Herramientas de Desarrollo

Semana 01
Introducción al control de versiones - Conceptos y principios básicos

Revisamos el sílabo

- Dentro de este documento encontraremos información sobre el curso, incluyendo:
  - Logro del curso.
  - Cronograma de actividades.
  - Sistema de evaluación.
  - Bibliografía.

SÍLABO
HERRAMIENTAS DE DESARROLLO (100000S61T)
2025 - 1

1. DATOS GENERALES

1.1.	Carrera:	Ingeniería de Sistemas e Informática Ingeniería de Software
1.2.	Créditos:	3
1.3.	Enseñanza de curso:	Presencial
1.4.	Horas semanales:	4

2. FUNDAMENTACION

Este curso proporciona al estudiante las habilidades necesarias para el desarrollo de soluciones de software colaborativas lo cual le permitirá construir soluciones de software de alta complejidad mediante la colaboración de distintos especialistas que buscan resolver la necesidad de cada una de las empresas que así lo requieran.

3. SUMILLA

Este curso es de naturaleza teórico-práctica donde se abordan tópicos sobre distintos controladores de versiones de software especializado y otros sistemas referentes en la industria del desarrollo de software. Asimismo, se abordan las mejores prácticas a tener en cuenta para un control de versiones exitoso.

4. LOGRO GENERAL DE APRENDIZAJE

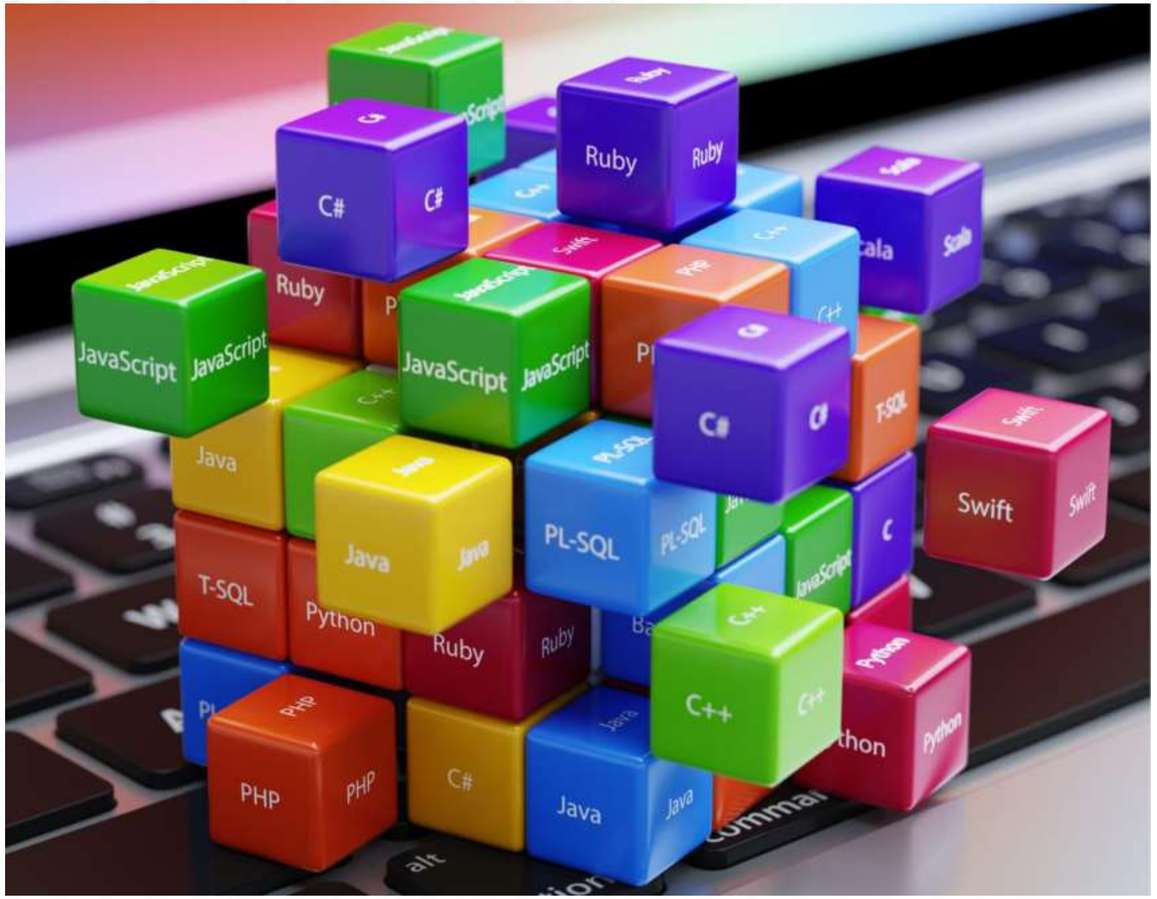
Al finalizar el curso el alumno desarrolla soluciones de software colaborativas utilizando un controlador de versiones de software apropiado.

Logro de aprendizaje

Al finalizar el curso, el estudiante gestiona herramientas de control de versiones, colaborativas y de soporte usadas en el desarrollo de soluciones de software.

Imagen obtenida de: https://www.euroschoolindia.com/wp-content/uploads/2023/08/impact-of-school-leadership.jpg

Inicio



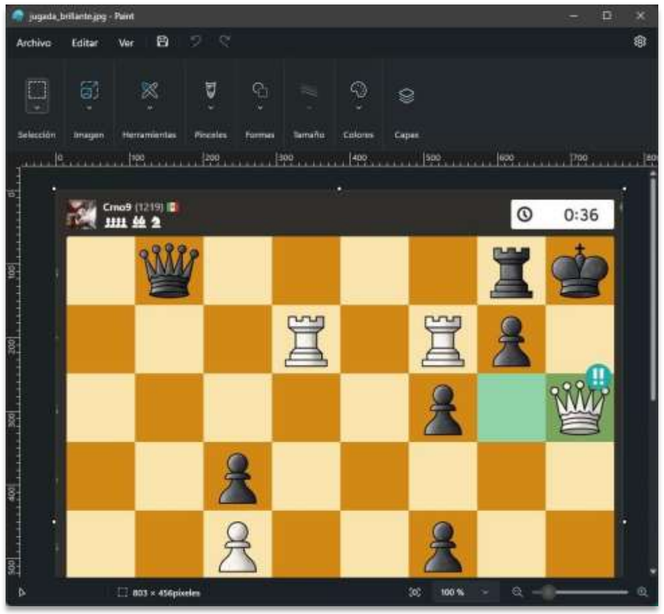
¿Sabías que…?

Existen muchas herramientas de desarrollo en el ámbito de la programación, desde editores de código hasta entornos de desarrollo integrados (IDEs) y herramientas de control de versiones. Entre ellas: **Visual Studio Code (VSCode), Git, IntelliJ IDEA, Docker, Jenkins, entre otros.**

¡A comenzar!

Utilidad

- Menciona tres herramientas de desarrollo que conozca o haya usado?.
- ¿Sabes qué se necesita para en su programación?



Conocer el proceso para la creación de un programa.

Universidad Tecnológica del Perú

contenido

1. Introducción al Control de Versiones
2. ¿Por qué es importante el Control de Versiones?
3. Conceptos y Principios Básicos
4. Principios Básicos del Control de Versiones

Transformación

Universidad Tecnológica del Perú

1 Introducción al Control de Versiones

Introducción al Control de Versiones

El **control de versiones** es un sistema que permite registrar los cambios realizados en los archivos a lo largo del tiempo.

Esta herramienta es fundamental en el desarrollo de software y otros proyectos colaborativos, ya que proporciona un mecanismo para gestionar la evolución del código fuente y facilita la colaboración entre desarrolladores.

Al permitir realizar un seguimiento detallado de todos los cambios realizados, el control de versiones también facilita la reversión a versiones anteriores, la resolución de conflictos y la mejora continua del software.

2 ¿Por qué es importante el Control de Versiones?

¿Por qué es importante el Control de Versiones?

En proyectos de desarrollo de software, varias personas pueden estar trabajando en el mismo conjunto de archivos, y cada persona puede realizar cambios en su parte del código. Sin un sistema de control de versiones, el proceso de gestionar estos cambios podría convertirse en un caos, ya que se perderían modificaciones importantes o se podrían sobrescribir los cambios de otros desarrolladores.

¿Por qué es importante el Control de Versiones?

Algunas razones por las que el control de versiones es esencial en proyectos de software incluyen:

Colaboración: Permite que múltiples desarrolladores trabajen en el mismo proyecto de manera simultánea sin interferir en el trabajo de otros.

Seguridad: El control de versiones permite guardar un historial completo de cambios, lo que proporciona una capa de seguridad al poder restaurar versiones anteriores del código si algo sale mal.

Trazabilidad: Permite conocer quién hizo qué cambio y por qué, lo cual es crucial para la gestión del proyecto y la identificación de problemas en el futuro.

Desarrollo ágil: El control de versiones es una parte esencial en el desarrollo ágil, ya que facilita el trabajo en equipo y la integración continua.

3 Conceptos y Principios Básicos

Conceptos y Principios Básicos

El **control de versiones** es una práctica fundamental en el desarrollo de software, permitiendo a los equipos gestionar los cambios en el código a lo largo del tiempo.

Mediante el uso de un sistema de control de versiones, como **Git**, los desarrolladores pueden llevar un registro de todos los cambios realizados en un proyecto, colaborar en equipo, y manejar diferentes versiones del proyecto de forma eficiente.

4 Principios Básicos del Control de Versiones

Principios Básicos del Control de Versiones

Historial de Cambios: Un sistema de control de versiones mantiene un historial de todos los cambios realizados en el proyecto, permitiendo revertir a versiones anteriores si es necesario.

Ramas (Branches): Las ramas permiten trabajar en diferentes características o correcciones de forma independiente sin afectar el código principal (por ejemplo, la rama "main" o "master").

Fusión (Merge): Una vez que se han realizado cambios en una rama, se pueden fusionar con la rama principal para integrar esos cambios en el proyecto.

Conflictos de fusión (Merge Conflicts): Ocurren cuando dos personas modifican la misma línea de código de manera diferente. El sistema de control de versiones pedirá que se resuelvan manualmente.

….Principios Básicos del Control de Versiones

Revisión de código: Facilita la revisión y comparación de cambios a lo largo del tiempo.

Distribución: En un sistema distribuido como **Git**, cada desarrollador tiene una copia completa del repositorio en su máquina local, lo que hace más seguro y eficiente el trabajo sin necesidad de depender de un servidor central.

Rastro de Autoría: Cada cambio es atribuido a un autor específico, lo que permite conocer quién hizo qué y cuándo.

Práctica

Tipos de Control de Versiones:

- Determinar cuales son y en que se diferencian.
- Cuales son las ventajas y desventajas de cada uno de ellos.
- Cuales son las aplicaciones más usadas en cada uno de ellos.

Cierre

1. ¿Qué es el control de versiones?
2. ¿Cuál es la diferencia entre control de versiones centralizado y distribuido?
3. ¿Qué es una rama en un sistema de control de versiones?
4. ¿Qué es una fusión (merge) en control de versiones?

Imagen obtenida de:
https://dpemfoco.com.br/wp-content/uploads/2019/08/Departamento-de-Pessoal-em-Foco-Checklist-Check-list-Checklists.png

Bibliografía

Hernández Bejarno, Miguel. *Ciclo de vida de desarrollo ágil de software seguro.* Fundación Universitaria Los Libertadores. https://tubiblioteca.utp.edu.pe/cgi-bin/koha/opac-detail.pl? biblionumber=36016

Guillamón Morales, Alicia. (). *Manual desarrollo de elementos software para gestión de sistemas.*
Editorial CEP, S.L. https://tubiblioteca.utp.edu.pe/cgi-bin/koha/opac-detail.pl?biblionumber=34982

Chacon, S., & Straub, B. (2014). *Pro Git* (2nd ed.). Apress.
Enlace: https://git-scm.com/book/es/v2

Poulton, N. (2017). *Docker Deep Dive*. Independently published.
Enlace: https://www.nigelpoulton.com/dvd/

Universidad
Tecnológica
del Perú