YEOMAN


YO
GRUNT
BOWER

Create a new webapp

yo webapp
OR
yo angular
yo angular:controller

Handle dependencies

bower search
bower install

Preview, test, build

grunt server
grunt test
grunt

GETTING STARTED

The Yo tool install also bower and grunt with it

```
$ npm install -g yo
```

Install the desired generator

```
$ mkdir yo-express
$ cd yo-express
# note that this generator perfers to be installed locally
$ npm install generator-express
```

Generate your express app

```
$ yo express
[?] Select a version to install: Basic
```

A **generator** is a way to share ideas and best practices, compacting them inside a 'recipe'.

TEST YOUR APPLICATION BY RUNNING

```
$ grunt
$ curl localhost:3000
<!DOCTYPE html><html><head><title>Express</title><link rel="stylesheet"
```

Test the live reload by changing one of the templates

index.jade

```
/// Add a new paragraph
p This is your shiny new Express app
```

INTERNALS OF A GENERATED EXPRESS APP

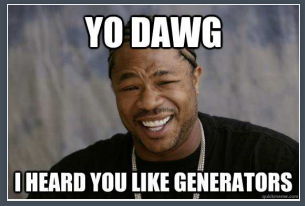
```
.
├── Gruntfile.js
├── app.js
├── bower.json
├── package.json
├── public
│   ├── components
│   ├── css
│   │   └── style.css
│   ├── img
│   └── js
├── routes
│   ├── index.js
│   └── user.js
└── views
    ├── index.jade
    └── layout.jade
```

GENERATORS

Other available generators

- angular - AngularJS
- mobile - Mobile first web apps
- generator-generator - create your own generator
- see more at: http://yeoman.io/community-generators.html

WRITE YOUR OWN GENERATOR

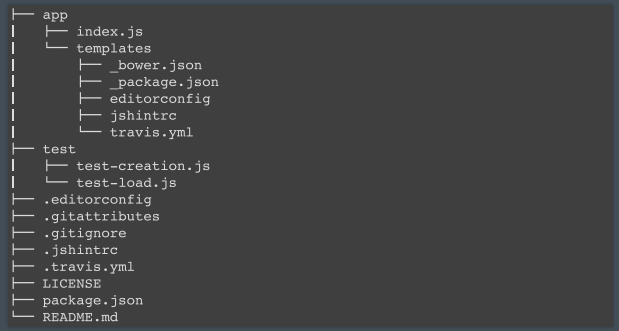


Let's create a generator with a generator

SETUP

```
$ npm install -g yo generator-generator
$ mkdir generator-blog && cd $_
$ yo generator
```

GENERATOR INTERNALS



HOW TO TEST YOUR GENERATOR WHILE IN DEV

Normal users will interect with it by typing:

```
$ yo blog
```

In order for you to have the same experience, you can link it:

```
$ npm link
```

TEST YOUR GENERATED GERENRATOR!

Move to another folder and yo'it

```
$ cd ..
$ mkdir generator-blog-playgroud && cd $_
$ yo blog
```

Yeoman will ask the default questions, don't worry about them for now

EXPLORE YOUR GENERATOR-BLOG

PROMPTS PART I

Go back to your generator-blog folder and open app/index.js

```
$ cd ../generator-blog
$ vim app/index.js
```

index.js

```
  /// Look at the prompts array
  var prompts = [{
    type: 'confirm',
    name: 'someOption',
    message: 'Would you like to enable this option?',
    default: true
  }];
```

Let's change it to ask for the blog name

```
var prompts = [{
  name: 'blogName',
  message: 'What do you want to call your blog?'
}];
```

EXPLORE YOUR GENERATOR-BLOG

PROMPTS PART II

Let's change this.prompt to make more sense for our application **index.js**

```
  /// this.prompt ate up our prompts array for it's first argument,
  // then a callback that will execute after all of the responses
  // have come in
  this.prompt(prompts, function (props) {
    this.someOption = props.someOption;

    cb();
  }.bind(this));
```

to this:

```
this.prompt(prompts, function (props) {
  // `props` is an object passed in containing the response values, name
  this.blogName = props.blogName;
  cb();
}.bind(this));
```

LANDING OFF THE GROUND

index.js

```
  /// This section is where you tell Yeoman what to create when
// generator-blog is called
BlogGenerator.prototype.app = function app() {
  this.mkdir('app');
  this.mkdir('app/templates');

  this.copy('_package.json', 'package.json');
  this.copy('_bower.json', 'bower.json');
};

BlogGenerator.prototype.projectfiles = function projectfiles() {
  this.copy('editorconfig', '.editorconfig');
  this.copy('jshintrc', '.jshintrc');
};
```

LANDING OFF THE GROUND

Now we are going to explain yeoman what should be generated when called yo blog

```
BlogGenerator.prototype.app = function app() {
  this.mkdir('posts');

  this.copy('_package.json', 'package.json');
  this.copy('_bower.json', 'bower.json');
};
```

```
BlogGenerator.prototype.projectfiles = function projectfiles() {
  this.copy('editorconfig', '.editorconfig');
  this.copy('jshintrc', '.jshintrc');
};
```

TEST OUR IMPROVEMENTS

```
$ cd ../generator-blog-playground
$ rm -R *
$ yo blog
```

SUMMARY

- Yeoman provides you with a workflow and a collection of tools to help your development
- Generators are used to share best practices between teams