TALENTOFUTURO

Docentes:
Roberto Arce
Pablo Cruz

Clase 2: Github Actions

Módulo 8: Fundamentos de Integración Continua

Apoyado por:
CORFO

Clase de hoy

01

CI/CD Conceptos Clave

Integración Continua, Despliegue Continuo y Entrega Continua.
Flujo de CI y Github Actions

02

Modificando un pipeline

Revisar el funcionamiento de pipeline aplicando algunas modificaciones

2A

CI/CD Conceptos Clave

Bloque A

Qué veremos en bloque A

Conceptos Clave

- Definición y diferencias entre Integración Continua, Despliegue Continuo y Entrega Continua.
- Flujo de proceso típico de Integración Continua

Integración Continua

- **Integración Continua (CI)** es una práctica donde los desarrolladores integran su código frecuentemente en un repositorio compartido.
- Cada integración **dispara procesos automáticos** para:
  - Construir el proyecto.
  - Ejecutar pruebas automatizadas.
- Permite **detectar errores de forma temprana**, antes de que lleguen a producción.
- Si las pruebas pasan, el código queda listo para ser lanzado o desplegado.
- CI promueve la colaboración efectiva entre desarrolladores, asegurando que el sistema esté siempre en un estado funcional.
- Un desarrollador normalmente prueba su módulo, pero **la CI verifica el sistema completo**.

Entrega y Despliegue Continuo

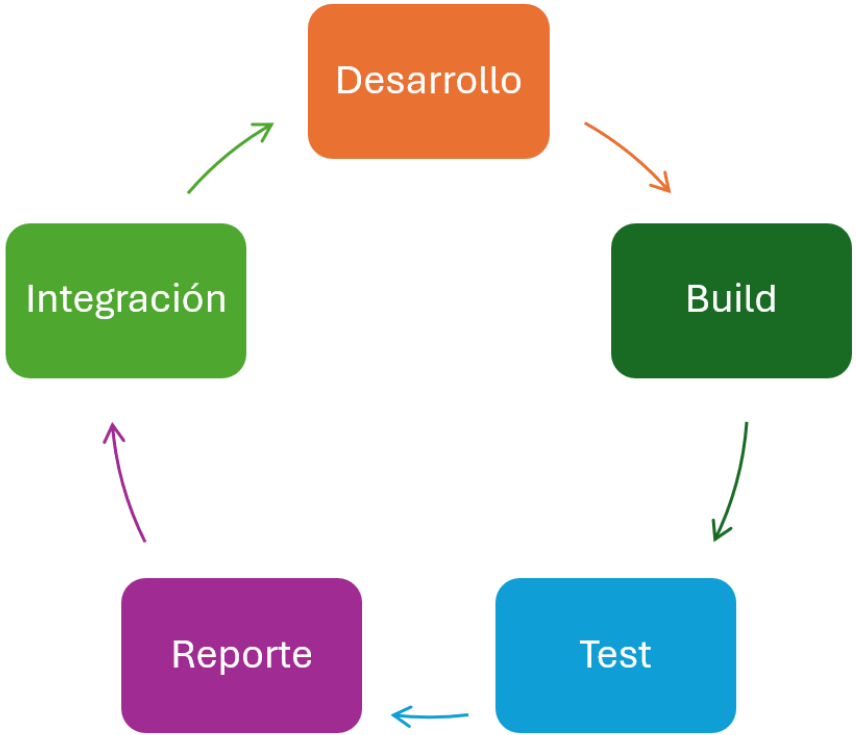
- **Entrega Continua (Continuous Delivery):**
  - Amplía la Integración Continua.
  - Permite desplegar en cualquier momento, de forma confiable.
  - Automatiza todo el proceso hasta antes del despliegue (que es manual).
  - Asegura que cada cambio pase por pruebas y validaciones robustas.

- **Despliegue Continuo (Continuous Deployment):**
  - Automatiza también el despliegue a producción.
  - Cada cambio aprobado se publica inmediatamente a los usuarios.
  - Ideal cuando se tiene alta confianza en los tests automáticos.

Comparación

	CI	Entrega Continua	Despliege Continuo
Compilación Automática	Si	Si	Si
Pruebas Automatizadas	Si	Si	Si
Despliege Automático	No	Manual	Si
Rollback Automático	No	No	Si, necesario
Monitoreo y Alarmas	No	No	Si, necesario
Objetivo	Detectar Errores de Integración	Preparación del Deploy	Automatización completa

Flujo de proceso típico de Integración Continua



Flujo de proceso típico de Integración Continua

1. Desarrollo y Commit de Código
   - **Desarrollo Local:** Se implementa el feature.
   - **Commit de Código**: El desarrollo terminado se sube al repositorio de código

2. Construcción Automática del Proyecto
   - **Lanzamiento**: El CI detecta el cambio en el repositorio e inicia el proceso CI.
   - **Compilación**: El sistema de CI construye el proyecto.
   - **Gestión de Dependencias**: Durante la construcción se resuelven las dependencias del proyecto.

Flujo de proceso típico de Integración Continua

3. Ejecución de Pruebas Automatizadas

- **Pruebas Unitarias**: las pruebas unitarias para verifican que cada componente del código funcione correctamente de manera aislada.
- **Pruebas de Integración**: Se aseguran de que los diferentes módulos o servicios del sistema interactúan correctamente.
- **Pruebas de Regresión**: Se comprueba que los cambios no han roto ninguna funcionalidad existente.

4. Reporte y Feedback

- **Generación de Reportes**: El CI genera reportes sobre la compilación y las pruebas.
- **Notificación al Equipo**: El CI notifica al equipo de desarrollo el resultado

Flujo de proceso típico de Integración Continua

5. Integración en la Rama Principal

- **Merge Automático**: Los cambios exitosos pueden pasar automáticamente a la rama principal
- **Creación de Pull Requests**: En otros escenarios, se puede requerir una revisión de código antes de que los cambios se integren en la rama principal.

Repetición del Proceso e Iteración Continua

Este ciclo se repite cada vez que se realiza un commit o un push de cambios, asegurando que el código se mantenga en un estado continuamente integrado y validado.

GitHub Actions

- **GitHub Actions** es una plataforma de CI/CD integrada directamente en GitHub, lo que permite a los desarrolladores automatizar flujos de trabajo de desarrollo directamente desde sus repositorios.
- Integración Nativa con GitHub
- Servicio Cloud
- Gratis para repositorios públicos
- Escalabilidad, pago por uso

GitHub Actions

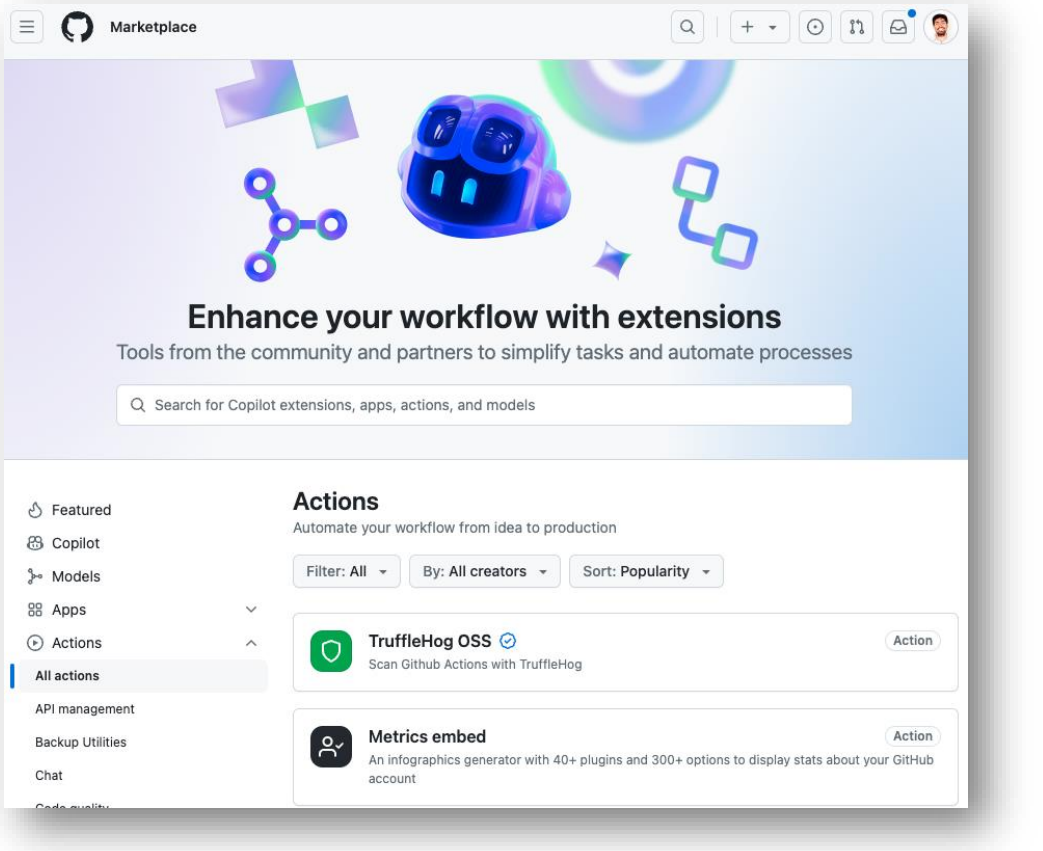
Características

- **Integración nativa con GitHub**: Integración con el ecosistema de GitHub, lo que facilita la configuración de pipelines basados en eventos del repositorio commits/pull-requests/ramas nuevas.

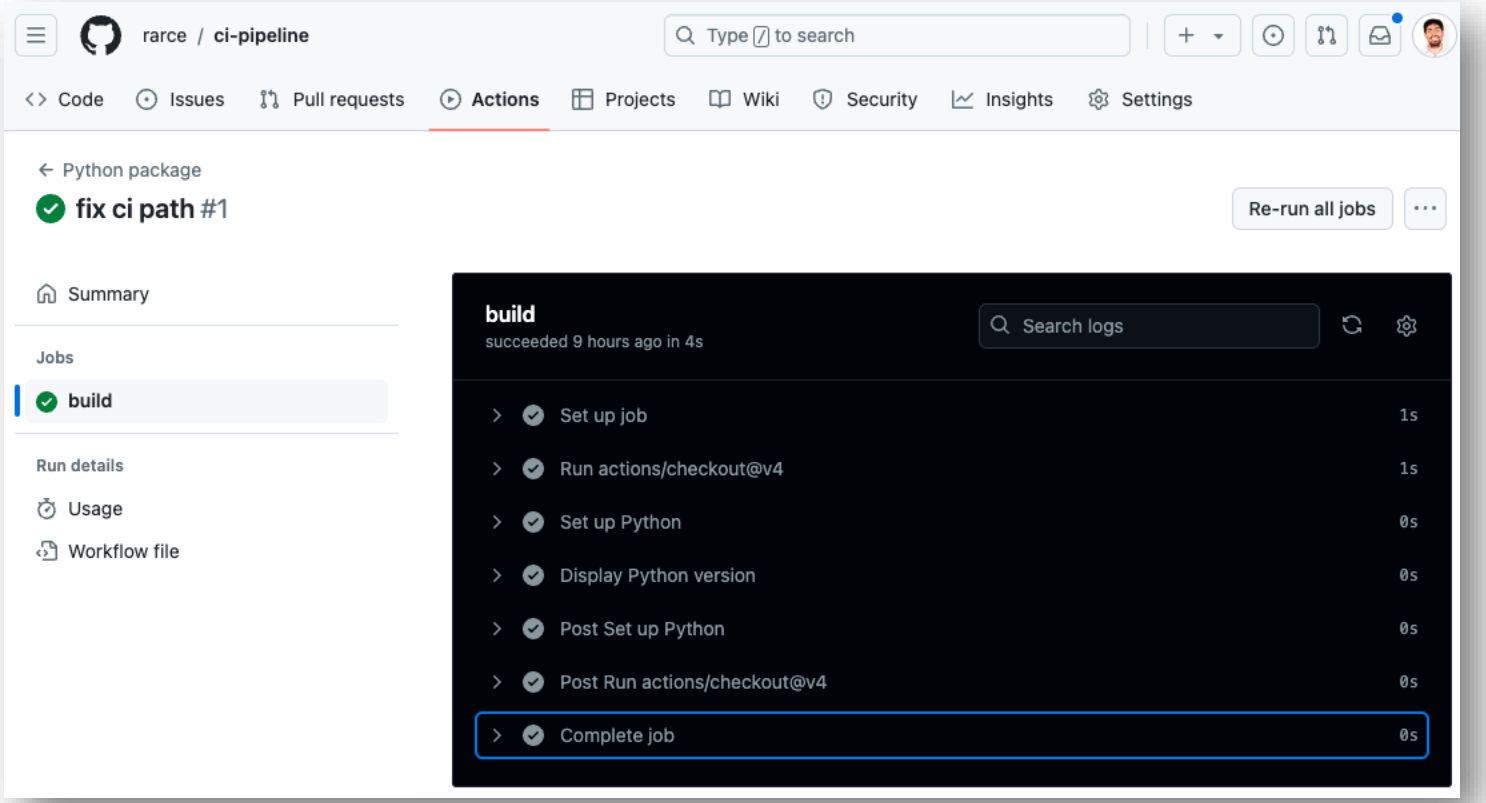
- **Pipelines como código:** Definición mediante archivos YAML, que describen los pasos del pipeline y las condiciones bajo las cuales se ejecutan.

- **Escalabilidad y Flexibilidad:** se ejecuta en runners alojados por GitHub, pero también permite utilizar runners auto-hospedados si se requiere mayor control o escalabilidad.

GitHub Actions



GitHub Actions



GitHub Actions

Ahora vamos a la guía paso a paso publicada en el Drive del curso!
☺

Trabajo grupal – Bloque A

Paralelo 2

G1	G2	G3	G4
Nicolas Mardones	Víctor Meza Herrera	Daniela Méndez Gándara	Claudia Blanco
Manuel Denis	Estefania Manriquez	Álvaro Pérez	Ariel Inostroza
Bryan Castillo	Patricio Vera	Pedro Nahum	Héctor Aguayo
GERALDY SUAREZ	Oscar Torres	Javier Gajardo	Ruben Sanhueza Ramirez
Scarlett Espinoza	Braulio Quiroz	Angela Proboste Neira	Félix González
Ulises Campodónico	Yerko Gallardo	Nicolás Guzmán	Ariel Mora
Carol Leiva	Rodrigo Araya		

G5	G6	G7	G8
Fabian Díaz	Camila Oyarzún	Mayerlyn Rodriguez	Daniela Porto
Natalia Rivera	Stefanya Pulgar	Sebastian Vega	Cristian Chavez Jara
Juan Salinas	Carlos Emilio Azócar Riquelme	Efrain Duarte Campos	Juan Rodrigo Vega
Rodrigo Pastén Cortés	Nicolas Rojas	Bianel Bianchini	Rodolfo Cantillana
Flavio Jara R.	luis.paillan.cnc@gmail.com	Bastián Gamboa Labbé	Abraham Ruiz
Daniel García	Cristóbal Gajardo	Pablo Uribe	Rodrigo Álvarez

Trabajo Grupal Bloque A

1. Modifica el pipeline con prueba fallida y verifica bloqueo de Pull Request

Verificar que un **Pull Request no se pueda aprobar** mientras exista una prueba que falla en GitHub Actions.

1. Cree una regla para proteger la rama main, se deben cumplir las siguientes condiciones
    1. Requiere un PR
    2. Requiere verificación de estado de las pruebas
2. Cree un cambio en una rama que rompa las pruebas, verifique localmente y haga el commit y push de la rama.
3. Abrir un Pull Request hacia la rama principal (main).
4. Observe que:
    1. La pestaña Actions.La indicación "Some checks were not successful" en el PR.
    2. GitHub bloquea el merge si se exige pasar las pruebas.
5. Corregir el test para que pase, hacer commit y push nuevamente.
6. Verificar que el PR ahora se puede aprobar.

Break!

2B

Editando el pipeline

Bloque B

Qué veremos en Bloque B

- Modificando el pipeline generado

Trabajo grupal – Bloque B

Trabajo Grupal Bloque B

2. Cambia el trigger del pipeline y verifica ejecución sólo en push

1. Modificar el archivo ci.yml para que el pipeline se ejecute solo en push a una rama creada en base a “main”, y no en PR.
2. Instrucciones.
    1. Editar el archivo .github/workflows/ci.yml.
    2. Hacer commit y push del cambio a la rama nueva.
    3. Abrir un PR (No se ejecutará ningún pipeline).
3. Verificar en la pestaña Actions que el pipeline se ejecutó con el push a la rama nueva.

Paralelo 2

G1	G2	G3	G4
Nicolas Mardones	Víctor Meza Herrera	Daniela Méndez Gándara	Claudia Blanco
Manuel Denis	Estefania Manriquez	Álvaro Pérez	Ariel Inostroza
Bryan Castillo	Patricio Vera	Pedro Nahum	Héctor Aguayo
GERALDY SUAREZ	Oscar Torres	Javier Gajardo	Ruben Sanhueza Ramirez
Scarlett Espinoza	Braulio Quiroz	Angela Proboste Neira	Félix González
Ulises Campodónico	Yerko Gallardo	Nicolás Guzmán	Ariel Mora
Carol Leiva	Rodrigo Araya		

G5	G6	G7	G8
Fabian Díaz	Camila Oyarzún	Mayerlyn Rodriguez	Daniela Porto
Natalia Rivera	Stefanya Pulgar	Sebastian Vega	Cristian Chavez Jara
Juan Salinas	Carlos Emilio Azócar Riquelme	Efrain Duarte Campos	Juan Rodrigo Vega
Rodrigo Pastén Cortés	Nicolas Rojas	Bianel Bianchini	Rodolfo Cantillana
Flavio Jara R.	luis.paillan.cnc@gmail.com	Bastián Gamboa Labbé	Abraham Ruiz
Daniel García	Cristóbal Gajardo	Pablo Uribe	Rodrigo Álvarez

Referencias

- **Introducción general a GitHub Actions**
  - https://docs.github.com/es/actions/writing-workflows
- **Configurar eventos como push y pull_request**
  - https://docs.github.com/es/actions/writing-workflows/choosing-when-your-workflow-runs/events-that-trigger-workflows
- **Sintaxis de los archivos YAML de workflows**
  - https://docs.github.com/es/actions/writing-workflows/workflow-syntax-for-github-actions
- **Acciones reutilizables (uses:)**
  - https://docs.github.com/es/actions/sharing-automations/creating-actions/about-custom-actions

¿Preguntas?

¡Hemos llegado al final de la clase!