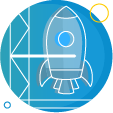
Ruta de aprendizaje

**Administración de Servidores y DevOps**

DevOps es una cultura que combina buenas prácticas y software para mejorar el despliegue y puesta en marcha de aplicaciones de manera rápida y segura.

Implementar DevOps te permitirá automatizar y mejorar los procesos de construcción, pruebas y lanzamiento del software.





**Básico**

Básico



**Curso de Digital Ocean**



**Curso de Fundamentos de AWS Cloud**



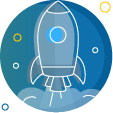
**Fundamentos de Docker**



**Curso Básico de Jenkins**



**Curso de Programación en Bash Shell**



**Intermedio**

Intermedio



**Curso de DevOps con GitLab**



**Curso de Deploy con Now.sh**



**Curso Profesional de DevOps**



**Fundamentos de IBM Cloud**



**Avanzado**

Avanzado



**Curso de Administración de Servidores Linux**



**Curso de Almacenamiento en Azure**



**Curso de SQL en Azure**



**Curso de Web Apps y Logic Apps en Azure**



**Curso de Azure IaaS**



**Curso de Swarm**



**Curso de Kubernetes**



**Curso de Infraestructura Como Código con Terraform**

¿Por qué es importante DevOps?

Primero, vamos a definir **¿qué es DevOps?** no es un cargo o una persona, **es una cultura** que agrupa una serie de prácticas y principios para mejorar y automatizar los procesos entre los equipos de desarrollo e infraestructura (IT) para hacer el lanzamiento de software de una manera rápida, eficiente y segura.

Un concepto importante y uno de los pilares fundamentales en DevOps es la **automatización de procesos**, que incluye los procesos de construcción, pruebas y lanzamiento del software con herramientas como [**Git**](https://platzi.com/cursos/git-github/), [**Jenkins**](https://platzi.com/cursos/jenkins-basico/), **Circle CI**, **Travis**, **Terraform**, entre otras.

Estas herramientas hacen pruebas a nuestra aplicación, antes de salir a producción, para identificar errores y solucionarlos, con esto los usuarios no se verán afectados.

**La implementación de DevOps en una organización trae dos beneficios principales:**

* **Confiabilidad**: la aplicación es probada para verificar su correcta funcionalidad antes de ser llevada al usuario final.
* **Velocidad**: optimización de tiempo y reducción de la intervención humana al probar la aplicación.

**Son impresionantes las ventajas que tiene el utilizar o implementar DevOps en el flujo de trabajo, ahora veamos cómo encaja DevOps dentro de este flujo**:

1. Desarrollo: nuevas características, mejoras, corrección de errores.
2. Se crea el Pull Request.
3. Se compila o construye lo que sea necesario y se ejecutan las pruebas: automatización de procesos con [**GitHub**](https://platzi.com/cursos/git-github/) y herramientas como Jenkins, Travis, CircleCI, entre otras.
4. Se aprueba el Pull Request.
5. Se hace merge con Master.
6. Se compila o construye lo que sea necesario para un entorno de staging o producción y se ejecutan las pruebas
7. Deploy: lo ejecuta automáticamente Jenkins, o la herramienta utilizada, una vez las pruebas pasan.

Con este flujo de trabajo, queda más claro como la implementación de DevOps **ayuda a optimizar la calidad del desarrollo**, debido a que no existe un solo momento donde no se ejecuten pruebas, de hecho, se hacen en varios puntos y esto permite asegurar la reducción de errores en producción. Este es un esquema básico que puede aumentar si se tienen servidores de staging.

**¿Cómo puedo trabajar en DevOps?**

Ya mencionamos que DevOps no es una persona, pero su correcta implementación, seguimiento y mejora de los procesos necesita personas que se encarguen de este flujo para que todo funcione de forma eficiente.

1. Podemos implementar sistemas de integración continua como Jenkins Travis y CircleCI, en algunos casos se deben instalar en un servidor y se configuran, y otros solo se deben configurar ya que son **SaaS** (Software As A Service) en la que solo creas una cuenta, personalizas el servicio y lo conectas con tu repositorio.
2. Siempre se pueden mejorar los procesos de DevOps, optimizar la construcción y los procesos de despliegue, que es otro campo de acción para las personas que se enfocan y especializan en DevOps.
3. Otro campo donde también se puede aplicar los conocimientos en DevOps es estandarizando los entornos de desarrollo, gracias a herramientas como Docker y puedes ver aquí el [**Curso de Docker**](https://platzi.com/cursos/docker/).
4. El campo de acción para alguien que estudie y aprenda sobre DevOps es bastante amplio y puede estar involucrado en varios puntos del desarrollo de software.

**Con el aumento de la complejidad de las aplicaciones se hace cada vez más indispensable la automatización de procesos que permitan asegurar la calidad, estabilidad y fiabilidad de las aplicaciones**, grandes empresas están siempre en busca de este tipo de perfiles que ayuden a mejorar todo el proceso de pruebas y lanzamientos.

No esperes más, este es el mejor momento para aprender y trabajar como DevOps.

**¿Necesito saber programar para trabajar en DevOps?**

No es un requisito programar en un lenguaje de alto nivel como [Java](https://platzi.com/cursos/java-basico/), [PHP](https://platzi.com/cursos/php/), [Ruby](https://platzi.com/cursos/ruby/), sin embargo, será de gran ayuda que conozcas de lenguajes de scripting como [Python](https://platzi.com/cursos/python/), esto te puede ayudar a automatizar tareas.  
Adicionalmente, conocer de [Bash Scripting](https://platzi.com/cursos/bash-shell/) y moverte con facilidad en la [Terminal de Comandos](https://platzi.com/cursos/terminal/).

**¡Qué esperas para comenzar!**

[**¡Prueba la suscripción de Platzi!**](https://platzi.com/precios/)

[**¡No te pierdas nuestros lanzamientos!**](https://platzi.com/agenda/)



**Glosario**

**Quality Assurance (QA)**: Son una serie de procedimientos que permite asegurar la calidad de un producto o pieza de software.

**Pruebas Unitarias (Unit Testing)**: Son pruebas que se realizan a pequeñas partes del código de manera separada para verificar su funcionamiento.

**Integración Continua (Continuos Integration)**: Es un proceso en el cual se puede construir, probar e incluso implementar aplicaciones cuando se realiza un cambio al código fuente.

**Entrega Continua (Continuous Delivery)**: Es un enfoque que permite con la ayuda de la integración continua automatizar pruebas y despliegue de aplicaciones con mínima intervención humana.

**Pull Request**: Es la solicitud para incluir los cambios hechos en un rama de un repositorio de Git a la rama master

