# Bower包管理工具的使用

> 本文主要介绍前端开发中常用的包管理工具Bower，具体包括Bower的基本情况、安装、使用和常见命令等内容，最后还介绍了依赖树管理的常见方式以及Bower采用的策略并进行了比较。

## 1.1 关于Bower

Bower是一款优秀的包管理器，它由Twitter公司开发，支持以命令行的方式来对包进行搜索、下载、更新和卸载。

**模块或组件**指独立完整的模块，可以是应用的一部分或者是扩展，依赖可以是jQuery或backbone这样的库，也可以像Bootstrap这样的UI框架或者是UI组件。

**包**英文（package）模块或组件的另一种叫法。

**依赖**一个模块为了满足独立完整原则所必须的其他模块，依赖提供了这个模块所需要的功能，如果没有这个功能，那么这个组件就无法工作。例如我们认为jQuery-ui这个组件依赖于jQuery。

**Bower的优点**

> ❑ 专为前端开发设计，几乎所有的主流前端库都可以使用该工具。
> ❑ 约束松散，使用简单。
> ❑ 依赖树扁平更适合Web应用开发。

官网参考：https://bower.io/

## 1.2 Bower的安装

在安装bower之前，必须确认你已经安装了Node.js和Git。

**安装Bower**

使用npm来安装Bower，-g表示全局安装

```
$ npm install -g bower
```

**查看Bower版本**

Bower安装完成后就能在命令行中直接使用bower命令了，可以通过下面的命令行来查看当前的版本。

```
$ bower -v 或者是 $ bower -version
```

**查看帮助信息**

使用help命令来查看帮助信息。

```
$ bower -help
```

```HTML
Usage:

    bower <command> [<args>] [<options>]
Commands:

    cache            Manage bower cache(管理缓存信息)
    help             Display help information about Bower（显示关于Bower的帮助信息）
    home             Opens a package homepage into your favorite browser
    info             Info of a particular package(显示特定包的详细信息)
    init             Interactively create a bower.json file（交互式创建bower.json文件）
    install          Install a package locally（安装包到本地）
    link             Symlink a package folder(在本地bower库建立一个项目链接)
    list             List local packages - and possible updates（列出本地包以及可能的更新）
    login            Authenticate with GitHub and store credentials（Github身份认证）
    lookup           Look up a single package URL by name（根据包名查询包的URL）
    prune            Removes local extraneous packages（删除项目没有用到的包）
    register         Register a package（注册一个包）
    search           Search for packages by name（通过名称来搜索包）
    update           Update a local package（更新项目的包）
    uninstall        Remove a local package（移除项目的包）
    unregister       Remove a package from the registry（注销包）
    version          Bump a package version（列出版本信息）
Options:...
```

## 1.3 Bower的使用

**初始化操作**

在桌面创建新的文件夹，用来演示Bower的使用。先使用命令行进入到文件夹路径，然后使用下面的命令来对Bower进行初始化操作。

```
$ bower init
```

根据提示来交互式的设置基本项，初始化操作完成之后，会在文件夹的根目录中创建一个bower.json文件，里面包含一些基本信息。

```
{
  "name": "bowerDemo",
  "authors": [
    "wendingding"
  ],
  "description": "Nothing",
  "main": "",
  "license": "MIT",
  "homepage": "wendingding.com",
  "private": true,
  "ignore": [
    "**/.*",
    "node_modules",
    "bower_components",
    "test",
    "tests"
  ]
}
```

**安装指定的包**

尝试执行下面的命令行，来把jQuery框架安装到当前项目中。

```
$ bower install --save jquery
```

命令行中的save参数会把包记录保存到bower.json文件中，install命令会把jQuery框架下载到bower_components目录中，该目录文件是保存所有组件和依赖的地方。当执行install命令的时候，Bower会从仓库中获取到jQuery组件的信息然后和bower.json文件中的信息进行比较，如果jQuery没有安装，那么就会默认安装最新版本，如果已经安装了，则bower会自动停止并提示。

具体的执行情况如下：

```
bogon:Demo wendingding$ bower install --save jquery
bower invalid-meta  for:/Users/文顶顶/Desktop/Demo/bower.json
bower invalid-meta  The "name" is recommended to be lowercase, can contain digits, dots
bower cached        https://github.com/jquery/jquery-dist.git#3.3.1
bower validate      3.3.1 against https://github.com/jquery/jquery-dist.git#*
bower install       jquery#3.3.1

jquery#3.3.1 bower_components/jquery
```

命令执行完毕后，Bower会在根目录下生成bower_components文件夹，并把下载好的包（jQuery框架）放在这个文件夹里面。此时目录结构如下：

```HTML
.
├── bower.json
└── bower_components
    └── jquery
        ├── AUTHORS.txt
        ├── LICENSE.txt
        ├── README.md
        ├── bower.json
        ├── dist
```

```
├── external
└── src
```

查看bower.json文件会发现，此时更新了包和对应的版本信息

```
"dependencies": {
    "jquery": "^3.3.1"
  }
```

> **Bower在执行安装操作的时候，主要做了如下操作：**
>
> ① 检查项目目录中的bower.json文件，以确定包是否已经安装了。
> ② 如果指定的包没有安装，那么Bower就会检查Bower仓库中是否存在名称对应的包
> ③ 如果Bower仓库中存在指定的包，那么就下载最新版本到本地
> ④ 把下载好的包的文件添加到项目目录中，并且更新bower.json文件(包名和版本)

## 1.4 Bower的常见命令

**安装指定包**

```
$ bower install # 通过 bower.json 文件安装
$ bower install jquery # 通过在github上注册的包名安装
$ bower install desandro/masonry # GitHub短链接
$ bower install http://example.com/x.js # URL路径
$ bower install git://github.com/user/package.git #Github上的 .git
```

想要下载安装的包可以是GitHub上的短链接、.git 、一个URL路径等。

**搜索指定的包**

```
$ bower search jquery
```

如果我们在使用框架或者是框架插件的时候，记不住或者是不确定包的名字，则可以尝试先通过关键字搜索，bower会列出包含关键字的所有可用包。

**安装包的指定版本**

```
$ bower install --save jquery#1.8.0
```

Bower通过#号来确定需要下载的版本，如果没有指定版本，则Bower自动帮我们下载最新的。所以，如果需要下载特定版本的包，可以在安装命令中使用#号来声明。

如果要安装指定的版本，也可以先在bower.json文件中对dependencies选项进行配置，然后执行 `$ bower install` 或者是 `$ bower update` 命令。

**查看已安装的包**

```
$ bower list
$ bower list --paths
```

具体的执行情况如下

```
wendingding$ bower list
bower check-new     Checking for new versions of the project dependencies...
demo /Users/文顶顶/Desktop/Test
├── jquery#3.3.1
└─┬ jquery-ui#1.12.1
  └── jquery#3.3.1
wendingding$ bower list --paths
          jquery: bower_components/jquery/dist/jquery.js
          jquery-ui': bower_components/jquery-ui/jquery-ui.js
```

`list命令` 可以查看项目中当前下载过的包，并提供最新版本号。
`paths命令` 可以查看当前下载过的所有包在项目中的对应路径，在其它工具中需要声明/配置前端依赖包地址的时候该命令可能会比较有用。

**卸载指定的包**

```
$ bower uninstall jquery
```

卸载本地项目中已经安装的jQuery框架。

**更新指定的包**

```
$ bower update jquery
```

更新包的过程和安装包的过程差不多，区别在于更新的时候会使用新文件替换旧文件，上面的命令强制安装最新版本的jQuery框架。

**查看指定包的详细信息**

```
$ bower info jquery
```

通过info指令可以查看指定包的详情，还指令会列出对应的git仓库地址，以及所有可用的版本。

## 1.5 Bower安装有依赖的包

通常当我们使用Bower命令( `bower install --save xxx` )来安装指定包的时候，Bower会查找包对应的git仓库地址然后下载到本地并在bower.json文件中记录版本等信息。但有的包在使用的时候可能存在依赖关系，比如ember这个包需要依赖于jQuery框架。

> **使用Bower安装有依赖包的两种方式**
> ❑ 先安装该包的依赖项（这里为jQuery），再安装指定的包（这里为ember）
> ❑ 直接安装（这里为ember）

**方式①**

先安装依赖,再安装指定包

```
$ bower install --save jquery
$ bower install --save ember
$ bower list
```

```HTML
Test /Users/文顶顶/Desktop/Test
├─┬ ember#2.18.2 (latest is 3.0.0-beta.2)
| └── jquery#3.3.1
└── jquery#3.3.1
```

**方式②**

直接安装指定包,会自动安装依赖(推荐)

```
$ bower install --save ember
```

```HTML
bower invalid-meta  for:/Users/文顶顶i/Desktop/Test/bower.json
bower cached        https://github.com/components/ember.git#2.18.2
bower validate      2.18.2 against https://github.com/components/ember.git#*
bower cached        https://github.com/jquery/jquery-dist.git#3.3.1
bower validate      3.3.1 against https://github.com/jquery/jquery-dist.git#>=1.7.0<4.0.0
bower install       ember#2.18.2
bower install       jquery#3.3.1
ember#2.18.2 bower_components/ember
└── jquery#3.3.1
jquery#3.3.1 bower_components/jquery`
```

列出项目中已经安装的所有包和依赖关系

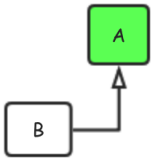
```
$ bower list
```

```HTML
......
Test /Users/文顶顶/Desktop/Test
└─┬ ember#2.18.2 (latest is 3.0.0-beta.2)
  └── jquery#3.3.1
```

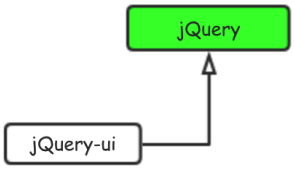
该命令行在安装ember包的时候，发现需要依赖于jQuery框架，就会自动下载对应版本的jQuery框架并安装到本地项目中。

## 1.6 依赖树管理

组件(包)并非总是相互独立的，有些组件(包)在使用的时候需要依赖于另外一些组件(包)，就像上文提到的ember需要依赖于jQuery，jQuery-ui需要依赖于jQuery一样，我们尝试使用下面的图示来描述这种关系。




该图示用于表明两个组件间的关系。图中的箭头表示依赖关系，描述"包(组件)B依赖于包（组件）A"。




jQuery-ui组件依赖于jQuery 组件，jQuery组件是jQuery-ui组件的依赖。

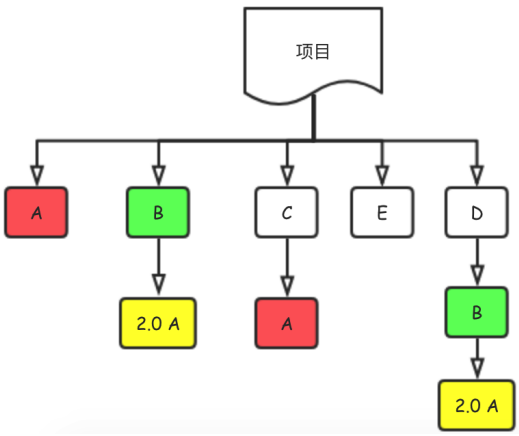
上面的图示揭示了包（组件）与包（组件）之间的依赖关系，非常简单容易理解。在实际中，每个组件可能都有多个依赖，而这些依赖自己可能还有别的依赖，或者多个不同组件都依赖于某个指定的组件，因此在处理的时候可能会非常复杂。为了理清楚这复杂的关系，依赖管理工具会把所有的依赖构成一颗 Dependency Tree （依赖树）。

> **依赖树主要有三种**
> ❑ 嵌套依赖树
> ❑ 扁平依赖树
> ❑ 混合依赖树

注意：在构造依赖树的时候，组件（包）必须要有唯一的标识，该标识由组件的名称和版本号构成，也就是说jQuery1.7.3和jQuery3.3.1是两个不同的组件。

### 嵌套依赖树

基本理念 ： 每个组件各自都有自己的依赖，而不会共用一个依赖。
主要问题 ： 项目中会产生同一组件的多个副本，且可能有多个版本的组件共存比较混乱。

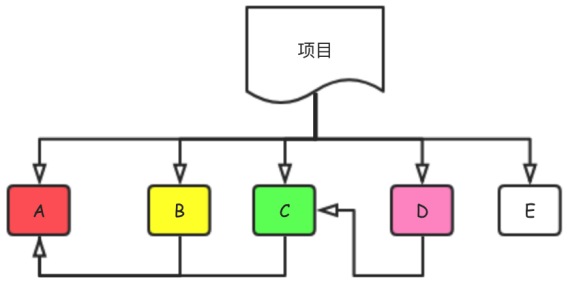



嵌套依赖树说明

☆ 组件E是独立的组件。
☆ 组件B需要的是组件A的2.0版本。
☆ 组件B、C依赖于组件A，它们各自都有一个A的副本。
☆ 组件D依赖于组件B，间接依赖组件A的2.0版本，B和2.0A则又被下载一次。

### 扁平依赖树

基本理念 ： 保证每个组件在项目只有一个版本，没有任何其它的副本。
主要问题 ： 容易产生冲突，如果两个组件需要同一个依赖的不同版本，就会导致出错，必须要用户自己来解决冲突，决定具体使用哪个依赖并解决潜在的不一致性。



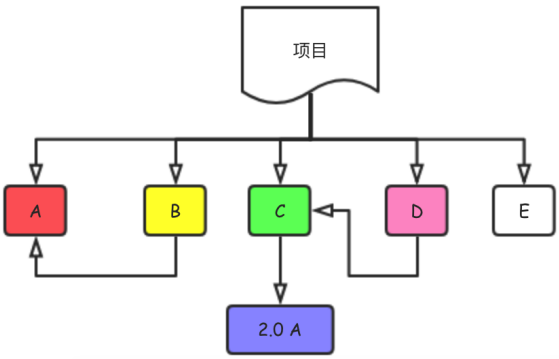

扁平依赖树说明

☆ 组件E是独立的组件。
☆ 组件B、C依赖于组件A。
☆ 组件D依赖于组件C，间接依赖组件A。
☆ 扁平依赖树的方式决定组件B、C、D必须使用同一个版本的组件A。

**混合依赖树**

基本理念 ： 使用最高效的办法来管理一个组件的不同版本。

主要特点 ： 混合依赖树是扁平依赖树和嵌套依赖树的折中方案，如果一个组件的依赖已经安装了，而且版本也兼容，那么就不必再次下载安装，只要指向已安装的那个组件即可，如果版本不兼容的话，则下载安装并版本兼容的组件。




混合依赖树说明

☆ 组件A和E是独立的组件。
☆ 组件B依赖于组件A的指定版本，组件B需要的依赖版本和A兼容则不再下载。
☆ 组件C依赖于组件A的2.0版本，跟已经安装的组件A不兼容所以安装2.0A组件。
☆ 组件D依赖于组件C，间接依赖于组件2.0 A。

## 1.7 Bower依赖树管理和冲突处理

Bower作为专为前端开发者设计的依赖管理工具，是完全基于扁平依赖树的。上文介绍了扁平依赖树在处理的时候要求**保证每个组件在项目只有一个版本，没有任何其它的副本**，优缺参半。那既然如此，Bower为什么不使用更高效的混合依赖树?

> **Bower采用扁平依赖树管理的原因**
>
> (1) 代码体积对于前端开发非常重要，而扁平树管理可以实现组件最(少)小化。
> (2) 所有组件默认都在浏览器的全局作用域中运行，不同版本的同名组件会产生冲突。

因为Bower采用了扁平依赖树的方式来处理，所以在使用的时候容易产生冲突，这种依赖管理方式要求开发者注重组件的版本兼容和依赖关系。接下来，我们简单演示Bower使用过程中会出现冲突的情况。

**冲突的产生和处理**

① 安装jQuery框架的1.2.3版本

```
$ bower install --save jquery#1.2.3
```

执行情况

```HTML
bower invalid-meta  for:/Users/文顶顶/Desktop/Test/bower.json
bower invalid-meta  The "name" is recommended to be lowercase, can contain...
bower not-cached    https://github.com/jquery/jquery-dist.git#1.2.3
bower resolve       https://github.com/jquery/jquery-dist.git#1.2.3
bower download      https://github.com/jquery/jquery-dist/archive/1.2.3.tar.gz
bower extract       jquery#1.2.3 archive.tar.gz
bower deprecated    Package jquery is using the deprecated component.json
bower resolved      https://github.com/jquery/jquery-dist.git#1.2.3
bower install       jquery#1.2.3
```

② 安装ember组件并解决冲突

```
$ bower install --save ember
```

具体的执行情况

```HTML
bogon:Test wendingding$ bower install --save ember
bower invalid-meta  for:/Users/文顶顶/Desktop/Test/bower.json
bower invalid-meta  The "name" is recommended to be lowercase, can contain digits, dots, dashes
bower cached        https://github.com/components/ember.git#2.18.2
bower validate      2.18.2 against https://github.com/components/ember.git#*
bower cached        https://github.com/jquery/jquery-dist.git#3.3.1
bower validate      3.3.1 against https://github.com/jquery/jquery-dist.git#>= 1.7.0 < 4.0.0

Unable to find a suitable version for jquery, please choose one by typing one of the numbers be
    1) jquery#1.2.3 which resolved to 1.2.3 and is required by Test
    2) jquery#>= 1.7.0 < 4.0.0 which resolved to 3.3.1 and is required by ember#2.18.2

Prefix the choice with ! to persist it to bower.json

? Answer 2
bower install       jquery#3.3.1
bower install       ember#2.18.2

jquery#3.3.1 bower_components/jquery

ember#2.18.2 bower_components/ember
└── jquery#3.3.1
```

说明 ： 我们先把jQuery的1.2.3版本安装到了项目中，然后又通过Bower来安装ember，而ember组件需要依赖于jQuery框架，这里有个关键信息就是 ember要求依赖的jQuery框架版本范围为1.70 ~ 4.0.0 和本地已经安装的jQuery 1.2.3冲突，Bower并不会自己处理这个问题而是抛出一个异常，把选择权交给用户，由用户来选择使用哪种方案。

通过命令行的打印，我们看到Bower为我们提供了两个可选项，第一个选项是保留本地已经安装的1.2.3版本，第二个选项是保存ember所依赖的版本，这里显示了依赖需要的版本范围jQuery#>=1.70<4.0.0和最终会决定(resolved)的版本(3.3.1)。上面的示例中，冲突产生后我们输入2，选择安装jQuery的3.3.1版本。

③ 查看项目已安装的组件信息

```
$ bower list
```

具体的组件和依赖结构

```
Test /Users/文顶顶/Desktop/Test

├─┬ ember#2.18.2 (latest is 3.0.0-beta.2)
| └── jquery#3.3.1
└── jquery#3.3.1 incompatible with 1.2.3 (1.2.3 available, latest is 3.3.1)
```

冲突处理完后，项目中原本下载安装好的jQuery1.2.3版本被重新下载的3.3.1版本替代。

## 1.8 Bower自定义组件目录

默认情况下，所有的依赖包都被下载保存到bower_components文件路径，如果想要把依赖包下载到自己指定的目录，使用.bowerrc文件配合bower.json就可以实现。

在项目根目录创建.bowerrc文件，使用json格式来设置文件路径。

```
{ "directory": "指定路径" }
```

保存好以后，执行bower install命令，就会把bower.json配置好的相关组件全部下载到指定的路径中。

命令行参考

HTML

```
bogon:Test wendingding$ touch .bowerrc
bogon:Test wendingding$ vim .bowerrc
bogon:Test wendingding$ cat .bowerrc
{
    "directory": "app/xxx/"
}
bogon:Test wendingding$ cat bower.json
{
  "name": "Test",
  "authors": [
    "flowerField <18681537032@163.com>"
  ],
  "description": "",
  "main": "",
  "license": "MIT",
  "homepage": "",
  "ignore": [
    "**/.*",
    "node_modules",
    "bower_components",
    "test",
    "tests"
  ],
  "dependencies": {
    "jquery": "^3.3.1"
  }
}
bogon:Test wendingding$ bower install
bower invalid-meta  for:/Users/文顶顶/Desktop/Test/bower.json
bower invalid-meta  The "name" is recommended to be lowercase, can contain digits...
bower cached        https://github.com/jquery/jquery-dist.git#3.3.1
bower validate      3.3.1 against https://github.com/jquery/jquery-dist.git#^3.3.1
```

```
bower install       jquery#3.3.1

jquery#3.3.1 app/xxx/jquery
bogon:Test wendingding$ tree -L 3
.
├── app
|   └── xxx
|       └── jquery
└── bower.json
```

- Posted by 博客园·文顶顶 | 花田半亩
- 联系作者 简书·文顶顶 新浪微博·Coder_文顶顶
- 