Yarn 是 Facebook, Google, Exponent 和 Tilde 开发的一款新的 JavaScript 包管理工具。
Yarn 同样是一个从 npm 注册源获取模块的新的 CLI 客户端。注册的方式不会有任何变化 —— 你同样可以正常获取与发布包。

安装

```
$ npm install -g yarn
```

查看版本

```
$ yarn -version
```

初始化

```
$ yarn init
```

添加依赖

```
$ yarn add <packagename>
# 一次性添加多个包
$ yarn add <packagename1> <packagename2>
```

Yarn在安装过程中，会自动生成一个 yarn.lock 文件，yarn.lock 会记录你安装的所有大大小小的包。
yarn.lock 锁定了安装包的精确版本以及所有依赖项，保留此文件，再次运行 yarn install 时，会根据其中记录的版本号获取所有依赖包。
有了这个文件，你可以确定项目团队的每个成员都安装了精确的软件包版本，部署可以轻松地重现，且没有意外的 bug。
你可以把 yarn.lock 提交到本库里，这样其他签出代码并运行 yarn install 时，可以保证大家安装的依赖都是完全一致的。

全局依赖

官网不建议像npm一样添加全局依赖，因为全局依赖比较隐式，更建议在本地添加所有依赖，这样比较显式。
如您尝试使用具有bin的CLI工具，则可以在./node_modules/.bin目录中访问这些工具。 您还可以使用全局命令：

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

Using --dev or -D will install one or more packages in your devDependencies.

```
$ yarn add <package...> [--dev/-D]
```

从 npm 迁移到 yarn：Yarn 与 npm 的许多功能是相同的，包的元数据格式也是一样的，因此你可以无痛迁移到 yarn。

yarn使用与npm的package.json相同的格式，并且可以从npm注册表安装任何包。

用npm管理的依赖项目可以使用以下命令迁移到yarn

```
$ yarn
```

该命令将重新生成node_modules文件夹，并且yarn会使用兼容算法处理node.js的模块。

运行yarn或者yarn add <package>时，Yarn将在项目根目录生成yarn.lock文件，无这是锁定依赖库的版本文件，无需理解和阅读，将其检入代码管理工具即可。其他人使用Yarn或者npm，yarn.lock能确保他们获得与自己相同的依赖库。

若package.json中的信息不够明确或开发人员经常删除重建node_modules，yarn命令或yarn add不起作用，先使用npm安装明确依赖信息，再转为yarn管理。

从Yarn 1.7.0开始，您可以使用yarn import将npm生成的package-lock.json状态导入Yarn。