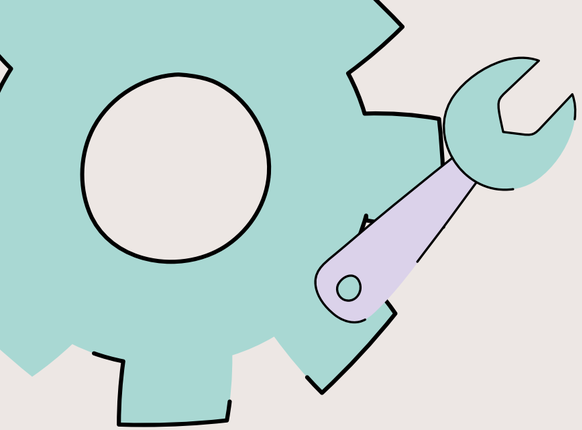
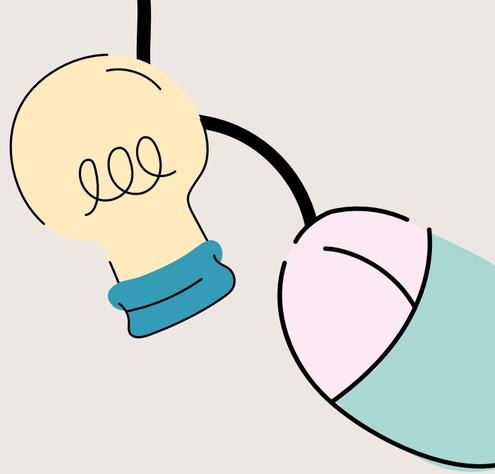
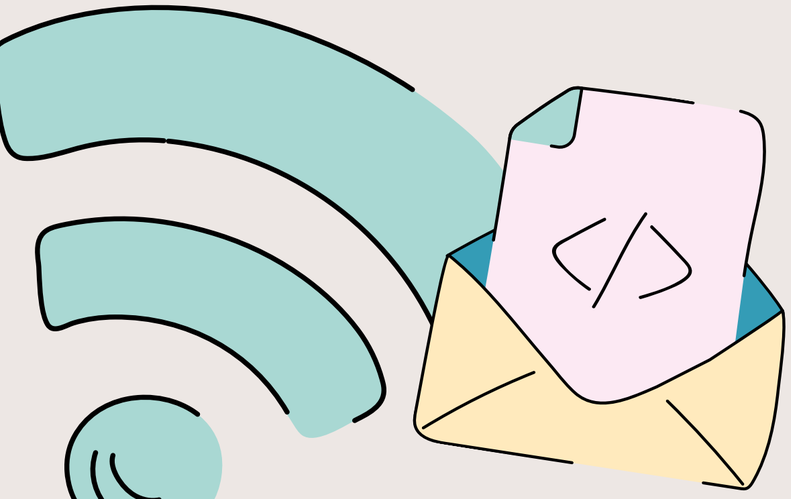


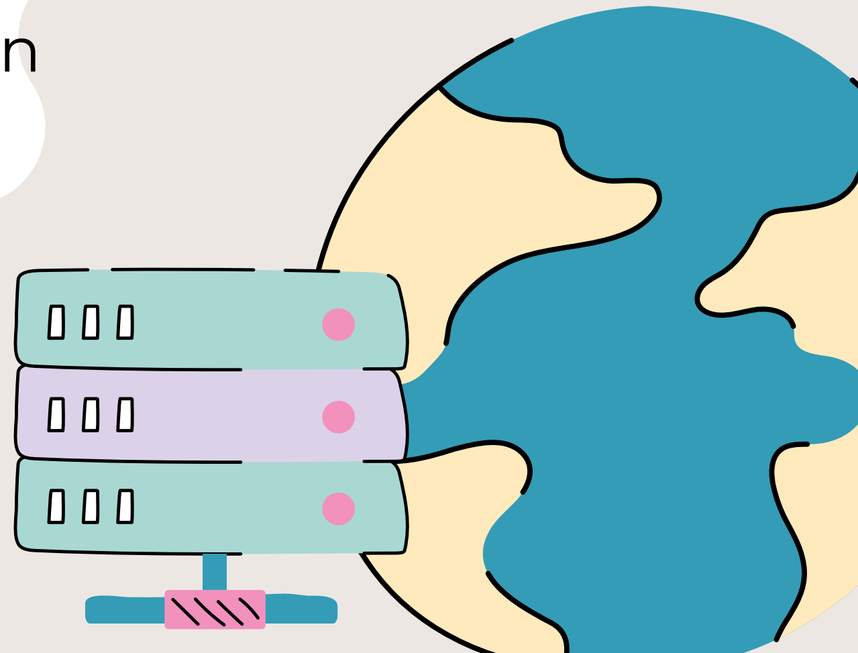


Roadmap DevOps Engineer

Una guía detallada para dominar desarrollo, operaciones y automatización

Por Juan Duran





Introducción

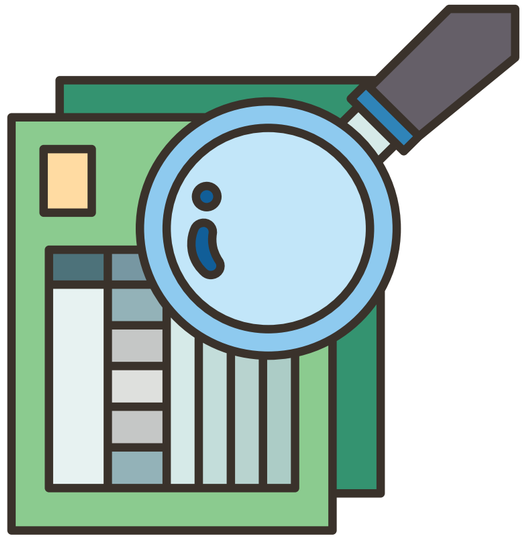
DevOps no es solo una colección de herramientas o un rol técnico, es una forma de trabajar: **colaboración entre desarrollo y operaciones**, **automatización** constante, mejora continua y foco en la entrega de valor real.

Esta **presentación** no busca abrumarte con conceptos, sino darte un **camino claro** y bien **estructurado** que puedas recorrer paso a paso.

Si estás empezando, este **roadmap** puede ayudarte a no perderte entre tantas opciones.

Y si ya llevas tiempo, puede ayudarte a detectar tus puntos débiles o próximos pasos.

No se trata de correr, sino de avanzar con sentido.

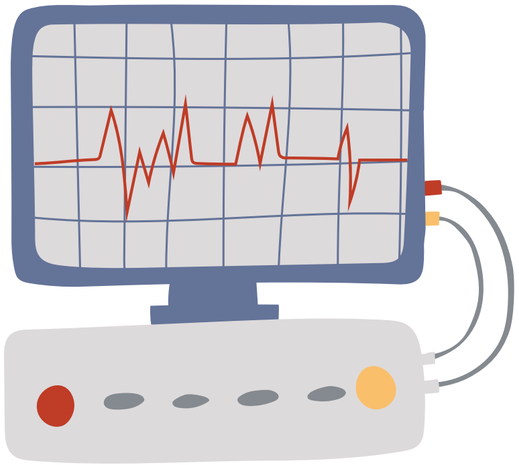


Fundamentos de sistemas

Antes de aprender herramientas, necesitas entender cómo funciona la tecnología que las soporta.
DevOps trabaja sobre **sistemas**, **redes**, **procesos** y **servicios**. Aquí está tu punto de partida:

🔹 Aprende a trabajar con **Linux**: comandos básicos, manejo de archivos, procesos, permisos, gestión de servicios, cron jobs.
🔹 Estudia **redes** básicas: modelos OSI, protocolos como HTTP/HTTPS, DNS, puertos, firewalls, NAT.
🔹 Comprende cómo funcionan **servidores** físicos y virtuales, máquinas locales y cloud.
🔹 Familiarízate con conceptos como **IPs**, **puertos**, **procesos** en segundo plano, puertos abiertos, etc.

Sin esta base, todo lo demás se convierte en repetir comandos sin saber realmente qué estás haciendo.



Control de versiones (Git)

El control de versiones es una habilidad fundamental en cualquier entorno profesional. En DevOps, **Git** es **esencial** para trabajar con código, scripts de automatización y pipelines.

- Aprende los **comandos básicos**: git init, clone, add, commit, push, pull, merge.
- Entiende los **flujos de trabajo colaborativos**: GitHub Flow, GitFlow, trunk-based development.
- Domina el uso de **ramas**, pull requests, revisiones y resolución de conflictos.
- Trabaja con plataformas como **GitHub**, **GitLab** o **Bitbucket**, donde también gestionarás tus pipelines, documentación o incluso issues.

Una buena práctica con Git no solo facilita el trabajo colaborativo, también te evita muchos dolores de cabeza.

Scripting y automatización

Un **DevOps** no necesita ser un desarrollador full-stack, pero sí debe **escribir scripts** que **automaticen** procesos o **configuren** entornos.

🔹 Aprende **Bash/Shell scripting**: automatización de tareas del sistema, backups, instalación de paquetes, despliegues simples.
🔹 Aprende **Python** para automatizar tareas más complejas, conectar APIs, generar archivos o analizar logs.
🔹 Conoce formatos como **YAML** (usado en pipelines y Kubernetes), JSON, y plantillas de configuración.

Automatizar tareas manuales te ahorra tiempo, reduce errores y es uno de los grandes pilares del enfoque DevOps.

CI/CD (Integración y Despliegue Continuos)

CI/CD es el corazón de un flujo **DevOps** moderno: **automatizar** desde que se sube código hasta que se **despliega** en producción.

- **CI** (Integración continua): **automatiza tests**, **builds** y análisis de calidad cada vez que se sube código.
- **CD** (Despliegue continuo): **automatiza** el paso a **staging** o **producción**, reduciendo tiempos y errores.
- **Herramientas** populares: **GitHub Actions**, **GitLab CI**, **Jenkins**, **Azure DevOps**.
- Aprende a escribir archivos de **pipeline**, **orquestar** pasos, manejar variables y condiciones.

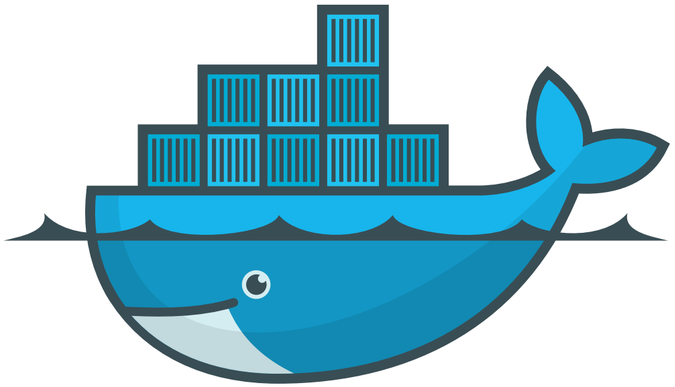
CI/CD bien hecho significa rapidez y confianza en cada entrega.

Contenedores y Docker

Docker cambió la forma en la que desarrollamos y desplegamos aplicaciones. Como DevOps, necesitas entenderlo a fondo.

🔹 Aprende qué es un **contenedor** y cómo se diferencia de una máquina virtual.
🔹 Crea **imágenes** personalizadas con **Dockerfiles**.
🔹 Usa **volúmenes** y **redes** para hacer tus contenedores más útiles.
🔹 Aprende a trabajar con **Docker Compose** para levantar entornos completos.
🔹 Familiarízate con **Docker Hub** y cómo versionar imágenes.

Con Docker puedes simular entornos, testear despliegues y garantizar que lo que funciona en tu máquina, también funcione en producción.



Kubernetes

Cuando tienes **muchos contenedores**, necesitas **orquestarlos**. Y ahí entra **Kubernetes**.

- Aprende **conceptos clave**: pods, deployments, services, namespaces, configmaps, secrets.
- Comprende cómo **escalar**, **actualizar** o **reiniciar** aplicaciones automáticamente.
- Trabaja con **clústers locales** (como Minikube o Kind) antes de pasar a producción.
- Familiarízate con **Helm** para instalar aplicaciones en Kubernetes.
- Conoce **alternativas**: OpenShift, Docker Swarm, Nomad.

Kubernetes es el estándar de la industria. Dominarlo es un gran paso adelante en tu carrera.

Infraestructura como Código (IaC)

Configurar entornos manualmente es lento y propenso a errores.

IaC te permite **describir** toda tu **infraestructura** como archivos de texto.

🔹 Aprende a usar **Terraform**: define servidores, redes, bases de datos, y más, de forma declarativa.
🔹 Descubre **Ansible** para automatizar configuraciones dentro de los servidores.
🔹 Usa **Git** para versionar tu infraestructura.
🔹 Aprende **buenas prácticas**: modularidad, separación por entornos, uso de variables, validación antes de aplicar cambios.

IaC es una de las habilidades más valoradas en cualquier perfil DevOps.

Monitoreo y observabilidad

No basta con desplegar una aplicación. Necesitas saber que está funcionando correctamente.

- Implementa **métricas**, **logs** y **alertas** en tus sistemas.
- **Herramientas** clave: **Prometheus** para métricas, **Grafana** para visualización, ELK stack para logs, Datadog para todo en uno.
- Aprende a crear **dashboards** útiles, detectar cuellos de botella y responder rápidamente a incidentes.
- Introducción a **trazabilidad** distribuida: detectar problemas en entornos de microservicios.

La observabilidad es clave para garantizar estabilidad, performance y una buena experiencia de usuario.

Seguridad en entornos DevOps

La seguridad ya no es una etapa final, se integra en todo el proceso: DevSecOps.

🔹 Usa **herramientas** de **escaneo** automático de vulnerabilidades en imágenes de contenedores.
🔹 Maneja **secretos** de forma segura: variables de entorno cifradas, herramientas como Vault o AWS Secrets Manager.
🔹 **Revisa** y **limita permisos** en cada sistema y herramienta.
🔹 Integra **análisis** de código, **tests** de seguridad y **validaciones** en tus pipelines.
🔹 **Familiarízate** con **normativas** y **estándares** como ISO 27001, SOC2, OWASP.

La seguridad es responsabilidad de todo el equipo, y tú serás una pieza clave en ello.

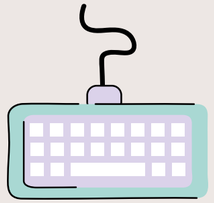
Cierre y próximos pasos

Convertirse en **DevOps Engineer** no es cuestión de memorizar herramientas.

Es adoptar una **mentalidad** de **mejora** continua, **automatización** y **colaboración** constante.

🧩 ¿Por dónde empezar?

- Empieza por los **fundamentos**. Aprende de verdad Linux, Git y redes.
- Practica con **proyectos** reales: un pipeline, un contenedor, una infraestructura reproducible.
- **Documenta** tu aprendizaje. Comparte. Colabora.
- Y sobre todo: **disfruta el proceso**. DevOps es un viaje constante, lleno de retos, aprendizajes... y recompensas.



Gracias

Por Juan Duran

“Coding, Gaming and Leveling Up”