

A grayscale photograph of an office environment. In the foreground, a person with long hair is seen from behind, working at a desk. To the right, another person wearing headphones is also working at a desk. The background shows more office cubicles and people working. The overall atmosphere is professional and focused.

From Zero to CI/CD

Building your first CI Process

Nir Koren
DevOps CI/CD Team Leader, SAP Gigya Israel

My challenge / goals today

Each one of you will **understand:**

What is CI / CD

What is Git / GitHub and why most of the industry use it.

Introduction to build concepts

Jenkins CI

Each one of you will **have:**

“hello-world” CI process on his/ her Laptop.

What will we cover today?

- Introduction (s)
- Introduction to CI/CD
- **Introduction & hands-on** to Git and GitHub Basics
- **Introduction & hands-on** to Jenkins Basics
- Connect all together



What will we **NOT** cover today?

We are not going to develop anything today.

Not all of us comes from the same background and this not in the Workshop agenda.



Containerized applications: Docker / Kubernetes

Not enough time BUT it goes the same for those technologies as well.

Jenkins Pipelines and Shared libs

Not enough time but we will play with the pipeline in a nutshell.

Agenda

09:00 – 10:30: Introduction to CI / CD

10:30 – 11:00: Coffee Break



11:00 – 12:30: Introduction to Git / GitHub

12:30 – 13:30: Lunch



13:30 – 15:00: Introduction to Jenkins

15:00 – 15:30: Coffee Break



15:30 – 17:00: Jenkins & connect all

WE WILL HAVE TO BE FLEXIBLE

\$ whoami

Nir Koren, DevOps CI/CD Team Leader, SAP Gigya Israel
Doing and implementing DevOps and CI / CD for 10+ years



<https://il.linkedin.com/in/nirkoren>



<https://www.facebook.com/koren.nir>



@KorenNir



@nir_koren

I work for SAP Gigya

100% Cloud SaaS company

Leader of customer identity and access management

Part of SAP Customer Data Cloud solutions

200 / 100K Employees

Acquired by SAP in 2017



THE 2014 LEADERBOARD OF JAVA TOOLS & TECHNOLOGIES



82.5%
JUnit*

TOP TESTING
FRAMEWORK
USED BY
DEVELOPERS

70%
Jenkins°
MOST USED CI SERVER
IN THE INDUSTRY

64%
Maven
MOST USED
BUILD TOOL
IN JAVA

64%
Nexus°
THE MAIN
REPOSITORY
USED BY
DEVELOPERS

69% Git*
#1 VERSION CONTROL
TECHNOLOGY OUT THERE!

67.5%
Hibernate*/°

THE TOP
ORM
FRAMEWORK
USED

65% **Java 7**
THE INDUSTRY LEADER FOR
SE DEVELOPMENT

56%
MongoDB°
THE NOSQL
TECHNOLOGY OF CHOICE

55%
FindBugs*/°
MOST-USED STATIC
CODE ANALYSIS TOOL

50%
Tomcat°
THE MOST POPULAR
APPLICATION SERVER

49%
Java EE 6°
FOUND IN THE MOST
ENTERPRISES

48%
Eclipse
THE IDE USED MORE
THAN ANY OTHER

40% **Spring MVC*/°**
MOST COMMONLY USED WEB FRAMEWORK

32% **MySQL°**
THE MOST POPULAR SQL TECHNOLOGY

* Multiple selections possible

° Normalized to exclude non-user base

Sample population of 2164 Java professionals, sample error 2.1%

RebelLabs Tools and Technologies Leaderboard 2016



Maven

Over **two in three (68%) devs** use Maven as their main build tool



git

Over **two in three (68%) devs** use Git as their version control



Java 8

Almost **two in three (62%) devs** use Java 8 in production



Jenkins

Three in five (60%) devs use Jenkins for CI



IntelliJ IDEA

Almost **one in two (46%) devs** use IntelliJ, the most popular IDE in the survey.



spring

Over **two in five (43%) devs** use Spring MVC



Tomcat

Over **two in five (42%) devs** use Tomcat server in production



ORACLE DATABASE

Almost **two in five (39%) devs** use Oracle DB in production



Microservices

Over **one in three (34%) devs** have adopted a microservices architecture



docker

Almost **one in three (32%) devs** use Docker in production



Java EE 7

Over **three in ten (31%) devs** use Java EE 7



spring

Almost **three in ten (29%) devs** use Spring Boot

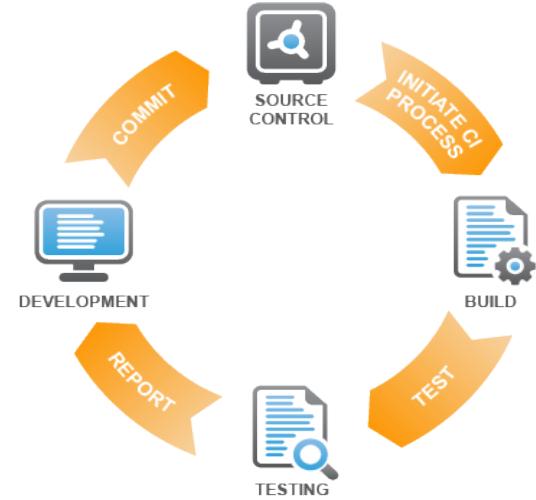


Introduction to **CONTINUOUS INTEGRATION / DELIVERY**

What is Continuous Integration?

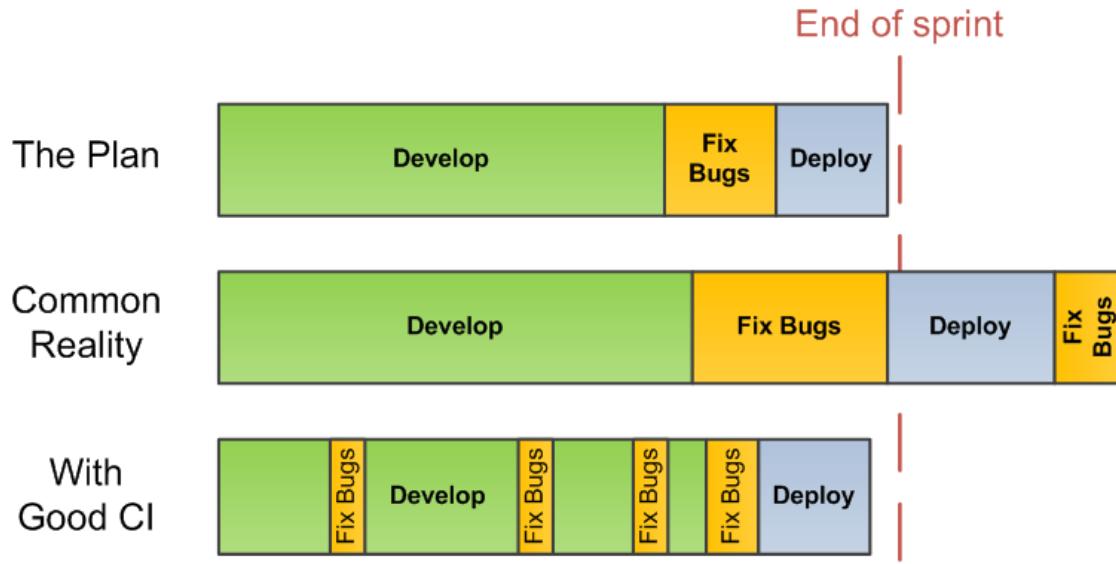
“A key **software development practice** where members of a team **integrate their work frequently**.”

Martin Fowler, Continuous Integration:
<http://martinfowler.com/articles/continuousIntegration.html>



What is Continuous Integration?

Continuous integration involves integrating early and often, in order to avoid "integration hell"



CI / CD Main Goal

Provides a **rapid feedback**.

If a defect is introduced into the code base, it can be **identified** and **corrected ASAP**.

Why CI/CD?

- Frequent changes → Less integration problems.
- Bugs are detected earlier → Saves money.
- Avoid last minute chaos.
- Transparency to all.
- Testing on Production-Like env.
- Easy to rollback in case of any issue.
- Enforce of automation culture.
- Enforce DevOps culture.
- Make the developers accountable and take ownership.

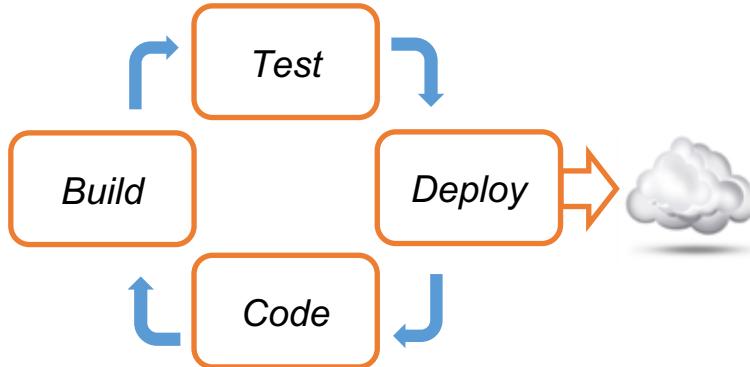
CI Principles

- **Automate** the build (single command), **Automate all**.
- Build is **self testable**.
- Baseline branch is **open consistently**.
- **Every commit** should be built.
- Build should be **fast**.
- Test Env is **clone of Production** (as much as we can).
- Everyone can see the status – **Transparency**.

What is
CONTINUOUS DELIVERY / DEPLOYMENT

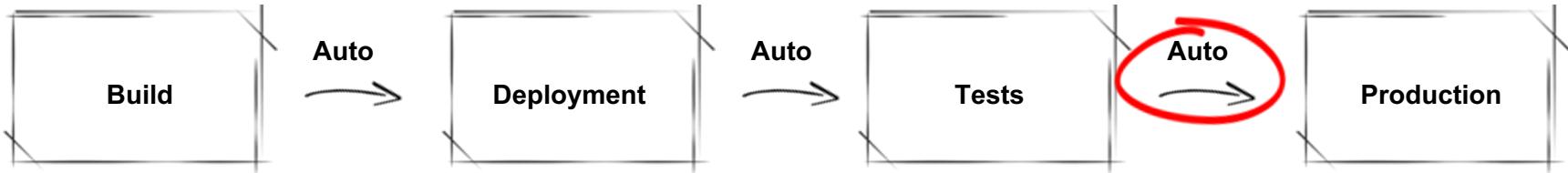
Continuous Delivery & Deployment

Continuous Delivery (CD) is a software development discipline where you build software in such a way that the software **can be released to production at any time**.

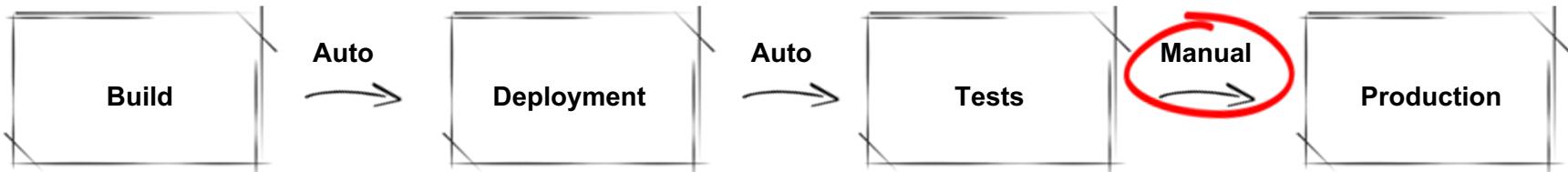


Continuous Deployment means that every change goes through the pipeline and automatically gets put into production, resulting in many production deployments every day.

Continuous Deployment



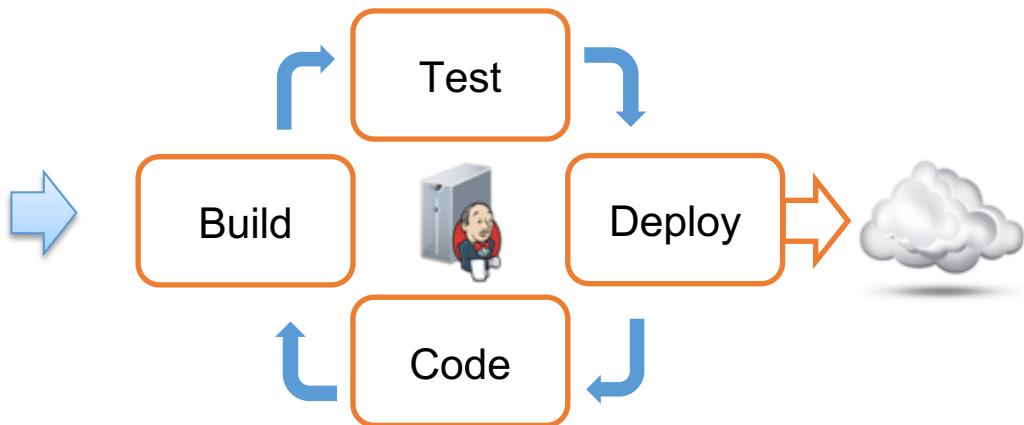
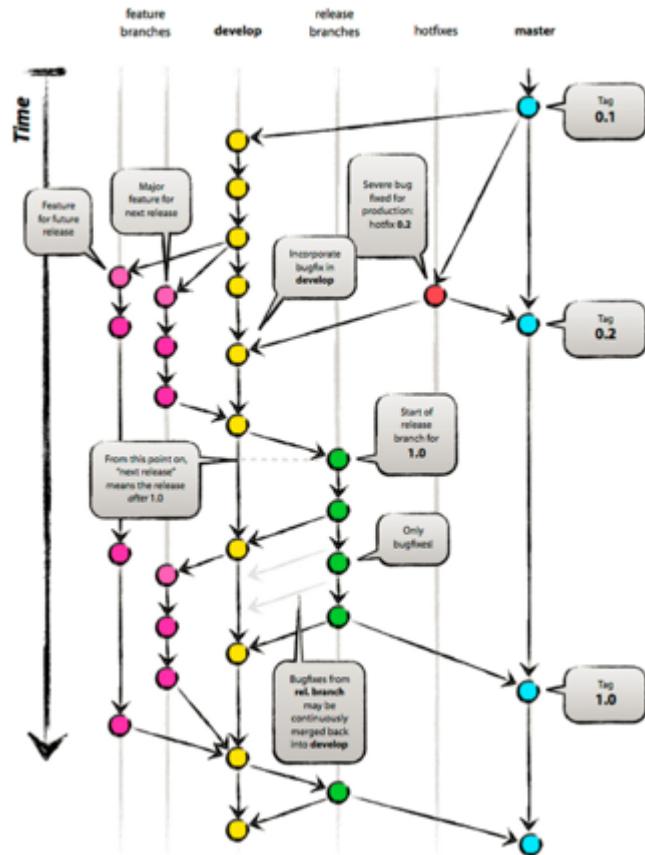
Continuous Delivery



CD Principles

- The Software is always **deployable** through it's lifecycle.
- Anybody can get **fast and automated state** of Production.
- You can perform **push-button deployments** any time

CD Common structure



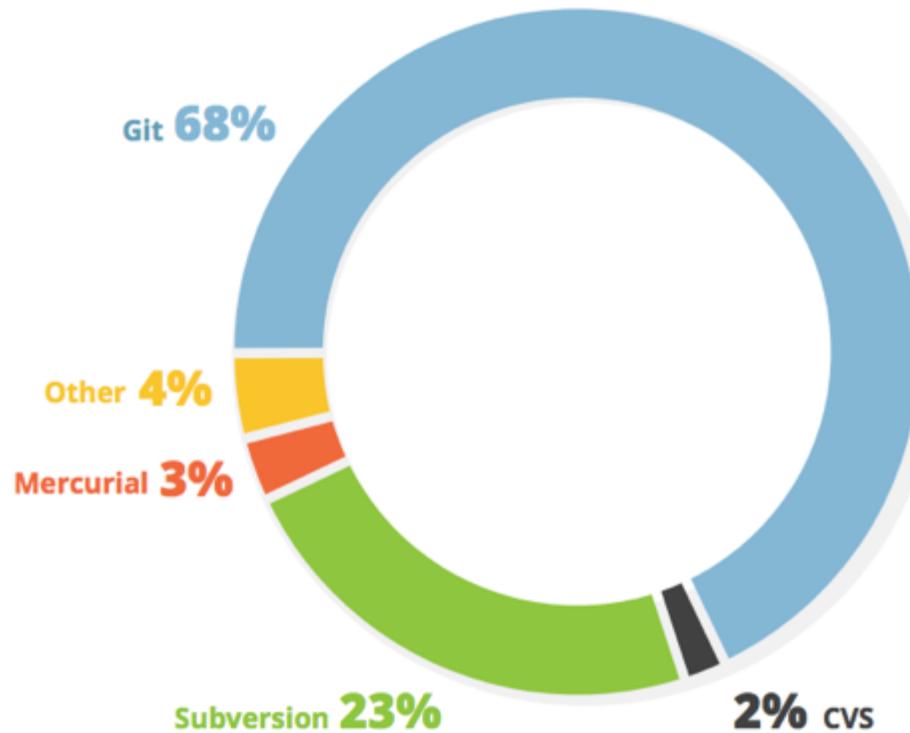
CD is not only in Computer Software

Tesla Model S gets firmware updates on regular basis for both UI and major elements (suspension, acceleration and more)



Introduction to **Git & GitHub**

Figure 1.18 Most Commonly Used VCS



Lets install git & create GitHub account



Linux based: \$ sudo apt-get install git

Windows: Download from <https://git-scm.com/>

Mac: \$ brew install git Or download from <https://git-scm.com/>

Initial configurations:

```
$ git config --global user.name "John Doe"  
$ git config --global user.email johndoe@example.com
```

Check your configurations:

```
$ git config -l
```



<https://github.com/>

Sign Up

Create personal access
token
<https://github.com/settings/tokens>

Git installation / configuration files.

INSTALLATION DIRECTORY



CONFIGURATION DIRECTORY



[user]
name = Nir Koren
email = nirk@liveperson.com

[merge]
tool = p4merge

...

What is Git?

Git is an **Open Source**, **Distributed** and
popular version control system.
Designed for **Speed** and **Efficiency**





Open Source

- Free
- Popular
- Highly contributed by the community and GitHub



Popular

- Multi-platform support
- Multiple GUI's
- Fully command lined – great for CI / Automation
- Contains any IDE / Integrated plugins



Fast

- Performing a diff
- Viewing file history
- Committing changes
- Merging branches
- Obtaining any other revision of a file
- Switching branches

No Network needed.



History & Founder

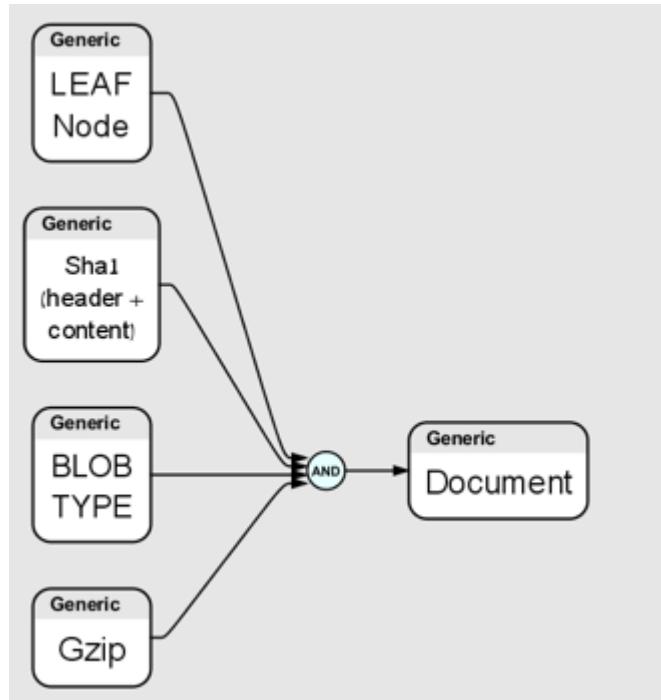
First release @ 2005 for Linux Kernel source code.
New free alternative for BitKeeper.

I'm an egotistical bastard, and I
name all my projects after myself.
First "Linux" and now "git".

Linus Torvalds



Git: Under the hood



Key Concepts

COMMIT

A **Snapshot** in the repository

State of the project in a certain time. Point of a tree

BRANCH

A label with **user-friendly name**. Point to a **commit**.

TAG

A descriptive name to a **list of commits**.

Technically – it's all the same

Key Concepts

REMOTE

Can be **origin** (where it cloned from) or **upstream** (custom)

The version that hosted in the server. Can be **remote host** or **local clone**.

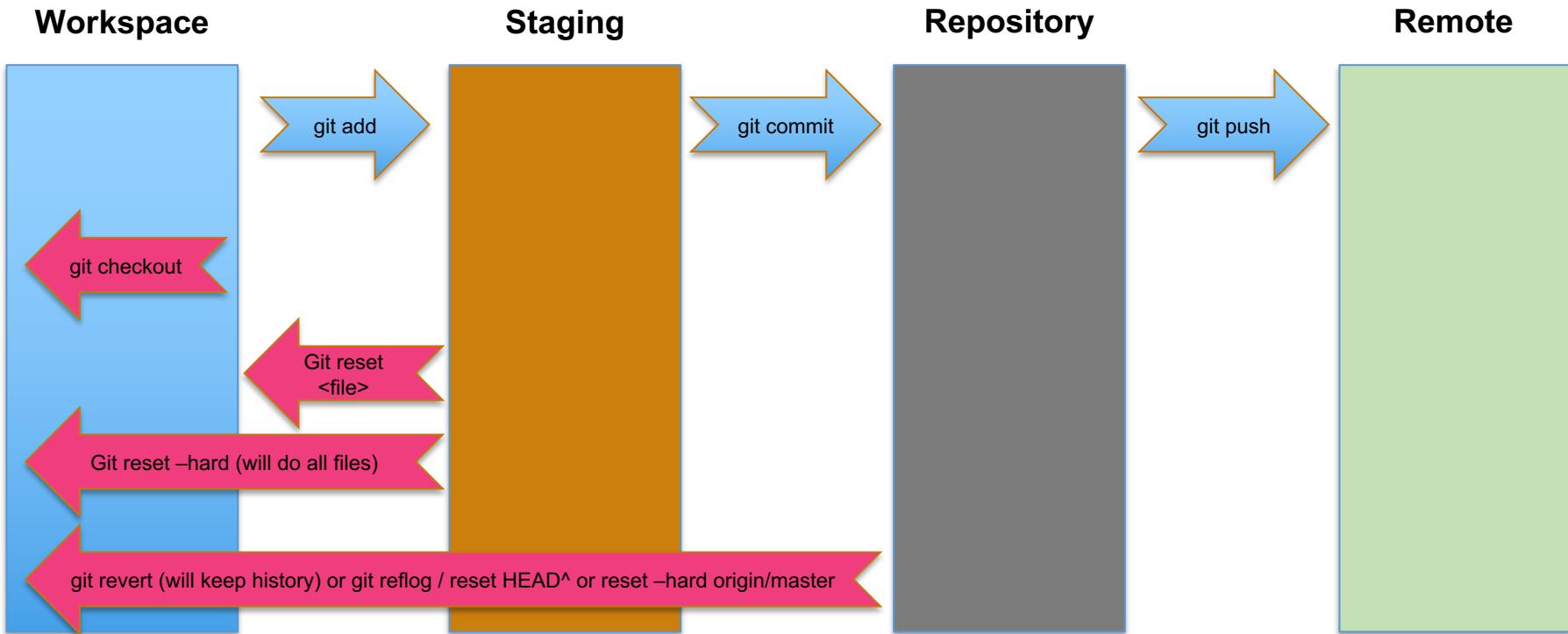
CLONE

Copy of a repository locally on your laptop. Comes with metadata relationship to the remote repo

FORK

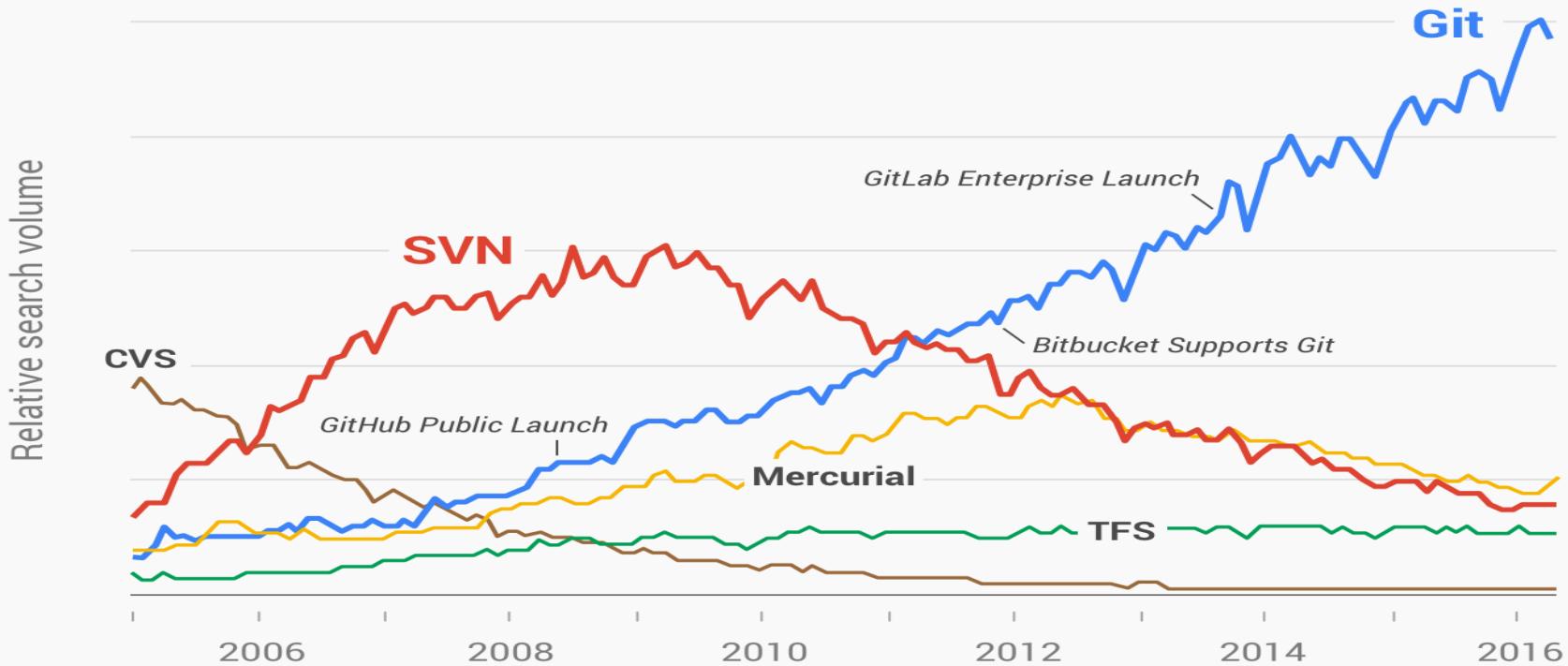
A **personal copy** of a repository that lives **remotely** on a user account and holds relationship with the original repo..

Git: Areas



How Did SVN Manage to Lose?

Version control interest over time



Companies & Projects Using Git

Google

facebook.

Microsoft

twitter

LinkedIn.

NETFLIX



PostgreSQL



Perforce new products
contains integration to
Git

Work Locally, Scale Globally

We've partnered with GitLab to introduce
our GitSwarm ecosystem...

- Merge-request workflow
- Easy repo management
- Project visibility & security
- Automatically mirrors work into Helix mainline repository

GET IT NOW



The best way to learn Git

FORGET Perforce

FORGET ClearCase

FORGET SVN

FORGET Any VCS you know

It's much easier to teach a person who doesn't know any
VCS

Philosophy

- While **SVN trunk** is development, **GIT Master** is Release.
- Anything in **origin master** is **deployable**.
- Create a descriptively **named branch** for new feature development
- **Commit early and often** to new branch locally
- Ensure that **each commit represents one idea** or complete change.
- Push your new branch work to the **same named branch on origin**

Introduction to



What is GitHub?

Web-based hosting service for version control using Git.
The largest source code in the world



GitHub Company

- Founded in 2007. Owned by Microsoft
- Headquartered in San Francisco, 2500+ employees
- GitHub.com > 73M users, 200M Repositories
- Core GitHub developers are core Git OS contributors
- Cool brand (shop, merchandizing, crazy company)



Any Microsoft employees here?

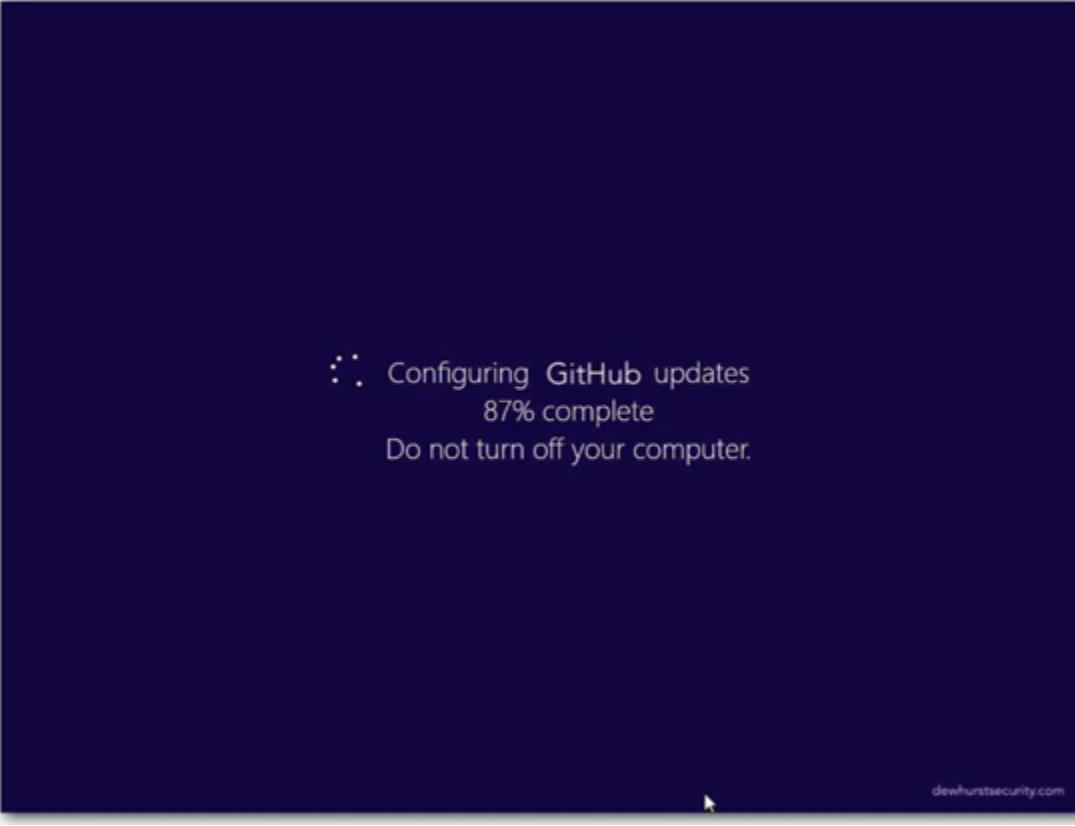


Microsoft[®]

GitHub 2019

Home edition

Please enter your product key :



Configuring GitHub updates
87% complete
Do not turn off your computer.

Warning



Please commit and push now.

You must commit and push now, otherwise we will do it
silently in the background anyways. It's your choice.

Remind me in:

10 minutes

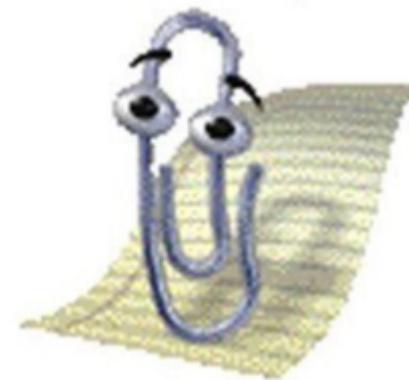


Ruin my repo

Postpone

- GitHub Starter
- GitHub Home Basic
- GitHub Home Premium
- GitHub Professional
- GitHub Enterprise
- GitHub Ultimate

Whoops, it looks like you
have some merge conflicts !



Just Kidding ☺

Pricing models

- **Free:** contains part of the features used.
- **Team:** extra features and privacy.
- **Enterprise (hosted):** private in the cloud Github.com
- **Enterprise (Enterprise):** On-premise. Install and maintain on your own.

Partial features list of
WHY GITHUB?

In-place editing using Ace Editor



Nir Koren init commit

Latest commit dc3e34a on Nov 24, 2021

0 contributors

17 lines (13 sloc) | 406 Bytes

[Raw](#) [Blame](#)

```
1 package org.nirkoren.maven.demo;
2
3 import org.junit.Assert;
4 import org.junit.Test;
5 import org.nirkoren.maven.demo.HelloHandler;
6
7 public class HelloHandlerTest {
8
9     @Test
10    public void validateNameNotNull() {
11        HelloHandler handler = new HelloHandler();
12        String response = handler.sayHello();
13        // Due to the code: This test will never fail :)
14        Assert.assertNotNull("String got null value",response);
15    }
16
17 }
```

Both Machine and Human Code Review

Issues **Pull requests** Labels Milestones Filters ▾ is:pr is:closed

0 Open 4 Closed Author ▾ Labels ▾ Milestones ▾

Changeplugininfra ✓ ✓ #4 opened on Dec 20, 2015 by nirk

Changeplugininfra ✓ #3 opened on Dec 15, 2015 by nirk



Code Search engine

The screenshot shows a search interface for a code search engine. The search term "StringBuffer" has been entered into the search bar. The results page displays 9,234,413 code results.

Repositories (247)

- Code (9M)
- Commits (90K)
- Issues (15K)
- Discussions (17)
- Packages (0)
- Marketplace (0)
- Topics (1)
- Wikis (1K)
- Users (3)

Languages

Language	Count
Markdown	20,814
C	68,333
HTML	382,318
Small	126,587
C++	375,193
Dart	112,743
Java Server Pages	53,859
Java	6,914,560
Text	94,175
XML	73,574

9,234,413 code results

Sort: Best match ▾

wspr-ncsu/cardpliance
java/src/main/resources/summaries/StringBuffer.safsu

```
1 java.lang.StringBuffer:value:java.lang.String;
2
3 `Ljava/lang/StringBuffer;.<init>():V`:
4     this.value = java.lang.String@-
5 ;
6
7 `Ljava/lang/StringBuffer;.<init>:(I)V`:
8     this.value = java.lang.String@-
```

Showing the top three matches Last indexed on Mar 26, 2021

qwertyscoin-org/qwertyscoin-forkable
tests/UnitTests/StringBufferTests.cpp

```
17 // along with Qwertyscoin. If not, see <http://www.gnu.org/licenses/>.
18
19 #include <Common/StringBuffer.h>
20 #include <gtest/gtest.h>
21
22 using namespace Common;
23
24 TEST(StringBufferTests, defaultConstructor) {
25     const StringBuffer<16> buffer;
```

C++ Showing the top two matches Last indexed on Mar 27, 2021

AgusFoundation/aguscoin_3.0
tests/UnitTests/StringBufferTests.cpp

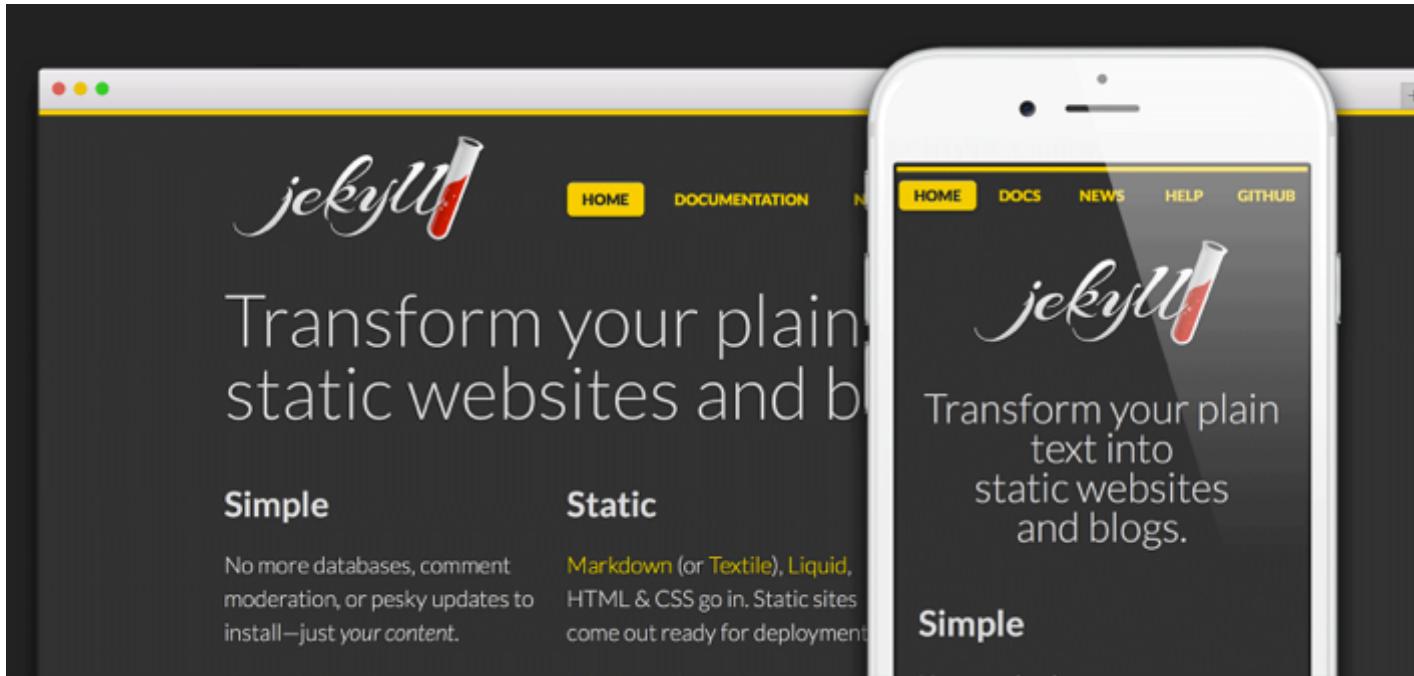
```
18 #include <Common/StringBuffer.h>
19 #include <gtest/gtest.h>
20
21 using namespace Common;
```

Rich documentation and API

The screenshot shows the Postman application interface. At the top, there is a header bar with a dropdown menu set to "GET" and a URL field containing "https://api.github.com/repos/CI/ci-dot-utils/commits?sha=master&author=vifat...". Below the header, there are tabs for "Authorization", "Headers (1)", "Body", "Pre-request Script", and "Tests". The "Headers (1)" tab is selected and highlighted with an orange underline. It contains a table with one row, where the "Key" column is "Authorization" and the "Value" column is "Basic bmllyazpNeU1pa2EwMDk5IQ==". There is also a "New key" input field. Below the headers, the "Body" tab is selected and highlighted with an orange underline. It contains sub-tabs for "Pretty", "Raw", "Preview", and "JSON". The "JSON" sub-tab is selected and highlighted with an orange underline. The JSON response is displayed in a code editor-like area with line numbers from 1 to 14 on the left. The JSON data is as follows:

```
1 [  
2 {  
3   "sha": "54e000405918e0665c7780ed8690b73178b91721",  
4   "commit": {  
5     "author": {  
6       "name": "Vladi Ifat",  
7       "email": "vifat@liveperson.com",  
8       "date": "2018-03-25T12:02:06Z"  
9     },  
10    "committer": {  
11      "name": "Vladi Ifat",  
12      "email": "vifat@liveperson.com",  
13      "date": "2018-03-25T12:02:06Z"  
14    },  
15  }]
```

GitHub Pages



GitHub WIKI

fabric8io / docker-maven-plugin

Code Issues 253 Pull requests 14 Projects 0 Wiki Insights

docker:start

Roland Huß edited this page on Feb 27, 2015 · 3 revisions

docker:start

Maven Goal for starting a container. This goal has various configuration parameters which influence its behaviour. This goal is best attached to the `pre-integration-test` phase of the Maven lifecycle, so that containers are started before the integration test runs.

If multiple containers are required they should be attached individually to the lifecycle, each with their own configuration. Common configuration options then should go into the main configuration section. Creates and starts a docker container.

GitHub Issues

fabric8io / docker-maven-plugin

Watch 75 Star 842 Fork 347

Code Issues 253 Pull requests 14 Projects 0 Wiki Insights

Want to submit an issue to fabric8io/docker-maven-plugin? Dismiss

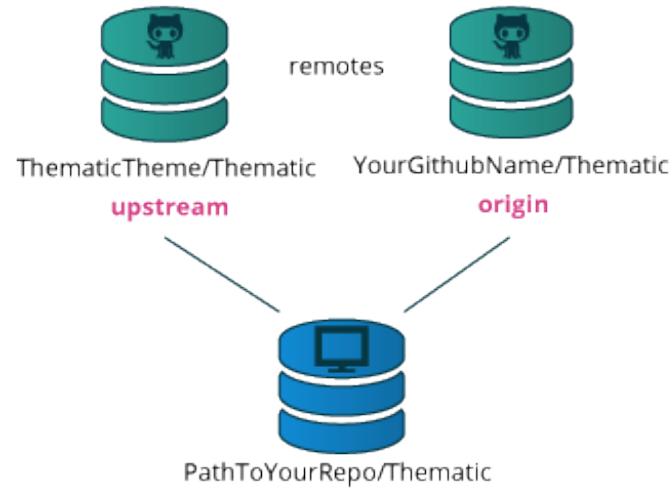
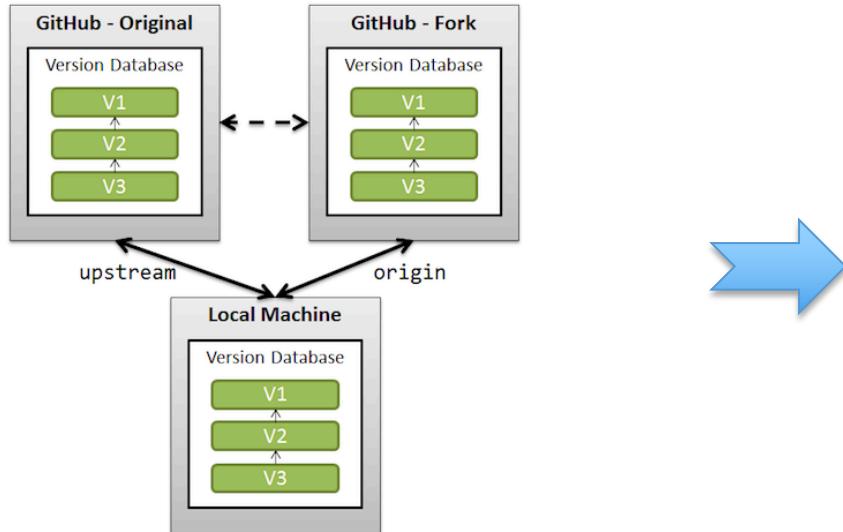
If you have a bug or an idea, read the [contributing guidelines](#) before opening an issue.

Filters ▾ is:issue is:open Labels Milestones New Issue

① 253 Open ✓ 391 Closed Author ▾ Labels ▾ Projects ▾ Milestones ▾ Assignee ▾ Sort ▾

- ① **How to add arguments while building a docker image using plugin**
#1011 opened 5 days ago by rajat-garg
- ① **log file output broken in 0.24.0, 0.25.0, 0.25.1, 0.25.2**
#1010 opened 5 days ago by chonton
- ① **Dockerfile with zero-config**
#1008 opened 7 days ago by burtsevyyg
- ① **Add support for passint --init parameter to docker when starting containers**
#1003 opened 13 days ago by thrawn-sh
- ① **Change default phase for docker:build to "package"**

Forks model – Open Source work model



Pull Request

Tell others about your change, Let human / machine to review your code.

The screenshot shows a GitHub pull request interface. At the top, it displays the repository name "hubot / Spoon-Knife" and the URL "forked from octocat/Spoon-Knife". To the right are buttons for "Watch", "Star", "Fork", and a count of 34,641 forks. Below this is a navigation bar with tabs for "octocat:master" and "hubot:master". A prominent green button labeled "Create pull request" is visible. The main area contains a yellow banner with the text "Discuss and review the changes in this comparison with others." Below the banner, summary statistics are shown: "1 commit", "1 file changed", "0 commit comments", and "1 contributor". A commit log for "Commits on Sep 10, 2014" shows a single commit by "hubot" with the message "Change description of the repository". The commit hash is Bb7afff. Below the commit log, it says "Showing 1 changed file with 1 addition and 1 deletion." A "Unified" and "Split" view switch is present. The bottom section shows a diff view for the file "README.md". The diff highlights changes in the file, such as the addition of "Well hello there!" and the explanatory text about forking. The commit message "Change description of the repository" is also visible in the diff.

```
diff --git a/README.md b/README.md
index 00... -1.6 +1.6 00
--- a/README.md
+++ b/README.md
@@ -1,2 +1,6 @@
 1 1  ## Well hello there!
 2 2
 3 -This repository is meant to provide an example for *forking* a repository on GitHub.
 4 +This is my fork of the octocat/Spoon-Knife repository.
 5
 6  Creating a *fork* is producing a personal copy of someone else's project. Forks act as a sort of bridge
 7  between the original repository and your personal copy. You can submit *Pull Requests* to help make other
 8  people's projects better by offering your changes up to the original project. Forking is at the core of
 9  social coding at GitHub.
```

Communicate with Emojis

 dannyfritz New emoji for deprecation notices	
 INTEGRATIONS.md	 Update emoji packages required for Atom
 LICENSE	 Initial commit
 README.md	New emoji for deprecation notices

Manage your tasks in Board

The screenshot shows a Jira board interface with four columns: ToDo, InProgress, BLOCKED, and Completed.

- ToDo (18 items):**
 - Add discovery for service health per BB. #96 opened by liranc
 - Move scriptier code to main DSL code as func. #86 opened by liranc
 - Investigate about dashboarding / transparency of the activity of the Jenkins / jobs #78 opened by nirk
 - when choosing alpha segment and pci yes then show only alpha servers and not va #69 opened by tomerb
 - cobrowse_app lpnova role does not match module
- InProgress (6 items):**
 - AC Metadata is identified as jboss instead of tomcat causes build to fail #95 opened by tomerb
 - If rnd-a person created pull request cannot assign to rnd-b person in lpgithub #80 opened by tomerb
 - Support for PCI #46 opened by liranc enhancement
 - puppet module d6r_web #4 opened by liranc
 - LProle_mng #47 opened by liranc
 - In D6R-PROLE where all nodes
- BLOCKED (1 item):**
 - ssh to Servers #7 opened by liranc
- Completed (49 items):**
 - PCI: svvr-mng40 is not open to ca/lojam which causes D6R to fail #94 opened by tomerb
 - SY-R10K no such command found (root@rmor-mng32 ~)\$ r10k -bash: r10k: command not found #87 opened by tomerb
 - SY-R10K itself fails on sydney see logs #90 opened by tomerb
 - deployer validation #44 opened by tomerb
 - Integrate Orca with system health #26 opened by kobyh

Key Concepts

ORGANIZATION

Group of **2 or more users** that typically acts like a real org / company. It administrated and contains teams, users and repos.

REPOSITORY

Collection of files with metadata (.git folder) that contains history, revisions etc'

COLLABORATOR

A person who had read / write permissions to a repository.

CONTRIBUTOR

A person who contributes to a repository but has no collaborator access. Only PULL requests.

NIRKOREN/SWIFT

APPLE/SWIFT

ORIGIN

UPSTREAM



Practice

Network might be slow

In the meantime - Please download from

<https://jenkins.io/download/>

 Download Jenkins 2.303.3 LTS for:

Generic Java package (.war)

SHA-256: 8a6ae7367755b3f31a050faa945f7a3991abdb43d941c7294cac890c1e2779d8

Introduction to **Build processes**

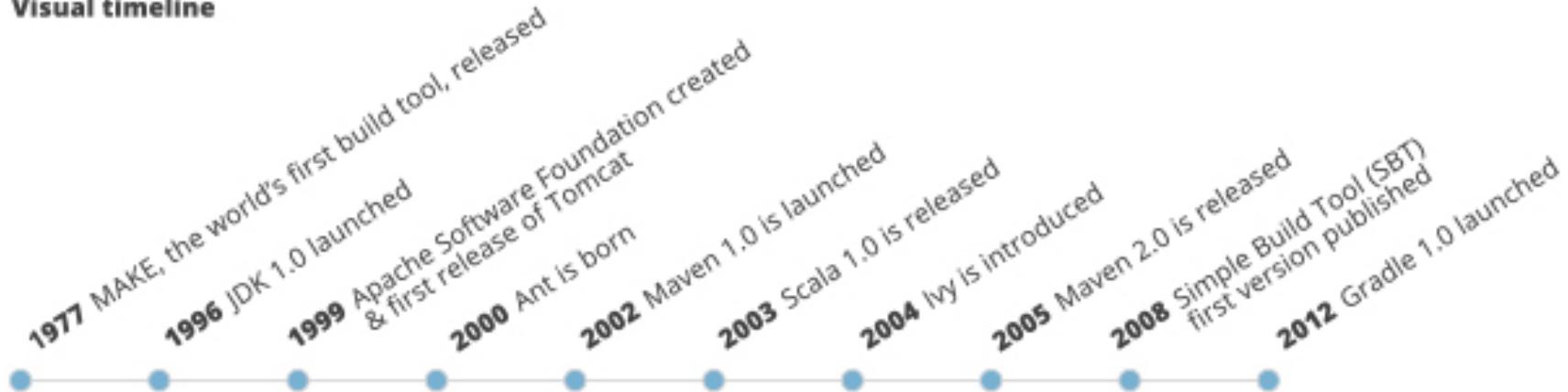
What is “Build”?



History of build tools

THE EVOLUTION OF BUILD TOOLS: 1977 - 2013 (AND BEYOND)

Visual timeline



Common Architecture



Build command



Local Repository
(~/.m2/repository)



Deploy



Proxy

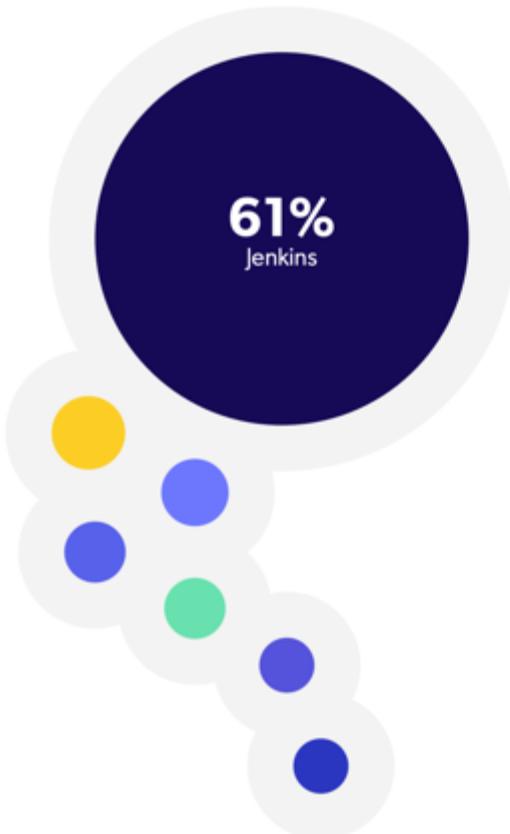


Common build pipeline



Introduction to **Jenkins CI Server**

Which CI/CD technologies are you using?



Jenkins **61%**

Bamboo **12%**

Travis CI **11%**

N/A **10%**

Other **10%**

TeamCity **9%**

Circle CI **9%**

Let's install Jenkins first

- Download Jenkins (Preferred LTS) <https://jenkins.io/>
- Open up a terminal in the download directory.
- Run `java -jar jenkins.war --httpPort=9090`.
- Browse to `http://localhost:9090`.
- Follow the instructions to complete the installation.



Docker?

```
$ docker run -d -p 9090:8080 -u root -v  
/Users/nirk/jenkins_home:/var/jenkins_home jenkins/jenkins:lts
```

What is Jenkins?

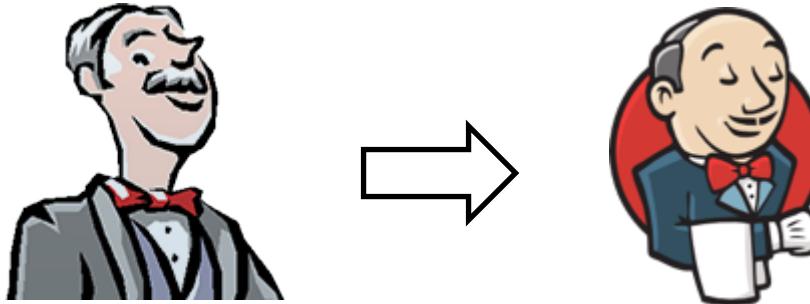
Jenkins is **open source automation server** which can be used to automate all sorts of tasks related to building, testing, and delivering or deploying software.

Jenkins can be installed through native system packages, Docker, or even run standalone by any machine with a Java Runtime Environment (JRE) installed.



History of Jenkins

- Funded by **Kohsuke Kawaguchi** as Hudson CI (in that time he was a developer in Sun Microsystems).
- First Hudson release was in 2004.
- Oracle acquired Sun in 2010 and claimed the right to the "**Hudson**" name and applied for a trademark in December 2010.
- Kohsuke forked Hudson open source and created the name "**Jenkins**" – the rest is history.



Why Jenkins?

A platform, Free, Open source. Easy to install
and easy to use





Jenkins

Provides built-in Authentication & Authorization

Login



Jenkins Server



Authorization Rules



LDAP



GITHUB User base



SAML IDP



Local User base



Jenkins

Open platform

1000+ plugins



The screenshot shows the Jenkins project dashboard for 'myJenkinsJob'. The left sidebar contains links for Status, Changes, Workspace, Build Now, Delete Project, Configure, Rebuild Last, Favorite, and Job Config History. The main content area displays the 'Project myJenkinsJob' header, workspace and recent changes links, and a 'Permalinks' section with a bulleted list of build URLs. Below this is a 'Build History' table with four rows, each showing a green status icon, a timestamp of 'Nov 8, 2017 11:07 AM', and a 'View' link. At the bottom are two 'RSS for' links: 'RSS for all' and 'RSS for failures'.

JSON API

```
{
  "_class": "hudson.model.FreeStyleProject",
  "actions": [
    {
      "name": "myJenkinsJob"
    }
  ],
  "description": "",
  "displayName": "myJenkinsJob",
  "displayNameOnWall": null,
  "fullDisplayName": "myJenkinsJob",
  "fullName": "myJenkinsJob",
  "name": "myJenkinsJob",
  "url": "http://ctv-jenkins:8080/job/myJenkinsJob/",
  "buildable": true,
  "builds": [
    {
      "number": 1,
      "url": "http://ctv-jenkins:8080/job/myJenkinsJob/1/"
    }
  ]
}
```

XML API

```
<freeStyleJob _class="hudson.model.FreeStyleProject">
  <actions/>
  <description/>
  <displayName>myJenkinsJob</displayName>
  <displayNameOnWall></displayNameOnWall>
  <fullDisplayName>myJenkinsJob</fullDisplayName>
  <name>myJenkinsJob</name>
  <url>http://ctv-jenkins:8080/job/myJenkinsJob/</url>
  <buildable>true</buildable>
  <builds>
    <build _class="hudson.model.FreeStyleBuild">
      <number>1</number>
      <url>http://ctv-jenkins:8080/job/myJenkinsJob/1/</url>
    </build>
  </builds>
  <lastBuild>
    <number>1</number>
    <url>http://ctv-jenkins:8080/job/myJenkinsJob/1/</url>
  </lastBuild>
  <lastStableBuild>
    <number>1</number>
    <url>http://ctv-jenkins:8080/job/myJenkinsJob/1/</url>
  </lastStableBuild>
  <lastSuccessfulBuild>
    <number>1</number>
    <url>http://ctv-jenkins:8080/job/myJenkinsJob/1/</url>
  </lastSuccessfulBuild>
  <lastCompletedBuild>
    <number>1</number>
    <url>http://ctv-jenkins:8080/job/myJenkinsJob/1/</url>
  </lastCompletedBuild>
</freeStyleJob>
```



Jenkins

Extendable

Write your own plugin



Support Groovy DSL



Groovy pipelines

```
Jenkinsfile (Declarative Pipeline)
pipeline {
    agent any
    stages {
        stage('Build') {
            steps {
                sh 'make'
            }
        }
        stage('Test') {
            steps {
                sh 'make check'
                junit 'reports/**/*xml'
            }
        }
        stage('Deploy') {
            steps {
                sh 'make publish'
            }
        }
    }
}
```



Jenkins

Popular by the community





Jenkins

Short-time (rapid) Development



Practice