The Dockerfile explosion

Gareth Rushgrove

Senior Software Engineer
Puppet

dockercon 16

The Dockerfile explosion and the need for higher level tools

Introductions

Who am I and what am I doing here

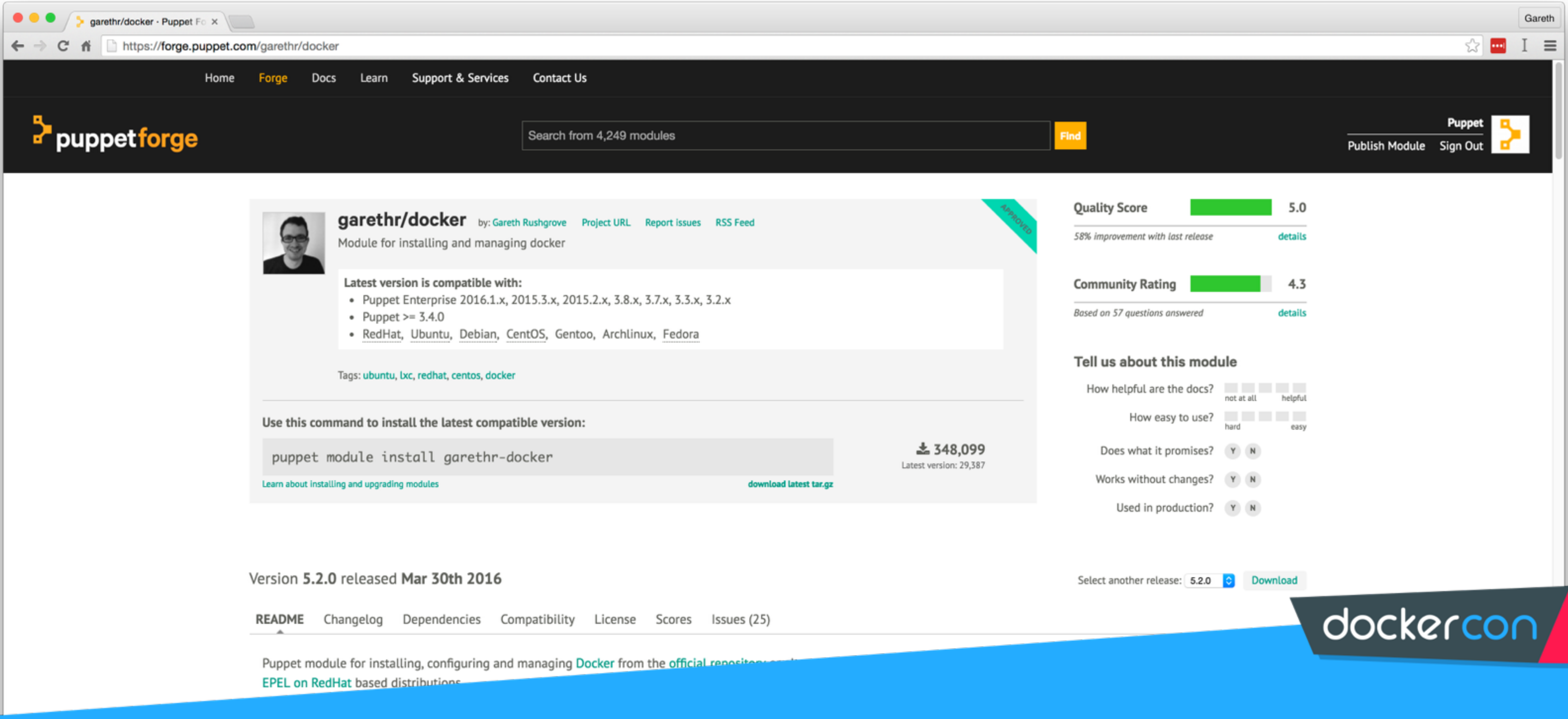
@garethr

dockercon 16

puppet

The shortest path to better software.

Built the Puppet Docker module



garethr/docker · Puppet Fo ×

https://forge.puppet.com/garethr/docker

Home Forge Docs Learn Support & Services Contact Us

puppetforge

Search from 4,249 modules Find

Puppet
Publish Module Sign Out

garethr/docker by: Gareth Rushgrove Project URL Report issues RSS Feed

Module for installing and managing docker

APPROVED

Latest version is compatible with:
- Puppet Enterprise 2016.1.x, 2015.3.x, 2015.2.x, 3.8.x, 3.7.x, 3.3.x, 3.2.x
- Puppet >= 3.4.0
- RedHat, Ubuntu, Debian, CentOS, Gentoo, Archlinux, Fedora

Tags: ubuntu, lxc, redhat, centos, docker

Use this command to install the latest compatible version:

```
puppet module install garethr-docker
```

Learn about installing and upgrading modules download latest tar.gz

348,099
Latest version: 29,387

Quality Score	🟩	5.0
58% improvement with last release		details
Community Rating	🟩	4.3
Based on 57 questions answered		details

Tell us about this module

How helpful are the docs? not at all … helpful
How easy to use? hard … easy
Does what it promises? Y N
Works without changes? Y N
Used in production? Y N

Version **5.2.0** released **Mar 30th 2016**

Select another release: 5.2.0 Download

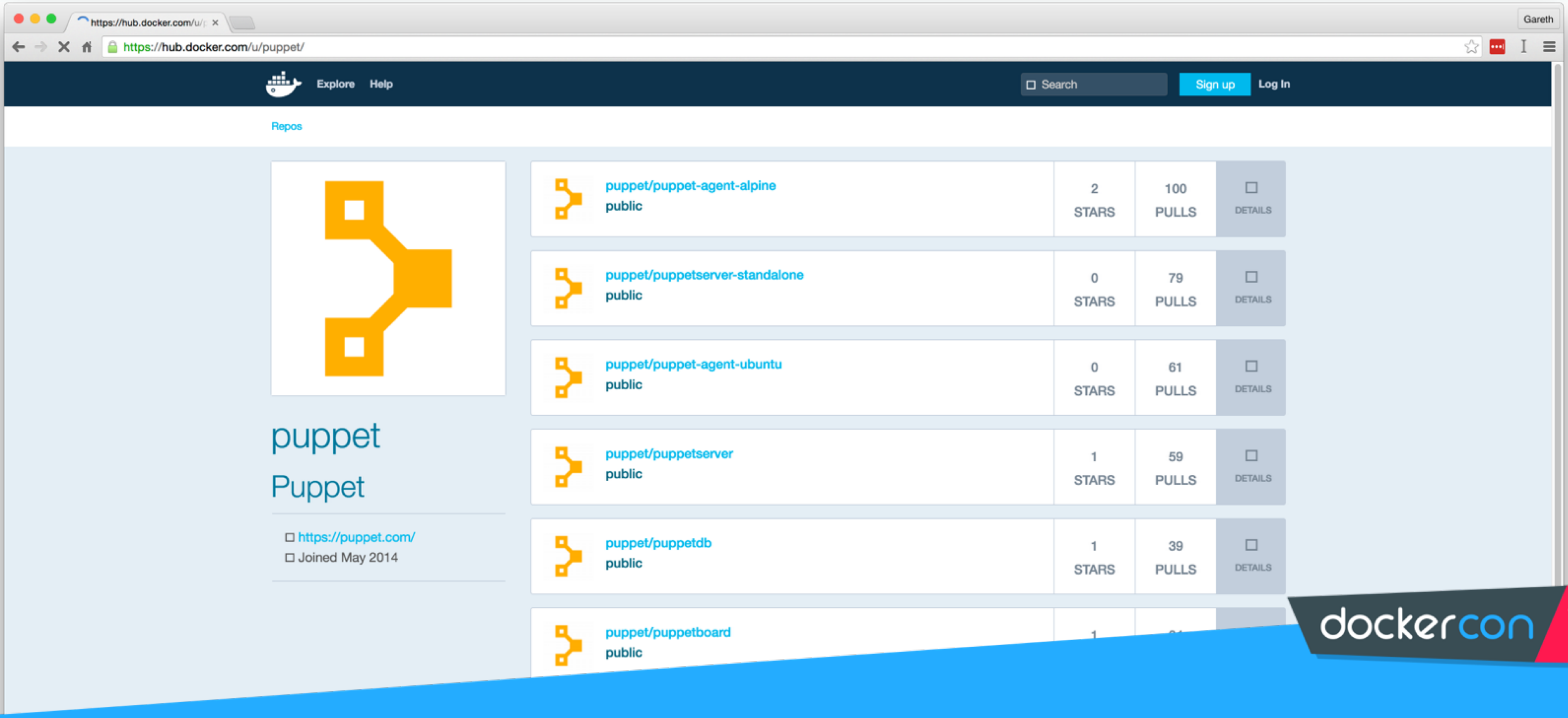
README Changelog Dependencies Compatibility License Scores Issues (25)

Puppet module for installing, configuring and managing Docker from the official repository

EPEL on RedHat based distributions

dockercon 16

Maintain the Puppet images



https://hub.docker.com/u/puppet/

Explore Help Search Sign up Log In

Repos

puppet

Puppet

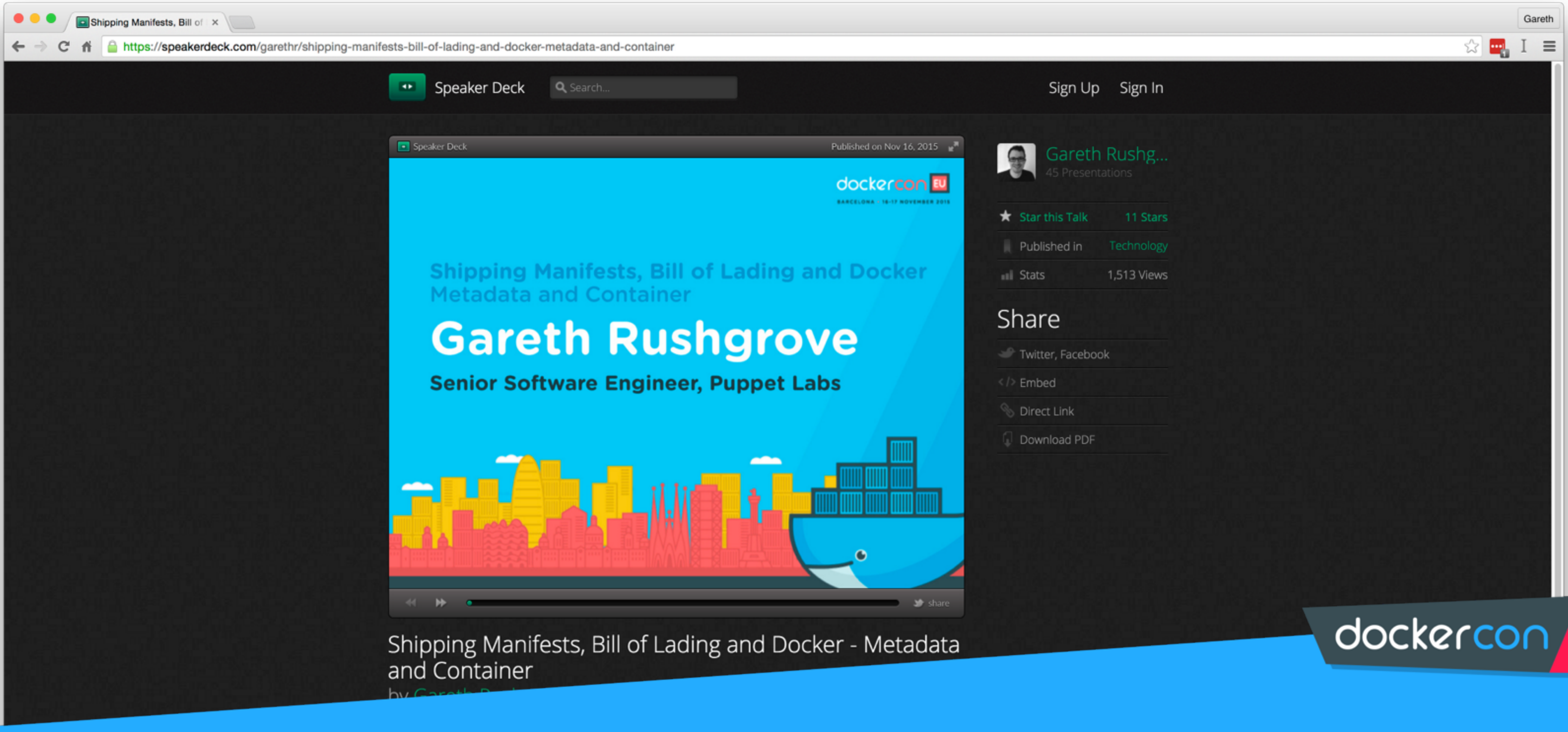
https://puppet.com/

Joined May 2014

puppet/puppet-agent-alpine public	2 STARS	100 PULLS	DETAILS
puppet/puppetserver-standalone public	0 STARS	79 PULLS	DETAILS
puppet/puppet-agent-ubuntu public	0 STARS	61 PULLS	DETAILS
puppet/puppetserver public	1 STARS	59 PULLS	DETAILS
puppet/puppetdb public	1 STARS	39 PULLS	DETAILS
puppet/puppetboard public	1		

dockercon 16

Obsessed with metadata



Shipping Manifests, Bill of

https://speakerdeck.com/garethr/shipping-manifests-bill-of-lading-and-docker-metadata-and-container

Speaker Deck

Published on Nov 16, 2015

Shipping Manifests, Bill of Lading and Docker Metadata and Container

Gareth Rushgrove

Senior Software Engineer, Puppet Labs

Gareth Rushg...

45 Presentations

Star this Talk 11 Stars

Published in Technology

Stats 1,513 Views

Share

Twitter, Facebook

Embed

Direct Link

Download PDF

Shipping Manifests, Bill of Lading and Docker - Metadata and Container

dockercon 16

A brief history of Dockerfile

Docker can build images automatically by reading the instructions from a **Dockerfile**

From the official docs at https://docs.docker.com/engine/reference/builder/

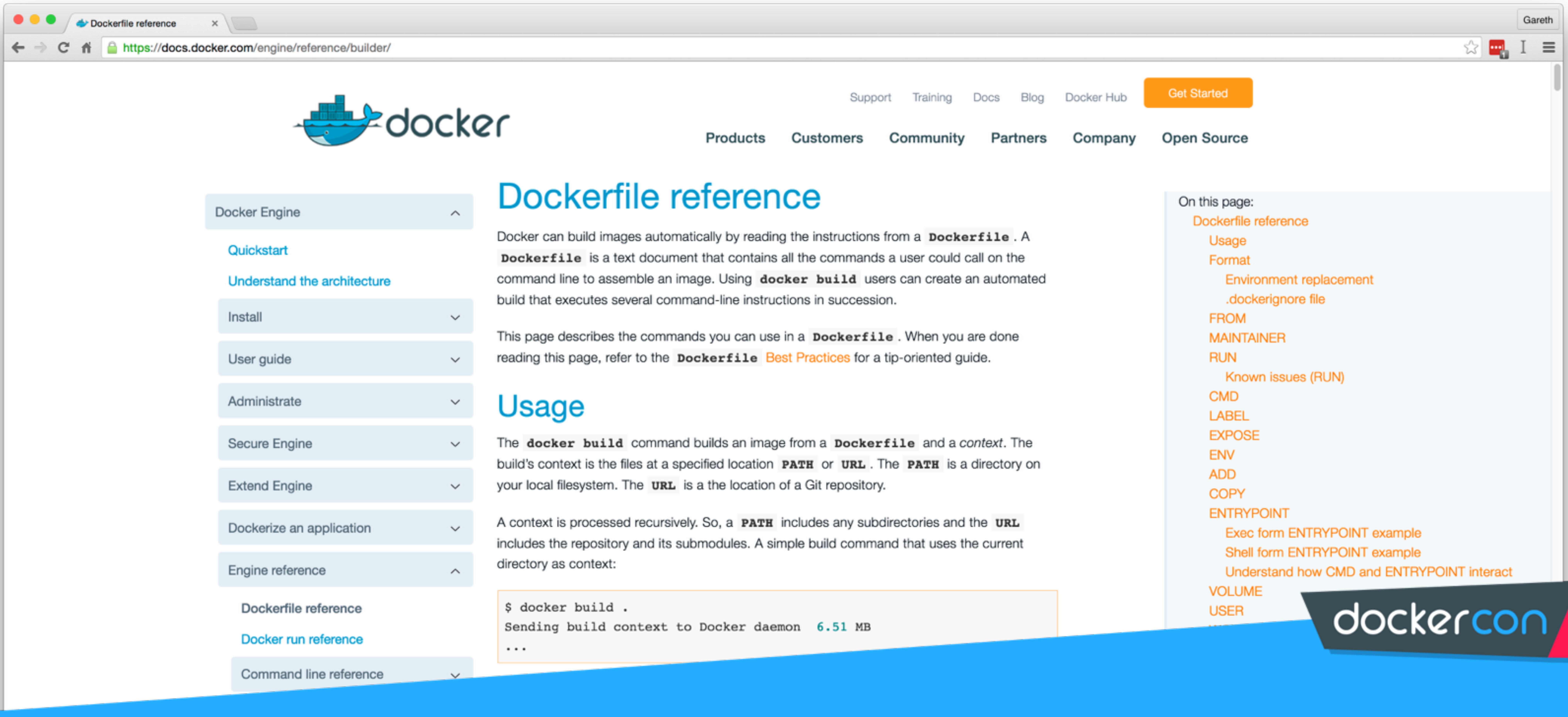
A **Dockerfile** is a text document that contains all the commands a user could call on the command line to assemble an image.

From the official docs at https://docs.docker.com/engine/reference/builder/

A simple Dockerfile

```
FROM ubuntu
# Install vnc, xvfb in order to create a 'fake' display and fire
RUN apt-get update && apt-get install -y x11vnc xvfb firefox
RUN mkdir ~/.vnc
# Setup a password
RUN x11vnc -storepasswd 1234 ~/.vnc/passwd
# Autostart firefox (might not be the best way, but it does the
RUN bash -c 'echo "firefox" >> /.bashrc'
EXPOSE 5900
CMD    ["x11vnc", "-forever", "-usepw", "-create
```

Dockerfile reference



Dockerfile reference

Docker can build images automatically by reading the instructions from a **`Dockerfile`**. A **`Dockerfile`** is a text document that contains all the commands a user could call on the command line to assemble an image. Using **`docker build`** users can create an automated build that executes several command-line instructions in succession.

This page describes the commands you can use in a **`Dockerfile`**. When you are done reading this page, refer to the **`Dockerfile`** Best Practices for a tip-oriented guide.

Usage

The **`docker build`** command builds an image from a **`Dockerfile`** and a *context*. The build's context is the files at a specified location **`PATH`** or **`URL`**. The **`PATH`** is a directory on your local filesystem. The **`URL`** is a the location of a Git repository.

A context is processed recursively. So, a **`PATH`** includes any subdirectories and the **`URL`** includes the repository and its submodules. A simple build command that uses the current directory as context:

```
$ docker build .
Sending build context to Docker daemon  6.51 MB
...
```

dockercon 16

Commands you know

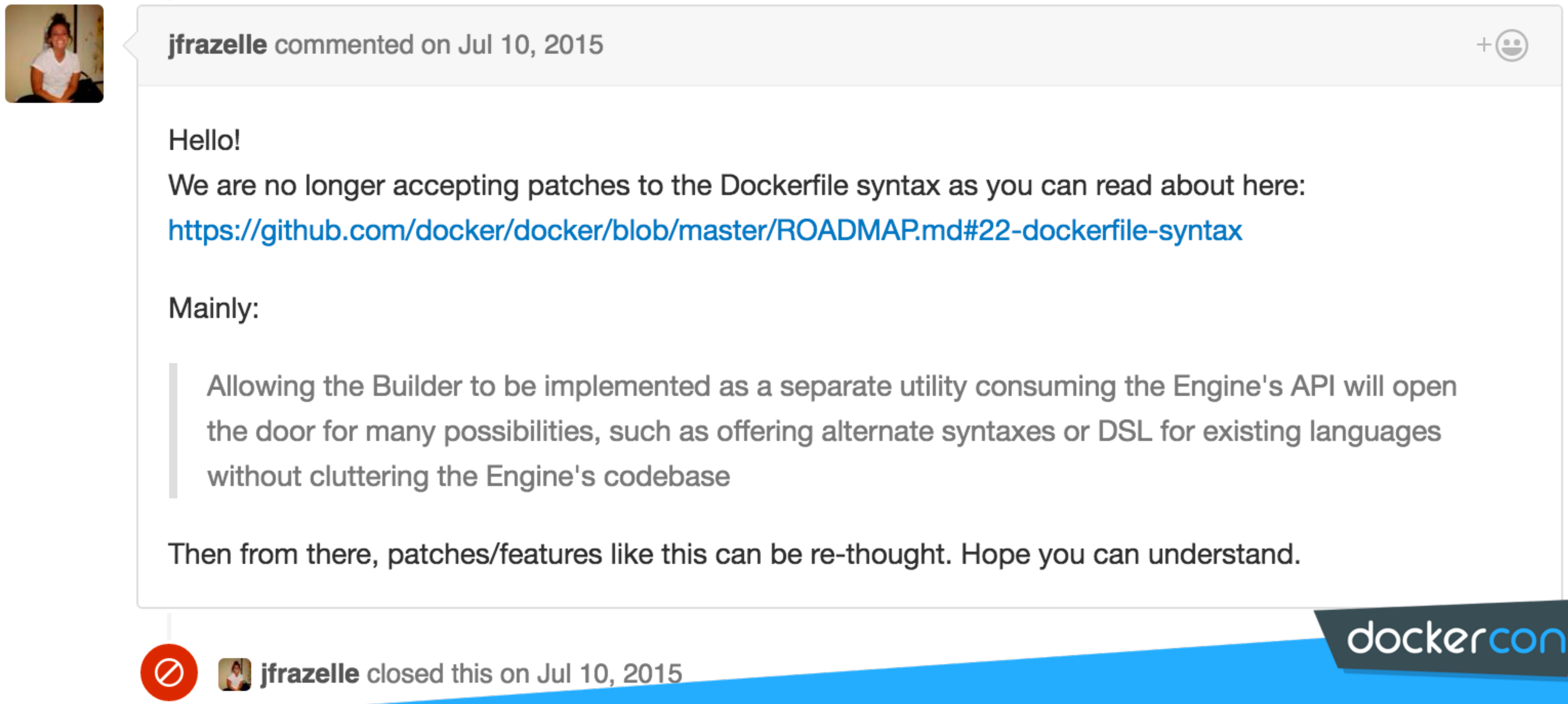
```
MAINTAINER <name>
RUN <command>
CMD ["executable","param1","param2"]
EXPOSE <port> [<port>...]
ADD <src>... <dest>
ENV <key> <value>
WORKDIR /path/to/workdir
USER daemon
VOLUME ["/data"]
ENTRYPOINT ["executable", "param1", “param2"]
COPY <src>... <dest>
```

Commands you don’t know

```
ONBUILD [INSTRUCTION]
STOPSIGNAL signal
ARG <name>[=<default value>]
LABEL <key>=<value> <key>=<value> <key>=<value> …
HEALTHCHECK [OPTIONS] CMD command
SHELL ["executable", "parameters"]
```

dockercon 16

Close ALL the issues



jfrazelle commented on Jul 10, 2015

Hello!
We are no longer accepting patches to the Dockerfile syntax as you can read about here:
https://github.com/docker/docker/blob/master/ROADMAP.md#22-dockerfile-syntax

Mainly:

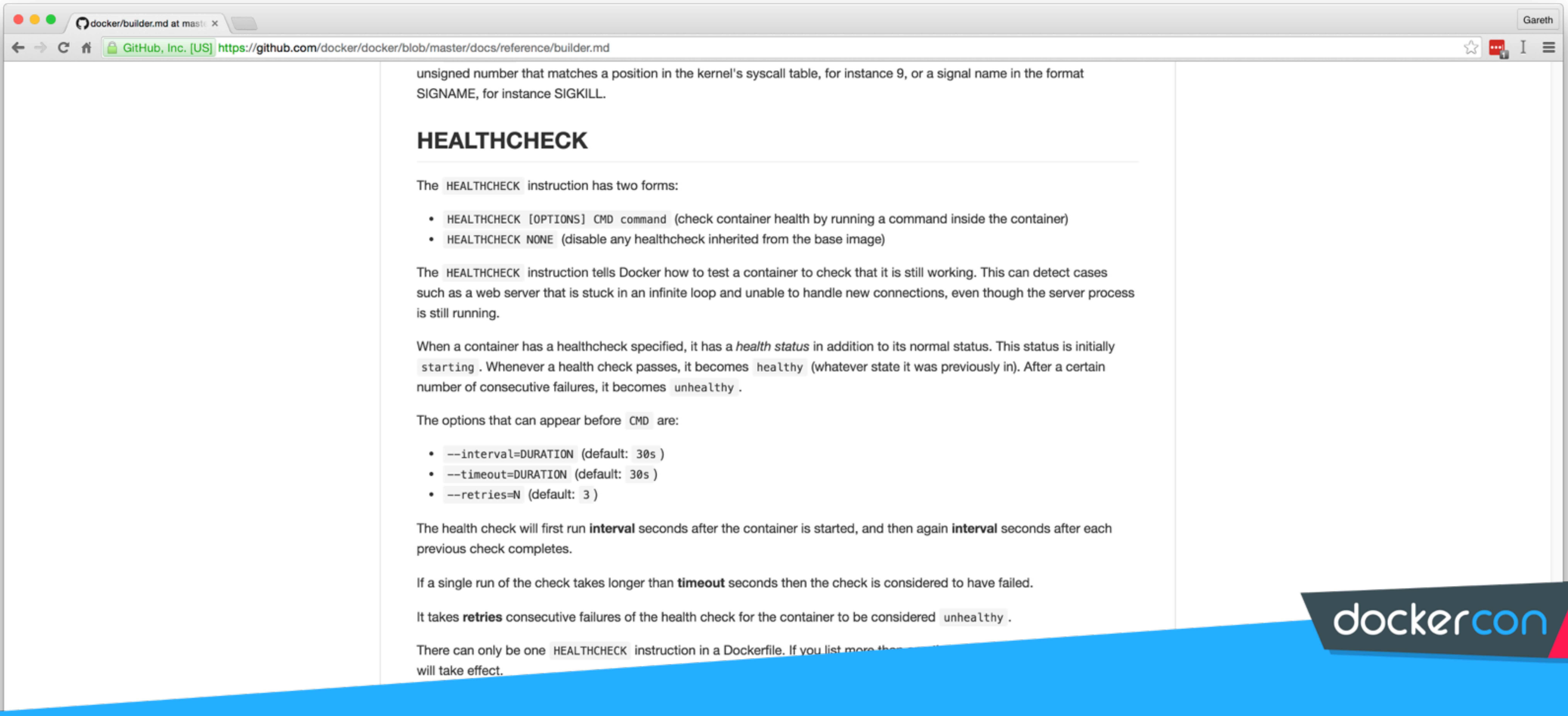
> Allowing the Builder to be implemented as a separate utility consuming the Engine's API will open the door for many possibilities, such as offering alternate syntaxes or DSL for existing languages without cluttering the Engine's codebase

Then from there, patches/features like this can be re-thought. Hope you can understand.

jfrazelle closed this on Jul 10, 2015

Although this is not a definitive move, we temporarily won't accept more patches to the Dockerfile syntax for several reasons

HEALTHCHECK coming in 1.12


docker/builder.md at maste
GitHub, Inc. [US] https://github.com/docker/docker/blob/master/docs/reference/builder.md
Gareth

unsigned number that matches a position in the kernel's syscall table, for instance 9, or a signal name in the format SIGNAME, for instance SIGKILL.

HEALTHCHECK

The `HEALTHCHECK` instruction has two forms:

- `HEALTHCHECK [OPTIONS] CMD command` (check container health by running a command inside the container)
- `HEALTHCHECK NONE` (disable any healthcheck inherited from the base image)

The `HEALTHCHECK` instruction tells Docker how to test a container to check that it is still working. This can detect cases such as a web server that is stuck in an infinite loop and unable to handle new connections, even though the server process is still running.

When a container has a healthcheck specified, it has a *health status* in addition to its normal status. This status is initially `starting`. Whenever a health check passes, it becomes `healthy` (whatever state it was previously in). After a certain number of consecutive failures, it becomes `unhealthy`.

The options that can appear before `CMD` are:

- `--interval=DURATION` (default: `30s`)
- `--timeout=DURATION` (default: `30s`)
- `--retries=N` (default: `3`)

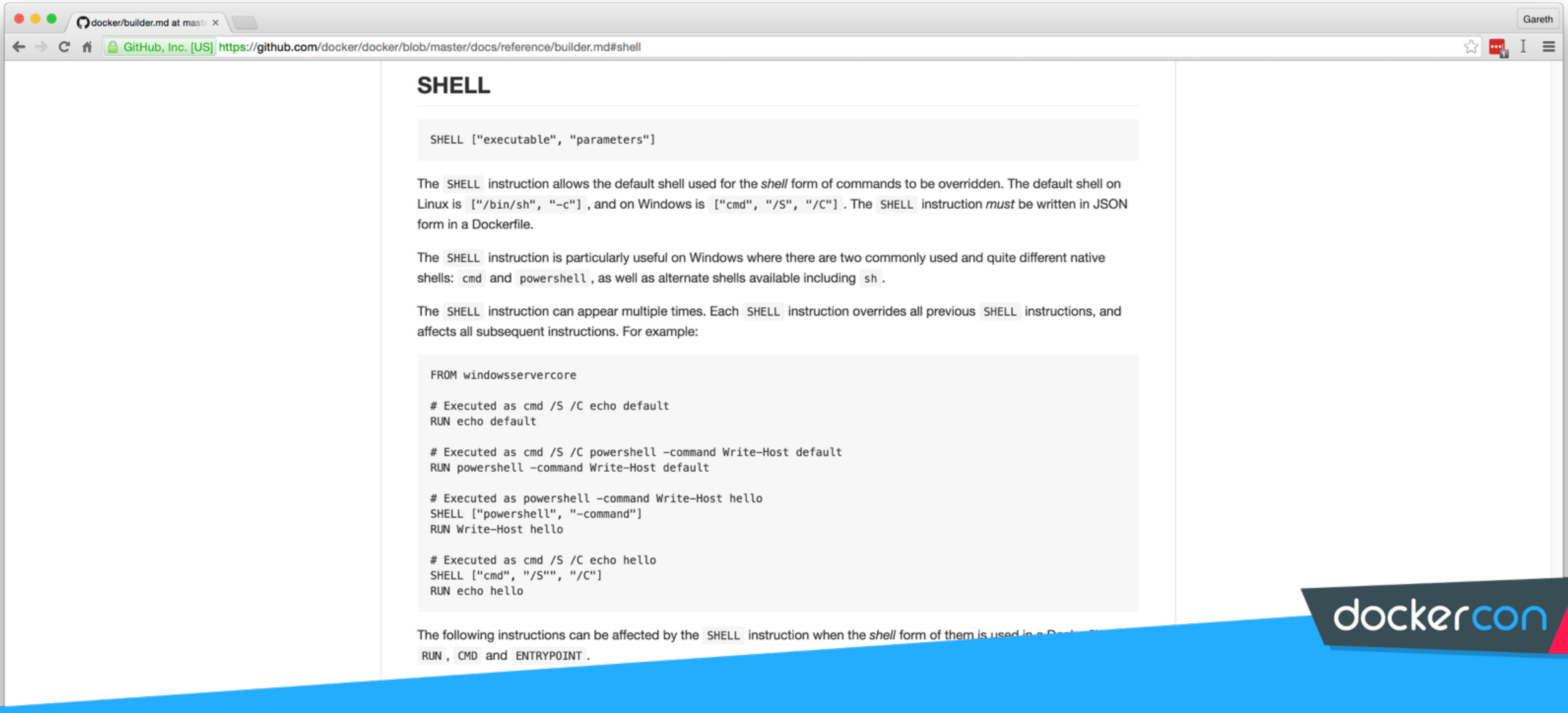
The health check will first run **interval** seconds after the container is started, and then again **interval** seconds after each previous check completes.

If a single run of the check takes longer than **timeout** seconds then the check is considered to have failed.

It takes **retries** consecutive failures of the health check for the container to be considered `unhealthy`.

There can only be one `HEALTHCHECK` instruction in a Dockerfile. If you list more than
will take effect.

SHELL coming in 1.12



docker/builder.md at mast

GitHub, Inc. [US] https://github.com/docker/docker/blob/master/docs/reference/builder.md#shell

Gareth

SHELL

```
SHELL ["executable", "parameters"]
```

The `SHELL` instruction allows the default shell used for the *shell* form of commands to be overridden. The default shell on Linux is `["/bin/sh", "-c"]`, and on Windows is `["cmd", "/S", "/C"]`. The `SHELL` instruction *must* be written in JSON form in a Dockerfile.

The `SHELL` instruction is particularly useful on Windows where there are two commonly used and quite different native shells: `cmd` and `powershell`, as well as alternate shells available including `sh`.

The `SHELL` instruction can appear multiple times. Each `SHELL` instruction overrides all previous `SHELL` instructions, and affects all subsequent instructions. For example:

```
FROM windowsservercore

# Executed as cmd /S /C echo default
RUN echo default

# Executed as cmd /S /C powershell -command Write-Host default
RUN powershell -command Write-Host default

# Executed as powershell -command Write-Host hello
SHELL ["powershell", "-command"]
RUN Write-Host hello

# Executed as cmd /S /C echo hello
SHELL ["cmd", "/S"", "/C"]
RUN echo hello
```

The following instructions can be affected by the `SHELL` instruction when the *shell* form of them is used in a
`RUN`, `CMD` and `ENTRYPOINT`.

Why Dockerfiles are great

Simplicity

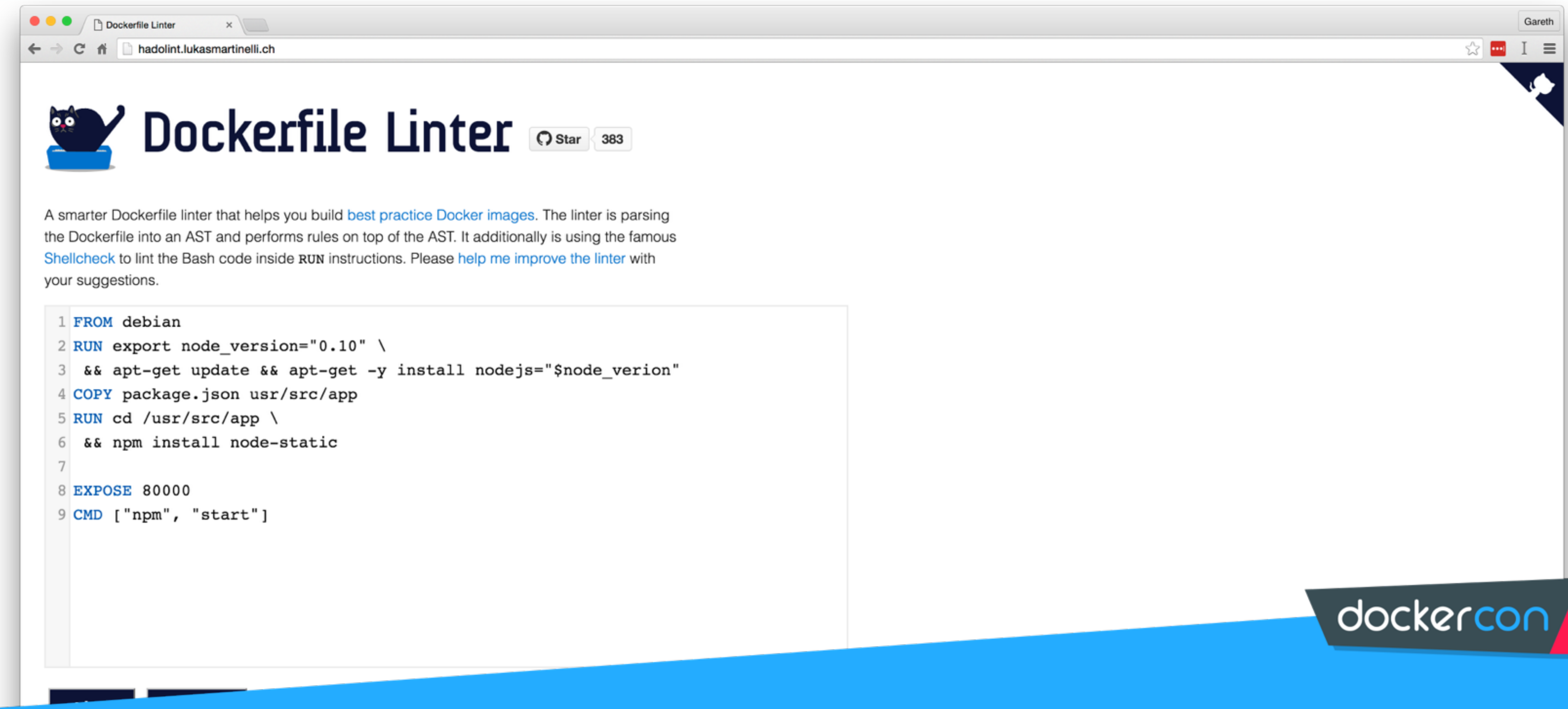
```
FROM scratch
COPY hello /
CMD ["/hello"]
```

Multi-platform support

```
PS> Install-PackageProvider ContainerImage -Force
PS> Install-ContainerImage -Name WindowsServerCore
PS> docker images

REPOSITORY          TAG                 IMAGE ID            CREA
windowsservercore   10.0.14300.1000     dbfee88ee9fd        7 we
```

Emerging tooling

Linting

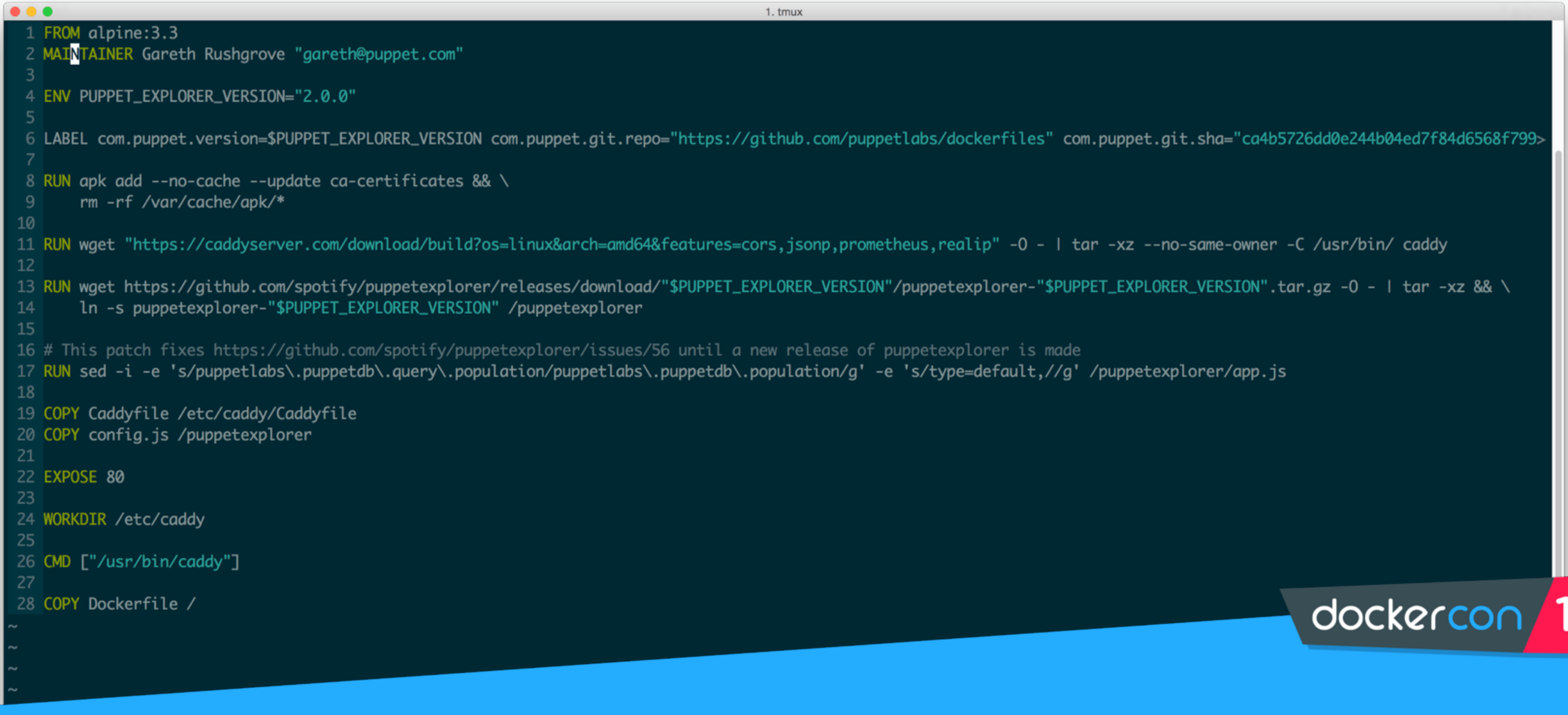


Dockerfile Linter

A smarter Dockerfile linter that helps you build best practice Docker images. The linter is parsing the Dockerfile into an AST and performs rules on top of the AST. It additionally is using the famous Shellcheck to lint the Bash code inside RUN instructions. Please help me improve the linter with your suggestions.

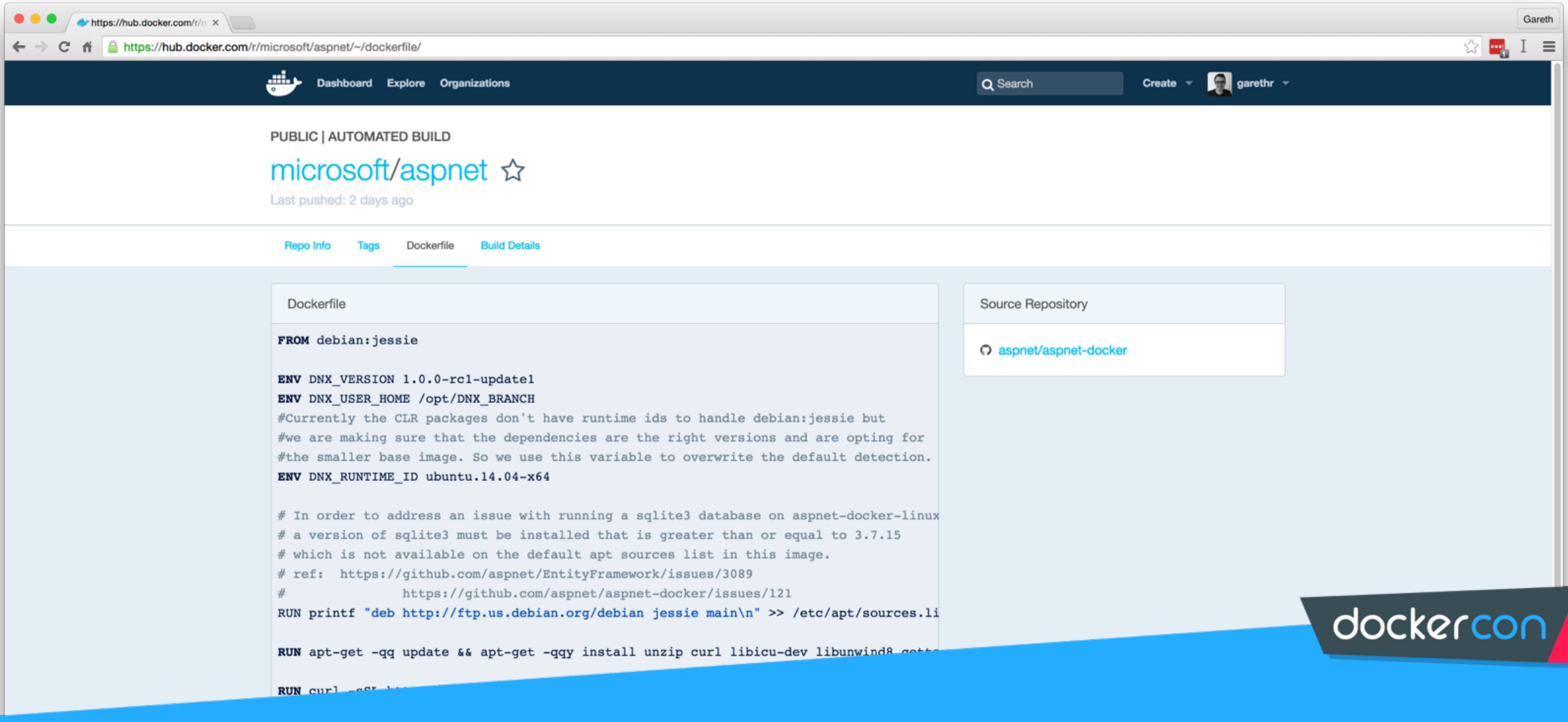
```
FROM debian
RUN export node_version="0.10" \
 && apt-get update && apt-get -y install nodejs="$node_verion"
COPY package.json usr/src/app
RUN cd /usr/src/app \
 && npm install node-static

EXPOSE 80000
CMD ["npm", "start"]
```

Editor support



```
FROM alpine:3.3
MAINTAINER Gareth Rushgrove "gareth@puppet.com"

ENV PUPPET_EXPLORER_VERSION="2.0.0"

LABEL com.puppet.version=$PUPPET_EXPLORER_VERSION com.puppet.git.repo="https://github.com/puppetlabs/dockerfiles" com.puppet.git.sha="ca4b5726dd0e244b04ed7f84d6568f799>

RUN apk add --no-cache --update ca-certificates && \
    rm -rf /var/cache/apk/*

RUN wget "https://caddyserver.com/download/build?os=linux&arch=amd64&features=cors,jsonp,prometheus,realip" -O - | tar -xz --no-same-owner -C /usr/bin/ caddy

RUN wget https://github.com/spotify/puppetexplorer/releases/download/"$PUPPET_EXPLORER_VERSION"/puppetexplorer-"$PUPPET_EXPLORER_VERSION".tar.gz -O - | tar -xz && \
    ln -s puppetexplorer-"$PUPPET_EXPLORER_VERSION" /puppetexplorer

# This patch fixes https://github.com/spotify/puppetexplorer/issues/56 until a new release of puppetexplorer is made
RUN sed -i -e 's/puppetlabs\.puppetdb\.query\.population/puppetlabs\.puppetdb\.population/g' -e 's/type=default,//g' /puppetexplorer/app.js

COPY Caddyfile /etc/caddy/Caddyfile
COPY config.js /puppetexplorer

EXPOSE 80

WORKDIR /etc/caddy

CMD ["/usr/bin/caddy"]

COPY Dockerfile /
~
~
~
~
```

Cross platform



PUBLIC | AUTOMATED BUILD

microsoft/aspnet

Last pushed: 2 days ago

Repo Info | Tags | Dockerfile | Build Details

Dockerfile

```
FROM debian:jessie

ENV DNX_VERSION 1.0.0-rc1-update1
ENV DNX_USER_HOME /opt/DNX_BRANCH
#Currently the CLR packages don't have runtime ids to handle debian:jessie but
#we are making sure that the dependencies are the right versions and are opting for
#the smaller base image. So we use this variable to overwrite the default detection.
ENV DNX_RUNTIME_ID ubuntu.14.04-x64

# In order to address an issue with running a sqlite3 database on aspnet-docker-linux
# a version of sqlite3 must be installed that is greater than or equal to 3.7.15
# which is not available on the default apt sources list in this image.
# ref:  https://github.com/aspnet/EntityFramework/issues/3089
#               https://github.com/aspnet/aspnet-docker/issues/121
RUN printf "deb http://ftp.us.debian.org/debian jessie main\n" >> /etc/apt/sources.li

RUN apt-get -qq update && apt-get -qqy install unzip curl libicu-dev libunwind8

RUN curl
```

Source Repository

aspnet/aspnet-docker

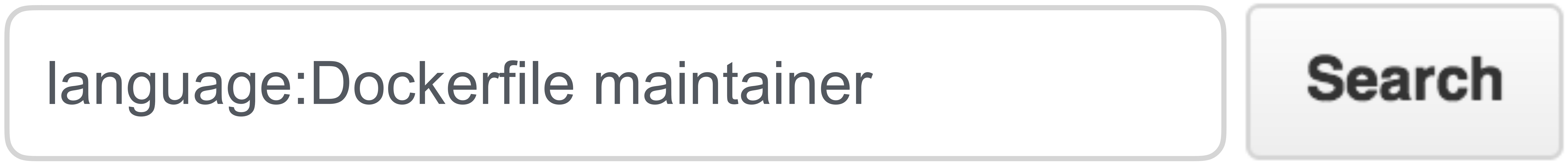
dockercon 16

Why Dockerfiles are problematic

Complexity

```
RUN apt-get update && \
    apt-get install -y wget=1.17.1-1ubuntu1 && \
    wget https://apt.example.com/release-"$UBUNTU_CODENAME".deb
    dpkg -i release-"$UBUNTU_CODENAME".deb && \
    rm release-"$UBUNTU_CODENAME".deb && \
    apt-get update && \
    apt-get install --no-install-recommends -y package=0.1.2 &&
    apt-get clean && \
    rm -rf /var/lib/apt/lists/*
```

Dockerfile proliferation


language:Dockerfile maintainer

138,062

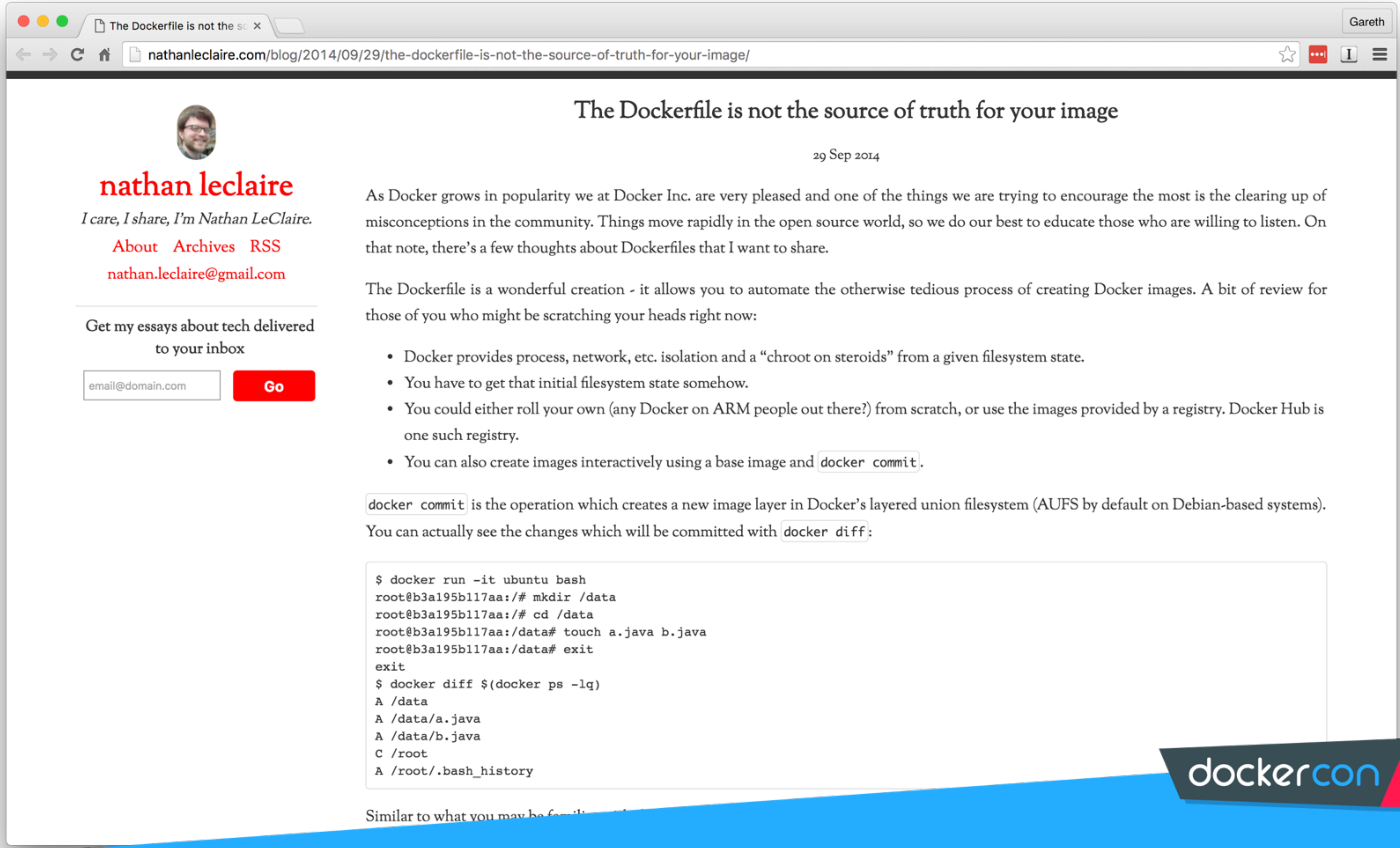
Only two approaches to reuse

Inheritance

```
FROM debian:jessie
```

Ctrl
C
Ctrl
V

Dockerfile is not the source of truth for your image



The Dockerfile is not the sc ×

Gareth

nathanleclaire.com/blog/2014/09/29/the-dockerfile-is-not-the-source-of-truth-for-your-image/

nathan leclaire

I care, I share, I'm Nathan LeClaire.

About Archives RSS

nathan.leclaire@gmail.com

Get my essays about tech delivered to your inbox

email@domain.com Go

The Dockerfile is not the source of truth for your image

29 Sep 2014

As Docker grows in popularity we at Docker Inc. are very pleased and one of the things we are trying to encourage the most is the clearing up of misconceptions in the community. Things move rapidly in the open source world, so we do our best to educate those who are willing to listen. On that note, there's a few thoughts about Dockerfiles that I want to share.

The Dockerfile is a wonderful creation - it allows you to automate the otherwise tedious process of creating Docker images. A bit of review for those of you who might be scratching your heads right now:

- Docker provides process, network, etc. isolation and a "chroot on steroids" from a given filesystem state.
- You have to get that initial filesystem state somehow.
- You could either roll your own (any Docker on ARM people out there?) from scratch, or use the images provided by a registry. Docker Hub is one such registry.
- You can also create images interactively using a base image and `docker commit`.

`docker commit` is the operation which creates a new image layer in Docker's layered union filesystem (AUFS by default on Debian-based systems). You can actually see the changes which will be committed with `docker diff`:

```
$ docker run -it ubuntu bash
root@b3a195b117aa:/# mkdir /data
root@b3a195b117aa:/# cd /data
root@b3a195b117aa:/data# touch a.java b.java
root@b3a195b117aa:/data# exit
exit
$ docker diff $(docker ps -lq)
A /data
A /data/a.java
A /data/b.java
C /root
A /root/.bash_history
```

Similar to what you may be famili

dockercon 16

The Dockerfile generally works beautifully for the class of problem for which it was designed

Nathan Leclair, Docker Inc

The Dockerfile is a tool for creating images, but it is not the only weapon in your arsenal

Nathan Leclair, Docker Inc

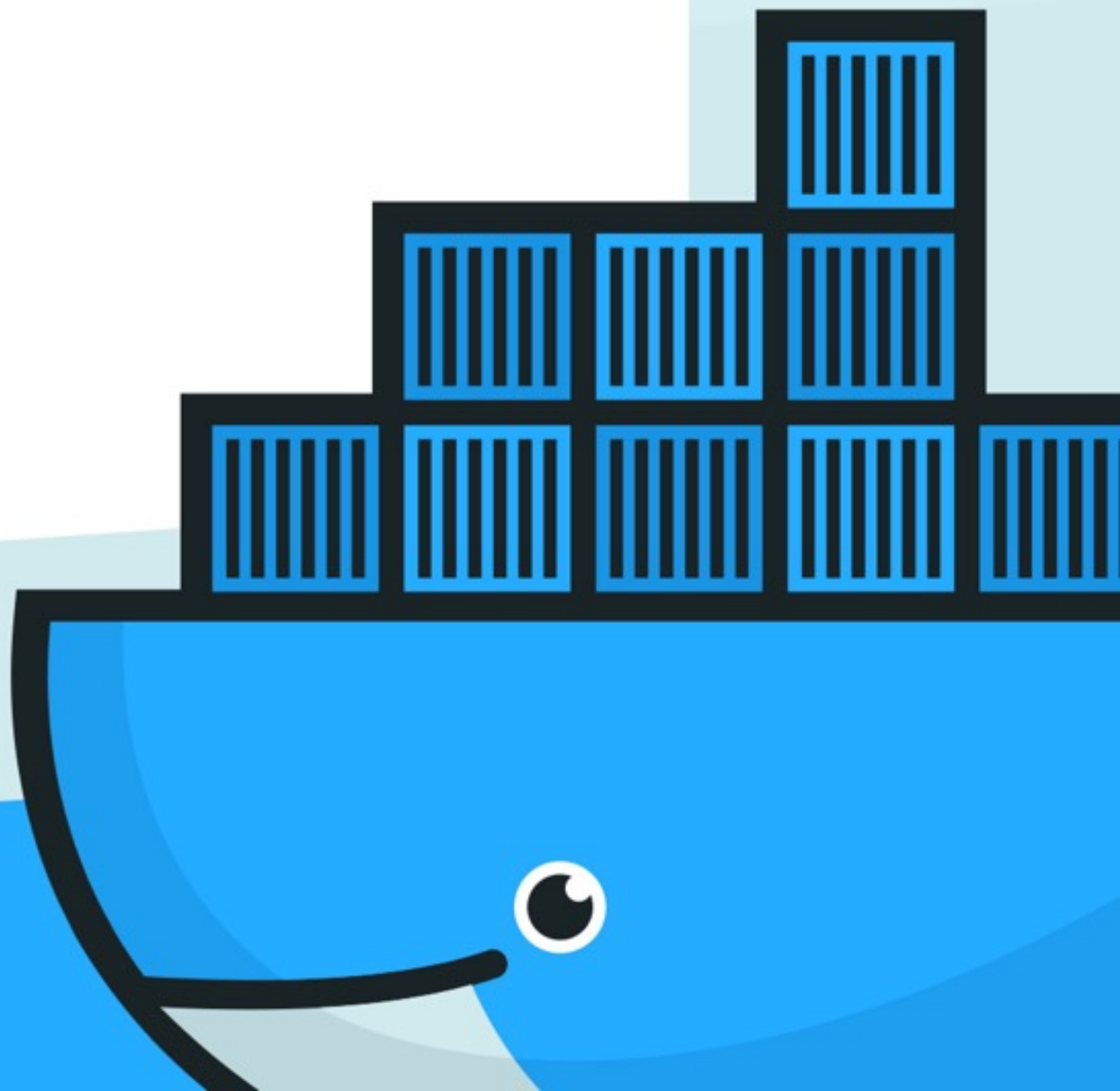
Putting the problems in context

If we dockerize all of our applications how many **Dockerfiles** is that?

If we build a complex hierarchy of **Dockerfiles**, how quickly can we trace/rebuild a specific image?

As best-practices develops how can we refactor our **Dockefiles** with confidence?

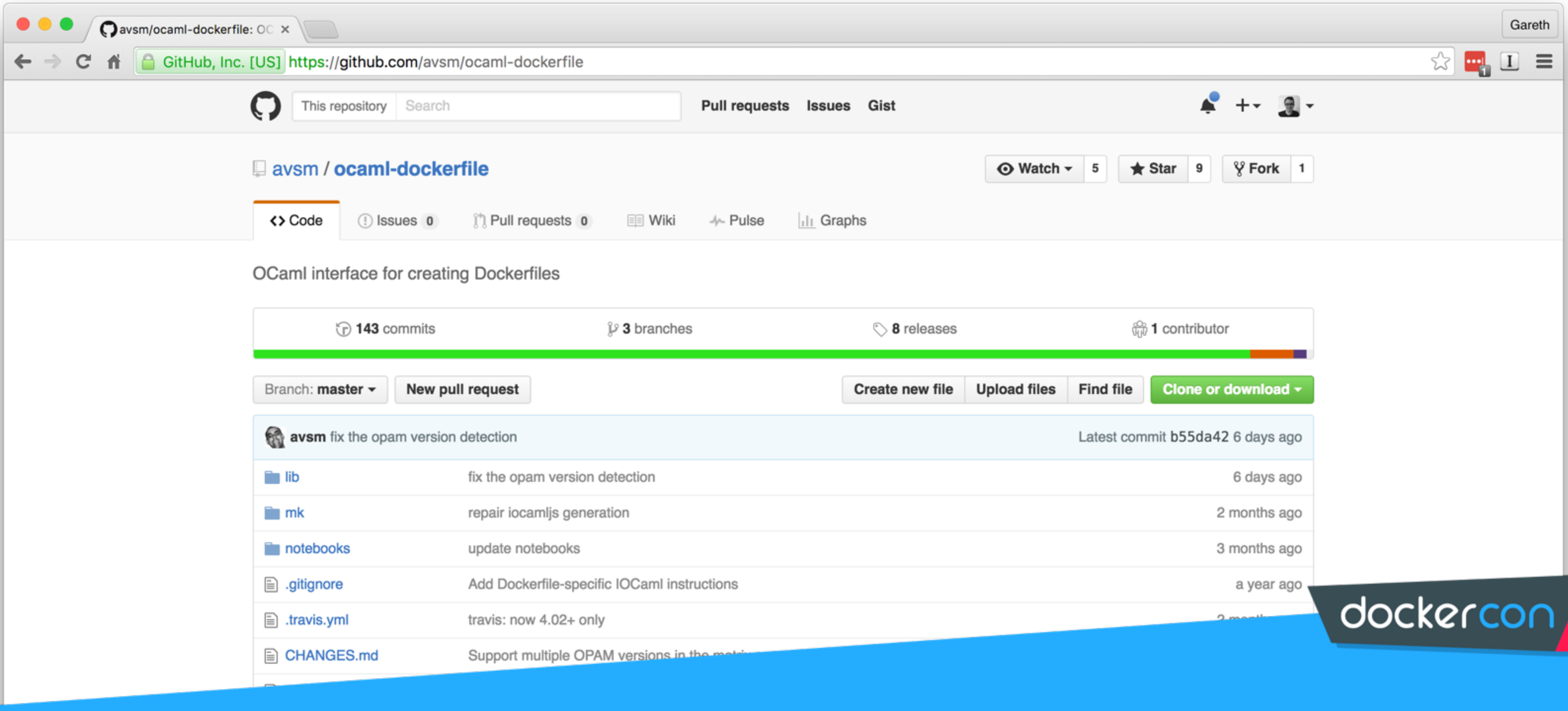
Are **Dockerfiles** best managed centrally or on a team-by-team basis?



Some community ideas

Generate Dockerfiles

Build Dockerfiles with OCAML



avsm/ocaml-dockerfile

https://github.com/avsm/ocaml-dockerfile

OCaml interface for creating Dockerfiles

143 commits · 3 branches · 8 releases · 1 contributor

avsm fix the opam version detection — Latest commit b55da42 6 days ago

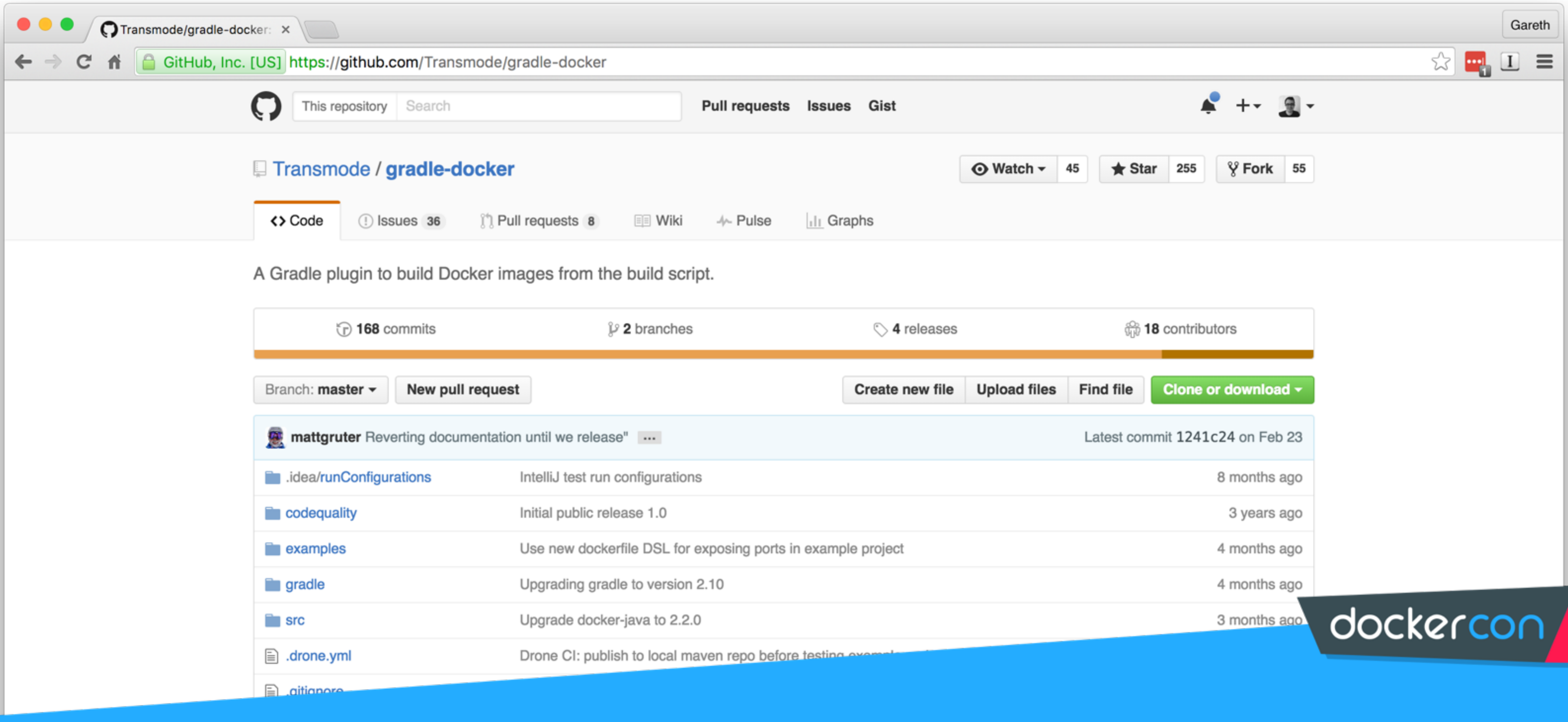
lib	fix the opam version detection	6 days ago
mk	repair iocamljs generation	2 months ago
notebooks	update notebooks	3 months ago
.gitignore	Add Dockerfile-specific IOCaml instructions	a year ago
.travis.yml	travis: now 4.02+ only	
CHANGES.md	Support multiple OPAM versions in the	

dockercon 16

OCAML example

```
let base =
  let email = "anil@recoil.org" in
  comment "Generated by OCaml Dockerfile" @@
  from "ubuntu" ~tag:"trusty" @@
  maintainer "Anil Madhavapeddy <%s>" email

let ocaml_ubuntu_image =
  base @@
  run "apt-get -y -qq update" @@
  run "apt-get -y install ocaml ocaml-native-compilers camlp4-ex
  onbuild (run "apt-get -y -qq update")
```

With Gradle


https://github.com/Transmode/gradle-docker

Transmode / gradle-docker

A Gradle plugin to build Docker images from the build script.

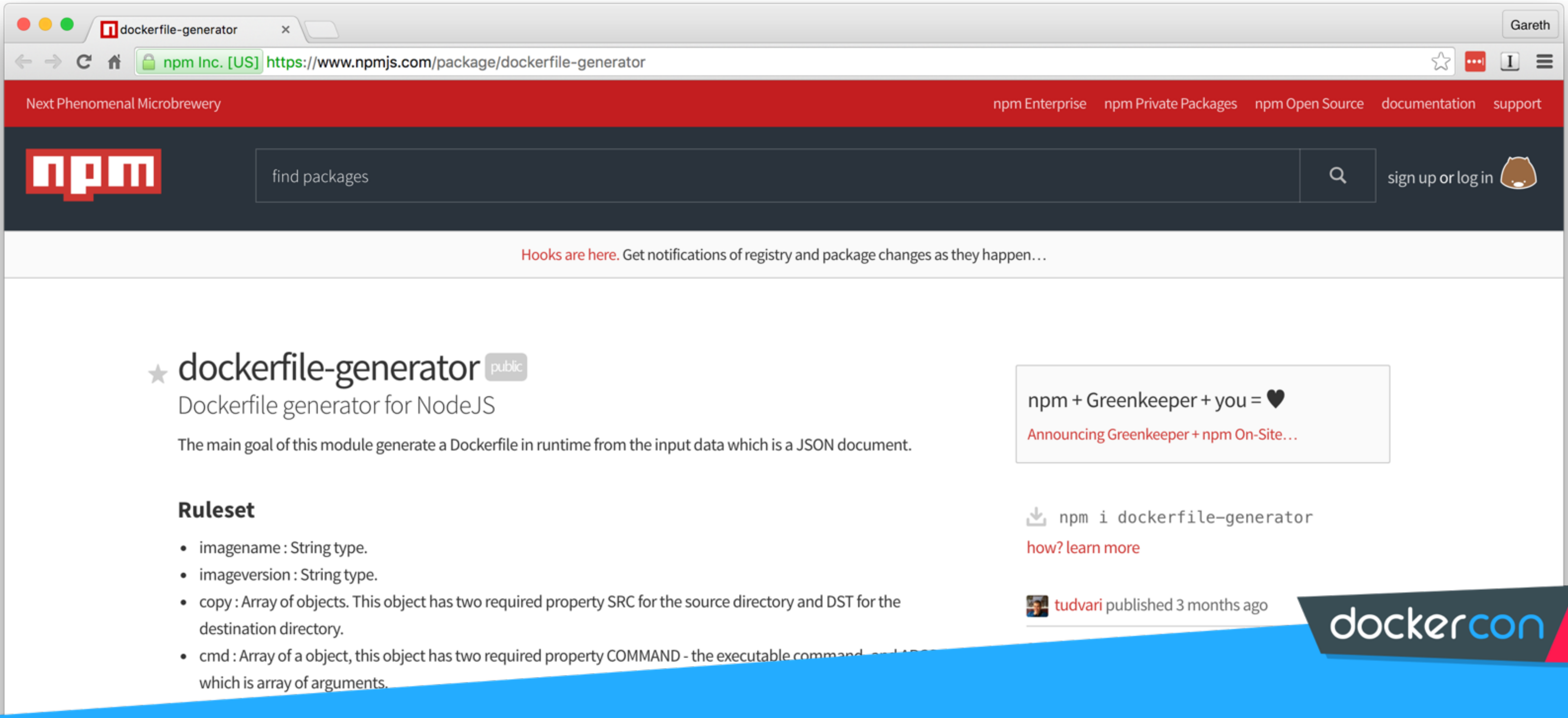
168 commits · 2 branches · 4 releases · 18 contributors

mattgruter Reverting documentation until we release" … Latest commit 1241c24 on Feb 23

.idea/runConfigurations	IntelliJ test run configurations	8 months ago
codequality	Initial public release 1.0	3 years ago
examples	Use new dockerfile DSL for exposing ports in example project	4 months ago
gradle	Upgrading gradle to version 2.10	4 months ago
src	Upgrade docker-java to 2.2.0	3 months ago
.drone.yml	Drone CI: publish to local maven repo before testing	
.gitignore		

dockercon 16

Or Javascript


dockerfile-generator

npm Inc. [US] https://www.npmjs.com/package/dockerfile-generator

Next Phenomenal Microbrewery

npm Enterprise npm Private Packages npm Open Source documentation support

find packages

sign up or log in

Hooks are here. Get notifications of registry and package changes as they happen…

dockerfile-generator public

Dockerfile generator for NodeJS

The main goal of this module generate a Dockerfile in runtime from the input data which is a JSON document.

Ruleset

- imagename : String type.
- imageversion : String type.
- copy : Array of objects. This object has two required property SRC for the source directory and DST for the destination directory.
- cmd : Array of a object, this object has two required property COMMAND - the executable command
  which is array of arguments.

npm + Greenkeeper + you = ♥

Announcing Greenkeeper + npm On-Site…

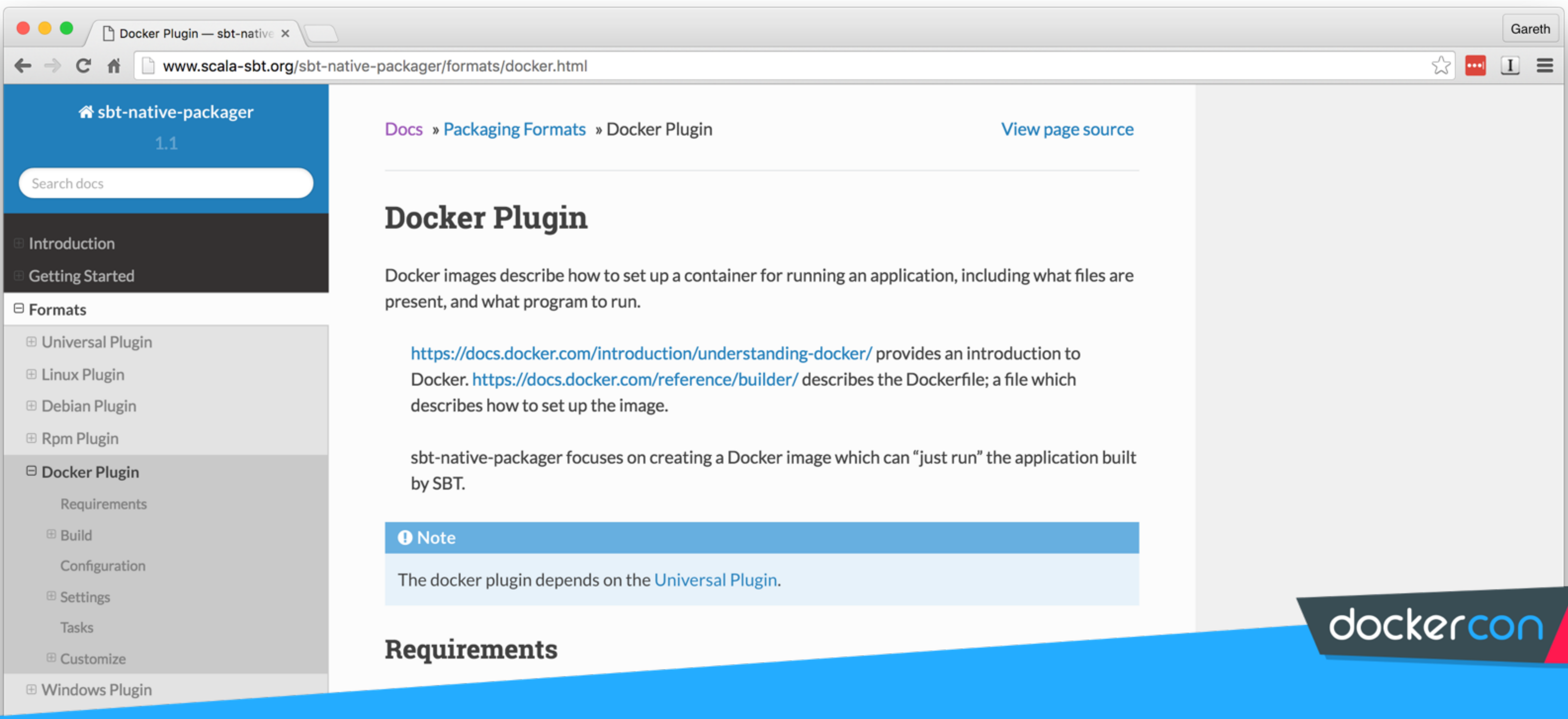
```
npm i dockerfile-generator
```

how? learn more

tudvari published 3 months ago

dockercon 16

Or Scala and SBT



Docs » Packaging Formats » Docker Plugin View page source

Docker Plugin

Docker images describe how to set up a container for running an application, including what files are present, and what program to run.

https://docs.docker.com/introduction/understanding-docker/ provides an introduction to Docker. https://docs.docker.com/reference/builder/ describes the Dockerfile; a file which describes how to set up the image.

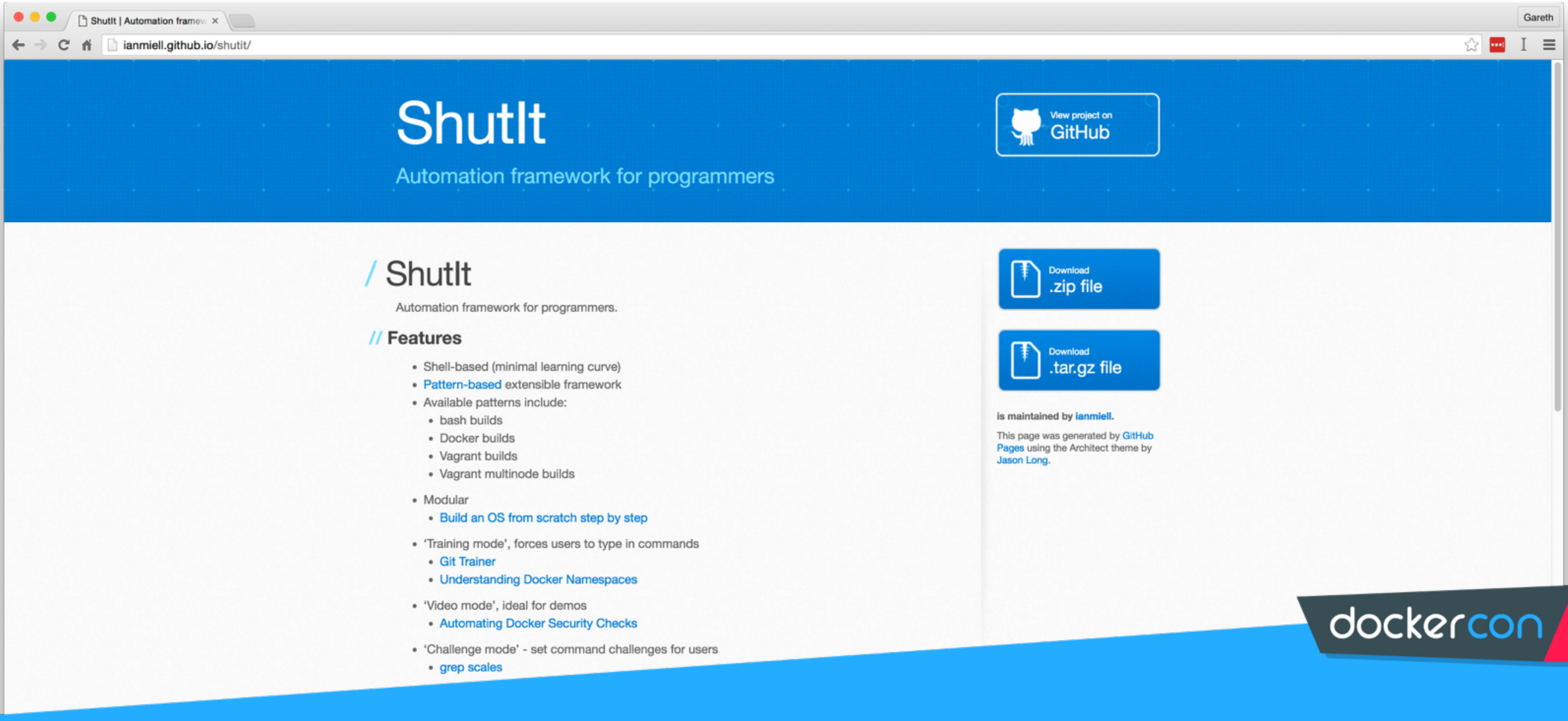
sbt-native-packager focuses on creating a Docker image which can “just run” the application built by SBT.

Note

The docker plugin depends on the Universal Plugin.

Requirements

Or with Python


ShutIt | Automation framew ×

Gareth

ianmiell.github.io/shutit/

ShutIt

Automation framework for programmers

/ ShutIt

Automation framework for programmers.

// Features

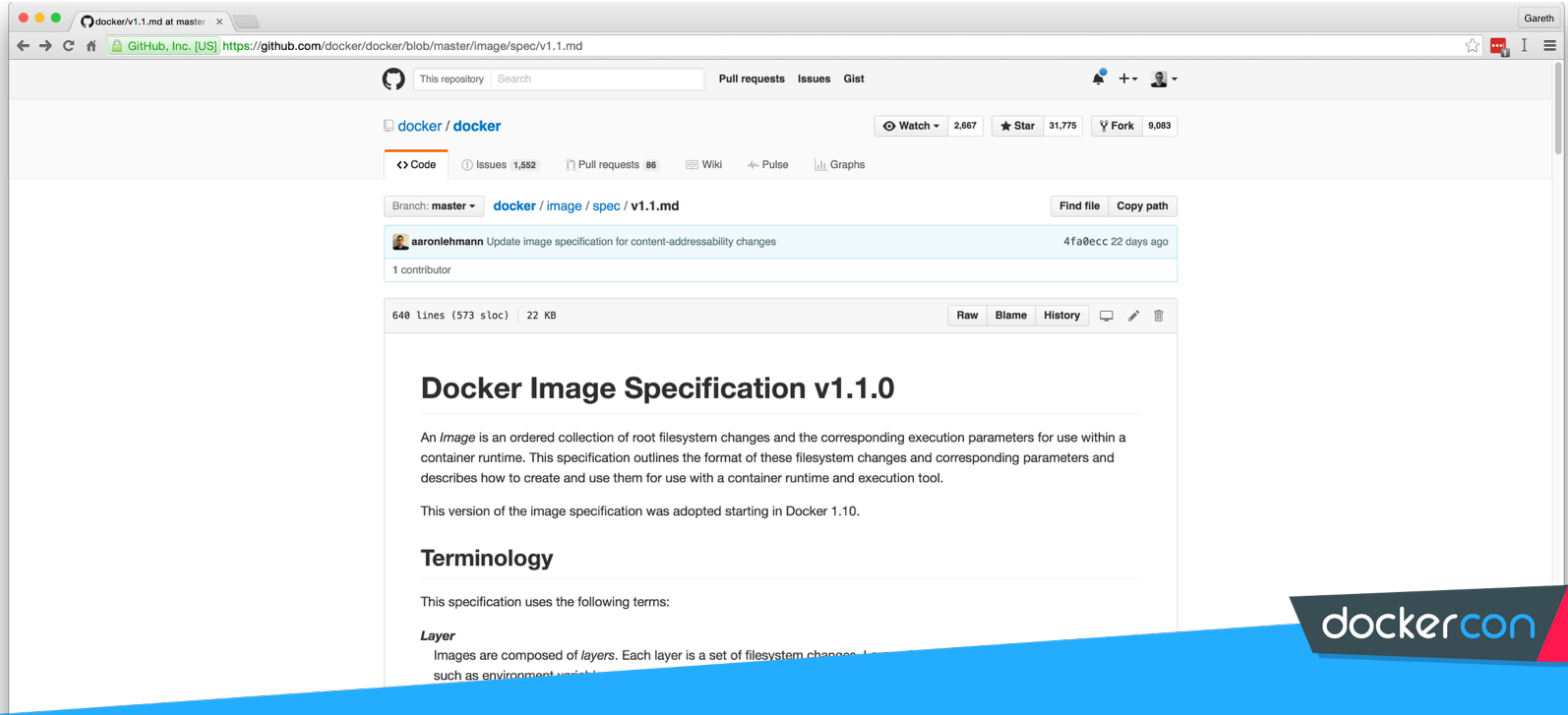
- Shell-based (minimal learning curve)
- Pattern-based extensible framework
- Available patterns include:
  - bash builds
  - Docker builds
  - Vagrant builds
  - Vagrant multinode builds
- Modular
  - Build an OS from scratch step by step
- 'Training mode', forces users to type in commands
  - Git Trainer
  - Understanding Docker Namespaces
- 'Video mode', ideal for demos
  - Automating Docker Security Checks
- 'Challenge mode' - set command challenges for users
  - grep scales

is maintained by ianmiell.

This page was generated by GitHub Pages using the Architect theme by Jason Long.

No Dockerfile to be seen

Docker Image Specification



docker/v1.1.md at master
GitHub, Inc. [US] https://github.com/docker/docker/blob/master/image/spec/v1.1.md
This repository Search
Pull requests Issues Gist
docker / docker
Watch 2,667 Star 31,775 Fork 9,083
Code Issues 1,552 Pull requests 86 Wiki Pulse Graphs
Branch: master docker / image / spec / v1.1.md
Find file Copy path
aaronlehmann Update image specification for content-addressability changes 4fa0ecc 22 days ago
1 contributor
640 lines (573 sloc) 22 KB
Raw Blame History

Docker Image Specification v1.1.0

An *Image* is an ordered collection of root filesystem changes and the corresponding execution parameters for use within a container runtime. This specification outlines the format of these filesystem changes and corresponding parameters and describes how to create and use them for use with a container runtime and execution tool.

This version of the image specification was adopted starting in Docker 1.10.

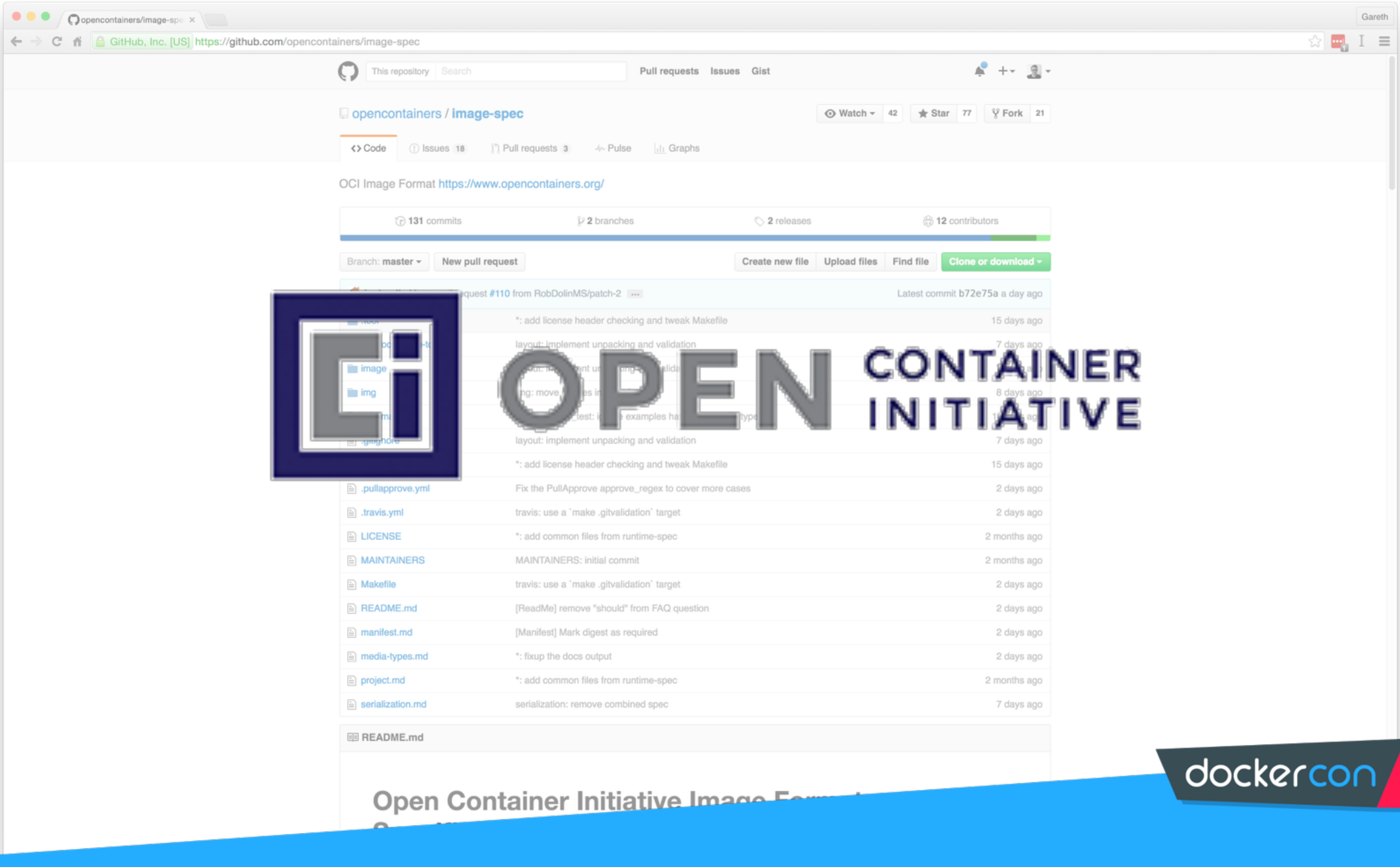
Terminology

This specification uses the following terms:

Layer

Images are composed of *layers*. Each layer is a set of filesystem changes

such as environment



opencontainers/image-spec

GitHub, Inc. [US] https://github.com/opencontainers/image-spec

This repository Search Pull requests Issues Gist

opencontainers / image-spec

Watch 42 Star 77 Fork 21

Code Issues 18 Pull requests 3 Pulse Graphs

OCI Image Format https://www.opencontainers.org/

131 commits 2 branches 2 releases 12 contributors

Branch: master New pull request Create new file Upload files Find file Clone or download

quest #110 from RobDolinMS/patch-2 … Latest commit b72e75a a day ago

	*: add license header checking and tweak Makefile	15 days ago
	layout: implement unpacking and validation	7 days ago
image		
img	img: move …	8 days ago
	…_test: … examples ha…	
.gitignore	layout: implement unpacking and validation	7 days ago
	*: add license header checking and tweak Makefile	15 days ago
.pullapprove.yml	Fix the PullApprove approve_regex to cover more cases	2 days ago
.travis.yml	travis: use a `make .gitvalidation` target	2 days ago
LICENSE	*: add common files from runtime-spec	2 months ago
MAINTAINERS	MAINTAINERS: initial commit	2 months ago
Makefile	travis: use a `make .gitvalidation` target	2 days ago
README.md	[ReadMe] remove "should" from FAQ question	2 days ago
manifest.md	[Manifest] Mark digest as required	2 days ago
media-types.md	*: fixup the docs output	2 days ago
project.md	*: add common files from runtime-spec	2 months ago
serialization.md	serialization: remove combined spec	7 days ago

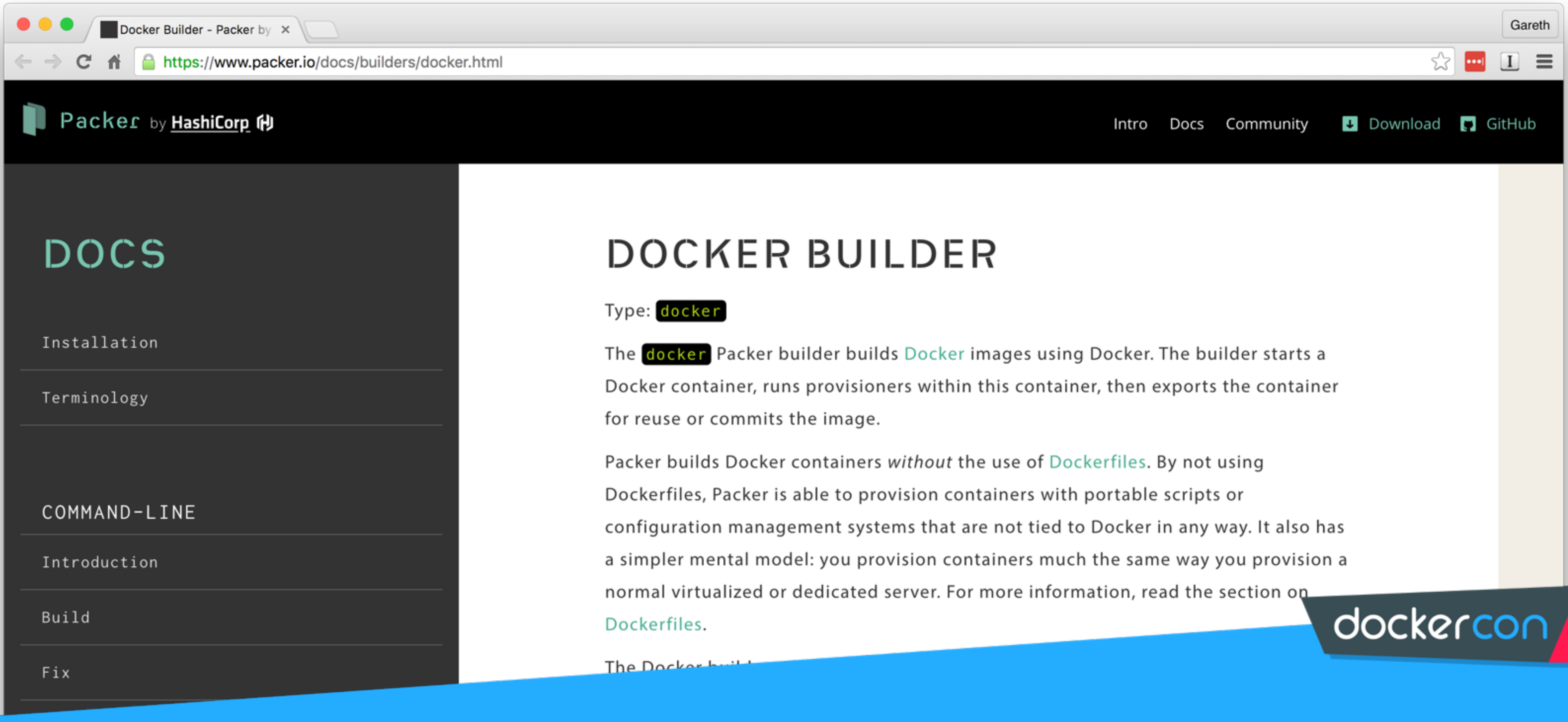
README.md

Open Container Initiative Image Format

OPEN CONTAINER INITIATIVE

dockercon 16

Packer



DOCKER BUILDER

Type: `docker`

The `docker` Packer builder builds Docker images using Docker. The builder starts a Docker container, runs provisioners within this container, then exports the container for reuse or commits the image.

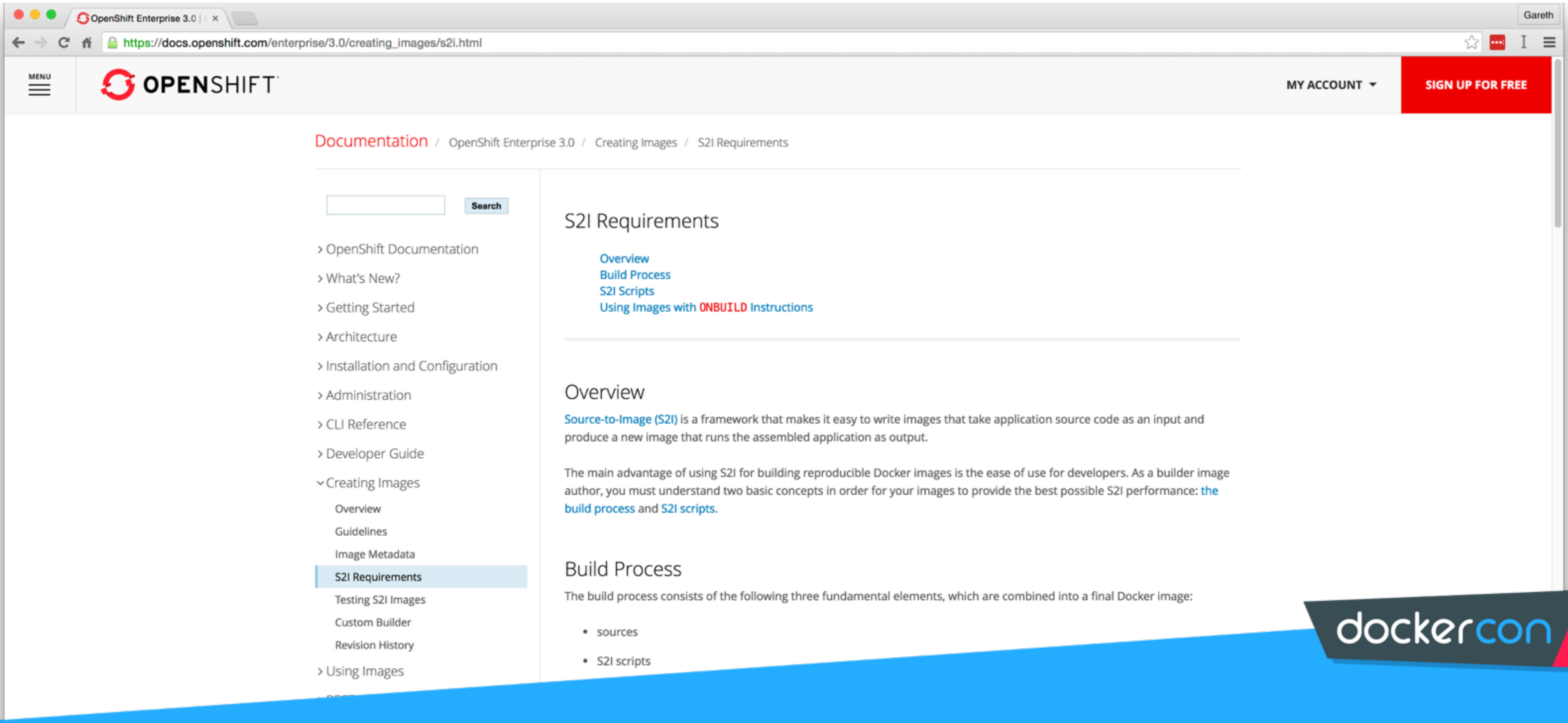
Packer builds Docker containers *without* the use of Dockerfiles. By not using Dockerfiles, Packer is able to provision containers with portable scripts or configuration management systems that are not tied to Docker in any way. It also has a simpler mental model: you provision containers much the same way you provision a normal virtualized or dedicated server. For more information, read the section on Dockerfiles.

The Docker buil

Packer example

```
{
  "builders":[{
   "type": "docker",
   "image": "ubuntu",
   "export_path": "image.tar"
  }],
  "provisioners":[
    {
      "type": "shell",
      "inline": ["apt-get -y update; apt-get install
    },
```

Source-to-Image



OpenShift Enterprise 3.0 |
https://docs.openshift.com/enterprise/3.0/creating_images/s2i.html
Gareth
MENU
OPENSHIFT
MY ACCOUNT
SIGN UP FOR FREE
Search

Documentation / OpenShift Enterprise 3.0 / Creating Images / S2I Requirements

S2I Requirements

Overview

Source-to-Image (S2I) is a framework that makes it easy to write images that take application source code as an input and produce a new image that runs the assembled application as output.

The main advantage of using S2I for building reproducible Docker images is the ease of use for developers. As a builder image author, you must understand two basic concepts in order for your images to provide the best possible S2I performance: the build process and S2I scripts.

Build Process

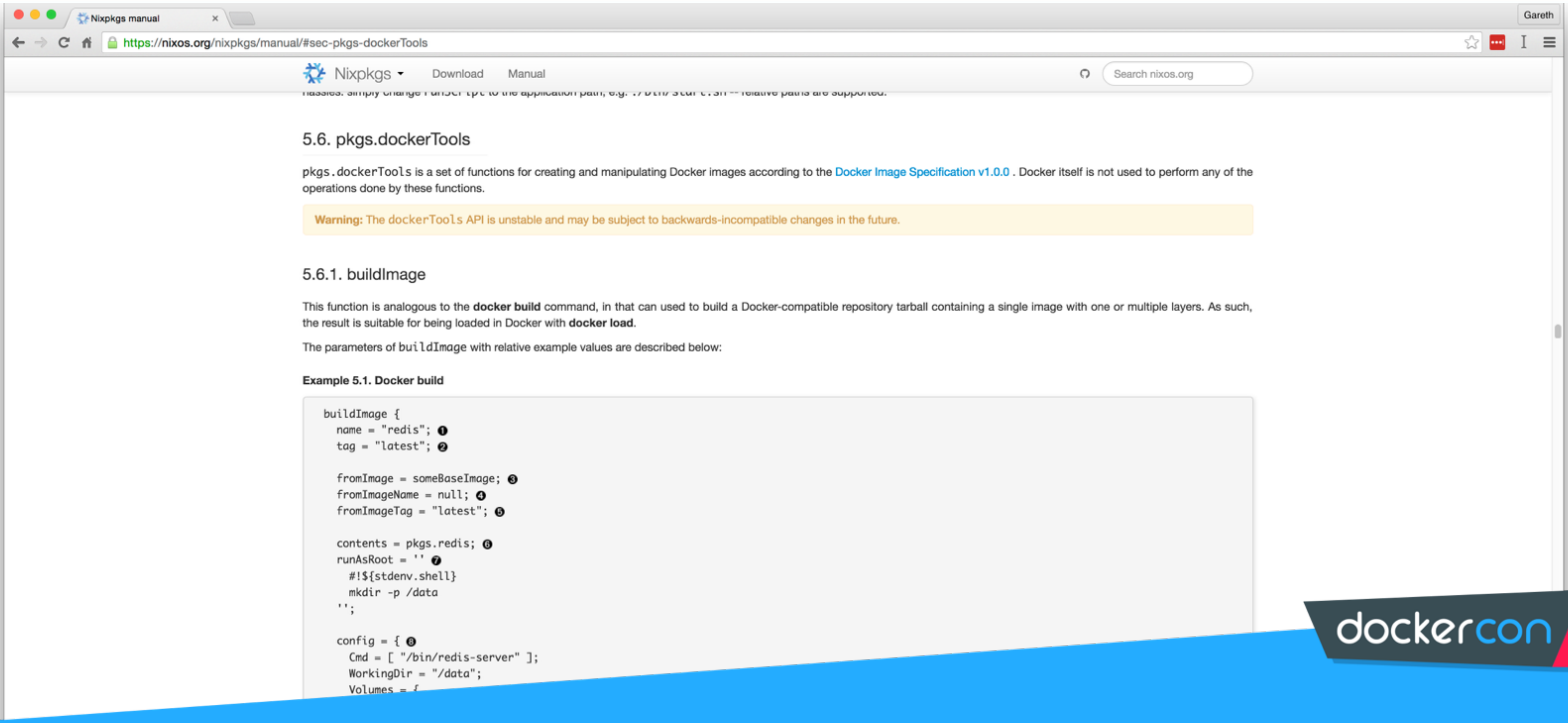
The build process consists of the following three fundamental elements, which are combined into a final Docker image:

- sources
- S2I scripts

s2i example

```
$ s2i create <image name> <destination directory>
$ s2i build <source location> <builder image> [<tag>] [flags]
$ s2i rebuild <image name> [<new-tag-name>]
$ s2i usage <builder image> [flags]

$ s2i build ./sinatra-app openshift/ruby-20-centos7 ruby-app
```

Nix



hassles. Simply change runScript to the application path, e.g. ./bin/start.sh -- relative paths are supported.

5.6. pkgs.dockerTools

`pkgs.dockerTools` is a set of functions for creating and manipulating Docker images according to the Docker Image Specification v1.0.0 . Docker itself is not used to perform any of the operations done by these functions.

Warning: The `dockerTools` API is unstable and may be subject to backwards-incompatible changes in the future.

5.6.1. buildImage

This function is analogous to the **docker build** command, in that can used to build a Docker-compatible repository tarball containing a single image with one or multiple layers. As such, the result is suitable for being loaded in Docker with **docker load**.

The parameters of `buildImage` with relative example values are described below:

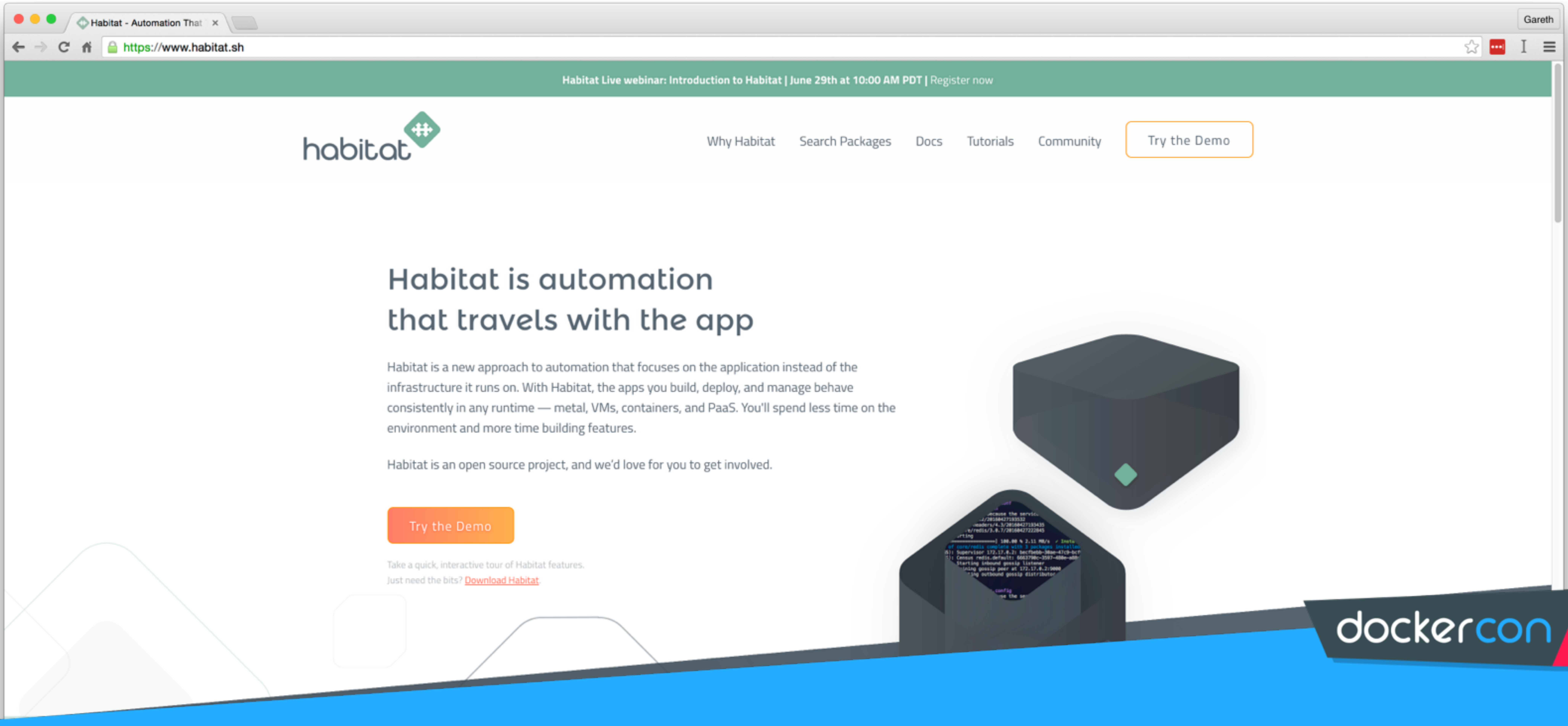
Example 5.1. Docker build

```
buildImage {
  name = "redis"; ❶
  tag = "latest"; ❷

  fromImage = someBaseImage; ❸
  fromImageName = null; ❹
  fromImageTag = "latest"; ❺

  contents = pkgs.redis; ❻
  runAsRoot = '' ❼
    #!${stdenv.shell}
    mkdir -p /data
  '';

  config = { ❽
    Cmd = [ "/bin/redis-server" ];
    WorkingDir = "/data";
    Volumes = {
```

Nix example

```
dockerTools.buildImage {
  name = "redis";
  runAsRoot = ''
    #!${stdenv.shell}
    ${dockerTools.shadowSetup}
    groupadd -r redis
    useradd -r -g redis -d /data -M redis
    mkdir /data
    chown redis:redis /data
  '';
```

Habitat



Habitat - Automation That

https://www.habitat.sh

Habitat Live webinar: Introduction to Habitat | June 29th at 10:00 AM PDT | Register now

Habitat is automation that travels with the app

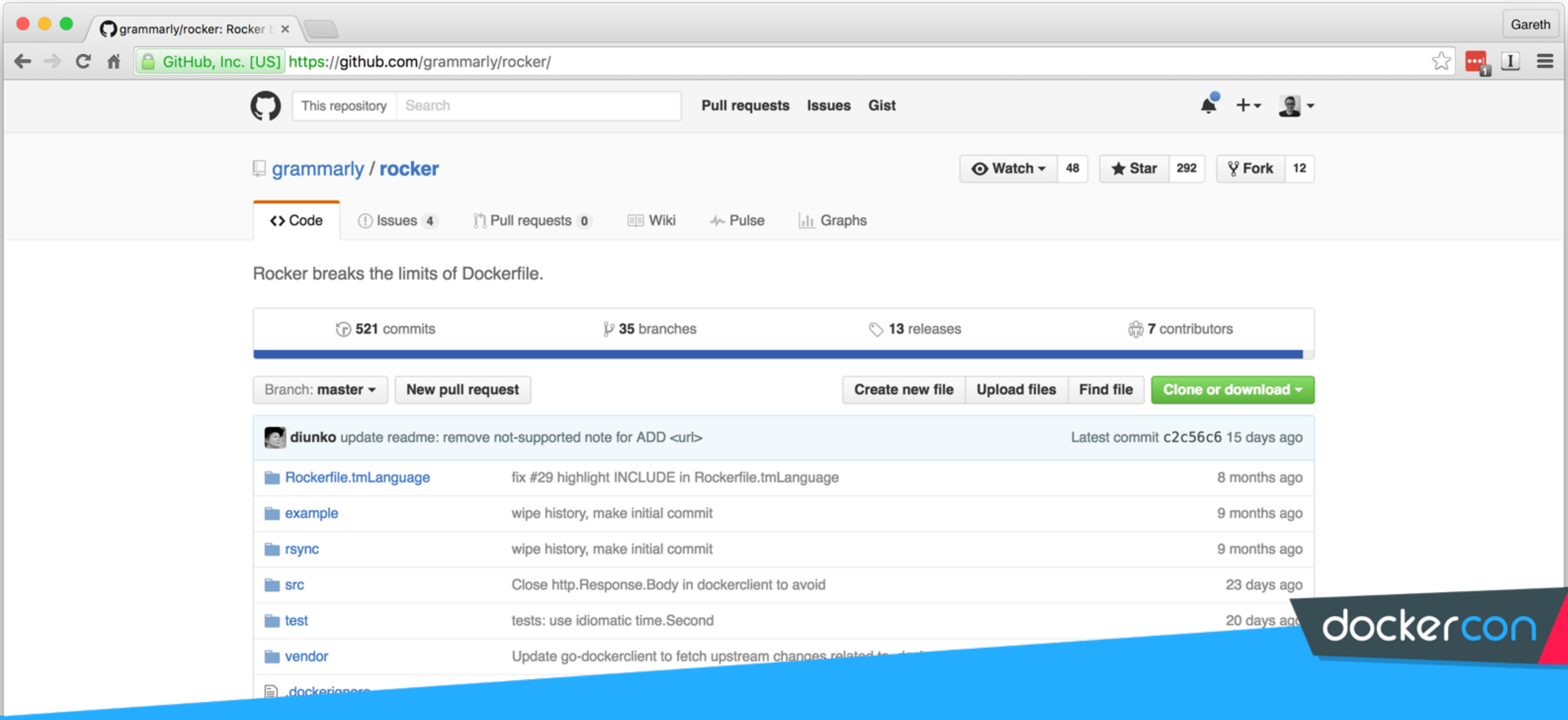
Habitat is a new approach to automation that focuses on the application instead of the infrastructure it runs on. With Habitat, the apps you build, deploy, and manage behave consistently in any runtime — metal, VMs, containers, and PaaS. You'll spend less time on the environment and more time building features.

Habitat is an open source project, and we'd love for you to get involved.

Take a quick, interactive tour of Habitat features.
Just need the bits? Download Habitat.

Expand on Dockerfile

Rocker



grammarly/rocker: Rocker

https://github.com/grammarly/rocker/

grammarly / rocker

Watch 48 · Star 292 · Fork 12

Code · Issues 4 · Pull requests 0 · Wiki · Pulse · Graphs

Rocker breaks the limits of Dockerfile.

521 commits · 35 branches · 13 releases · 7 contributors

Branch: master · New pull request · Create new file · Upload files · Find file · Clone or download

diunko update readme: remove not-supported note for ADD <url> Latest commit c2c56c6 15 days ago

Rockerfile.tmLanguage	fix #29 highlight INCLUDE in Rockerfile.tmLanguage	8 months ago
example	wipe history, make initial commit	9 months ago
rsync	wipe history, make initial commit	9 months ago
src	Close http.Response.Body in dockerclient to avoid	23 days ago
test	tests: use idiomatic time.Second	20 days ago
vendor	Update go-dockerclient to fetch upstream changes related to	
.dockerignore		

dockercon 16

Rocker adds some crucial features that are missing from Dockerfile while keeping Docker's original design

Rockerfile example

```
FROM ubuntu:16.04
MAINTAINER Gareth Rushgrove "gareth@puppet.com"

ENV PUPPET_AGENT_VERSION="1.5.0" UBUNTU_CODENAME="xenial" PATH=/

LABEL com.puppet.version="0.1.0" com.puppet.dockerfile="/Dockerf

MOUNT /opt/puppetlabs /etc/puppetlabs /root/.gem

RUN apt-get update && \
    apt-get install
```

dockercon 16

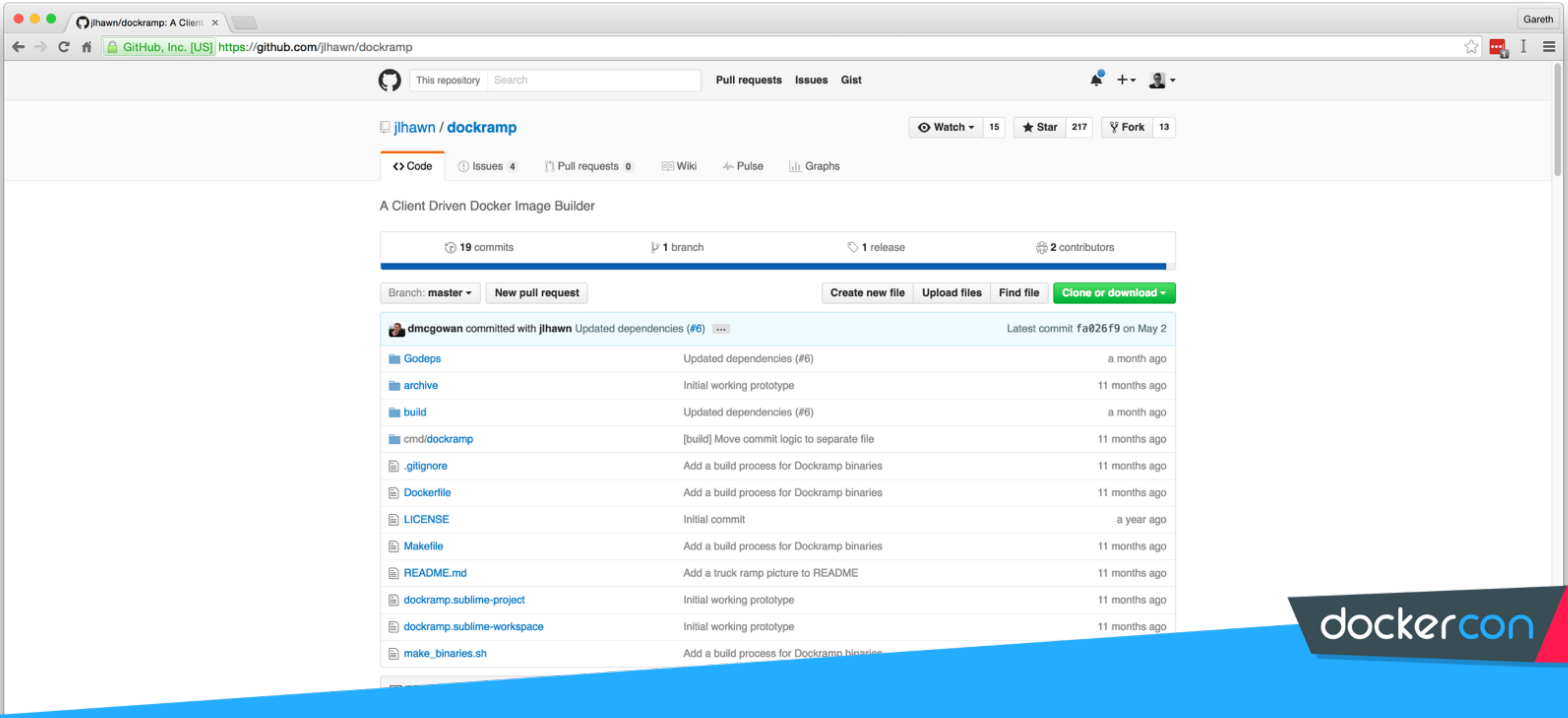
Includes new instructions

```
FROM ubuntu:16.04
MAINTAINER Gareth Rushgrove "gareth@puppet.com"

ENV PUPPET_AGENT_VERSION="1.5.0" UBUNTU_CODENAME="xenial" PATH=/

LABEL com.puppet.version="0.1.0" com.puppet.dockerfile="/Dockerf

MOUNT /opt/puppetlabs /etc/puppetlabs /root/.gem

RUN apt-get update && \
    apt-get install
```

More new instructions

```
    rm -rf /var/lib/apt/lists/*

EXPOSE 80

CMD ["nginx"]

COPY Rockerfile /Dockerfile

TAG puppet/puppet-rocker-example
```

Dockramp



jlhawn / dockramp

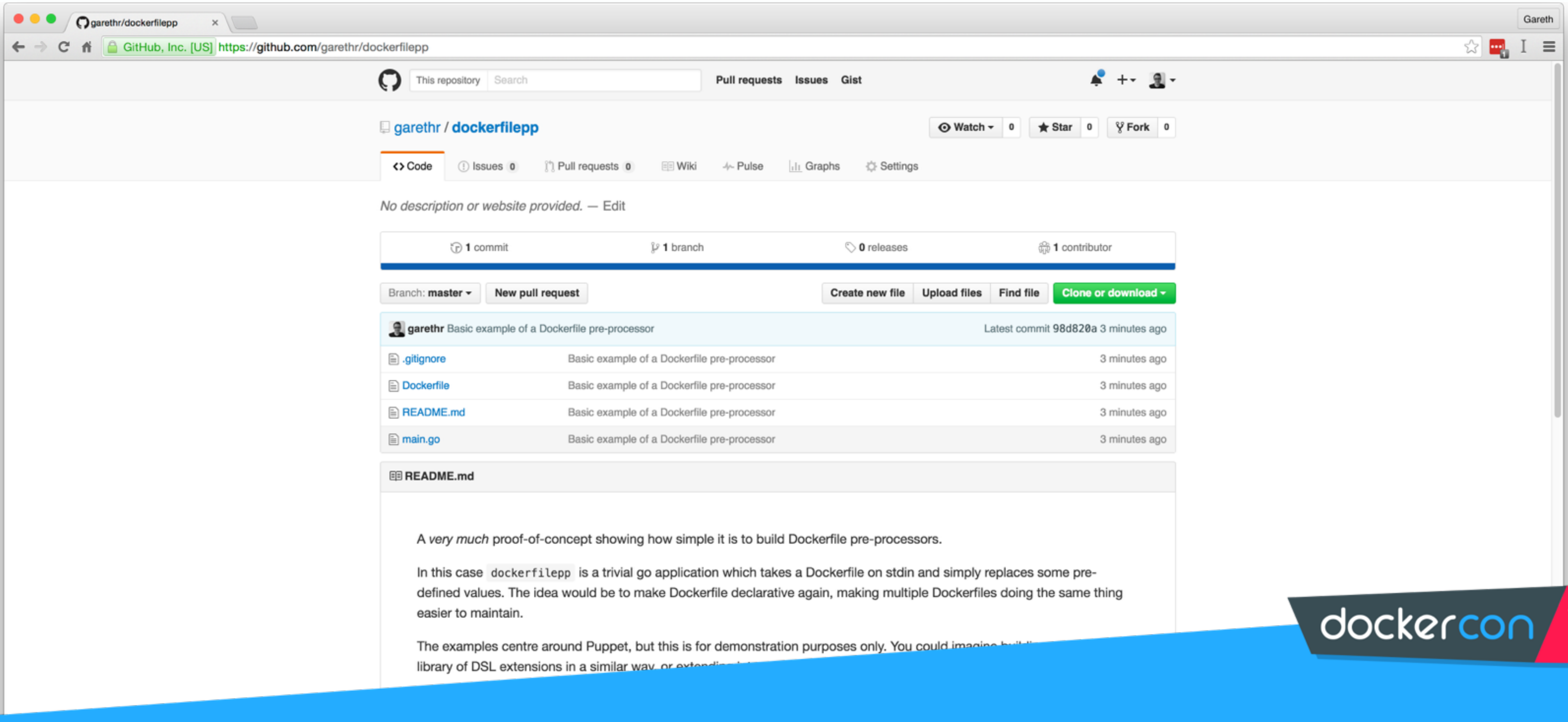
A Client Driven Docker Image Builder

19 commits · 1 branch · 1 release · 2 contributors

dmcgowan committed with jlhawn Updated dependencies (#6) … Latest commit fa026f9 on May 2

Godeps	Updated dependencies (#6)	a month ago
archive	Initial working prototype	11 months ago
build	Updated dependencies (#6)	a month ago
cmd/dockramp	[build] Move commit logic to separate file	11 months ago
.gitignore	Add a build process for Dockramp binaries	11 months ago
Dockerfile	Add a build process for Dockramp binaries	11 months ago
LICENSE	Initial commit	a year ago
Makefile	Add a build process for Dockramp binaries	11 months ago
README.md	Add a truck ramp picture to README	11 months ago
dockramp.sublime-project	Initial working prototype	11 months ago
dockramp.sublime-workspace	Initial working prototype	11 months ago
make_binaries.sh	Add a build process for Dockramp binaries	

Dockerfile pre-processors



garethr / dockerfilepp

No description or website provided. — Edit

1 commit | 1 branch | 0 releases | 1 contributor

garethr Basic example of a Dockerfile pre-processor Latest commit 98d820a 3 minutes ago

.gitignore	Basic example of a Dockerfile pre-processor	3 minutes ago
Dockerfile	Basic example of a Dockerfile pre-processor	3 minutes ago
README.md	Basic example of a Dockerfile pre-processor	3 minutes ago
main.go	Basic example of a Dockerfile pre-processor	3 minutes ago

README.md

A *very much* proof-of-concept showing how simple it is to build Dockerfile pre-processors.

In this case `dockerfilepp` is a trivial go application which takes a Dockerfile on stdin and simply replaces some pre-defined values. The idea would be to make Dockerfile declarative again, making multiple Dockerfiles doing the same thing easier to maintain.

The examples centre around Puppet, but this is for demonstration purposes only. You could imagine buildi
library of DSL extensions in a similar way, or extendi

Domain-specific extensions

```
FROM ubuntu:16.04
MAINTAINER Gareth Rushgrove "gareth@puppet.com"

ENV PUPPET_AGENT_VERSION="1.5.0" R10K_VERSION="2.2.2" \ UBUNTU_C

PUPPET_INSTALL
PUPPET_COPY_PUPPETFILE
PUPPET_COPY_MANIFESTS
PUPPET_RUN

EXPOSE 80
```

dockercon 16

Simple expansion

```
$ cat Dockerfile | dockerfilepp
FROM ubuntu:16.04
MAINTAINER Gareth Rushgrove "gareth@puppet.com"

ENV PUPPET_AGENT_VERSION="1.5.0" R10K_VERSION="2.2.2" UBUNTU_COD

RUN apt-get update && \
    apt-get install -y wget=1.17.1-1ubuntu1 && \
    wget https://apt.puppetlabs.com/puppetlabs-release-pc1-"$UBU
    dpkg -i puppetlabs-release-pc1-"$UBUNTU_CODEN
    rm puppetlabs-rele
```

The future

Speculation and things I’d like to see

Formal specification for Dockerfile

RUN, FROM, COPY, etc.
as first class API primitives

Opinionated workflow tooling around image build

Shared libraries and support for pre-processors

Complementary tools that take an organizational view of image building

Conclusions

If all you take away is…

Dockerfile is a great starting point for many use cases

But we will need better tools for managing many Dockerfiles

And Dockerfile is just one interface to building images

We’ll need different types of tools for different use cases

Questions?

And thanks for listening

Thank you!