

# Continuous integration hacks for Angular with Jenkins



**ANGULAR**  
TRAINING

@AlainChautard - [angulartraining.com](http://angulartraining.com)

# About me - Alain Chautard (or just AI)

Google Developer Expert in Web technologies / Angular

Java developer since 2006

Angular JS addict since 2011

Organizer of the Sacramento Angular Meetup group

Web consultant (60%) / trainer (40% of the time) @ [angulartraining.com](http://angulartraining.com)



## To do list for CI with Angular and Jenkins

- Have Node.JS and Angular CLI installed
- Run **ng test** and **ng e2e** to make sure all is good
- Run **ng build --prod** or anything close to it (AOT build)
- Make sure your app size is under control
- On success, deploy your build artifacts

# Have Node.JS and Angular CLI installed

- Use the NodeJS plugin for Jenkins: Multiple version support as well as global npm packages like Angular CLI
- This plugin can be found at:  
<https://wiki.jenkins.io/display/JENKINS/NodeJS+Plugin>

NodeJS installations

NodeJS

Name

Install automatically ?

Install from nodejs.org

Version

Global npm packages to install

Specify list of packages to install globally -- see npm install -g. Note that you can fix the packages version by using the syntax "packageName@version"

Global npm packages refresh hours

Duration, in hours, before 2 npm cache update. Note that 0 will always update npm cache

[Delete Installer](#)

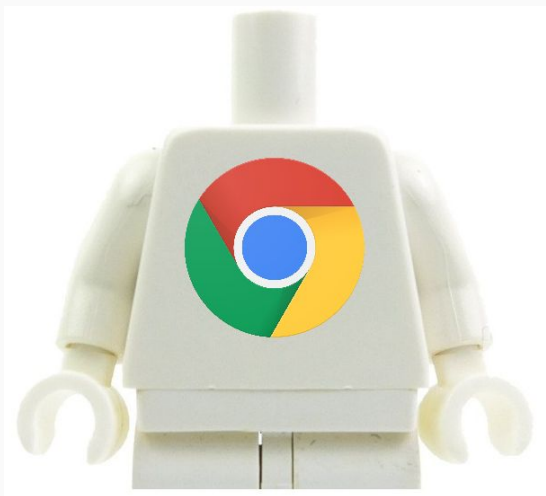
## Run `ng test` and `ng e2e` to make sure all is good

- The NodeJS plugin gets Angular CLI installed for us: Running these two commands is not an issue
- The main problem then is... Those tests need a browser to run!
- And most CI servers use Linux, which means:
  - No Google Chrome available
  - No UI to launch a browser
- So what are our options?

Run `ng test` and `ng e2e` to make sure all is good

- **Solution:** Tests can be run with a UI-less browser
- Two main options:

## Chrome Headless



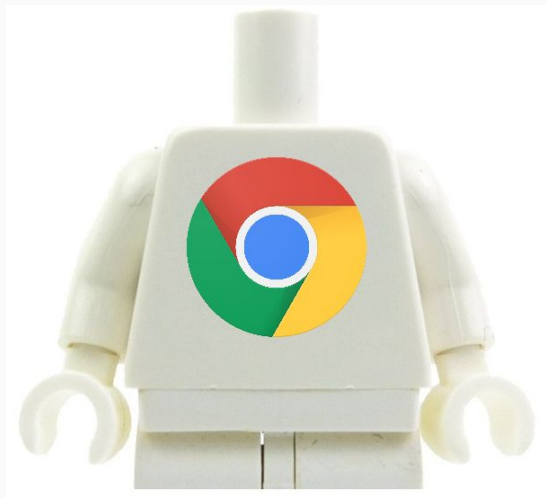
## Phantom JS



# Run `ng test` and `ng e2e` to make sure all is good

- **Chrome Headless** relies on a simple flag in Karma config:

```
customLaunchers: {  
  ChromeHeadless: {  
    base: 'Chrome',  
    flags: [  
      '--headless', '--disable-gpu',  
      '--no-sandbox',  
      '--remote-debugging-port=9222']  
    }  
  },  
  browsers: ['ChromeHeadless'],  
  singleRun: true
```



# Run `ng test` and `ng e2e` to make sure all is good

- Phantom JS requires a specific Karma launcher (to be installed with npm)

```
plugins: [  
  require('karma-jasmine'),  
  require('karma-chrome-launcher'),  
  require('karma-phantomjs-launcher'),  
  require('karma-jasmine-html-reporter'),  
  ...  
],  
...  
browsers: ['PhantomJS'],  
singleRun: true
```





# Generate a code coverage report

- Instead of running just `ng test`, use:

`ng test --code-coverage`

- This will generate a coverage report for your project:

## All files

89.47% Statements 34/38 66.67% Branches 2/3 55.56% Functions 5/9 87.5% Lines 28/32

File	Statements	Branches	Functions	Lines
src	100%	16/16	100%	100%
src/app	84.62%	11/13	100%	77.78%
src/app/book-list	77.78%	7/9	66.67%	71.43%

Run `ng build --prod` or anything close to it (AOT build)

- Why run an AOT build?
  - To validate your HTML templates
  - To make sure your app is performant
  - To get an idea of how big your application is

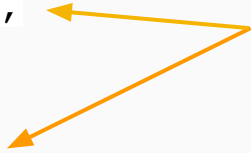
# Make sure your app size is under control

- Use budgets to make the build fail if too many dependencies get added:

```
"configurations": {  
  "production": {  
    "budgets": [  
      {  
        "type": "bundle",  
        "name": "vendor",  
        "baseline": "750kb",  
        "warning": "100kb",  
        "error": "200kb"  
      }  
    ],  
  },  
}
```

Build fails is vendor bundle size > 950kb

Warning if > 850 kb and < 950 kb



# How to build for different environments?

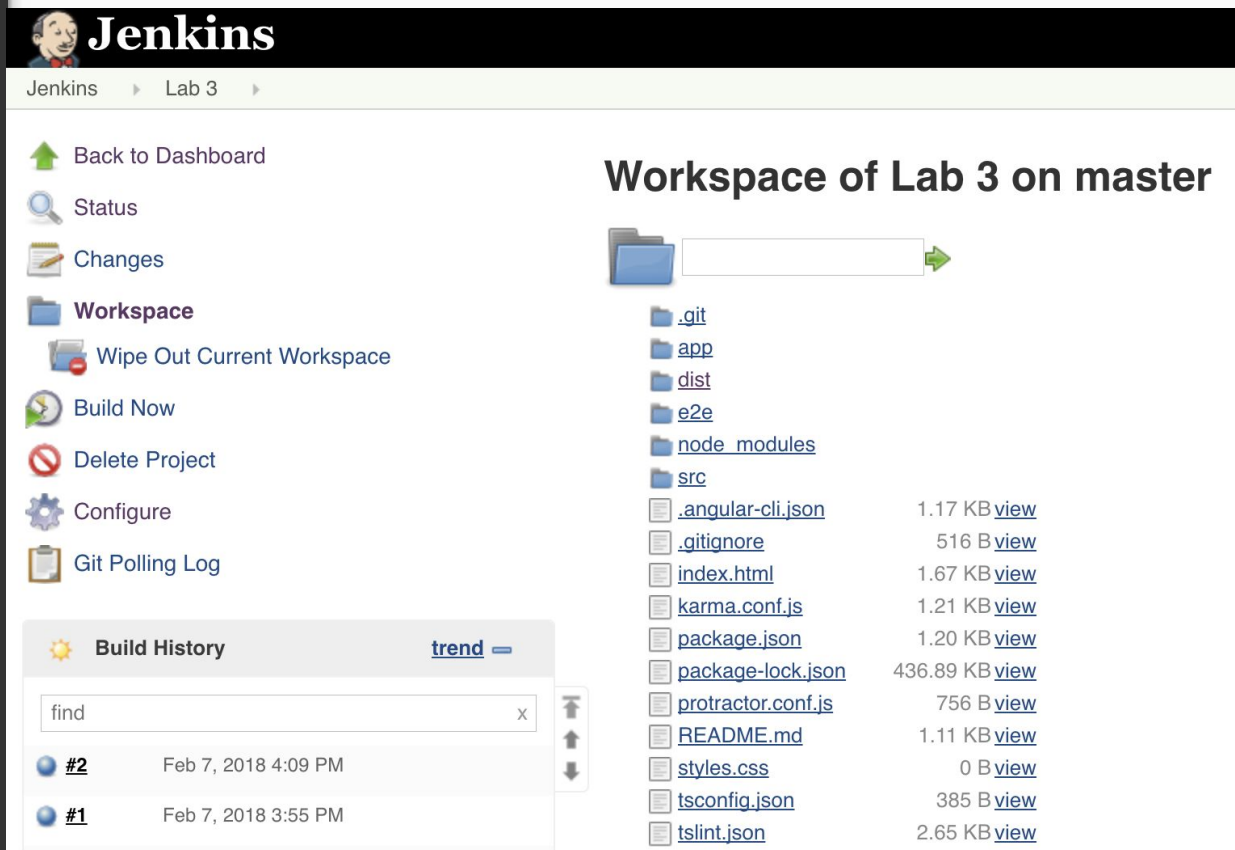
- You might need different builds (QA, prod, test, dev)
- Configurations are the solution for these distinct builds:

```
"configurations": {  
  "production": {  
    "fileReplacements": [  
      { "replace": "src/environments/environment.ts",  
        "with": "src/environments/environment.prod.ts"}  
    ]  
  }, "qa": {  
    "fileReplacements": [ ...  
      "with": "src/environments/environment.qa.ts"}  
    ]  
  }  
}
```

# Build Artifacts

Build artifacts can be found in the workspace directory in Jenkins.

That's where builds can generate reports such as code coverage from `ng test --code-coverage`.



The screenshot displays the Jenkins web interface for a job named "Lab 3". The top navigation bar includes "Jenkins" and "Lab 3". The main content area is titled "Workspace of Lab 3 on master" and shows a file explorer view of the workspace directory. The directory contains several folders and files, including configuration files, source code, and build artifacts. A "Build History" section is visible at the bottom left, showing the last two builds.

**Jenkins**

Jenkins > Lab 3

- Back to Dashboard
- Status
- Changes
- Workspace**
- Wipe Out Current Workspace
- Build Now
- Delete Project
- Configure
- Git Polling Log

### Workspace of Lab 3 on master

- .git
- app
- dist
- e2e
- node\_modules
- src
- .angular-cli.json 1.17 KB [view](#)
- .gitignore 516 B [view](#)
- index.html 1.67 KB [view](#)
- karma.conf.js 1.21 KB [view](#)
- package.json 1.20 KB [view](#)
- package-lock.json 436.89 KB [view](#)
- protractor.conf.js 756 B [view](#)
- README.md 1.11 KB [view](#)
- styles.css 0 B [view](#)
- tsconfig.json 385 B [view](#)
- tslint.json 2.65 KB [view](#)

#### Build History

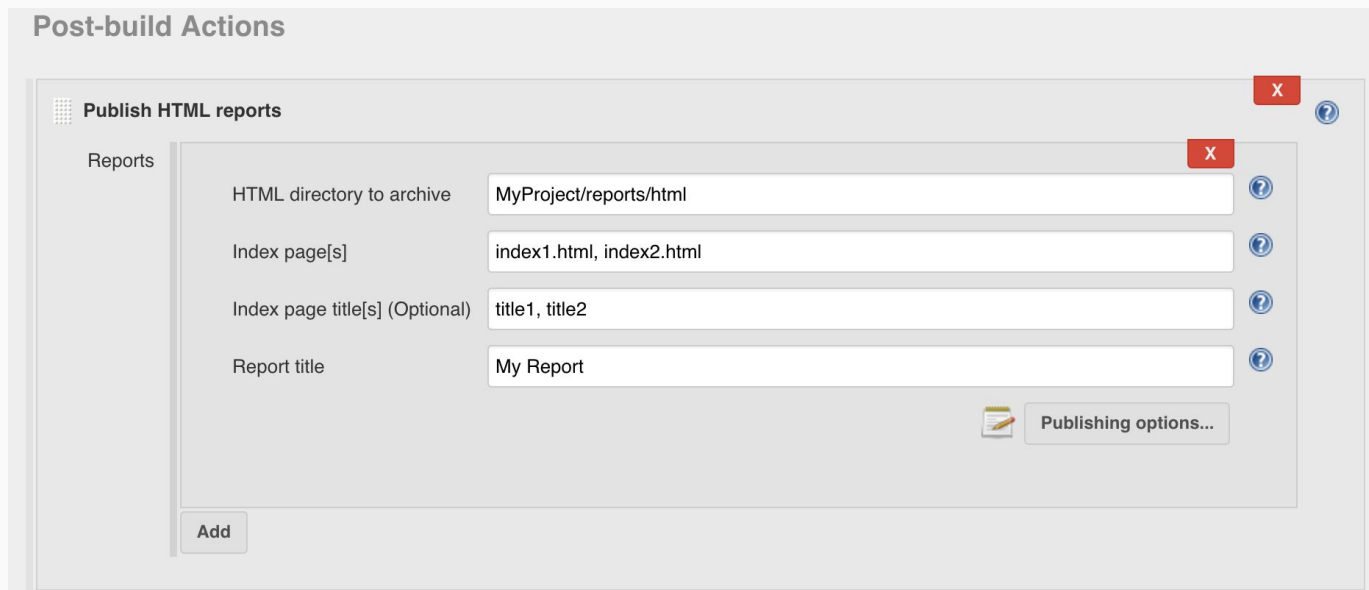
find x

#2	Feb 7, 2018 4:09 PM
#1	Feb 7, 2018 3:55 PM

# Make sure you test reports are easy to find

- HTML Publisher plugin creates links to reports created by your build.
- This plugin can be found at:

<https://wiki.jenkins.io/display/JENKINS/HTML+Publisher+Plugin>



The screenshot displays the 'Post-build Actions' configuration interface in Jenkins. A section titled 'Publish HTML reports' is expanded, showing a list of reports to be published. The 'Reports' list contains one entry with the following fields:

Field	Value
HTML directory to archive	MyProject/reports/html
Index page[s]	index1.html, index2.html
Index page title[s] (Optional)	title1, title2
Report title	My Report

Below the report list is an 'Add' button. To the right of the report list is a 'Publishing options...' button with a pencil icon. The interface includes standard UI elements like close (X) and help (?) buttons for the section and individual fields.

# On success, deploy your build artifacts

- If you need to deploy your artifacts, a simple FTP / SCP / CP of the **dist** folder is all you need. Many plugins exist to do so:

## Publish Over

Created by Bap bap2000, last modified on Mar 27, 2013

### Goal

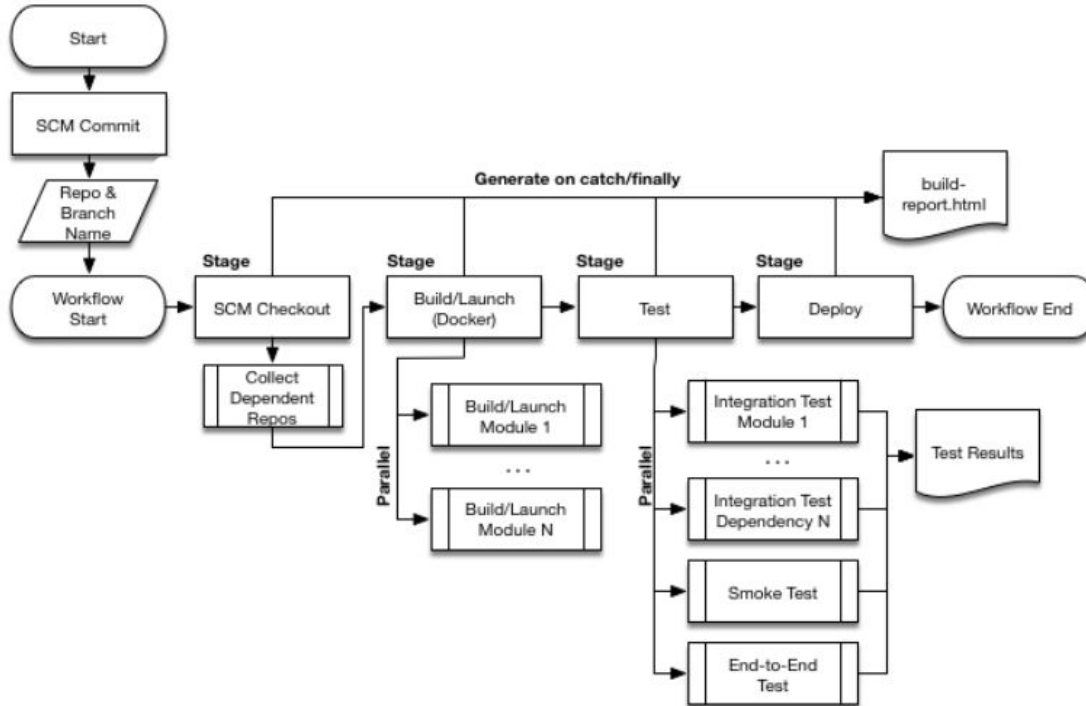
The goal of the Publish Over plugins is to provide a consistent set of features and behaviours when sending build artifacts ... somewhere.

### Publish Over plugins

- Publish Over CIFS Plugin - send artifacts to a windows share
- Publish Over FTP Plugin - send artifacts to an FTP server
- Publish Over SSH Plugin - send artifacts to an SSH server (using SFTP) and/or execute commands over SSH

# We now have a full continuous delivery process in place

- Congratulations! You now have a process to move your code from development to production





# Thanks for your attention

Link to slides:

<https://goo.gl/pmPZ91>



My blog:

<https://blog.angulartraining.com>

Twitter:

@AlainChautard



**ANGULAR**  
TRAINING