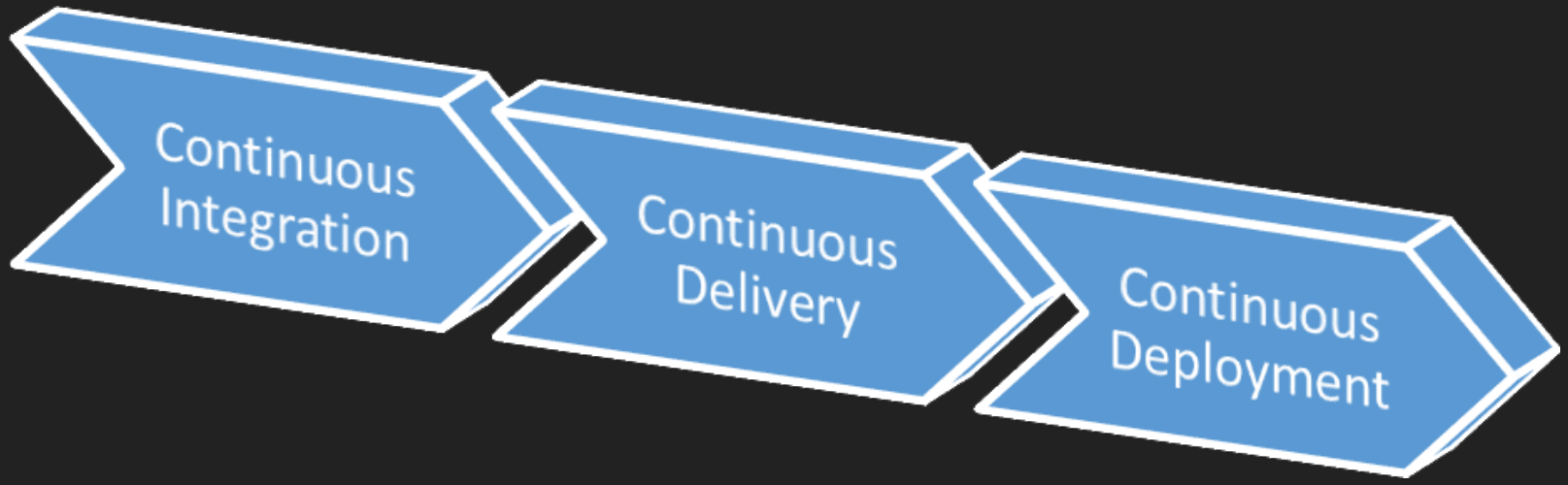
HOW-TO DEVOPS



- Welcome "How to DevOps"
- high-level overview
- continuous integration, delivery, deployment
- applicable for all projects

AN INTRODUCTION TO CI/CD

But, what is DevOps?

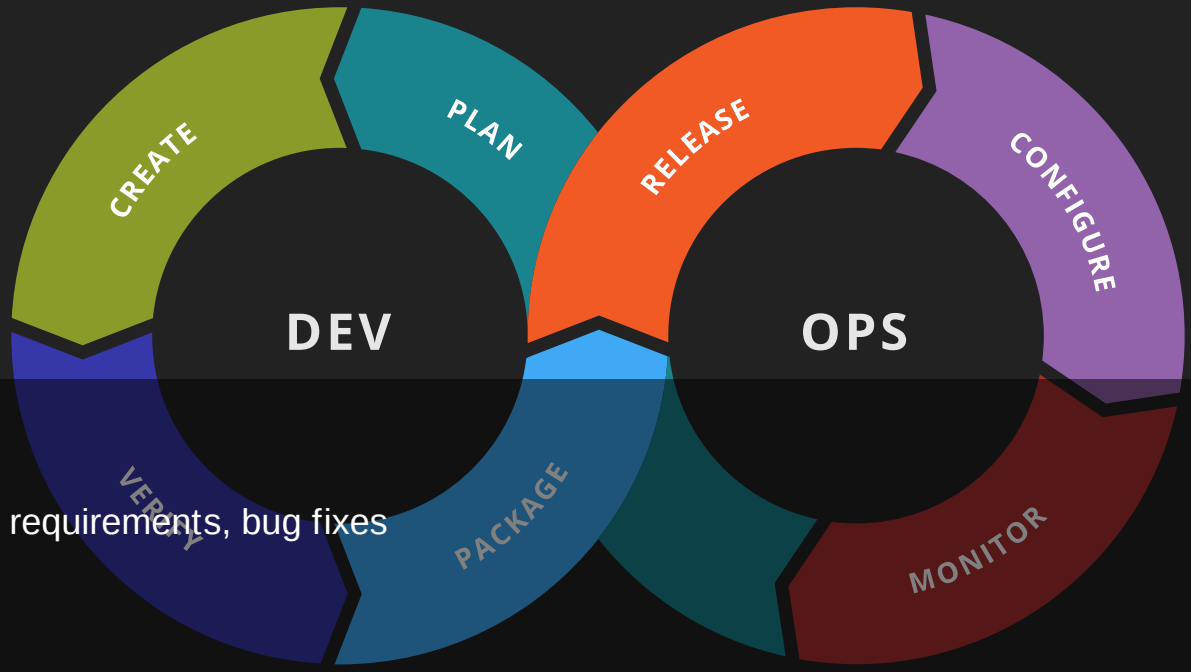
OBJECTIVES

- High-level Overview (What & Why)
- Components (How)
- Demo!

- high-level, whats and whys
- parts and how to put together
- light-weight example

DEVOPS

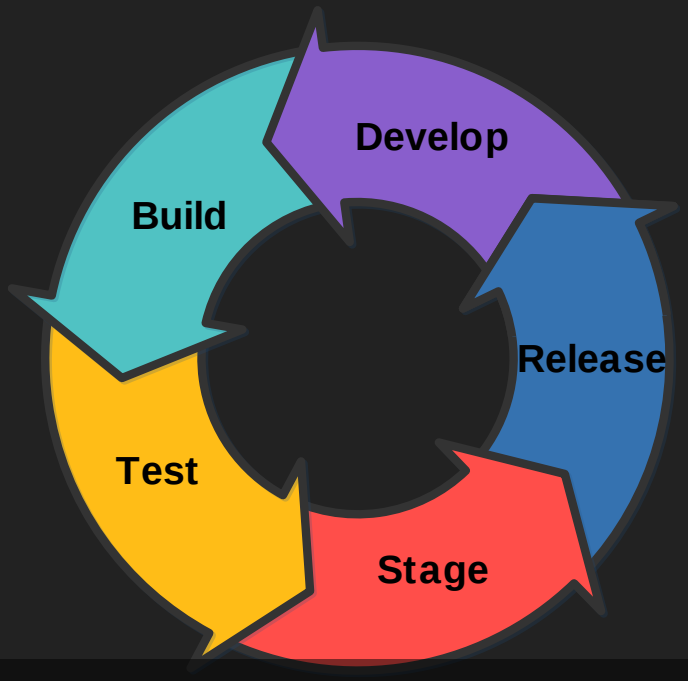
Integrate developer and operation teams to improve collaboration and productivity



- Collaboration teams
- idea to production.
  - enhancements, requirements, bug fixes
- DevOps includes
  - Culture
  - Philosophy
  - Different Processes
- CI/CD helps us do the DevOps.

CI/CD

Continuous
Integration / Delivery / Deployment



Practice to improve software development

WHO IS CARLOS MEZA

SYSTEMS ADMINISTRATOR

- Works on the EdgeCast CDN at Verizon Digital Media Services
- Configuration Management

 @DIGITALR00TS

- EdgeCast/VDMS
- Salt, Chef - started doing CI/CD
- Twitter - slides

DISCLAIMER

- Views and opinions expressed are of my own and not of my employer or any other entity.
- Third party content used is under "Fair Use" as this intended for educational purposes.
- Logos and trademarks belong to their respective holders. Use of them does not imply any affiliation with or endorsement by them.

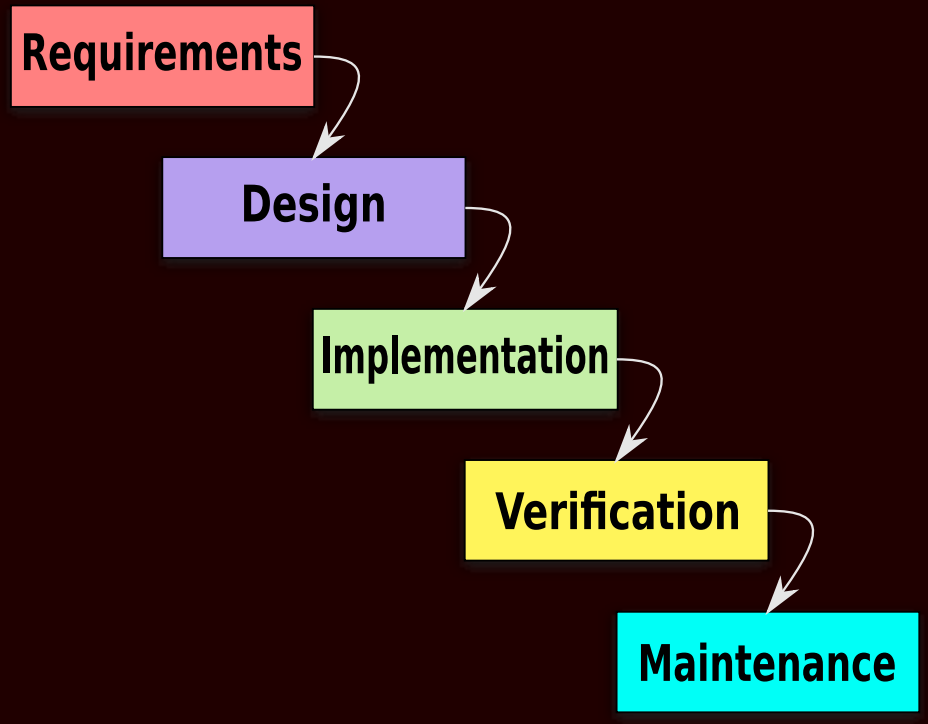
- My opinion
- Fair Use
- Does not imply endorsement

TRADITIONAL

Anti-patterns

- done traditionally

WATERFALL MODEL



- sequential (non-iterative) design process
- pass down the chain
- w/o good communication back up
- throw over wall to Ops

DEV VS OPS

- Old adage of, "It worked in Dev"
- So as Ops, I am interested in doing the DevOps

INFREQUENT LARGE CHANGES

- Higher risk
- More complex to debug
- More time to production
- Late feedback on impact

- Thought to be safer not to touch production often
- changes accumulate

MANUAL PROCESSES

(builds, tests, etc)

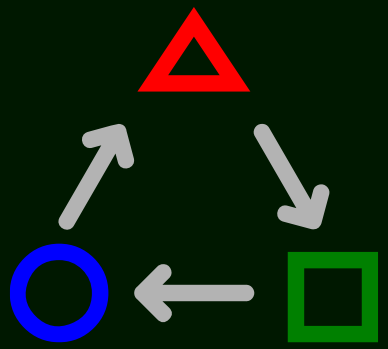
- Inconsistent
- Error Prone
- Slow

manual
builds, test, deployments

- cannot reliable reproduce
- error prone
- slower

CI / CD

A software development practice intended to produce better code faster.

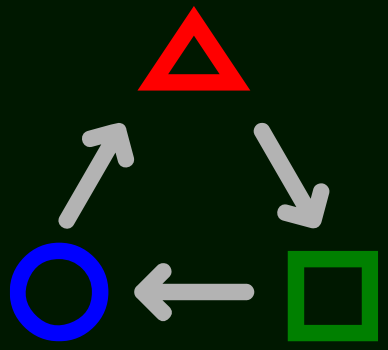


Solution - CI CD

- Development practice
- to produce better code faster

CI / CD

By shipping *small and frequent changes* with *automated* builds, tests, and reports.



- small and frequent changes
- automated

SMALL FREQUENT CHANGES

- Modular, less complex code
- Integration is easier
- Easier to respond to changing demands

- easier to debug and manage
- frequent integration becomes easier
- more able to adjust changes
  - implantation, priorities, etc

AUTOMATION

- Consistency
- Enforced testing
- Immediate feedback

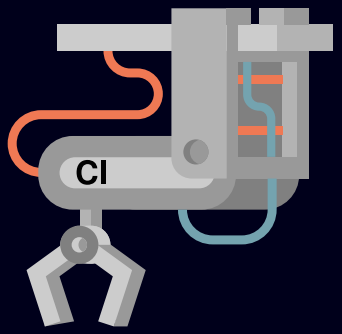
- Same process;
  - avoid mistakes and shortcuts
- tests are not skipped
- quickly find out impact
  - of shipped code

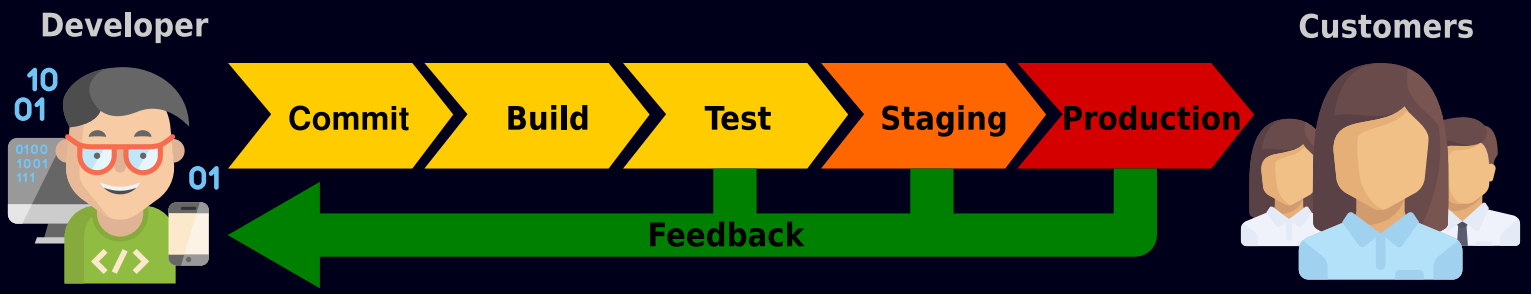
OTHER BENEFITS

- Reduced risk
- Greater visibility into the process
- Faster to production

- Consistency / Less error prone
- Higher confidence in deployment
- Visibility into the status and evolution
- Faster ROI

PIPELINE





- yellow CI
- orange CD
- red CD - to customers
- feedback at every step of the process/workflow

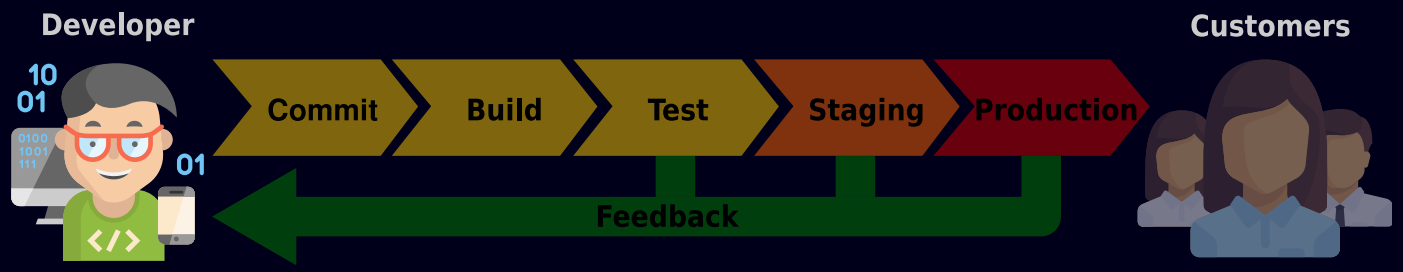
PIPELINE COMPONENTS

- parts of workflow
- tools/software

PROJECT CODE

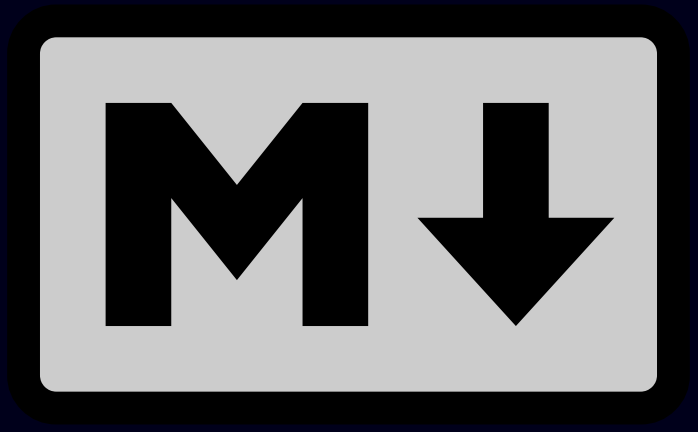
Code can represent:

- Web application
- Infrastructure
- Documentation



- anything that can be represented as code

PROJECT CODE

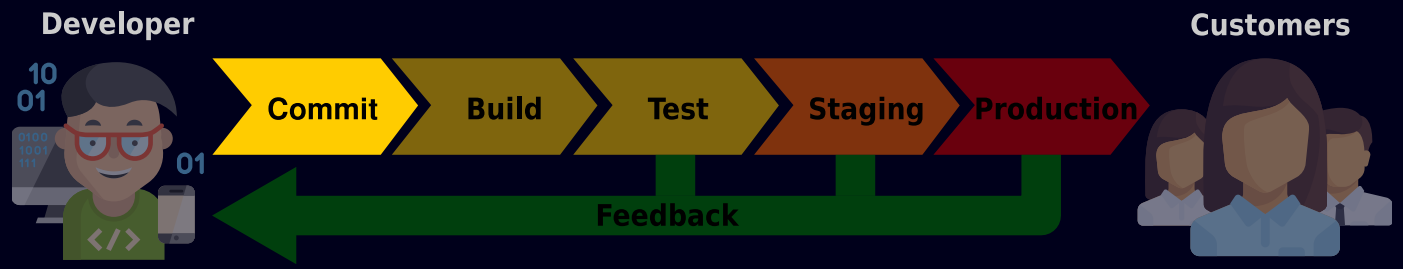


Markdown + Jekyll

- lightweight markup language
- that Jekyll can process
- popular, open-source
- static site generator

VERSION CONTROL SYSTEM

Manage changes to source code over time.



Version Control System

- Record
- track
- over time

VERSION CONTROL

Git + GitLab

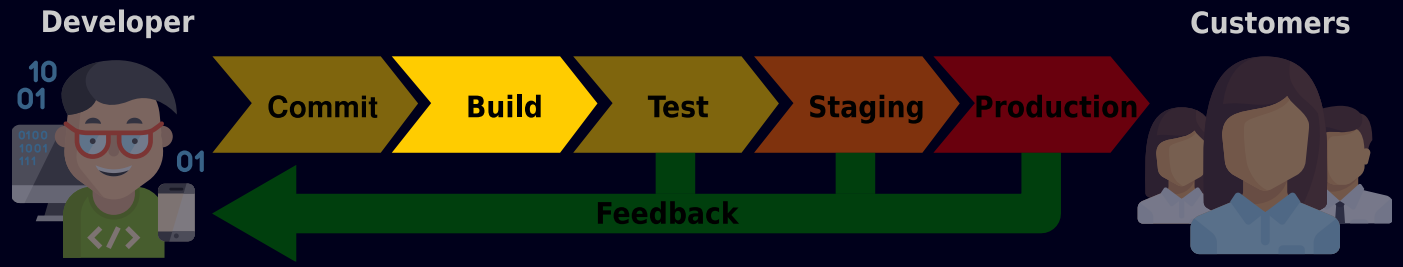
- Git - open source version control system
- GitLab - open source web interface, freemium hosted

Other free hosted options include:

- GitHub, Bitbucket

BUILD

Compile the project code



- Generate the product
- hope to ship

BUILD

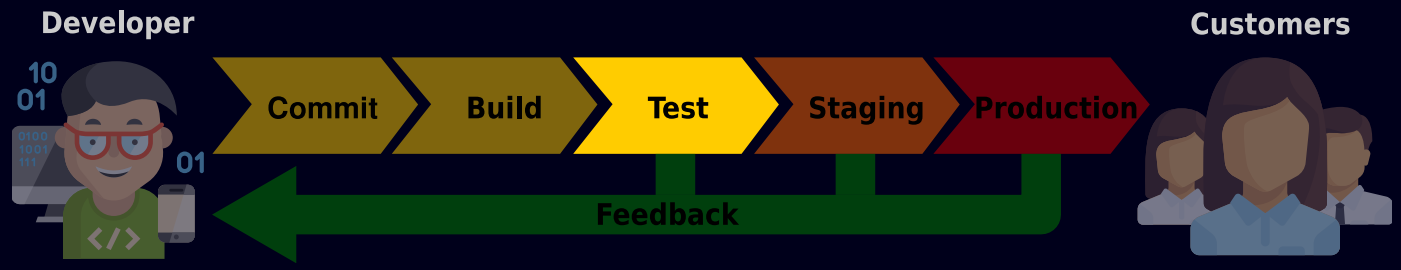
Jekyll

- Jekyll generates our site

TESTS

Verification of code

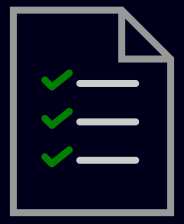
- Unit
- Integration
- System



- Automation needs testing
- Many categories/types

- Unit - checks the feature being working on
- Integration - checks code change works with rest of project
- System - works with external dependencies, e.g. hw

TESTS

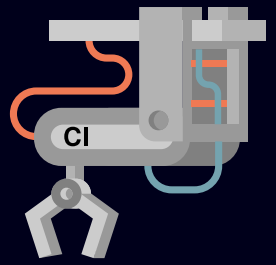


- Jekyll
- HTML-Proofer
- Hunspell

- check configuration sane and builds
- syntax, images exists, links
- spell-check

CI SOFTWARE

Orchestrates builds, tests, and deployments.



- CI software manages jobs to automate:
  - builds, test, deployments

CI SOFTWARE

GitLab CI

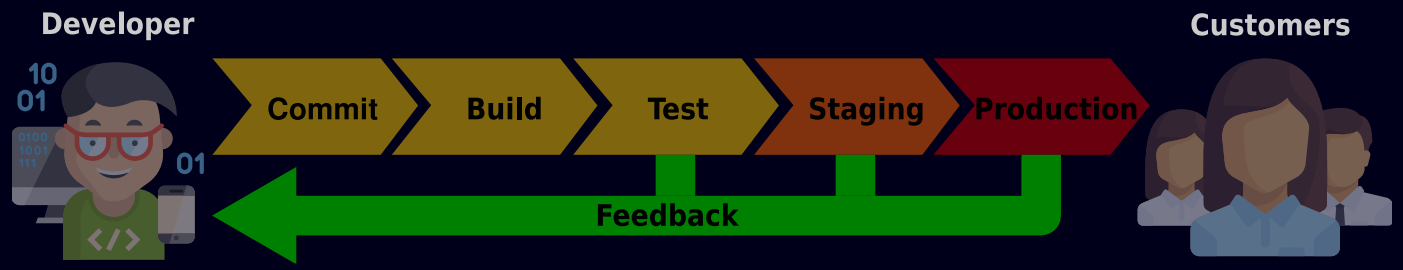
GitLab\s a continuous integration service.

Other popular CI solutions include:

- Jenkins
- Travis

NOTIFICATIONS

Provides feedback on successfulness of tests, builds, and deployments.



- Feedback loop
- info comes to us

NOTIFICATIONS

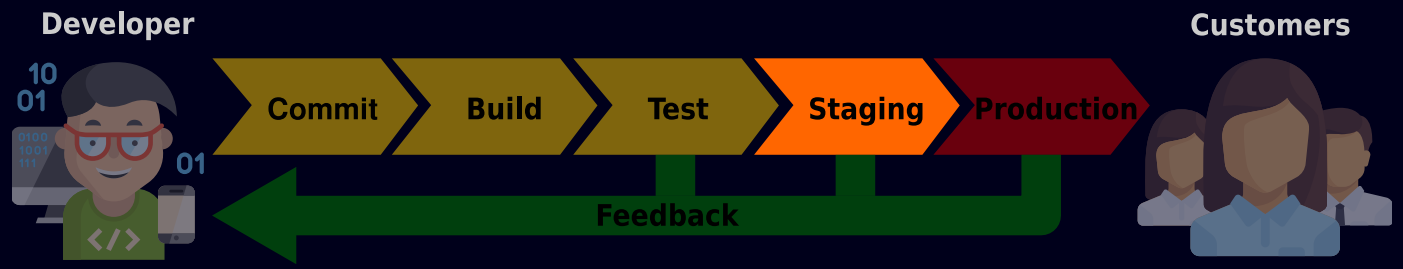
Slack

Popular freemium hosted chat service.

- Email
- IRC
- SMS

STAGING ENVIRONMENT

Delivers a release to a pre-production environment.



- production-like
- see site before it published

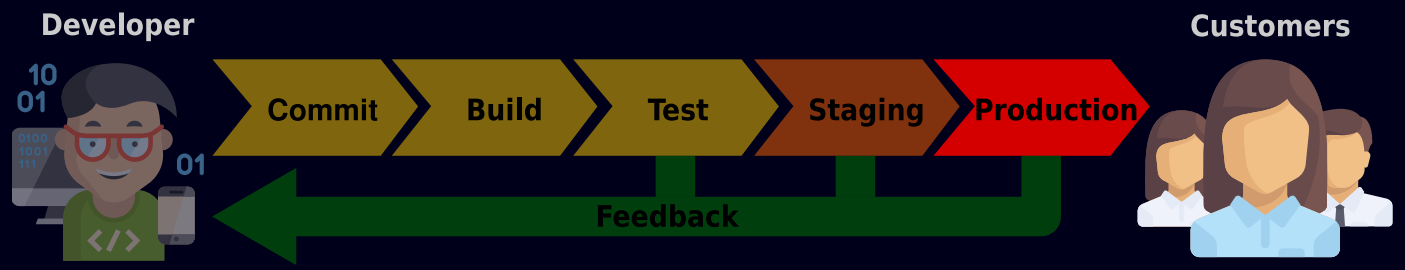
STAGING ENVIRONMENT

GitLab Pages

Host static websites on GitLab.com

PRODUCTION ENVIRONMENT

Deploy code to customers.



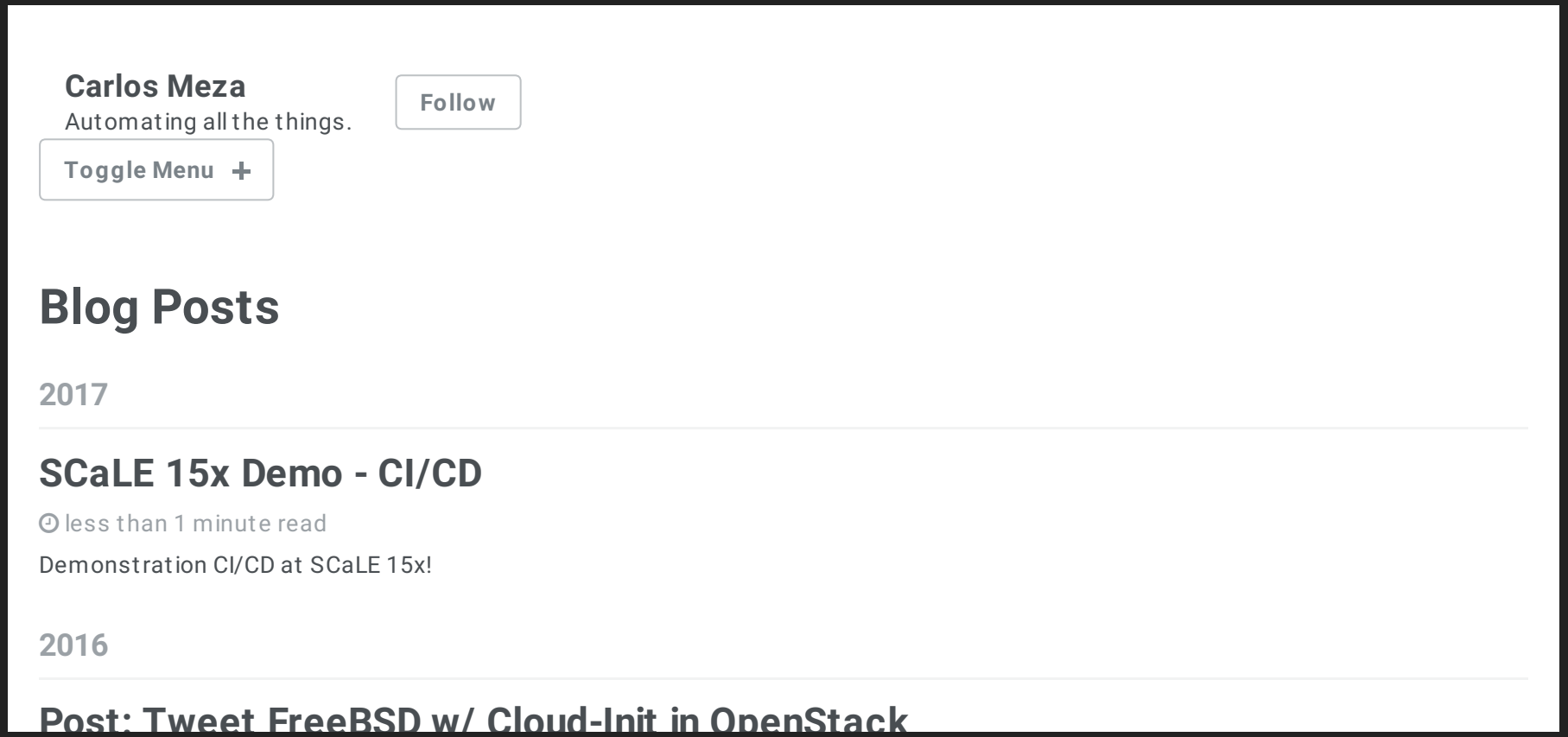
Get code to your customer

PRODUCTION ENVIRONMENT

EdgeCast CDN

- Content delivery network

DEMO



https://gitlab.com/DIGITALR00TS-Demo/ci-jekyll/

You can do this yourself by creating your own fork on GitLab.

CI/CD - WHAT

- A software development practice
- Small frequent code changes
- Automated processes

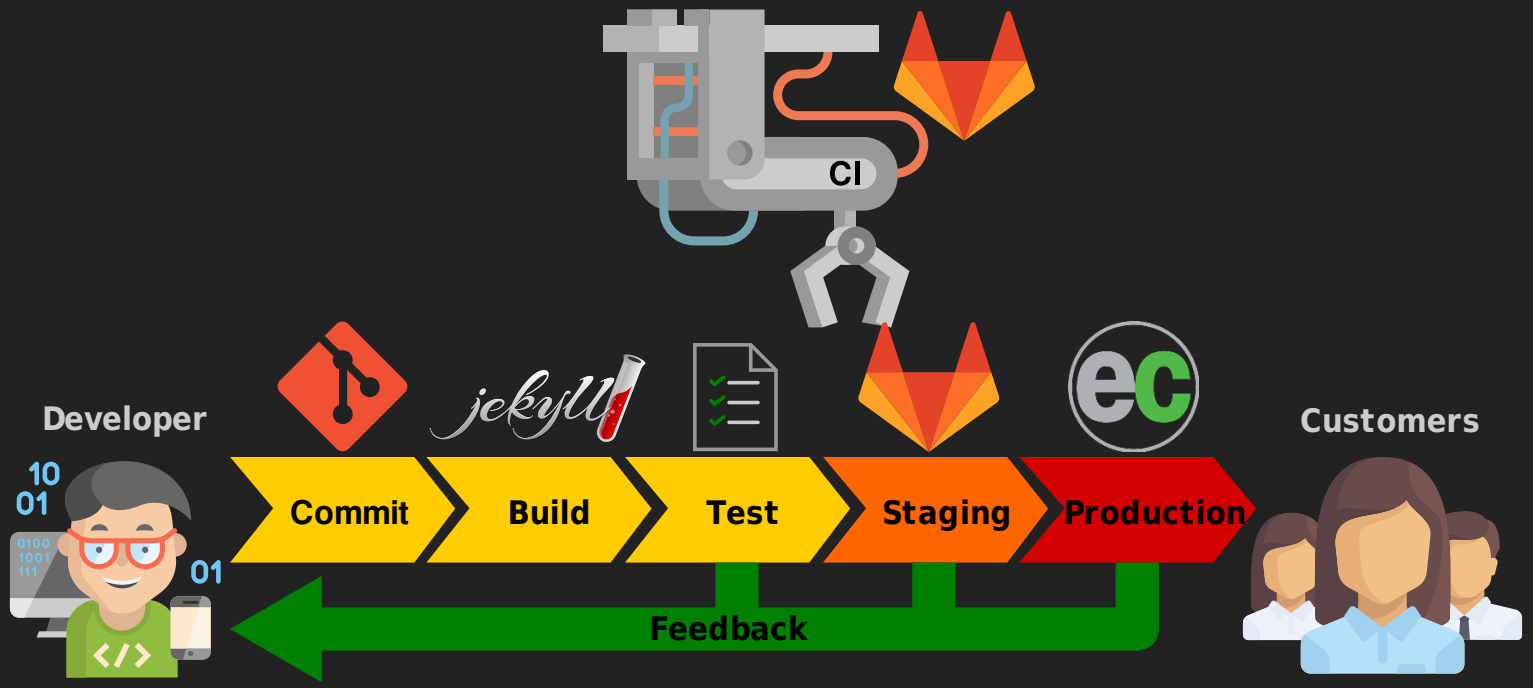
- development workflow
- Check in small changes frequently
- Automated builds, tests, deployments

CI/CD - WHY

- Faster ROI
- Higher confidence in releases
- More agile

- Get to production faster
- Consistent processes, testing, feedback
- short, frequent iteration, better suited to adjust changing demands

CI/CD - HOW



- Tool chain created

LINKS

- ThoughtWorks - CI
  https://www.thoughtworks.com/continuous-integration
- Agile Alliance Glossary
  https://www.agilealliance.org/agile101/agile-glossary/
- DevOps For Dummies(pdf)
  http://www.ibm.com/ibm/devops/us/en/resources/dummiesbooks/

LINKS - VCS / GIT

- What is Version Control
  https://www.atlassian.com/git/tutorials/what-is-version-control
- Git - Getting Started
  https://git-scm.com/book/en/v1/Getting-Started
- Git Cheat Sheet(PDF)
  https://services.github.com/resources/index.html

LINKS - VIDEOS

- What is DevOps? - In Simple English
  https://www.youtube.com/watch?v=_I94-tJlovg
- What is Version Control?
  https://git-scm.com/video/what-is-version-control
- Martin Fowler – Continuous Delivery
  https://www.youtube.com/watch?v=aoMfbgF2D_4

In case you don't like reading like me.

THANK YOU

@DIGITALR00TS

QUESTIONS?