**Bower**

**Índice**

1. Introdução
2. O que é?
3. Bower VS NPM
4. Instalação
5. Inicialização
6. Gerenciamento de pacotes 7.1. Buscando 7.2. Se informando 7.3. Instalando 7.4. Atualizando 7.5. Removendo 7.6. Pacotes instalados 7.7. Cache 7.9. Sufixos de versões
7. Dicas
8. Conclusão

**1. Introdução**

Depois de um artigo abordando uma ["Introdução ao NPM"](https://github.com/eudiasrafael/artigo__introducao_ao_npm), irei dar agora sequência a série de artigos visando ajudar alguns desenvolvedores que podem ter ficado perdidos no tempo no mercado de front-end.

Irei abordar neste segunda artigo uma introdução ao Bower, pacote baseado no NodeJS e MUITO similar ao NPM.

**2. O que é?**

Assim como o NPM, o Bower é um gerenciador de pacotes, oque significa que possui online um repositório de projetos de código aberto os quais podem ser baixados com um único comando da linha de comando para dentro do nosso projeto.

Ainda como o NPM, possui um controle de versões dos pacotes de tal maneira que ele alerta quando um pacote não é compatível com outro, ou seja se estamos tentando utilizar um plugin do jQuqey que não seja compatível com a versão do jQuery instalada, ele irá nos alertar e perguntar oque desejamos fazer (Atualizar jQuery?, Manter versão menos recente do plugin?, Não instalar o plugin?). Isso vale tanto para instalação como atualização.

Mas então... por quê utilizar ele ao invés do NPM?

**3. Bower VS NPM**

Bower e NPM são de fato muito similares, tanto que aqueles que leram meu artigo anterior ["Introdução ao NPM"](https://github.com/eudiasrafael/artigo__introducao_ao_npm) notarão que este é exatamente a mesma coisa, com algumas pequenas alterações.

Embora muito similares, existe uma diferença básica entre os dois pacotes que se resume em para oquê eles foram desenvolvidos.

O NPM foi desenvolvido pensando em ser um repositório de pacotes para o NodeJS e apenas isso, porém passou a ser utilizado também para pacotes frontend.

Já o Bower foi criado com foco nos pacotes frontend e com isso, otimizado para este trabalho.

O recomendado é utilizar o NPM somente para pacotes do NodeJS e o Bower para pacotes Frontend (embora não seja obrigatório).

Algumas vantagens do Bower sobre o NPM para o gerenciamento de pacotes Frontend:

1. **Busca de pacotes não quebra** - O NPM estoura a memória em algumas máquinas quando tenta fazer a busca pela primeira vez. Isso ocorre justamente pelo NPM ter sido pensado para pacotes do NodeJS apenas e não deveria haver tantos pacotes assim. Já o Bower dá conta do recado uma vez que foi desenvolvido pensando nisso.
2. **Separação de responsabilidades** - São coisas totalmente diferentes os pacotes do Bower - Focados em adicionar recursos à sua aplicação frontend - e os pacotes do NPM - Focados em prover recursos para uma máquina, tais como programas que podem executar uma infinidade de tarefas. Sendo assim faz todo o sentido controlar estes diferentes pacotes em locais diferentes.

No final das contas trata-se de uma questão de escolha, existem muitos desenvolvedores que abandonaram o Bower e utilizam somente o NPM.

Eu particularmente, prefiro separar!

**4. Instalação**

Uma vez que o Bower é baseado no NodeJS, ele dev ser instalado via NPM.

Caso não tenha o NPM e/ou não saiba utilizá-lo adequadamente, recomendo (novamente) a leitura do meu artigo ["Introdução ao NPM"](https://github.com/eudiasrafael/artigo__introducao_ao_npm).

Uma vez tento o NPM corretamente instalado e configurado no projeto, abra o diretório do projeto e execute o seguinte comando:

npm install -g bower

Isso irá instalar o Bower globalmente, permitindo a executação dele via linha de comando.

**5. Inicialização**

<https://bower.io/docs/api/#init>

Uma vez que temos o Bower instalado na nossa máquina, vamos precisar inicializar ele no diretório do projeto onde desejamos utilizá-lo.

Para isso vamos acessar o diretório pela linha de comando e dentro dele executar o seguinte comando:

bower init

Será feita uma série de perguntas, não precisa se assustar com elas uma vez que não irão afetar o funcionamento do Bower no seu projeto.

Estas perguntas dizem respeito tão somente aos detalhes do seu projeto caso você venha a disponibilizá-lo no Bower, tais como nome, descrição, versão, repositório GIT e etc.

**6. Gerenciamento de pacotes**

**6.1. Busca**

Ocasionalmente acontecerá de não sabermos o nome do pacote que desejamos instalar, por exemplo o AngularJS, qual o nome? Será angular, angularjs, angularjs1?

Para resolver esta questão nós podemos efetuar uma busca, e existem duas maneiras para isso. Uma através da linha de comando e outra através de um navegador web.

**6.1.1 Linha de comando**

<https://bower.io/docs/api/#search>

Acesse o diretório do projeto pela linha de comando e dentro dele execute o seguinte comando:

bower search angular

O Bower irá buscar e exibir os resultados que baterem com "angular".

**6.1.2. Navegador WEB**

Para buscar um pacote do Bower pelo navegador, basta acessar o [site do Bower](https://bower.io/search/) e entrar com o termo de pesquisa desejado na caixa de busca (acessar ítem do menu "Search Packages"). Uma lista de pacotes associados será exibida.

**5.2. Se informando**

<https://bower.io/docs/api/#info>

Uma vez que temos um nome de pacote, pode ocorrer de querermos ter maiores informações a respeito dele, para isso também tempos um comando.

bower info angular

Serão exibidas todas as informações a respeito do pacote do angular.

**6.3. Instalando**

<https://bower.io/docs/api/#install>

Existem várias maneiras - e suas combinações - de instalar um pacote, veremos cada uma:

**6.3.1. Instalação simples**

Desejamos instalar o angular, simples assim.

Para isso acesse o diretório do projeto pela linha de comando e dentro dele execute o seguinte comando:

bower install angular

Desta forma o angular estará adicionado dentro do diretório "bower\_components" na raiz do projeto.

**6.3.2. Instalação de uma dada versão**

Desejamos instalar uma versão específica do o angular (1.6.0 por exemplo).

Para isso acesse o diretório do projeto pela linha de comando e dentro dele execute o seguinte comando:

bower install angular@1.6.0

Desta forma o angular v1.6.0 estará adicionado dentro do diretório "bower\_components" na raiz do projeto.

**6.3.3. Salvando a dependência**

Se estamos adicionando um pacote ao projeto, possivelmente significa que ele é uma dependência para que o mesmo funcione corretamente.

Para que seja possível futuramente instalar todas as dependências com um único comando - sem ter que instalar uma a uma -, é necessário salvar as dependências.

Este processo também é necessário para que o Bower faça atualização dos pacotes e mantenha um controle sobre a compatibilidade das versões dos pacotes.

Para salvar uma dependência (durante ou depois da instalação) acesse o diretório do projeto pela linha de comando e dentro dele execute o seguinte comando:

bower install --save angular

Pronto, agora o arquivo bower.json do Bower no seu projeto possuirá uma posição para dependências com o nome do pacote (angular) e a versão do mesmo.

**6.3.4. Salvando a dependência de desenvolvimento**

Um pacote também pode ser uma dependência de desenvolvimento, ou seja, ele somente é necessário para o ambiente de desenvolvimento, nunca no ambiente de produção.

Para salvar uma dependência de desenvolvimento (durante ou depois da instalação) acesse o diretório do projeto pela linha de comando e dentro dele execute o seguinte comando:

bower install --save-dev angular

Pronto, agora o arquivo bower.json do Bower no seu projeto possuirá uma posição para dependências de desenvolvimento com o nome do pacote (angular) e a versão do mesmo.

**6.3.5. Instalando todas as dependências**

Digamos que acabamos de clonar um projeto e ele esta sem as suas dependências, porém com todas elas informadas no bower.json. Como podemos fazer para instalar todas elas de uma vez só?

Acesse o diretório do projeto pela linha de comando e dentro dele execute o seguinte comando:

bower install

Tão simples que chega a doer né?

**6.3.7. Instalando todas as dependências de produção**

Ao invés de instalar todas as dependências, queremos instalar somente aquelas que não sejam de dependência.

Para isso acesse o diretório do projeto pela linha de comando e dentro dele execute o seguinte comando:

bower install --production

**6.4. Atualizando**

<https://bower.io/docs/api/#update>

Também podemos atualizar nossos pacotes que estejam desatualizados, veja:

**Atualizar todos os pacotes do projeto**

bower update

**Atualizar determinado pacote do projeto**

bower update angular

Lembrando que a atualização só irá rodar com pacotes que estiverem presentes na lista de dependências do bower.json.

**6.5. Removendo**

<https://bower.io/docs/api/#uninstall>

Também podemos remover um determinado pacote que não desemos mais, veja:

**Remoção simples**

bower uninstall angular

**Remoção salvando nas dependências**

bower uninstall --save angular

**6.6. Pacotes instalados**

<https://bower.io/docs/api/#list>

Para ver quais os pacotes estão instalados no projeto, acesse o diretório do projeto pela linha de comando e dentro dele execute o seguinte comando:

bower list

O Bower poderá informar de pacotes instalados que não se encontram no bower.json.

**6.7. Cache**

<https://bower.io/docs/api/#cache>

O Bower possui um sistema de cache que armazena os pacotes baixados de forma que, ao tentarmos instalar um deles em um novo projeto, ele não realiza um novo download, somente copia do cache.

Existem alguns comandos do cache que podem vir a ser úteis:

**Listar pacotes cacheados**

bower cache list

**Limpar cache**

bower cache clean

**6.9. Sufixos de versões**

Assim como no NPM, a grande vantagem do Bower é o controle de compatibilidade dentre os pacotes, isso se deve ao fato de, através dos sufixos no bower.json, podermos especificar quais as versões podemos utilizar (fato também utilizado pelos desenvolvedores dos pacotes).

Isso nos permite controlar melhor a estabilidade dos pacotes do projeto. Irei aqui mostrar como isso funciona.

Primeiro vamos entender como funciona uma versão de acordo com o [SEMVER](http://semver.org/). Suponhamos uma versão 1.2.3, eis oque significa cada posição:

* Primeira posição - É a posição que diz respeito a mudanças no pacote que alterar a compatibilidade do mesmo, alguma função removido ou algum parâmetro alterado... Se o seu projeto não for alterado para se adaptar a esta mudança, ele irá quebrar.
* Segunda posição - É a posição que diz respeito a novas funcionalidades ou recursos no pacote, porém que não tenham alterado a compatibilidade do mesmo. Nenhuma alteração em seu projeto será necessária para continuar funcionando.
* Terceira posição - É a posição que diz respeito a correções no pacote, porém que não tenham alterado a compatibilidade do mesmo. Nenhuma alteração em seu projeto será necessária para continuar funcionando.

Sabendo disso, vamos compreender os sufixos:

* \* - Se encontrar um asteristico no lugar versão do pacote, significa que o Bower poderá instalar/atualizar sempre para a versão mais recente do pacote.
* 1.2.3 - Se encontrar uma versão sem qualquer prefixo no pacote, sígnica que o Bower irá sempre instalar aquela versão, nunca podendo atualizar.
* ^1.2.3 - Se encontrar uma versão sufixada com um circunflexo, significa que o Bower poderá atualizar somente para versões que alteram a terceira posição da versão (ou seja, atualizações de correção que mantenham a compatibilidade)
* ~1.2.3 - Se encontrar uma versão sufixada com um til, significa que o Bower poderá atualizar somente para versões que alteram a segunda e a terceira posição da versão (ou seja, atualizações novos recursos e de correção, porém mantendo a compatibilidade)
* >1.2.3 - Se encontrar uma versão sufixada com um um simbolo de maior, significa que o Bower poderá instalar e atualizar somente versões maiores que a versão dada.
* >=1.2.3 - Se encontrar uma versão sufixada com um um simbolo de maior ou igual, significa que o Bower poderá instalar e atualizar somente versões maiores ou igual que a versão dada.
* <1.2.3 - Se encontrar uma versão sufixada com um um simbolo de menor, significa que o Bower poderá instalar e atualizar somente versões menores que a versão dada.
* <=1.2.3 - Se encontrar uma versão sufixada com um um simbolo de menor ou igual, significa que o Bower poderá instalar e atualizar somente versões menores ou igual que a versão dada.

Dependendo de como forem adicionados os sufixos, o Bower poderá informar alguma incompatibilidade. Por exemplo... Se utilizamos uma versão fixa e antiga do jQuery e um plugin do jQuery com asteristico (ou seja, o mais recente), esta versão não será compatível com o jQuery. Logo, o Bower irá informar e perguntar oque fazer.

**7. Dicas**

Existem algumas poucas dicas que gostaria de acrescentar.

Primeiro, a maior parte dos comandos (ver links da documentação) aceitam o parâmetro --json para exibir o resultado no formato JSON.

Segundo, a linha de comando permite enviar a saída do comando para um arquivo, combinando este fato com a dica acima nós podemos fazer por exemplo:

bower info --json angular > detalhes.json

E teremos todos os detalhes do angular disponíveis no arquivo detalhes.json.