钟。Yarn还兼容nom注册表，但包安装方法有所区别。其使用了lockfiles和一个决定性安装算法，能够为参与一个项目的所有用户维持相同的节点模块（node_modules）目录结构，有助于减少难以追踪的bug和在多台机器上复制。

Yarn官网地址：<https://yarnpkg.com/>

- 作者：Shellway Ho，转载请注明出处！
- 来源：<http://docs.shellway.cn/learning-yarn/>

使用流程

安装

MacOS

Homebrew

1. `brew update` ##我的机器上无需这一步，Homebrew在install之前会自动update
2. `brew install yarn`

Windows

- 1、下载 `.smi` 按装包。确保Node.js已经安装并可用。>[下载Yarn安装包](#)
- 2、通过Chocolatey安装Chocolatey是Windows平台下的包管理器。安装好后打开命令行，执行下面的命令：
`choco install yarn`

Linux

1、Debian/Ubuntu Linux

编辑软件源：

1. `curl -sS https://dl.yarnpkg.com/debian/pubkey.gpg | sudo apt-key add -`
2. `echo "deb https://dl.yarnpkg.com/debian/ stable main" | sudo tee /etc/apt/sources.list.d/yarn.list`

执行安装：

```
sudo apt-get update && sudo apt-get install yarn
```

2、CentOS / Fedora / RHEL

通过RPM包安装：

```
sudo wget https://dl.yarnpkg.com/rpm/yarn.repo -O /etc/yum.repos.d/yarn.repo
```

如果没有Node.js，先配置一下：

```
curl -s -L https://rpm.nodesource.com/setup_6.x | bash -
```

然后再用yum安装：

```
sudo yum install yarn
```

3、Arch Linux

```
yaourt -S yarn
```

4、Solus

```
sudo eopkg install yarn
```

最后检查安装是否成功：

```
yarn -version
```

换上淘宝源，加速安装过程

```
1. $ yarn config set registry "https://registry.npm.taobao.org"
```

Yarn版本升级

运行以下命令，可以升级Yarn：

```
1. $ curl -o- -L https://yarnpkg.com/install.sh | bash
```

常用使用流程

- 创建新项目 - Creating a new project
- 添加、更新、移除依赖包 - Adding/updating/removing dependencies
- 安装、重装依赖包 - Installing/reinstalling your dependencies
- 使用版本控制系统，如git - Working with version control (i.e. git)
- 持续集成 - Continuous Integration

开启新项目

```
yarn init
```

管理依赖包

1、添加依赖

```
1. yarn add [package] # 通过名称添加一个依赖包
2. yarn add [package]@[version] #"包名@版本号"格式
3. yarn add [package]@[tag] #"包名@标签"格式
```

2、更新依赖包

```
1. yarn upgrade [package]
2. yarn upgrade [package]@[version]
3. yarn upgrade [package]@[tag]
```

3、删除依赖包

```
yarn remove [package]
```

4、安装项目依赖包

1. yarn
2. yarn install

就是这么简单，就是这么任性~~

1. 好吧，来复杂点的
2. - - - - -
3. 1、安装（`package.json`中）所有依赖包：yarn **or** yarn install
4. 2、安装依赖包的单版本（仅安装一个版本）：yarn install --flat
5. 3、强制所有包都预下载：yarn install --force
6. 4、仅安装生产环境依赖包：yarn install --production

嗯，就这样！

关于怎么为Yarn世界做贡献，请参考[官网](#)！

- 作者：Shellway Ho，转载请注明出处！
- 来源：<http://docs.shellway.cn/learning-yarn/>

命令列表

常用命令：

创建项目： `yarn init`

安装依赖包： `yarn` == `yarn install`

添加依赖包： `yarn add`

Yarn命令列表

命令	操作	参数	标签	
yarn add	添加依赖包	包名	-dev/-D	
yarn bin	显示yarn安装目录	无	无	
yarn cache	显示缓存	列出缓存包： <code>ls</code> ，打出缓存目录路径： <code>dir</code> ，清除缓存： <code>clean</code>	无	
yarn check	检查包			
yarn clean	清理不需要的依赖文件			
yarn config	配置	设置： <code>set <key> <value></code> ，删除： <code>delete</code> ，列出： <code>list</code>	[-g \	-global]
yarn generate-lock-entry	生成锁定文件	无	无	
yarn global	全局安装依赖包	yarn global [-prefix]	-prefix 包路径前缀	
yarn info	显示依赖包的信息	包名	-json: json格式显示结果	
yarn init	交互式创建/更新 package.json 文件	无	-yes/-y: 以默认值生成package.json文件	
yarn install	安装所有依赖包		-flat: 只安装一个版本；-force: 强制重新下载安装；-har: 输出安装时网络性能日志；-no-lockfile: 不生成 yarn.lock文件；-production: 生产模式安装（不安装 devDependencies中的依赖）	
yarn licenses	列出已安装依赖包的证书	ls: 证书列表；generate-disclaimer: 生成免责声明		

yarn link	开发时链接依赖包，以便在其他项目中使用	包名	
yarn login	保存你的用户名、邮箱		
yarn logout	删除你的用户名、邮箱		
yarn list	列出已安装依赖包		-depth=0: 列表深度，从0开始
yarn outdated	检查过时的依赖包	包名	
yarn owner	管理拥有者	ls/add/remove	
yarn pack	给包的依赖打包	-filename	
yarn publish	将包发布到npm		-tag: 版本标签；-access: 公开(public)还是限制的(restricted)
yarn remove	卸载包，更新package.json和yarn.lock	包名	
yarn run	运行package.json中预定义的脚本		
yarn self-update	yarn自身更新-未实现		
yarn tag	显示包的标签	add/rm/ls	
yarn team	管理团队	create/destroy/add/rm/ls	
yarn test	测试 = yarn run test		
yarn unlink	取消链接依赖包		
yarn upgrade	升级依赖包		
yarn version	管理当前项目的版本号	-new-version : 直接记录版本号；-no-git-tag-version: 不生成git标签	
yarn why	分析为什么需要安装依赖包	包名/包目录/包目录中的文件名	-

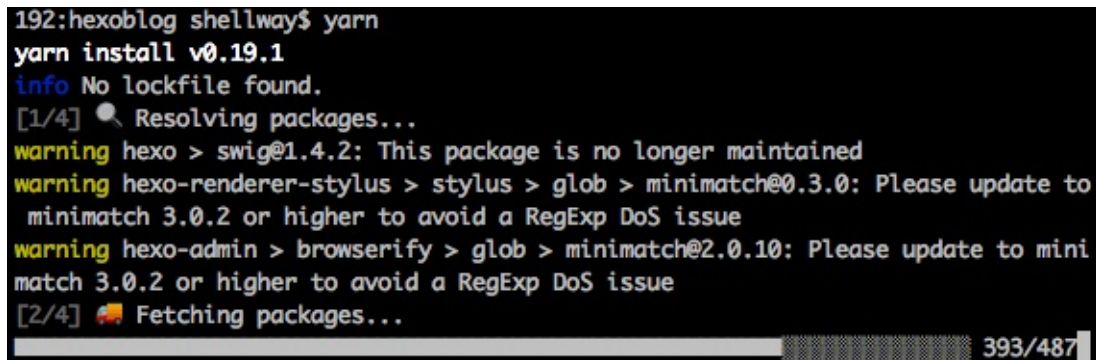
- 作者: Shellway Ho, 转载请注明出处！
- 来源: <http://docs.shellway.cn/learning-yarn/>



从NPM客户端迁移

大多情况，从npm客户端迁移至Yarn客户端都是很方便的。package.json文件依然可用，你只要在已有目录执行以下命令即可：

```
1. $ yarn
```



```
192:hexoblog shellway$ yarn
yarn install v0.19.1
info No lockfile found.
[1/4] 🔍 Resolving packages...
warning hexo > swig@1.4.2: This package is no longer maintained
warning hexo-renderer-stylus > stylus > glob > minimatch@0.3.0: Please update to
  minimatch 3.0.2 or higher to avoid a RegExp DoS issue
warning hexo-admin > browserify > glob > minimatch@2.0.10: Please update to mini
  match 3.0.2 or higher to avoid a RegExp DoS issue
[2/4] 📦 Fetching packages...
393/487
```

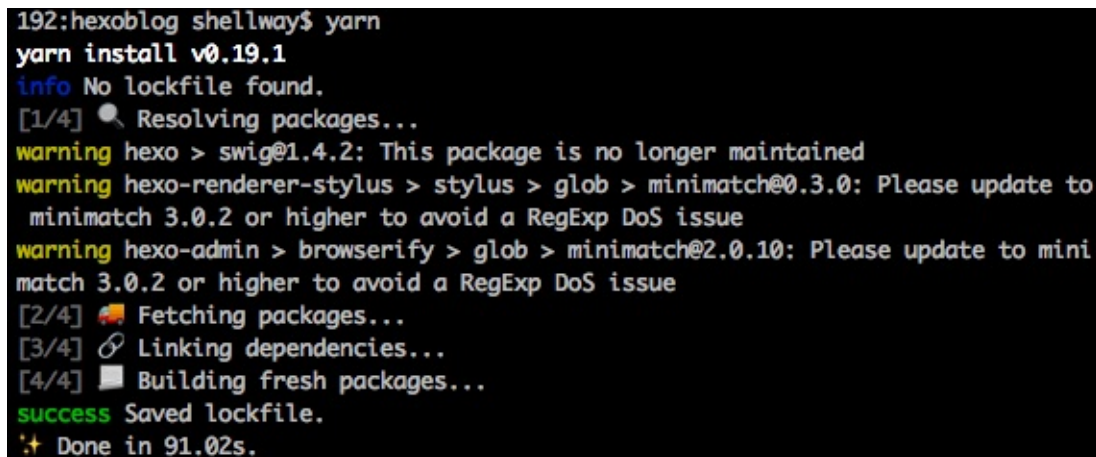
要添加依赖，你可以使用：

```
1. $ yarn add <package>
```

执行以上两个命令后，yarn都会在项目根目录创建或更新yarn.lock文件。如果你要将项目发布或者共享给同事，请确保包含yarn.lock文件。

执行以上两个命令中的任一个，项目都会转换成yarn客户端。So easy，对吧？！

执行完成如图：



```
192:hexoblog shellway$ yarn
yarn install v0.19.1
info No lockfile found.
[1/4] 🔍 Resolving packages...
warning hexo > swig@1.4.2: This package is no longer maintained
warning hexo-renderer-stylus > stylus > glob > minimatch@0.3.0: Please update to
  minimatch 3.0.2 or higher to avoid a RegExp DoS issue
warning hexo-admin > browserify > glob > minimatch@2.0.10: Please update to mini
  match 3.0.2 or higher to avoid a RegExp DoS issue
[2/4] 📦 Fetching packages...
[3/4] 🔗 Linking dependencies...
[4/4] 📦 Building fresh packages...
success Saved lockfile.
🎉 Done in 91.02s.
```

执行成功

```
192:hexoblog shellway$ yarn
yarn install v0.19.1
info No lockfile found.
[1/4] 🔍 Resolving packages...
warning hexo > swig@1.4.2: This package is no longer maintained
warning hexo-renderer-stylus > stylus > glob > minimatch@0.3.0: Please update to
minimatch 3.0.2 or higher to avoid a RegExp DoS issue
warning hexo-admin > browserify > glob > minimatch@2.0.10: Please update to mini
match 3.0.2 or higher to avoid a RegExp DoS issue
[2/4] 📦 Fetching packages...
[3/4] 🔗 Linking dependencies...
error An unexpected error occurred: "ENOENT: no such file or directory, open '/U
sers/shellway/playground/hexoblog/node_modules/astw/node_modules/acorn/package.j
son'".
info If you think this is a bug, please open a bug report with the information p
rovided in "/Users/shellway/playground/hexoblog/yarn-error.log".
info Visit https://yarnpkg.com/en/docs/cli/install for documentation about this
command.
```

执行报错

如果有错误，yarn会给你生成一个yarn-error.log，是不是很Nice，很暖男呢？

这时，你可以先删除node_modules目录，再次执行 `$ yarn`。

- 作者：Shellway Ho，转载请注明出处！
- 来源：<http://docs.shellway.cn/learning-yarn/>

Yarn命令 vs NPM命令

NPM和Yarn命令对比

npm 命令	Yarn 命令	备注
npm install	yarn install	安装依赖包
(N/A)	yarn install --flat	单版本模式
(N/A)	yarn install --har	生成har文件，记录安装时网络请求性能
(N/A)	yarn install --no-lockfile	不读写lockfile方式
(N/A)	yarn install --pure-lockfile	不生成yarn.lock文件
npm install [package]	(N/A)	安装依赖
npm install --save [package]	yarn add [package]	添加生产模式依赖到项目
npm install --save-dev [package]	yarn add [package] [--dev/-D]	添加开发模式的依赖
(N/A)	yarn add [package] [--peer/-P]	对等模式添加依赖，发布/分享项目时的依赖
npm install --save-optional [package]	yarn add [package] [--optional/-O]	
npm install --save-exact [package]	yarn add [package] [--exact/-E]	
(N/A)	yarn add [package] [--tilde/-T]	
npm install --global [package]	yarn global add [package]	
npm rebuild	yarn install --force	
npm uninstall [package]	(N/A)	
npm uninstall --save [package]	yarn remove [package]	
npm uninstall --save-dev [package]	yarn remove [package]	
npm uninstall --save-optional [package]	yarn remove [package]	
npm cache clean	yarn cache clean	
rm -rf node_modules && npm install	yarn upgrade	-

- 作者：Shellway Ho，转载请注明出处！
- 来源：<http://docs.shellway.cn/learning-yarn/>

各种依赖类型

项目 `package.json` 文件中通常会包含多种依赖，如下：

```
1. {
2.   "name": "my-project",
3.   "dependencies": {
4.     "package-a": "^1.0.0"
5.   },
6.   "devDependencies": {
7.     "package-b": "^1.2.1"
8.   },
9.   "peerDependencies": {
10.    "package-c": "^2.5.4"
11.  },
12.  "optionalDependencies": {
13.    "package-d": "^3.1.0"
14.  }
15. }
```

一、dependencies — 普通依赖

运行项目时需要用到的依赖。如React、ImmutableJS。

二、devDependencies — 开发依赖

开发时使用到的依赖。如Babel（ES6转ES5）、Flow（JS静态类型检查）

三、peerDependencies — 对等依赖，发布依赖包时用

对等依赖—依赖的特殊类型，你在发布依赖包时使用。

使用对等依赖意味着，别人使用你的程序时，安装的依赖需要跟你安装的一毛一样。这对象React这样的包特别有用，它经常需要复制一份react-dom供安装者使用。

四、optionalDependencies — 可选依赖

可选的依赖包，如果此包安装失败，Yarn依然会提示安装进程成功。对非必须的依赖包很实用，如果安装失败，你可以选择其他的包替代。如Watchman。

五、bundledDependencies — 要打包的依赖/捆绑依赖

在发布包时，会一起打包的依赖包。它是由包名组成的数组。捆绑依赖包应该下载到项目本地。功能跟普通依赖是一样的。运行 `$ yarn pack` 时，它们也会被打包。捆绑依赖是为了解决有些包可能在npm找不到，或者在你需要把你自己的项目作为模块使用时。

- 作者：Shellway Ho，转载请注明出处！

- 来源: <http://docs.shellway.cn/learning-yarn/>