WORKSHOP BUENAS PRáCTICAS DE PROGRAMACIóN

SOMOS Y FORMAMOS EXPERTOS EN T.I

Metodología

100% PRACTICO

Duración

10 HRS.

ACERCA DEL CURSO

OBJETIVOS PRINCIPALES

- Aprender a escribir código que sea legible y comprensible
- Conozca los principios, reglas y conceptos clave que le permiten escribir código limpio
- Mantenga el código vivo aumentando la capacidad de mantenimiento con código limpio
- Aprenda con ejemplos prácticos y transformaciones de "código malo a bueno"
- Al finalizar el workshop lograras escribir código limpio y crear mejores aplicaciones en cualquier lenguaje, esto basado en la constante práctica de las funcionalidades enseñadas a lo largo del workshop.

OBJETIVOS SECUNDARIOS

- Aprender los principios de SOLID

PREREQUISITOS

- Se requieren conocimientos básicos de programación (sin importar el lenguaje)
- Se recomienda el workshop Aprende a Programar.
- Contar con conexión a Internet estable.

¿Cuál es mi nivel en programación? clic aquí

¡NUNCA DEJES DE APRENDER!

1.- Introducción

1.1 ¿Qué es el código limpio? 1.2 Puntos clave y cómo escribir código limpio 1.3 Principios, patrones y arquitectura limpia 1.5 Código limpio vs código rápido

2.- Asignación de nombres a variables, clases, funciones y más.

2.1 ¿Por qué son importantes los buenos nombres? 2.2 Elegir buenos nombres 2.3 Convenciones de carcasa y lenguajes de programación 2.4 Nombrar variables y propiedades 2.5 Nombrar funciones y métodos 2.6 Clases de nombres 2.7 Excepciones que debe conocer 2.8 Errores y trampas comunes

3.- Estructura de código, comentarios y formato.

3.1 Comentarios malos 3.2 Buenos comentarios 3.3 ¿De qué se trata realmente el "formato de código"? 3.4 Formato vertica 3.5 Formato: consideraciones específicas del lenguaje 3.6 Formato horizontal

4.- Funciones y métodos.

4.1 Análisis de partes de funciones clave 4.2 ¡Mantenga bajo el número de parámetros! 4.3 Parámetros de función de refactorización: ideas y conceptos 4.4 Cuando un parámetro es el correcto 4.5 Dos parámetros y cuándo refactorizar 4.6 Lidiar con demasiados valores 4.7 Funciones con un número dinámico de parámetros 4.8 Tenga cuidado con los "parámetros de salida" 4.9 Las funciones deben ser pequeñas y hacer una cosa 4.10 Por qué son importantes los "niveles de abstracción" 4.11 ¿Cuándo deberías partir? 4.12 Funciones de división 4.13 No se exceda: evite las extracciones inútiles 4.14 Comprender y evitar efectos secundarios (inesperados) 4.15 ¿Por qué las pruebas unitarias son importantes?

5.- Estructuras de control y errores.

5.1 Presentación de "guards" 5.2 Guards en acción 5.3 Extracción de estructuras de control y preferencia por frases positivas 5.4 Extraer estructuras de control en funciones 5.5 Escribir funciones limpias con estructuras de control 5.6 Inversión de la lógica condicional 5.7 Adopte los errores y el manejo de errores 5.8 Crear más protecciones contra errores 5.9 Extrayendo código de validación 5.10 Uso de funciones de fábrica y polimorfismo 5.11 Trabajar con parámetros predeterminados 5.12 Evite los "números y strings"

6.- Objetos, clases y contenedores.

6.1 Objetos frente a contenedores de datos / estructuras de datos 6.2 Clases y polimorfismo 6.3 Las clases deben ser pequeñas 6.4 Entender la "cohesión" 6.5 Los principios SOLID 6.6 El principio de responsabilidad única (SRP) 6.7 El principio abierto-cerrado (OCP) 6.8 El principio de sustitución de Liskov 6.9 El principio de segregación de interfaces 6.10 El principio de inversión de dependencia

TecGurus
Somos y formamos expertos en T.I.

CERTIFICADO DIGITAL

Obtén una constancia que avala tu preparación, si cumples con la asistencia a tu capacitación y elaboras el proyecto final de cada curso, bootcamp o diplomado.

Registrado por la Secretaria del Trabajo y Previsión Social (México).

¡Te esperamos!

55 5211 6931

+52 55 6186 8835

TecGurusNet

Somos y formamos expertos en T.I. www.tecgurus.net