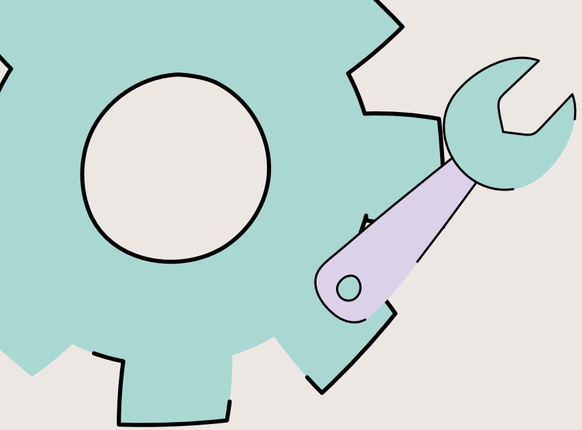
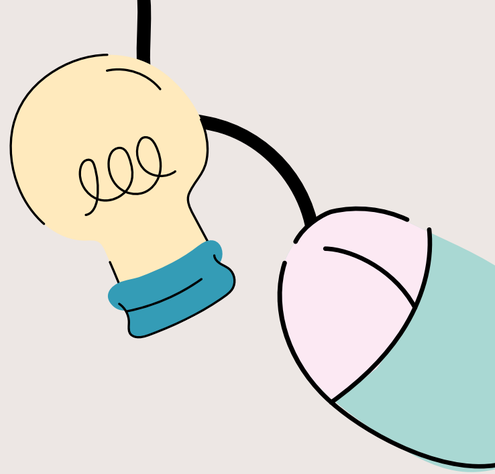
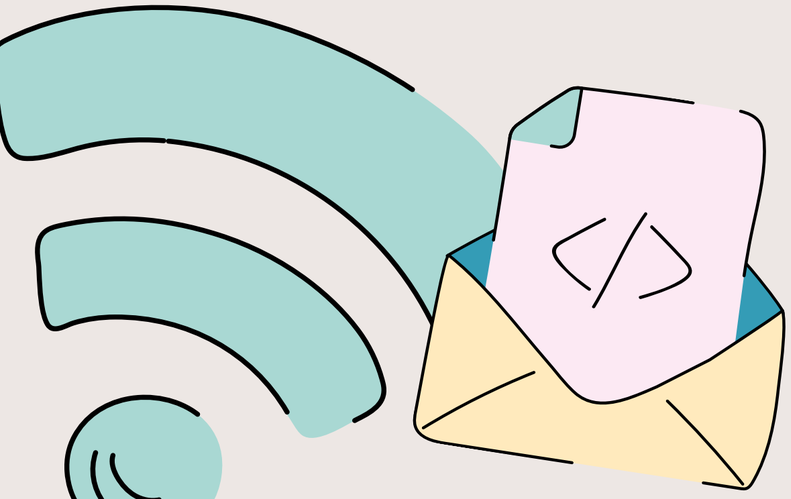
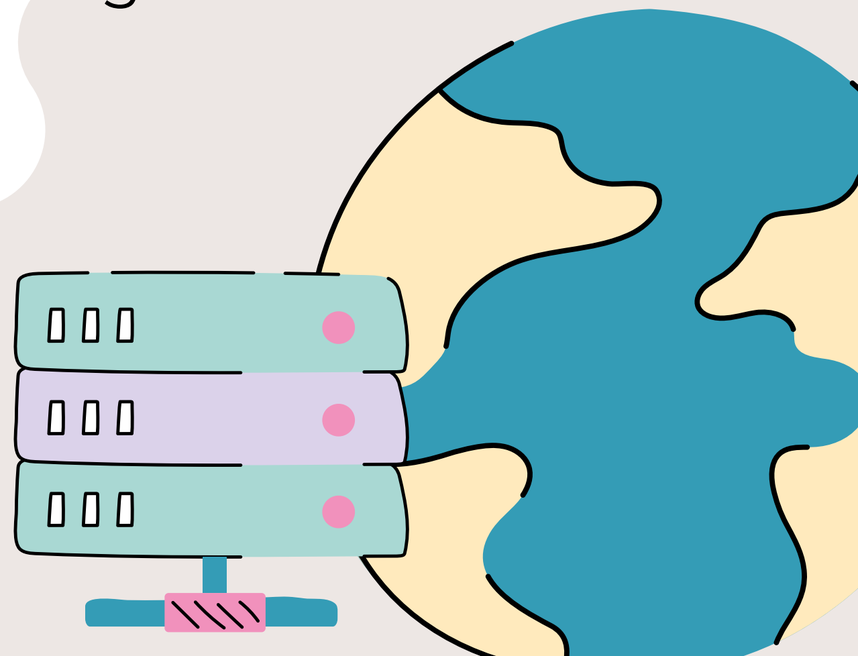




GitHub Actions

Una introducción clara y práctica a la integración continua desde GitHub





Por Juan Duran

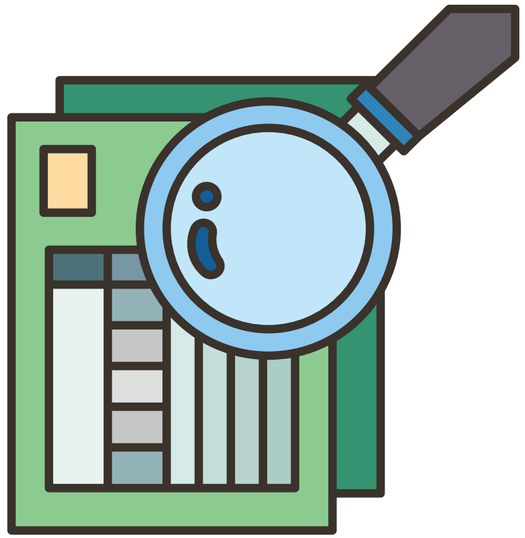
¿Qué es GitHub Actions?

En el desarrollo de software, la **automatización** no es un lujo, es una necesidad. **GitHub Actions** te permite **automatizar** tareas repetitivas, **mejorar** la **calidad** del código y **desplegar** más rápido, todo sin salir de GitHub.

GitHub Actions es una **herramienta** nativa de **GitHub** que permite crear **flujos** de **trabajo** automáticos en respuesta a eventos dentro del repositorio. Esto incluye acciones como hacer push, crear un pull request, publicar una nueva versión o simplemente pasar cierto tiempo.

Imagina que cada vez que subes código nuevo, GitHub automáticamente ejecuta **pruebas**, genera **informes**, **actualiza** tu sitio web y te avisa si algo ha fallado. Todo eso es posible con GitHub Actions.

Es como tener un asistente que trabaja en segundo plano mientras tú sigues desarrollando.

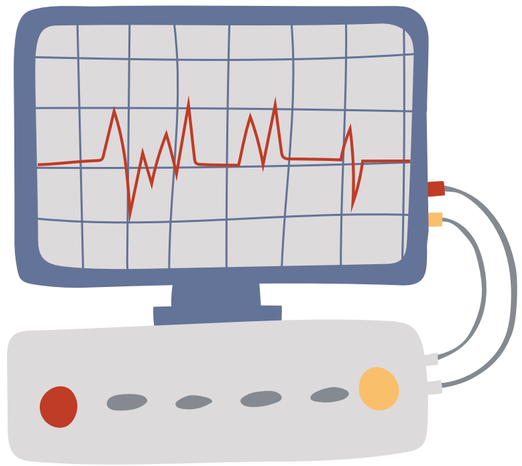


¿Por qué usar GitHub Actions?

La automatización del flujo de trabajo tiene múltiples beneficios:

- **Ahorro de tiempo**: tareas repetitivas como correr pruebas, generar documentación o desplegar código ya no requieren intervención manual.
- **Mayor calidad del código**: al ejecutar pruebas automáticas en cada cambio, puedes detectar errores antes de que lleguen a producción.
- **Mejora de la colaboración**: al integrarse con pull requests, todo el equipo puede ver si un cambio rompe algo antes de aprobarlo.
- **Flujos consistentes y confiables**: el entorno siempre se configura igual, eliminando los errores “funciona en mi máquina”.

GitHub Actions no solo mejora el flujo técnico, también optimiza la experiencia de trabajo en equipo.



¿Qué se puede automatizar?

GitHub Actions es extremadamente flexible. Algunas de las tareas más comunes que puedes automatizar son:

- **Pruebas automáticas**: valida que tu código funciona cada vez que haces un commit.
- **CI/CD (Integración y entrega continua)**: construye y despliega automáticamente tu aplicación.
- **Generación de documentación**: cada vez que actualizas tus archivos, se puede regenerar la documentación automáticamente.
- **Despliegues a servicios en la nube**: como Heroku, AWS, Azure...
- **Mantenimiento de proyectos**: desde borrar ramas antiguas hasta etiquetar nuevas versiones.
- **Notificaciones automáticas**: enviar mensajes por Slack, Discord o email cuando ocurre algo relevante.

Si puedes imaginarlo, probablemente puedas automatizarlo.

¿Cómo funciona?

Todo en **GitHub Actions** se basa en los llamados **workflows**, que son archivos escritos en formato .yml y colocados dentro de la carpeta .github/workflows.

Un **workflow** es básicamente una **receta** que le dice a GitHub qué hacer, cuándo hacerlo y cómo.

Los componentes clave de un workflow son:

- **Evento**: qué lo dispara (por ejemplo, un push).
- **Jobs**: grupos de tareas que se ejecutan en paralelo o en secuencia.
- **Steps**: pasos individuales dentro de un job.
- **Runner**: el servidor que ejecuta los steps.

Esta estructura modular permite que puedas empezar con algo muy simple e irlo haciendo más potente según las necesidades de tu proyecto.

Eventos que activan los workflows

GitHub Actions se basa en **eventos**, es decir, acciones que ocurren dentro del repositorio y que sirven como disparadores. Algunos ejemplos son:

- **push**: cuando haces un commit a una rama.
- **pull_request**: al abrir, actualizar o cerrar una PR.
- **schedule**: para ejecutar tareas de manera periódica (como un cron).
- **release**: cuando publicas una nueva versión del software.
- **issue_comment**: cuando alguien comenta en un issue o pull request.
- **workflow_dispatch**: permite ejecutar manualmente un workflow desde la interfaz de GitHub.

Estos eventos hacen que los **flujos** sean altamente **personalizables**. Puedes ejecutar tareas solo cuando sea necesario y evitar procesos innecesarios.

¿Qué es un runner?

Un **runner** es una **máquina virtual** que ejecuta los **jobs** definidos en un **workflow**.

GitHub ofrece **runners hospedados**, gratuitos para repositorios públicos y con ciertos límites para repositorios privados. Puedes elegir entre máquinas con **Linux**, **Windows** o **macOS** según las necesidades de tu proyecto.

También puedes usar runners auto-hospedados, es decir, servidores propios donde se ejecutan tus workflows. Esto es útil si necesitas más control, recursos específicos o quieres evitar los límites de los runners gratuitos.

Cada job se ejecuta de forma aislada, lo que garantiza seguridad y evita que un proceso interfiera con otro.

Jobs y Steps: la estructura de un workflow

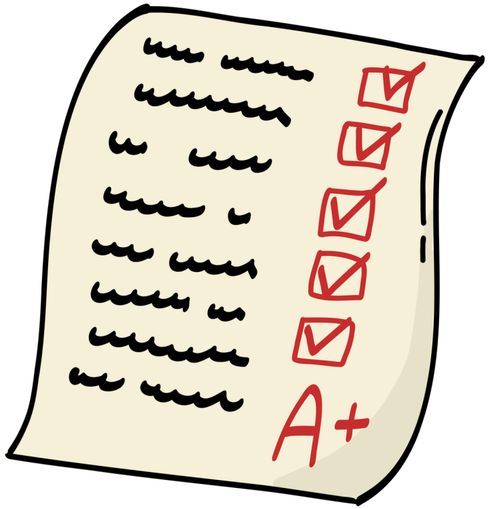
Los **flujos de trabajo** están compuestos por **jobs**, que son conjuntos de tareas que se ejecutan en un runner.

Dentro de cada job, hay múltiples steps (pasos), que pueden ser:

- **Acciones reutilizables** (actions).
- **Comandos definidos** directamente en el archivo.

Por ejemplo, un job podría instalar dependencias, ejecutar pruebas y luego subir resultados. Cada una de esas tareas es un step.

Los jobs pueden ejecutarse en paralelo o depender unos de otros. Esto permite optimizar el tiempo de ejecución y establecer una lógica clara de lo que debe ocurrir y cuándo.



Acciones: componentes reutilizables

Las **acciones** son el corazón de GitHub Actions. Son **unidades reutilizables** de lógica que realizan tareas específicas como:

- Instalar una versión de Node.js
- Subir archivos a un servidor
- Enviar una notificación por Slack

GitHub tiene un **Marketplace** con miles de acciones listas para usar, muchas de ellas creadas por la comunidad y otras por GitHub o proveedores oficiales.

Esto te permite construir **flujos complejos** sin tener que escribir cada paso desde cero.

También puedes crear tus propias acciones, si necesitas algo personalizado o específico para tu equipo o proyecto.

Conclusiones

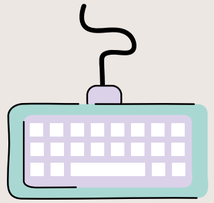
GitHub Actions es una **herramienta** extremadamente **poderosa** para cualquier persona que trabaje con repositorios de código.

No necesitas ser un experto DevOps para empezar: con solo entender los conceptos de eventos, **workflows**, **jobs** y **actions**, ya puedes construir automatizaciones útiles.

Beneficios clave:

- **Automatiza tareas repetitivas**
- **Mejora la calidad del código**
- **Reduce errores humanos**
- **Acelera los ciclos de desarrollo**
- **Mejora la colaboración en equipo**

En resumen, GitHub Actions te permite hacer que tu código no solo esté almacenado, sino también vivo, funcionando para ti las 24 horas del día.



Gracias

Por Juan Duran

“Coding, Gaming and Leveling Up”