

Continuous Integration with Docker, Buildbot and Git

About Me

- Python user since 2007
- Build web based systems using Python + Django mostly
- Gain interests in infrastructure gradually
- Start to build my own software company recently




twitter: [@adieu](https://twitter.com/adieu)
github: github.com/adieu
website: www.adieu.me

Quiz

- How many of you know Docker
- How many of you know Buildbot
- How many of you know Git
- How many of you deploy software to servers
- How many of you have any kinds of experience with virtualization


Continuous Integration **with Docker, Buildbot and Git**


What is it



WIKIPEDIA
The Free Encyclopedia

- Main page
- Contents
- Featured content
- Current events
- Random article
- Donate to Wikipedia

- Interaction
 - Help
 - About Wikipedia
 - Community portal
 - Recent changes
 - Contact page
- Tools
- Print/export
- Languages 
 - العربية
 - Cestina
 - Deutsch
 - Español
 - Français
 - 한국어
 - Italiano
 - עברית
 - 日本語
 - Polski
 - Română
 - Русский
 - Українська

 Edit links

ArticleTalk

ReadEditView history

Search

Continuous integration

From Wikipedia, the free encyclopedia

Continuous integration (CI) is the practice, in [software engineering](#), of merging all developer working copies with a shared [mainline](#) several times a day. It was first named and proposed as part of [extreme programming](#) (XP). Its main aim is to prevent integration problems, referred to as "integration hell" in early descriptions of XP. CI can be seen as an intensification of practices of periodic integration advocated by earlier published methods of incremental and iterative software development, such as the [Booch method](#). CI isn't universally accepted as an improvement over frequent integration, so it is important to distinguish between the two as there is disagreement about the virtues of each.

CI was originally intended to be used in combination with automated unit tests written through the practices of [test-driven development](#). Initially this was conceived of as running all unit tests and verifying they all passed before committing to the mainline. This helps avoid one developer's work in progress breaking another developer's copy. If necessary, partially complete features can be disabled before committing using [feature toggles](#).

Later elaborations of the concept introduced build servers, which automatically run the unit tests periodically or even after every commit and report the results to the developers. The use of build servers (not necessarily running unit tests) had already been practised by some teams outside the XP community. Nowadays, many organizations have adopted CI without adopting all of XP.

In addition to automated unit tests, organizations using CI typically use a build server to implement *continuous* processes of applying [quality control](#) in general — small pieces of effort, applied frequently. In addition to running the unit and integration tests, such processes run additional static and dynamic tests, measure and profile performance, extract and format documentation from the source code and facilitate manual QA processes. This continuous application of quality control aims to improve the [quality of software](#), and to reduce the time taken to deliver it, by replacing the traditional practice of applying quality control *after* completing all development. This is very similar to the original idea of integrating more frequently to make integration easier, only applied to QA processes.

In the same vein the practice of [continuous delivery](#) further extends CI by making sure the software checked in on the mainline is always in a state that can be deployed to users and makes the actual deployment process very rapid.

Contents [hide]

1 Theory

2 Principles

- 2.1 Maintain a code repository
- 2.2 Automate the build
- 2.3 Make the build self-testing
- 2.4 Everyone commits to the baseline every day
- 2.5 Every commit (to baseline) should be built
- 2.6 Keep the build fast
- 2.7 Test in a clone of the production environment
- 2.8 Make it easy to get the latest deliverables
- 2.9 Everyone can see the results of the latest build
- 2.10 Automate deployment

Why it matters

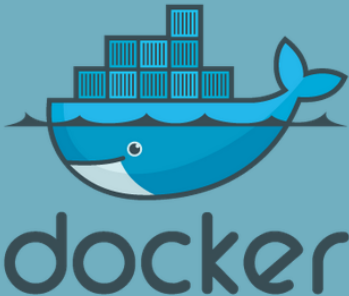
- More and more integration points in modern software system
- Successful deployment became a challenge
- Catch the bug before it hits production
- DevOps and DRY
- **Knowledge** is kept by source code instead of human brain

Continuous Integration with **Docker**, Buildbot and Git

What is it

 [Home](#) [Learn More](#) [Getting started](#) [Community](#) [Documentation](#) [Blog](#) [INDEX](#)

an open source project to pack, ship and run any application as a lightweight container



Fork us on GitHub

Docker is an open-source project to easily create lightweight, portable, self-sufficient containers from any application. The same container that a developer builds and tests on a laptop can run at scale, in production, on VMs, bare metal, OpenStack clusters, public clouds and more. [Read more ->](#)

[Learn more](#) [Get started!](#)

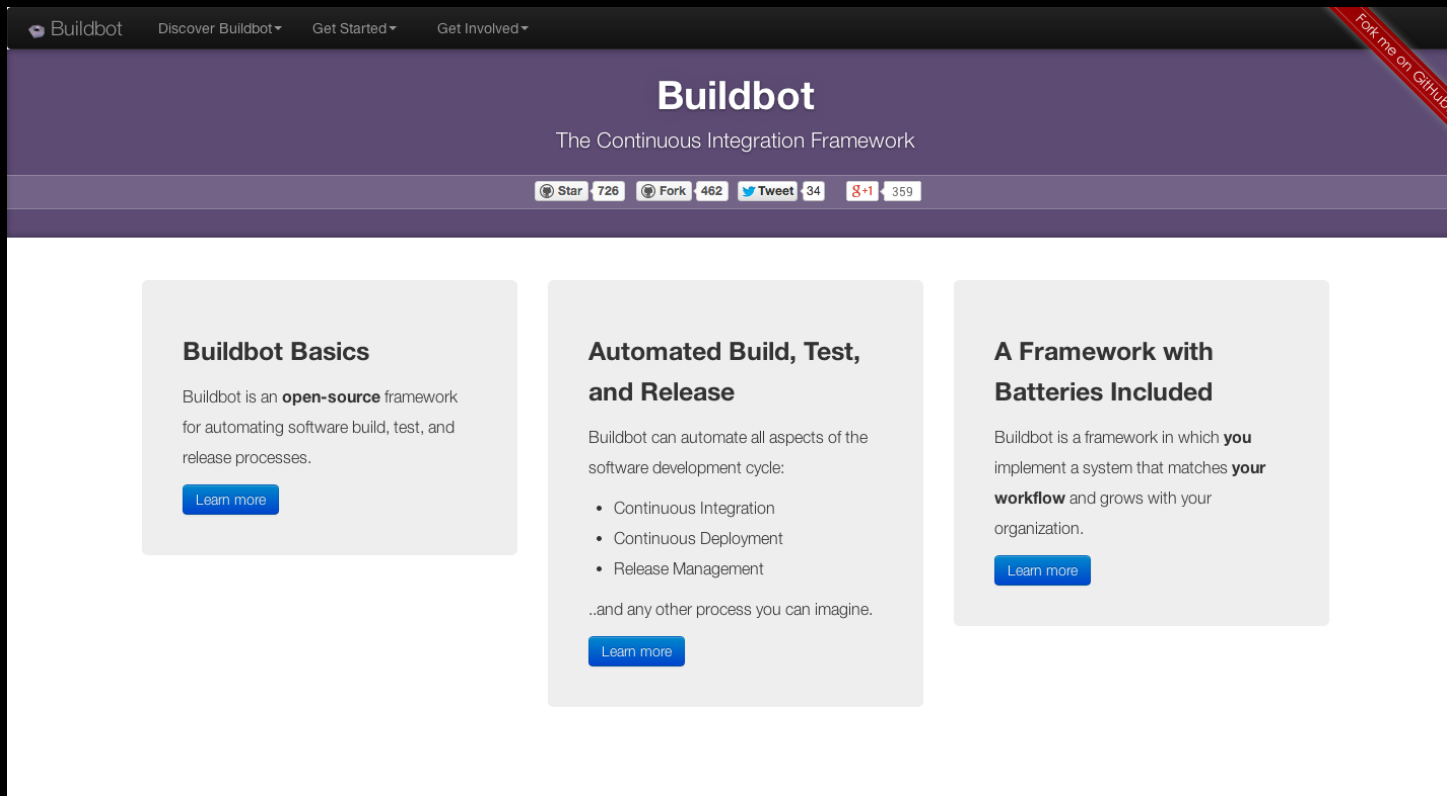
Demo

Why it matters

- Thin layer above the hardware provides a fast and unified environment
- Program runs in an isolated container with resource and network control
- Easy to use CLI and remote api
- One image could be built from a **Dockerfile** and runs on multiple machines

Continuous Integration with Docker, Buildbot and Git

What is it



The screenshot shows the Buildbot website homepage. At the top is a dark navigation bar with the Buildbot logo and links for 'Discover Buildbot', 'Get Started', and 'Get Involved'. Below this is a purple hero section with the 'Buildbot' title and subtitle 'The Continuous Integration Framework'. A social stats bar shows 726 stars, 462 forks, 34 tweets, and 359 GitHub+1s. A red diagonal banner in the top right corner says 'Fork me on GitHub'. The main content area features three light gray boxes: 'Buildbot Basics', 'Automated Build, Test, and Release', and 'A Framework with Batteries Included'. Each box contains a brief description and a 'Learn more' button. The 'Automated Build, Test, and Release' box includes a bulleted list of capabilities.

Buildbot

The Continuous Integration Framework

Star 726 Fork 462 Tweet 34 GitHub+1 359

Buildbot Basics

Buildbot is an **open-source** framework for automating software build, test, and release processes.

[Learn more](#)

Automated Build, Test, and Release

Buildbot can automate all aspects of the software development cycle:

- Continuous Integration
- Continuous Deployment
- Release Management

..and any other process you can imagine.

[Learn more](#)

A Framework with Batteries Included

Buildbot is a framework in which **you** implement a system that matches **your workflow** and grows with your organization.

[Learn more](#)

Why it matters

- Highly customizable
- Lightweight but battery included
- All configuration could be kept in the source code so that version control works
- It's **Python**!

Continuous Integration with Docker, Buildbot and Git

What is it



Git is a [free and open source](#) distributed version control system designed to handle everything from small to very large projects with speed and efficiency.

Git is [easy to learn](#) and has a [tiny footprint with lightning fast performance](#). It outclasses SCM tools like Subversion, CVS, Perforce, and ClearCase with features like [cheap local branching](#), convenient [staging areas](#), and [multiple workflows](#).



Learn Git in your browser for free with [Try Git](#).



About

The advantages of Git compared to other source control systems.



Documentation

Command reference pages, Pro Git book content, videos and other material.



Downloads

GUI clients and binary releases for all major platforms.



Community

Get involved! Mailing list, chat, development and more.



[Pro Git](#) by Scott Chacon is available to [read online for free](#). Dead tree versions are available on [Amazon.com](#).



Mac GUIs



Tarballs



Windows Build



Source Code

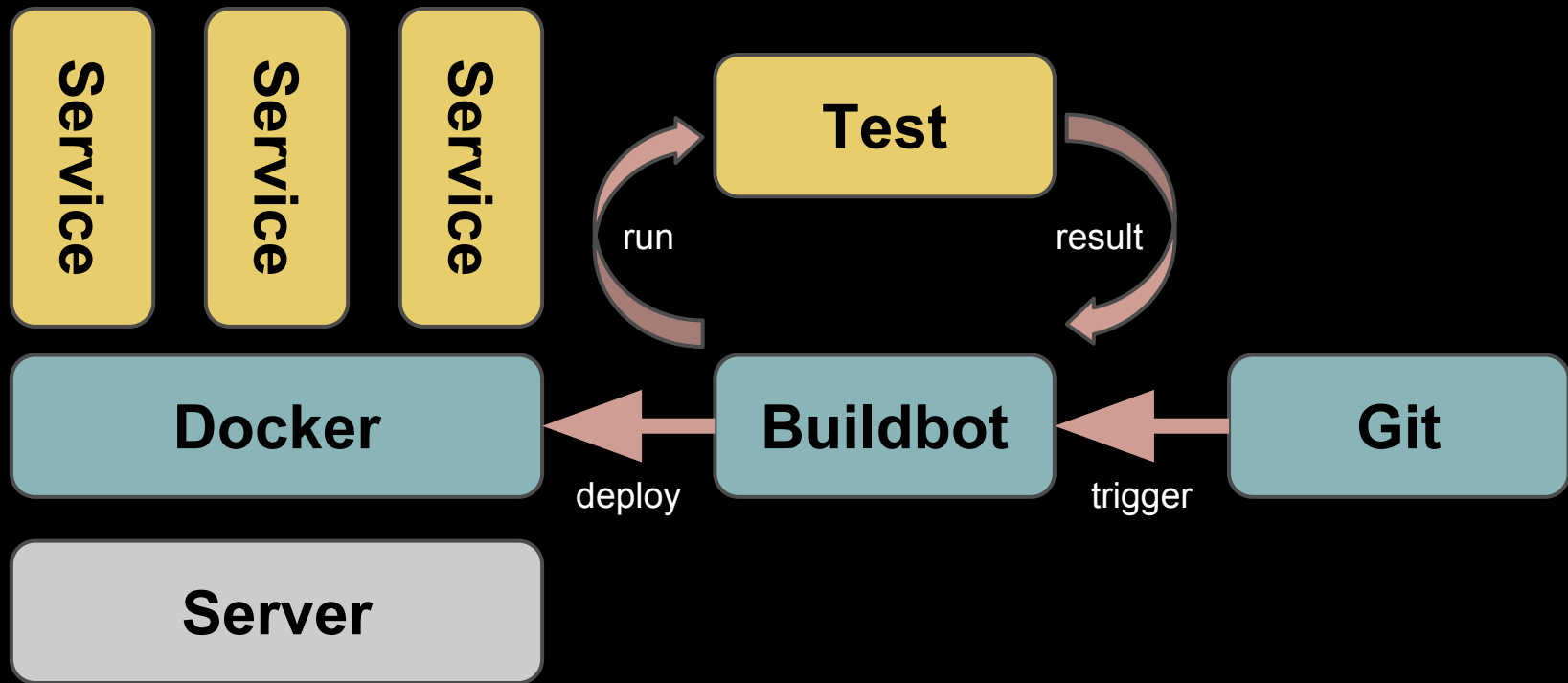
Why it matters

- The version control system that works
- Fast and efficient
- Suitable for collaboration workflow
- Common choice by the open source community
- Github!

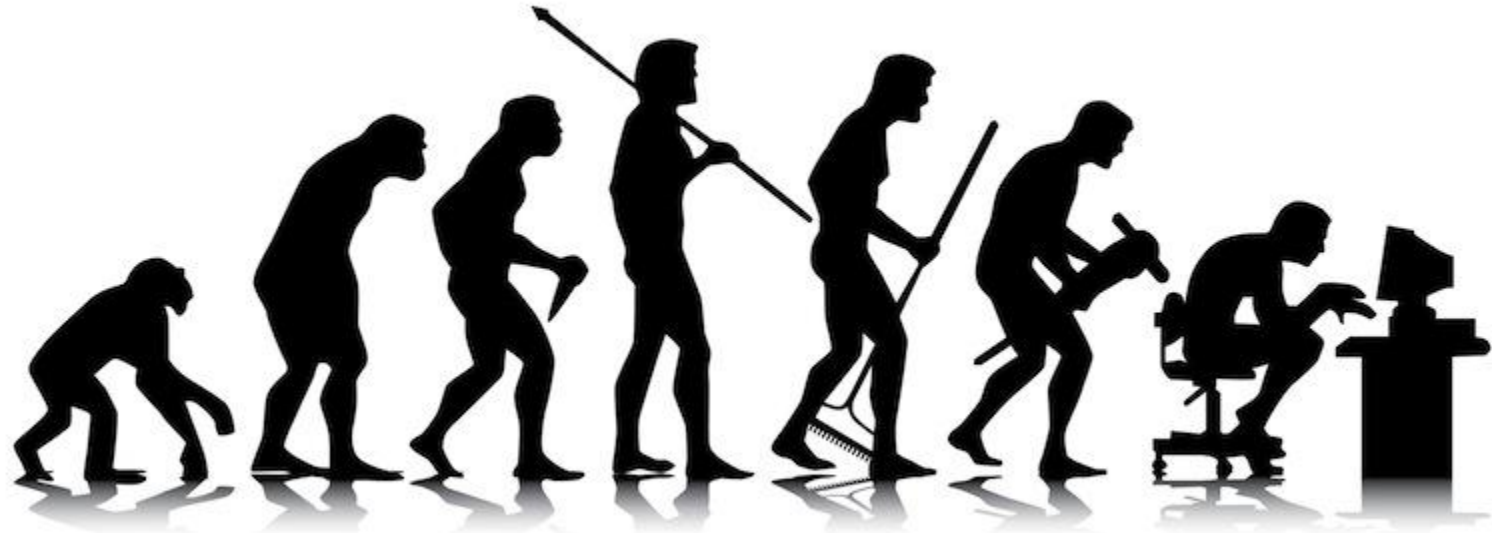
Build a Programmable Software Company

A Case Study

The End State



The Evolution



Fabric

Fabric 1.8 documentation >

next

modules

index

Table Of Contents

Fabric

About

Installation

Development

Documentation

Tutorial

Usage documentation

FAQ

Troubleshooting

API documentation

Core API

Contrib API

Changelog

Roadmap

Getting help

Mailing list

Twitter

Bugs/ticket tracker

IRC

Next topic

Overview and Tutorial

This Page

Show Source

Show on GitHub

Edit on GitHub

Fabric

About

build passing

Fabric is a Python (2.5 or higher) library and command-line tool for streamlining the use of SSH for application deployment or systems administration tasks.

It provides a basic suite of operations for executing local or remote shell commands (normally or via `sudo`) and uploading/downloading files, as well as auxiliary functionality such as prompting the running user for input, or aborting execution.

Typical use involves creating a Python module containing one or more functions, then executing them via the `fab` command-line tool. Below is a small but complete “fabfile” containing a single task:

```
from fabric.api import run

def host_type():
    run('uname -s')
```

Once a task is defined, it may be run on one or more servers, like so:

```
$ fab -H localhost,linuxbox host_type
[localhost] run: uname -s
[localhost] out: Darwin
[linuxbox] run: uname -s
[linuxbox] out: Linux
```

```
Done.
Disconnecting from localhost... done.
Disconnecting from linuxbox... done.
```

The Problem

- No version history
- Only server admin could deploy
- Have to setup development environment to run tests locally
- No deploy history
- Does not work well with multiple repositories
- Non-isolated build environment

Git + Fabric



The Problem

- ~~No version history~~
- ~~Only server admin could deploy~~
- ~~Have to setup development environment to run tests locally~~
- No deploy history
- Does not work well with multiple repositories
- Non-isolated build environment
- Hard to setup

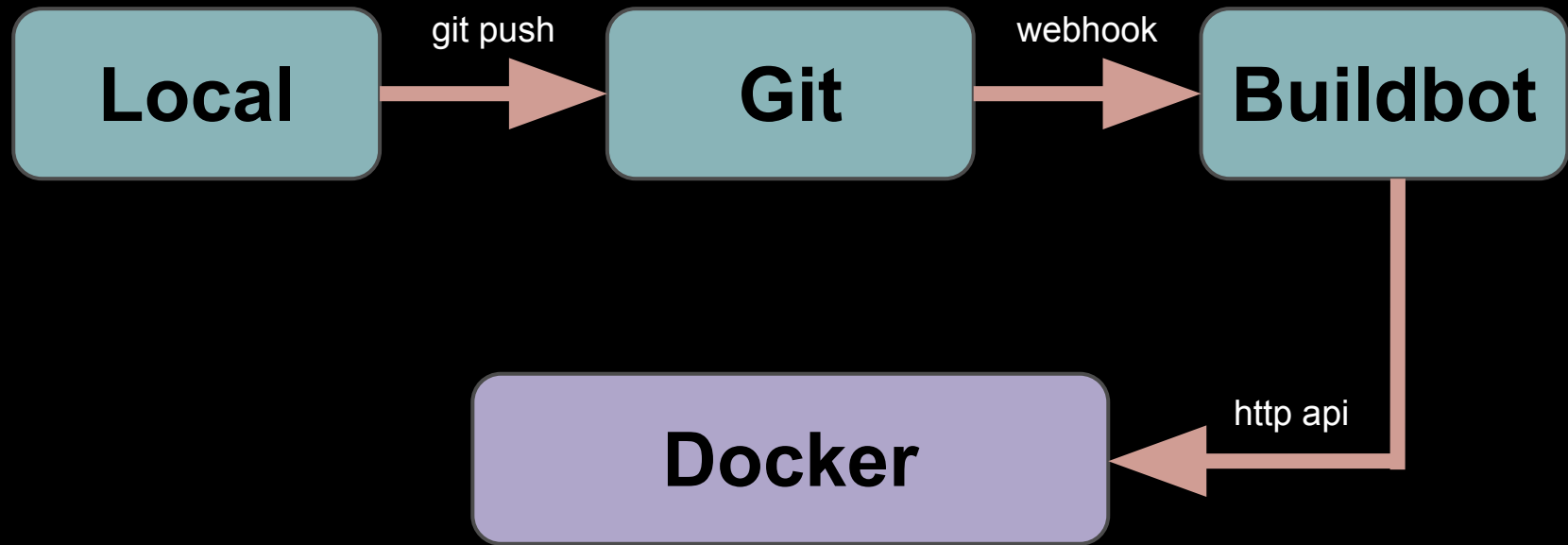
Git + Buildbot



The Problem

- ~~No version history~~
- ~~Only server admin could deploy~~
- ~~Have to setup development environment to run tests locally~~
- ~~No deploy history~~
- ~~Does not work well with multiple repositories~~
- ~~Hard to setup~~
- Non-isolated build environment
- Complex build steps
- Non-repeatable deploy workflow

Git + Buildbot + Docker



docker-py

The screenshot shows the GitHub repository page for `dotcloud/docker-py`. The repository is public and has 260 commits, 3 branches, 14 releases, and 34 contributors. The current branch is `master`. A merge pull request #113 from `jorisvddonk/patch-1` is being reviewed. The repository includes a `Dockerfile`, `requirements.txt`, and `tox.ini`. The right sidebar shows links to Issues (18), Pull Requests (6), Wiki, Pulse, Graphs, and Network. The bottom of the sidebar shows the HTTPS clone URL and buttons for cloning on desktop and downloading ZIP.

260 commits 3 branches 14 releases 34 contributors

branch: master docker-py

Merge pull request #113 from jorisvddonk/patch-1

shin- authored a day ago latest commit a60c1cd3a8

docker	Merge pull request #100 from mpetazzoni/pull-timeout	4 days ago
tests	Ensure sorted order on links to make unit test deterministic	17 days ago
.gitignore	Add python 3 support	5 months ago
.travis.yml	Added travis.yml file	2 months ago
ChangeLog.md	Bumped version to 0.2.2	a month ago
Dockerfile	Added basic Dockerfile	3 months ago
LICENSE	Add Apache 2.0 License	4 months ago
MANIFEST.in	Fixed MANIFEST file	2 months ago
README.md	Update README.md	2 days ago
requirements.txt	Merge branch 'attach-websocket' of github.com:aanand/docker-py into a...	a month ago
setup.py	Bumped version to 0.2.2	a month ago
tox.ini	Flake8 compliance + flake8 tests in tox.ini	2 months ago

Code

Issues 18

Pull Requests 6

Wiki

Pulse

Graphs

Network

HTTPS clone URL



<https://github.com>



You can clone with HTTPS, SSH, or Subversion.

Clone in Desktop

Download ZIP


DockerLatentBuildSlave


  This repository ▾ Search or type a command 🔍 Explore Gist Blog Help

  **adieu / buildbot**
forked from buildbot/buildbot

Unwatch ▾ 1 ★ Star 0 Fork 462

add DockerLatentBuildSlave

 docker

 **adieu** authored 5 months ago 1 parent 4f1c7b2 commit eff651daac179b53676ebd8d487c9e9b21609bc7

Showing 1 changed file with 90 additions and 0 deletions. Show Diff Stats

90 █████ master/buildbot/buildslave/dockerbuildslave.py View file @ eff651d

```
... .. @@ -0,0 +1,90 @@
1  + # This file is part of Buildbot. Buildbot is free software: you can
2  + # redistribute it and/or modify it under the terms of the GNU General Public
3  + # License as published by the Free Software Foundation, version 2.
4  + #
5  + # This program is distributed in the hope that it will be useful, but WITHOUT
6  + # ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS
7  + # FOR A PARTICULAR PURPOSE. See the GNU General Public License for more
8  + # details.
9  + #
10 + # You should have received a copy of the GNU General Public License along with
11 + # this program; if not, write to the Free Software Foundation, Inc., 51
12 + # Franklin Street, Fifth Floor, Boston, MA 02110-1301 USA.
13 + #
14 + # Portions Copyright Buildbot Team Members
15 +
16 + from twisted.internet import defer, threads
17 + from twisted.python import log
18 +
19 + from buildbot.buildslave import AbstractLatentBuildSlave
20 + from buildbot import config, interfaces
```

Benefits

- Each service has its own git repo and runs in a docker container
- Every change is kept track of by git
- Once receiving the change, buildbot will run tests, build a new docker image, stop the old container and start a new one with the newly built image
- The whole system is like a program, one could change a service by modifying the source code of its image
- Empower everyone to change everything

Demo

A close-up photograph of Steve Jobs, wearing his signature round glasses and a black turtleneck. He is pointing his right index finger directly at the viewer. The background is a dark, solid color.

One More Thing...

SaltStack



SALTSTACK

[COMPANY](#)

[BLOG](#)

[PRODUCTS](#)

[COMMUNITY](#)

[SALTCONF 2014](#)



SALT SPEED

Automate faster...much faster

Fast, scalable and flexible software for data center automation, from
infrastructure and any cloud, to the entire application stack

SaltStack wins the 2013 GigaOm Launchpad competition

SaltStack is a Gartner "Cool Vendor in DevOps, 2013"



Gitlab



GitLab.org

[GitLab CE](#)

[GitLab CI](#)

[Community](#)

[Team](#)

[Donate](#)

[API](#)

[Blog](#)

[Try the Demo](#)

GitLab

GitLab is open source software to collaborate on code.
Create projects and repositories, manage access and do code reviews.

[Source code](#)

[Read more](#)

The screenshot displays the GitLab web interface for a project named 'gitlabhq'. The top navigation bar includes the GitLab logo, the project name, and a search bar. Below the navigation bar, there are tabs for 'Files', 'Commits', 'Network', 'Graphs', 'Issues (41)', 'Merge Requests (2)', 'Snippets', and 'Settings'. The 'Files' tab is active, showing a list of files and directories. The 'master' branch is selected. The file list includes 'app', 'config', 'db', 'doc', 'features', 'lib', 'log', 'public', 'script', and 'spec'. Each file entry shows its name, last update time, and the commit it belongs to. The commit details include the commit hash and a brief description of the changes.

Name	Last Update	Last Commit
app	about 18 hours ago	Dmitriy Zaporozhets Fix avatar margin for files view
config	6 days ago	Dmitriy Zaporozhets Merge branch 'broader-message-matching' of /home/git/repositories/gitlabhq...
db	12 days ago	Dmitriy Zaporozhets Namespaces.owner_id can be null
doc	43 minutes ago	Islam Amer Expose votes in merge request api
features	9 days ago	Boyan Tabakov Added Flowdock integration support via a service.
lib	43 minutes ago	Islam Amer Expose votes in merge request api
log	almost 2 years ago	gitlabhq init commit
public	8 months ago	Dmitriy Zaporozhets Add logo to deploy.html
script	4 months ago	Andrew Daugherty use git user in check script
spec	about 16 hours ago	Dmitriy Zaporozhets Fix merge request model scope

Hubot



PASTE EMBLEM HERE

HUBOT

(note: it's pronounced
hew-bot)

1. GIVEN NAME OF INVENTION
PLEASE INCLUDE ANY PRONUNCIATION NUANCES

A CUSTOMIZABLE,
KEGERATOR-POWERED
LIFE EMBETTERMENT ROBOT

2. DESCRIPTION OF INVENTION
PLEASE BE AS CLEAR AND CONCISE AS POSSIBLE

COMMISSIONED BY GITHUB

4. AFFILIATED COMPANY
(INCLUDE A WEBSITE URL IF APPLICABLE)

[Signature]

5. SIGNATURE OF INVENTOR
PLEASE DO NOT ATTEMPT TO SIGN WHILE INEBRIATED

☐ I WOULD LIKE TO VIEW HUBOT'S SOURCE CODE
(OBTAINABLE AT [HTTP://GITHUB.COM/GITHUB/HUBOT/](http://github.com/github/hubot/))

☐ I WOULD LIKE TO CREATE MY OWN HUBOT
(UPGRADE TO NEW VERSIONS VIA PACKAGE.JSON)

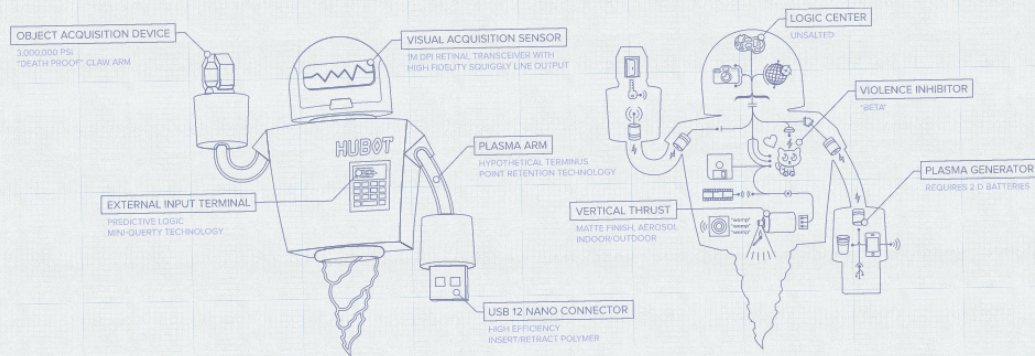


FIG. 1 — HUBOT
SCHEMATICS



contact@sebible.com

Links

Continuous Integration

http://en.wikipedia.org/wiki/Continuous_integration

Docker

<http://www.docker.io/>

Buildbot

<http://buildbot.net/>

Git

<http://git-scm.com/>

Fabric

<http://fabfile.org/>

docker-py

<https://github.com/dotcloud/docker-py>

DockerLatentBuildSlave

<https://github.com/adieu/buildbot/commit/eff651d>

SaltStack

<http://www.saltstack.com/>

Gitlab

<http://gitlab.org/>

Hubot

<http://hubot.github.com/>

Thank You