ESCUELA POLITÉCNICA NACIONAL

FACULTAD DE INGENIERÍA DE SISTEMAS

INGENIERÍA EN SOFTWARE

MANUAL DE ESTÁNDARES DE DESARROLLO DE SOFTWARE PARA LA PROGRAMACIÓN DE UN CONTROLADOR DE ASISTENCIAS DE EMPLEADOS

AUTOR: MATEO SIMBAÑA

DOCENTE: Ing. PATRICIO PACCHA

QUITO, FEBRERO 2024

Contenido

1 Propósito

El propósito fundamental de este proyecto es implementar un sistema de registro de asistencias para empleados mediante un lector de código de barras, asegurando una gestión precisa y segura de las horas laborales. Este sistema no solo registra con exactitud las horas de entrada y salida, sino que también garantiza un control de asistencia confiable, utilizando la autenticación de un código de barras para reforzar la seguridad. Además, su integración facilita la gestión de recursos humanos al optimizar la administración del tiempo de trabajo.

2 Destinatario

Este manual está diseñado para los integrantes del proyecto y para aquellos interesados en la Programación Orientada a Objetos mediante JAVA. Su finalidad es establecer pautas claras sobre la nomenclatura de clases, variables, interfaces, entre otros elementos. Estas directrices aseguran la consistencia en el formato a lo largo del desarrollo del proyecto, incluyendo actualizaciones, correcciones de errores y futuras iteraciones. El propósito fundamental es facilitar la comprensión del código y fomentar la uniformidad, lo que conduce a un desarrollo más eficiente y mantenible.

3 Lineamientos a seguir

3.1 Estilo de codificación

Emplear un estilo de codificación adecuado, incluyendo la convención de nomenclatura, sangrías, uso de espacios en blanco y comentarios.

3.2 Documentación

Implementar comentarios claros y precisos que expliquen el propósito y la funcionalidad del código, principalmente en partes complejas para facilitar su comprensión y mantenimiento.

3.3 Uso de herramientas de gestión de versiones

Ocupar una herramienta de control de versiones, como Git, para facilitar la colaboración en equipo y el seguimiento de cambios en el código.

3.4 Pruebas y control de calidad

Verificar que el programa cumpla con las funciones esperadas y descritas.

Evaluar el desempeño del programa, incluyendo su velocidad, capacidad de manejo de carga y eficiencia.

Evaluar la facilidad de uso y la experiencia del usuario para garantizar que el producto sea intuitivo y accesible.

3.5 Mantenibilidad y escalabilidad

Realizar actualizaciones periódicas del software para incluir nuevas características, mejoras de seguridad y correcciones de errores, manteniendo el sistema actualizado y funcional.

4 Estilo de codificación y convenciones de nomenclatura

4.1 Nomenclatura de variables y métodos

Se utilizará el formato camelCase, en el cual las palabras se escriben juntas, es decir sin espacios, y cada palabra después de la primera comenzarán con una letra en mayúscula. Además, es recomendable el uso de nombres descriptivos para mejorar la comprensión del código. Por otra parte, los métodos deben ser verbos o un conjunto de palabras que comiencen con un verbo y los nombres de las variables no deben comenzar con los caracteres guion bajo o el signo de dólar.

4.2 Nomenclatura de clases e interfaces

Se manejará el formato UpperCamelCase, que guarda similitud con el CamelCase solo que la primera letra de cada palabra será en mayúscula. Para las interfaces se coloca una I en mayúscula al principio del nombre. Es importante recordar que todas las clases e interfaces se escriben en singular.

4.3 Nomenclatura de constantes

Se ocupará el formato UPPERCASE el cual consiste en letras mayúsculas para las constantes y separa las palabras con guiones bajos.

4.4 Nomenclatura de paquetes y archivos

Se nombrarán en minúscula, los archivos de acuerdo con los tipos de datos guardados y los paquetes en relación con su funcionalidad.

4.5 Nomenclatura de scripts

El estándar de codificación para los scripts (Base de Datos) tendrá el siguiente formato:

Para nombrar tablas y campos, seguiremos la convención UpperCamelCase, donde cada nombre de tabla no incluirá abreviaturas.

Ejemplo:

Tabla:

- Sexo
- Persona

Campos:

- IdPersona
- Nombre
- Cédula
- Horario

4.6 Nomenclatura de arquitectura N-TIER

El estándar de codificación para la arquitectura N-TIER será el siguiente:

Data Access:

Las propiedades o atributos llevarán el mismo formato que los campos de una tabla en Base de Datos; de igual manera las clases llevarán el nombre de las tablas, sin embargo utilizarán el sufijo DAO o DTO para distinguirlas.

Ejemplo:

- PersonaDAO

- PersonaDTO

Bussiness Logic:

Para el nombramiento de las clases en esta sección, se empleará el sufijo BL.

Ejemplo:

- PersonaBL

- SexoBL

Bussiness Entities

El estándar para codificación de nuestra ampliación será camelCase para variables globales, locales, paquetes, clases, atributos y métodos.

Por ejemplo:

Librería:

- /lib/Color.java

- /lib/Util.java

- void setColor(){...}

- void getNumber(){...}

Variables:

- int suma;

Atributos:

- public int nombre;

UserInterface

Los componentes de las interfaces gráficas harán uso de una abreviatura, la cual está dada por el tipo de componente.

Por ejemplo:

JPanel:

- pnlRegistroHorario

JBoton:

- btnRegistroHorario

JLabel:

- lblRegistroHorario

5 Bibliotecas externas

java.io.InputStream: Proporciona un flujo de entrada de bytes para leer datos binarios desde fuentes como archivos o redes.

java.util.Scanner: Permite la lectura de datos de diversos tipos, como enteros, flotantes o cadenas, desde fuentes como la entrada del usuario o archivos.

java.io.BufferedWriter: Ofrece eficiencia al escribir caracteres en un flujo de salida, almacenándolos en búfer antes de escribirlos en el archivo subyacente, mejorando el rendimiento.

java.io.FileWriter: Simplifica la escritura de datos en archivos de texto, facilitando la apertura, cierre y escritura de datos en estos archivos.

io.jsonwebtoken.io.IOException: Parte de la implementación de JSON Web Tokens (JWT) en Java, maneja excepciones relacionadas con la manipulación de tokens JWT, como errores de decodificación o validación.

com.fazecast.jSerialComm.SerialPort: Proporciona una interfaz para la comunicación con puertos serie en Java, permitiendo la conexión, configuración y comunicación con dispositivos como Arduino u otros dispositivos electrónicos.

6 Referencias

[1] L. Amparo, «UNAM,» 12 5 2019. [En línea]. Available: http://hp.fciencias.unam.mx/~alg/normas/estilo.html. [Último acceso: 8 1 2024].

[2] D. T. Agency, «Digital Talent Agency,» 17 4 2023. [En línea]. Available: https://dtagency.tech/10-buenas-practicas-para-programadores/. [Último acceso: 8 1 2024].

[3] A. U. Cartagena99, «Academia Cartagena99,» 5 3 2021. [En línea]. Available: https://www.cartagena99.com/recursos/alumnos/apuntes/210302115131-Tema4.pdf. [Último acceso: 8 1 2024].