1 CheatSheet: DevOps & Software Development Life Cycle Tool

- PDF Link: cheatsheet-devops-A4.pdf, Category: tool
- Blog URL: `https://cheatsheet.dennyzhang.com/cheatsheet-devops-A4`
- Related posts: CheatSheet: Container Compliance, Cheatsheet: Linux Security #denny-cheatsheets

File me Issues or star this repo.

1.1 Overview

Name	Summary
`Automate, Automate, Automate`	Avoid manual effort and reduce process delay
`Unify` the ways of doing the same thing	Build code, test features, etc. Thus reduce invalid failures
Enable `faster iteration` by reducing dependencies	Implement or leverage mockup services
Definition of unit tests	Unit tests focus on function-level scope
Definition of integration tests	Does integration tests include setup an entire system to validate?
Definition of e2e tests	
Reference	CheatSheet: Container Compliance
Reference	CheatSheet: DevOps & Software Development Life Cycle

1.2 Use container for CI/CD

Name	Summary
Dockerd hang issue	docker build image hang; docker stop container hang; docker prune hang;
Make sure dockerd up-to-date	dockerd might have known issues or improvements
More CPU/RAM for dockerd machine	machine is overloaded
Use ephemeral dockerd machines for testing	cats vs cattle
Dockerd storage driver: overlay2	
Free disk of /var/lib is slow	

1.3 Network dependency download flaky

Name	Summary
Mirror the external repos	
Add retry for downloading	

1.4 Code Build

Name	Summary
Unify different ways of building code	`docker build` vs `make/mvn/gradle`
List versions & enforce prelight check	List versions and dependencies, thus people can easily debug
Reduce pulling dependencies from internet/GitHub	Modify iptables rules in the build box to enforce it
Automatically recommend build after validation	

1.5 Pre-Merge Hook

Name	Summary
Pre-merge tests help to limit trouble shooting to individuals	This reduce cross-team join-effort debugging
Code lint checks	
Unit test/BDD tests	
Monitor test coverage	Test coverage includes unit test and integration test suites

1.6 Component Tests

Name	Summary
Move tests closer to develoers' machine	
Utilize mockup services	
Test component individually	
Test component quickly	

1.7 Integration Tests

Name	Summary
Define lightweight and heavyweight e2e tests	
Examine log for errors/exceptions, even for successful builds	
Airgap deployment and tests	
Track the data of successful builds	logs, timespan for each stage, etc
Testbed management for complicated projects	vanilla testbed validator
Enforce nightly builds and file tickets for each new failure	
Seperate dependencies failures from code issues	Document the failures from unmanaged dependencies
Examine build/test pipelines for performance tuning	Slowness may from test workflow or products themselves

1.8 Additional Tests

Name	Summary
Test categories	Chaos/Longevity tests, Workload tests, Upgrade tests, Scale tests

1.9 Speedup process

Name	Summary
Speed up the review and merge of pull requests	
Reminders for team members	Regular meeting; regular activities

1.10 DevSecOps

Name	Summary
Enforce compliance check for OSS packages	Avoid legal issues and security vulnerability
Log scan for security compliance	
Don't save credentials in code repo	
Don't display credentials in log files	

1.11 Metrics-Driven Quality Control

Name	Summary
Key Metrics	Pass rate for acceptance e2e pipeline; Pass rate for code build pipeline

1.12 Product Debuggability

Name	Summary
Components can switch log level on fly	
Watch out log velocity and values	Log tons of useless entries would be annoying
Implement a docker tool: `health_check`	

1.13 More Resources