GitLab

GitLab CI/CD Introduction

WiFi Code

msevent49901

Agenda

- Introduction
  - Who are we and who is GitLab?
  - Where is your team today?
  - Concepts defined - what is CI/CD?
  - Why?
  - How?
- GitLab CI/CD Architecture
- GitLab CI/CD Runners
- CI/CD Pipeline Definition
- Hands-On Workshop
- Q&A

GitLab is the most popular solution for the Enterprise

COMPANY

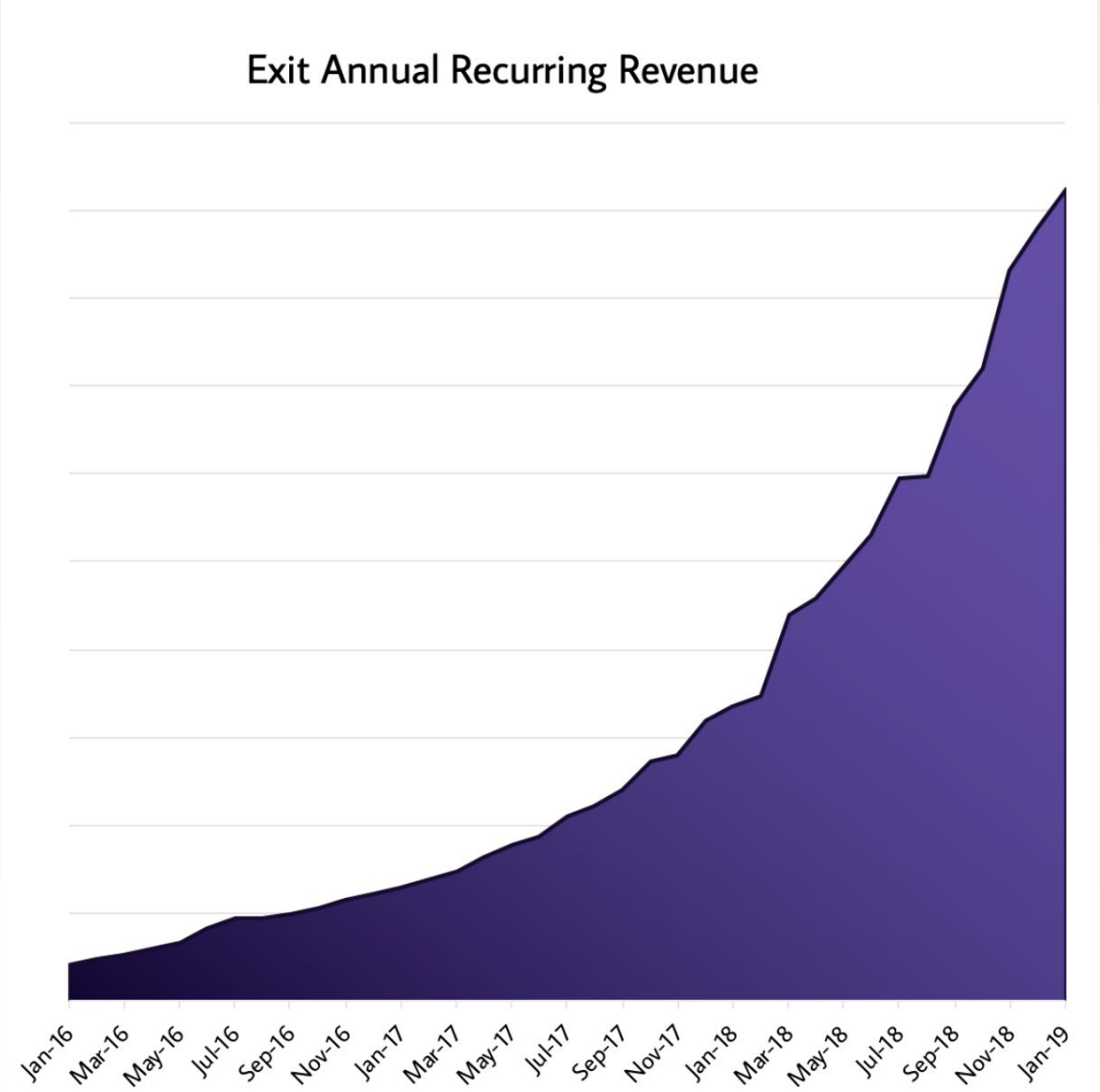
- Incorporated in 2014
- 700+ employees across 56 countries
- GitLab Federal entity est. in 2018

BROAD ADOPTION

- 100,000+ organizations
- Millions of users
- 70% share of self-managed DevOps repository market

STRONG COMMUNITY

- Open source model
- 2,200+ code contributors
- 10,000+ total contributors



Values

Collaboration

- **Work asynchronously** with fully remote workforce.
- Use GitLab to build GitLab. There's an Issue and/or Merge Request for everything.

Results

- **Track outcomes, not hours.**

Efficiency

- **Boring solutions** win. Complexity slows cycle time.

Diversity

- Remote-only tends toward global diversity, but we still have a ways to go.
- Hire those who add to culture, not those who fit with culture. We want **cultural diversity** instead of cultural conformity.

Iteration

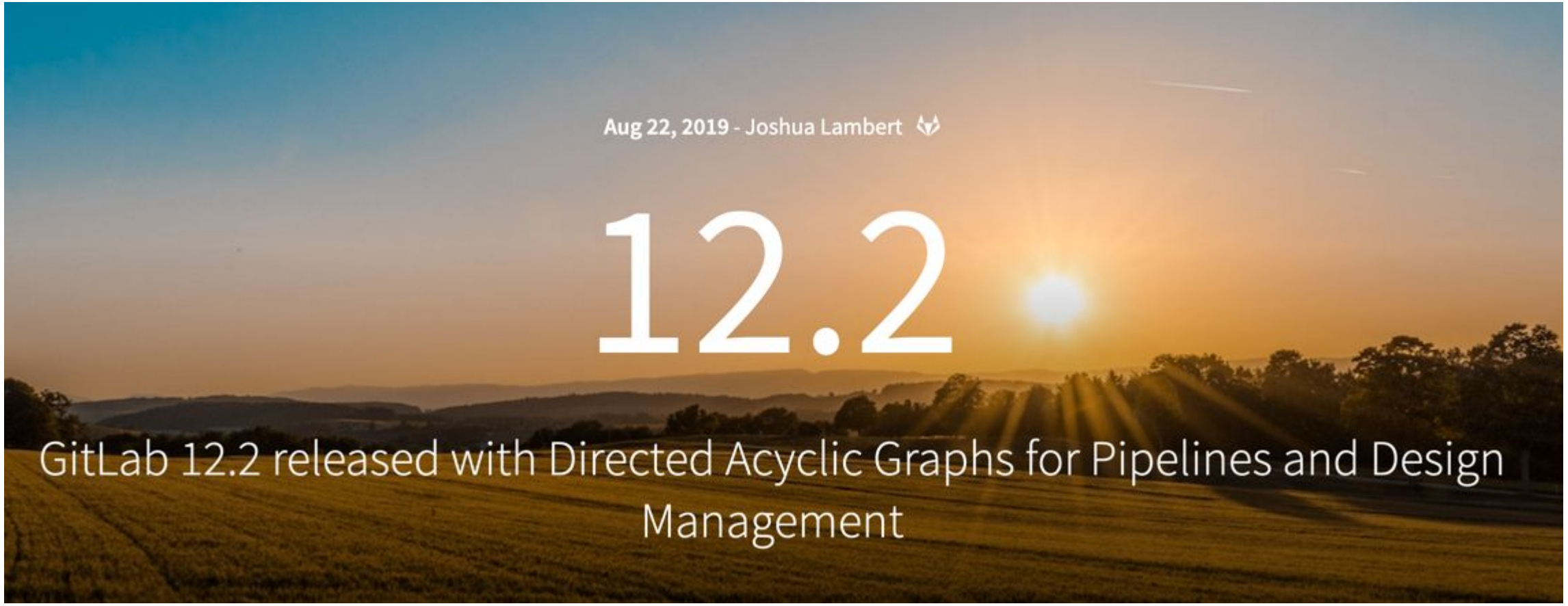
- **Minimum Viable Change** (MVC); if the change is better than the existing solution, ship it.

Transparency

- Everything at GitLab is **public by default**: Strategy, Roadmap, Quarterly goals, Handbook, and Issue Trackers.

It’s Release Day!

- We release the 22nd of EVERY month
- We are close to 100 consecutive months of releases



Why Are We Here

You Own GitLab

Using it for code repo only

Lots more to GitLab than just code repo, we're going to show you another major piece

GitLab is the first single application for the entire DevOps lifecycle

Manage	Plan	Create	Verify	Package	Secure	Release	Configure	Monitor	Defend
Since 2016	**Since 2011**	**Since 2011**	**Since 2012**	**Since 2016**	**Since 2017**	**Since 2016**	**Since 2018**	**Since 2016**	**Coming soon:**
Cycle Analytics	Kanban Boards	Source Code Management	Continuous Integration (CI)	Container Registry	SAST	Continuous Delivery (CD)	Auto DevOps	Metrics	Runtime Application Self Protection
DevOps Score	Project Management	Code Review	Code Quality	Maven Repository	DAST	Release Orchestration	Kubernetes Configuration	Logging	Web Application Firewall
Audit Management	Agile Portfolio Management	Wiki	Performance Testing	NPM Registry	Dependency Scanning	Pages	ChatOps	Cluster Monitoring	Threat Detection
Authentication and Authorization	Service Desk	Snippets	**Coming soon:**	**Coming soon:**	Container Scanning	Review Apps	Serverless	Tracing	Behavior Analytics
Coming soon:	**Coming soon:**	Web IDE	System Testing	Rubygem Registry	License Management	Incremental Rollout	**Coming soon:**	Error Tracking	Vulnerability Management
Code Analytics	Value Stream Management	**Coming soon:**	Usability Testing	Linux Package Registry	**Coming soon:**	Feature Flags	PaaS	**Coming soon:**	Data Loss Prevention
Workflow Policies	Requirements Management	Design Management	Accessibility Testing	Helm Chart Registry	Secret Detection	**Coming soon:**	Chaos Engineering	Synthetic Monitoring	Container Network Security
	Quality Management	Live Coding	Compatibility Testing	Dependency Proxy	IAST	Release Governance	Runbook Configuration	Incident Management	
					RASP		Cluster Cost Optimization	Status Page	

GitLab

Where is your team today?

What is CI/CD?

Continuous Integration (CI)

Automated testing and artifact creation (build)

Continuous Delivery (CD)

Automated deployment to test and staging environments
Manual deployment to Production

Continuous Deployment (CD)

Automated deployment to Production

Why…? Getting it right matters

High-performing teams deliver more, faster and cheaper

200X

More frequent deployments

29%

More time on new work

2,555X

Shorter lead times

22%

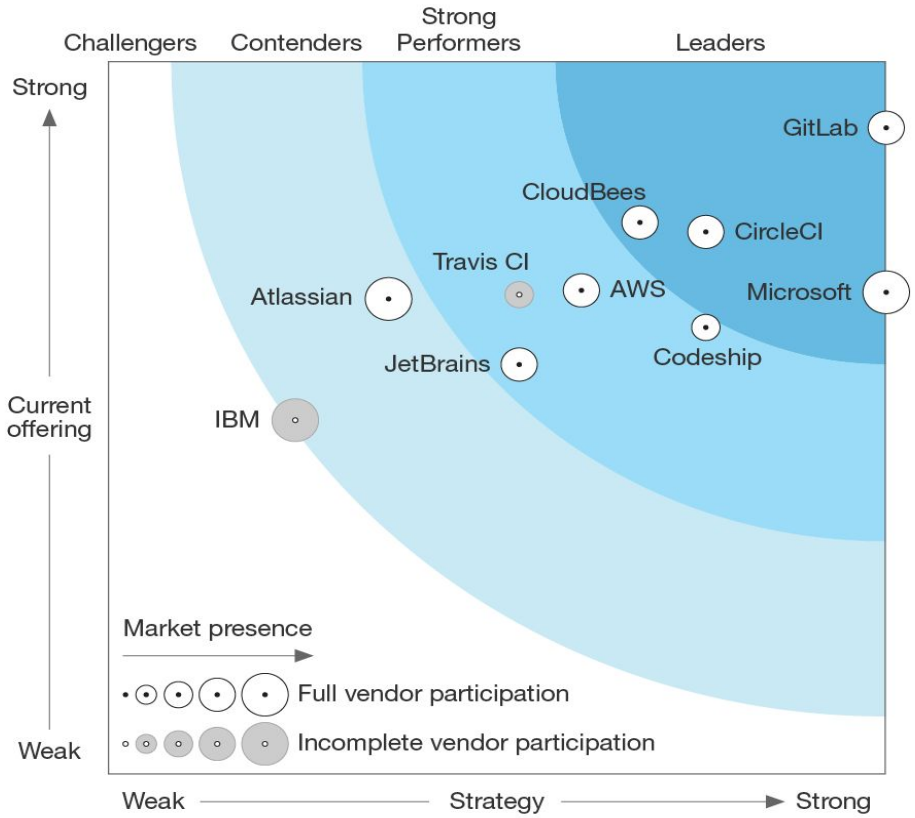
Less time on rework

2016 State of DevOps Report

GitLab - Leader in Forrester CI Tools Wave™

"GitLab's vision is to serve enterprise-scale, integrated software development teams that want to spend more time writing code and less time maintaining their tool chain."

- The Forrester Wave™: Continuous Integration Tools, Q3 2017 report

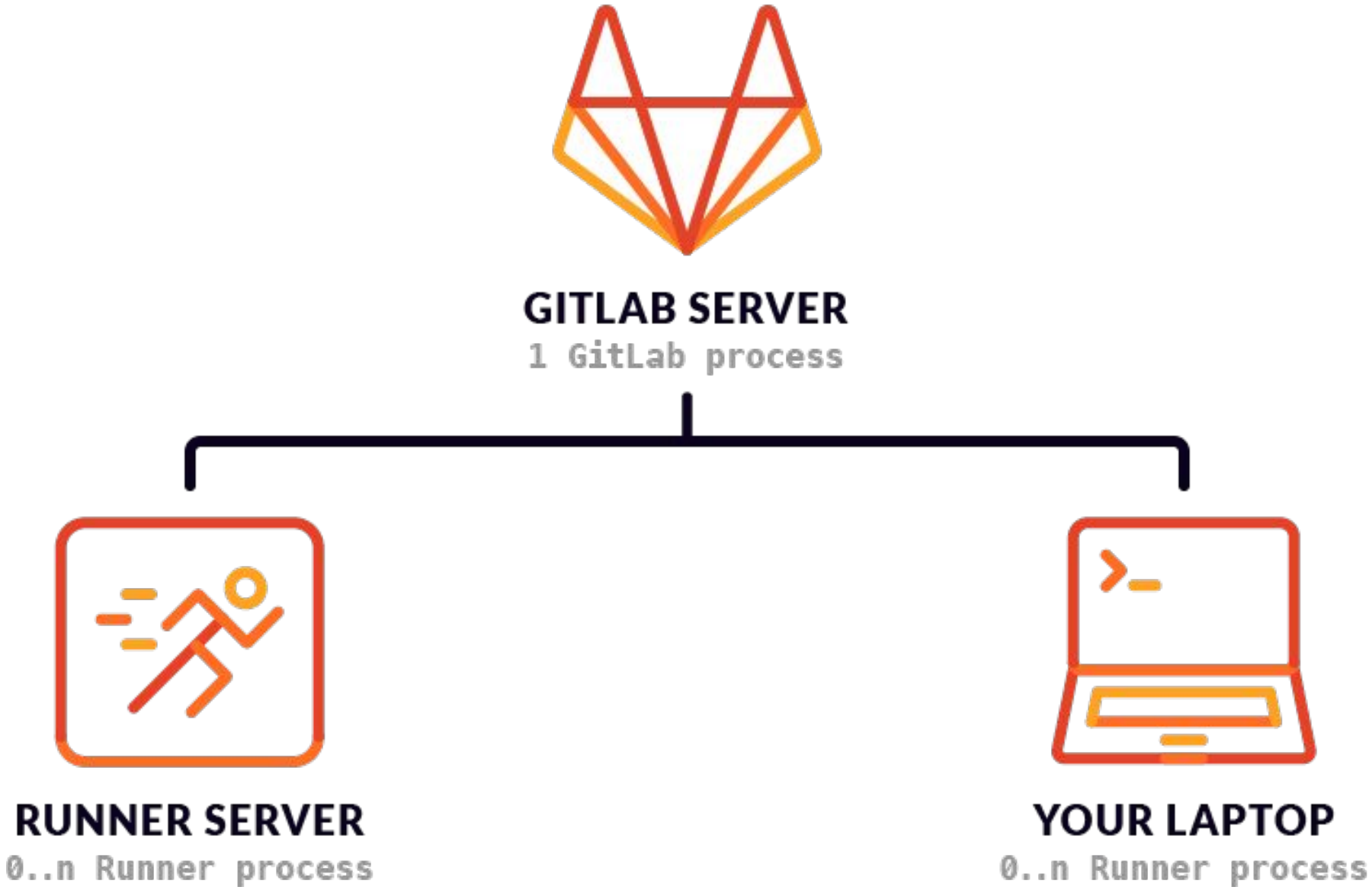


https://get.gitlab.com/forrester-wave-ci/

- **Highest score overall** for **Current Offering**
- **Highest possible** score for **Strategy**
- Top scores in Ease of Installation/Configuration, Configuring Builds and Build Reuse, Platform Support, Security Features, Container Build Support, Container Runtime Support, GUI, Analytics

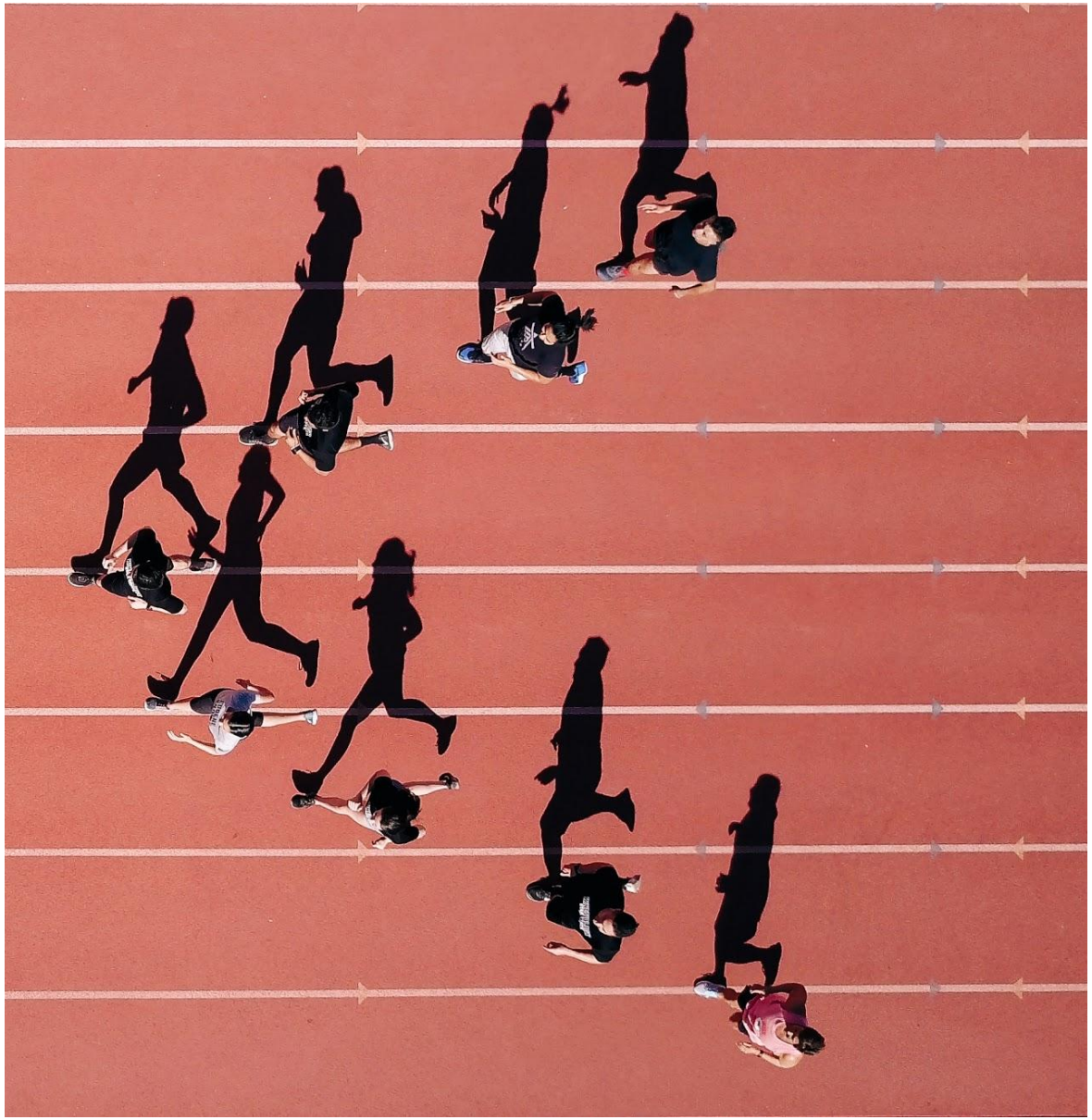
GitLab CI/CD Architecture

CI/CD Architecture



GitLab Runners

- Multi-platform
- Multiple environments
- Parallel builds
- Built for docker
- One install ➡️ many runners
- Pooled model for job execution, with exceptions
- Easier to set up and manage than slave machines



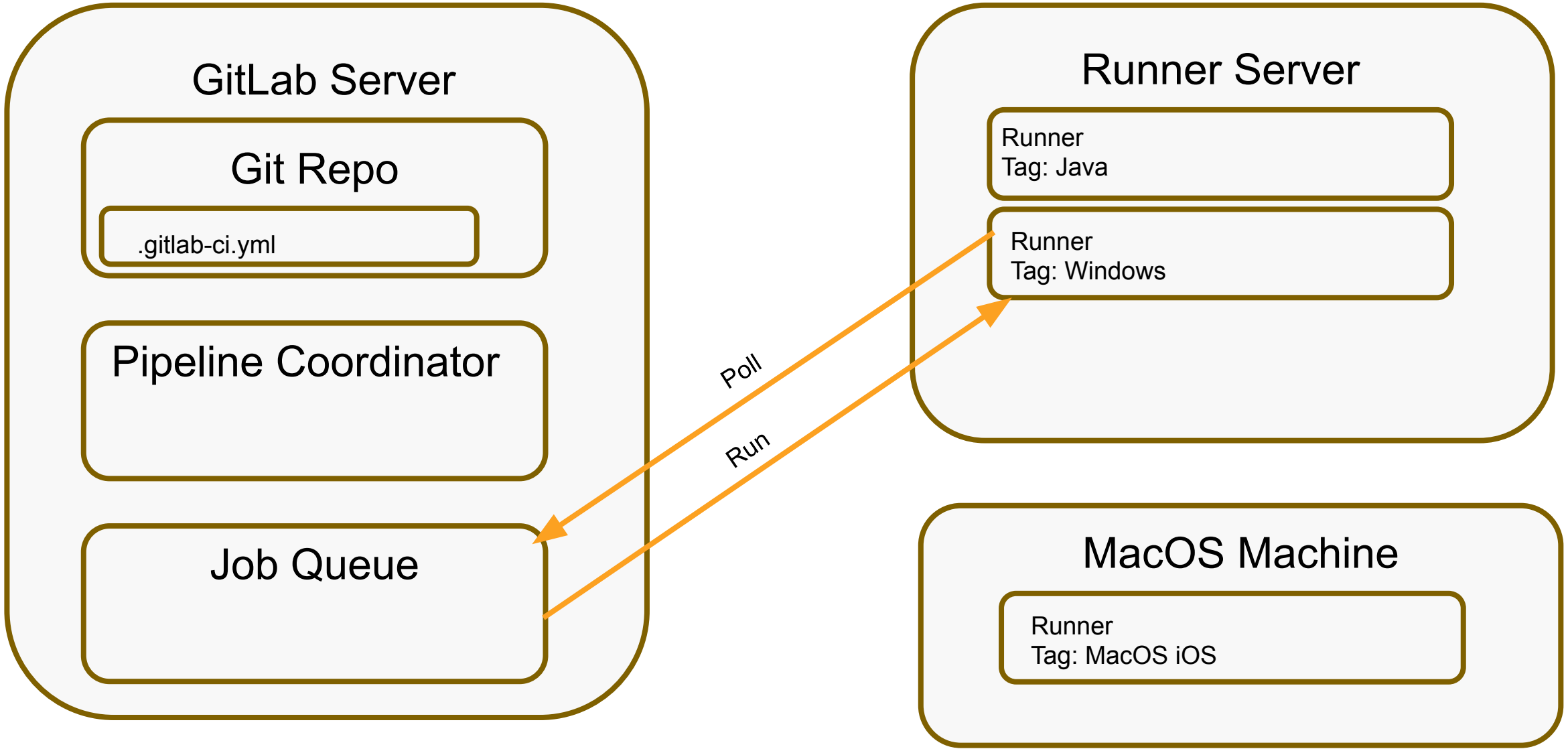
A runner can be...

Shared
or
Specific

Tagged
or
Untagged

Protected
or
Not Protected

GitLab CI/CD Architecture Detail



Shared

- Can be used by any project
- Included in pool for ALL projects
- Managed by GitLab Admin
- Typically auto scaling or otherwise scaled

Specific

- Tied to one or more specific projects
- In pool for ONLY specific projects
- Managed by Runner Owner(s)
- Typically for specialized builds, or if an org needs to do so for billing

Tagged

- Only used to run jobs tagged with **same** tag

Untagged

- Used to run jobs with **no** tags

Protected

- ONLY runs jobs from
  - Protected Branches
  - Protected Tags
- Typically used for runners containing deploy keys or other sensitive capabilities

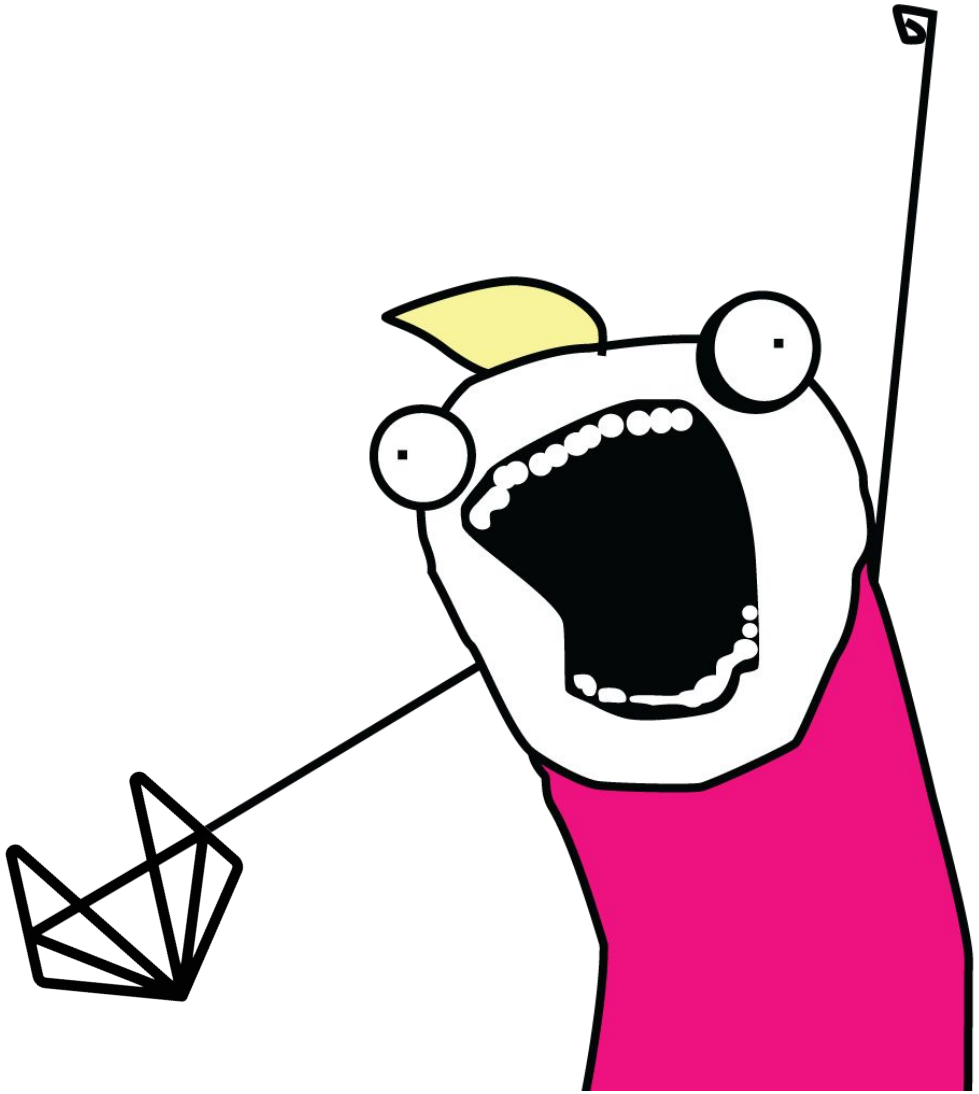
Not Protected

- Runs jobs from ANY branch
- Used for ANY build

Runner Platforms & Executors

Platforms

- Linux
  - Debian/Ubuntu/CentOS/RedHat
  - ANY
- macOS
- Windows
- Docker service
- Docker machine
- FreeBSD
- Kubernetes



Executors

Executors support different platforms and methodologies for building your code

- 🍳 Shell
- 🔐 SSH
- 🐳 Docker
- 🐳 Docker Machine (auto-scaling)
- 🛳️ Kubernetes
- 💻 Parallels
- 🕶️ VirtualBox

Runner Auto Scaling methods

AWS

- Many active examples
- EC2 Spot Instances

Docker+Machine

- Works with most cloud providers and many private cloud solutions
- Docker gives Digital Ocean and AWS examples
- Drivers also listed for:

AWS, Azure, GCP, DO, Exoscale, Hyper-V, OpenStack, Rackspace, IBM Softlayer, VirtalBox, VMWare vCloud Air, VMWare Fusion, VMWare vSphere

Kubernetes

- Executor and a scaling method
- Allows you to spin up a pod-per-job
- K8s ConfigMap

GitLab

GitLab Pipeline Definition

Anatomy of a GitLab CI/CD build

Pipeline

- Set of one or more jobs. Optionally organized into stages

Stages

- Collection of jobs to be run in parallel
- e.g. Build, Test, Deploy

Jobs

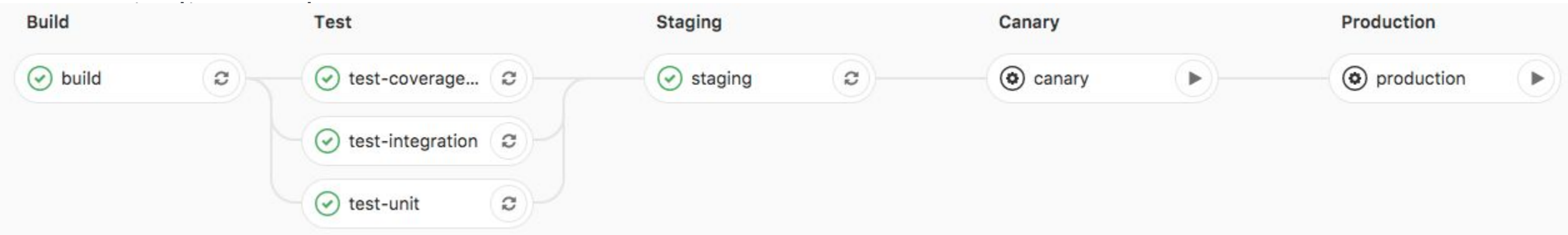
- Scripts that perform tasks
- e.g. npm test; mvn install; etc.

Environments

- Where we deploy (Test, Review, Staging, Canary, Prod)

All in one file - .gitlab-ci.yml

What does it look like?



GitLab CI/CD Basics - The .gitlab-ci.yml file

- YAML syntax
- Stored in root of project repo
- Version controlled along with the rest of your code
  - Enables (and enforces) different configurations for different branches
  - Allows entire DevOps team to participate in pipeline definition
- Contains
  - Jobs
  - Stages
  - Environments

- Other contents
  - Include
  - Image
  - Services
  - Before & After Scripts
  - Caching
  - Artifacts & On Success
  - Only & Except
  - When

GitLab

.gitlab-ci.yml examples

Simple `bash` .gitlab-ci.yml example

```
before_script:
  - echo "Before script section"
  - hostname
  - uname -a

after_script:
  - echo "After script section"
  - echo "For example you might do some cleanup here"

build1:
  tags:
      - steevo
      - macos
  stage: build
  script:
      - echo "Do your build here"

test1:
  tags:
      - steevo
      - macos
  stage: test
  script:
      - echo "Do a test here"
      - echo "For example run a test suite"

test2:
  tags:
      - steevo
      - macos
  stage: test
  script:
      - echo "Do another parallel test here"
```


deploy1:

Simple `bash` .gitlab-ci.yml example

.gitlab-ci.yml 677 Bytes

Lock Edit Web IDE Replace Delete

```yaml
before_script:
  - echo "Before script section"
  - hostname
  - uname -a

after_script:
  - echo "After script section"
  - echo "For example you might do some cleanup here"

build1:
  tags:
    - steevo
    - macos
  stage: build
  script:
    - echo "Do your build here"

test1:
  tags:
    - steevo
    - macos
  stage: test
  script:
    - echo "Do a test here"
    - echo "For example run a test suite"

test2:
  tags:
    - steevo
    - macos
  stage: test
  script:
    - echo "Do another parallel test here"
    - echo "For example run a lint test"

deploy1:
  tags:
    - steevo
    - macos
  stage: deploy
```

Containers and Services gitlab-ci.yml example

.gitlab-ci.yml 693 Bytes

Lock Edit Web IDE Replace Delete

```yaml
# Official framework image. Look for the different tagged releases at:
# https://hub.docker.com/r/library/node/tags/
image: node:latest

# Pick zero or more services to be used on all builds.
# Only needed when using a docker container to run your tests in.
# Check out: http://docs.gitlab.com/ce/ci/docker/using_docker_images.html#what-is-a-service
services:
  - mysql:latest
  - redis:latest
  - postgres:latest

# This folder is cached between builds
# http://docs.gitlab.com/ce/ci/yaml/README.html#cache
cache:
  paths:
  - node_modules/

test_async:
  script:
   - npm install
   - npm run test

.test_db:
  script:
   - npm install
   - node ./specs/start.js ./specs/db-postgres.spec.js
```

New gitlab-ci.yml example file from template

GitLab | Projects | Groups | Activity | Milestones | Snippets | Search or jump to... | 20 | 29

S sg-spring

Project

Repository

Files

Commits

Branches

Tags

Contributors

Graph

Compare

Charts

Locked Files

Issues 3

Merge Requests 1

Midwest Apps › sg-spring › Repository

New file | Template | .gitlab-ci.yml | Auto-DevOps | Template applied | Undo

master / .gitlab-ci.yml | Soft wrap | text

```
1  # This file is a template, and might need editing be
2  # Auto DevOps
3  # This CI/CD configuration provides a standard pipel
4  # * building a Docker image (using a buildpack if ne
5  # * storing the image in the container registry,
6  # * running tests from a buildpack,
7  # * running code quality analysis,
8  # * creating a review app for each topic branch,
9  # * and continuous deployment to production
10 #
11 # Test jobs may be disabled by setting environment v
12 # * test: TEST_DISABLED
13 # * code_quality: CODE_QUALITY_DISABLED
14 # * license_management: LICENSE_MANAGEMENT_DISABLED
15 # * performance: PERFORMANCE_DISABLED
16 # * sast: SAST_DISABLED
17 # * dependency_scanning: DEPENDENCY_SCANNING_DISABLED
18 # * container_scanning: CONTAINER_SCANNING_DISABLED
19 # * dast: DAST_DISABLED
20 # * review: REVIEW_DISABLED
21 # * stop_review: REVIEW_DISABLED
22 #
23 # In order to deploy, you must have a Kubernetes cluster configured either
24 # via a project integration, or via group/project variables.
25 # AUTO_DEVOPS_DOMAIN must also be set as a variable at the group or project
26 # level, or manually added below.
27 #
```

Apply a template

Filter

General

Android

Auto-DevOps

Bash

C++

Chef

Clojure

Auto DevOps - The Easiest Way to Set Up A Pipeline

Simply commit code and GitLab does the rest

Auto DevOps . . .

- Detects the language of the code
  - Builds with a dockerfile if there is one
  - Uses Heroku and herokuish build packs if there isn't
    - Build packs analyze the code in the project and figure out the best way to build based on *convention not configuration*.
- Automatically builds, tests, and measures code quality
- Scans for security and licensing issues
- Packages
- Instruments (for monitoring)
- And deploys the application

Getting Started with CI/CD Yourself

- All of the capabilities discussed tonight are available in the no-cost versions of GitLab
  - GitLab CE - community edition, open source, good for personal projects
  - GitLab EE Core - a good place to start on projects for work at no charge
  - Gitlab.com - the free tier here also has CI/CD but with limited runner minutes
    - Create your own local runners to avoid the runner minutes limitation
- https://about.gitlab.com/product/continuous-integration/

GitLab is the single tool for the entire DevOps Lifecycle

- We’ve only talked about one part of GitLab tonight
- We cover everything from
  - Agile Planning and Portfolio Management
  - Git Repo, Branching, Merge Request Details
  - CI/CD
  - Security Scanning
  - Security Defense (emerging)
- https://about.gitlab.com
-

Thank You!