

Control de Versiones con Git y Github

Código: GITH-001

Propuesta de Valor: DESARROLLO - PROGRAMACIÓN - METODOLOGÍAS

Duración: 40 Horas



El curso de control de versiones con git y github le brinda la base sólida y práctica para comprender el sistema de control de versiones de git.

Con el cuál podrá administrar archivos para proyectos grandes, pequeños y puedan mejorar continuamente su producto teniendo un control absoluto de todo el software.

Git es la herramienta número uno en las empresas de desarrollo de software de calidad, en el presente curso aprenderá desde las bases con distintos ejemplos utilizados para que comprenda y pueda aplicarlo de manera profesional en sus proyectos.



AUDIENCIA

- Estudiantes, profesionales y desarrolladores de software que trabajen en equipo y necesiten aprender del sistema de control de versiones más avanzado.



PRE REQUISITOS

- Experiencia básica de programación.



OBJETIVOS

- Comprender el sistema de control de versiones de git.
- Administrar archivos para mantener un control absoluto del software.
- Introducción a la terminal y línea de comandos.
- Trabajando con repositorios remotos: Github.



CERTIFICACIÓN DISPONIBLE

- Certificado emitido por **COGNOS**.



CONTENIDO

1. CONTROL DE VERSIONES

- 1.1. ¿QUÉ ES EL CONTROL DE VERSIONES?
- 1.2. CONTROL DE VERSIONES EN EL USO DIARIO
- 1.3. SISTEMAS DE CONTROL DE VERSIONES

2. INTRODUCCIÓN A GIT

- 2.1. ¿QUÉ ES GIT?
- 2.2. TERMINOLOGÍA DE GIT Y SISTEMA DE CONTROL DE VERSIONES
- 2.3. INSTALANDO GIT Y GITBASH EN WINDOWS/LINUX
- 2.4. EDITORES DE CÓDIGO, ARCHIVOS BINARIOS Y DE TEXTO PLANO
- 2.5. INTRODUCCIÓN A LA TERMINAL Y LÍNEA DE COMANDOS

3. COMANDOS BÁSICOS EN GIT

- 3.1. CREAR UN REPOSITORIO
- 3.2. CLONAR UN REPOSITORIO
- 3.3. DETERMINAR EL ESTADO DEL REPOSITORIO

4. REVISIÓN DEL HISTORIAL DE UN REPOSITORIO

- 4.1. MOSTRANDO LOS COMMITS DE UN REPOSITORIO
- 4.2. CAMBIANDO LA FORMA EN QUE GIT LOG MUESTRA INFORMACIÓN
- 4.3. VIENDO ARCHIVOS MODIFICADOS
- 4.4. VIENDO CAMBIOS DE LOS ARCHIVOS
- 4.5. VIENDO UN COMMIT ESPECÍFICO

5. ADICIONANDO COMMITS AL REPOSITORIO

- 5.1. ¿CUÁL ES EL CICLO BÁSICO DE TRABAJO EN GIT?
- 5.2. ¿QUÉ ES EL ÁREA DE TRABAJO, ESPERA Y REPOSITORIO?
- 5.3. COMANDO GIT ADD
- 5.4. COMANDO GIT COMMIT
- 5.5. MENSAJES DE UN COMMIT
- 5.6. ¿PARA QUÉ SIRVE GIT DIFF?
- 5.7. CREANDO UN ARCHIVO GITIGNORE

6. TAGGING, BRANCHING Y MERGING

- 6.1. ¿QUÉ ES EL TAGGING?
- 6.2. ¿QUÉ ES EL BRANCHING?
- 6.3. CREANDO RAMAS DE MANERA EFECTIVA
- 6.4. ¿QUÉ ES EL MERGING?
- 6.5. CONFLICTOS DE MERGING Y SOLUCIÓN DE CONFLICTOS

7. DESHACIENDO CAMBIOS

- 7.1. MODIFICANDO EL ÚLTIMO COMMIT
- 7.2. REVIRTIENDO UN COMMIT
- 7.3. RESETEANDO COMMITS

8. TRABAJANDO CON REPOSITORIOS REMOTOS: GITHUB

- 8.1. CAMBIOS EN GITHUB: DE MASTER A MAIN
- 8.2. USO DE GITHUB
- 8.3. CÓMO FUNCIONAN LAS LLAVES PÚBLICAS Y PRIVADAS
- 8.4. CONFIGURA TUS LLAVES SSH EN LOCAL
- 8.5. CONEXIÓN A GITHUB CON SSH
- 8.6. TAGS Y VERSIONES EN GIT, GITHUB
- 8.7. MANEJO DE RAMAS EN GITHUB
- 8.8. CONFIGURAR MÚLTIPLES COLABORADORES EN UN REPOSITORIO DE GITHUB

9. FLUJOS DE TRABAJO

- 9.1. HACIENDO MERGE DE RAMAS DE DESARROLLO A MASTER
- 9.2. FLUJO DE TRABAJO CON PULL REQUESTS
- 9.3. UTILIZANDO PULL REQUESTS EN GITHUB
- 9.4. CREANDO UN FORK, CONTRIBUYENDO A UN REPOSITORIO
- 9.5. HACIENDO DEPLOYMENT A UN SERVIDOR
- 9.6. HAZME UN PULL REQUEST
- 9.7. README.MD ES UNA EXCELENTE PRÁCTICA

10. MÚLTIPLES ENTORNOS DE TRABAJO EN GIT

- 10.1. GIT REBASE: REORGANIZANDO EL TRABAJO REALIZADO
- 10.2. GIT STASH: GUARDAR CAMBIOS EN MEMORIA Y RECUPERARLOS DESPUÉS
- 10.3. GIT CLEAN: LIMPIAR TU PROYECTO DE ARCHIVOS NO DESEADOS
- 10.4. GIT CHERRY-PICK: TRAER COMMITS VIEJOS AL HEAD DE UN BRANCH

11. COMANDOS GIT EN CASO DE PROBLEMAS

- 11.1. RECONSTRUIR COMMITS EN GIT CON AMEND
- 11.2. GIT RESET Y REFLOG: ÚSESE EN CASO DE EMERGENCIA
- 11.3. BUSCAR EN ARCHIVOS Y COMMITS DE GIT CON GREP Y LOG

12. SERVICIOS DE GITHUB

- 12.1. ISSUES
- 12.2. WIKIS
- 12.3. GITHUB PAGES
- 12.4. ADMINISTRAR Y ASEGURAR GITHUB
- 12.5. AUTOMATIZACIÓN DE GITHUB
- 12.6. LÍNEA DE COMANDOS DE GITHUB

13. GIT GUI

13.1. CLIENTES GRÁFICOS PARA GIT

13.2. GITHUB DESKTOP GUI

13.3. GITKRAKEN

★ BENEFICIOS

- Al finalizar el curso, el participante estará apto para administrar proyectos en GitHub.