WORKFLOW AUTOMATION FOR WEB APPLICATIONS

Created by Mayank Patel / @maxy_ermayank

WHY WORKFLOW AUTOMATION?

- Boilerplate
- Dependency management
- Framework
- Abstrations
- Build
- Automation test
- Docs
- Continuous integration
- Deployment
- Performance optimization
- Workflow
- Deployment

WORK FLOW

SETUP

- Download dependecies
- Download frameworks
- Download libraries
- Scaffolding

DEVELOP

- Non minified
- Linting (HTML, JS)
- Seperated files
- Generate responsive images
- Optimize images
- Compilation (CoffeScript, SASS, LESS..)
- Test configuration
- Unit testing & e2e testing
- Generate test report
- Watchers
- Live reload

BUILD

- Annotate (JS)
- Generate copyright and license information
- Sourcemap (JS, CSS)
- Concatenation (JS, CSS)
- Minification (HTML, JS, CSS)
- Uglification (HTML, JS, CSS)
- Compress (JS, CSS)
- Live configuration
- Compiled
- Renamed
- Cache templates (HTML)
- Inject resources in Template
- Optimize performance
- Deployment setup

DEPENDENCY MANAGEMENT TOOLS

Downloads dependencies using Git, HTTPS, ZIP, npm

- npm
- Bower

NPM

- Package manager for the web
- Comes with node.js
- Default for node.js modules

INSTALL NODE.JS

Filename: package.json

Download node.js and follow installation guide

NPM EXAMPLES

Find available package Here.

```
> npm install <package>

-g installs package globally
> npm install -g <package>

use --save to save dependecy in package.json & -dev to lock package
> npm install <package> --save-dev
```

```
> npm init
```

SAMPLE PACKAGE.JSON

```
{
    "name": "project-name",
    "version": "1.0.0",
    "description": "Description goes here",
    "main": "index.html",
    "scripts": {
        "test": "gulp e2e"
    },
    "repository": {
        "type": "git",
        "url": "https://example.com/project-name"
    },
    "author": "Mayank Patel <maxy.ermayank@gmail.com>",
    "license": "MIT",
    "bugs": {
        "url": "https://example.com/project-name/issues"
```

BOWER

- Package manager for the web
- Designed solely for web and it is optimized with that in mind.

INSTALL

Filename: bower.json

```
Install Globally
> npm install -g bower

Install in your project
> npm install bower
```

EXAMPLES

Packages available Here

```
bower install <package>

bower install git://github.com/user/package.git

bower install http://example.com/script.js
```

SAMPLE BOWER.JSON

```
{
  "name": "project-name",
  "version": "1.0.0",
  "author": "Mayank Patel <maxy.ermayank@gmail.com>",
  "homepage": "https://example.com/project-name",
  "description": "Description goes here",
  "main": "index.html",
  "license": "apache 2",
  "dependencies": {
    "angular": "1.3.15",
    "json3": "~3.2.4",
    "es5-shim": "~2.0.8",
    "angular-resource": "1.3.15",
    "d3": "3.3.x"
  },
  "devDependencies": {
```

SCAFFOLDING TOOL / GENERATOR

YEMON (YO)

- Scaffolds out boilerplate
- Abstraction
- Performance optimization
- Testing and Build process
- Custom generators are available

```
Install YO globally
> npm install -g yo
```

EXAMPLES

```
> yo
[?] What would you like to do?
› Install a generator
Run the Angular generator (0.4.0)
Run the Backbone generator (0.1.9)
Run the Blog generator (0.0.0)
Run the jQuery generator (0.1.4)
Run the Gruntfile generator (0.0.6)
(Move up and down to reveal more choices)
```

```
yo jquery-boilerplate
Boom. You just created a jQuery plugin.
```

CUSTOM GENERATOR

Find available generators Here.

```
> npm install generator-bootstrap -g
> yo bootstrap

> npm install generator-webapp -g
> yo webapp
```

GENERATE ENTERPRISE APP USING ANGULAR

```
> npm install generator-angular -g
> yo angular
> yo angular:view user
> yo angular:controller user
> yo angular:directive mydirective
```

GULP STRENGTH

- Mature: ~ Aug 2013, relatively mature
- Community Support: New kid in town, Picking up popularity
- Code over configuration
- Easy to read & use
- Tons of plugins available
- Provides node streams - no need for tmp files/folders
- Plugins do ONE thing
- Provides plugin to run Grunt tasks
- Only has 5 functions to learn!
- Runs with maximum concurrency by default

GULP WEEKNESSES

- Can be daunting to learn streams
- Sometimes setting up src/dest can be tricky (use base)

INSTALL GULP

Filename: gulpfile.js

```
Install Gulp in your project
> npm install gulp --save
```

```
Install Gulp Globaly
> npm install -g gulp
```

THE GULP API

- gulp.task(name[, deps], fn)
- gulp.src(globs)
- gulp.dest(path)
- gulp.watch(glob[, opts], tasks)
- gulp.run(tasks...)

GULP.TASK(NAME[, DEPS], FN)

```
gulp.task('somename', function() {
// Do stuff
});
```

```
gulp.task('build', ['somename', 'test'];
```

```
> gulp build
```

GULP.SRC(GLOBS)

```
gulp.src('client/templates/*.jade')
      .pipe(jade())
```

```
gulp.src(['src/**/*.js', 'test/spec/**/*.js'])
    .pipe(jshint())
```

GULP.DEST(PATH)

```
gulp.src('./client/templates/*.jade')
      .pipe(jade())
      .pipe(gulp.dest('./build/templates'))
      .pipe(minify())
      .pipe(gulp.dest('./build/minified_templates'));
```

GULP.WATCH(GLOB[, OPTS], TASKS)

```
gulp.watch('app/**/*.js', ['test','reload']);
```

GULP.RUN(TASKS...)

```
gulp.task('hello-world', function () {
  run('echo Hello World').exec()  // prints "[echo] Hello World\n".
    .pipe(gulp.dest('build'))    // Writes "Hello World\n" to output/
})
```

STREAMS

```
gulp.task('scripts', function () {
      return gulp.src('src/app/**/*.js') // <-- read from filesystem
        // In memory transform
        .pipe(jshint('.jshintrc'))    // <-- lint the code
        .pipe(concat('app.min.js'))   // <-- concatenate to one file
        .pipe(uglify())               // <-- minify the file
        .pipe(rev())                  // <-- add revision to filename
        .pipe(gulp.dest('dist/app')); // <-- write to filesystem
    });
```

SAMPLE GULPFILE.JS

```
'use strict';

var gulp = require('gulp'),
gutil = require('gulp-util'),
del = require('del'),
jshint = require('gulp-jshint'),
ngAnnotate = require('gulp-ng-annotate'),
concat = require('gulp-concat'),
sourcemaps = require('gulp-sourcemaps'),
uglify = require('gulp-uglify'),
concatCss = require('gulp-concat-css'),
minifyCSS = require('gulp-minify-css'),
imagemin = require('gulp-imagemin'),
minifyHtml = require('gulp-minify-html'),
templateCache = require('gulp-angular-templatecache'),
inject = require('gulp-inject'),
```

GRUNT STRENGTH

- Mature: Mar 2012, very mature
- Community Support: Most Popular
- Configuration over code
- Based on files
- Tons of plugins available
- Flexibility
- Scaffolding is available through generators
- Provides plugin to run Gulp tasks

GRUNT WEEKNESSES

- Plugins do multiple things
- Headache of temp files/folders
- Not one solid control flow
- Configuration can get lengthy - 500+ lines / Hard to read
- Very lengthy & vast API
- Can get pretty slow when tasks increase

INSTALL GRUNT

Filename: gruntfile.js

```
Install Grunt in your project
> npm install grunt --save
```

```
Install Grunt-cli / Global install of Grunt command line
> npm install -g grunt-cli
```

STRUCTURE

```
module.exports = function(grunt) {

  grunt.initConfig({
    // Configuration des tâches
  });

  // Enregistrement d'une tâche
  grunt.registerTask(taskName, [description, ] taskFunction)

  // Chargement d'un plugin
  grunt.loadNpmTasks('package');

};
```

BRUNCH

- Mature: Jan 2011, very mature
- Community Support: fairly new, plenty of plugins & skeletons
- Easy to set up - use skeleton
- Introduces conventions for you
- Simple CLI - only a few commands
- Commands for dev/staging/production releases

BRUNCH WEEKNESSES

- Not using conventions causes headaches
- Not easy to set up with existing projects
- Skeleton set ups not maintained as fast as plugins
- Not as supported as Grunt/Gulp

INSTALL BRUNCH

Filename: brunch-config.js

```
> npm install -g brunch
```

BROCCOLI

Mainly focused on Ember.js apps, and ships with Ember CLI

- Mature: Feb 2014, still in beta
- Community Support: in Beta
- Trees allow dev to think of assets
- Provides caching for map files
- Makes some conventions for you - assets
- Watching files handled by serve, only rebuilds whats needed

BROCCOLI WEEKNESSES

- No parallelism

INSTALL BROCCOLI

Filename: brocfile.js

```
Install Broccoli globaly
> npm install --g broccoli-cli
```

```
Install Broccoli in your project
> npm install --save-dev broccoli
```

TEST FRAMEWORKS

- Jasmine
- Karma
- Istanbul
- Protractor

JASMINE

- Started in 2010
- Huge community - Most popular
- Behavior-driven development framework for testing JavaScript code
- Doesn't require DOM, can be used serverside or in the browser
- Obvious syntax
- Easy to write tests
- Async Support
- Continuous Integration

INSTALL JASMINE

```
Install jasmine globaly
> npm install -g jasmine

Install jasmine plugin for Grunt/Gulp
> npm i grunt-jasmine-runner --save-dev
> npm install gulp-jasmine --save-dev
```

BASICS

```
describe("Test suite", function() {
  it("contains spec with an expectation", function() {
    expect(true).toBe(true);
  });
});
```

JASMINE USAGE WITH GULP

```
var gulp = require('gulp');
var jasmine = require('gulp-jasmine');

gulp.task('default', function () {
    return gulp.src('spec/test.js')
        .pipe(jasmine());
});
> gulp
```

AVAILABLE MATCHERS

- toBe()
- toEqual()
- toMatch()
- toBeDefined()
- toBeUndefined()
- toBeNull()
- toBeTruthy()
- toBeFalsy()
- toContain()
- toBeLessThan()
- toBeGreaterThan()
- toBeCloseTo()
- toThrow()

Above matchers can be chained with the Not() function. e.g. not.toBe()

KARMA

- Executes tests and source in a browser
- Lots of plugins available
- Can drive multiple browsers at once
- Built in JUnit reporter

INSTALL KARMA

```
Install Karma command line tool globaly
> npm install -g karma-cli

Install Karma in project
> npm install karma --save-dev
```

```
> karma init
```

KARMA CONFIGURATION

```
module.exports = function(config) {
  config.set({

    basePath: '',

    frameworks: ['jasmine','browserify'],

    files: ['test/spec/**/*Spec.coffee'],

    preprocessors: {
        'test/spec/**/*.coffee': ['coffee', 'browserify']
    },

    port: 9876,

    browsers: ['Chrome', 'Firefox', 'PhantomJS']
```

ISTANBUL

- Instrument your source code
- Run your test suite against your instrumented source code
- Store your coverage results
- Allows you to generate coverage report
- HTML and LCOV reporting

INSTALL

```
> npm install istanbul

> npm install grunt-istanbul --save-dev

> npm install gulp-istanbul --save-dev
```

PROTRACTOR

- AngularJS E2E Testing Framework
- Built on Selenium's WebDriver API
- Built on top of Jasmine framework
- Extension for all browsers
- Every action is asynchronous.
- Rapid development.
- Allows to test your app the way end user will use it.

INSTALL PROTRACTOR

```
Install protractor globally
> npm install protractor -g

Install protractor in your project using Grunt
> npm i grunt-protractor-runner --save-dev

Install protractor in your project using Gulp
> npm install gulp-protractor --save-dev

Scaffolding is available through YO as well
> npm install -g generator-protractor
> yo protractor

Update WebDriver
> webdriver-manager update
```

SAMPLE CONFIGURATION FILE

```
exports.config = {
  seleniumAddress: 'http://localhost:4444/wd/hub',
  specs: ['spec.js'],
  multiCapabilities: [{
    browserName: 'firefox'
  }, {
    browserName: 'chrome'
  }]
}
```

SAMPLE SPEC.JS

```
describe('angularjs homepage', function() {
  var firstNumber = element(by.model('first'));
  var secondNumber = element(by.model('second'));
  var goButton = element(by.id('gobutton'));
  var latestResult = element(by.binding('latest'));
  var history = element.all(by.repeater('result in memory'));

  function add(a, b) {
    firstNumber.sendKeys(a);
    secondNumber.sendKeys(b);
    goButton.click();
  }

  beforeEach(function() {
    browser.get('http://juliemr.github.io/protractor-demo/');
  });
```

EXECUTE E2E TEST

```
> webdriver-manager start

> protractor <path to conf file>
```

PROTRACTOR API

SELECTORS

- by class name
- by css
- by id
- by linkText
- by partialLinktext
- by name
- by xpath
- by binding
- by input
- by repeater
- by model

METHODS

- clear()
- click()
- getAttribute(name)
- getCSSValue(propertyName)
- getLocation()
- getSize()
- getTagName()
- getText()
- isDisplayed()
- isEnabled()
- isSelected()
- sendKeys(keysToSend)

CHOOSING WORKFLOW AUTOMATION TOOL

- No tool is wrong, just different approaches
- Each tool has strengths and weaknesses.
- Your job will be to identify which tool may be best suited for your needs.

RESOURCES

- nodeJS
- npm
- Bower
- Yemon (YO)
- Gulp
- Grunt
- Brunch
- Broccoli
- List of JavaScript Build Tools
- Jasmine
- Karma
- Istanbul
- Protractor

QUESTIONS ?

@MAXY_ERMAYANK