



Home



My Network



Jobs



Messaging



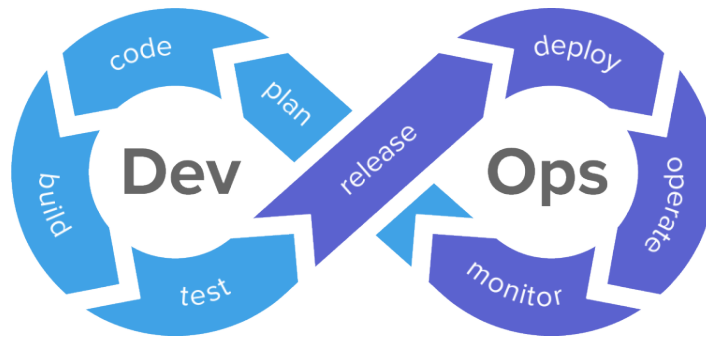
Notifications



Me



For Business

[Learn](#)[Try.f](#)

Images from: Roadmap.sh and Google images

DevOps Roadmap

**Sandip Das**Senior DevOps Engineer | AWS (Cloud)
Architect | Full Stack Developer | AWS

62 articles

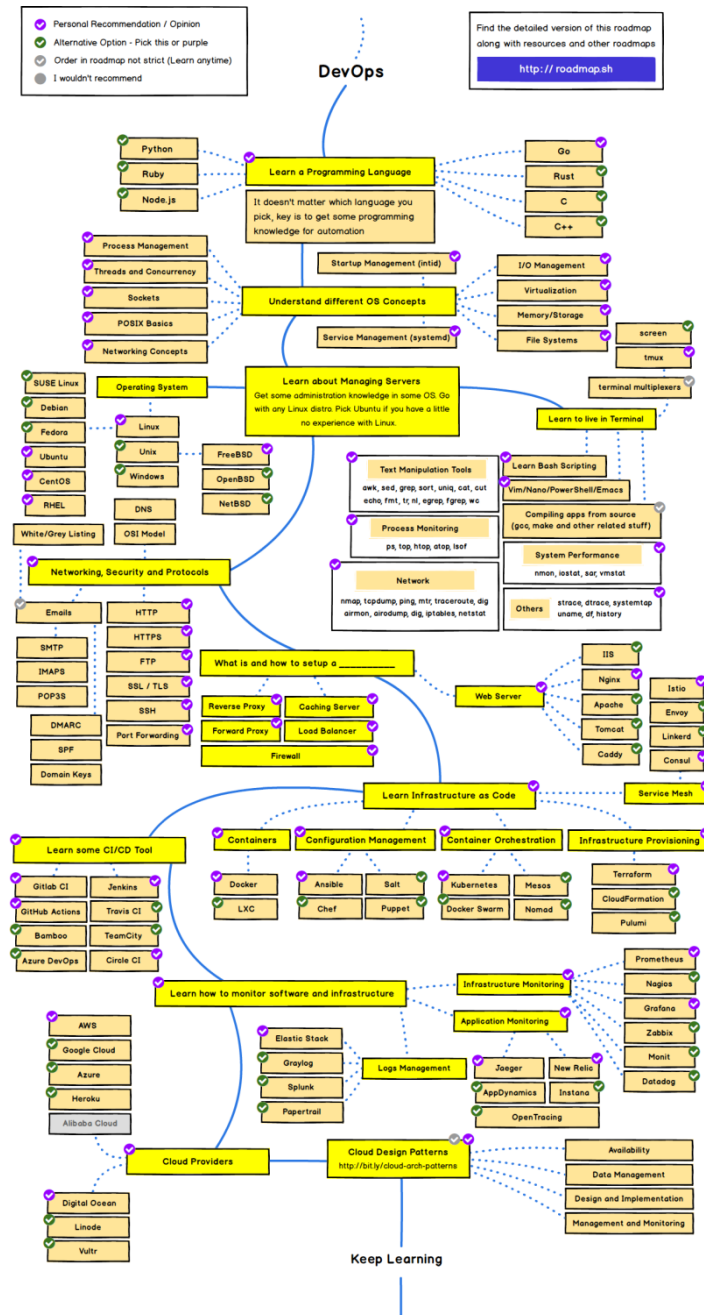
[+ Follow](#)

July 14, 2020

[Open Immersive Reader](#)

There has been a constant request on giving a roadmap on becoming a DevOps Engineer and of course the required skills that you need to master to become a DevOps Engineer.

In this article, I am going to explain all that in greater detail. But looks at this first 📌



Ok! Ok! I know it's a bit too much, don't get overwhelmed, let get started from basics.

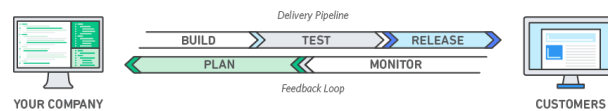
What is DevOps?

DevOps is a set of practices that combines software development and IT operations. It aims to shorten the systems development life cycle and provide continuous delivery with high software quality. DevOps is complementary with Agile software development; several

DevOps aspects came from Agile methodology. - [Wikipedia](#)

I like this definition from AWS too:

DevOps is the combination of cultural philosophies, practices, and tools that increases an organization's ability to deliver applications and services at high velocity: evolving and improving products at a faster pace than organizations using traditional software development and infrastructure management processes. This speed enables organizations to better serve their customers and compete more effectively in the market.



From the definition and above image you are already getting the idea which might be DevOps.

In simple terms, it's anything that can make the below process faster & smoother :

Coding -> Building -> Testing -> Packaging -> Releasing ->
Configuring -> Monitoring -> Repeat & Improve <-

We talked about what is DevOps, now let's talk about DevSecOps

I like Sumologic's explanation on DevSecOps 🙌

What is DevSecOps:

"DevSecOps is the philosophy of integrating security practices within the DevOps process. DevSecOps involves creating a 'Security as Code' culture with ongoing, flexible collaboration between release engineers and security teams. The DevSecOps movement, like DevOps itself, is focused on creating new solutions for complex software development processes within an agile framework.

DevSecOps is a natural and necessary response to the bottleneck effect of older security models on the modern continuous delivery pipeline. The goal is to bridge traditional gaps between IT and security while ensuring fast,

safe delivery of code. Silo thinking is replaced by increased communication and shared responsibility for security tasks during all phases of the delivery process."

It simply means making sure security is part of your DevOps process and making it automated.

Before even going further it's important to understand the daily tasks of DevOps Engineers, below are some examples :

- Design, build, test and deploy scalable, secure, distributed systems from development through production
- Manage the code repository(such as Git, SVN, BitBucket, etc.) including code merging and integrating, branching and maintenance, and remote repository management
- Manage, configure and maintain infrastructure system
- Design the database architecture and database objects and synchronize the various environments
- Design implement and support DevOps Continuous Integration and Continuous Delivery pipelines
- Research and implement new technologies and practices
- Document processes, systems, and workflows
- Creation and enhancement of dynamic monitoring and alerting solutions using industry-leading services
- Continuously analyze tasks that are performed manually and can be replaced by code
- Creation and enhancement of Continuous Deployment automation built on Docker and Kubernetes.

You might be feeling that it's all responsibility of the developer and operational personals combined, and it's it is, that's exactly why it's:

Dev(elopment) + Op(eration)s = DevOps

Ok, enough with the definitions, now let's talk about recommended roadmap:

1) Learn More on DevOps Culture & Practices:

DevOps is everyone's responsibility, it's not a hard written rule or such, instead, it's when all team comes together to achieve the ultimate goal i.e. optimize both the productivity of developers and the reliability of operations, make a quality product, release features quickly and fix bugs faster, introduce more automation and follow best DevOps practices such as:

Continuous Integration:

Continuous integration is a software development practice where developers regularly merge their code changes into a central repository, after which automated builds and tests are run. The key goals of continuous integration are to find and address bugs quicker, improve software quality, and reduce the time it takes to validate and release new software updates.

Continuous Delivery:

Continuous delivery is a software development practice where code changes are automatically built, tested, and prepared for a production release. It expands upon continuous integration by deploying all code changes to a testing environment and/or a production environment after the build stage. When continuous delivery is implemented properly, developers will always have a deployment-ready build artifact that has passed through a standardized test process.

Microservices:

The microservices architecture is a design approach to build a single application as a set of small services. Each service runs in its process and communicates with other services through a well-defined interface using a lightweight mechanism, typically an HTTP-based application programming interface (API). Microservices are built around business capabilities; each service is scoped to a single

purpose. You can use different frameworks or programming languages to write microservices and deploy them independently, as a single service, or as a group of services.

Infrastructure as Code:

Infrastructure as code is a practice in which infrastructure is provisioned and managed using code and software development techniques, such as version control and continuous integration. The cloud's API-driven model enables developers and system administrators to interact with infrastructure programmatically, and at scale, instead of needing to manually set up and configure resources. Thus, engineers can interface with infrastructure using code-based tools and treat infrastructure like how they treat application code. Because they are defined by code, infrastructure and servers can quickly be deployed using standardized patterns, updated with the latest patches and versions, or duplicated in repeatable ways.

Monitoring and Logging:

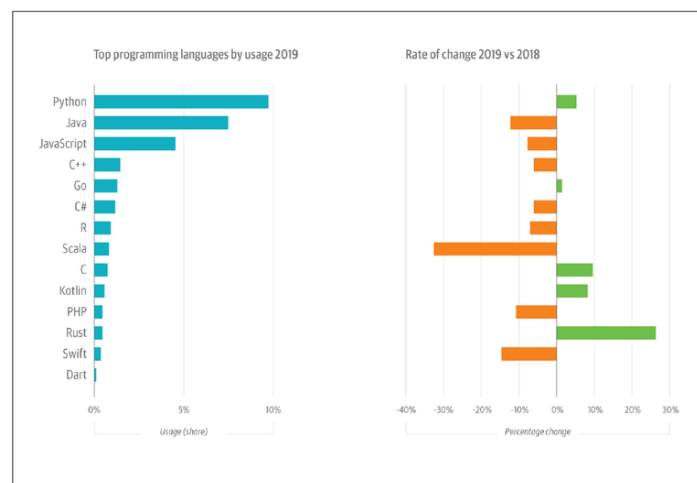
Organizations monitor metrics and logs to see how application and infrastructure performance impacts the experience of their product's end-user. By capturing, categorizing, and then analyzing data and logs generated by applications and infrastructure, organizations understand how changes or updates impact users, shedding insights into the root causes of problems or unexpected changes. Active monitoring becomes increasingly important as services must be available 24/7 and as application and infrastructure update frequency increases. Creating alerts or performing real-time analysis of this data also helps organizations more proactively monitor their services.

Communication and Collaboration :

Increased communication and collaboration in an organization are some of the key cultural aspects of DevOps. The use of DevOps tooling and automation of the software delivery process establishes collaboration by physically bringing together the workflows and

responsibilities of development and operations. Building on top of that, these teams set strong cultural norms around information sharing and facilitating communication through the use of chat applications, issue or project tracking systems, and wikis. This helps speed up communication across developers, operations, and even other teams like marketing or sales, allowing all parts of the organization to align more closely with goals and projects.

2) Learn at least one back-end Programming language:



Yes, you have to learn at least one programming language so that you can program automation scripts and if you could able to learn multiple demanding languages, it's even better but not all 😊

My recommendations are: Go, Python, JavaScript (Node.js)

Good Courses:

Go:

Paid: <https://www.udemy.com/course/go-the-complete-developers-guide/>

Free: <https://www.youtube.com/watch?v=YS4e4q9oBaU>

Python:

Paid: <https://www.udemy.com/course/python-the-complete-python-developer-course/>

Free:

https://www.youtube.com/watch?v=_uQrJ0TkZlc

<https://www.youtube.com/watch?v=rfscVS0vtbw>

<https://www.youtube.com/watch?v=WGJJlRtnfpk>

Node.js

Paid: <https://www.udemy.com/course/the-complete-nodejs-developer-course-2/>

Free:

<https://www.youtube.com/watch?v=JnvKXcSI7yk>

<https://www.youtube.com/watch?v=RLtyhwFtXQA>

3) Understand Different OS Concepts:



Essential OS concepts necessary for a DevOps engineer, since as a DevOps engineer ultimately will be working with different OS (Mostly Linux) during development and/or deployment.

DevOps engineer often has to do system-level OS optimizations to increase performance and/or do other optimizations as per application needs, knowledge of

Process Management, Threads and Concurrency, Sockets, I/O Management, Virtualization, Memory storage, and File systems would be very much useful in this process.

Good Courses:

Paid:

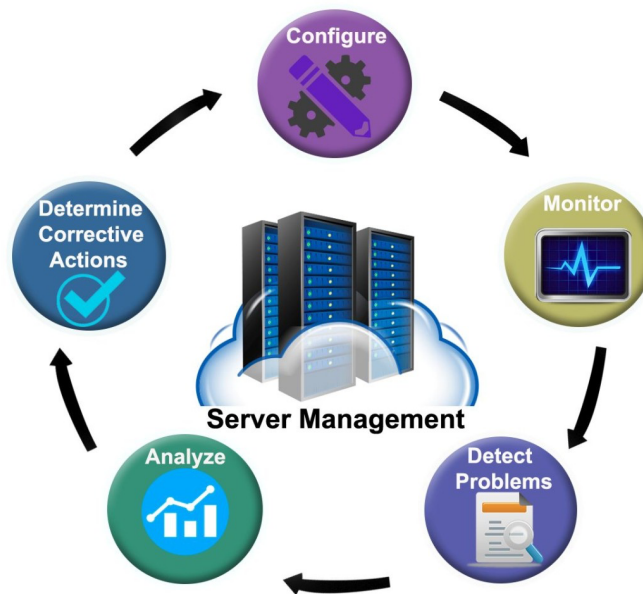
<https://www.udemy.com/course/operating-system-concepts/>

Free:

<https://www.youtube.com/watch?v=Snrh580U3tI>

<https://www.youtube.com/watch?v=mXw9ruZaxzQ>

4) Learn about managing servers:



As a DevOps engineer, we have to manage all kinds of servers, and interesting enough it's not just 1 or 2 but a fleet of servers (sometimes even 100+), mostly consisting of Linux instances (and rarely other OS) and that's why the previous step i.e. basic understanding different OS concepts are needed, and after that deepen your knowledge in Linux

Systems, that will make your daily life easier 🤖

As DevOps Engineer, We should know about dynamically scaling up/down the servers, without rewriting the configuration files and making a necessary recommendation to developers so that the application itself complies with the scalability approach.

Also, you will have to handle web servers at some point in time if not all the time 😊, so having knowledge and practical experience of Reverse Proxy servers such as Nginx and/or Apache is also necessary as well.

Good Courses:

Paid:

<https://www.udemy.com/course/linux-administration-bootcamp/>

<https://www.udemy.com/course/nginx-fundamentals/>
(Nginx)

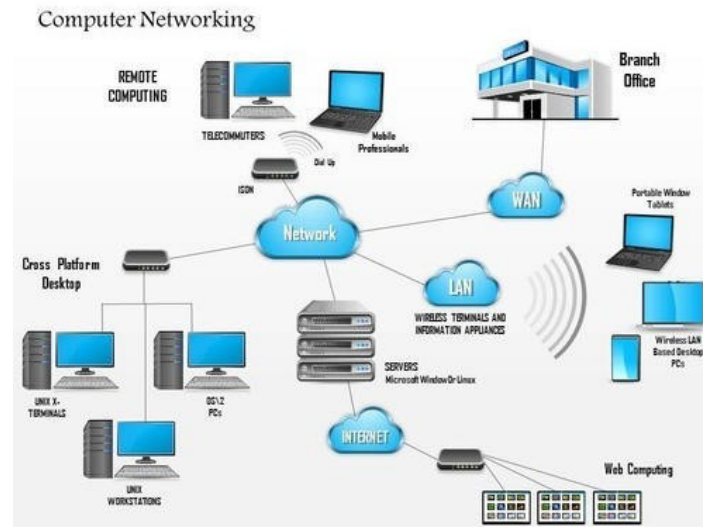
Free:

<https://www.youtube.com/watch?v=wsh64rjnRas>

https://www.youtube.com/watch?v=v_1zB2WNN14

<https://www.youtube.com/watch?v=IWjZSgXu5VU>
(Nginx)

5) Networking, Security, and Protocols



Knowledge of networking, security, and basic protocols are necessary, as DevOps engineer we have to set up VPC and security groups/firewalls and utilize various protocols such as TCP/IP (Transfer Control Protocol/Internet Protocol), HTTP (Hypertext Transfer Protocol), SSL (Secure Sockets Layer), SSH (Secure Shell), FTP (File Transfer Protocol), SMTP (Simple Mail Transfer Protocol) and more.

In particular to networking concepts, necessary concepts such as DNS record management, routing, firewalls and ports, basic utilities like ping, ssh, netstat, NCR and IP, load balancing, and TLS encryption are also necessary.

Good Courses:

Paid:

<https://www.udemy.com/course/introduction-to-computer-networks/>

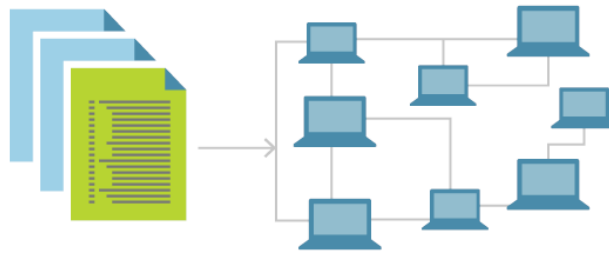
<https://www.udemy.com/course/network-security-course/>

Free:

<https://www.youtube.com/watch?v=QKfk7YFILws>

https://www.youtube.com/watch?v=U_P23SqJaDc

6) Learn Infrastructure as Code:



"Infrastructure as code is the process of managing and provisioning computer data centers through machine-readable definition files, rather than physical hardware configuration or interactive configuration tools."

I also like this definition from AWS and Microsoft:

AWS:

"Infrastructure as code is a practice in which infrastructure is provisioned and managed using code and software development techniques, such as version control and continuous integration. The cloud's API-driven model enables developers and system administrators to interact with infrastructure programmatically, and at scale, instead of needing to manually set up and configure resources. Thus, engineers can interface with infrastructure using code-based tools and treat infrastructure like how they treat application code. Because they are defined by code, infrastructure and servers can quickly be deployed using standardized patterns, updated with the latest patches and versions, or duplicated in repeatable ways."

Microsoft:

"Infrastructure as Code (IaC) is the management of infrastructure (networks, virtual machines, load balancers, and connection topology) in a descriptive model, using the same versioning as the DevOps team uses for source code. Like the principle that the same source code generates the same binary, and the IaC model generates the same environment every time it is applied. IaC is a key DevOps practice and is used in conjunction with continuous delivery.

Infrastructure as Code evolved to solve the problem of *environment drift* in the release pipeline. Without IaC, teams must maintain the settings of individual deployment environments. Over time, each environment becomes a *snowflake*, that is, a unique configuration that cannot be reproduced automatically. Inconsistency among environments leads to issues during deployments. With snowflakes, administration and maintenance of infrastructure involve manual processes which were hard to track and contributed to errors."

All In Simple Words:

Infrastructure as code (IaC) means managing your IT infrastructure using configuration files.

As DevOps engineers, we should know about containers (Docker), Container Orchestration (Kubernetes, Docker Swarm), Configuration management tools like Ansible, Chef, Salt, and Puppet, Infrastructure Provisionings tools like Terraform and Cloud formation.

Good Courses:

Paid:

<https://www.udemy.com/course/docker-mastery/>
(Docker)

<https://www.udemy.com/course/learn-kubernetes/>
(Kubernetes)

<https://www.udemy.com/course/learn-ansible/> (Ansible)

<https://www.udemy.com/course/chef-fundamentals-a->

[recipe-for-automating-infrastructure/](#) (Chef)

<https://www.udemy.com/course/learning-salt/> (Salt)

<https://www.udemy.com/course/learn-puppet/> (Puppet)

<https://www.udemy.com/course/learning-path-automation-with-ansible-puppet-and-salt/> (Ansible, Puppet and salt)

<https://www.udemy.com/course/learn-devops-infrastructure-automation-with-terraform/> (Terraform)

<https://www.udemy.com/course/aws-cloudformation-master-class/> (Cloud Formation)

Free:

<https://www.youtube.com/watch?v=RS1stPUiEjY> (Docker)

<https://www.youtube.com/watch?v=M13Lx7yk3Hg>
(Kubernetes)

<https://www.youtube.com/watch?v=EcnqJbxBcM0>
(Ansible)

<https://www.youtube.com/watch?v=LTijUJEehDA> (Chef)

<https://www.youtube.com/watch?v=XovHdJZp1X4&list=PLgGQIE0cGrkiDRmJ3YWBKII2KuYfFvfzV> (Salt)

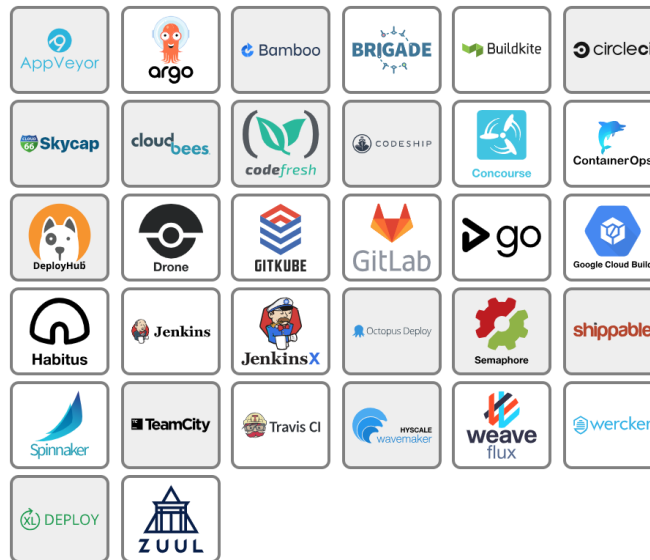
<https://www.youtube.com/watch?v=kHD4KQKKP5Y>
(Puppet)

<https://www.youtube.com/watch?v=0-z1G0BFZSU>
(Terraform)

<https://www.youtube.com/watch?v=6R44BADNJA8>
(Cloud Formation)

7) Learn Some CI/CD Tools:

Continuous Integration & Delivery



Continuous Integration (CI):

Continuous Integration (CI) is a development practice where developers integrate code into a shared repository frequently, preferably several times a day. Each integration can then be verified by an automated build and automated tests. While automated testing is not strictly part of CI it is typically implied.

Continuous Delivery (CD):

Continuous Delivery (CD) is a software engineering approach in which teams produce software in short cycles, ensuring that the software can be reliably released at any time and, when releasing the software, doing so manually. It aims at building, testing, and releasing software with greater speed and frequency.

As a DevOps Engineer, widely used tool you should be aware of, learn and utilize:

Jenkins, GitLab CI, CircleCI, Microsoft VSTS, CodeShip, Bamboo, GitHub Actions

Good Courses:

Paid:

<https://www.udemy.com/course/jenkins-from-zero-to-hero/> (Jenkins)

Free:

<https://www.youtube.com/watch?v=FX322RVNGj4>
(Jenkins)

8. Learn to monitor software and infrastructure:



Organizations monitor metrics and logs to see how application and infrastructure performance impacts the experience of their product's end-user. By capturing, categorizing, and then analyzing data and logs generated by applications and infrastructure, organizations understand how changes or updates impact users, shedding insights into the root causes of problems or unexpected changes. Active monitoring becomes increasingly important as services must be available 24/7 and as application and infrastructure update frequency increases. Creating alerts or performing real-time analysis of this data also helps organizations more proactively monitor their services.

A DevOps Engineer needs to collect feedback and

implement the changes quickly, for that you should know monitoring tools like [Nagios](#), [Prometheus](#), [Grafana](#)

Good courses:

Paid:

<https://www.udemy.com/course/setting-up-nagios-4/>
(Nagios)

<https://www.udemy.com/course/monitoring-and-alerting-with-prometheus/> (Prometheus)

<https://www.udemy.com/course/grafana-graphite-and-statsd-visualize-metrics/> (Grafana)

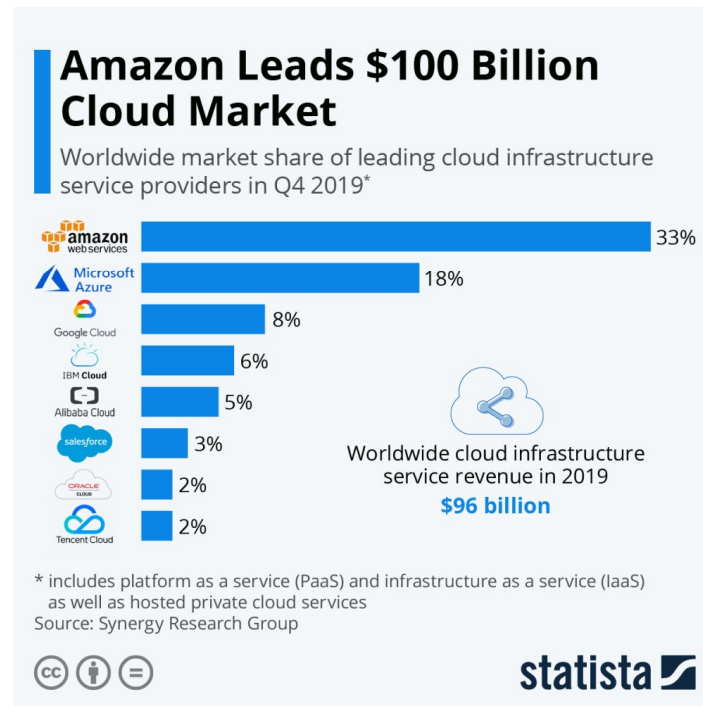
Free:

<https://www.youtube.com/watch?v=knCO6wzCW3w>
(Nagios)

<https://www.youtube.com/watch?v=9GMWvFcQjYI>
(Prometheus)

<https://www.youtube.com/watch?v=sTP7yzXmdFk&list=PLyJqGMym0vnO9osZ-EBV6iu2l10muE2A-> (Grafana)

9) Cloud Providers:



I always say:

There is NO DevOps Without Cloud

It's self-explanatory but still, let me simplify a bit, for that let me ask you a question?

Would you think about doing DevOps for your local machine?

The answer is straight cut no, isn't it!! yes, we don't bother DevOps for local development environments but we do care when we are thinking about hosting applications in the Cloud, managing multiple applications & servers, and monitoring both applications as well as Infrastructure for staging and production environments.

Currently, there are multiple cloud service providers out there out of which 3 are holding top market share, are AWS, Azure, and GCP, let me tell you what tools for each cloud provider you should learn:

AWS: The Cloud Leader

CI & CD: [AWS CodePipeline](#), [AWS CodeBuild](#), [AWS CodeDeploy](#), [AWS CodeStar](#)

Microservices: [Amazon Elastic Container Service \(ECS\)](#), [AWS Lambda](#),

Infrastructure as Code: [Cloud Formation](#), [AWS OpsWorks](#)

Configuration Management: [AWS System Manager](#)

Policy as Code: [AWS Config](#)

Monitoring and Logging: [Amazon CloudWatch](#), [X-Ray](#), [AWS CloudTrail](#)

Platform as a Service: [AWS Elastic Beanstalk](#)

Version Control: [AWS CodeCommit](#)

Microsoft Azure:

CI & CD: [Azure Pipelines](#), [Azure Test Plans](#), [Azure Artifacts](#)

Version Control: [Azure Repos](#)

GCP:

CI/CD: [Cloud Build](#), [Tekton](#), [Artifact Registry](#), [Deployment Manager](#)

Good Courses:

Paid:

<https://www.udemy.com/course/aws-certified-developer-associate/> (AWS)

<https://www.udemy.com/course/aws-certified-developer-associate-dva-c01/> (AWS)

<https://www.udemy.com/course/aws-certified-devops-engineer-professional-hands-on/> (AWS)

<https://www.udemy.com/course/azure-devops-for-beginners/> (Azure)

<https://www.udemy.com/course/azure-devops-ci-cd-pipelines/> (Azure)

<https://www.udemy.com/course/microsoft-azure-devops/> (Azure)

<https://www.udemy.com/course/google-certified-architect-developer-engineer-data-devops/> (GCP)

Free:

<https://www.youtube.com/watch?v=RrKRN9zRBWs>
(AWS)

<https://www.youtube.com/watch?v=PxGGeNEdb3E>
(AWS)

<https://www.youtube.com/watch?v=k1RI5locZE4> (AWS)

<https://www.youtube.com/watch?v=tDuruX7XSac> (Azure)

<https://www.youtube.com/watch?v=QmvAYDc4UUw>
(Azure)

<https://www.youtube.com/watch?v=RfbXgK-T0dM> (GCP)

Summary:

I would say it's a long and never-ending journey, keep learning, keep improving, don't try to learn everything at once, instead of one at a time, and related topics together to make the learning process a bit faster. Don't just learn, do hands-on, that's what is important and will help you get quality jobs afterward.

I wish you good luck on your DevOps journey 😊

Update: [Read even more Simplified DevOps Roadmap 2021 & 2022 \(Click here\)](#)

About the Author:



Sandip Das works as a Sr. Cloud Solutions Architect & DevOps Engineer for multiple tech product companies/start-ups, also holding the title of "[AWS Container Hero](#)".

He is always in "keep on learning" mode, enjoys sharing knowledge with others, and currently holds 4 AWS Certifications. Sandip finds blogging as a great way to share knowledge: he writes articles on [LinkedIn](#) about Cloud, DevOps, Programming, and more. He also creates video tutorials on his [YouTube channel](#).

[Report this](#)

Published by



Sandip Das

Senior DevOps Engineer | AWS (Cloud) Architect | Full Stack Developer | ...
Published · 3y

62

articles

[+ Follow](#)

Interested in learning about DevOps, and want to pursue as your career!!! then this article is a must-read for you 😊 This article I am just not saying what to learn but also from where to learn 👍 After reading this article if you think it's useful then don't forget to share with others and please put your valuable feedback in the comment section. [Sandip Das #devops](#)



Like



Comment

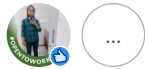


Share

431 67 comments

Reactions





67 Comments

Most relevant ▾



Add a comment...



Subha Swetha Nandyala · 3rd+

1y ...

SysDev @ Amazon | Previously Software Engineer at HCL Technologies

Clean Roadmap and Guidelines to follow! Great work [Sandip Das](#) ✨

Like · ❤️ 1 | Reply · 1 Reply



Sandip Das · 3rd+

1y ...

Senior DevOps Engineer | AWS (Cloud) Architect | Full Stack Developer | AWS Container Hero | Hashicorp Ambassador | Top Cloud Computing Voice | Teacher | Mentor | Youtube @learnTechWithSandip

Thanks [Subha Swetha](#) 😊

Like | Reply



Liju Philip · 3rd+

1y ...

self taught developer

Great work sir thank you

Like | Reply

Load more comments




Sandip Das

Senior DevOps Engineer | AWS (Cloud) Architect | Full Stack Developer | AWS Container Hero | Hashicorp Ambassador | Top Cloud Computing Voice | Teacher | Mentor | Youtube @learnTechWithSandip


+ Follow

More from Sandip Das




Python vs GoLang Vs JavaScript

Sandip Das on LinkedIn



Scaling Cloud Resources: Your Ultimate Guide


Sandip Das on LinkedIn



The Power of Generative AI: Transforming DevOps with ChatGPT

Sandip Das on LinkedIn

[See all 62 articles](#)



Messaging

...

