

Strong Foundations

Insightful Learning

VS

Trial-and-Error

in CI/CD

SPEAKER INTRO

Ixchel Ruiz



- Developer Advocate, JFrog

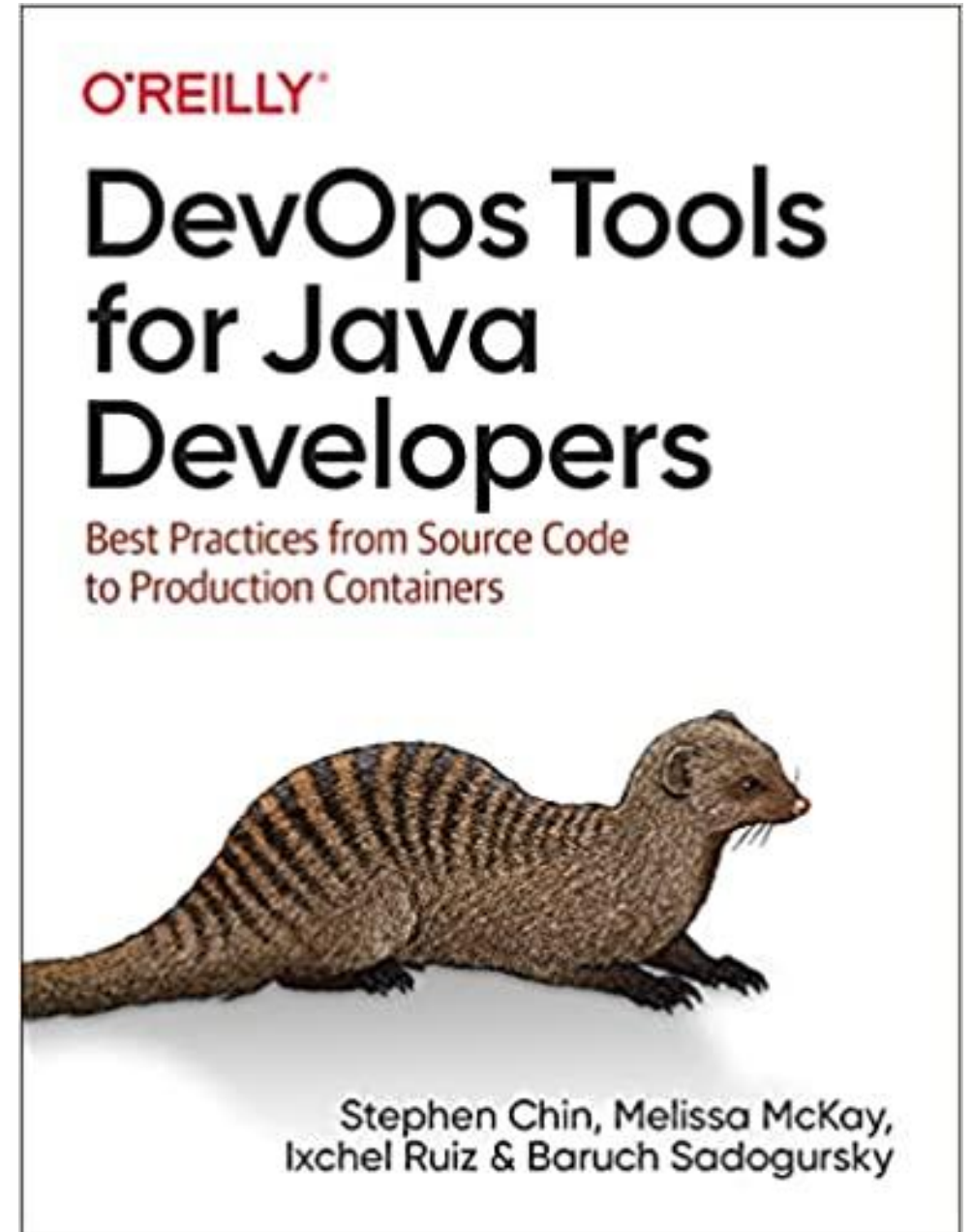
Melissa McKay



- Developer Advocate, JFrog
- Developer!
- Co-Chair of CDF Interoperability SIG
- Likes to play Pokemon Go

DEVOPS TOPICS

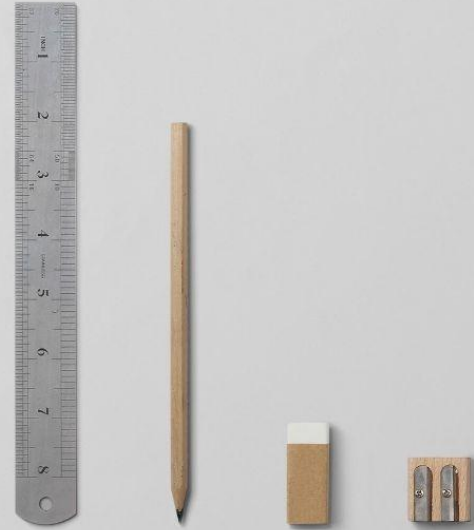
- DevOps Methodology
- Source Control
- Containers
- Microservices
- Continuous Integration
- Package Management
- Securing Your Binaries
- Deployment



AGENDA

- The Initial Build of Your CI/CD Pipeline
- Developer vs DevOps
- Why DevOps?
- How To Measure Success
- Best Practices
- Why Not DevOps?
- Resources: Tried and True!

THE INITIAL BUILD OF YOUR CI/CD PIPELINE



DEVOPS & CI/CD TOPICS

- Dev Resources
- Code
- Source Control
- Build Server
- Security Scanning
- Artifact Management
- Cloud Resources
- Deployment Infrastructure
- Automation
- Observability Tools



App Definition & Development

Database: KV, V, etc. Streaming & Messaging: cloudevents, etc. Application Definition & Image Build: HELM, etc. Continuous Integration & Delivery: argo, flux, etc.

Orchestration & Management (circled in red)

Scheduling & Orchestration: Kubernetes, etc. Coordination & Service Discovery: etcd, etc. Remote Procedure Call: gRPC, etc. Service Proxy: envoy, etc. API Gateway: Kong, etc. Service Mesh: Istio, etc.

Runtime

Cloud Native Storage: MinIO, etc. Container Runtime: cri-o, etc. Cloud Native Network: Cilium, etc.

Provisioning

Automation & Configuration: Ansible, etc. Container Registry: Harbor, etc. Security & Compliance: Falco, etc. Key Management: Spiffe, etc.

Special

Kubernetes Certified Service Provider: etc. Kubernetes Training Partner: etc. Certified CNFs: etc.

Platform

Certified Kubernetes - Distribution: AWS, etc. Certified Kubernetes - Hosted: AWS, etc. Certified Kubernetes - Installer: AWS, etc. PaaS/Container Service: etc.

Observability and Analysis

Monitoring: Prometheus, etc. Logging: Fluentd, etc. Tracing: Jaeger, etc. Chaos Engineering: Litmus, etc. Continuous Optimization: etc.

Serverless

Member

CD Foundation Landscape

CLOUD NATIVE Landscape

CLOUD NATIVE COMPUTING FOUNDATION

Request Amplify

This landscape is intended as a map through the previously uncharted terrain of cloud native technologies. There are many routes to deploying a cloud native application, with CNCF Projects representing a particularly well-traveled path.

l.cncf.io

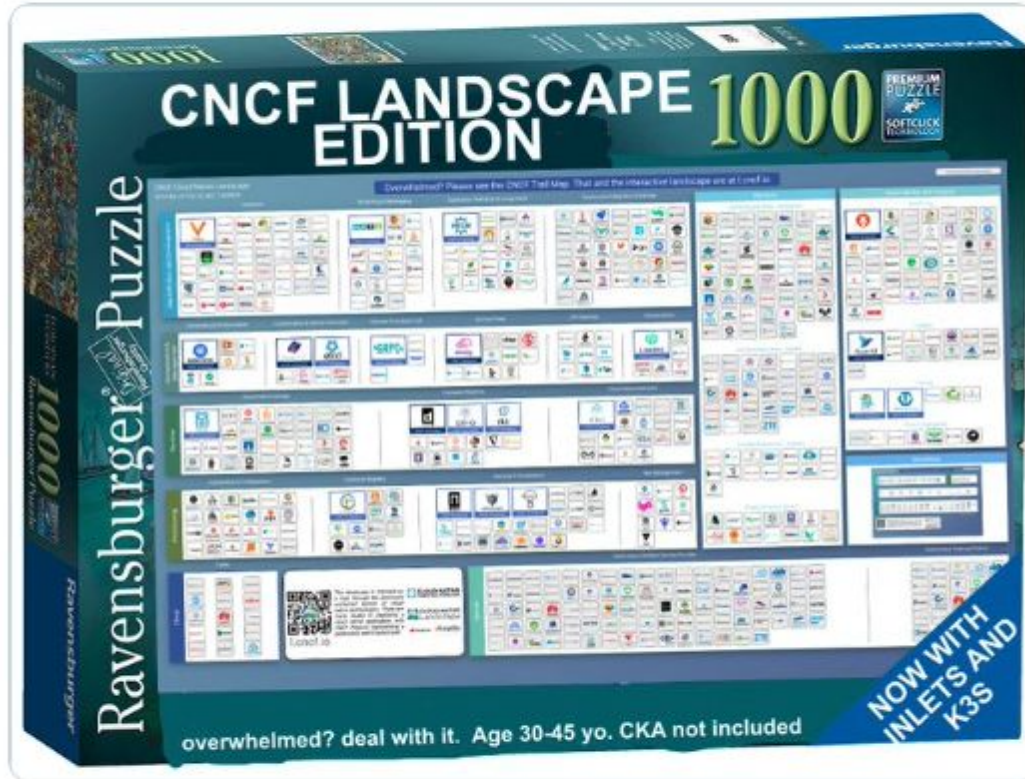




memenetes
@memenetes



Kid: I want a really difficult puzzle for my birthday
Me:



3:00 PM · Apr 7, 2020



1.8K Reply Copy link

Read 22 replies

urbandecoder ❤️

42 sales

CNCF Landscape Puzzle

\$34.99

Out of stock

1 shop review ☆☆☆☆☆

★★★★★

The puzzle to end all puzzles.

Purchased item: [CNCF Landscape Puzzle](#)



[tashy1019](#) Jan 4, 2021

Helpful?



CLOUD NATIVE TRAIL MAP

The Cloud Native Landscape <https://github.com/cncf/landscape> has a growing number of options. This Cloud Native Trail Map is a recommended process for leveraging open source, cloud native technologies. At each step, you can choose a vendor-supported offering or do it yourself, and everything after step #3 is optional based on your circumstances.

HELP ALONG THE WAY

A. Training and Certification

Consider training offerings from CNCF and then take the exam to become a Certified Kubernetes Administrator <https://www.cncf.io/training>

B. Consulting Help

If you want assistance with Kubernetes and the surrounding ecosystem, consider leveraging a Kubernetes Certified Service Provider <http://cncf.io/kcsp>

C. Join CNCF's End User Community

For companies that don't offer cloud native services externally <http://cncf.io/enduser>

WHAT IS CLOUD NATIVE?

- **Operability:** Expose control of application/system lifecycle.
- **Observability:** Provide meaningful signals for observing state, health, and performance.
- **Elasticity:** Grow and shrink to fit in available resources and to meet fluctuating demand.
- **Resilience:** Fast automatic recovery from failures.
- **Agility:** Fast deployment, iteration, and reconfiguration.

www.cncf.io
info@cncf.io

v1.0



1. CONTAINERIZATION

- Normally done with Docker containers
- Any size application and dependencies (even PDP-11 code running on an emulator) can be containerized
- Over time, you should aspire towards splitting suitable applications and writing future functionality as microservices

3. ORCHESTRATION

- Pick an orchestration solution
- Kubernetes is the market leader and you should select a Certified Kubernetes Platform or Distribution
- <https://www.cncf.io/ck>



kubernetes
CNCF Graduated

5. SERVICE MESH

- Connects services together and provides ingress from the Internet
- Service discovery, health checking, routing, load balancing
- Consider Envoy, Linkerd and CoreDNS



envoy **CoreDNS**
CNCF Incubating CNCF Incubating

7. DISTRIBUTED DATABASE

When you need more resiliency and scalability than you can get from a single database, Vitess is a good option for running MySQL at scale through sharding.



Vitess
CNCF Incubating

9. CONTAINER RUNTIME

You can use alternative container runtimes. The most common, all of which are OCI-compliant, are containerd, rkt and CRI-O.



containerd **rkt**
CNCF Incubating CNCF Incubating

2. CI/CD

- Setup Continuous Integration/Continuous Delivery (CI/CD) so that changes to your source code automatically result in a new container being built, tested, and deployed to staging and eventually, perhaps, to production
- Setup automated rollouts, roll backs and testing

4. OBSERVABILITY & ANALYSIS

- Pick solutions for monitoring, logging and tracing
- Consider CNCF projects Prometheus for monitoring, Fluentd for logging and Jaeger for Tracing
- For tracing, look for an OpenTracing compatible implementation like Jaeger



Prometheus **fluentd**
CNCF Incubating CNCF Incubating

JAEGER **OPENTRACING**
CNCF Incubating CNCF Incubating

6. NETWORKING

To enable more flexible networking, use a CNI-compliant network project like Calico, Flannel, or Weave Net.



CNI
CNCF Incubating

8. MESSAGING

When you need higher performance than JSON-REST, consider using gRPC.



gRPC
CNCF Incubating

10. SOFTWARE DISTRIBUTION

If you need to do secure software distribution, evaluate Notary, an implementation of The Update Framework.



Notary **TUF**
CNCF Incubating CNCF Incubating



CORE LIFECYCLE PILLARS

Source Code
Repository

Single
Source of
Truth

Continuous
Integration

Build and test
on every
integration

Package
Manager

Collect
dependencies
and publish
artifacts

Security
Scanner

Scanning of
dependencies
and
containers

Continuous
Delivery/
Deployment

Automated
deployment

Observability

Observe
and
Measure

DEVELOPER VS DEVOPS



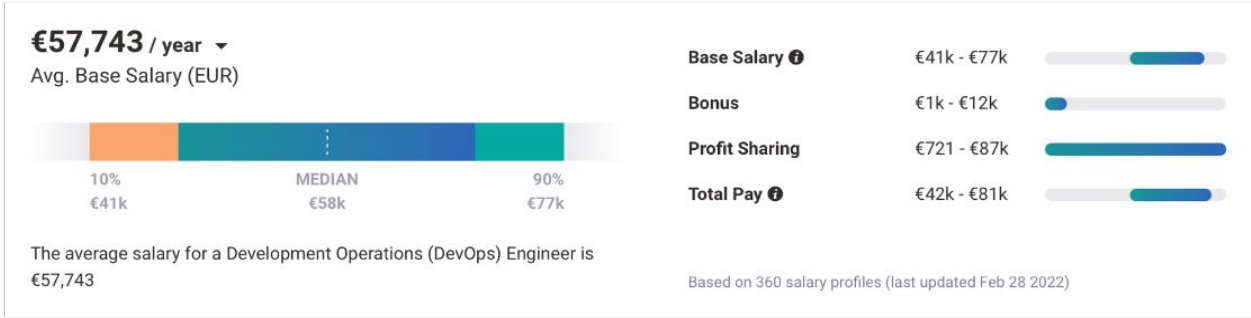
Why Should A Developer Care About DevOps



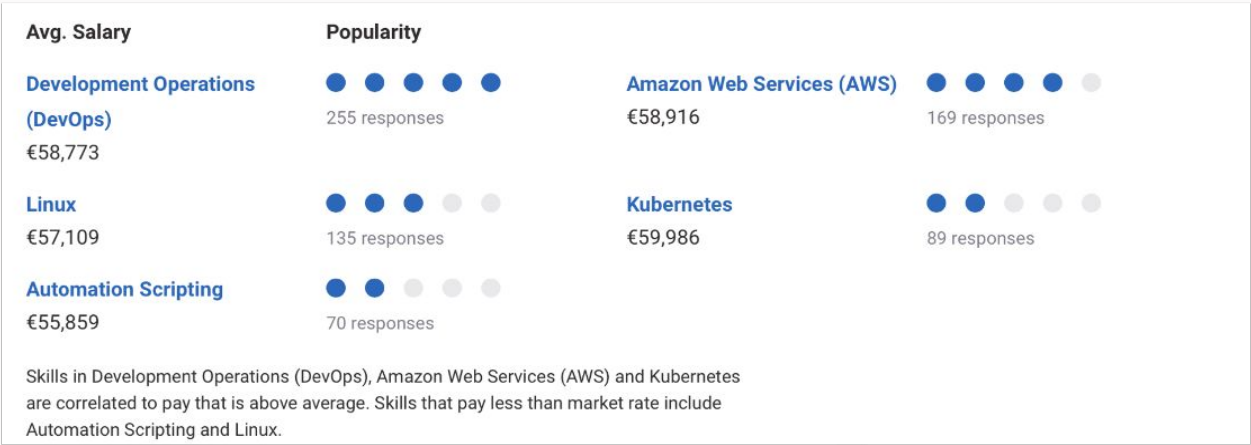
???

JOBS & SALARIES





~ 58K



Average Development Operations (DevOps) Engineer Salary Germany

[https://www.payscale.com/research/DE/Job=Development_Operations_\(DevOps\)_Engineer/Salary](https://www.payscale.com/research/DE/Job=Development_Operations_(DevOps)_Engineer/Salary)



€54,856 / year
Avg. Base Salary (EUR)



The average salary for a Software Engineer is €54,856

| | |
|----------------|-------------|
| Base Salary | €41k - €72k |
| Bonus | €1k - €10k |
| Profit Sharing | €487 - €20k |
| Commission | €0 - €10k |
| Total Pay | €41k - €75k |

Based on 2,662 salary profiles (last updated Mar 02 2022)

~54K

Avg. Salary

Java
€54,966

Popularity

● ● ● ● ●
817 responses

Python
€54,549

● ● ● ● ●
631 responses

JavaScript
€52,861

● ● ● ● ●
678 responses

C++ Programming Language
€56,800

● ● ● ● ●
483 responses

SQL
€53,626

● ● ● ● ●
529 responses

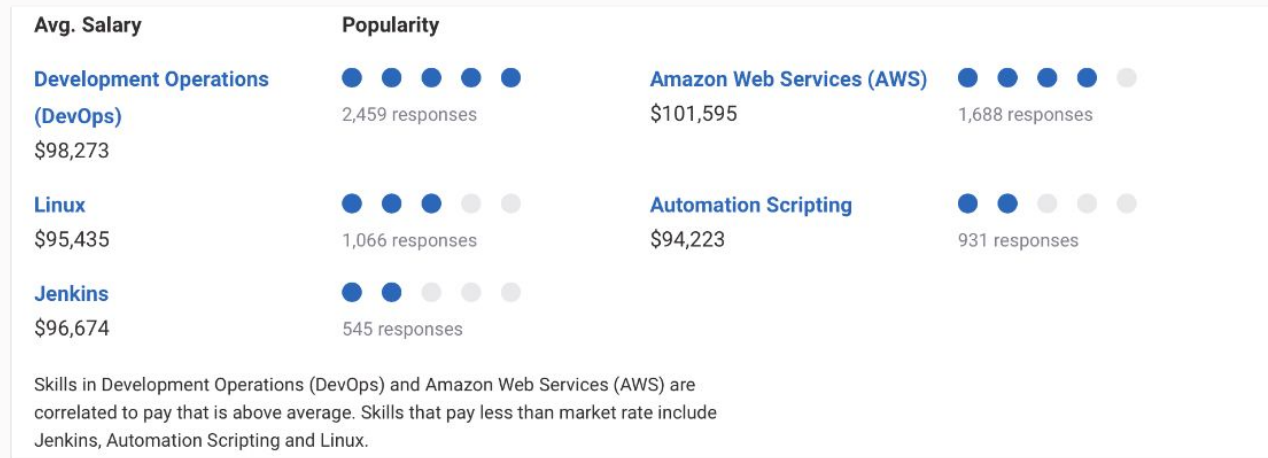
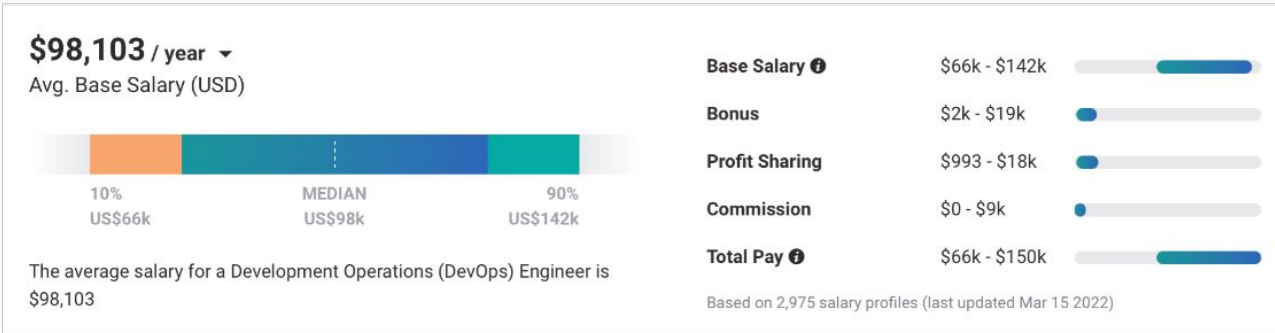
Skills in Java and C++ Programming Language are correlated to pay that is above average. Skills that pay less than market rate include SQL, Python and JavaScript.

Average Software Engineer Salary Germany

https://www.payscale.com/research/DE/Job=Software_Engineer/Salary



~98K



Average Development Operations (DevOps) Engineer Salary USA

[https://www.payscale.com/research/US/Job=Development_Operations_\(DevOps\)_Engineer/Salary](https://www.payscale.com/research/US/Job=Development_Operations_(DevOps)_Engineer/Salary)



\$88,540 / year ▾
Avg. Base Salary (USD)



The average salary for a Software Engineer is \$88,540



Based on 42,507 salary profiles (last updated Mar 16 2022)

~ 88K

Avg. Salary

JavaScript

\$87,845

Popularity



16,775 responses

SQL

\$84,943



13,758 responses

C# Programming Language

\$81,372



9,794 responses

Java

\$87,977



15,475 responses

Python

\$90,924



12,587 responses

Skills in Python are correlated to pay that is above average. Skills that pay less than market rate include C# Programming Language, SQL, Java and JavaScript.

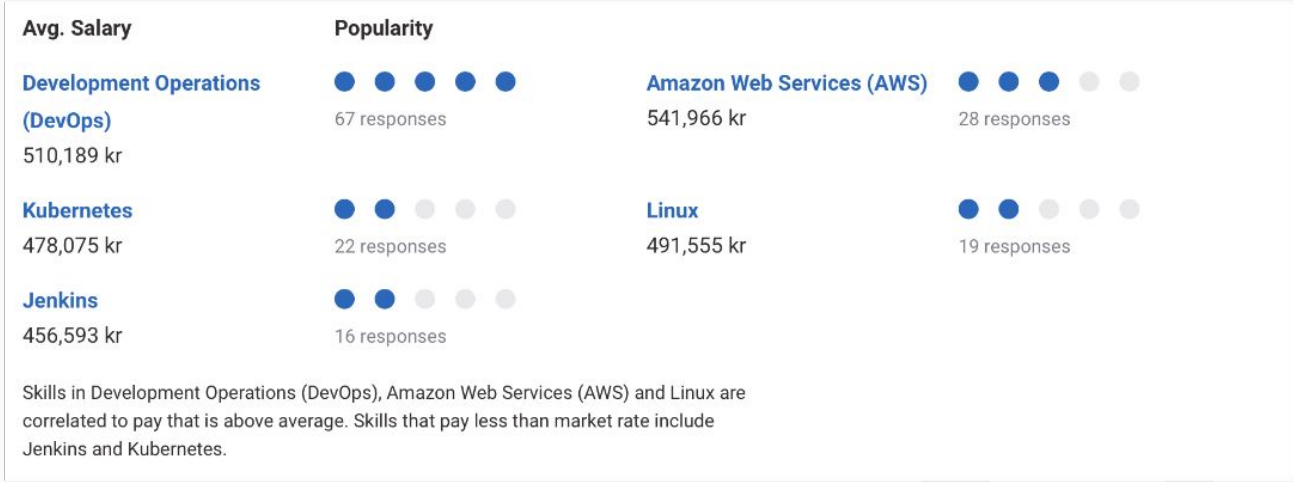
Average Software Engineer Salary USA

https://www.payscale.com/research/US/Job=Software_Engineer/Salary





~490K

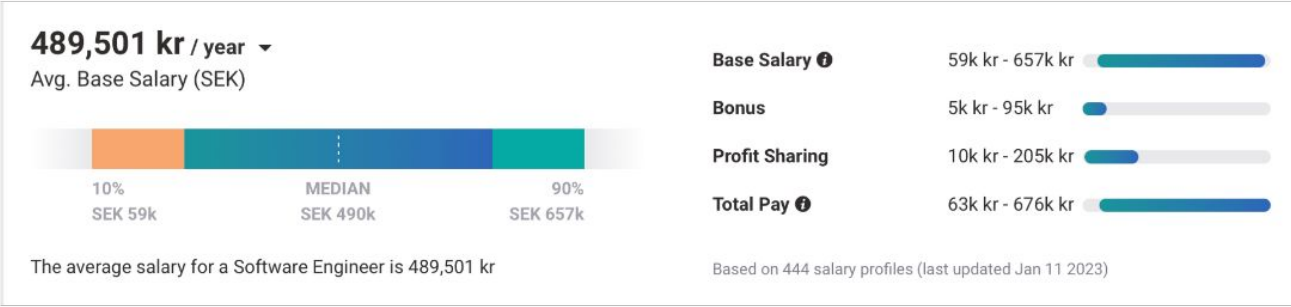


Average Development Operations (DevOps) Engineer Salary Sweden

[https://www.payscale.com/research/SE/Job=Development_Operations_\(DevOps\)_Engineer/Salary](https://www.payscale.com/research/SE/Job=Development_Operations_(DevOps)_Engineer/Salary)



~490K



Average Software Engineer Salary Sweden

https://www.payscale.com/research/SE/Job=Software_Engineer/Salary



WHY DEVOPS?

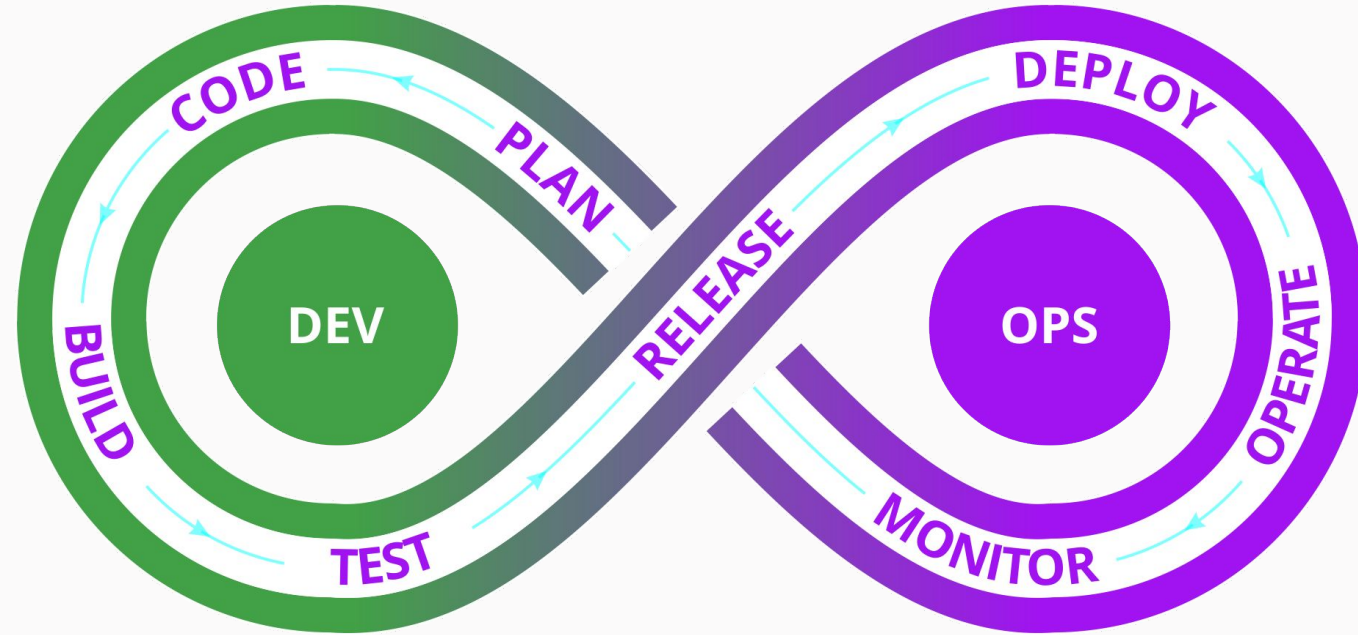


WHAT DO YOU REALLY CARE ABOUT?

- Fast?
- Efficient?
- Reliable?
- Economical?

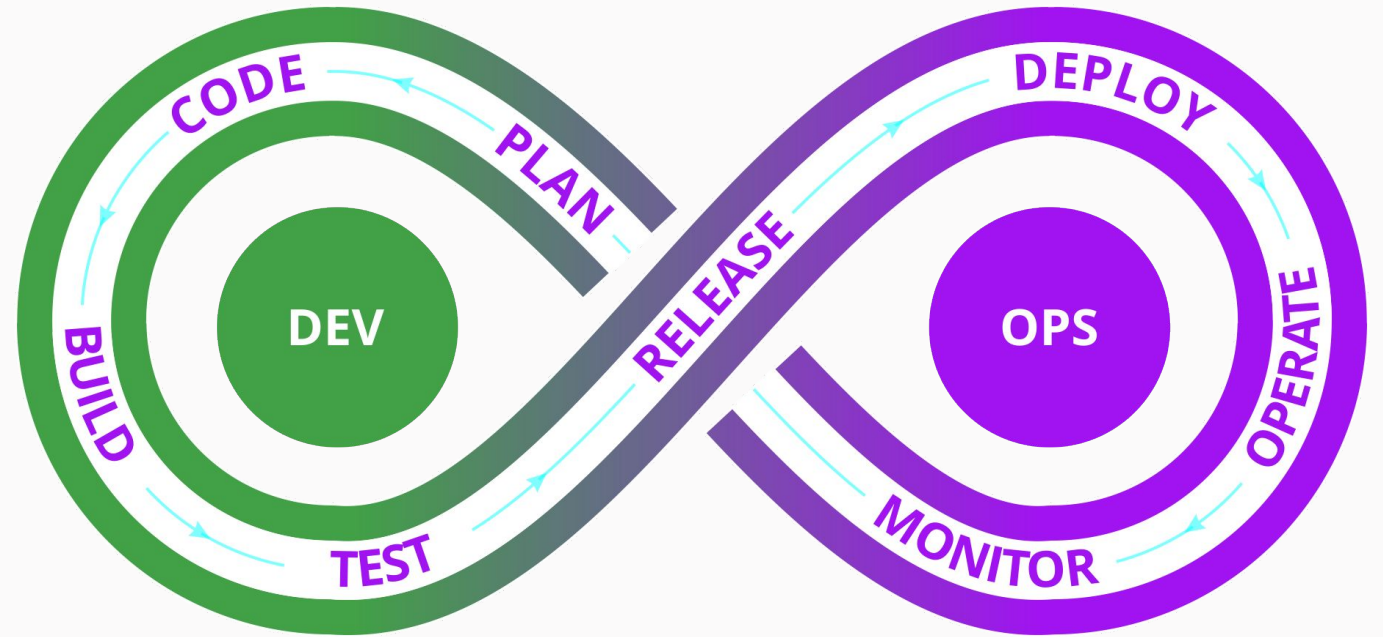


YOUR IT VALUE STREAM

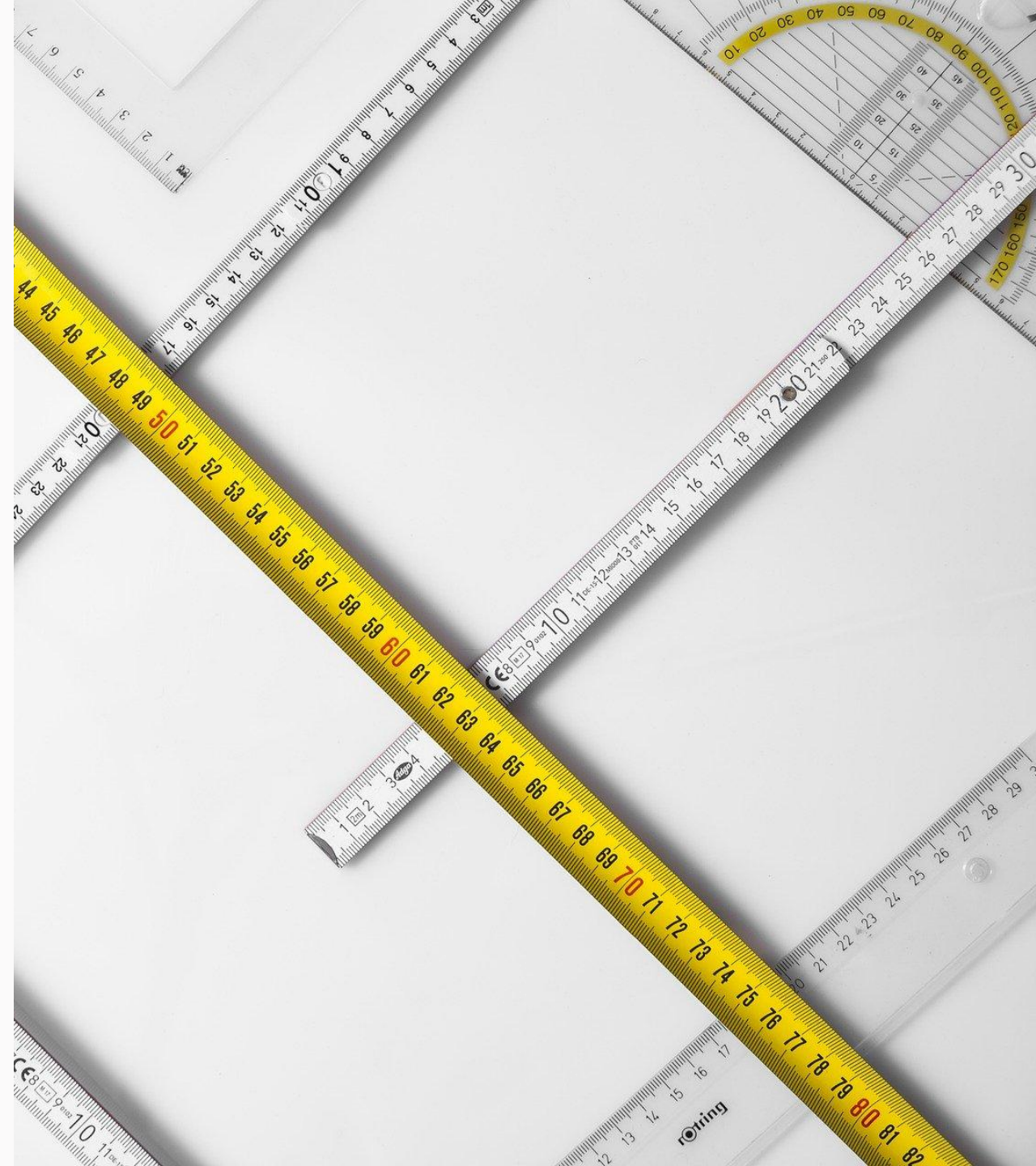


IMPROVE BY 1%

- Fast *Enough?*
- Efficient *Enough?*
- Reliable *Enough?*
- Economical *Enough?*



HOW TO MEASURE SUCCESS



Software delivery performance



Lead time
for changes



MTTR



Deployment
frequency



Change
failure rate



- Lead time for *changes*
- Deployment *frequency*
- *Time to restore* service
- Change failure *rate*

Throughput

Stability

Software delivery performance **The Four Key Metrics**

- Lead time for *changes*
- Deployment *frequency*
- *Time to restore* service
- Change failure *rate*

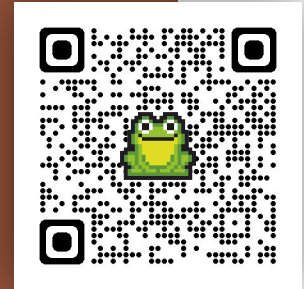
| Low | Mid | High |
|--------------|------------|-----------|
| 1/Mo - 1/6Mo | 1/W - 1/Mo | On demand |
| 1Mo - 6Mo | 1W - 1Mo | 1D - 1W |
| 1W - 1Mo | 1D - 1W | <1D |
| 46% - 60% | 16% - 30% | 0% - 15% |

Software delivery performance **The Four Key Metrics**

BEST PRACTICES



Continuous Delivery Best Practices



[Learn More](#) 

Learn about how best practices for continuous delivery can help you to develop, build, and deliver software faster, more reliably, and more securely.



Assess

Assess your current state and identify changes to prioritize.

[Read more ...](#)



Learn

Learn about continuous delivery best practices.

[Read more ...](#)



Community

Explore the ways community members implement best practices.

[Read more ...](#)

BEST PRACTICES (a few among many!)

- Understand the problem
- Address team culture
- Understand (and accept!) version control
- Implement Continuous Integration/Deployment/Testing
- Manage your Software Supply Chain
- Configuration Management (manage changes to both software and infrastructure)
- Reflect, Measure, Improve!



**WHY
NOT
DEVOPS?**



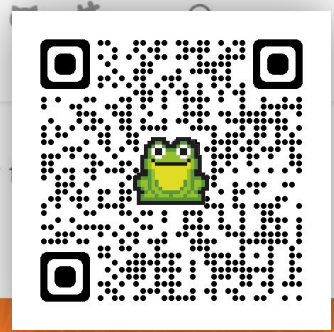
Blockers to DevOps evolution

- Organizational resistance to change
- Legacy architecture
- Shortage of skills
- Limited or lack of automation
- Unclear goals or objectives



RESOURCES DEVOPS





🇺🇦 We stand with the people of Ukraine. Please assist humanitarian efforts for the Ukrainian people and those affected by invasion of Ukraine by supporting international aid organizations, including [UNICEF](#).

cdCon Call for Papers Open



Submit your talk for the upcoming cdCon + GitOpsCon, taking place May 8 - 9, 2023 in Vancouver, Canada! The Call for Papers closes on February 10.

[Call for Papers](#)

CD Foundation is an open source community improving the world's ability to deliver software with security and speed.

We help you figure out the best DevOps path towards becoming a high-performing team using open source.

[Join Slack Channel](#)

[Adopt Best Practices](#)

[Join our Community](#)



View our "2022 in Review" blog and download the OpenSSF annual report

The Open Source Software Security Mobilization Plan



OpenSSF and The Linux Foundation propose 10 streams of investment to improve cybersecurity practices within open source development, code reviews, developer training, and software distribution.

Read the Plan

OpenSSF is committed to collaboration and working both upstream and with existing communities to advance open source security for all.



Hey there 🐧! Looking to learn more about OpenSSF?



OpenSSF is committed to collaboration and working both upstream and with existing communities to advance open source security for all.



Working Groups

Collaborate on the planning, design, and delivery of security tooling and best practices that secure critical open source projects.

[Join](#)



Town Halls

Stay informed about the latest happenings in open source security and engage with experts in our community,

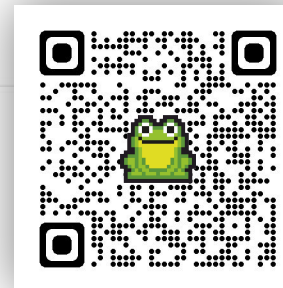
[Participate](#)



Training

Take free courses on secure coding practices as part of our Software Development Fundamentals Professional Certificate.

[Get Started](#)



Now Hiring: Chief of Staff

We are seeking a Chief of Staff to help us execute on our purpose alongside our member organizations, working groups and community leaders.

ossf / wg-best-practices-os-developers Public

generated from ossf/project-template

 Notifications Fork[Code](#) [Issues 6](#) [Pull requests](#) [Discussions](#) [Actions](#) [Projects](#) [Security](#) [Insights](#)[main](#) / [wg-best-practices-os-developers](#) / [docs](#) / Concise-Guide-for-Developing-More-Secure-Software.md [mrybczyn](#) docs/concise-guide*: ask for suggestions/updates ... ✓Latest commit e5972f8 on Jan 3 [History](#) 6 contributors

32 lines (29 sloc) | 6.67 KB

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

Concise Guide for Developing More Secure Software 2023-01-03

Here is a concise guide for all software developers for software development, building, and distribution. All tools or services listed are merely examples.

1. Ensure all privileged developers use [multi-factor authentication \(MFA\) tokens](#). This includes those with commit or accept privileges. MFA hinders attackers from "taking over" these accounts.

ossf / [wg-best-practices-os-developers](#) Publicgenerated from [ossf/project-template](#)

Notifications

Fork

[Code](#) [Issues](#) 6 [Pull requests](#) [Discussions](#) [Actions](#) [Projects](#) [Security](#) [Insights](#)main ▾ [wg-best-practices-os-developers](#) / [docs](#) / [Concise-Guide-for-Evaluating-Open-Source-Software.md](#) **mrybczyn** docs/concise-guide*: ask for suggestions/updates ... ✓Latest commit e5972f8 on Jan 3 [History](#)

3 contributors



51 lines (44 sloc) | 6.97 KB

[Code](#) [Raw](#) [Blame](#) [Edit](#) [Copy](#) [Trash](#)

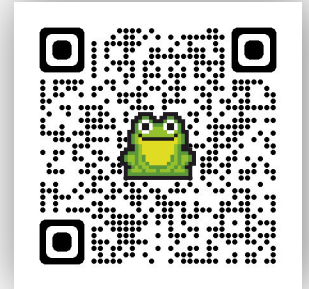
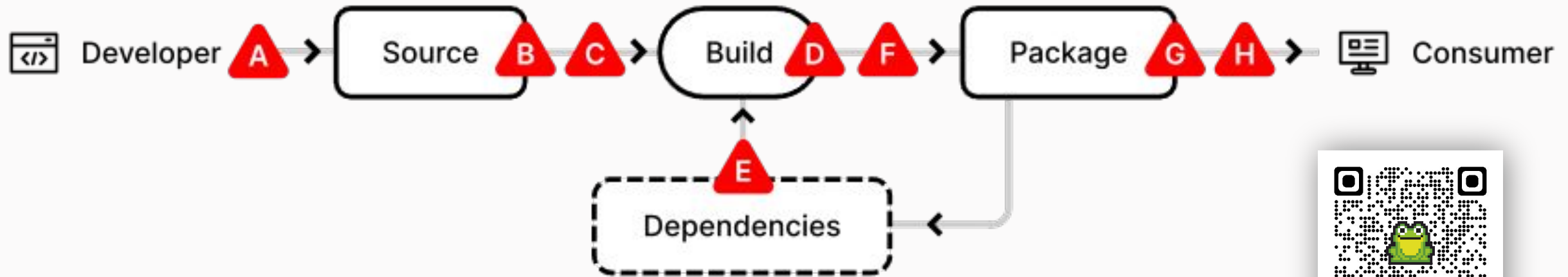
Concise Guide for Evaluating Open Source Software 2023-01-03

As a software developer, before using open source software (OSS) dependencies or tools, identify candidates and evaluate the leading ones against your needs. To evaluate a potential OSS dependency for security and sustainability, consider these questions (all tools or services listed are merely examples):

1. **Can you avoid adding it?** Can you use an existing (possibly indirect) dependency instead? Every new dependency increases the attack surface (a subversion of the new dependency, or its transitive dependencies, may subvert the system).
2. **Are you evaluating the intended version?** Ensure you are evaluating the intended version of the software, not a personal fork nor an attacker-controlled fork. These techniques help to counter the common "typosquatting" attack (where an attacker creates an "almost-

Supply-chain Levels for Software Artifacts

<https://slsa.dev/>



A Bypassed code review

B Compromised source control system

C Modified code after source control

D Compromised build platform

E Using a bad dependency

F Bypassed CI/CD

G Compromised package repo

H Using a bad package



THANK YOU!

IXCHEL RUIZ
Developer Advocate, JFrog

@ixchelruiz@mastodon.social
www.linkedin.com/in/ixchelruiz

MELISSA MCKAY
Developer Advocate, JFrog

@melissajmckay
www.linkedin.com/in/melissajmckay